Feature selection and hierarchical classifier design with applications to human motion recognition

by

Cecille Freeman

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2014

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The performance of a classifier is affected by a number of factors including classifier type, the input features and the desired output. This thesis examines the impact of feature selection and classification problem division on classification accuracy and complexity.

Proper feature selection can reduce classifier size and improve classifier performance by minimizing the impact of noisy, redundant and correlated features. Noisy features can cause false association between the features and the classifier output. Redundant and correlated features increase classifier complexity without adding additional information.

Output selection or classification problem division describes the division of a large classification problem into a set of smaller problems. Problem division can improve accuracy by allocating more resources to more difficult class divisions and enabling the use of more specific feature sets for each sub-problem.

The first part of this thesis presents two methods for creating feature-selected hierarchical classifiers. The feature-selected hierarchical classification method jointly optimizes the features and classification tree-design using genetic algorithms. The multi-modal binary tree (MBT) method performs the class division and feature selection sequentially and tolerates misclassifications in the higher nodes of the tree. This yields a piecewise separation for classes that cannot be fully separated with a single classifier. Experiments show that the accuracy of MBT is comparable to other multi-class extensions, but with lower test time. Furthermore, the accuracy of MBT is significantly higher on multi-modal data sets.

The second part of this thesis focuses on input feature selection measures. A number of filter-based feature subset evaluation measures are evaluated with the goal of assessing their performance with respect to specific classifiers. Although there are many feature selection measures proposed in literature, it is unclear which feature selection measures are appropriate for use with different classifiers. Sixteen common filter-based measures are tested on 20 real and 20 artificial data sets, which are designed to probe for specific feature selection challenges. The strengths and weaknesses of each measure are discussed with respect to the specific feature selection challenges in the artificial data sets, correlation with classifier accuracy and their ability to identify known informative features.

The results indicate that the best filter measure is classifier-specific. K-nearest neighbours classifiers work well with subset-based RELIEF, correlation feature selection or conditional mutual information maximization, whereas Fisher's interclass separability criterion and conditional mutual information maximization work better for support vector machines.

Based on the results of the feature selection experiments, two new filter-based measures are proposed based on conditional mutual information maximization, which performs well

but cannot identify dependent features in a set and does not include a check for correlated features. Both new measures explicitly check for dependent features and the second measure also includes a term to discount correlated features. Both measures correctly identify known informative features in the artificial data sets and correlate well with classifier accuracy.

The final part of this thesis examines the use of feature selection for time-series data by using feature selection to determine important individual time windows or key frames in the series. Time-series feature selection is used with the MBT algorithm to create classification trees for time-series data. The feature selected MBT algorithm is tested on two human motion recognition tasks: full-body human motion recognition from joint angle data and hand gesture recognition from electromyography data. Results indicate that the feature selected MBT is able to achieve high classification accuracy on the time-series data while maintaining a short test time.

Acknowledgements

Fist and foremost I would like to thank my supervisors, Dr. Dana Kulić and Dr. Otman Basir. Dr. Basir, our discussions were always interesting and challenging your encouragement to constantly push forward in the research benefited this thesis greatly. Dr. Kulić, our discussions were always enlightening and informative, exposing me to different problems and research areas that I had not previously considered. Your personal commitment to your students is amazing. You were at every meeting, read every paper and responded to every email. I am so very grateful for all of your support.

Thank you to my committee members for their careful consideration of this thesis and their many insightful comments. Thanks also to the ECE administrative staff for all of your help setting up the defense and getting this thesis completed.

Each week, our lab would meet for a discussion of our ongoing research and other work in our fields. The discussions we had both inside and outside of these meetings were always so interesting. I loved hearing about all your research and am so grateful for your many new perspectives and insights.

I have always found teaching to be extremely enjoyable and the courses I TA'd at the University of Waterloo were no exception. This is in no small part due to the excellent instructors, Dr. Hiren Patel and Dr. Fakhri Karray, and my co-TA Jamil.

Thanks to Dr. Areibi and Dr. Dony for convincing me to do this PhD in the first place.

This project was made possible by the numerous grants and scholarships, including grants from the National Science and Engineering Research Council (NSERC), the Ontario Graduate Scholarship program (OGS), grants from the University of Waterloo and the the facilities of the Shared Hierarchical Academic Research Computing Network (SHARCNET:www.sharcnet.ca) and Compute/Calcul Canada. Thanks also to Thalmic Labs Inc. for allowing us to work with their prototype data and for the numerous helpful meetings.

Not all of the people I need to thank made research contributions to this thesis. My amazing friends kept me sane during this whole crazy endeavor. My party in crime: Morri, Meghan, Alicia and Heather. You guys are amazing for so many reasons. Thanks to Lish, Kev, Alex, James, Pymer, Ashley, Alana, Dom, Kyle and the rest of you crazies for a good night out every once in a while. Thanks to the old grad crew: Anna, who managed to get out, and Zoe, who did not. Thank you Danny, for generally being amazing and for showing me that it is actually possible to complete one of these things.

A big, gigantic, huge thanks to the tipsters, fly by night and all the WODS people. I swear that at some points ultimate was the only thing keeping me from losing it completely.

Lastly, thanks to my amazing family. Lily and Tooty, who were always supportive of everything I did. Mark and Drew and my crazy cousins for making the many trips to Toronto so much easier. To my new in-law family for being so great and accepting. Thanks to my brother Jacob for being a younger, more fun version of me. Thanks to my parents for being amazing and supportive and pretending like I made sense when I was talking about my research. Lastly, thanks to Jon for agreeing to put up with my particular brand of crazy for this long.

Dedication

This thesis is dedicated to Jon, who knows the best way to show someone you love them is to dedicate a gigantic, academic tome to them.

Table of Contents

List of Tables

List of Figures

Acronyms

1-NN 1-nearest neighbour.

ABT Adaptive binary tree.

ACO Ant colony optimization.

ANN Artificial Neural Network.

AR Auto-regressive.

BDT Binary decision tree.

CART Classification and regression tree.

CFS Correlation feature selection.

CMI Conditional mutual information.

CMIM Conditional mutual information maximization.

CPFS Convex principal feature selection.

CV Cross validation.

DAG-SVM Directed acyclic graph support vector machine.

dCMIM Dependency aware conditional mutual information maximization.

dFOU Dependency aware fist order utility.

DOS Dynamic oscillating search.

DWT Discrete wavelet transform.

EMG Electromyography.

FDA Fisher's discriminant analysis.

FIS Fuzzy inference system.

FOU First order utility.

GA Genetic algorithm.

GPLVM Gaussian process latent variable model.

GSBS Generalized sequential backwards search.

GSFS Generalized sequential forward search.

HMM Hidden Markov model.

ICA Independent component analysis.

IFFS Improved forward floating search.

JS Jensen-Shannon.

KL Kullback-Leibler.

KNN / K-NN K-Nearest neighbours.

KPCA Kernel principal component analysis.

LASSO Least-absolute selection and shrinkage operator.

LDA Linear discriminant analysis.

LE Laplacian eigenmaps.

LLE Locally linear embedding.

LOOCV Leave-one-out cross validation.

LOSO / LOSO-CV Leave-one-subject-out cross validation.

LS Laplacian score.

LTSA Local tangent space alignment.

LVF Las Vegas Filter.

MAV Mean absolute value.

MBT Multi-modal binary tree.

MCFS Multi-cluster feature selection.

MDS Multi-dimensional scaling.

MI Mutual information.

MLP Multi-layer perceptron.

mRMR minimal-redundancy maximal-relevancy.

MVU Maximum variance unfolding.

OS Oscillating search.

OVO One vs. one.

OVR One vs. rest.

PCA Principal component analysis.

PPCA Probabilistic principal component analysis.

PSO Particle swarm optimization.

PTA Plus-l-minus-r.

RBF Radial basis function network.

RMS Root mean squared.

SA Simulated annealing.

SBFS Sequential backwards floating search.

SBS Sequential backwards search.

SDE Semi-definite embedding.

SFFS Sequential forward floating search.

SFS Sequential forward search.

SNE Stochastic neighbour embedding.

SOM Self-organizing map.

SPCA Sparse principal component analysis / Supervised principal component analysis.

SS-GPLVM Simple sequence Gaussian process latent variable model.

SSI Simple squared integral.

ST-Isomap Spatio-temporal Isomap.

SU Symmetric uncertainty.

SVM Support Vector Machine.

ZC Zero crossings.

Glossary

K-folds cross validation K-folds cross validation is a method of estimating classifier accuracy. The data set is divided into K approximately equal sized groups or folds. K classifiers are trained, each leaving one fold out of the training set and assessing the trained classifier on the removed points. K-folds cross validation can be used as a wrapper-based feature subset evaluation measure.

1-nearest neighbour The 1-nearest neighbour is a form of the K-nearest neighbours classification algorithm that uses only a single neighbour.

Adaptive binary tree (ABT) The adaptive binary tree is a type of hierarchical classifier based on a one vs. one training [27].

Additive multivariate Additive multivariate feature selection measures consider the benefit of each unselected feature with respect to the selected set. These are different than univariate measure because they consider the value of each feature with respect to the selected features rather than in isolation. Additive multivariate measures are different than grouped multivariate measures, which consider the benefit of a group of features as a whole rather than the additional benefit gained by adding a new feature.

Ant colony optimization (ACO) Ant colony optimization is a type of stochastic optimization technique that can be used for feature selection.

Artificial neural network (ANN) An artificial neural network is an algorithm used for classification or regression.

Auto-regressive coefficients (AR coefficients) An autoregressive model of a time-series signal models the current sample as a weighted sum of past samples in the time-series. The AR coefficients are the weights used for the past samples.

Autoencoder An autoencoder is a type of artificial neural network where the output values are designed to match the input values. Autoencoders can be used for feature extraction by using the hidden node values as features.

Bhattacharyya divergence Bhattacharyya divergence is a divergence measure used to assess the difference between probability distributions. Bhattacharyya divergence can be used as a univariate feature selection measure, where it is used to measure the difference between the distributions of different classes [14].

Binary classifier A binary classifier is a classifier that is only capable of distinguishing between two classes of inputs.

Binary decision tree (BDT) The binary decision tree is a type of hierarchical classifier based on a technique similar to k-means clustering [103].

Classification and regression tree (CART) The classification and regression tree is a decision tree algorithm that can be used for classification or regression.

Classifier Classifiers are algorithms that determine the label or class of a sample based on its characteristic features.

Clustering Clustering techniques aim to divide points into groups or clusters based on their characteristic features. Clustering techniques are unsupervised and hence do not use class labels to help determine the grouping.

Conditional mutual information Conditional mutual information measures the mutual information between two variables given a third variable. The notation for conditional mutual information is $I(A; B|C)$, which indicates the mutual information between A and B given C.

Conditional mutual information maximization (CMIM) Conditional mutual information maximization is an additive multivariate feature subset evaluation measure based on the conditional mutual information between a feature and the class given the selected features [48].

Consistency Consistency is a grouped multivariate feature subset evaluation measure that is based on the number of inconsistent points in a set. Two samples are considered to be inconsistent if they have the same input feature values, but a different class label [37].

Convex principal feature selection (CPFS) Convex principal feature selection is an algorithm for feature selection that treats feature selection as a direct optimization problem [109].

Correlated features A correlated feature is any feature whose value can be predicted to some extent from other features.

Correlation feature selection (CFS) Correlation feature selection is a grouped multivariate feature subset evaluation measure where the goal is to maximize the mutual information between the selected features and the class and minimize the mutual information between the selected features [58].

Dependency aware conditional mutual information (dCMIM) Dependency aware conditional mutual information is a feature subset evaluation measure proposed in this thesis. The dCMIM is based on the CMIM feature measure, but includes an extra step that explicitly looks for dependent features in the set.

Dependency aware first order utility (dFOU) Dependency aware first order utility is a feature subset evaluation measure proposed in this thesis. The dFOU is based on the FOU and CFS feature measures, but includes an extra step that explicitly looks for dependent features in the set.

Dependent features A data set with dependent features includes one or more features that work together to separate the data set, but cannot separate the data set on their own.

Directed acyclic graph support vector machine (DAG-SVM) Directed acyclic graph support vector machine is a multi-class extension for support vector machines that is based on a one vs. one training. A classifier is trained on each pair of classes and these pairwise classifiers are used to successively eliminate classes from consideration. This means that the time required to classify new points is shorter than the one vs. one multi-class extension on which it is based, because fewer classifiers need to be queried. DAG-SVM was proposed in [126].

Discrete wavelet transform (DWT) The discrete wavelet transform is the type of wavelet decomposition used for discrete signals. The discrete wavelet transform divides the signal by passing it through a high and a low pass filters, downsamples each side and successively applies the high and low pass filters to the low frequencies.

Dynamic oscillating search (DOS) Dynamic oscillating search is a type of sequential feature subset selection technique used in feature selection problems. DOS is an extension of OS and unlike its forwards and backwards counterparts, SFFS and SBFS, DOS begins with a set of pre-selected features [144].

Electromyography (EMG) Electromyography uses sensors to detect muscle activation.

Envelope The envelope of a signal gives the general signal shape. It is a low pass filtered version of the rectified signal.

Feature A feature is the input to a classifier. Features describe some aspect of the data and can be quite simple (ex. raw data) or extremely complex (ex. values calculated from raw data).

Feature extraction Feature extraction techniques are used to transform a given set of input features to create a new set of features that have some desired property.

Feature measure Please see "feature subset evaluation measure".

Feature Selected Hierarchical Classifier (FSHC) The feature selected hierarchical classifier is a hierarchical classifier proposed in this thesis that uses a genetic algorithm to jointly perform feature selection and to design the class separation that defines the classifier structure.

Feature selection Feature selection is used to select a subset of good features from a larger set of candidate features. The selected subset forms the set of classifier inputs.

Feature set selection technique The feature set selection technique is a component of feature selection algorithms. The feature set selection technique guides the feature subset search based on the information provided by the feature subset evaluation measure.

Feature subset evaluation measure The feature subset evaluation measure is a component of feature selection algorithms. The feature subset evaluation measure is used to estimate the "goodness" of a particular set of features.

Filter A filter is a type of feature subset evaluation measure, where the feature subset is evaluated using a calculated value instead of direct testing on the desired classifier.

First order utility (FOU) First order utility is an additive multivariate feature subset evaluation measure that combines the conditional mutual information terms used in CMIM and feature-feature mutual information terms used in mRMR [20].

Fisher's discriminant analysis (FDA) Fisher's discriminant analysis is a supervised linear feature extraction technique. It is closely related to linear discriminant analysis (LDA) and the terms are sometimes used interchangeably, though FDA most commonly refers to the feature extraction technique while LDA most commonly refers to the classification technique.

Fisher's interclass separability criterion Fisher's interclass separability criterion is a grouped multivariate feature subset evaluation measure that evaluates feature subsets based on the ratio of within-class to between-class distances [43].

Fuzzy inference system (FIS) A fuzzy inference system is a type of classifier of regression system based on fuzzy logic.

Gaussian process latent variable model (GPLVM) Gaussian process latent variable models can be used as an unsupervised, non-linear feature extraction technique [94].

Generalized sequential backwards search (GSBS) Generalized sequential backwards search is a type of sequential feature subset selection technique used in feature selection problems. It is an extension of SBS. GSBS is less commonly used than its floating counterpart SFBS, becasue GSBS requires parameter settings [80].

Generalized sequential forwards search (GSFS) Generalized sequential forwards search is a type of sequential feature subset selection technique used in feature selection problems. It is an extension of SFS. GSFS is less commonly used than its floating counterpart SFFS, becasue GSFS requires parameter settings [80].

Genetic algorithm (GA) Genetic algorithms are a type of stochastic optimization technique that can be used for feature selection.

Graph Laplacian The graph Laplacian is a representation of the similarity graph where the matrix entries are based on the difference between the similarity graph and the diagonal graph representing all the weight connections to each point.

Grouped multivariate Grouped multivariate feature selection measures consider the benefit of a feature subset as a whole. Grouped multivariate measures are different than additive multivariate measures, which consider the additional benefit gained by adding a new feature rather than the benefit of the entire group.

Hidden Markov model (HMM) A hidden Markov model is a method for modeling time series data. HMMs can be used for classification by training one model per class.

Hierarchical classifier A hierarchical classifier is a multi-class extension where the set of classes is progressively divided into smaller subsets until there is only one class remaining at each node. See also Tree-based classifier.

Hjorth parameters The Hjorth parameters are EMG-specific features. There are three Hjorth parameters: activity, mobility and complexity. The parameters measure the ratio of variances for the signal and its derivatives [64].

Improved forward floating search (IFFS) Improved forward floating search is a type of sequential feature subset selection technique used in feature selection problems. It is an extension of SFFS [116].

Independent component analysis (ICA) Independent component analysis is a method of blind signal separation that can be used as an unsupervised feature extraction technique [33].

Isomap Isomap is an unsupervised, non-linear, neighbourhood based feature extraction technique [149].

Jensen-Shannon divergence (JS) Jensen-Shannon is a divergence measure used to assess the difference between probability distributions. JS can be used as a univariate feature selection measure, where JS is used to measure the difference between the distributions of different classes [97].

K-means clustering K-means clustering is a clustering technique that separates points into K clusters with the smallest grouped distance to the class center.

K-nearest neighbours K-nearest neighbours is a classification algorithm where the class of a point is determined by a vote of the K closest points in the training set [34].

Kernel principal component analysis (KPCA) Kernel principal component analysis is an unsupervised, non-linear feature extraction technique based on PCA [141].

Kullback-Leibler divergence (KL) Kullback-Leibler is a divergence measure used to assess the difference between probability distributions. KL can be used as a univariate feature selection measure, where KL is used to measure the difference between the distributions of different classes [90].

Kurtosis The kurtosis of a signal measures its peakedness.

Laplacian eigenmap (LE) Laplacian eigenmap is an unsupervised, non-linear, neighbourhood based feature extraction technique [13].

Laplacian score (LS) Laplacian score is an unsupervised, neighbourhood based feature subset evaluation measure [61].

Las Vegas filter (LVF) Las Vegas filter is randomized search technique that can be used for feature selection.

Least absolute selection and shrinkage operator (LASSO) The least absolute selection and shrinkage operator can be used to perform embedded feature selection. LASSO adds the minimization of a 1-norm operator to a standard regression or classification optimization problem. The inclusion of this additional term encourages the weights for some inputs to become zero. This sparsity in the weights is a form of embedded feature selection [150].

Leave-one-out cross validation (LOOCV) Leave-one-out cross validation is a method of estimating classifier accuracy. N classifiers are trained,where N is the number of data points. Each classifier leaves one point out from the training set and evaluates the classifier on the untrained point. LOOCV is also called N-folds cross validation. LOOCV can be used as a wrapper-based feature subset evaluation measure.

Leave-one-subject-out cross validation (LOSO) Leave-one-subject-out cross validation is a method of estimating classifier accuracy when the data set contains data from more than one subject. S classifiers are trained,where S is the number of subjects. Each classifier leaves the data from one subject out from the training set and evaluates the classifier on the data from the untrained subject.

Linear discriminant analysis (LDA) Linear discriminant analysis is a binary, linear classification technique. It is closely related to Fisher's discriminant analysis (FDA) and the terms are sometimes used interchangeably, though FDA most commonly refers to the feature extraction technique while LDA most commonly refers to the classification technique.

Local tangent space alignment (LTSA) Local tangent space alignment is an unsupervised, non-linear, neighbourhood-based feature extraction technique [173].

Locally linear embedding (LLE) Locally linear embedding is an unsupervised, non-linear, neighbourhood based feature extraction technique [135].

Maximum variance unfolding (MVU) Maximum variance unfolding is an unsupervised, non-linear, neighbourhood based feature extraction method [161].

Mean absolute value (MAV) The mean absolute value is measured over a finite portion of a time-series signal. The absolute value of each sample is taken, then these values are averaged.

Minimal-redundancy maximal-relevancy (mRMR) Minimal-redundancy maximal-relevancy is an additive multivariate feature subset evaluation measure based on the mutual information between the feature and the class and between the feature and other features in the selected set [122].

Monotonic A monotonic data set is a data set where the classification accuracy will never decrease when a new feature is added to the selected set.

Motion recognition Motion recognition is a type of classification problem where the challenge is to recognize the movement being performed from a set of known movements.

Multi-class classifier A multi-class classifier is a classifier that is capable of distinguishing between more than two classes of inputs.

Multi-class extension A multi-class extension is a method for dividing multi-class classification problems into sets of binary problems, normally for use with binary classifiers.

Multi-cluster feature selection (MCFS) Multi-cluster feature selection is an unsupervised, neighbourhood-graph-based feature subset evaluation measure [23].

Multi-dimensional scaling (MDS) Multi-dimensional scaling is an unsupervised feature extraction technique [18].

Multi-layer perceptron (MLP) A multi-layer perceptron is a type of artificial neural network.

Multi-modal Binary Tree (MBT) The multi-modal binary tree is a hierarchical classifier proposed in this thesis that tolerates misclassification of the training samples by adding additional classifiers to re-classify initially misclassified points. This creates a linear piecewise separation of the data.

Multivariate Multivariate indicates that multiple variables are being considered together. Multivariate feature selection measures consider subsets of features as a group.

Mutual information (MI) Mutual information is a measure of the dependence between a pair of variables. Mutual information can be used as a univariate feature set evaluation measure by measuring the mutual information between the feature and the class label [102].

Naive Bayes Naive bayes is a type of classifier that is based on Bayesian probability.

Neighbourhood graph The neighbourhood graph is a type of similarity graph where neighbouring points are connected by some weight, but non-neighbouring points are unconnected (zero weight).

Non-monotonic A non-monotonic data set is a data set that contains at least one feature that is not monotonic. In this case, adding a feature to the selected set will decrease classification accuracy.

One vs. one One vs. one is a multi-class extension that trains a classifier for each pair of classes. The overall classification of a point is determined by vote.

One vs. rest One vs. rest is a multi-class extension that requires training one classifier per class. Each classifier classifies between points from a single class and points in the remaining classes.

Oscillating search (OS) Oscillating search is a type of sequential feature subset selection technique used in feature selection problems. Unlike its forwards and backwards counterparts, SFFS and SBFS, OS begins with a set of pre-selected features [145].

Particle swarm optimization (PSO) Particle swarm optimization is a type of stochastic optimization technique that can be used for feature selection.

Plus-l-minus-r (PTA) Plus-l-minus-r is a type of sequential feature subset selection technique used in feature selection problems. It is an extension of SFS. PTA is less commonly used than its floating counterparts SFFS because PTA requires parameter settings [146].

Principal component analysis (PCA) Principal component analysis is an unsupervised, linear feature extraction technique [75].

Probabilistic principal component analysis (PPCA) Probabilistic principal component analysis is an unsupervised, linear feature extraction technique based on PCA [153].

Pure noise feature A pure noise feature is a feature that contains no information about the class label. See also "Random feature".

Radial basis function network (RBF) A radial basis function network is a type of artificial neural network.

Random feature A random feature is a feature that contains no information about the class label. See also "Pure noise feature".

Redundant features A redundant feature is a feature that is an exact copy of another feature in the set.

RELIEF / RELIEF-F RELIEF-F is a univariate feature subset evaluation measure that assesses the value of a feature or feature subset (subset-RELIEF) based on the distance to neighbouring points in the same and different classes [79, 82].

Root mean squared (RMS) The root mean squared value is measured over a finite portion of a time-series signal. Each of the values is squared, the squared values are then averaged and the square root of the average of the squared values is taken.

Self-organizing map (SOM) Self-organizing map is a clustering technique that is commonly used for data visualization.

Semi-definite embedding (SDE) Semi-definite embedding is a method for performing maximum variance unfolding (MVU) feature extraction [162].

Sequential backwards floating search (SBFS) Sequential backwards floating search is a sequential feature subset selection technique used in feature selection problems. SBFS is based on SBS, but allows features to be re-added at each step [127].

Sequential backwards search (SBS) Sequential backwards search is a sequential feature subset selection technique used in feature selection problems where features are removed one at a time from a full set [107].

Sequential forwards floating search (SFFS) Sequential forwards floating search is a sequential feature subset selection technique used in feature selection problems. SFFS is based on SFS, but allows features to be removed at each step [127].

Sequential forwards search (SFS) Sequential forwards search is a sequential feature subset selection technique used in feature selection problems where features are added one at at time to the selected set [165].

Similarity graph A similarity graph is a way of representing data in graph form. Each input point represents one node in the graph and the weights between each point describes how similar the points are.

Similarity matrix A similarity matrix is a representation of the similarity graph in matrix form. The matrix entry at ij represents the weight (similarity) between nodes i and j in the similarity graph..

Simple sequence Gaussian process latent variable model (SS-GPLVM) The SS-GPLVM is a type of GPLVM that can be used for motion recognition [16].

Simple squared integral (SSI) The simple square integral is measured over a finite portion of a time-series signal. Each sample is squared and summed to give the SSI.

Simulated annealing (SA) Simulated annealing is a type of stochastic optimization technique that can be used for feature selection.

Skewness The skewness of a signal measures its asymmetry.

Sparse Principal component analysis (Sparse PCA / SPCA) Sparse principal component analysis is an unsupervised, sparse, linear feature extraction technique based on PCA [177].

Spatio-temporal Isomap (ST-Isomap) ST-Isomap is a version of the Isomap feature extraction technique that can be used with temporal data [72].

Spectral clustering Spectral clustering is a clustering technique that is based on the similarity graph [157].

Stochastic neighbour embedding (SNE) Stochastic neighbour embedding is an unsupervised, non-linear, neighbourhood based feature extraction technique [62].

Supervised Supervised techniques use the class labels as well as the input features as a part of an algorithm.

Support vector machine (SVM) A support vector machine is a binary classifier that separates points in different classes using a hyperplane whose location is determined by the largest margin between the classes.

Symmetric uncertainty (SU) Symmetric uncertainty is a univariate mutual-information-based feature subset evaluation measure [169].

Tabu search Tabu search is a type of stochastic optimization technique that can be used for feature selection.

Tree-based classifier A tree-based classifier is a multi-class extension where the set of classes is progressively divided into smaller subsets until there is only one class remaining at each node. See also Hierarchical classifier.

UCI machine learning repository The UCI machine learning repository is a collection of data sets maintained by the University of California, Irvine [8].

Univariate Univariate indicates that only one variable is being considered. Univariate feature selection measures consider each feature in isolation.

Unsupervised Unsupervised techniques do not use the class labels as a part of the algorithm, and base the algorithm output only on the properties of the input features.

Validation set A validation set is a portion of the data set that is withheld from the training set when training a classifier or a regression algorithm. This portion of the data set is then used for testing in order to test the generalization ability of the algorithm.

Wavelet The wavelet decomposition of a signal is similar to short time Fourier analysis, but instead of using equally sized windows for each frequency component, wavelets match the size of the window to the frequency being analyzed. Rather than the sine wave decomposition used in Fourier analysis, wavelet decomposition performs a successive convolution of the signal with scaled and shifted versions of a finite length mother wavelet.

Willson Amplitude The Willson amplitude is measured over a finite portion of a time-series signal. The Willson amplitude counts the number of times the signal value change between samples is larger than a pre-determined threshold.

Wrapper A wrapper is a type of feature subset evaluation measure where the performance of the feature subset is evaluated directly on the classifier.

Zero crossings (ZC) The number of zero crossings is measured over a finite portion of a time-series signal. The ZC measures the number of time the signal value changes signs.

Chapter 1

Introduction

Classification algorithms estimate the class of a sample based on its characteristic features. Classifiers are used in many different domains for tasks ranging from simple taxonomy to complex medical classification and computer vision applications.

The performance of a classifier is affected by a number of different factors including classifier type and parameters, the input features and the desired outputs. This thesis examines the selection of input features and how the selected features and output groupings jointly contribute to the overall performance of the classifier.

Feature selection is an important and difficult problem in machine learning. Good feature selection can improve classifier performance in a number of ways. Selecting a good subset of features not only decreases the computational load, but can also improve accuracy [50]. Feature selection lessens the impact of noisy features, which can cause false association between relatively random features and the classifier output. Feature selection can also help to identify redundant or correlated features, which increase classifier complexity without adding much additional information [57]. In some cases, the features may be informative for differentiating between some classes, but not others. Including these partially informative features may also degrade performance. Feature sets may also contain dependent features, which are not useful by themselves but provide significant information when combined with other features [57].

Feature selection is complicated by the fact that multi-way dependencies and correlations can exist, meaning simply selecting the best individually scored features may not produce the best set. Hence good feature selection requires not just individual or pairwise feature evaluation, but evaluations of entire subsets [57]. Additionally, because of feature

dependence, the optimal set of N features may not include the optimal set of $N-1$ features [127]. This greatly increases the number of evaluations required to properly evaluate a feature set.

Different classifiers tend to have different tolerances for feature sets with redundant and correlated features [57]. This also means that the feature set evaluation criterion needs to be classifier-specific to properly capture the utility of the feature set being investigated.

The most obvious way to tailor a feature set to a specific classifier is to evaluate the feature set directly on the desired classifier. This type of feature evaluation is called a "wrapper" method. While wrappers tend to perform well, since they give direct feedback about the ability of the classifier to use the feature set [81], they are computationally expensive. They can also select features that are too specific to the training set, causing generalization problem. This is a particular problem if the number of samples is small [143].

Filter measures do not evaluate the feature set directly on the classifier, but instead use a calculated value as an estimate of the feature set performance. Filter measures tend to be less computationally expensive than wrappers, but do not directly measure the ability of the selected classifier to use the feature set. The ideal measure of a feature set would be a filter measure that is known to be a good estimate of accuracy for a particular classifier. This measure would likely be different for different classifiers.

The decision about which outputs to use can also significantly affect the classification accuracy. Trying to classify a set with a small number of classes will often be easier than trying to differentiate between larger numbers of possibilities. It is also often easier to classify points when the classes are very different from each other than it is to make fine distinctions between similar categories. Unfortunately, the output classes are most often set a-priori and cannot be changed.

The standard approach to multi-class classification is to use a single classifier with many outputs. However, another option is to split the problem into a set of classification problems that each use only a subset of the classes or samples. Dividing the problem is always necessary when binary classifiers are used to classify multi-class problems. Splitting the problem can increase the overall accuracy multi-class classifiers [137]. Problem division allows the use of a more specific set of features for each set of classes. For some classifiers, better generalization can be achieved by sparse set of inputs [106]. Splitting a multi-class problem also affects the input feature selection process since some features may be more informative about certain class subsets.

The question of how to divide the problem is not trivial and remains an active research area. There are a number of common methods for dividing a multi-class problem for use

with binary classifiers. These include one vs. rest, where a classifier is trained for each class to classify it from the others, and one vs. one, where classifiers are trained on each pair of classes and a voting scheme is used. Unfortunately, it does not appear that any one method is the best solution in all cases [132, 65]. Tree-based classifiers divide the problem hierarchically, with each level splitting the classification task into a progressively smaller set of classes [29, 103]. Trees are intuitively a good choice because they do not require all the trained classifiers to be queried to perform a classification, reducing classification time.

There are three major components to this thesis, each exploring a different aspect of feature selection and output division. The first part of this thesis examines the joint contribution of feature selection and problem division on classification accuracy and computational complexity when using tree-based classifiers. Two different feature-selected tree-based classifiers are proposed and compared to other common multi-class extensions. The first proposed method uses genetic algorithms to jointly optimize the tree design and selected features. The second method performs the class division and feature selection sequentially. It also allows classes to appear at more than one leaf node. This allows it to classify multi-modal or non-linearly separable classes using piecewise classification.

The second part of the thesis examines the feature selection measure problem more closely. There have been a large number of feature evaluation measures proposed in the literature. For a researcher looking to use feature selection as a tool for solving a specific problem, the large number of potential solutions and the lack of direct comparison currently available in the literature can make it difficult to determine which measures are most suitable for a particular application or classifier. Testing a large number of filter measures is impractical in many cases, especially when the feature selection method is not the only parameter to be tuned. The second part of this thesis examines a number of well-known filter measures and compares their performance based on their ability to select informative features and correlation with classifier accuracy. Several different promising filter-based measures are recommended and two new measures are proposed that overcome some of the shortcomings of the tested methods.

The final part of the thesis examines feature selection for time-series data. Adding time-delay inputs to a classifier is essentially the same as adding new features, and their contribution to the overall accuracy and speed needs to be jointly considered when selecting features. The final part of the thesis examines whether feature selection techniques can be used to select individual time windows from within a time series as a way to improve classifier accuracy or reduce classification time. It then applies the selected feature selection measure and tree-based classifier to two time-series human motion recognition problems.

The feature selected hierarchical classifier is well suited for multi-class time-series clas-

sification problems that have noisy, correlated or irrelevant features. The classifier is particularly beneficial for multi-modal or non-linearly separable problems where the training can be performed offline, but the classification of new points must be fast in order to classify incoming data points in an online manner. Human motion recognition is selected as an exemplar multi-class time-series class problem. The human motion recognition problems in this thesis have non-linearly separable classes and have many noisy, correlated and irrelevant features. Motion recognition was therefore considered to be an interesting and challenging domain in which to test the proposed algorithm.

This thesis makes the following contributions

- The development of a feature-selected tree-based classifier that can be used to classify multi-modal and non-linear classes using piecewise separation
- An extensive empirical evaluation of current filter-based feature set evaluation measures with respect to specific classifiers
- The development of a new filter-based feature selection measure that can account for dependent features
- The application of feature selection and the MBT algorithm to two different time-series human motion recognition problems

The remainder of this thesis is organized as follows. Chapter 2 reviews the current literature on feature selection and multi-class extensions including tree-based hierarchical classifiers. It also provides background on the techniques employed in this thesis. Chapter 3 presents and evaluates two feature-selected tree-based classification algorithms. Chapter 4 presents an empirical evaluation of common filter-based feature selection techniques and proposes two new filter-based measures that explicitly look for dependent features. Chapter 5 presents an evaluation of feature selection techniques for time-series data and applies feature selection and the proposed hierarchical classifier to two time-series human motion recognition problems. Chapter 6 presents the conclusions and proposes future directions for this work.

Chapter 2

Previous Work

This chapter overviews the current work on feature extraction and selection and multi-class extensions for binary classifiers including hierarchical classifiers. It also provides background on the techniques employed in this thesis, including genetic algorithms, K-nearest neighbours classification (KNN), support vector machines (SVM) and K-means clustering.

2.1 Notation

A classification problem has a set of inputs $X \in \mathbb{R}^F$ and a set of class labels Y. X is an $F \times N$ matrix, where N is the number of samples and F is the number of variables or "features" in each sample. Y is an N dimensional vector of class labels, where the number of classes is C. Each feature can either be continuous, where it can take any value in its range, or discrete, where it can take only a certain set of values. Continuous features can be discretized by placing the continuous samples into a set number of bins N_b.

Supervised techniques use both the inputs X and the class labels Y. Unsupervised measures use only the inputs X.

2.2 Feature Selection and Feature extraction

Feature selection and feature extraction are often used as pre-processing steps for pattern recognition to improve accuracy and reduce computational complexity. While feature

extraction techniques are often formulated as optimization problems, feature selection techniques are typically formulated as search problems.

Feature extraction applies a transform to the given set of measurements to create a set of derived features:

$$Z = f(X) \tag{2.1}$$

For linear feature extraction techniques, this can be formulated as a matrix transformation.

$$Z = BX \tag{2.2}$$

where B is the $Q \times F$ transformation matrix.

Dimensionality reduction is achieved as part of the transformation, by setting $Q < F$. Alternately, B can be an $F \times F$ matrix, and a feature selection technique can be used to select from the transformed features.

Feature selection aims to find a Q-dimensional subset of features, S_Q, $(Q \leq F)$ that optimizes classifier performance and optionally minimizes the feature set size.

Feature selection and extraction problems are closely related; feature selection can be viewed as a restricted form of linear feature extraction (Equation 2.2), where the matrix B is restricted to be a diagonal binary matrix. However, feature selection is more often formulated as a search problem, where the goal is to find the best subset of features, based on some measure of feature set fitness.

Feature selection techniques are sometimes preferred over feature extraction techniques because they return a set of features that have real-world meaning. This can provide insight into which parts of the system are important. Additionally, feature selection can reduce the cost associated with acquiring features. While some pattern recognition tasks use features that have little associated cost (for example, the individual pixel values in an image), it is also possible to have features that have an associated computational or monetary cost (for example, frequency domain features that have a computational cost or medical tests that have a monetary cost). When features have an associated cost of any type, it can be beneficial to use feature selection to reduce the number of features that initially have to be generated. On the other hand, feature extraction can be used to generate features that are not in the original set, which can be more powerful.

Although feature extraction and feature selection are often presented as being two different, competing methods for dimensionality reduction, they can also be complimentary. Feature extraction techniques can be viewed as a feature generation stage that expands the number of available features, and feature selection can be viewed as the dimensionality

reduction stage that reduces the size of the feature set [44]. Depending on the data set and the classifier, the original feature set may be better for classification [162]. Hence, including the original feature set along with the transformed features may improve accuracy, if the feature selection technique is able to properly account for correlations in the features.

There are a large number of both feature extraction and feature selection techniques. These techniques are discussed and compared in the following sections.

2.3 Feature extraction techniques

Although feature extraction techniques are not the focus of this thesis, feature extraction and feature selection can be used together and share many goals. Ideas and techniques from the field of feature extraction have been incorporated into feature selection algorithms and the fields are closely related. For example, Fisher's discriminant analysis feature extraction is closely related to Fisher's interclass separability criterion in feature selection. Neighbourhood-graph based feature selection techniques such as Laplacian score were motivated by similar neighbourhood-graph based feature extraction techniques such as Laplacian Eigenmaps. For these reasons, a brief survey of interesting feature extraction techniques is presented in this section.

2.3.1 Properties of feature extraction techniques

Common properties of feature extraction techniques are discussed in this section. Table 2.1 shows the properties of each of the feature extraction techniques discussed in this chapter.

Supervised vs. unsupervised

Feature extraction techniques that use the given class labels or function values to determine the extraction function f are called supervised. Unsupervised feature extraction techniques do not use labels or function values, and instead attempt to extract new features based only on the properties of the input data.

Although supervised feature extraction would be expected to work better for supervised pattern recognition tasks, supervised feature extraction techniques do not always outperform unsupervised feature extraction techniques [108].

technique	reference	supervised / unsupervised	global / local	linear / non-linear	manifold / projection	probabilistic / non-probabilistic	parameters
PCA	[75, 141]	unsupervised	global	linear	projective	non-probabilistic	-
Probabilistic PCA	[153]	unsupervised	global	linear	manifold	probabilistic	-
Sparse PCA	[177]	unsupervised	global	linear	projective	non-probabilistic	λ (elastic net tuning)
Sparse PPCA	[55]	unsupervised	global	linear	projective	probabilistic	λ (penalty terms)
Kernel PCA	[139]	unsupervised	global	non-linear	projective	non-probabilistic	K (kernel)
Supervised PCA	[10]	supervised	global	linear	projective	non-probabilistic	-
Supervised PPCA	[170]	supervised	global	linear	projective	probabilistic	-
FDA	[108]	supervised	global	linear	projective	non-probabilistic	-
Classical MDS	[36]	unsupervised	global	linear	projective	non-probabilistic	-
MDS	[36]	unsupervised	global	non-linear	projective	non-probabilistic	D (dissimilarities)
Landmark MDS	[36]	unsupervised	global	non-linear	projective	non-probabilistic	ℓ (# landmarks), D
Isomap	[149]	unsupervised	global	non-linear	manifold	non-probabilistic	k or θ (neighbours)
Landmark Isomap	[40]	unsupervised	global	non-linear	manifold	non-probabilistic	k or θ, ℓ
Extended Isomap	[167]	supervised	global	non-linear	manifold	non-probabilistic	k or θ
LLE	[135]	unsupervised	local	non-linear	manifold	non-probabilistic	k or θ
SDE/MVU	[162, 161]	unsupervised	global	non-linear	projective	non-probabilistic	k or θ
LE	[13]	unsupervised	local	non-linear	manifold	non-probabilistic	k or θ, kernel
SNE	[62]	unsupervised	local	non-linear	projective	probabilistic	σ (tuning)
LTSA	[173]	unsupervised	local	non-linear	manifold	non-probabilistic	k
ICA	[69, 158]	unsupervised	global	non-linear	manifold	non-probabilistic	non-Gaussianity
Supervised ICA	[92]	supervised	global	non-linear	manifold	non-probabilistic	non-Gaussianity
SOM	[100]	unsupervised	global	non-linear	projective	non-probabilistic	# nodes
Spectral clustering	[154, 157]	unsupervised	global	non-linear	projective	non-probabilistic	k or θ
GP-EM	[56]	supervised	global	non-linear	projective	non-probabilistic	various tuning

Table 2.1: Properties of common feature extraction techniques

Linear vs. non-linear

Linear techniques create new features that are linear combinations of the inputs, as in Equation 2.2. Non-linear techniques use an arbitrary function of the original feature set, as in Equation 2.1.

While a non-linear technique can theoretically capture more complex behaviour, non-linear techniques do not necessarily outperform linear techniques [41, 155]. Many non-linear systems make assumptions about the structure of the data; the efficacy of the method will depend on whether or not these assumptions hold for the particular data set.

Global vs. local

Global techniques apply a single transform over the entire space of the data, whereas local techniques apply different versions of the transform to different areas of the input space. Global methods tend to give a more intuitive picture of the overall structure of the data. Local methods can be less computationally expensive [39] because they allow the use of simpler, linear techniques to approximate non-linear behaviour. However, they also

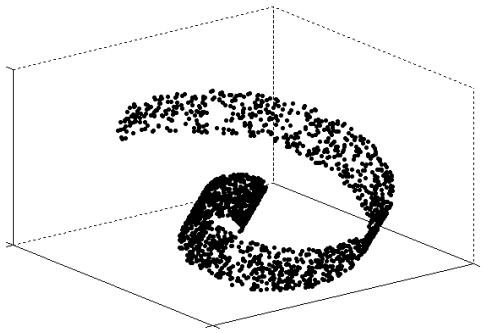


Figure 2.1: Swiss roll data set illustrating the concept behind manifold-based dimensionality reduction techniques

require a way to determine the boundaries of the areas that are being covered by each local approximation function.

Local methods can give better results on a wider variety of manifolds [39], but may have difficulty on data sets that have high intrinsic dimensionality [155].

Projective vs. manifold

Manifold based methods conjecture that the features are transformed versions of data that lies on a hidden, lower dimensional, manifold. For example, Fig. 2.1 shows a data set where the data clearly lies on a 2-D curved manifold but the co-ordinates are given in the 3-D space. For manifold methods dimensionality reduction consists of finding the lower dimensional embedding and undoing the transformation.

Projective methods make no assumptions about the existence of an underlying manifold, and instead seek to find a projection of the data that maximizes some quality or objective function over the data.

Projective and manifold methods can be seen as two sides of a coin. Whereas manifold methods aim to undo a higher dimensional embedding and uncover the "true" latent variables, projective methods treat the observed variables as the "true" variables and the extracted features as a simple projection of this data.

Probabilistic vs. non-probabilistic

Probabilistic techniques assume that there is some type of underlying probability distribution for either the given features, and/or for latent variables in a manifold-based approach. By assuming a probability distribution, probabilistic techniques attempt to determine the optimal set of parameters for a given model of the system. This restricts the problem significantly, but can be problematic if the model is incorrect. Probabilistic estimates can be used directly with certain Bayesian classification techniques [152].

Non-probabilistic techniques do not assume an underlying probability distribution. For some methods, this results in a more general model. In other cases, non-probabilistic techniques simply make different assumptions about what makes a feature informative.

2.3.2 Variance-based feature extraction techniques

Variance-based techniques aim to create a projection that preserves as much of the variance in the data as possible. One of the most widely used feature extraction techniques is principal components analysis (PCA), which is a variance-based technique. Many feature extraction techniques either use PCA as a starting point or have cases where the PCA solution emerges as a result.

Principal Component Analysis (PCA)

Principal components analysis (PCA) [75] projects the original data onto a set of orthogonal bases. The first axis of projection is in the direction of maximum variance of the data. The remaining axes are orthogonal and in the direction of the maximum residual variance. This is illustrated in Figure 2.2.

The covariance matrix gives the pairwise covariance between each two dimensions in the data [74]:

$$cov(X_1, X_2) = E[(X_1 - E(X_1))(X_2 - E(X_2))] \tag{2.3}$$

where E denotes the expected value.

For a zero-mean set of features, the sample covariance matrix is $S = \frac{1}{N}XX^T$ [141]. The eigenvectors of the covariance matrix indicate the directions of the largest variance, or principal components [141]:

$$\lambda U = SU \tag{2.4}$$

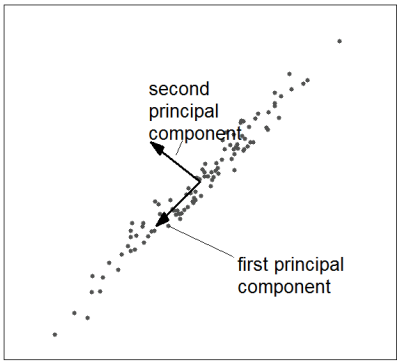


Figure 2.2: Data set illustrating the PCA procedure. The arrows show the direction of the principal components of the data, and the new features are created by projecting each point onto the basis formed by the PCA vectors

where the columns of U give the eigenvectors and λ is a diagonal matrix of the eigenvalues. Dimensionality reduction is achieved by projecting the data onto a lower dimensional basis consisting of the Q eigenvectors of the covariance matrix that correspond to the Q largest eigenvalues.

In addition to maximizing the variance, PCA also gives the solution that minimizes the least-squared error between the projected data and the original data [22, 153]. Hence, PCA can also be expressed as an optimization problem, where the objective is to find a set of weights W such that the squared error between the projection and the original data is minimized

$$\min_{W} ||X - WX||^2 \tag{2.5}$$

Lastly, PCA also gives the solution that maximizes the mutual information between the input and extracted features if the input distribution is Gaussian [22].

Supervised versions of the PCA algorithm have also been proposed [10].

Probabilistic PCA (PPCA)

Probabilistic PCA (PPCA) [153] assumes that the observed variables are linear combinations of underlying latent variables z with offset μ, plus noise σ^2

$$\mathbf{x} = W\mathbf{z} + \mu + \sigma^2 \tag{2.6}$$

The PPCA equations can be solved using a closed-form maximum likelihood solution [153]:

$$W_{ML} = U_Q(\Lambda_Q - \sigma^2 I)^{\frac{1}{2}} R \tag{2.7}$$

where W_{ML} is the maximum likelihood estimate of the weights, U_Q is an $F \times Q$ matrix whose columns are the Q principal eigenvectors of the sample covariance matrix S, Λ_Q is a $Q \times Q$ diagonal matrix of the eigenvalues, σ is the noise in the observed variables, I is the identity matrix, and R is an arbitrary rotation matrix, which can be just I.

If W is set to W_{ML}, σ can be found as [153]:

$$\sigma^2_{ML} = \frac{1}{F-Q} \sum_{i=Q+1}^{F} \lambda_i \tag{2.8}$$

where λ_i is the i^{th} eigenvector and σ^2_{ML} represents the variance that is lost when the observed variables are projected onto the latent variable space.

As the noise in the observed dimension approaches 0 ($\sigma^2 \rightarrow 0$), probabilistic PCA collapses to the standard PCA format.

Sparse PCA (SPCA)

One of the major drawbacks of PCA and is that the projected variables are a linear combination of all the input variables. This makes the projected variables difficult to interpret. Additionally, PCA is sensitive to noisy inputs and can give better results if these are removed.

Sparse PCA [177] automatically selects features and weighs the inputs by applying an elastic net penalty (see Section 2.4.2) to the PCA least-squares optimization problem.

The elastic net is a combination of a least-absolute selection and shrinkage operator (LASSO), which minimizes the $\mathcal{L}_1$ norm of the weights, and a ridge regression operator, which minimizes the $\mathcal{L}_2$ norm of the weights. Due to the nature of the LASSO operator, some of the weight values are set to exactly zero, providing the feature selection. The sparse PCA is a minimization problem of the form [177]:

$$||X - WX||^2 + \gamma||W||^2 + \gamma_1|W| \tag{2.9}$$

Similar sparse versions of other PCA-based algorithms also exist, such as the sparse probabilistic PCA (SPPCA) [55].

Kernel PCA (KPCA)

Kernel PCA (KPCA) [141] is a non-linear generalization of PCA. PCA is performed on a set of points that is transformed to lie in a higher dimensional space, resulting in a non-linear transformation in the original space.

In linear PCA, the principal components of are the eigenvalues of the covariance matrix $S = \frac{1}{N} X X^T$ (see equation 2.4). Because the inputs appear only as a dot product they can be replaced by a kernel matrix K [141]:

$$N\lambda\alpha = K\alpha \tag{2.10}$$

Kernel PCA is closely related to PCA as well as semi-definite embedding (SDE) (see Section 2.3.4), which is a form of kernel PCA where the kernel is learned from the data.

Gaussian process latent variable models (GPLVM)

Gaussian process latent variable models (GPLVM) [93, 94] are closely related to PPCA and KPCA. Like PPCA, GPLVM is a manifold model, and assumes that there is a set of latent variables Z that can be used to explain the observed variables X. However, whereas PPCA assumes that the underlying latent variables are combined linearly, GPLVM allows the observed variables to be an arbitrary Gaussian process. GPLVM can therefore create a non-linear mapping by applying a kernel to the latent variables. This is similar to KPCA, except that in KPCA the algorithm searches for linear projections in the kernel space of the observed variables, whereas in GPLVM the algorithm searches for latent variables that lie in a kernel space [93].

GPLVM is solved iteratively by optimizing the kernel parameters and the latent variables [94].

2.3.3 Distance-based feature extraction techniques

Distance-based techniques aim to preserve or optimize some property of the distance between points.

Fisher's discriminant analysis (FDA)

Fisher's discriminant analysis (FDA) is a supervised technique for feature extraction. It finds features that maximize the distance between points from different classes, and minimize the distance between points in the same class.

FDA finds the vector $\mathbf{w}$ that maximizes [112]

$$J = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}} \tag{2.11}$$

where S_B is the between class scatter matrix

$$S_B = \sum_{c=1}^{C} (\mu_c - \mu)(\mu_c - \mu)^T \tag{2.12}$$

μ_c is the mean of class c and μ is the mean of all the points. The within-class scatter matrix S_w is given as

$$S_w = \sum_{c=1}^{C} \sum_{i=1}^{N_c} (x_{ic} - \mu_c)(x_{ic} - \mu_c)^T \tag{2.13}$$

where N_c is the number of points in class c and x_{ic} is the i^{th} sample in class c.

The solution is given by the generalized eigenproblem

$$S_B \mathbf{w}_i = \lambda S_w \mathbf{w}_i \tag{2.14}$$

where $\mathbf{w}_i$ is the i^{th} largest eigenvector. The extracted features Z are [112]:

$$Z = W^T X \tag{2.15}$$

where W has the Q largest eigenvectors $\mathbf{w}_i$ as its columns.

Multidimensional scaling (MDS)

Multidimensional scaling (MDS) [18] creates a lower dimensional representation of the data such that the distance between each pair of points is as close as possible to the corresponding set of pairwise dissimilarities in the original space. The lower dimensional representation gives the set of extracted features.

In classical MDS the set of dissimilarities is the Euclidean distance between pairs of points. Classical MDS minimizes

$$\min_{Z} \sum_{i=1}^{N} \sum_{j=1}^{N} (||z_i - z_j|| - ||x_i - x_j||)^2 \tag{2.16}$$

where Z are the projected points in the lower dimensional space and X are the points in the higher dimensional space.

Metric MDS uses a more general dissimilarity matrix, and MDS solutions can be found using an eigendecomposition of the data. Define a matrix B such that [40]:

$$B = -HDH \tag{2.17}$$

where D is the $N \times N$ matrix of distances or dissimilarities between points and H is a centering matrix.

The Q dimensional embedding is given as [40]:

$$L = \begin{bmatrix} \sqrt{\lambda_1} u_1^T \\ \sqrt{\lambda_2} u_2^T \\ \cdots \\ \sqrt{\lambda_Q} u_Q^T \end{bmatrix} \tag{2.18}$$

where λ gives the Q largest eigenvalues and U gives the Q largest eigenvectors of the decomposition of B.

MDS can be kernelized [30], where the dissimilarity between two points in the original space is replaced by a kernel matrix.

2.3.4 Neighbourhood graph-based feature extraction techniques

A number of techniques for dimensionality reduction are formulated to preserve some property of the neighbourhood graph. For manifold-based techniques, the assumption is that the distance between neighbouring points is a good approximation of the distance on the underlying manifold [161].

Neighbourhood graph-based techniques can also be projective. These techniques aim to preserve the distances of the neighbours, because neighbouring points are likely informative and are more likely to be in the same cluster, class or function region, but do not attempt to recover an underlying manifold.

Isomap

Isomap [149] approximates the distance between points on the underlying manifold as the shortest path distance over a neighbourhood graph.

Isomap constructs a neighborhood graph where point i is connected to point j by a weight equal to their distance if j is one of the K nearest neighbours of i or if their distance is less than some threshold η. Matrix D describes the neighbourhood graph, where d_{ij} is the distance between the points, if they are connected in the graph, and ∞ otherwise. This matrix is transformed to hold the shortest path distances by replacing each d_{ij} with $\min(d_{ij}, d_{ik} + d_{kj}) \forall k$. MDS is then used to find the lower-dimensional coordinates, which are the extracted features.

Locally linear embedding (LLE)

Locally linear embedding (LLE) [135] finds a set of weights W that can reconstruct each point i from its neighbours j using least squares The weights are constrained such that $w_{ij} = 0$ if j is not a neighbour of i, and $\sum_j w_{ij} = 1$.

After finding an appropriate set of weights W for each point, LLE finds an embedding Z such that [135]:

$$\min_Z \sum_i |\mathbf{z_i} - \sum_j w_{ij} \mathbf{z}_j|^2 \tag{2.19}$$

where $\mathbf{z}_i$ is the low dimensional representation of point i.

Semidefinite embedding (SDE) / Maximum variance unfolding (MVU)

Maximum variance unfolding (MVU) is a manifold-based technique that attempts to "unfold" the underlying manifold by maximizing the total distance between all the points, while maintaining the distances between neighbouring points [161].

Semidefinite embedding (SDE) [161] is a method for performing MVU using semidefinite programming and kernel PCA , where the kernel is learned from the data itself.

Stochastic neighbour embedding (SNE)

Stochastic neighbour embedding (SNE) [62] finds an embedding that matches the probability of each point selecting each other point as its neighbour in the observed space and the lower dimensional projection.

The probability of a point selecting another point as a neighbour is based on their distance. The lower dimensional representation minimizes the sum of the Kullback-Leibler divergences in the higher and lower dimensional space. Hinton and Roweis [62] use a steepest descent algorithm with random jitter, but note that other optimization methods can be used.

Laplacian eigenmaps (LE)

Laplacian eigenmaps [13] (LE) form a lower dimensional representation of the data by solving a generalized eigenvalue problem using the graph Laplacian of the neighbourhood graph.

The algorithm for Laplacian eigenmaps feature extraction is closely related to spectral clustering [157], and can be interpreted as a type of soft clustering of the data [13]. As such, using Laplacian eigenmaps may be appropriate when the data naturally forms clusters. However, the method is also reasonable on data sets where there is no clustering [13].

Local tangent space alignment (LTSA)

Local tangent space alignment (LTSA) [173] aims to preserve the local tangent space Θ_i of each point. The local tangent space of a point is approximated linearly from its nearest neighbours, and these spaces are aligned to approximate the lower dimensional manifold. The goal is to find a mapping from the local tangent space coordinates to the global, low-dimensional coordinates. LTSA finds the linear mapping and the low-dimensional co-ordinates that satisfy

$$\min_{Z,L} \sum_i ||Z_i H_i - L_i \Theta_i||^2 \tag{2.20}$$

where Z_i is the low dimensional embedding of the neighbours of point i, H_i is a centering matrix, $\Theta_i = [\theta_1 \dots \theta_k]$ is the matrix of the PCA transformations of the neighbours of point i, and L_i is a transformation matrix.

2.3.5 Independent Component Analysis (ICA) for feature extraction

Independent component analysis (ICA) [33, 69] begins with the assumption that the observed signals can be modeled as a linear combination of independent, non-Gaussian latent variables.

Central limit theorem states that the sum of independent identically distributed variables tends towards Gaussian. If the latent variables are assumed to be non-Gaussian, then the observed variables will be more Gaussian than the latent variables. Hence, ICA aims to find a set of independent latent variables that are maximally non-Gaussian [69].

There are a number of methods for solving ICA. The simplest method involves defining a measure of non-Gaussianity, such as kurtosis or approximate negentropy, and using an optimization technique such as gradient descent to find the signals that are maximally non-Gaussian. Another alternative is to use optimization techniques to find a solution that minimizes the mutual information between latent variables, or to use a maximum likelihood estimate. One of the more common algorithms is fastICA [68].

ICA solutions are not unique and the components are not ordered. Therefore a second feature selection step is required. Wang and Chang [158] some methods for selecting independent components using ranking based on negentropy or by selecting components that appear multiple times in different runs of the fastICA algorithm. It is also possible to use a different criterion for ranking, or to perform a full feature selection on the selected ICs.

Supervised versions of the ICA algorithm [92] include the class as another dimension in the problem.

2.3.6 Feature extraction techniques based on clustering

It is possible to adapt clustering techniques for feature extraction. Self-organizing maps (SOM) and spectral clustering are sometimes used for feature extraction.

Self-organizing maps (SOM) for feature extraction

SOMs are most commonly used for visualization, where the original higher dimensional data is mapped to a 1- or 2-dimensional plane for visualization [76]. For the purposes of feature extraction, each sample is mapped to its lower dimensional node, which gives the reduced feature set [100].

Spectral clustering for feature extraction

Spectral clustering is another clustering technique that can be used for feature extraction [154]. It is closely related to feature extraction techniques that work with the nearest

neighbour graph. Spectral clustering lowers the dimensionality of the space by using the eigenvectors of the graph Laplacian of the similarity matrix [157]. The dimensionality of the output space can be changed by changing the number of selected eigenvectors.

2.3.7 Autoencoders for feature extraction

Autoencoders are regression neural networks that are trained such that the output of the system is equal to the input. A hidden layer with fewer nodes than the input creates a bottleneck where node values are a good approximation of the inputs. Features can be extracted by feeding inputs to the system and taking the outputs of the hidden nodes [63].

An extension proposed by Rifai et. al [131] adds a regularization term to the autoencoder to encourage robustness to noise and smaller weights in the network.

2.3.8 Feature extraction as a stochastic optimization problem

Guo, Bhattacharya and Kharma [56] present a method for non-linear feature extraction using genetic programming and an expectation maximization algorithm (GP-EM). The genetic program creates a tree with input features at the leaf nodes. Each leaf node is connected to a higher-level node that performs one function on its connected nodes. The algorithm generates one feature at the top level of the tree that is built from a number of the base features in the set.

2.3.9 Domain specific feature extraction

One easily overlooked, but extremely helpful method for feature extraction is to use domain-specific tools to design features specifically for the application. For example, speech recognition applications often employ signal processing techniques to generate a set of features that are more informative than the raw audio signal. Some features that are commonly used for speech recognition include frequencies, zero-crossing points (ZCP) [121], mean frequency [121, 77], bandwidth [121], cepstral and mel-frequency cepstral components (MFCCs), and features generated by linear predictive coding [42]. These features are also used for other audio classification tasks such as music genre classification [140, 47] and audio scene recognition [104, 128, 121].

Image processing applications also use many signal processing-based features, from relatively simple features such as the 2-D Fourier transform or histogram data, to more complex features such as features based on edge, corner or contour detection [120].

This type of domain specific feature can be quite powerful, but the techniques used to generate the features are not generally applicable to other domains. If a researcher is uncertain if a certain extracted feature will be helpful, one solution is to generate a large set of candidate features and then use a feature selection technique to determine which features are helpful.

2.4 Feature selection techniques

Although feature selection can be considered as a specialized case of linear feature extraction, feature selection techniques are rarely framed this way. Instead, feature selection is more often formulated as a search problem, where the goal is to select the set of features that maximizes some measure of feature set "goodness".

Traditional feature selection consists of two components. The feature selection measure is used to measure the fitness of a particular feature set. The feature set selection technique is the algorithm used to select and improve the feature set being tested, based on the information provided by the performance measure. The separation between feature set evaluation and feature set selection is also described by Kohavi and John [81] using the terms feature evaluation and feature selection search. Molina, Belanche and Nebot [113] describe feature selection algorithms in terms of their evaluation measure, search organization and the generation of successors.

Feature selection is a complex problem and there a number of challenges for both the feature set evaluation measure and the search technique. Classifiers have different tolerances for noisy and correlated features. Adding random or noisy features may cause the classification algorithm to generate incorrect associations between these features and the classes. Redundant or correlated features increase complexity without adding new information and can be difficult to identify because multi-way redundancies can exist [54]. Individually high scoring features may not be beneficial if similar features are already included in the set. Additionally, features that are weak individually can become strong when used in combination. Hence, feature set evaluation measures need to be able to properly evaluate the contribution of noisy, correlated, redundant and dependent features.

Dependent and correlated features also complicate the search process. Because correlated features carry similar information, the optimal set of N features may not include the best scoring individual N features. Because dependent features may appear weak when used alone, the optimal set of N features is not guaranteed to include the optimal set of $N-1$ features. The tendency of some simpler feature set selection techniques to always

include the selected $N-1$ features in the set of N features is called the "nesting problem" [127]. A good optimization technique for feature selection therefore needs to consider not just the individual or pairwise scores of the features, but the contributions of entire subsets of features.

2.4.1 Properties

Like feature extraction, feature selection techniques can be described by a number of properties that describe either the feature set evaluation measure or the search technique. These properties closely match the properties that describe the feature extraction techniques (see Section 2.3.1). Feature selection techniques can also be described by two additional categories: univariate vs. multivariate and filter vs. wrapper vs. hybrid vs. embedded. Two of the feature extraction properties do not apply to feature selection techniques: linear vs. non-linear and projective vs. manifold. The linear vs. non-linear distinction does not apply because, as described in equation 2.2, feature selection techniques are always linear. The manifold vs. projective distinction does not apply because even if the samples do lie on a lower dimensional manifold, feature selection alone would not be able to recover it.

Table 2.2 gives the properties of the feature set evaluation measures discussed in this chapter.

Supervised vs. Unsupervised

As described in Section 2.3.1, supervised techniques use the class labels or function values as a way to guide the search. Unsupervised techniques do not use class labels and simply seek to optimize some property of the inputs. Unlike feature extraction, feature selection techniques are most commonly supervised, unless they are being used for clustering. The feature set evaluation measure determines if the technique is supervised or unsupervised.

Global vs. local

In general, feature selection techniques are global, selecting a single set of features for the entire space. Local feature extraction techniques apply different transformations in different areas of the input space. This is uncommon in feature selection, but algorithms applying feature selection to different parts of the classification problem may be thought of as local. Two examples of this type of local feature selection are presented in this thesis

measure	citation	wrapper/ filter/ embedded	supervised / unsupervised	global / local	probabilistic / non-probabilistic	univariate / multivariate
validation set	[57]	wrapper	supervised	global	non-probabilistic	multivariate
cross-validation	[57]	wrapper	supervised	global	non-probabilistic	multivariate
LOOCV	[57]	wrapper	supervised	global	non-probabilistic	multivariate
Fisher's interclass separability	[43]	filter	supervised	global	non-probabilistic	multivariate
RELIEF	[79]	filter	supervised	global	non-probabilistic	univariate
RELIEF-F	[82]	filter	supervised	global	non-probabilistic	univariate
RELIEF extensions	[51, 7]	filter	supervised	global	non-probabilistic	multivariate
probability-based measures	[43]	filter	supervised	global	probabilistic	univariate
Mutual information	[43, 20]	filter	supervised	global	probabilistic	univariate
Symmetric uncertainty	[58, 169]	filter	supervised	global	probabilistic	multivariate
CMIM	[48]	filter	supervised	global	probabilistic	multivariate
mRMR	[122]	filter	supervised	global	probabilistic	multivariate
MFIS	[11]	filter	supervised	global	probabilistic	multivariate
First order utility	[20]	filter	supervised	global	probabilistic	multivariate
FOCUS	[4]	filter	supervised	global	non-probabilistic	multivariate
Dash's consistency measure	[37]	filter	supervised	global	non-probabilistic	multivariate
CFS	[58]	filter	supervised	global	probabilistic	multivariate
Laplacian score	[61]	filter	unsupervised	global	non-probabilistic	multivariate
MCFS	[23]	filter	unsupervised	global	non-probabilistic	multivariate
Binary feature measures	[50]	filter	supervised	global	non-probabilistic	univariate
LASSO	[150]	embedded	unsupervised	global	non-probabilistic	multivariate
Elastic net	[176]	embedded	unsupervised	global	non-probabilistic	multivariate

Table 2.2: Feature set evaluation measures and their properties

in Chapter 3. Feature selected boosted classifiers are also an example of a type of local feature selection [129].

Probabilistic vs. non-probabilistic

Probabilistic feature evaluation measures assume a probability distribution over the feature or features. All the probability-based and information theoretic measures are probabilistic, because they require an estimate of the probability distribution of a feature or features, and joint probability between a feature and the class.

Univariate vs. Multivariate

The feature set evaluation measure can be univariate or multivariate. Univariate feature evaluation measures evaluate each feature individually, while multivariate feature set evaluation measures evaluate sets of features as a group. Univariate feature set evaluation measures can only be used with ranking, and are generally less powerful than multivariate measures, as they are unable to detect feature dependence and correlation. However, univariate measures are also often less computationally expensive and require fewer samples

to measure accurately. A thorough discussion of the benefits and drawbacks of univariate measures and ranked selection is given in Section 2.4.3.

Wrapper vs. filter vs. embedded vs. hybrid

Feature set evaluation measures are most often described by this property [98]. Wrapper techniques evaluate the fitness of a feature set by evaluating the set on the intended classifier. Filter techniques use a calculated measure that does not require testing on the classifier. Embedded techniques incorporate both feature set selection and evaluation, where features are selected during training. Because these use the classifier to evaluate the feature set, they can also be thought of as wrapper techniques. Hybrid refers to any combination of these.

2.4.2 Feature subset evaluation measures

The feature set evaluation measure is a measure of the fitness of a feature set for the desired task.

Wrapper

Wrapper techniques tend to perform better than filter techniques [81], but can select features that are too specific to the training set and do not generalize well to the test set [130, 91, 57]. There are a number of different wrapper methods for estimating the performance of the classifier.

Validation Set A portion of the data is set aside as a validation or testing set, and the classifier is trained on the remaining data. The testing set is then used to estimate the accuracy of the classifier. This technique is relatively fast compared to the other wrapper techniques, as it requires training only one instance of the classifier. However, if the selected validation set is not a good representation of the underlying structure of the data, the results may be skewed. Additionally, if the classifier is sensitive to different initializations or randomization during the training, different classifier trainings can result in different results even on the same validation set. Thus, it is common to average over a number of different runs, or to use a more complex wrapper measures such as k-folds validation [133].

k-folds cross validation k-folds cross validation partitions the data set into k roughly equally sized portions called "folds". It then trains k versions of the classifier, setting aside the k^{th} fold for testing and the remaining folds for training. The accuracy is estimated as the average of the k tests. k-folds cross validation results can also be averaged to give a better estimate of the classifier accuracy [133].

Leave-one-out cross validation (LOOCV) Leave-one-out cross validation (LOOCV) is an extension of the k-folds validation, where the number of folds is equal to the number of samples in the set. Hence, the classifier is trained N times, each time omitting only one sample for testing.

Filter

Filter measures are commonly used when the number of features is large relative to the number of samples, for example, in text processing. In these cases, filter measures are applied to each individual feature, and a ranking method is used (see Section 2.4.3). However, it is possible to use filter measures for data sets with fewer features [44].

Because there are so many different filter measures, it can be difficult to determine which measure is best for use with which data set and classifier, especially because the performance of filter measures is classifier dependent [96]. An overview of common filter measures is given below.

Distance-based measures Distance-based measures focus on selecting a set of features such that points from different classes are separated by a large distance, and points from the same class are close together.

Fisher's interclass separability criterion attempts to directly maximize the average between-class distance, while minimizing the average within-class distance. This is similar to the goal of FDA/LDA feature extraction (see Section 2.3.3), but instead of creating new features, the original feature set is used. The average within class distance is found using the within class scatter matrix, defined as [43]:

$$Q_w = \sum_{c=1}^{C} \sum_{i=1}^{N_c} (x_{ic} - \mu_c)(x_{ic} - \mu_c)^T \tag{2.21}$$

where Q_w is the within class scatter matrix, N_C is the number of points in class c, x_{ic} is the i^{th} point in class c and μ_c is the mean of the points in class c.

The between class distance is measured as [43]:

$$Q_b = \sum_{c=1}^{C} \frac{N_c}{N}(\mu_c - \mu)(\mu_c - \mu)^T \tag{2.22}$$

where μ is the mean of all the points.

Fisher's interclass separability criterion is given as [66]:

$$J = Tr([Q_w + Q_b]^{-1} Q_b) \tag{2.23}$$

where J is a value between zero and one. If the distance between all points is zero, the points are entirely inseparable and the value of J is set to zero.

Another distance-based measure is RELIEF. Unlike Fisher's interclass separability criterion, which looks at the average inter and intra-class distances, RELIEF looks at the distance between a point and its nearest neighbours in the same class and a different class. These are referred to as the nearest hit (same class) and nearest miss (different class). The original paper [79] examines each feature individually, and weights each feature by the average squared distance to the nearest miss minus the average squared distance to the nearest hit. Features with weights over a certain threshold are selected. This is essentially a ranking or best-individual-feature search technique (see Section 2.4.3); random selection is added in order to improve the computational efficiency.

A common extension to the RELIEF algorithm is the RELIEF-F algorithm [82]. In [79], only two classes are used. In RELIEF-F, multiple classes can be used by defining the nearest miss to come from any class that is not in the same class as the selected sample point. RELIEF-F also uses k nearest hits and misses as a way to make the algorithm more robust to noise [82, 83].

RELIEF can easily be extended to measure the fitness of a subset of features as the average squared Euclidean distance between the k nearest hits and misses [51], or by combining the individual differences using any t-norm function [7].

While Fisher's interclass separability criterion attempts to ensure that all the points from the same class are close, and all points from different classes are distant, RELIEF and its extensions attempt to find a set of features such that points are more similar to nearby points in the same class than any of the points in other classes. Hence RELIEF will give a higher score than Fisher's interclass separability criterion on feature sets that have a multi-modal distribution.

Probability-based measures Probability-based measures aim to find features such that the probability density functions (PDF) between different classes are as far apart as possible. Estimating an arbitrary probability density function over a set of features is difficult, and as such these measures are normally simplified by assuming a normal distribution, and estimating the PDFs for each feature individually with ranked selection. However, it is possible to use different distributions if warranted. If there are sufficient samples, researchers may also elect to estimate the PDF over the entire set of features, using a multivariate density estimate [43].

The best features have the largest difference between the PDFs for the different classes, as measured by the Kullback-Liebler (KL) divergence between the classes; but other measures can also be used [43]. The Kullback-Liebler divergence between the probability distributions of feature x on two classes is [90]:

$$KL(x, c_1, c_2) = \sum_{i=1}^{N_b} \ln\left(\frac{P_{(x|c1)}(i)}{P_{(x|c2)}(i)}\right) P_{(x|c1)}(i) \tag{2.24}$$

where $P_{x|c1}$ is the discrete probability of feature x given class c_1, and N_b is the number of bins. This measure is asymmetric. The score for a feature is the summed KL divergence between each pair of classes.

$$J_{KL}(x) = \sum_{i=1}^{C} \sum_{j=1}^{C} KL(x, c_i, c_j) \tag{2.25}$$

The Jensen-Shannon (JS) divergence is another probability measure. Jensen-Shannon is symmetric and defined even if there are bins with zero probability. It is calculated as [97]:

$$JS(x, c_1, c_2) = \frac{KL(x, c_1, M) + KL(x, c_2, M)}{2} \tag{2.26}$$

where M is the average of the two class distributions $M = (P_{(x|c_1)} + P_{(x|c_2)})/2$.

The Bhattacharyya (BH) divergence is defined as [14]:

$$BH(x, c_1, c_2) = -\ln\left(\sum_{i=1}^{N_b} \sqrt{P_{(x|c1)}(i) P_{(x|c1)}(i)}\right) \tag{2.27}$$

Information-Theoretic measures Mutual information measures the dependence between two variables, in this case, the feature and the class. It is also known as the information gain, because the measure describes how much more information is known about one variable, when the other variable is known. It is measured as [102]:

$$I(X_F; Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \tag{2.28}$$

where $p(x)$ is the probability of x and $p(x, y)$ is the joint probability of x and y.

The mutual information can also be defined in terms of the entropies as [102]:

$$\begin{aligned} I(X_F; Y) &= H(Y) - H(Y|X_f) \\ &= H(X_f) + H(Y) - H(X_f, Y) \end{aligned} \tag{2.29}$$

where X_f is feature f of input X, $H(Y)$ is the entropy of Y, $H(Y|X_f)$ is the conditional entropy of Y given X_f and $H(X_f, Y)$ is the joint entropy of X_f and Y. The entropy values are calculated as [35]:

$$H(X) = -\sum_{x \in X} p(x) \log p(x) \tag{2.30}$$

where $p(x)$ is the probability of x. The joint entropy is calculated as [35]:

$$H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y) \tag{2.31}$$

where $p(x, y)$ is the joint probability of x and y. The conditional entropy is calculated as [35]:

$$H(Y|X) = -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)} \tag{2.32}$$

It is possible to use information-theoretic measures with continuous valued features either by discretizing the features, or by using the differential entropy [35]. Mutual information alone tends to favour features that have a larger number of nominal values [83]. Symmetric uncertainty compensates for this by dividing by the summed entropies [169]. Symmetric uncertainty is calculated as [58, 169]:

$$SU = 2 \times \frac{I(X_f; Y)}{H(X_f) + H(Y)} \tag{2.33}$$

Conditional mutual information maximization (CMIM) adds a feature to a set if it caries information about the class that is not already captured by the features in the selected set [48]. Thus, a feature should be selected if the conditional mutual information is high for all the features in the set. The goodness of a feature is measured as [48]:

$$J = \min_{X_i \in S_{m-1}} I(Y; X_m | X_i) \tag{2.34}$$

where X_m is the feature under consideration, and S_{m-1} is the set of $m-1$ features that are already selected.

Other mutual information based approaches, such as minimal-redundancy-maximal-relevance (mRMR) [122] and mutual information feature selection (MIFS) [11], use the pairwise mutual information between features as a way to avoid adding correlated features. The preferred method for feature selection would be to select a set of features S_Q such that the mutual information between the set of features and the class is maximized. However, this is difficult to implement because estimating the mutual information between a set of features and a class requires estimating a multivariate probability density, which is difficult to compute is practice, and requires a large number of points for accuracy. Instead, the authors approximate this condition by finding a set of features such that the information between the feature and the class is maximized (max-dependency), and the information between the features in the set is minimized (min-redundancy). These are limited to the first order terms, which make the computation feasible. However, the measure will not capture any multi-way dependencies.

The mRMR criterion is defined as follows. The most relevant m^{th} feature to add to the set of selected features S_{m-1} is the feature [122]:

$$J = I(X_m; Y) - \frac{1}{m-1} \sum_{i}^{m-1} I(X_i; X_j) \tag{2.35}$$

Brown [20] presents a more general information theoretic measure called first order utility (FOU), and shows that other information theoretic measures can be written as specific versions of the general measure. Like Peng, Long and Ding [122], Brown starts with the premise that the best feature set will maximize the mutual information between the feature set and the class $\max_S I(S, Y)$. This joint probability distribution can be expanded by summing over all the possible subsets of S. Brown then approximates the mutual information term by taking only the pairwise and conditional pairwise (first order) sets as [20]:

$$J = I(X_m; Y) - \sum_{i=1}^{m-1} [I(X_m; X_i) - I(X_m; X_i | Y)] \tag{2.36}$$

This equation can then be parameterized as [20]:

$$J = I(X_m; Y) - \beta \sum_{i=1}^{m-1} I(X_m; X_i) + \gamma \sum_{i=1}^{m-1} I(X_m; X_i|Y) \tag{2.37}$$

A number of common information-theoretic measures can be modeled as specific versions of this measure by varying the parameter values. For example, mRMR uses γ value of 0 and sets β to $1/(m-1)$. CMIM [48] can also be re-written in this form, as can a number of other information-theoretic criteria. However, Brown also shows that different data sets work well with different parameter values, and there is currently no theoretically sound way to select the best parameters.

Consistency-based measures A set of features is considered to be inconsistent if there are two or more examples that have the same values for all features, but have different class labels. The idea behind consistency-based feature selection measures is to find the minimum set of features that is consistent. The best known consistency-based feature selection technique is FOCUS [4]. Starting with single feature sets and working up to larger sized feature sets, FOCUS looks for inconsistencies, and returns the first set where the number of inconsistent examples is below a pre-defined bound. The number of inconsistent examples can also be used as a measure of the feature set fitness [37]. One major drawback to this technique is that it will only work when the features are discrete. With continuous features, it is unlikely that any two points will ever match perfectly, regardless of whether they are in the same class. Hence, continuous features must be discretized.

Correlation-based measures The intuition behind correlation-based measures is that it is desirable for features to have a high correlation with the class, and a low correlation with each other. Selecting features that have a high correlation with the class will remove noisy features, and selecting features that have a low correlation with each other will ensure that redundant features are not selected [58]. The correlation-based feature set selection (CFS) measure is defined as [58]:

$$CFS = \frac{M\bar{r_{cf}}}{\sqrt{M + M(M-1)\bar{r_{ff}}}} \tag{2.38}$$

where M is the number of selected features in the subset being measured, $\bar{r_{cf}}$ is the average correlation between the each feature in the subset and the class, and $\bar{r_{ff}}$ is the average feature-feature correlation, where correlation is measured using the symmetric uncertainty.

Neighbourhood graph estimators Many feature extraction techniques aim to preserve some feature of the neighbourhood graph as a method of preserving the underlying structure of the data (see Section 2.3.4). This is becoming more common in feature selection also. For example, the "Laplacian score" selects the set of features that best preserves the local neighbourhood of a point [61]. The "Laplacian score" for each feature is [61]:

$$J_f = \frac{\tilde{X_f}^T L \tilde{X_f}}{\tilde{X_f}^T D \tilde{X_f}} \tag{2.39}$$

where D is the diagonal degree matrix where $d_i = \sum_{n=1}^{N} s_{in}$, S is the similarity matrix, or matrix of weights between each point, set such that $S_{ij} = 0$ if points i and j are not neighbours, and $S_{ij} = e^{-\frac{1}{t}||x_i - x_j||^2}$ and t is a constant. L is the graph Laplacian $L = D - S$ and $\tilde{X_f}$ is calculated as [61]:

$$\tilde{X_f} = X_f - \frac{X_f D \mathbf{1}}{\mathbf{1}^T D \mathbf{1}} \mathbf{1} \tag{2.40}$$

where $\mathbf{1}$ is a vector of all 1s.

This is an unsupervised feature set measure, which was at one point fairly unusual in the literature, but is now becoming more common, particularly for clustering algorithms. The score is calculated on a per-feature basis, but the measure itself is not completely univariate, as the graph Laplacian includes information about other features.

Another unsupervised neighbourhood graph-based feature selection method is multi-cluster feature selection (MCFS) [23]. This technique selects features with the lowest squared reconstruction of the underlying flat embedding. Starting with the graph Laplacian of the neighbourhood graph L, a flat embedding of the data points Y is found such that $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_K]$, where K is the expected number of clusters in the data set and the values $\mathbf{y}_k$ are the solutions to the eigenvalue problem

$$L\mathbf{y} = \lambda D\mathbf{y} \tag{2.41}$$

For each eigenvector, the relative importance of each feature can be estimated as [23]:

$$\min_{\mathbf{a}_k} ||\mathbf{y}_k - X^T \mathbf{a}_k||^2 + \beta |\mathbf{a}_k| \tag{2.42}$$

where $\mathbf{a}_k$ is an F dimensional vector and the term $\beta|\mathbf{a}_k|$ is an L_1 regularization term (or LASSO term) to produce a sparse vector (please see Section 2.4.2 for details on LASSO penalty terms).

After solving K problems, there will be K vectors a_k. The score for each feature f is then found as

$$J(f) = \max_K |a_{k,f}| \tag{2.43}$$

Hybrid

One of the most common hybrid measures is the 1-D classifier. In these measures, one classifier is trained on each feature, and the accuracy results are used to rank the features. For example, [96] uses a 1-D SVM and ranked selection for a text classification task. Similarly, Yang et al. [166] train a neural network on the full feature set and then assess each filter based on how much the accuracy decreases when each filter is removed, while Verikas and Bacauskiene [156] assess features based on the sensitivity of the neural network when the feature is removed.

Cantu-Paz [24] presents a hybrid genetic algorithm. A filter measure is used to initialize the genetic algorithm, where a feature is more likely to be used if the filter measure is high. Then, the genetic algorithm is run using a wrapper measure to rate the solutions. Using a filter measure as a method of initialization is a good way to speed feature selection. In [24], the search algorithm is a genetic algorithm, but this technique would work well with many stochastic search algorithms (see Section 2.4.3), or with oscillating search or dynamic oscillating search sequential search algorithms (see Section 2.4.3), which also require initialization.

Peng, Long and Ding [122] suggest using a wrapper as a post processor after applying a filter measure. This is similar to [145] and [144] for oscillating search and dynamic oscillating search.

It is also possible to use a combination of filter measures. Dhir and Lee [44] present a feature selection algorithm that combines Fisher's criterion and mutual information filter measures. They find that the combined filter measure outperforms the individual filter measures.

Forman [50] compares a number of different filter measures for text classification and finds that filter measures can have complimentary errors, and like feature sets themselves, the best two individual filter measures may not be the best two filter measures to use together. It appears that, like feature sets themselves, filter measures may suffer from a nesting problems.

Embedded

Embedded feature selection techniques select features as a part of the training process for the classifier. The search and the feature evaluation measures are usually incorporated directly into the training technique.

One of the more common embedded feature selection methods is the least-absolute shrinkage and selection operator (LASSO) [150]. LASSO applies a penalty term to the classifier optimization problem, minimizing the $\mathcal{L}_1$ norm of the input weights. Due to the nature of this penalty term, some of the input weights are set to exactly zero, resulting in a feature selection process. Although the LASSO was originally proposed for least squares regression problems, the penalty term can be applied to a variety of pattern recognition techniques. The LASSO penalty has be applied to logistic regression classifiers [134] and support vector machines (SVM) [174].

An alternate to the LASSO is the elastic net [176], which includes both $\mathcal{L}_1$ and $\mathcal{L}_2$ penalty terms. This formulation tends to give solutions where highly correlated features are included or discarded as a group. This may or may not be desirable for a particular problem, but the elastic net does outperform the LASSO on many real-world problems, particularly if the number of features is much larger than the number of available examples in the data set [176]. Like the LASSO, elastic net can be used with a variety of classifiers [119].

Several embedded feature selection techniques have been proposed for neural networks, for example node pruning [106]. A node saliency measure is defined as the cost of removing each input or hidden node from the network. The node with the lowest saliency is then removed and the process is repeated. Because input nodes can be removed, this is essentially a feature selection technique. This is similar to SVM embedded feature selection [118], which repeatedly removes the feature that least affects the accuracy.

A genetic algorithm can be used as an embedded feature selection technique for neural networks by jointly training the neural network weights and optimizing the structure [171]. Because input nodes can be removed as a part of the structure optimization, the technique also performs feature selection.

2.4.3 Feature set selection techniques

The feature set selection technique is used to select and adjust the set of features to be included in the set, based on the information provided by the feature set selection measure. These fall into a number of broad categories, outlined below.

Ranked selection

Ranked feature selection techniques apply a performance measure to each feature and select the desired number of features based on the highest score or threshold. This method

assumes that the features are independent and hence the performance of feature combinations is equivalent to the combined performance of the individual features, which is a bad assumption in many cases [57].

Ranked selection is common when using filter-based feature set evaluation measures, as many measures are univariate, and can only evaluate the individual features rather than the subset. Ranked selection is also common in domains where the number of features is very large, such as text categorization or gene expression data [96, 99, 168, 6].

If a set has redundant features that are well ranked, ranked selection will include all the copies of the features. This can reduce classifier accuracy as the different features do not include any additional information.

Ranked selection will also not include features that are uninformative alone, but informative when combined with other features [143]. Consider, for example, the standard XOR problem. Individually, neither feature is informative, but taken together, these two features create a well separated set of classes. For these data sets, ranked selection will not be sufficient to find a good feature set.

Simple adjustments to ranked search can improve performance, remove correlated features or find features that are not informative alone. Yu and Liu [169] present an information-theoretic feature-set evaluation measure called symmetric uncertainty for use with ranked selection. Symmetric uncertainty also gives a measure of the correlation between the feature and the class. However, the authors recommend also calculating the symmetric uncertainty between the features, and only including a feature if the symmetric uncertainty between the feature and the class is over a certain threshold, and is higher than the symmetric uncertainty between the feature and any other feature in the set. In this way, the ranked selection method is adjusted to remove highly correlated features. This method only accounts for pairwise correlation, but is still better than ranked selection alone.

Another variation on ranked selection is dependency-aware feature selection (DAF) [143], which ranks each feature based on its performance in various sets of features. The method uses a series of probe data sets of various numbers of features, selected randomly. Each probe set is evaluated on the classifier using a wrapper method. Each feature is then ranked using the results of these probes, where the measure of each feature is the average accuracy of the probe sets that include the feature, minus the average accuracy of the probe sets that do not include that feature. This is less computationally expensive than using a full search technique, but better captures the performance of features that only perform well when combined with other features. Such a technique can be used when the number of features makes a full optimization technique computationally infeasible, or

when there are concerns that the distributions of the training and testing sets differ. It is, however, important to note that like any ranked technique this technique may include redundant features.

Basic search

Basic search-based feature selection techniques include techniques such as exhaustive search and branch-and-bound. These tend to be impractical for most data sets because of the time required, or because they assume the data is monotonic. This is a particular problem for branch and bound [175, 127]. Hence, basic search techniques are not often used.

Sequential search

The most basic sequential techniques are sequential forward search (SFS) [165] and sequential backwards search (SBS) [107]. For sequential techniques, forward implementations add features to an empty set, whereas backwards techniques remove features from a complete set. SFS adds significant features one by one to the selected set until the desired set size is reached. This is different than ranking, since the significance of each feature is evaluated with respect to the selected set. Once a feature is added, however, it cannot be removed and hence these basic techniques suffer from nesting problems. They can also get stuck in local minima. A number of different sequential search algorithms have been proposed to overcome these problems.

Generalized sequential forward search (GSFS) and generalized sequential backwards search (GSBS) [80] are an extension of SFS/SBS. GSFS/GSBS help solve the problem of local minima, but still suffer from nesting problems. Plus-l-minus-r (PTA) [146] allows features to be removed after they are added, with l features being added at each iteration, and r features removed. However, there is no theoretically sound way to set the l and r values [127, 175]. A further extension of the PTA algorithm are floating search methods [127]. These methods allow the l and r values to "float" and change their value. In general, these floating methods, the sequential forward floating search (SFFS) and the sequential backwards floating search (SBFS) are quite common because they solve the nesting problem but are still less computationally expensive than exhaustive search.

A number of other algorithms build off the ideas in SFFS/SBFS. Improved forward floating search (IFFS) [116, 115] adds the ability to conditionally replace features in the selected subset. Oscillating search (OS) [145] starts with a feature set of the desired size, rather than a null or full feature set. The initial set of features is selected normally by

SFS, but can be selected using any algorithm. Features are then conditionally removed and added to the set in a similar manner to SFFS/SBFS. Dynamic oscillating search (DOS) [144] starts with a set of the desired size, and conditionally removes and adds features like oscillating search. However, dynamic oscillating search allows the feature set size to change if this causes an improvement. Because both OS and DOS begin with a feature set of the desired size, these techniques can be used also as a post-processing step for other feature selection techniques, as a way to refine the feature set.

Research on forwards vs. backwards algorithms generally assumes they are fairly comparable. If the aim of the algorithm is to generate nested subsets, forward algorithms are, in general, more computationally efficient because the feature sets contain fewer features. Some authors postulate that backwards techniques can generate stronger feature subsets because even at the start of the algorithm the features are considered alongside other features that can affect their performance. There are, however, cases where forward selection is able to select a more robust set than backwards selection [57]. In cases where nested sets are not selected, for example when using a floating search method, forwards and backwards algorithms tend to give similar results [127, 175, 85]. Forward algorithms tend to be used because they are generally more computationally efficient, although using a combination of the forward and backwards algorithms can also help [85].

Stochastic

Stochastic search techniques use a randomized initialization or updates. This category includes algorithms such as simulated annealing, genetic algorithms, ant colony and particle swarm optimization. These can perform well [85], but are more computationally expensive than sequential search. They can also get trapped in local minima, and because they have a random initialization it is possible for different runs to produce different feature sets.

One of the simplest stochastic techniques is the Las Vegas technique [117]. This technique randomly selects a subset of features, evaluates the subset, and tracks the best performing set. It is commonly used with a consistency-based filter measure, and is called the Las Vegas Filter (LVF). Sampling can also be used to reduce the computational requirements [117]. Because this technique selects sets at random, there is no guarantee that it will find or approach the best subset. Hence, more sophisticated techniques tend to be used.

Genetic algorithms [76] are a common technique for feature selection. The genome is used to represent the selected feature subset, with each binary gene representing one

feature. Genetic algorithms can also be used for feature weighting, by using continuously valued genes [59].

Simulated annealing and particle swarm optimization have also been tested for feature selection [54]. Chuang, Tsai and Yang [32] present a particle swarm algorithm for feature selection that allows some particles to be overwritten with poor "catfish" particles to reduce premature convergence. They find that their new algorithm outperforms sequential and genetic algorithms.

Aghdam, Ghasem-Aghaee and Basiri [3] present a method for using ant colony optimization. First, the problem is constructed as a graph, with each node representing one feature. Ant colony optimization is then used to find the shortest path through the feature graph that can meet an objective for the data set.

Zhang and Sun [172] suggest Tabu search for feature selection. Tabu starts with one feature and moves to the best "neighboring point" (one feature added or removed from the set) provided that the point is not on the "tabu" list. Points are entered on the tabu list once the algorithm moves to a new point, and the list is used to prevent backtracking.

Feature selection as a direct optimization problem

Although feature selection is normally presented as a search problem, feature selection can also be formulated as a direct optimization problem, as described in Section 2.2. In convex principal feature selection (CPFS) [109], the goal is to select a subset of features such that the squared reconstruction error is minimized. This is very close to PCA, except that the transformation matrix B is constrained to be a diagonal matrix of zeros and ones. The method optimizes

$$\min_B ||X - BX||^2 + \lambda \sum_i = 1^F ||\mathbf{b}_i||_L \infty \tag{2.44}$$

where B is an $F \times F$ diagonal matrix of zeros and ones used to select the features, and the term $\lambda \sum_i = 1^F ||\mathbf{b}_i||_L \infty$ is an L_∞ penalty term used to encourage sparse solutions. The L_∞ norm is found by taking the max absolute value of $\mathbf{b}_i$. Equation 2.44 is a quadratic optimization problem with linear constraints and can be solved using a variety of methods.

Hybrids

Hybrid search algorithms use more than one type of search technique to better optimize the feature set. Two of the most common hybrid techniques are oscillating search [145]

and dynamic oscillating search [144], which require an initial set of features as an inputs and can also be used as post-processing techniques for any feature set selection technique. Normally, a filter-based measure and a ranked selection process is used to generate the input, and a wrapper measure is used to perform the oscillating search.

Cantu-Paz [24] suggests an extension to the genetic algorithm technique, where genomes are initialized based on the output of a filter measure. A feature with a higher filter measure score is more likely to be selected. This type of hybrid method was able to produce feature subsets with fewer features than the uninitialized genetic algorithm, with similar accuracy. This method, where a filter measure is used to initialize an more complex optimization method could theoretically be applied to a number of the stochastic feature selection techniques.

Gheyas and Smith [54] present a hybrid technique that uses multiple search algorithms to generate a feature set, using simulated annealing, genetic algorithm, then greedy hill climbing. Their algorithm does outperform others when the run time is limited, but given a sufficient training time, other techniques also perform well. Unfortunately, the stopping points are set by run time, which is particular to the implementation, which makes this specific technique difficult to replicate.

Comparison

Ranked selection is generally used when the number of features is high, and it is computationally infeasible to use a more complex feature set selection method. Aghdam et. al. [3] test ant colony optimization and genetic algorithm against ranking-based selection and also find that the more complex search strategies outperform simple ranking-based selection.

Liu et al. [99] compare ranked selection with different feature measures to correlation-based feature selection (CFS). Their study indicates that CFS, the only subset-based measure, is superior to the other ranking-based filter measures.

However, other authors suggest that using complex search techniques can lead to the selection of features that are too specific to the training set and hence do not generalize well [91, 130], and that better results can be obtained using simpler search techniques.

A number of papers have tested the SFFS and SBFS algorithms against various other sequential search algorithms. The works in [127] [175] both recommend SFFS and SBFS over other sequential search algorithms. Jain and Zongker [71] also test floating search against ANN node pruning [106] and again recommend floating search. Kudo and Sklansky

[85] compare floating search techniques against other sequential techniques and against stochastic search. They recommend either SFFS/SBFS or genetic algorithms.

However, other authors argue that the floating techniques may select features that are too specific to the training set and simpler techniques may select features that generalize better to unseen data. Reunanen [130] compares SFS and SFFS and finds that SFS outperforms SFFS in a significant number of cases.

Kudo and Sklansky [85] compare genetic algorithms to a number of sequential and simple search techniques and find that the genetic algorithm performs very well and is quite fast. They recommend either a genetic algorithm or a floating sequential search method.

However, ant colony optimization [3] and tabu search [172] are both able to slightly outperform genetic algorithms. Tabu search has the additional benefit of being less computationally expensive [172].

Gheyas and Smith [54] test a number of optimization techniques including SFS/SBS, SFFS/SBFS, ant colony optimization, simulated annealing, and particle swarm optimization and their own hybrid technique. They recommend their hybrid technique or simulated annealing, or SFS if simulated annealing is considered too computationally expensive.

Overall, there does not appear to be one single method that consistently outperforms the others.

2.5 Hierarchical classifiers and multi-class extensions for binary classifiers

Classifiers can either be multi-class, meaning they are capable of separating data into many classes, or binary, meaning they are capable of separating only two classes. Binary classifiersA can be extended for use in multi-class problems in a number of ways. These methods can also be applied to multi-class classifiers and can improve accuracy in some cases [137]. Common multi-class extension methods, including hierarchical methods, are discussed in this section.

2.5.1 One vs. rest

The most basic extension from a binary classifier to a multi-class classifier is one vs. rest. In this formulation, C binary classifiers are trained, where C is the number of classes. Each

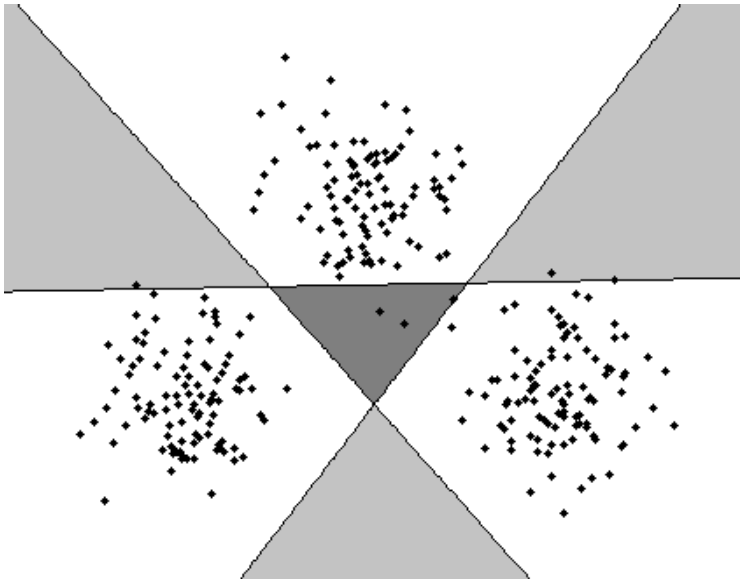


Figure 2.3: One vs. one extension for a three class problem. Points in the light grey areas are claimed by more than one classifier. Points in the darker grey area are claimed by no classifier

classifier is trained with one of the classes as the "one", and the remaining classes grouped as a single "rest" class and hence all the samples are used to train all the classifiers in this scheme. If a single classifier claims a sample is in the "one" class, and all of the remaining classifiers classify the sample as being in the "rest" category, then the point is classified as being in that class. Classifying a new point requires querying C classifiers. It is also possible for a point to be claimed by more than one classifier, or by no classifiers (see Fig. 2.3). In this case, the point is considered to be unclassifiable.

2.5.2 Fuzzy one vs. rest

In the fuzzy extension to the one vs. rest method for SVM, [70] the dividing hyperplane for each classifier is a fuzzy membership function, where the degree of membership is decided by the distance from the hyperplane. The authors then show that a simple rule-based system is equivalent to their fuzzy formulation. If the "one" class for each classifier is given a class value of 1, and the "rest" class is given a value of -1, then the point is classified into the class of the classifier returning the maximum value, which is the same as the method described in [21]. This requires the classifier to return an actual value, rather

than a simple class indicator.

Similar to the one vs. rest formulation, the fuzzy one vs. rest formulation requires training C classifiers, and classifying a new point requires querying all C classifiers.

2.5.3 One vs. one

The one vs. one extension trains a single classifier for each pair of classes. Hence, $C(C-1)/2$ classifiers are required, and each is trained using $2N/C$ samples, assuming a balanced class distribution. To classify a point, each classifier is queried, and casts one vote for its winning class. A point is classified into the class with the largest number of votes.

2.5.4 DAG-SVM

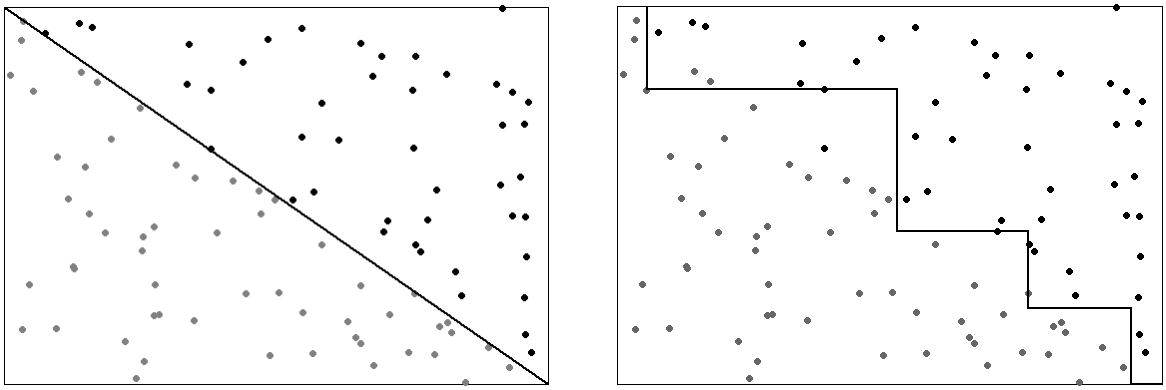
The DAG-SVM [126], similar to the one vs. one formulation, trains a single classifier for each pair of classes, which requires $C(C-1)/2$ classifiers, each trained on $2N/C$ samples. However only $(C-1)$ queries are required to classify a point. A classifier is selected randomly and is used to eliminate one class from consideration. This is repeated $(C-1)$ times, and the point is classified into the last remaining class.

2.5.5 Hierarchical classifiers

In a hierarchical classifier, the set of classes is progressively divided into subsets of fewer classes until there is only one class remaining at each node.

Cheong, Oh and Lee [29] propose a method to create binary trees for SVM using a self-organizing map (SOM) to create the groupings. A human-drawn line on the SOM separates the classes into two outputs. The results are comparable to one vs. one, but manually selected groupings appear to give the best results. Overall, this approach appears to be promising, but is still somewhat subjective, and leaves a human researcher to determine how to best divide the SOM.

Madzarov, Gjorgjevikj and Chorbev [103] use a hard clustering method, similar to k-means. Two clusters are created, starting with the two input classes with the largest distance between their centers. The unassigned class that is closest to a center is added, and the center is updated. This is repeated until all the classes are added and the method is applied recursively.



(a) SVM (b) decision tree

Figure 2.4: SVM and decision tree divisions of a linearly separable data set. The SVM can separate the data using a single, diagonal line, whereas the decision tree is restricted to a piecewise staircase division.

Another tree-based approach is the adaptive binary tree (ABT) [27], which is an extension of an earlier work on binary tree SVM [26]. The goal of ABT is to design a tree-based classifier that reduces the number of support vectors in the nodes, regardless of the number of nodes in the tree. SVMs are trained pairwise on each set of classes. The classifier with the smallest number of support vectors is selected and the remaining classes are classified by the selected classifier and assigned to one or both sides. This procedure is then applied recursively to each side until a single class remains at each node. Trees designed using the ABT method are different than those in [103] because ABT can have multiple leaf nodes for the same class. However, although ABT allows classes to be placed at both outputs, it begins with a pairwise separation. This assumes that the original pair of classes can actually be separated, which is not always the case, particularly for multi-modal distributions.

Tree-based extensions are structurally similar to the classification and regression tree (CART) or decision tree algorithm [19]. In a decision tree, the data set is partitioned into two parts at each node using a simple thresholding on a single feature. Like ABT, classes can be assigned to both sides of a single node and therefore classes can appear at multiple leaf nodes. Decision trees use only a single feature for each division. Because the features can be reselected, decision trees can actually be quite powerful, but create complex, piecewise divisions when dependent features are included. An illustration of this problem is given in Fig. 2.4.

2.5.6 Comparison

Hsu and Lin [65] compare the standard one vs. rest and one vs. one formulations as well as DAG-SVM [126] and formulations that natively extend SVM to handle multiple classes. They find that the one vs. one and DAG-SVM tend to perform better than one vs. all and the multi-class SVMs.

However, Rifkin and Klautau [132] present the view that the most important step is to properly select and tune the underlying binary classifier, and the scheme used for combination makes little difference. They recommend using a one vs. rest scheme due to the fact that fewer classifiers need to be trained.

Compared to other multi-class classification methods, binary trees with a single output node per class require fewer classifiers to be trained, and fewer classifiers to be queried in order to perform a classification.

A binary tree with a single output node per class requires training $(C-1)$ classifiers. The number of samples required to train each base classifier in the tree is N at the top of the tree and gets smaller in the lower levels. Assuming a balanced class distribution, for a fully balanced tree the total number of samples required to train a single classifier at each level is half the number required for the level immediately above, but there are also twice as many classifiers in that level. This gives a total of $log_2(C)N$ samples, over $(C-1)$ classifiers. For a completely unbalanced tree the number of samples decreases by N/C at each level, for a total of $(C-1)$ levels, giving a total of $N((C-1)-((C-1)/C))$ samples, which can be simplified as less than $N(C-2)$ samples over $(C-1)$ classifiers.

The major advantage of the tree-based solution is not the training time, but the testing time. For a fully balanced tree, the number of queries required is $log_2(C)$. For a fully unbalanced tree, the worst-case number of queries approaches $(C-1)$ when the largest class is at the bottom of the tree, which is still fewer than the DAG-SVM. In the best case for an unbalanced tree, the number of queries approaches one and on average, assuming a balanced class distribution, a fully unbalanced tree will require $(C-1)/2$ queries. The tree therefore requires the fewest number of queries of any of these methods and this advantage increases as the number of classes increases.

A comparison is presented in Table 2.3.

method	# classifiers	# queries	# samples / classifier
one vs. rest	C	C	N
fuzzy one vs. rest	C	C	N
one vs. one	$\frac{C(C-1)}{2}$	$\frac{C(C-1)}{2}$	$\frac{2N}{C}$
DAG-SVM	$\frac{C(C-1)}{2}$	$C-1$	$\frac{2N}{C}$
binary tree	$C-1$	$\log_2(C)$	$\frac{N\log_2(C)}{C-1}$ (balanced)
(single leaf node per class)		$\frac{C-1}{2}$	$< \frac{N(C-2)}{(C-1)}$ (unbalanced)

Table 2.3: Comparison of multi-class extension methods

2.6 Algorithms used in this work

2.6.1 Support Vector Machines (SVM)

A support vector machine separates two classes using a dividing hyperplane. The SVM selects the hyperplane that maximizes the distance between the hyperplane and the nearest point on either side. This is called the maximum margin hyperplane, and the points closest to the hyperplane are called the support vectors.

The classifier is trained on a set of N training points $X \in \mathbb{R}^F$. A set of labels Y designates each point as being in one of two classes, 1 or -1.

Assuming that the classes are separable, the separating hyperplane is designated as [21]:

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{2.45}$$

The support vectors form hyperplanes at $\mathbf{w} \cdot \mathbf{x} + b = 1$ and $\mathbf{w} \cdot \mathbf{x} + b = -1$ and all other points lie outside the support vectors [21]:

$$y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1, \forall i \tag{2.46}$$

The maximum margin can be found by maximizing $||\mathbf{w}||^2$, subject to $y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1, \forall i$. Using a Lagrangian, the problem can be formulated as [21]:

$$L_p = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \alpha_i y_i(\mathbf{w} \cdot \mathbf{x_i} + b) + \sum_{i=1}^{N} \alpha_i \tag{2.47}$$

where α_i are the Lagrangian multipliers. The solution is found by minimizing L_p with respect to $\mathbf{w}$ and b, and requiring that derivatives of L_p with respect to all α_i are zero and $\alpha_i \geq 0$.

This can also be formulated as a dual problem, maximizing L_p subject to the constraint that the gradient of L_p with respect to $\mathbf{w}$ and b are zero. This gives the conditions [21]:

$$\mathbf{w} = \sum_{i=1}^{N} a_i y_i \mathbf{x_i} \tag{2.48}$$

and

$$\sum_{i=1}^{N} a_i y_i = 0 \tag{2.49}$$

Adding these constrains gives the dual formulation [21]:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j} \tag{2.50}$$

subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{2.51}$$

and

$$\alpha_i > 0, \forall i \tag{2.52}$$

Non-linearly separable classes can be separated by allowing a limited number of training errors, controlled by a slack variable. This produces a dual problem that is the same as the original formulation, but with a tighter constraint on the α variables such that

$$0 \leq \alpha_i \leq C, \forall i \tag{2.53}$$

where C is chosen by the user and controls the amount of penalty applied for misclassified points.

Points can be classified simply by taking the sign of the distance to the normal of the hyperplane. Therefore the test time for a linear SVM is only dependent on the dimensionality.

Training an SVM requires solving the constrained optimization problem, and the solution can be found via quadratic programming, or using any of a variety of SVM-specific training methods such as sequential minimal optimization (SMO) [125, 126] or decomposition [73], which break the large optimization problem into a set of smaller problems, or least squares SVM, which solves a slightly different formulation of the problem [147].

2.6.2 K-nearest neighbours (KNN)

K-Nearest Neighbours (KNN) is a relatively simple classification technique. For each new input, the distance between the input vector and each of the training vectors is calculated and the class of the new input is determined as the majority class of the K closest training vectors. In the case of a tie, the class is recursively calculated using $K-1$ points, until a class is found [34].

2.6.3 Genetic algorithms

Genetic algorithms (GA) are a guided random search technique that is a form of evolutionary computing.

Genetic algorithms function similarly to biological evolution and the components of the system are similarly named. A set of possible solutions is called the "population" and each potential solution to the problem is called an "individual". The set of an individual's traits are called its "genome". The genome is made of individual "genes", which are a single, limited, discrete value that controls some part of the individuals structure or behavior. [76].

The genetic algorithm proceeds as follows. An initial population of individuals is randomly generated. The fitness of each individual is assessed. High scoring individuals are paired randomly and a combination of one or more evolutionary operators is then used to create two new solutions that replace low scoring individuals in the population. The algorithm then repeats until the stopping condition is reached. The stopping condition can be a threshold on the solution or the amount of change in the population, or it can be a limit on the number of generations in the model. This procedure is illustrated in Fig. 2.5.

The evolutionary operators are crossover and mutation. The crossover operator creates two children by combining the genes of two parents. One point crossover, illustrated in Fig. 2.6a, randomly selects one cut point in the genome and creates new genomes using the first part of one parent and the second part of the other parent [76]. This can be extended to K-point crossovers, which select K points and alternate sections from each parent (see Fig. 2.6). The mutation operation makes a random change to one or more genes, as illustrated in Fig. 2.6d. The amount of mutation allowed in a population is controlled by a parameter called the mutation rate.

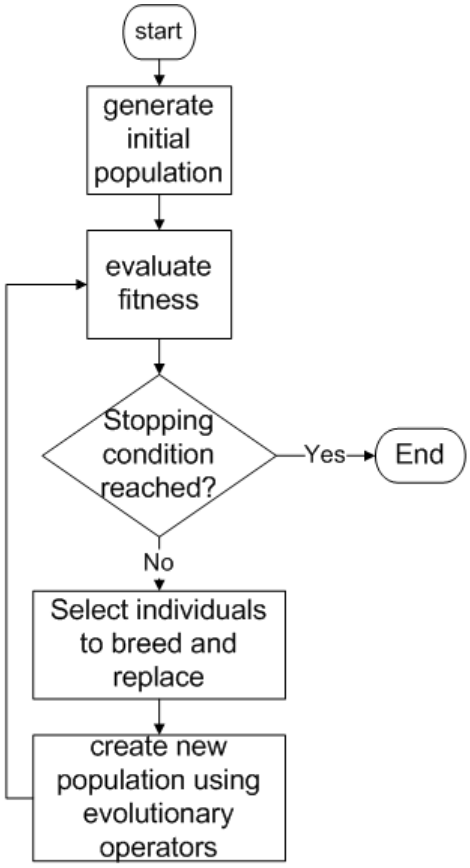
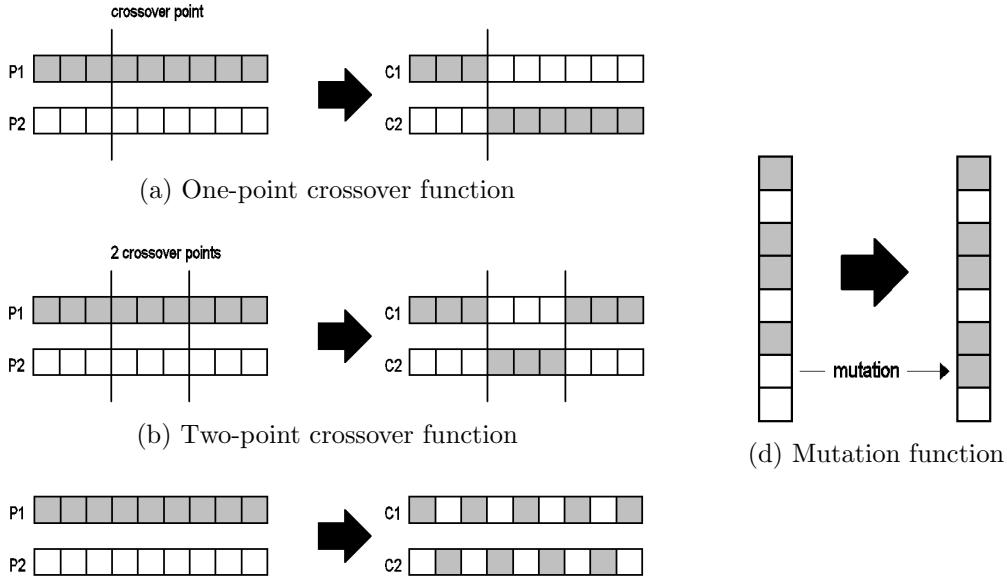


Figure 2.5: General flow of a genetic algorithm



(a) One-point crossover function

(b) Two-point crossover function

(c) Uniform crossover function

(d) Mutation function

Figure 2.6: Common evolutionary operators

2.6.4 K-means clustering

K-means clustering is a centroid based clustering where the objective is to find a partitioning of the data such that the sum squared distance between points in a cluster is minimized. The objective therefore is to find a set of clusters that minimizes

$$\sum_{i=1} K \frac{1}{|C_i|} \sum_{x,y \in C_i} d(x,y)^2 \tag{2.54}$$

where K is the desired number of clusters, C_i is the set of points in cluster i, and d is a distance metric between points x and y [1]. Euclidean distance is a common choice.

Unfortunately, this is an NP-hard problem, so often heuristic algorithms are used to estimate a solution. The most commonly used algorithm is the K-means algorithm. The algorithm is initialized first by assigning k random center points, where k is the desired number of clusters. In the first stage of the algorithm, each of the points is assigned to the cluster of its closest center point. In the second stage, the cluster centers are updated as the average value of all the points assigned to the cluster. The algorithm then continues back to the first stage, assigning points to the updated cluster centers. This is repeated until the centroid values converge.

2.7 Human motion recognition

Motion recognition is used in a variety of different research areas. Imitation learning for humanoid robotics is inspired by human and animal studies indicating that movements can be learned through observations of other actors [111] [138]. Imitation learning for humanoid robots builds off this idea and uses exemplar human motions to build motion models for robots. This type of imitation learning incorporates motion recognition [84], as motions first need to be recognized before they can be used to build movement models.

Motion recognition is also becoming more common as a component in human computer interaction. Gesture and motion control interfaces such as the Kinect [142] use motion recognition as a way to control on-screen events. In this system, body part positions are recognized from image and depth sensor data and motion recognition is performed using skeletal data. Other types of gesture recognition interfaces classify motions based on different types of input data. For example, muscle-computer interfaces use gesture recognition from electromyography (EMG) signals for control of games or other user interfaces [110], [31].

Many different classifiers have been used to recognize motion data. Lösch et. al. [101] test motion recognition with Bayes and Naive Bayes networks, MLP and radial basis function (RBF) ANNs, and SVM. They do not find that any one classifier is significantly better in classifying motion data. Other works have also tested using time-delay ANNs [138] and randomized decision forests [142].

Hidden Markov Models (HMM) are a particularly common choice for this application [88, 15, 138, 2] for a number of reasons. Motion data naturally has a time-based component, as the demonstrator moves through various poses, and HMM natively handles time-series data. The model representation of the motions has an intuitive representation, where each state is a primitive motion or pose, and the state transition matrix describes the flow of the motion through these various primitives. Lastly, HMMs are generative models and hence they can also be used to generate a primitive trajectory for a controller [89]. Extended versions of the HMM algorithm, such as factorial hidden Markov models, have also been successfully applied to motion recognition [89].

A number of papers have examined and implemented feature selection and dimensionality reduction for human motion recognition. Human studies indicate that partial knowledge of the body position may be sufficient for motion recognition. Humans are able to discern motion types even in abstract human-like shapes. Blake and Shiffrar [17] review research on point-light displays, where the motion of a human is represented as point markers on the major joints. Humans are able to robustly recognize movements even from these very

simple displays, in a wide variety of configurations and noise conditions. Hence even for humans, motion can be represented by quite a small subset of features.

PCA is another common method for dimensionality reduction. Fod, Mararić and Jenkins [49] present a method for motion imitation and achieve dimensionality reduction using PCA. The reduced data set is clustered, with each cluster representing a motion primitive. Incoming motions are classified as being a part of a cluster and the cluster data can be used to recreate the motion. PCA is able to reduce the dimensionality from a 400D vector to a 30D vector that is suitable for movement generation.

Takano and Nakamura [148] use multi-dimensional scaling to create a lower-dimensional representation of HMMs. The system is tested on-line using motion capture data with seven distinct motions, and after the multi-dimensional scaling, seven clusters are identified. They find that the lower dimensional representations can be used for both recognition and generation of motions.

Bitzer and Vijayakumar [16] present a method for dimensionality reduction using Gaussian process latent variable models (GPLVM). In GPLVM, the mapping from the latent to the observed values is modeled as a Gaussian process. In their work, Bitzer and Vijayakumar add an additional constraint that the observed sequences be translated versions of a template action. This maps directly to the dynamic motion primitive (DMP) representation of motion as having a goal state, a start state and a trajectory. DMPs can be used to generate unobserved motions of the same type by changing the start and goal states and regenerating the trajectory. They term this constrained version the simple sequence prior GPLVM (SS-GPLVM). They test this technique on motion data, where the goal is to generate a new DMP using the dimensionally reduced inputs. They find that the SS-GPLVM is able to create a set of lower-dimensional inputs that can be used to generate reasonable new motions. SS-GPLVM performs better than other tested dimensionality reduction techniques (PCA and standard GPLVM), and using joint-space data directly.

Jenkins and Matarić [72] present an extension to the Isomap dimensionality reduction technique (see section 2.3.4), called spatio-temporal-Isomap (ST-Isomap) that can incorporate temporal sequences. In ST-Isomap, the distances between nearest neighbour points are adjusted to account for temporal closeness before the distances are placed in the graph. The authors test the technique on data from a teleoperated robot performing repetitions of a grasping movement, and from a human subject performing dancing, punching and waving motions. They find that ST-Isomap is able to identify the repetition in the grasping data set and can extract clusters that match to the major primitives within the movements and provides a significant reduction in the number of features. However, these new features are not tested for classification.

Billard et. al. [15] present a system for motion recognition and imitation using a simple feature selection method. The authors use four specific types of count-based features that are extracted from the video of the demonstration and define the most important feature as being the feature that is most common. Probabilities for each demonstration type are calculated using its related feature. While this technique is able to use a single salient feature to determine the demonstration class, it is important to note that the features have been manually pre-selected specifically to match these scenarios. Selecting the most salient of these features is not the same as selecting a salient feature from a large group of input features, whose relation to the output class is unknown.

EMG-based gesture recognition has been performed using a number of common classifiers, including linear discriminant analysis (LDA) [123], artificial neural networks (ANN) [95][78], naive Bayes [28], hidden Markov models [164] and fuzzy inferencing systems [78].

Much of the work on EMG gesture recognition focuses on domain-specific feature extraction techniques, applying standard signal processing techniques to create descriptive features for EMG signals. A comparison of EMG features in [123] describes multiple time and frequency based features used by many researchers. Each feature is calculated on clean and noisy signals and uses single feature classifiers to compare feature strengths. Using an ad-hoc feature selection technique, they also show that multi-feature classifiers are beneficial.

Standard feature extraction techniques have also been used as a post processing technique to reduce the number of extracted input features. For example, Khezri and Jahed f[78] use PCA and class separation after extracting time and frequency domain features from the EMG data.

2.8 Summary

While there has been a good deal of research in the areas of dimensionality reduction, multi-class classification extensions and human motion recognition, there remain a number of areas that would benefit from further study.

Feature set selection can be performed in a number of ways, and researchers have achieved good results with a variety of different techniques from floating sequential methods such as SFFS [127, 175] to more complex stochastic optimization methods such as genetic algorithms [59], simulated annealing [54], particle swarm optimization [54], ant colony optimization [3] and Tabu search [172]. One concern with using complex feature set selection techniques is that they may select features that are too specific to the training set

and do not generalize well [91, 143]. This is a particular problem if the number of samples is small, if the feature set is not the only parameter being optimized or when the testing and training sets are known to have different distributions.

Feature set evaluation can also be performed in a number of ways. Wrapper methods have been found to perform well [81], but may select features that are too specific to the feature set and do not generalize well [130]. They are also often computationally expensive and therefore impractical for many applications. Filter-based measures do not require training a classifier and therefore can be less computationally expensive. Many different filter-based measures have been proposed but there has been little work examining how well these perform for different classifiers, and no study directly comparing many of these metrics. Different classifiers work well with different feature sets [96] and hence the appropriate filter measure is likely classifier specific.

Multi-class extensions such as one vs. rest and one vs. one are mostly studied for use with binary classifiers, but can also be beneficial when used with multi-class classifiers [137]. Binary trees are promising multi-class extensions because they require querying fewer classifiers to classify each point. There are multiple different algorithms for creating hierarchical tree-based classifiers [29, 103, 27]. Both the SOM-based tree [29] and the adaptive binary tree [27] work well when classes are allowed to appear at more than one leaf node. There is little work examining the design of tree-based classifiers with feature selected inputs.

One additional benefit of tree-based classifiers are that the tree structure mimics the natural clusterings of human motions, as discussed in the work of Kulić, Takano and Nakamura [88], and thus a hierarchical classifier structure may work well for human motion applications.

Much of research in dimensionality reduction for human motion applications has focused on feature extraction techniques. While these techniques may provide useful features, there are several drawbacks to using feature extraction alone. Feature extraction can be computationally expensive as it requires computation of all features as well as a transformation into the new space. The new features also do not have a direct physical interpretation. This makes it difficult to use these new features when building a movement model. Feature selection or a combination of feature extraction and feature selection may be beneficial for human motion recognition and modeling.

Chapter 3

Hierarchical Classifier Design

The performance of a classifier is affected by a number of factors including classifier type, the input features and the desired output. This chapter presents and evaluates two methods for selecting both the input features and classifier outputs through the use of classifier trees.

Feature selection can improve performance in a number of ways. Reducing the number of input features can reduce classifier complexity and improve classification speed. Proper feature selection reduces the impact of noisy features, which can cause false association between relatively random features and the classifier output. Feature selection can also mitigate the impact of redundant or correlated features, which increase classifier complexity without adding much additional information, adversely affecting performance [57].

Output selection, or classification problem division, describes the process of subdividing a multi-class classification problem into a set of classification problems that each use only a subset of the classes or samples and determining the overall classification through the combined decision of the sub-classifiers. Dividing the problem can improve accuracy [137], and for some classifiers, better generalization can be achieved by using a sparse set of inputs [106]. Additionally, in some cases, different classes can be best separated using different sets of features. Splitting the problem allows the use of a more specific set of features for each set of classes.

The question of how best to divide the problem is not trivial and remains an active research area, particularly for support vector machine (SVM) classifiers, which are binary classifiers and thus require any multi-class problem to be divided. Although there are direct extensions to the SVM classifier that allow it to handle more than two classes natively [163, 160], it is more common to divide the multi-class problem into a set of binary problems as these tend to achieve better accuracy [163].

There are a number of common methods for dividing a multi-class problem, such as one vs. rest (OVR) or one vs. one (OVO). However, it does not appear that any one method is the best solution in all cases [132, 65]. Binary trees divide the problem hierarchically, with each level splitting the classification task into a progressively smaller set of classes [29, 103]. Trees are intuitively a good choice because they do not require all the trained classifiers to be queried when classifying new points and thus the testing time is shorter.

This chapter presents two methods for creating feature selected hierarchical trees of classifiers. The feature-selected hierarchical classifier (FSHC) method performs the feature selection and classifier design simultaneously using genetic algorithms. The easiest division in the class space depends on the features selected, while the best features to use at each node depend on the class division. Because these two factors are mutually dependent, the algorithm simultaneously optimizes the base classifier outputs and the features, and evaluates the tree as a whole. The algorithm can be used with either binary or multi-class classifiers, where there are more than two outputs [51]. A new feature set evaluation measure is also presented for use with this algorithm.

The feature-selected multi-modal binary tree (FS-MBT) method aims to overcome some of the challenges of the feature selected hierarchical classifier. There are two main challenges with FSHC. FSHC appears to select a class structure and features that are too specific to the training set and therefore the designed classifier does not always generalize well. It also cannot separate non-linearly separable class divisions. FS-MBT uses a sequential technique for tree design and feature selection where first the class separation is determined then the features are selected. Additionally, FS-MBT tolerates misclassification in the tree, allowing points to be classified in either output regardless of the initial specified class groupings. This creates a piecewise linear separation of the data from different classes, which means non-linear and multi-modal data sets can be separated using a set of linear SVMs. Hence, this algorithm creates SVM trees that can separate classes that have a multi-modal distribution or multiple clusters of samples within the same class.

3.1 Feature-selected hierarchical classifier (FSHC) [1]

The feature selected hierarchical classifier (FSHC) uses a hierarchical set of classifiers to progressively separate a large set of classes into smaller subsets of classes. This work extends the ideas presented by Cheong, Oh and Lee [29] by providing a more robust method

[1] Versions of the work in this section have been previously published in [51] and [52]

for designing the tree structure and extending the tree structure to work with multi-class classifiers. It also performs feature selection for the individual base classifiers in the tree.

The feature selected hierarchical classifier is designed using a genetic algorithm that simultaneously optimizes the features and tree structure. Genetic algorithms are selected as a basis for this work due to their proven effectiveness for feature selection [85], and due to the ease of specifying both the feature selection and the tree structure in the genome.

3.1.1 Feature Selected Hierarchical Classifier Design

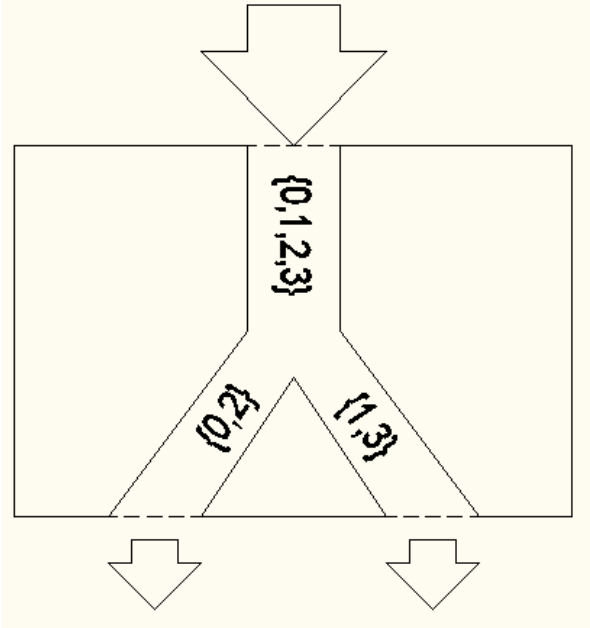
The feature selected hierarchical classifier is designed using a genetic algorithm to jointly specify the tree structure and features at each node. Genetic algorithms (GA) are a guided random search technique that are a form of evolutionary computing.

Genetic algorithms function similarly to biological evolution and the components of the system are similarly named. A set of possible solutions is called the "population" and each potential solution to the problem is called an "individual". The set of an individual's traits are called its "genome". The genome is made of individual "genes", which are a single, limited, discrete value that controls some part of the individuals structure or behavior. [76].

The genetic algorithm proceeds as follows. An initial population of individuals is randomly generated. The fitness of each individual is assessed. High scoring individuals are paired randomly and a combination of one or more evolutionary operators is then used to create two new solutions that replace low scoring individuals in the population. The evolutionary operators are crossover and mutation. The crossover operator creates two children by combining the genomes of two parents. The mutation operation makes a random change to one or more genes. The algorithm then repeats until the stopping condition is reached. The stopping condition can be a threshold on the solution or the amount of change in the population, or a limit on the number of generations in the model.

Genetic algorithms have been used in a variety of optimization problems including feature selection [85, 24] and have also been used in SVM-based clustering problems [159].

This section describes how the tree-based hierarchical structure is defined and represented in the genome, how it is built and scored, and the selection and evolutionary operators used.

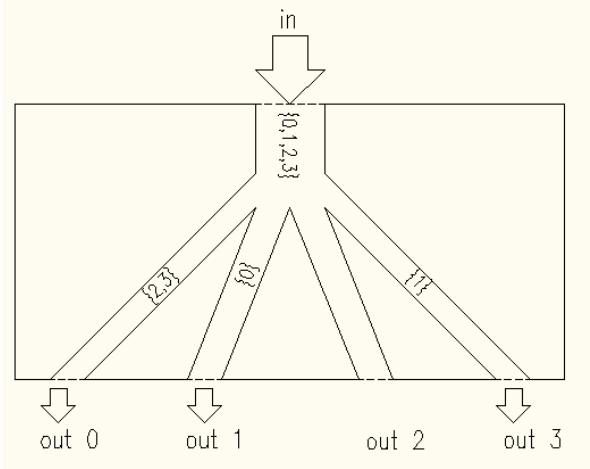


...	Classifier sec.				
...	level 0				...
...	c_0	c_1	c_2	c_3	...
...	0	1	0	1	...

Figure 3.1: Dividing a multi-class problem into binary outputs. The left side of the figure illustrates a base classifier dividing the classes into different outputs and the right side of the figure shows the portion of the gene that describes this base classifier. Points that are in classes 0 or 2 are sent to output 0, and points that are in class 1 or 3 are sent to output 1. Another base classifier below each output would further separate the points from the outputs.

Hierarchical classifiers

A hierarchical classifier uses a set of hierarchically organized base classifiers that separate the data into progressively smaller groups of classes. Consider each base classifier as a black box with two outputs. The job of each base classifier is to place points from each class into its assigned output. Each output can either represent a single class or a set of classes (see Figs. 3.1 and 3.2). If the base classifier has more than one class assigned to one output, another base classifier is added below to further separate the classes. Hence the output of the higher level base classifiers determines if there are lower level base classifiers. The tree structure is specified in the genome by specifying the outputs for each class at each level of the tree.



...	Classifier sec.				
...	level 0				...
...	3	1	0	0	...

Figure 3.2: Dividing a multi-class problem into multiple outputs. The left side of the figure illustrates a base classifier dividing the classes into different outputs and the right side of the figure shows the portion of the gene that describes this base classifier. Points that are in classes 0 or 2 are sent to output 0, and points that are in class 1 or 3 are sent to output 1. Another base classifier below each output would further separate the points from the outputs.

Genome Representation

The genome consists of two parts. The first part of the genome is used to determine which features are used in each base classifier. The second portion of the genome is used to determine the classifier structure by specifying the class outputs at each level of the classifier.

In a multi-class classifier, the smallest number of base classifiers required is one, where each of the C classes in the first classifier maps to a single output. The largest number of base classifiers required is $(C-1)$, which occurs if each base classifier is binary (see Fig. 3.4). The largest number of classifier levels is $(C-1)$, which occurs when each classifier separates a single class. This is illustrated in Fig. 3.4b.

The first portion of the gene gives a true or false (one or zero) value for each feature for each possible base classifier. The largest number of base classifiers required in a system is $C-1$. Hence, the first portion of the genome consists of $F(C-1)$ genes, where F is the number of features.

The second portion of the genome gives the output number of each class in each layer of the genome. The largest possible number of layers required in a hierarchical classifier is

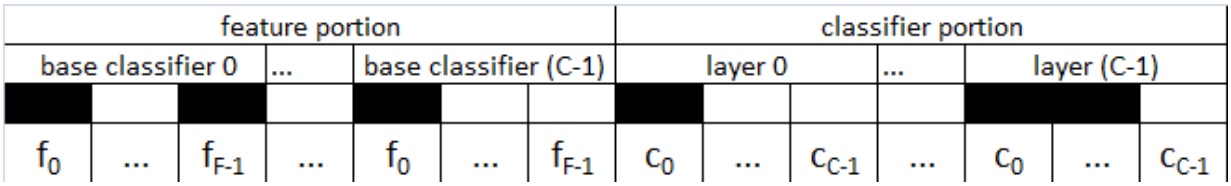


Figure 3.3: Full genome used for feature selection and classifier construction. The first portion is used for feature selection and has one gene per feature for each base classifier in the tree. The second portion describes the tree by giving the output number for each class in each level of the tree.

$C-1$. Hence to specify the output for each class in each layer, the second portion of the classifier is $C(C-1)$ genes each of which can take a value between zero and $C-1$ for a multi-class classifier or zero and one for a binary classifier.

The full genome is illustrated in Fig. 3.3. Fig. 3.4 illustrates how different tree structures can be generated by specifying the outputs for different classes at different levels of the tree.

Hierarchical Classifier Construction

The hierarchical classifier is built from the second part of the genome, starting from the root.

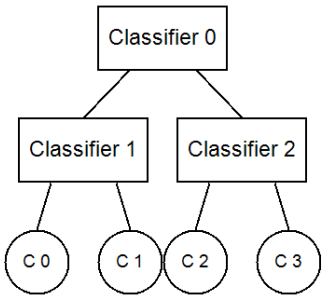
Genes for classes that are fully separated in higher layers are maintained, but once a class is separated, no further base classifiers are constructed for that class and these gene values do not affect the structure. In Fig. 3.4, these genes are marked as 'X'.

Each base classifier knows its immediate parent, its level in the tree, its children classifiers, the features it is using and the output assignment of each class.

The hierarchical classifier is constructed in stages. First, the tree is examined to determine in which layer each class is separated and a root node is added. The tree is then constructed by building the base classifiers off the existing structure for each class one at at time. Pseudocode is given in Fig. 3.5 and Fig. 3.6.

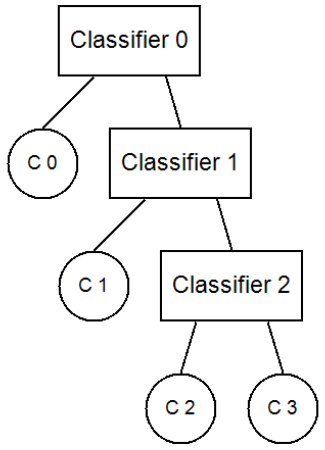
Feature selection for base classifiers

The features used by each base classifier are selected in the first portion of the genome. There is one section for each base classifier, with F integers that can be one or zero, where



...	Second portion of gene											
...	level 0				level 1				level 2			
...	c_0	c_1	c_2	c_3	c_0	c_1	c_2	c_3	c_0	c_1	c_2	c_3
...	0	0	1	1	0	1	0	1	X	X	X	X

(a) balanced binary tree and gene. In level 0, classes 0 and 1 are sent to one output (output 0) and classes 2 and 3 are sent to another (output 1). Hence, in level 1, classes 0 and 1 are separated by one base classifier, added under output 0 of the top level classifier, and classes 2 and 3 are separated by another base classifier, added under output 1 of the top level classifier. All classes are fully separated in level 1, so the level 2 portion of the gene does not affect the tree structure.



...	Second portion of gene											
...	level 0				level 1				level 2			
...	c_0	c_1	c_2	c_3	c_0	c_1	c_2	c_3	c_0	c_1	c_2	c_3
...	0	1	1	1	X	0	1	1	X	X	0	1

(b) long binary tree and gene. In level 0, class 0 is sent to output 0 and all the other classes are sent to output 1. Hence class 0 is fully separated in the first layer. The gene continues separating one class each level by placing a single class in a different output. Genes marked as X are genes for classes that have been separated in higher layers. These are maintained, but do not affect the tree structure.

Figure 3.4: Genes for two four-class hierarchical classifiers. The right side of the figure shows the portion of the genome describing the hierarchical classifier structure and the left side shows the corresponding hierarchical classifier.

```
//determine the layer where class c is separated
for c = 1 to C {
  //classes not yet separated from class c
   initialize array stillInGroup to all TRUE
   numInGroup = numClasses - 1;
   //move layer by layer and check which classes are
   //separated from c in each layer
   for lvl = 1 to nLayers {
      outc = output of class c in level lvl
      for i = 1 to C {
         outi = output of class i in level lvl
         //if the outputs for classes i and c are
         //different, then they are being separated
         //in this layer
         if((outi != outc) && stillInGroup[i]==TRUE) {
            stillInGroup[i] = FALSE;
            numInGroup --;
         }
      }
      if(numInGroup == 0) {
         set class c as fully separated in level lvl
      }
   }
}
```

Figure 3.5: Pseudocode for calculating FSHC class separation layers

```
add root (level 0) base classifier
for c = each class {
   currClassifier = 0//root node
   sepLvl = level where c is fully separated
   numShared = # base classifiers in path already built
   for lvl = 0 to (numShared-1)  {
      outc = output for class c in level lvl
      mark outc in currClassifier
      lastClassifier = currClassifier
      currClassifier = child classifier for output outc
   }
   for lvl = numShared to sepLvl  {
      add a new classifier and initialize
      set parent of new classifier as lastClassifier
      outc = output for class c in level lvl
      set new classifier as child of lastClassifier for outc
      mark outc in new classifier
      set features of new classifier
      lastClassifier = new classifier
   }
}
```

Figure 3.6: Pseudocode for adding FSHC base classifiers

F is the number of features. If the value is one, the feature is used by the base classifier, if it is zero, the feature is not used. This structure can also be extended to perform feature weighting by using non-binary values.

Scoring

The score for each classifier is based on an estimate of the accuracy. Two different methods are used. The first method estimates the accuracy of the tree as a whole, which requires trained classifiers and a wrapper-based measure. The entire tree is built, and the accuracy is estimated by testing directly on the hierarchical classifier.

The second method combines the accuracy estimates of each individual base classifier to estimate the accuracy of the tree. Because wrapper methods can be computationally expensive, filter measures can be beneficial if the classifier requires training or the dimensionality of the data set is high [50]. However, filter measures cannot be directly applied to a tree structure with multiple base classifiers each using different feature sets. Instead, the second method evaluates the accuracy of each base classifier individually and then combines the estimates into an estimate for the entire tree. This allows the use of simpler filter-based measures that can be calculated only on a single base classifier using a single feature set.

When using the second method for estimating accuracy, the number of correctly classified points in a class is estimated by multiplying the total number of points in that class in the training set by the estimated accuracy of each base classifier that is used to classify that class. These numbers are then used to estimate the overall accuracy, as shown in equation 3.1.

$$A_{all} = \frac{1}{N} \sum_{c=1}^{C} N_c \prod_{i \in B_c} A_{ic} \tag{3.1}$$

where A_{all} is the estimated accuracy of the entire tree, B_c is the set of base classifiers that have class c as an output, and A_{ic} is the estimated accuracy of base classifier i on class c.

The accuracy estimates for the scores are calculated using various measures including a validation set on the entire tree, cross validation on the individual base classifiers, Fisher's interclass separability criterion (see Section 2.4.2), RELIEF (see Section 2.4.2) and a new metric termed count-based RELIEF, described below. A scaled version of Fisher's interclass separability criterion is also tested, where the criterion is divided by the square root of the number of features.

Count-based RELIEF is a newly proposed measure that is related to the RELIEF measure. Instead of measuring the distance between the nearest hits and misses, count-based RELIEF counts the number of hits before the nearest miss. While RELIEF attempts to identify features that maximize the average distance between points from different classes and minimize the average distance between points in the same class, the count-based RELIEF attempts to find features that cluster points such that points are close to a large number of points in their own class, regardless of the absolute distance. The new measure has the advantage of being bounded and normalized, even for a non-normalized data sets. This is important since not all base classifiers classify the entire data set, which can result in base classifiers with non-normalized features.

The new measure is calculated as:

$$r_c = \frac{1}{N_c(N_c - 1)} \sum_{p=1}^{N_c} \sum_{i=1}^{N_c} t(x_{pc}, x_{ic}) \tag{3.2}$$

and

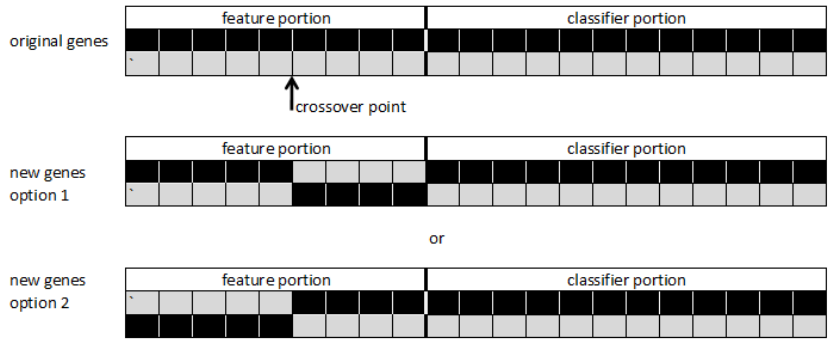
$$t(x_{pc}, x_{ic}) = \begin{cases} 1, & \text{if } d(x_p, x_i) < M_p, i \neq p \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

The score also includes penalty terms for the number of layers in the classifier, the number of features used by the tree and the sum total number of features used by the base classifiers. The score is given as:

$$S = A_{all} - \alpha_t \frac{F_t}{F} - \alpha_b \frac{F_b}{F(C-1)} - \alpha_l \frac{N_l}{C-1} \tag{3.4}$$

where S is the score, A_{all} is the estimated accuracy of the tree, F_t is the total number of features used by the tree as a whole, F_b is the total number of features being input to the base classifiers, N_l is the number of layers being used, $(C-1)$ is the number of base classifiers and the maximum number of layers in the tree, and the terms α_t, α_b and α_l are the penalty terms.

A distinction is made between the number of features used by the entire tree, and the total number of features used by the base classifiers. The term F_b totals the features used by the base classifiers and this penalizes against classifiers with a large number of features. The term F_t counts the number of features used by the base classifiers, but if more than one classifier uses the same feature, it is counted only once. This gives the total number of features that will need to be calculated to use the classifier. In this work, all features are weighted equally, but it also possible to adjust this formula to penalize more heavily against features that are more computationally expensive.



(a) crossover in the feature portion on the genome

(b) crossover in the classifier portion on the genome

Figure 3.7: Crossover functions

Genetic algorithm evolutionary operators

Because a small change to the tree portion of the gene can result in a large change to how the tree is built, a simple one-point crossover function is not the ideal way to breed genes.

The crossover function randomly selects to either change the feature or the gene portion of the tree in each breeding. If the feature portion is selected, the algorithm performs a one point crossover in the feature portion, and randomly attaches each new feature portion to the second portion of each gene. If the classifier portion is selected, the algorithm performs a one point swap, as illustrated in Fig. 3.7b. Mutations can affect a point in either part of the genome.

3.1.2 Experiments

Two different sets of experiments are conducted with the FSHC. The first set of experiments uses a multi-class K-nearest neighbours (KNN) classifier as the base classifier and tests

Performance measure	flat or tree?	estimated on tree (method 1) or base classifiers (method 2)
all features	flat	N/A
validation set	flat	N/A
validation set	tree	entire tree
cross-validation	tree	base classifiers
Fisher's	tree	base classifiers
Fisher's (scaled)	tree	base classifiers
RELIEF	tree	base classifiers
count-based RELIEF	tree	base classifiers

Table 3.1: FSHC accuracy estimation measures

Class	Feature 0	Feature 1	Feature 2	Feature 3	Feature 4
0	$0 + x\%$	$0.05 + x\%$	$1 - x\%$	noise	noise
1	$0.05 + x\%$	$0 + x\%$	$1 - x\%$	noise	noise
2	noise	noise	$0 + x\%$	$0 + x\%$	$0.05 + x\%$
3	noise	noise	$0 + x\%$	$0.05 + x\%$	$0 + x\%$

Table 3.2: Description of *FSHC artificial data set 1*, where x is the% of noise (0% or 5%)

the ability of the algorithm to design trees with multiple outputs. The second set of experiments uses a binary SVM and compares the algorithm to other multi-class extensions for binary classifiers.

Multi-class experiments with KNN

The first set of experiments use FSHC with KNN base classifiers. The algorithm is tested on two different artificial data sets using a variety of different accuracy estimation measures for scoring. The FSHC tree structures are compared against a flat classifier, which is single base classifier that separates all of the classes. Feature selection for the flat classifiers is also performed using genetic algorithms. A summary of all the tests is given in Table 3.1.

The *FSHC artificial data sets* are designed to test the ability of the algorithm to select an appropriate tree or flat classifier. Neither data set is fully separable by a single level (flat) KNN classifier.

FSHC artificial data set 1 tests the ability of the system to find an appropriate tree structure. It is a four class data set, with five partially informative features. Feature two separates classes zero and one from classes two and three. Features zero and one separate classes zero and one and are noise otherwise. Features three and four separate classes two and three, and are noise otherwise. A second version of the data set includes 5% uniform, random noise. The data set is described in Table 3.2.

	Feature 0	Feature 1	Feature 2	Feature 3
Class 0	$0 \pm x\%$	noise	noise	noise
Class 1	$0.25 \pm x\%$	noise	noise	noise
Class 2	$0.5 \pm x\%$	noise	noise	noise
Class 3	$0.75 \pm x\%$	noise	noise	noise

Table 3.3: Description of *FSHC artificial data set 2* with noise, where x is the amount of noise (0%, 5% or 20%)

FSHC artificial data set 2 tests the ability of the system to detect that a single level classifier is sufficient, and also tests how it performs in the presence of noise. It is a four class problem with four features. The first feature separates the classes fully and the remaining features are noise. This data set is also tested with 5% and 20% noise on the informative feature. With 20% noise, the data set is not fully separable even when using only feature zero. The data set is described in Table 3.3.

The data sets are divided randomly into test and training sets and all results are reported on the test sets. When validation sets are used for scoring, the validation portion is 20% of the data set, with 60% of the data set used for training, and 20% used for testing. In all other cases, 80% is used for training and 20% is used for testing.

The genetic algorithm is coded in C++, and the performance measures are evaluated in MATLAB by calling the MATLAB engine from the C++ program. The parameters are set such that the initial population is 500 and 50 individuals are replaced every generation. The algorithm is allowed to run for 70 generations.

Binary experiments with SVM

The second set of experiments tests FSHC against several well known techniques for multi-class extension, described in Section 2.5. For the one vs. rest, fuzzy one vs. rest, one vs. one and DAG-SVM, the extensions are tested using both the full feature set, and using a genetic algorithm to select features for each individual classifier. The binary tree method (SVM-BDT) [103], the adaptive binary tree (ABT) [27], and the classification and regression tree (CART) [19] are also tested, using the entire feature set.

These methods are tested on nine real data sets from the UCI machine learning repository [8]. They are also tested on *FSHC artificial data set 1*, described in Table 3.2, which is designed so that the different classes are separable by different features.

The real data sets tested are *abalone*, *covertype*, *ecoli*, *flag*, *glass*, *iris*, *image segmentation*, *statlog (vehicle)* [114] and *wine*. The original *covertype* database is very large (581012

Data Set	F	C	N	Description
Abalone	8	30	4177	Predict age of abalone from physical measurements
Covertype	56	7	7000	Predict forest cover type from information about region
Ecoli	7	8	336	Predict localization site from tests
Flag	28	8	194	Predict country's major religion from its flag attributes
Glass	9	7	214	Predict glass type from oxide content
Iris	4	3	150	Predict iris type from measurements
Image Segmentation	18	7	2310	Predict texture type from high level image attributes
Statlog (vehicle) [114]	18	4	946	Predict vehicle type from image data
Wine	13	3	178	Predict wine origin from chemical analysis

Table 3.4: Description of real data sets used for testing FSHC algorithm. F gives the number of features, C gives the number of classes, N gives the number of samples.

samples), and hence the database used in this work is shortened, and uses only 1000 samples from each of the 7 classes, for a total of 7000 samples. This is done to reduce the amount of time required for training and evaluating the classifiers, since a wrapper method is being used for evaluation.

The real data sets used in this study are described in detail in Table 3.4. The methods are also tested on versions of these data sets that include an extra copy of the real features and an additional four features that are just noise, to test the feature selection capability of the algorithm.

In all cases, the features of the data are individually normalized to lie in the range of [0,1]. The data set is divided such that 20% of the data is used for testing, 20% of the data is used as a validation set for scoring and the remainder is used for training. The reported accuracy results are for accuracy on the testing set.

The genome employed for feature selection for the one vs. rest, fuzzy one vs. rest, one vs. one and DAG-SVM uses one gene for each feature for each individual classifier in the set. Each of these genes take a value of either zero or one, to signify whether that particular feature is used in that individual classifier. For the one vs. rest and fuzzy one vs. rest cases, this gives a genome of size FC, and for the one vs. one and DAG-SVM cases, the genome size is $FC(C-1)/2$. This is similar to the feature selection portion of the hierarchical classifier genome, as described in Section 3.1.1.

The scoring for the feature selection is based on an estimate of accuracy of the full classifier, found using a validation set. In all cases, there are penalties for the total number of features used by each of the individual classifiers, and for the number of features used by the entire system. This is similar to the penalties applied to the hierarchical classifier design, as described in Section 3.1.1.

In all cases, the genetic algorithm is initialized with a random population of 500. For the feature selected hierarchical classifier, the randomly generated genomes are reselected if they do not generate a valid tree. At each generation, 50 solutions are replaced and bred, and the mutation rate on the newly generated solutions is 10%. The maximum number of epochs is set to 70, and the genetic algorithm will stop early if it has run for more than 20 epochs, and the total score for all individuals in the last two epochs is within 0.5% of the total for the current run. The penalty terms in the score (α_l, α_b, α_t) are all set to a very low value (0.005), to prioritize accuracy, but to give a very small preference to classifiers that use fewer numbers of features if the accuracies are the same.

All of the SVMs use linear kernels. Although non-linear kernels can achieve better accuracy on many data sets [21], all non-linear kernels have parameters that need to be tuned. Poorly tuned parameters can affect results, and it is difficult to determine if differences in performance are the result of the different multi-class extension, or just differences in parameter tuning. Hence, a linear kernel is used for these experiments. Linear SVMs also have an advantage over kernel SVMs in terms of testing time. For linear SVM, the class of a point can be determined by projecting the point onto the normal of the separating hyperplane and the testing time is therefore proportional to the number of input features only and not the number of support vectors.

The ABT and decision tree code is written in Matlab. The code for the remaining tests is written in C++. The SVM implementation is taken from the SVM Lite implementation by Joachims [73].

Timing is measured using the standard C internal timer (clock()). The timer for the entire genetic algorithm is started right before the genetic algorithm is started, and stopped before the results are printed to file. Timing values for training the classifiers are also given. A timer is started before a base classifier is trained, but after the training data set is prepared. The timer is stopped when the training is complete and the timing value is recorded. The total classifier training time is the sum of the training times of the base classifiers. This gives an indication of how difficult it is to train the classifier.

The accuracy results shown are based on the average accuracy on the test set of the top 10 best scoring trees, averaged over at least three runs.

The number of base classifiers used for testing is also reported. For the one vs. rest, fuzzy one vs. rest, one vs. one and DAG-SVM, the same number of base classifiers have to be queried to classify each point, regardless of the class distribution. For the tree-based solutions, the number of base classifiers that need to be queried differs depending on the tree design and on the distribution of the classes. The number of queries required is estimated empirically by building each tree, and finding the number of base classifiers required to

classify each class, as illustrated in Fig. 3.8. The number of classifiers is estimated in two ways. First, by assuming that the number of points in each class is balanced and second by assuming that the class distribution in the data set is representative. For a balanced class distribution, the number of queries required is given as:

$$Q_{bal} = \frac{1}{C} \sum_{c=1}^{C} Q_c \tag{3.5}$$

where Q_{bal} is the number of queries required, assuming a balanced number of classes, C is the number of classes, and Q_c is the number of queries required to classify class c.

If the class balance is the same as the data set, the number of queries is estimated as

$$Q_{rep} = \frac{1}{N} \sum_{c=1}^{C} N_c Q_c \tag{3.6}$$

where Q_{rep} is the number of queries required, assuming the class balance in the data set, N is the number of samples in the data set and N_c is the number of samples from class c.

The results from these tests as well as the theoretical bounds are presented in 3.1.3.

The tests are all run on the Shared Hierarchical Academic Research Computing Network (SHARCNET) clusters.

3.1.3 Results and Discussion

Results from tests with the multi-class KNN base classifier and binary SVM classifier are presented in this section.

Multi-class KNN

Results for *FSHC artificial data set 1* are presented in Table 3.5. The hierarchical structures are shown in Fig. 3.9 and 3.10. The accuracy of a flat classifier with no feature selection is fairly low for this data set. The distance separating different classes is relatively low compared to the noise, which is difficult for a KNN classifier. Feature selection alone raises the accuracy significantly for the data set without noise. For the data set with noise, the gain of feature selection is not as significant.

The hierarchical classifiers are able to give the best accuracy. However, the different accuracy estimate methods produce quite different results. The validation set appears to

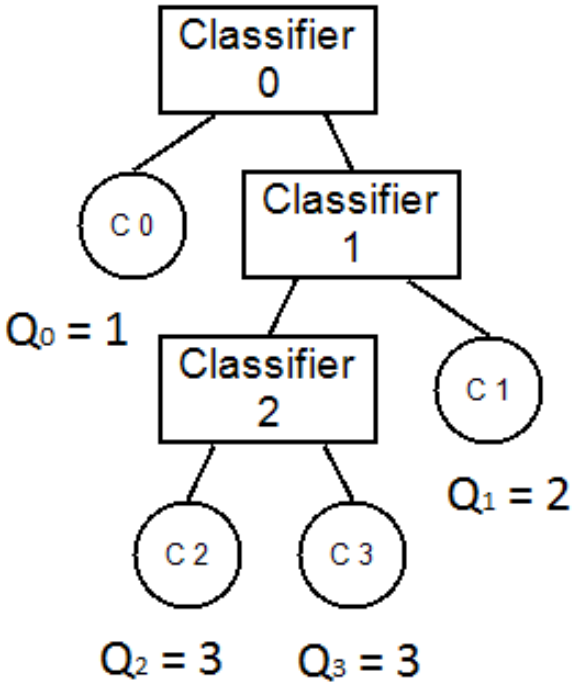
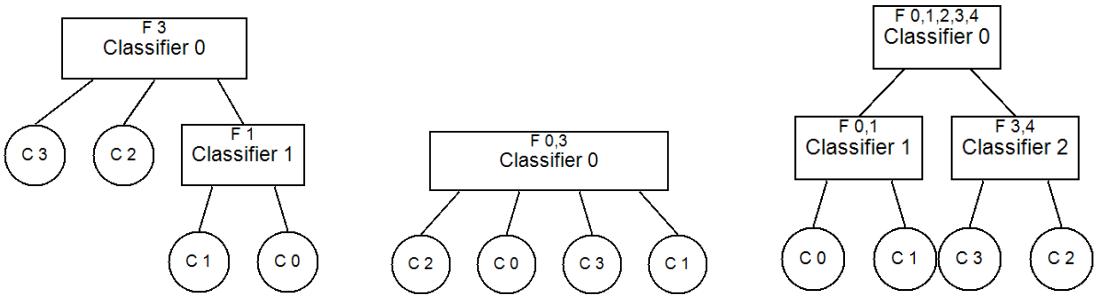


Figure 3.8: Example classification tree, with the number of queries required to classify each class, where Q_c is the number of classifiers required to be queried to classify a point in class c.

	Accuracy		# features (no repeats)		# features (repeats)		# base classifiers	
Test	0%	5%	0%	5%	0%	5%	0%	5%
All features	0.68	0.68	5	5	5	5	1	1
Features only	0.96	0.87	2	2	2	2	1	1
Validation set	1.00	1.00	2	3	2	3	2	2
Cross-validation	0.99	1.00	2	3	2	3	1	2
RELIEF	1.00	0.85	5	5	9	7	3	2
Fisher's	1.00	0.81	5	5	9	7	3	2
Fisher's (scaled)	1.00	0.57	5	1	5	1	3	1
Count RELIEF	1.00	1.00	3	5	3	5	2	2

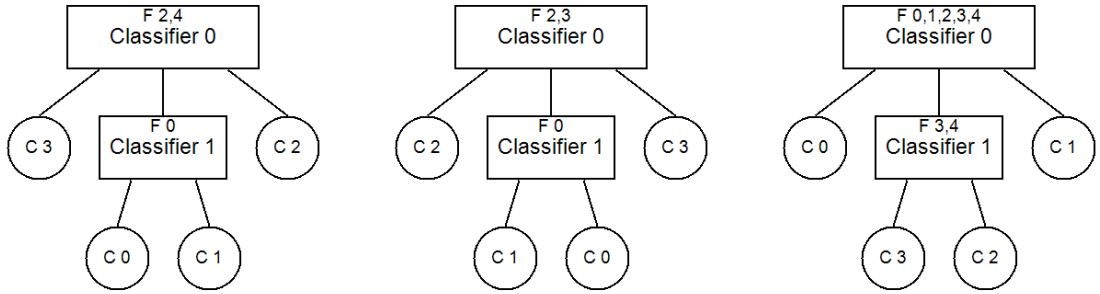
Table 3.5: Results for *FSHC artificial data set 1* with 0% and 5% noise. The column "# features (no repeats)" gives the total number of features used in the hierarchical classifier (F_t). If the same feature is used by more than one base classifier, it is counted only once in this column. The column "# features (repeats)" sums the total number of features used by all the base classifiers (F_b). If a feature is used by more than one base classifier, it is counted each time it is used.



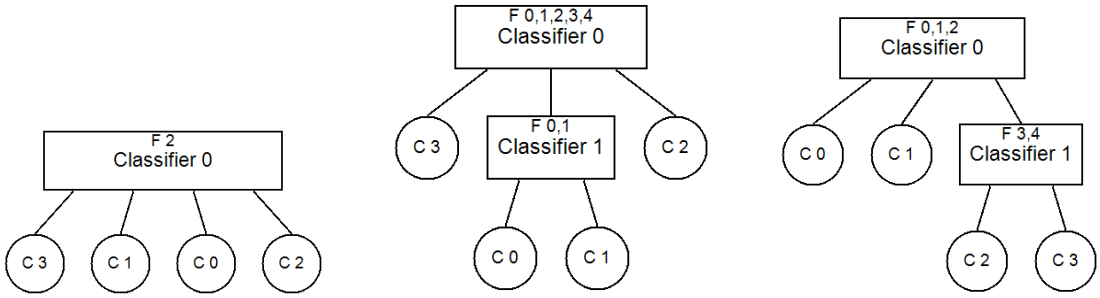
(a) validation set (b) cross-validation (c) Fisher's unscaled

(d) Fisher's scaled (e) RELIEF (f) count-based RELIEF

Figure 3.9: Hierarchical classifiers generated for *FSHC artificial data set 1* with no noise



(a) validation set (b) cross-validation (c) Fisher's unscaled



(d) Fisher's scaled (e) RELIEF (f) count-based RELIEF

Figure 3.10: Hierarchical classifiers generated for *FSHC artificial data set 1* with 5% uniform random noise

produce good results. In the data set with no noise, the classifier generated using cross-validation collapses down to the same single-level classifier generated with feature selection alone.

For the data set with no noise, scaled Fisher performs quite well, and selects the same tree as was originally envisioned (see Fig. 3.9d). Unfortunately, it does not perform nearly as well in the data set with additive noise, selecting only a single feature and a flat classifier. It is likely that the scaling is disproportionately large and the accuracy gain from adding additional features is outweighed by the scaling factor. Scaling Fisher's to the maximum possible size is clearly not a good way to discourage the use of additional features. It is likely better to simply re-tune the scoring parameters. However, this does mean that the algorithm will need to be re-tuned for different data sets to give good results.

The unscaled version of Fisher's criterion selects a similar tree to the scaled version for the noise-free data set, but selects all five features for the top level of the tree (see Fig. 3.9c). Although the tree is able to attain 100% accuracy on the test set, the selected tree is not ideal, as it uses a large number of features and also has an increased chance of a mis-classified point in the top level of the tree. In the data set with noise, the tree selected is similar to the tree selected by RELIEF. In both cases, the tree is reasonable, separating two classes in the top level and the other two classes in a lower level (see Fig. 3.10c and 3.10e). However, in both cases more features than necessary are selected in the top level of the tree.

The count-based RELIEF measure performs well on both versions of this data set, creating the same tree as the validation set in the non-noisy data set, and adding two additional features in the noisy data set that increase the robustness of the classifier. (see Fig. 3.9f and 3.10f). In fact, using a validation set or count-based RELIEF, the algorithm was able to obtain a more compact classifier than was originally envisioned (see Fig. 3.9d). These classifiers separate two classes completely in the first layer and use only two base classifiers total, as pictured in Fig. 3.9a and 3.9f. This tree uses both fewer base classifiers and fewer features in the noise-free case, and would be faster on average.

The results for data set two are presented in Table 3.6.

Fisher's criterion is able to find the correct single-level tree for the data sets with 5% and 20% noise. However, for the noiseless data set, Fisher's separates the outputs into three base classifiers that all use the same feature. For the 5% and noiseless data sets, RELIEF also splits the classifier into two or three base classifiers, all using the same features. It is unable to find the correct feature for the 20% data set. The count-based RELIEF measure is able to find the appropriate one level classifier for the separable data sets, but splits the 20% noise set into a two level tree.

Test	Accuracy			# features (no repeats)			# features (repeats)			# base classifiers			correct feature?		
	0%	5%	20%	0%	5%	20%	0%	5%	20%	0%	5%	20%	0%	5%	20%
All features	0.92	0.93	0.62	4	4	4	4	4	4	1	1	1		N/A	
Features only	1.00	1.00	0.74	1	1	1	1	1	1	1	1	1	Y	Y	Y
Validation set	1.00	1.00	0.58	1	1	4	1	1	5	1	1	2	Y	Y	N
Cross-validation	1.00	1.00	0.66	1	1	2	1	1	3	1	1	2	Y	Y	N
RELIEF	1.00	1.00	0.74	1	1	2	3	2	3	3	2	2	Y	Y	N
Fisher's	1.00	1.00	0.74	1	1	1	3	1	1	3	1	1	Y	Y	Y
Fisher's (scaled)	1.00	1.00	0.70	1	1	1	3	1	1	3	1	1	Y	Y	Y
Count RELIEF	1.00	1.00	0.78	1	1	1	1	1	2	1	1	2	Y	Y	Y

Table 3.6: Results for *FSHC artificial data set 2* with 0%, 5% and 20% noise on feature 0. The column "# features (no repeats)" gives the total number of features used in the hierarchical classifier (F_t). If the same feature is used by more than one base classifier, it is counted only once in this column. The column "# features (repeats)" sums the total number of features used by all the base classifiers (F_b). If a feature is used by more than one base classifier, it is counted each time it is used.

Since the only feature used in all these trees is feature 0, the division is unnecessary for the 1-NN classifier. However, for a classifier with a finite capacity, splitting the data set in this manner may actually provide a benefit, by allocating more resources to classes that are more difficult to classify.

In all the tests, the choice of performance measure affects the tree and feature set selected. This indicates the importance of selecting an appropriate performance measure.

Binary SVM

Results from tests using different multi-class extension methods are shown in Tables 3.7, 3.10, 3.11 and 3.12. The methods are compared on the basis of accuracy, memory footprint, training and testing time.

Accuracy Table 3.7 shows how the different methods compare in terms of accuracy. Results are averaged over at least three runs. For tests that use feature selection, accuracy for each run is averaged over the best 10 trees. The best accuracy is the best individual solution over all runs.

It does not appear that there is one single method that is the best choice for all cases. The only method that does not generate good results is the one vs. rest formulation with no tie-breaking mechanism. The only case where this method generates moderately good

		abalone				covertype				ecoli			
		accuracy				accuracy				accuracy			
method	FS?	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall
one vs. rest	no	0.00	0.00	0.00	0.00	0.34	0.34	0.61	0.64	0.63	0.63	0.74	0.71
	yes	0.10	0.14	0.02	0.00	0.42	0.47	0.61	0.64	0.55	0.67	0.78	0.80
fuzzy OVR	no	0.13	0.13	0.11	0.16	0.63	0.63	0.63	0.63	0.81	0.81	0.78	0.82
	yes	0.20	0.21	0.13	0.14	0.60	0.62	0.63	0.61	0.79	**0.85**	0.72	0.70
one vs. one	no	0.20	0.20	**0.26**	0.17	0.65	0.65	0.65	0.66	0.82	0.82	0.79	0.68
	yes	0.18	0.23	0.25	0.17	0.66	0.66	0.67	0.69	0.80	0.84	0.72	0.63
DAG-SVM	no	**0.24**	0.24	0.24	0.17	0.65	0.65	0.62	0.64	0.77	0.77	0.77	0.72
	yes	0.11	0.17	0.24	0.16	0.58	0.66	0.59	0.61	0.74	0.82	0.74	0.76
SVM-BDT	no	0.19	0.19	0.18	0.11	0.63	0.63	0.61	0.63	0.79	0.79	0.75	0.68
ABT	no	**N/A**	**N/A**	**N/A**	**N/A**	0.71	0.71	0.71	0.71	**0.84**	0.84	**0.85**	**0.84**
CART	no	0.21	0.21	0.20	**0.21**	**0.79**	**0.79**	**0.79**	**0.79**	0.78	0.78	0.77	0.78
FSHC	yes	0.21	**0.25**	0.21	0.20	0.65	0.67	0.62	0.62	0.71	0.82	0.78	0.80
		flag				glass				iris			
		accuracy				accuracy				accuracy			
method	FS?	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall
one vs. rest	no	0.11	0.11	0.52	0.42	0.12	0.12	0.52	0.38	0.63	0.63	0.61	0.61
	yes	0.24	0.32	0.43	0.42	0.44	0.45	0.52	0.46	0.55	0.70	0.82	0.98
fuzzy OVR	no	0.42	0.42	0.42	0.40	0.49	0.49	0.54	0.51	0.84	0.84	0.79	0.89
	yes	0.40	**0.55**	0.44	0.45	0.57	0.62	0.50	0.50	0.96	0.97	0.75	0.86
one vs. one	no	0.45	0.45	0.36	0.36	0.52	0.52	0.49	0.45	0.96	0.96	0.98	0.97
	yes	0.39	0.45	0.38	0.42	0.56	**0.71**	0.48	0.42	0.95	**1.00**	0.83	0.86
DAG-SVM	no	0.45	0.45	0.41	0.38	0.54	0.54	0.57	0.47	0.96	0.96	0.95	0.94
	yes	0.40	0.53	0.41	0.44	0.50	0.55	0.50	0.46	0.92	**1.00**	**0.98**	**0.98**
SVM-BDT	no	0.39	0.39	0.39	0.35	0.48	0.48	0.43	0.33	**0.97**	0.97	0.96	0.96
ABT	no	0.44	0.44	0.45	0.44	0.52	0.52	0.60	0.54	0.97	0.97	0.97	0.97
CART	no	**0.54**	0.54	**0.53**	**0.53**	**0.63**	0.63	**0.62**	**0.63**	0.93	0.93	0.94	0.93
FSHC	yes	0.38	0.50	0.38	0.41	0.58	0.64	0.57	0.49	0.95	**1.00**	**0.98**	**0.98**
		image segm.				statlog(vehicle)				wine			
		accuracy				accuracy				accuracy			
method	FS?	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall	(avg)	(best)	precision	recall
one vs. rest	no	0.74	0.74	0.79	0.81	0.28	0.28	0.50	0.45	0.94	0.94	0.99	**0.99**
	yes	0.72	0.75	0.79	0.81	0.36	0.41	0.46	0.42	0.94	0.94	**0.99**	0.99
fuzzy OVR	no	0.87	0.87	0.87	0.89	0.62	0.62	0.61	0.64	**0.98**	0.98	0.96	0.94
	yes	0.91	0.92	0.85	0.86	0.68	0.72	0.60	0.59	0.97	**1.00**	0.97	0.97
one vs. one	no	0.90	0.90	0.91	0.92	0.68	0.68	0.68	0.66	0.97	0.97	0.92	0.96
	yes	0.91	0.95	0.91	0.91	0.67	0.75	0.66	0.66	0.92	0.97	0.95	0.95
DAG-SVM	no	0.91	0.91	0.93	0.93	0.69	0.69	0.69	0.67	0.97	0.97	0.92	0.96
	yes	0.89	0.93	0.90	0.90	0.70	0.75	0.68	0.69	0.92	0.97	0.92	0.94
SVM-BDT	no	0.91	0.91	0.91	0.91	0.65	0.65	0.66	0.71	0.97	0.97	0.94	0.95
ABT	no	**0.95**	**0.95**	**0.94**	**0.95**	**0.80**	**0.80**	**0.80**	**0.80**	0.94	0.94	0.95	0.94
CART	no	0.94	0.94	0.94	0.94	0.65	0.65	0.65	0.65	0.83	0.83	0.83	0.83
FSHC	yes	0.90	0.92	0.91	0.91	0.66	0.75	0.67	0.66	0.90	0.94	0.90	0.92

Table continued on page 74

Table 3.7: Accuracy of FSHC and common multi-class extensions on different data sets. FS column indicates if feature selection is used. The average accuracy gives the average of the 10 best scoring trees from each genetic algorithm, averaged over at least three runs. The best accuracy is the best individual solution over all runs. Bold values indicate the highest accuracy in a column.

Continued from Table 3.7 on page 73					
		artificial			
		accuracy			
method	FS?	(avg)	(best)	precision	recall
one vs. rest	no	0.42	0.42	0.55	0.42
	yes	0.44	0.57	0.49	0.43
fuzzy OVR	no	0.61	0.61	0.62	0.63
	yes	0.78	0.96	0.69	0.72
one vs. one	no	0.63	0.63	0.63	0.63
	yes	0.85	0.99	0.97	0.98
DAG-SVM	no	0.59	0.59	0.65	0.53
	yes	0.92	**1.00**	0.99	0.99
SVM-BDT	no	0.62	0.62	0.60	0.60
ABT	no	**1.00**	1.00	1.00	**1.00**
CART	no	0.99	0.99	0.99	0.99
FSHC	yes	0.97	**1.00**	**1.00**	**1.00**

Table 3.7 (cont.): Accuracy of FSHC and common multi-class extensions on different data sets. FS column indicates if feature selection is used. The average accuracy gives the average of the 10 best scoring trees from each genetic algorithm, averaged over at least three runs. The best accuracy is the best individual solution over all runs. Bold values indicate the highest accuracy in a column.

results is on the *wine* data set, indicating that the classes are likely very separated and the data is quite well behaved. In all the other cases, the other methods perform better.

It is interesting to note that the much simpler non-SVM-based decision tree classifier is able to outperform the more complex SVM-based classifiers on several data sets. ABT also performs well on some of the data sets. These two hierarchical classifiers differ from the SVM-BDT and FSHC in that they allow classes to be classified as being on both sides of some nodes. This improves accuracy, but creates a tree with a larger number of nodes, which increases the memory footprint as well as the test time. For ABT, each node is based on a classifier that is trained on one pair of classes. If none of the other classes are well separated by any node, only one class is eliminated at each level. The maximum number of nodes in the tree is $2^{(C-1)}$, which is a problem for data sets with a large number of classes, such as the 30-class *abalone* data set.

FSHC has the highest best-run accuracy on three of the tested data sets, indicating that this method can perform well if the genetic algorithm run is evaluated carefully. Feature selection helps for all multi-class extensions. The best-run feature selected accuracy is higher than the non-feature-selected accuracy in 39 out of 45 real tests and in all the artificial tests. Feature selection also creates classifiers that have a smaller memory footprint and are faster during the testing phase. Feature selection is the least successful for the one vs. one and DAG-SVM extensions where in three cases the best-run accuracy is actually

lower than the un-feature selected case, and an additional three where the difference is less than 1%. Wrapper methods can select a set of features that works well for the training set, but not do generalizing well [91]. Both the one vs. one and DAG-SVM train classifiers on each pair of classes. The number of samples for each classifier is therefore relatively small, and this increases the possibility of selecting a set of features that does not generalize well to unseen data.

It is interesting to note that the performance of the fuzzy one vs. rest method is much better than the one vs. rest with no tie-breaking mechanism. This means that in the one vs. rest formulation, there are a significant number of points that are being misclassified by the individual base classifiers in the system. Similarly, it is also possible for the one vs. one method to recover the correct classification if a point is misclassified by one of the base classifiers. The tree-based solutions cannot do this. If a point is misclassified by a single base classifier, then the point will be misclassified.

The variance between different runs is quite high. Looking at only the best run generates accuracies that are much higher, indicating either that the genetic algorithm is not converging to an appropriate minimum, or that the selected features do not generalize well. In particular for the *flag* and *glass* data sets, the difference between the average and best runs for almost all the methods is quite significant, indicating that there may be a large amount of variance in the data set. This makes intuitive sense for the *flag* data set, where the objective is to determine the religion of a country based on characteristics of its flag. While there are likely to be some similarities between countries of similar religions, there are also likely to be many outliers.

In order to determine if the large variance is a result of poor initialization, or an unrepresentative validation set, new tests are run using the same initial population or the separation of the data set over 10 runs of the algorithm. These tests are run on the *flag* and *glass* data sets, since these data sets exhibit a large amount of variance. The results from these tests are presented in Table 3.8.

In these tests, the accuracy appears to be lower than average when the test set is held constant. This most likely occurs because the selected test set is more difficult than the average. If the test set contains a larger number of difficult to classify points, the accuracy will be lower.

Although there is a slight decrease in variance when the test set and initialization are held constant, the variance is still quite high. While using different test sets and initializations does appear to contribute to the variance between runs, there are also likely other sources of this variation. It may be that the distribution of the validation set and test set are very different, and while different configurations give similar results on the

method	FS?	parameter held		flag		glass	
		init.	data set	avg	std	avg	std
fuzzy OVR	yes	yes	yes	0.42	0.05	0.42	0.06
	yes	yes	no	0.38	0.06	0.50	0.12
	yes	no	yes	0.35	0.07	0.36	0.10
	yes	no	no	0.42	0.06	0.50	0.15
one vs. one	yes	yes	yes	0.42	0.04	0.51	0.05
	yes	yes	no	0.38	0.12	0.54	0.12
	yes	no	yes	0.41	0.06	0.45	0.11
	yes	no	no	0.38	0.09	0.57	0.06
DAG-SVM	yes	yes	yes	0.37	0.04	0.42	0.08
	yes	yes	no	0.36	0.05	0.46	0.08
	yes	no	yes	0.38	0.05	0.38	0.07
	yes	no	no	0.38	0.07	0.46	0.08
FSHC	yes	yes	yes	0.34	0.04	0.38	0.05
	yes	yes	no	0.36	0.07	0.49	0.07
	yes	no	yes	0.34	0.04	0.41	0.12
	yes	no	no	0.36	0.07	0.47	0.08

Table 3.8: Accuracy of FSHC across different runs while holding initialization or data set division constant

validation set, the results are quite different on the test set.

In order to test this hypothesis, a new test was run using leave-one-out cross validation on the training set instead of a validation set as an estimate of the accuracy for the scoring. This gives a larger number of points on which to base the estimate of the accuracy. However, this requires that a new model be trained for each point in the training set. Similar to the validation set, the results from previous tests are saved and re-used when possible, but the time required for training is still significantly longer than the validation set. The results from these tests are given in Table 3.9. In all tests, the initialization and data set division are the same for all runs.

The variation between the tests is lower when using leave-one-out cross validation, but it is still fairly high, indicating that even the expanded test does not fully capture the behavior of the test set. The accuracy values themselves are not significantly improved. Hence, using a leave-one-out cross validation gives results that are slightly more robust, but at the cost of a much longer training time, with little improvement in accuracy.

Memory footprint The total memory footprint of the classifiers after training is dependent on the number of classifiers that are trained, and also on the number of features that each classifier uses. For a non-linear SVM, the memory footprint is also dependent on the number of support vectors. However, for a linear classifier only the normal of the hyperplane needs to be stored, meaning there is a single vector for each base classifier. The

		flag	
method	FS?	acc. (avg)	acc. (std)
fuzzy OVR	yes	0.42	0.04
one vs. one	yes	0.40	0.02
DAG-SVM	yes	0.39	0.02
FSHC	yes	0.35	0.03

Table 3.9: Average and standard deviation of FSHC accuracies for ten runs using leave-one-out cross validation to estimate accuracy, using the same data set division and initialization parameters for all tests

total number of features, base classifiers and the total memory footprint for all classifiers is presented in Table 3.10.

As expected, in most cases the classifiers with the largest memory footprint are the one vs. one and the DAG-SVM, due to the larger number of base classifiers required. As the number of classes becomes smaller, this difference becomes less pronounced, and with only three classes, the number of classifiers required for a one vs. rest and one vs. one classification are the same. Using a smaller number of classifiers and using feature selection reduces the memory footprint.

Training time The training times for the feature selection algorithms are presented in Table 3.11. T_{mc} gives the average time to train all the base classifiers in one multi-class classifier. T_{GA} gives the time required to run the entire genetic algorithm. The Epochs column indicates the number of epochs that the genetic algorithm required to converge.

In most cases, the time required to train the final classifiers is negligible. The time required to run the full genetic algorithm whether for feature selection or the full feature-selected hierarchical classifier, is similar in most cases. The time required seems to be more dependent on the data set than on the extension method.

In some cases, there are outliers that greatly affect the average time required. The variance between training times in some cases is quite high, and time required for some training runs appears to be dominated in individual classifiers that are difficult to train and therefore account for a disproportionately large amount of the training time. This tends to occur more often with the one vs. one and DAG-SVM and is likely because there are a larger number of classifiers being trained. The time required for the FSHC method described in this paper appears to be comparable to using any of the other methods and using a genetic algorithm for feature selection.

		abalone			covertype			ecoli		
method	FS?	F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base
one vs. rest	no	**8.0**	240.0	30.0	56.0	392.0	7.0	7.0	56.0	8.0
	yes	**8.0**	**106.8**	30.0	55.7	208.3	7.0	**6.3**	27.4	8.0
fuzzy OVR	no	**8.0**	240.0	30.0	56.0	392.0	7.0	7.0	56.0	8.0
	yes	**8.0**	110.3	30.0	**53.7**	197.9	7.0	7.0	24.7	8.0
one vs. one	no	**8.0**	3480.0	435.0	56.0	1176.0	21.0	7.0	196.0	28.0
	yes	**8.0**	1725.0	435.0	56.0	603.0	21.0	7.0	87.7	28.0
DAG-SVM	no	**8.0**	3480.0	435.0	56.0	1176.0	21.0	7.0	196.0	28.0
	yes	**8.0**	1748.5	435.0	56.0	584.0	21.0	7.0	100.2	28.0
ABT	no	N/A	N/A	N/A	56.0	1332.8(851.2)	23.8(15.2)	7.0	337.4(147.0)	48.2(21)
SVM-BDT	no	**8.0**	232.0	**29.0**	56.0	336.0	**6.0**	7.0	49.0	**7.0**
FSHC	yes	**8.0**	116.1	**29.0**	55.7	**167.3**	**6.0**	7.0	**24.2**	**7.0**
		flag			glass			iris		
		F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base
one vs. rest	no	28.0	224.0	8.0	9.0	63.0	7.0	4.0	12.0	3.0
	yes	**27.0**	96.9	8.0	**7.7**	**22.3**	7.0	2.0	3.0	3.0
fuzzy OVR	no	28.0	224.0	8.0	9.0	63.0	7.0	4.0	12.0	3.0
	yes	27.7	101.6	8.0	8.7	25.7	7.0	4.0	6.0	3.0
one vs. one	no	28.0	784.0	28.0	9.0	189.0	21.0	4.0	12.0	3.0
	yes	28.0	364.6	28.0	9.0	79.6	21.0	2.2	3.8	3.0
DAG-SVM	no	28.0	784.0	28.0	9.0	189.0	21.0	4.0	12.0	3.0
	yes	28.0	369.8	28.0	9.0	86.1	21.0	2.6	4.8	3.0
ABT	no	28.0	3511.2(778.4)	125.4(27.8)	9.0	169.2(118.8)	18.8(13.2)	4.0	8.0(8.0)	2.0(2.0)
SVM-BDT	no	28.0	196.0	**7.0**	9.0	54.0	**6.0**	4.0	8.0	**2.0**
FSHC	yes	27.1	**87.5**	**7.0**	8.5	29.0	**6.0**	**1.8**	**2.5**	**2.0**
		image segm.			statlog (vehicle)			wine		
		F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base	F_{all}	F_{tot}	# base
one vs. rest	no	**18.0**	126.0	7.0	18.0	72.0	4.0	13.0	39.0	3.0
	yes	**18.0**	65.1	7.0	16.3	34.3	4.0	9.3	18.9	3.0
fuzzy OVR	no	**18.0**	126.0	7.0	18.0	72.0	4.0	13.0	39.0	3.0
	yes	**18.0**	66.2	7.0	17.3	34.7	4.0	7.5	12.8	3.0
one vs. one	no	**18.0**	378.0	21.0	18.0	108.0	6.0	13.0	39.0	3.0
	yes	**18.0**	191.1	21.0	17.4	55.6	6.0	7.2	12.2	3.0
DAG-SVM	no	**18.0**	378.0	21.0	18.0	108.0	6.0	13.0	39.0	3.0
	yes	**18.0**	180.5	21.0	17.7	59.3	6.0	7.2	12.2	3.0
ABT	no	**18.0**	450.0(248.4)	25.0(13.8)	18.0	122.4(104.4)	6.8(5.8)	13.0	39.0(39.0)	3.0(3.0)
SVM-BDT	no	**18.0**	108.0	**6.0**	18.0	54.0	**3.0**	13.0	26.0	**2.0**
FSHC	yes	**18.0**	**57.1**	**6.0**	**16.3**	**30.2**	**3.0**	**6.8**	**9.5**	**2.0**

Continued on page 79

Table 3.10: Number of features, base classifiers and memory footprint for FSHC and common multi-class extensions. F_{all} gives the overall number of features used by any of the base classifiers in the system. F_{tot} gives the total number of inputs to all the base classifiers. The column "# base" gives the number of base classifiers that need to be trained and stored for each method. For ABT, it is possible for the same classifier to appear at multiple points in the tree. For ABT the total number of classifiers in the tree is given in the "# base" column, and the number of different base classifiers that would need to be stored is given in parentheses.

Continued from Table 3.10 on page 78				
		artificial		
		F_{all}	F_{tot}	# base
one vs. rest	no	5.0	20.0	4.0
	yes	4.0	8.7	4.0
fuzzy OVR	no	5.0	20.0	4.0
	yes	4.5	7.5	4.0
one vs. one	no	5.0	30.0	6.0
	yes	4.3	8.3	6.0
DAG-SVM	no	5.0	30.0	6.0
	yes	4.3	9.0	6.0
ABT	no	5.0	15.0(15.0)	3.0(3.0)
SVM-BDT	no	5.0	15.0	**3.0**
FSHC	yes	**3.8**	**4.4**	**3.0**

Table 3.10 (cont.): Number of features,base classifiers and memory footprint for FSHC and common multi-class extension methods. F_{all} gives the overall number of features used by any of the base classifiers in the system. F_{tot} gives the total number of inputs to all the base classifiers. The column "# base" gives the number of base classifiers that need to be trained and stored for each method. For ABT, it is possible for the same classifier to appear at multiple points in the tree. For ABT the total number of classifiers in the tree is given in the "# base" column, and the number of different base classifiers that would need to be stored is given in parentheses.

Testing time The major advantage of a tree-based solution is the time required for testing. Because the kernel used is linear, testing a single point on a single base classifier consists of finding the distance to the normal of the separating hyperplane. This distance is used directly in the fuzzy one vs. rest formulation, and the others use the classification, which is found by taking the sign of the distance. For a kernel SVM, the test time is proportional to the number of support vectors, and these are also reported in Table 3.12. It is important to note that this implementation does not directly penalize solutions with larger numbers of support vectors. However, a penalty term could be added to the scoring function (Eq. 3.4) if a kernel SVM is being used as the base classifier.

Assuming a squared Euclidean distance, each distance measurement consists of F_b multiplies and $2F_b$ additions, where F_b is the number of features used in the base classifier. Hence the time required for the overall classification is dependent both on the number of features used by each individual classifier, and the number of base classifiers that need to be queried.

For the one vs. rest, fuzzy one vs. rest, one vs. one and DAG-SVM, the number of queries required is always the same for the same data set, regardless of the class balance. For the one vs. rest and fuzzy one vs. rest methods, all of the C trained classifiers need to

		abalone			covertype			ecoli			flag		
method	FS?	T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs
one vs. rest	yes	0.06	388	25.3	3.53	748	26.7	0.03	34	24.3	0.00	436	30.0
fuzzy OVR	yes	0.05	383	21.0	5.36	1310	25.3	0.00	35	22.3	0.05	78	31.8
one vs. one	yes	86.01	9010	21.3	1.07	676	25.0	0.03	1500	22.4	0.67	166	27.2
DAG-SVM	yes	0.10	10550	21.0	0.72	967	22.0	0.00	39	23.3	0.02	161	23.7
FSHC	yes	0.72	843	28.0	1.72	1070	25.7	0.05	36	34.0	0.01	112	54.7
		glass			iris			image segm.			statlog(vehicle)		
		T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs
one vs. rest	yes	0.00	39	23.3	2.40	37	21.0	0.06	388	25.3	0.06	84	22.7
fuzzy OVR	yes	0.00	37	23.3	0.00	29	21.0	0.05	383	21.0	0.07	83	25.7
one vs. one	yes	0.00	52	21.8	0.00	32	21.0	86.01	9010	21.3	0.24	1360	21.8
DAG-SVM	yes	0.00	52	22.0	6.40	30	21.0	0.10	10550	21.0	0.04	42	21.0
FSHC	yes	0.00	38	26.5	0.00	31	30.0	0.72	843	28.0	0.03	59	28.3
		wine			artificial								
		T_{mc}	T_{GA}	epochs	T_{mc}	T_{GA}	epochs						
one vs. rest	yes	0.01	34	22.0	15.4	n/a	21.0						
fuzzy OVR	yes	0.00	30	21.0	7.3	26700	21.4						
one vs. one	yes	0.01	336	21.0	28.0	n/a	21.0						
DAG-SVM	yes	0.03	63	21.2	7.9	46	21.0						
FSHC	yes	0.00	32	36.5	2.3	301	21.8						

Table 3.11: Training times for FSHC and common multi-class extensions. The column FS indicates if a genetic algorithm is used for feature selection. The column T_{mc} gives the average combined time required to train all the base classifiers in one multi-class classifier. The column T_{ga} gives the average time required to run the full genetic algorithm. The column "epochs" indicates the number of epochs that the genetic algorithm required before it hit a stopping condition.

queried. For the one vs. one method, all of the $C(C-1)/2$ classifiers need to be queried. The DAG-SVM requires $(C-1)$ queries, with each query eliminating a single class.

For the tree-based solutions, the number of queries required depends heavily on the structure of the tree. For a balanced class distribution, the smallest number of queries required occurs when the number of classes is a power of two, and the tree is perfectly balanced. In this case, the number of queries required is $\log_2(C)$. The largest number of queries occurs on a fully unbalanced tree, when one class is separated in each level. In this case, the number of queries required is:

$$Q_{bal-longTree} = \frac{1}{C}\left(\left(\sum_{c=1}^{C} c\right) - 1\right) \tag{3.7}$$

For an unbalanced set of classes, the number of queries can become quite skewed. If the tree is balanced, the number of queries is again $\log_2(C)$. However, if the tree is also heavily unbalanced, the average number of queries depends on where in the tree the largest classes are separated. The worst case occurs when the largest class is at the lowest point on a heavily unbalanced tree. In this case, the number of queries approaches (C-1), which is still fewer than the number required for a DAG-SVM. In the best case, if the largest class is separated at the top of the tree, the number of queries approaches one.

Because the number of queries required for the tree can vary, the number of queries is estimated empirically, as described in Section 3.1.2. The average number of features, queries and multiplications for each method are show in Table 3.12.

The time required for testing is also dependent on the time required to calculate any features. In some cases, there is almost no cost for acquiring features. For example, if the features are pixels in an image, the features are easy to extract once the image is obtained. Calculated features may each have an associated cost. For example, averaging over a number of pixels. Feature cost can also be affected by the features already included in the set. For example, if some of the features are frequency components in a certain range or bin, then the cost of adding the first bin is high, because it would require performing a fast Fourier transform on the data. The added cost of any further bins is lower, because most of the calculations are already being performed.

In the proposed FSHC algorithm, a penalty for the number of features that need to be calculated is included in the scoring function as α_t (see Equation 3.4), although the penalty is set to be very small for this test. In this case, the scoring function penalizes each feature equally. However, because the scoring function of the genetic algorithm is fairly flexible, it is also possible to use a more complex formulation, penalizing a different amount for using

		abalone							covertype						
method	FS?	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}
one vs. rest	no	8.0	30.0	30.0	240	240	7120	7120	56.0	7.0	7.0	392	392	8870	8870
	yes	**3.6**	30.0	30.0	106	106	6500	6500	29.8	7.0	7.0	208	208	9360	9360
fuzzy OVR	no	8.0	30.0	30.0	240	240	7100	7100	56.0	7.0	7.0	392	392	8910	8910
	yes	3.7	30.0	30.0	110	110	6220	6220	28.3	7.0	7.0	197	198	9430	9420
one vs. one	no	8.0	435.0	435.0	3480	3480	22600	22600	56.0	21.0	21.0	1180	1180	7700	7710
	yes	4.0	435.0	435.0	1725	1725	21700	21700	28.7	21.0	21.0	603	603	8110	8110
DAG-SVM	no	8.0	29.0	29.0	232	232	1500	1500	56.0	6.0	6.0	336	336	2190	2190
	yes	4.0	29.0	29.0	116	116	**1450**	**1450**	**27.8**	6.0	6.0	166	166	2530	2530
ABT	no	N/A	N/A	N/A	N/A	N/A	N/A	N/A	56.0	4.4	4.4	248	248	**1000**	**1000**
SVM-BDT	no	8.0	6.7	6.7	53	53	2670	3610	56.0	**3.0**	**3.0**	168	168	2170	2170
FSHC	yes	4.0	**5.3**	**5.3**	**21**	**21**	4320	4600	27.9	3.2	3.2	**88**	**88**	2840	2840
		ecoli							flag						
		F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}
one vs. rest	no	7.0	8.0	8.0	56	56	405	405	28.0	8.0	8.0	224	224	377	377
	yes	3.4	8.0	8.0	27	27	392	392	**12.1**	8.0	8.0	96	97	373	373
fuzzy OVR	no	7.0	8.0	8.0	56	56	409	408	28.0	8.0	8.0	224	224	385	385
	yes	**3.1**	8.0	8.0	24	24	409	409	12.7	8.0	8.0	101	101	387	387
one vs. one	no	7.0	28.0	28.0	196	196	478	478	28.0	28.0	28.0	784	784	670	670
	yes	3.1	28.0	28.0	87	87	510	509	13.0	28.0	28.0	365	364	605	605
DAG-SVM	no	7.0	7.0	7.0	49	49	121	121	28.0	7.0	7.0	196	196	163	163
	yes	3.6	7.0	7.0	25	25	124	124	13.2	7.0	7.0	92	92	151	151
ABT	no	7.0	6.1	6.1	42	42	**53**	**53**	28.0	7.0	7.0	195	195	**106**	**106**
SVM-BDT	no	7.0	3.4	3.5	23	25	119	142	28.0	3.7	4.1	104	114	160	184
FSHC	yes	3.5	**3.1**	**3.4**	**10.8**	**11.7**	149	159	12.5	**3.2**	**3.1**	**40**	**38**	179	181
		glass							iris						
		F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}
one vs. rest	no	9.0	7.0	7.0	63	63	370	370	4.0	3.0	3.0	12	12	139	139
	yes	**3.2**	7.0	7.0	22	22	369	369	**1.0**	3.0	3.0	3	3	140	140
fuzzy OVR	no	9.0	7.0	7.0	63	63	357	357	4.0	3.0	3.0	12	12	140	140
	yes	3.7	7.0	7.0	26	25	366	366	2.0	3.0	3.0	6	6	142	142
one vs. one	no	9.0	21.0	21.0	189	189	366	366	4.0	3.0	3.0	12	12	70	70
	yes	3.8	21.0	21.0	80	79	423	423	1.3	3.0	3.0	3	4	106	106
DAG-SVM	no	9.0	6.0	6.0	54	54	415	415	4.0	2.0	2.0	8	8	48	48
	yes	4.1	6.0	6.0	25	24	119	**119**	1.6	2.0	2.0	3	3	50	50
ABT	no	9.0	5.0	5.0	45	45	28600	28600	4.0	**1.7**	**1.7**	7	7	**12**	**12**
SVM-BDT	no	9.0	3.3	3.7	30	33	**113**	158	4.0	1.7	1.7	7	7	50	50
FSHC	yes	4.8	**3.1**	**3.0**	**15**	**14**	192	193	1.3	1.7	1.7	**2**	**2**	49	49

Continued on page 83

Table 3.12: Average number of classifiers, features and multiplications required to classify a point using FSHC and common multi-class extensions. F_{avg} gives the average number of features per base classifier, Q_{bal} gives the number of base classifiers that need to be queried assuming the number of samples in each class is balanced, Q_{rep} gives the number of base classifiers that need to be queried using the class distribution in the data set, and M_{bal} and M_{rep} are the number of multiplications required to classify a point using the two different methods of estimating the class balance. SV_{bal} and SV_{rep} give the average number of support vectors in all queried base classifiers for the two different methods of estimating the class balance. Time gives the testing time for each test data set.

Continued from Table 3.12 on page 82

		image segm.							statlog (vehicle)						
		F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}
one vs. rest	no	18.0	7.0	7.0	126	126	1930	1930	18.0	4.0	4.0	72	72	1350	1350
	yes	9.3	7.0	7.0	65	65	1930	1930	**8.6**	4.0	4.0	34	34	1350	1350
fuzzy OVR	no	18.0	7.0	7.0	126	126	1940	1940	18.0	4.0	4.0	72	72	1350	1350
	yes	9.5	7.0	7.0	66	66	2000	2000	8.7	4.0	4.0	35	35	1370	1370
one vs. one	no	18.0	21.0	21.0	378	378	1610	1610	18.0	6.0	6.0	108	108	1360	1360
	yes	9.1	21.0	21.0	191	191	1880	1880	9.3	6.0	6.0	56	56	1380	1380
DAG-SVM	no	18.0	6.0	6.0	108	108	463	463	18.0	3.0	3.0	54	54	682	682
	yes	**8.6**	6.0	6.0	52	52	511	511	9.9	3.0	3.0	30	30	676	676
ABT	no	18.0	4.5	4.5	81	81	**90**	**90**	18.0	2.9	2.9	53	53	**240**	**240**
SVM-BDT	no	18.0	3.6	3.6	64	64	460	460	18.0	2.3	2.3	40	41	712	719
FSHC	yes	9.5	**3.0**	**3.0**	**29**	**29**	569	569	10.1	**2.1**	**2.1**	**21**	**21**	733	734
		wine							artificial						
		F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}	F_{avg}	Q_{bal}	Q_{rep}	M_{bal}	M_{rep}	SV_{bal}	SV_{rep}
one vs. rest	no	13.0	3.0	3.0	39	39	155	154	5.0	4.0	4.0	20	20	628	628
	yes	6.3	3.0	3.0	19	19	150	150	2.2	4.0	4.0	8	8	602	602
fuzzy OVR	no	13.0	3.0	3.0	39	39	154	154	5.0	4.0	4.0	20	20	636	636
	yes	4.3	3.0	3.0	13	13	144	144	1.9	4.0	4.0	7	7	610	610
one vs. one	no	13.0	3.0	3.0	39	39	107	107	5.0	6.0	6.0	30	30	336	336
	yes	**4.1**	3.0	3.0	12	12	97	97	**1.4**	6.0	6.0	8	8	133	133
DAG-SVM	no	13.0	2.0	2.0	26	26	71	71	5.0	3.0	3.0	15	15	173	173
	yes	4.1	2.0	2.0	8	8	68	68	1.5	3.0	3.0	4	4	38	38
ABT	no	13.0	2.0	2.0	26	26	**18**	**18**	5.0	**2.0**	**2.0**	10	10	**13**	**13**
SVM-BDT	no	13.0	**1.7**	1.7	22	22	69	72	5.0	**2.0**	**2.0**	10	10	157	157
FSHC	yes	4.7	**1.7**	**1.7**	**8**	**8**	75	76	1.5	**2.0**	**2.0**	**3**	**3**	37	37

Table 3.12 (cont.): Average number of classifiers, features and multiplications required to classify a point using FSHC and common multi-class extensions. F_{avg} gives the average number of features per base classifier, Q_{bal} gives the number of base classifiers that need to be queried assuming the number of samples in each class is balanced, Q_{rep} gives the number of base classifiers that need to be queried using the class distribution in the data set, and M_{bal} and M_{rep} are the number of multiplications required to classify a point using the two different methods of estimating the class balance. SV_{bal} and SV_{rep} give the average number of support vectors in all queried base classifiers for the two different methods of estimating the class balance. Time gives the testing time for each test data set.

different features, or adjusting the cost based on the features already included in the set. This approach would also work if the cost of the features is not a computational cost, but, for example, a monetary cost, like the cost of a medical test.

Feature selection for noisy and redundant features As shown in Table 3.10, in most cases almost all the features are selected for use in the tree, even if the individual classifiers use only a subset of the features. This is not entirely surprising, given that the features are most often selected for inclusion in a data set because they are thought to hold some information. Hence, an additional experiment is conducted specifically to test the effect of redundant and noisy features in a data set. New data sets are created by adding a redundant copy of each of the features as well as four noisy features. The results for the various algorithms, both with and without feature selection, are presented in Table 3.13.

In most cases there is a fairly significant reduction in the summed total number of features used by all the base classifiers. There is a less significant reduction in the total number of features used by the classifier. In these tests, the penalty for including extra features is very low ($\alpha_t = 0.005$), to prioritize accuracy. In data sets where the number of noisy features is large compared the size of the data set (ex. *iris*), the noisy features and redundant features are more likely to be eliminated, as they have a larger effect on the accuracy.

Both the ABT and CART classifiers perform quite well in these tests, at the expense of a larger number of nodes in the tree. For the fuzzy one vs. rest method and tree methods, the improvement in accuracy due to feature selection is significant. Most likely the noisy features are contributing to this reduction in accuracy when feature selection is omitted. The one vs. rest, one vs. one and tree-based classifiers are more robust to noise, and the gains in accuracy are much less significant. The one exception is the *wine* data set, where the number of features under consideration is 30 (13 original features and 13 copied features, plus 4 noise features), and the number of samples is only 178. The relatively small number of samples and large feature set size means the selected features may be too specific to the training set and do not generalize well, contributing to a reduction in accuracy.

3.1.4 FSHC Summary

Experiments with the multi-modal KNN classifier show that FSHC is capable of finding a feature-selected hierarchical classifier that can outperform a single level classifier. It can also find an appropriate single level classifier, although it has some difficulty with

		ecoli				flag				glass			
	FS?	accuracy	F_{all}	F_{tot}	base	accuracy	F_{all}	F_{tot}	base	accuracy	F_{all}	F_{tot}	base
fuzzy OVR	no	0.85	144	18	8	0.45	480	60	8	0.41	154	22	7
	yes	0.78	62.1	17	8	0.51	215.4	59.7	8	0.56	61.9	20.8	7
one vs. one	no	0.80	504	18	28	0.45	1680	60	28	0.40	462	22	21
	yes	0.82	210.6	18	28	**0.55**	757.4	60	28	0.46	188.7	22	21
DAG-SVM	no	0.82	504	18	28	**0.55**	1680	60	28	0.44	462	22	21
	yes	0.85	225.5	18	28	0.42	803.7	60	28	0.48	181.2	22	21
SVM-BDT	no	0.80	126	18	**7.0**	0.37	420	60	**7.0**	0.48	132	22	**6.0**
ABT	no	**0.88**	1008	18	56	0.42	7608	60	126.8	0.50	475	22	21.6
CART	no	0.78	**20.0**	**9.0**	20	0.51	**18.0**	**10.7**	18	**0.65**	**21.7**	**10.7**	21.7
FSHC	yes	0.83	53.8	17	**7.0**	0.41	189.8	59.3	**7.0**	0.55	55.5	19.3	**6.0**
		iris				image segm.				vehicle			
	FS?	accuracy	F_{all}	F_{tot}	base	accuracy	F_{all}	F_{tot}	base	accuracy	F_{all}	F_{tot}	base
fuzzy OVR	no	0.82	36	12	3	0.84	280	40	7	0.62	160	40	4
	yes	0.93	9.3	6	3	0.88	130.9	39.5	7	0.65	71.9	35.3	4
one vs. one	no	0.93	36	12	3	0.90	840	40	21	0.63	240	40	6
	yes	0.93	8.3	4.7	3	0.90	394.2	40	21	0.67	116.7	39.2	6
DAG-SVM	no	0.95	36	12	3	0.92	840	40	21	0.65	240	40	6
	yes	**96**	8.7	4.8	3	0.91	397.1	40	21	0.67	115.6	38.6	6
SVM-BDT	no	0.93	24	12	**2.0**	0.91	240	40	**6.0**	0.56	120	40	**3.0**
ABT	no	0.94	24	12	**2.0**	**0.94**	1064	40	26.6	**0.81**	280	40	7
CART	no	0.94	**4.0**	**2.6**	4	0.93	**43.4**	**15.3**	43.4	0.69	63.3	**21.0**	63.3
FSHC	yes	0.95	6.6	4.7	**2.0**	0.92	126.3	39.1	**6.0**	0.70	**56.2**	33.9	**3.0**
		wine				artificial							
	FS?	accuracy	F_{all}	F_{tot}	base	accuracy	F_{all}	F_{tot}	base				
fuzzy OVR	no	**1.00**	90	30	3	0.53	56	14	4				
	yes	0.93	22	14.3	3	0.59	20.7	11.5	4				
one vs. one	no	0.98	90	30	3	0.47	84	14	6				
	yes	0.94	19.1	13.6	3	0.98	24.7	10.9	6				
DAG-SVM	no	0.99	90	30	3	0.55	84	14	6				
	yes	0.93	23	15.3	3	0.99	28.3	10.3	6				
SVM-BDT	no	0.97	60	30	**2.0**	0.53	42	14	**3.0**				
ABT	no	0.98	90	30	3	0.99	62	14	4.4				
CART	no	0.91	**6.0**	**5.0**	6	**1.00**	**3.0**	**3.0**	**3.0**				
FSHC	yes	0.93	13.3	12	**2.0**	**1.00**	10.7	8.3	**3.0**				

Table 3.13: Accuracy, number of features and number of base classifiers for data sets with added redundant and noisy features for FSHC and multi-class extensions with and without feature selection

inseparable sets. Results with the different accuracy estimation measures show that the filter measure is an important consideration.

Experiments with the binary SVM show that FSHC is able to achieve accuracies comparable with other multi-class methods, but with a shorter testing time. These results also indicate the importance of feature selection, as feature selection is shown to improve accuracy particularly if there are noisy features, or features that are only partially informative about the classes.

The use of a genetic algorithm does present some problems with robustness, as different initializations can generate different results. Additionally, both CART and ABT achieve good results in these tests. The trees generated by these algorithms are different than the tree structures generated by FSHC. ABT and CART add the ability to let points from a single class be output on either side of a node, which appears to improve accuracy.

A different tree-generation algorithm is therefore proposed in Section 3.2, which adds the ability to have classes at more than one node output.

3.2 Multi-modal binary-tree classification (MBT) [2]

In this section, the multi-modal binary tree classifier is introduced. Experiments with the FSHC algorithm indicate that a tree-based approach can achieve good accuracy and is particularly useful if test time is important. However, FSHC does not allow classes to appear at more than one node output.

Both ABT and CART achieve good results on many data sets and both allow classes to appear at both sides of a node. This increases the potential number of nodes of the tree, but generally improves accuracy. However, the ABT still requires that at least two classes be separated at each node. The decision tree does not have this restriction, but uses only a single feature at each node.

Multi-modal binary tree (MBT) tolerates misclassified points by adding additional classification nodes to re-classify initially misclassified training points as required. This creates a piecewise linear separation of the data from different classes, which means non-linear and multi-modal data sets can be separated using a set of simple classifiers such as linear SVMs.

Unlike FSHC, MBT uses a sequential approach to tree design and feature selection rather than jointly optimizing with a genetic algorithm. There are a number of reasons

[2] A version of the work in this section was previously published in [53]

for this change. Because the number of nodes in the tree is variable and restricted by the number of classes, the tree structure cannot be easily represented by a fixed size genome. Additionally, the results from the FSHC experiments indicate that there may be some problems with robustness, and with the increased number of nodes of the tree there is an increased potential that the training set may not be representative and hence the selected features may not generalize well.

Figure 3.11 gives a diagram of the MBT algorithm. Starting with all the samples, MBT first finds the mean of each class in the current set of points. The class means are clustered into two groups using k-means clustering. The samples are separated into these two sets of classes, and used to train an SVM.

The points in the training set are then re-classified using the same SVM. If the initial set of classes was separable, the output should match the class groups. If the classes are not fully separable, some points will be incorrectly classified.

The classified samples are also checked at this point to ensure that at least some samples appear at each output. If not, the box constraints are adjusted and the SVM is retrained.

The output of the SVM determines which points are used to train the lower levels of the tree. The method is applied recursively using the points classified into each output until the number of incorrectly classified points is below a set misclassification rate (r_{mc}) or the tree has reached a preset maximum depth. The outputs containing points from only a single class (or with fewer than r_{mc} points from other classes) are set as leaf nodes of this class.

Feature selection can also be applied at each node as a part of the training process (FS-MBT). Features are selected individually for each SVM in the tree. Features are selected using a sequential forward search (SFS) technique, guided by the conditional mutual information maximization (CMIM) filter measure [48]. The selection of this feature selection measure is discussed in detail in Chapter 4. The number of features to use in each node is selected using a backwards search. Starting with the maximum number of features, features are removed one at a time, starting with the least useful feature according the the SFS/CMIM ordering. Features continue to be removed until the accuracy on the training set has decrease by more than $r_{mc}\%$. The number of features can also be set manually.

3.2.1 Experiments

The proposed algorithm is tested against several techniques for multi-class extension, described in Section 2.5. The proposed method is compared with one vs. rest, one vs. one,

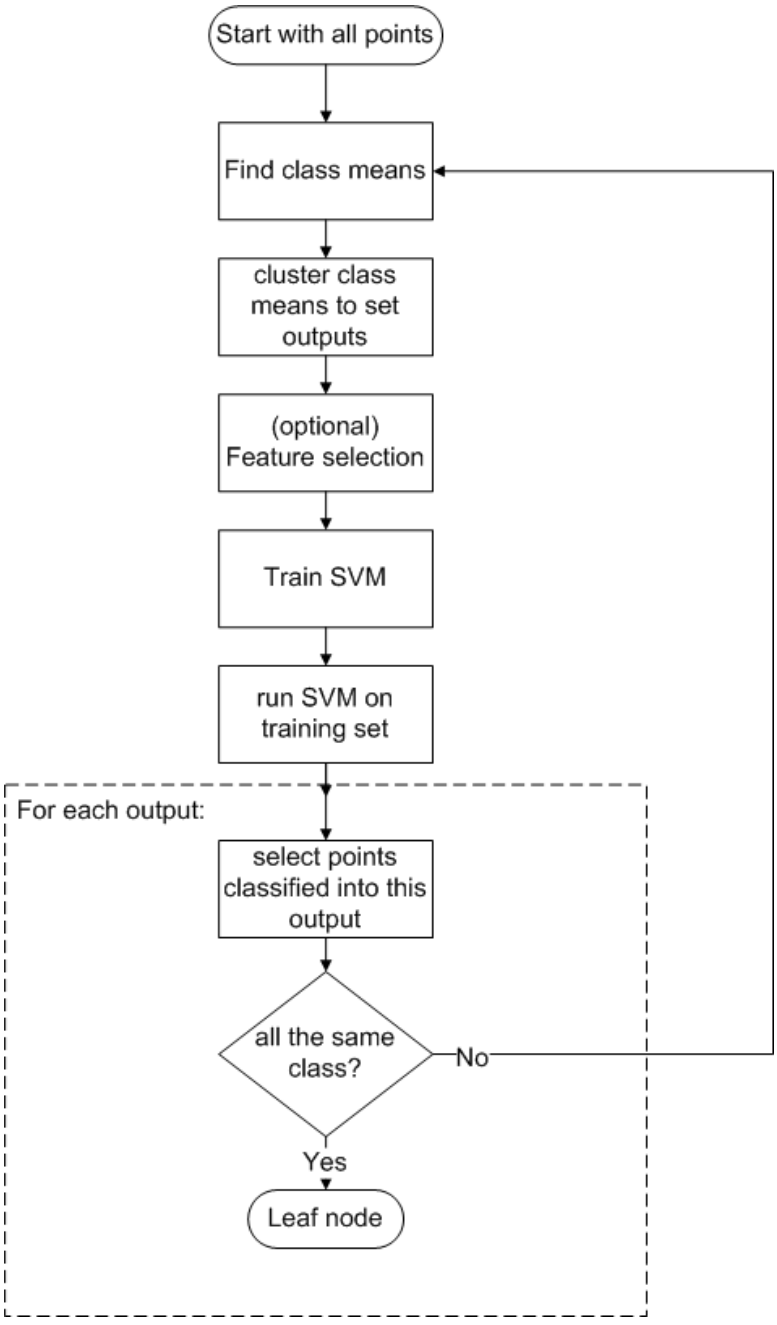


Figure 3.11: MBT algorithm flow chart

ABT, and CART on the basis of classification accuracy, test time, number of tree nodes and total number of features, which corresponds to the memory footprint.

All algorithms are implemented in Matlab, using the Matlab SVM implementation with sequential minimal optimization [125] for training. The decision tree is implemented using the Matlab CART implementation. All SVMs are tested using a linear kernel. Although non-linear kernels can achieve better accuracy on many data sets [21], there are two major advantages to using a linear kernel. First, linear kernels are faster when testing because points can be classified by simply projecting onto the normal of the separating plane rather than using all the support vectors. Second, non-linear kernels have parameters that need to be tuned. Poorly tuned parameters can affect results, and it is difficult to determine if differences in performance are the result of the different multi-class extension, or parameter tuning. Hence, a linear kernel is used for these experiments. Tests are performed with misclassification acceptance rate (r_{mc}) of 0%. An r_{mc} of and 6% is also tested and discussed in Section 3.2.2.

The methods are tested on 20 artificial and 20 real data sets. The data sets are discussed in detail in Section 3.2.1.

Accuracy values are calculated using five-folds cross validation, averaged over three runs, as recommended in [133]. Statistical difference between data sets is calculated using 5×2-folds cross validation F-test [5], which is based on the 5×2-folds cross validation paired t-test recommended in [45]. Two-folds cross validation is used to ensure independence of both the test and training set. The F-measure is calculated as [5]:

$$f = \frac{\sum_{i=1}^{5}\sum_{k=1}^{2}\left(p_i^{(k)}\right)^2}{2\sum_{i=1}^{5} s_i^2} \tag{3.8}$$

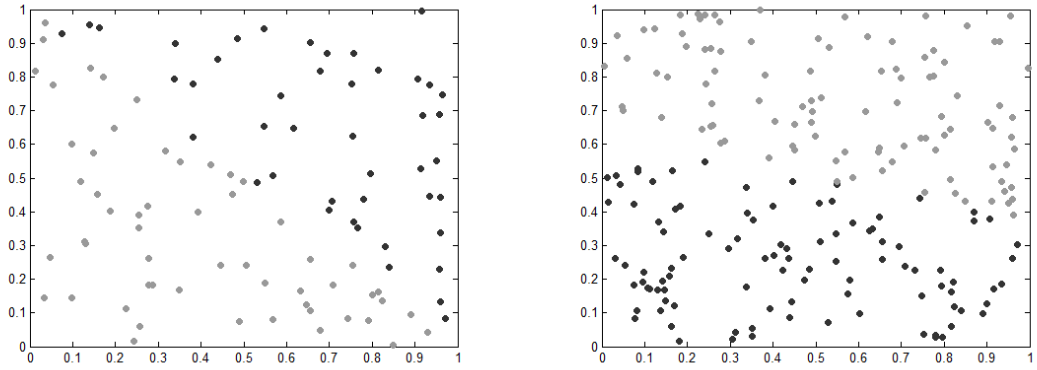
where $p_i^{(k)}$ is the difference between the error rates of the two classifiers on fold k of test i, and s_i^2 is the variance of test i, calculated as:

$$s_i^2 = (p_i^{(1)} - \overline{p_i})^2 + (p_i^{(2)} - \overline{p_i})^2 \tag{3.9}$$

where $\overline{p_i}$ is the average error difference for test i:

$$\overline{p_i} = \frac{p_i^{(1)} + p_i^{(1)}}{2} \tag{3.10}$$

This F test has ten degrees of freedom in the numerator and five degrees of freedom in the denominator. The null hypothesis that the two tested classifiers are equal can be rejected with 95% confidence if $f > 4.74$.



(a) Simple dependent data set

(b) Partially dependent data set

Figure 3.12: Dependent data sets

Data Sets

Table 3.14 gives the properties of all the tested data sets and gives a brief description of the real data set problems. The artificial data sets are described in detail below.

The three *binary* data sets [58] are two class data sets with three noisy binary features. In *binary 1*, points are in class 1 if all three features are 1. In *binary 2*, points are in class 1 if exactly two features are 1. In *binary 3*, points are in class one if all three features take the same value.

The *Monk* data sets [46] use six discrete features and two classes. The six features use a different number of values and all are individually normalized before testing. For *Monk 1*, points are in class 1 if $(f_1 = f_2)|(f_5 = 1)$. For *Monk 2*, points are in class 1 if exactly two features have the value one. *Monk 3* contains 5% class noise, and the class is 1 if $(f_4 = 1 \& f_3 = 1)|(f_4 \neq 4 \& f_1 \neq 3)$.

The *simple dependent*, *partially dependent* and *linearly separable* sets are all linearly separable. The features for the two *dependent* sets are drawn from a uniform distribution, while the *linearly separable* set has features drawn from a Gaussian distribution. The separating lines for *simple dependent* and *linearly separable* sets are diagonal lines from the upper left to the bottom right corner. The *partially dependent* set uses a dividing line with a shallower slope, such that one feature is more informative than the other. Please see Figure 3.12.

The *two-way correlated*, *three-way linear correlated* and *three-way non-linear correlated*

Artificial data sets	N	F	C
binary 1	80	3	2
binary 2	80	3	2
binary 3	80	3	2
monk 1	556	6	2
monk 2	601	6	2
monk 3	554	6	2
partially dependent	200	2	2
simple dependent	100	2	2
two way linear	200	2	2
three way linear	200	3	2
three way non-linear	200	3	2
monotonic	100	4	4
noise with 1 good feature	100	4	4
pure noise	100	3	4
redundant	100	4	4
un-nested	100	3	4
one cluster per class	200	2	2
two u's	400	2	2
Multi-modal	200	2	2
linearly separable	200	2	2

Real data sets	N	F	C	Classification description
Car Evaluation	1727	6	4	Acceptability rating from car features and price
Congressional Voting	435	16	2	Republican or Democrat from voting records
Contraceptive Choice	1473	9	3	Contraceptive choice from info. about partners
Credit Approval	653	15	2	Loan extension granted from personal info.
Dermatology	358	34	6	Eryhemato-Squamous disease type from skin description
E-coli	336	7	18	Localization site from tests
Flag	194	28	8	Country's major religion from flag attributes
Glass	214	9	7	Glass type from oxide content
Haberman's Survival	306	3	2	Breast-cancer surgery survival from patient info.
Ionosphere	351	34	2	Ionosphere detection from radar data
Iris	150	4	3	Iris type from measurements
Page blocks	5473	10	5	Block type from segmented image
Post-op	87	8	3	Post-surgery assignment from patient measurements
Segmentation	2100	18	7	Texture type from high level image attributes
Statlog (vehicle) [114]	846	18	4	Vehicle type from image data
WI Breast Cancer (orig.) [105]	683	9	2	Malignant or benign from cell descriptions
WI Breast Cancer (diag.)	569	30	2	Malignant or benign from cell descriptions
WI Breast Cancer (prog.)	194	32	2	Cancer patient outcome from tumor description
Wine	178	13	3	Wine origin from chemical analysis
Yeast	1484	8	10	Protein localization site from tests

Table 3.14: Properties of artificial and real data sets used for testing MBT and comparing filter-based feature set evaluation measures. Real data sets as well as the *Monk* data sets are taken from the UCI machine learning repository [8]. The *binary* data sets are described in [58].

data sets test how the measures handle correlated features. In the *two-way linear*, the second feature is equal to the first feature with 10% uniform random noise and the class is based on a sum of the features. In *three-way linear*, the third feature is the average of the first two features, which are both random. The class is one if the third feature is above 0.5. In *three way non-linear*, the third feature is an XOR of the first two noisy binary features. The class is one when all three features are zero, but the class can be accurately predicted using any two features. These data sets are illustrated in Figure 3.13.

The *monotonic*, *redundant*, *noise with one good feature* and *un-nested* data sets are all four-class problems built around a single feature that corresponds directly to the class. The *redundant* set uses four copies of this feature. The *noise with one good feature* includes three random features. The *monotonic* set uses four noisy (20%) copies of this feature. The *un-nested* data set uses a noisy (20%) copy of this feature and two features that can be used together to fully separate the set.

The *cluster per class* data set is a unimodal data set with two classes centered at $(0, 0)$ and $(1, 1)$. Either feature can be used to predict the class.

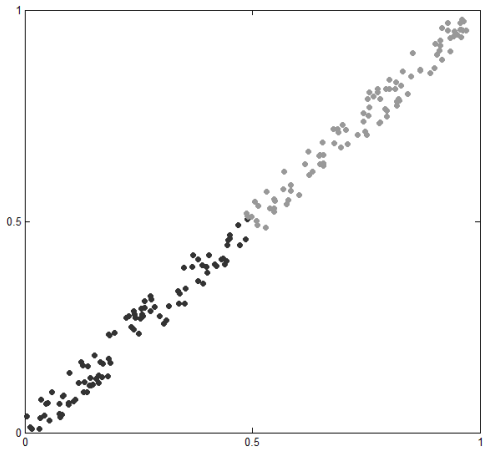
The *multi-modal* data set is a standard XOR function.

The *two U's* data set is a two feature set where the data form two interlocking "u" shapes, where each "u" is one class. This is illustrated in Figure 3.14.

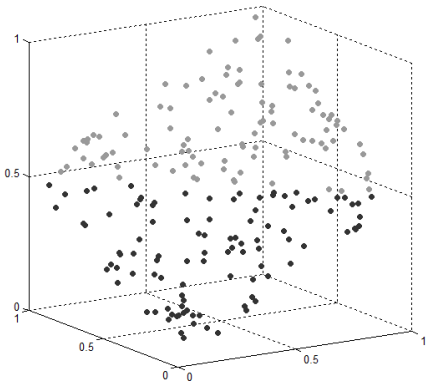
3.2.2 Results and Discussion

The accuracy values for the different methods are presented in Tables 3.15 and 3.16. Both the feature selected and non-feature selected versions of the MBT perform quite well on the artificial data sets and the performance of the MBT algorithms is significantly higher than the other SVM-based alorithms on the multi-modal data sets (*binary 3*, *monk 1, 2 and 3* and *multi-modal*).

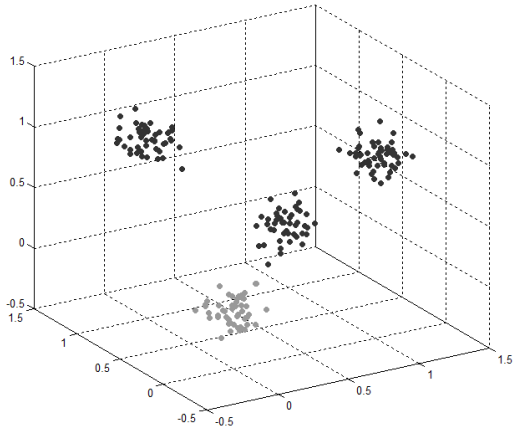
The MBT solutions give the best accuracy on 14 of the 20 artificial data sets, and are within 2% of the best accuracy for all the remaining data sets with the exception of *Monk 2* and the *pure noise* data set, which is not theoretically separable by any classifier. The performance of the MBT on the *Monk 2* data set is significantly higher than the other SVM based approaches. It has a lower average accuracy than CART on the five folds cross validation tests, but the performance of the decision tree is much lower on the two-folds validation tests used in the f-test calculations. The accuracy of the decision tree also varies a large amount in the two-folds tests. Hence, they are not considered to be significantly different using the 5×2-folds cross validation f-test although the decision tree does appear to perform better when using the five-folds tests.



(a) two-way linear



(b) three-way linear



(c) three-way non-linear

Figure 3.13: Correlated data sets

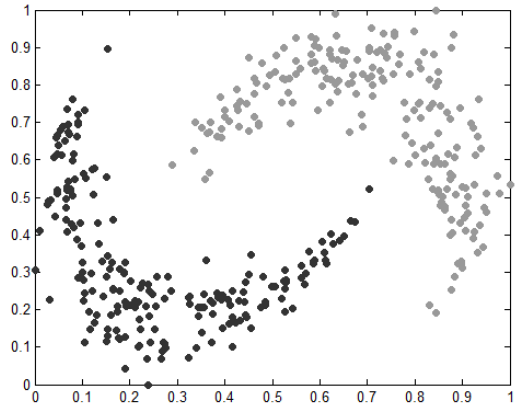


Figure 3.14: Two U's data set

The MBT performs particularly well on the *multi-modal* data set, where a single, linear classifier cannot separate the data, but the data is easily separated using a piecewise linear approach. None of the other SVM-based methods are able to separate this data set, but MBT finds a solution using only two linear classifiers. Using two linear classifiers is also faster than using a kernel classifier in most cases, because the test time for a linear classifier is only proportional to the number of features in the classifier, not the number of support vectors.

MBT also performs well on the real data sets, with the FS-MBT algorithm achieving the best accuracy on seven of the real data sets, which is more than any of the other algorithms.

Feature selection appears to help in most cases, with the FS-MBT outperforming or matching the performance of MBT on all but four real data sets (*congressional voting*, *dermatology*, *wine* and *yeast*), all of which experience a decrease of less than 2%. For the artificial data sets, the accuracy of the feature selected version is comparable to the non-feature selected versions in most cases, with the exception of the *binary 3* data set, where the performance is slightly better, and the *Monk 1* and *Monk 2* data sets, where the performance is slightly lower. For these data sets, a different feature selection measure may perform better, as they contain dependent features, and *Monk 2* contains features with a greater than pairwise dependency.

One potential way to further improve the accuracy of MBT is to use a slightly different method for clustering. Although the class means can indicate which classes are furthest from the others, class means do not fully capture the separability of the classes. Instead,

	accuracy (3 × 5-folds CV)						5 × 2-folds CV F-test 95% ($f > 4.74$)			
	MBT	FS-MBT	ABT	OVO	OVA	CART	MBT vs SVM	FS-MBT vs SVM	MBT vs CART	FS-MBT vs CART
Binary 1	0.96	**0.99**	**0.99**	0.98	**0.99**	0.93			higher	higher
Binary 2	**0.99**	0.98	0.96	0.96	0.98	0.76			higher	higher
Binary 3	0.90	**0.96**	0.75	0.75	0.75	0.77	higher	higher	higher	higher
Monk 1	**0.90**	0.84	0.69	0.67	0.68	0.88		higher		
Monk 2	0.77	0.71	0.66	0.66	0.66	**0.94**	higher			
Monk 3	0.98	0.97	0.80	0.80	0.80	**0.99**	higher	higher		
Partially Dependent	**0.99**	0.98	0.97	0.98	0.98	0.94				
Simple Dependent	0.95	0.93	0.96	0.95	**0.97**	0.84				
2-way linear corr.	0.99	0.98	**1.00**	0.99	0.99	0.99				
3-way linear corr.	1.00	1.00	0.97	0.99	0.99	**1.00**				
3-way non-linear corr.	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99				
1 good feature + noise	**1.00**	**1.00**	**1.00**	**1.00**	0.69	**1.00**				
Pure noise	0.26	0.28	**0.34**	0.33	0.23	0.30				
Un-nested	**1.00**	**1.00**	**1.00**	**1.00**	0.87	0.98				
Monotonic	**1.00**	0.96	**1.00**	**1.00**	0.63	0.92				
Two U's	0.98	**0.99**	0.97	0.97	0.97	0.99				
Multimodal	0.98	**1.00**	0.66	0.59	0.59	0.98	higher	higher		
Linear sep. Gaussian	**0.98**	0.96	0.97	0.97	0.96	0.93				
Redundant	**1.00**	**1.00**	**1.00**	**1.00**	0.64	**1.00**				
Cluster per class	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**				

Table 3.15: Accuracy of MBT and common multi-class extensions on artificial data sets. The second column shows where there is a significant difference between the accuracy of the MBT algorithms and the other SVM-based methods (ABT and OVO) or CART, measured using 5 × 2-folds cross validation f-test to a 95% confidence level ($f > 4.74$).

it may be more beneficial to use a class separability measure such as Fisher's interclass separability criterion [43].

The memory footprint of the MBT classifiers is larger than the ABT, as the number of potential nodes in the tree is larger (see Tables 3.17 and 3.18). This translates into a larger total number of features in the classifier (see Tables 3.19 and 3.20), which directly affects the amount of memory required to store the classifier. Even the feature-selected version of MBT has a larger memory footprint than the other methods. However, for these tests, r_{mc} is set to 0%, which will theoretically make the trees larger, as any misclassified point will cause a new node to be added. A discussion of how the parameter r_{mc} affects the tree size and accuracy is presented in Section 3.2.2. A particular problem for this algorithm is the *pure noise* data set, where there is no way to derive the class from the features. In this case, the number of nodes in the tree is very large, but the accuracy is still approximately chance.

Despite the larger number of nodes and features, the test time for the MBT is shorter than many of the other algorithms. Because the SVM is linear, the test time is directly related to the number of features at each node, since points are classified by taking the

	MBT	FS-MBT	ABT	OVO	OVA	CART
Car Evaluation	0.94	**0.96**	0.84	0.85	0.81	0.96
Congressional Voting	0.95	0.94	**0.96**	0.95	0.96	0.94
Contraceptive Choice	0.48	**0.52**	0.51	0.52	0.48	0.51
Credit Approval	0.84	0.85	0.86	**0.86**	**0.86**	0.84
Dermatology	**0.97**	0.95	0.95	**0.97**	**0.97**	0.95
E-coli	0.79	0.79	0.86	0.86	**0.87**	0.81
Flag	0.42	0.45	0.48	0.43	0.46	**0.57**
Glass	0.63	0.66	0.49	0.63	0.58	**0.69**
Haberman's Survival	0.71	0.72	**0.73**	0.72	0.73	0.68
Ionosphere	0.86	**0.89**	**0.89**	0.88	0.88	0.88
Iris	0.95	0.95	**0.97**	0.96	0.92	0.94
Page blocks	0.94	**0.96**	0.96	0.96	0.95	0.96
Post-op	0.56	0.60	0.69	0.69	**0.69**	0.64
Segmentation	**0.96**	**0.96**	0.95	0.95	0.92	0.95
Statlog (vehicle)	0.78	**0.80**	**0.80**	**0.80**	0.78	0.70
WI Cancer (orig.)	**0.96**	**0.96**	**0.96**	**0.96**	**0.96**	0.94
WI Cancer (diag.)	0.97	0.97	0.97	0.97	**0.98**	0.92
WI Cancer (prog.)	0.66	0.66	**0.73**	0.72	0.71	0.66
Wine	0.96	0.94	0.97	0.96	**0.98**	0.91
Yeast	0.50	0.48	0.58	**0.58**	0.50	0.53

Table 3.16: Accuracy of MBT and common multi-class extensions on real data sets

normal to the separating hyperplane. Tables 3.21 and 3.22 show the average number of features required to classify a point. Tree classifiers do not require all the nodes to be queried to classify a point, hence the number of stored features is higher than the number required to classify a point. Additionally, FS-MBT employs feature selection and therefore the number of features in each node is relatively small, making the classifier test time quite fast. Having a fast test time is important even for classifiers that are trained offline if the intention is to use the classifier in an online manner, where incoming points are classified as they arrive.

Effect of r_{mc} on accuracy, number of nodes and memory footprint

In theory, increasing the value of r_{mc} should decrease the accuracy of the classifier, but also decrease the number of nodes. Experiments were run using $r_{mc} = 0\%$ and also using $r_{mc} = 6\%$. The results are presented in Tables 3.23 and 3.24.

Increasing the value of r_{mc} appears to have only a small effect on accuracy for both the artificial and the real data sets. However, increasing r_{mc} decreases the number of nodes in the generated trees. This also decreases the total number of features and the number of features required to classify a point. This is quite beneficial, as it indicates that this parameter can be used to control the tree size. However, past a certain point, the accuracy will begin to be affected and the tree will degenerate to a standard binary tree where the

	MBT	FS-MBT	ABT		OVO	OVA	CART
Binary 1	2.4	1.8	**1.0**	**(1.0)**	**1.0**	2.0	2.9
Binary 2	1.7	1.5	**1.0**	**(1.0)**	**1.0**	2.0	4.1
Binary 3	5.5	4.3	**1.0**	**(1.0)**	**1.0**	2.0	6.2
Monk 1	9.3	10.4	**1.0**	**(1.0)**	**1.0**	2.0	19.1
Monk 2	13.3	14.4	**1.0**	**(1.0)**	**1.0**	2.0	40.5
Monk 3	9.5	9.4	**1.0**	**(1.0)**	**1.0**	2.0	4.7
Partially Dependent	4.8	4.3	**1.0**	**(1.0)**	**1.0**	2.0	3.2
Simple Dependent	5.2	5.3	**1.0**	**(1.0)**	**1.0**	2.0	4.8
2-way linear corr.	3.6	4.0	**1.0**	**(1.0)**	**1.0**	2.0	1.5
3-way linear corr.	3.9	2.5	**1.0**	**(1.0)**	**1.0**	2.0	**1.0**
3-way non-linear corr.	1.1	1.1	**1.0**	**(1.0)**	**1.0**	2.0	3.4
1 good feature + noise	**2.9**	**2.9**	4.6	(4.6)	6.0	4.0	3.0
Pure noise	48.7	46.9	6.8	(5.8)	6.0	**4.0**	13.3
Un-nested	6.3	5.9	**3.0**	**(3.0)**	6.0	4.0	**3.0**
Monotonic	**3.0**	**3.0**	4.6	(4.6)	6.0	4.0	4.7
Two U's	8.7	7.3	**1.0**	**(1.0)**	**1.0**	2.0	7.4
Multi-modal	2.7	2.9	**1.0**	**(1.0)**	**1.0**	2.0	8.5
Linear sep. Gaussian	4.8	6.9	**1.0**	**(1.0)**	**1.0**	2.0	5.8
Redundant	3.1	3.3	**3.0**	**(3.0)**	6.0	4.0	**3.0**
Cluster per class	1.1	1.1	**1.0**	**(1.0)**	**1.0**	2.0	**1.0**

Table 3.17: Number of base classifiers for MBT and common multi-class extensions extensions using artificial data sets. The number in parenthesis under ABT is the number of different pairwise trained classifiers.

	MBT	FS-MBT	ABT		OVO	OVA	CART
Car Evaluation	67.9	51.3	7.0	(6.0)	6.0	**4.0**	36.1
Congressional Voting	9.2		**1.0**	**(1.0)**	**1.0**	2.0	6.7
Contraceptive Choice	58.7	50.5	**3.0**	**(3.0)**	**3.0**	**3.0**	158.7
Credit Approval	16.5	14.7	**1.0**	**(1.0)**	**1.0**	2.0	33.3
Dermatology	**5.6**	34.2	21.2	(13.0)	15.0	6.0	9.3
E-coli	37.9	51.7	41.6	(20.4)	28.0	**8.0**	18.6
Flag	22.5	20.3	116.6	(27.2)	28.0	**8.0**	20.5
Glass	37.8		18.4	(13.4)	21.0	**7.0**	19.5
Haberman's Survival	16.4	14.4	**1.0**	**(1.0)**	**1.0**	2.0	25.3
Ionosphere	6.3	7.9	**1.0**	**(1.0)**	**1.0**	2.0	14.3
Iris	6.3	7.9	**2.0**	**(2.0)**	3.0	3.0	3.4
Page blocks	108.9	97.8	11.8	(8.4)	10.0	**5.0**	60.8
Post-op	22.8	15.1	**2.6**	**(2.6)**	3.0	3.0	8.3
Segmentation	33.1	49.3	26.0	(14.0)	21.0	**7.0**	41.0
Statlog (vehicle)	44.0	39.9	7.0	(6.0)	6.0	**4.0**	65.0
WI Cancer (orig.)	6.7	11.3	**1.0**	**(1.0)**	**1.0**	2.0	11.3
WI Cancer (diag.)	1.3	7.5	**1.0**	**(1.0)**	**1.0**	2.0	12.7
WI Cancer (prog.)	8.5	7.3	**1.0**	**(1.0)**	**1.0**	2.0	14.3
Wine	**2.5**	2.5	3.0	(3.0)	3.0	3.0	5.8
Yeast	377.3	389.9	176.0	(33.0)	45.0	**10.0**	161.8

Table 3.18: Number of base classifiers for MBT and common multi-class extensions using real data sets. The number in parenthesis under ABT is the number of different pairwise trained classifiers.

	MBT	FS-MBT	ABT	OVO	OVA	CART
Binary 1	7.2	3.8	**3.0**	**3.0**	**3.0**	6.0
Binary 2	5.2	3.5	**3.0**	**3.0**	**3.0**	6.0
Binary 3	16.4	8.1	**3.0**	**3.0**	**3.0**	6.0
Monk 1	55.6	15.3	**6.0**	**6.0**	**6.0**	12.0
Monk 2	79.6	37.7	**6.0**	**6.0**	**6.0**	12.0
Monk 3	56.8	23.1	**6.0**	**6.0**	**6.0**	12.0
Partially Dependent	9.6	5.8	**2.0**	**2.0**	**2.0**	4.0
Simple Dependent	10.4	7.1	**2.0**	**2.0**	**2.0**	4.0
2-way linear corr.	7.2	5.1	**2.0**	**2.0**	**2.0**	4.0
3-way linear corr.	11.8	3.3	**3.0**	**3.0**	**3.0**	6.0
3-way non-linear corr.	3.4	**2.0**	3.0	3.0	3.0	6.0
1 good feature + noise	11.5	**3.0**	18.4	18.4	24.0	16.0
Pure noise	146.2	77.1	20.4	17.4	18.0	12.0
Un-nested	18.8	**3.1**	9.0	9.0	18.0	12.0
Monotonic	12.0	**6.9**	18.4	18.4	24.0	16.0
Two U's	17.3	9.2	**2.0**	**2.0**	**2.0**	4.0
Multimodal	5.5	4.1	**2.0**	**2.0**	**2.0**	4.0
Linear sep. Gaussian	9.6	8.6	**2.0**	**2.0**	**2.0**	4.0
Redundant	12.5	**3.0**	12.0	12.0	24.0	16.0
Cluster per class	2.3	**1.0**	2.0	2.0	2.0	4.0

Table 3.19: Total number of features used by MBT and common multi-class extensions for artificial data sets

	MBT	FS-MBT	ABT	OVO	OVA	CART
Car Evaluation	407.6	122.6	42.0	36.0	36.0	**24.0**
Congressional Voting	147.2	58.9	**16.0**	**16.0**	**16.0**	32.0
Contraceptive Choice	528.0	198.7	**27.0**	**27.0**	**27.0**	**27.0**
Credit Approval	248.0	58.9	**15.0**	**15.0**	**15.0**	30.0
Dermatology	190.4	**25.3**	720.8	442.0	510.0	204.0
E-coli	265.5	124.5	291.2	142.8	196.0	**56.0**
Flag	630.9	285.1	3264.8	761.6	784.0	**224.0**
Glass	340.2	156.7	165.6	120.6	189.0	**63.0**
Haberman's Survival	49.2	19.3	**3.0**	**3.0**	**3.0**	6.0
Ionosphere	215.3	111.3	**34.0**	**34.0**	**34.0**	68.0
Iris	25.1	12.3	**8.0**	**8.0**	12.0	12.0
Page blocks	1089.3	289.3	118.0	84.0	100.0	**50.0**
Post-op	182.4	47.1	**20.8**	**20.8**	24.0	24.0
Segmentation	595.2	307.5	468.0	252.0	378.0	**126.0**
Statlog (vehicle)	792.0	371.7	126.0	108.0	108.0	**72.0**
WI Cancer (orig.)	60.6	36.7	**9.0**	**9.0**	**9.0**	18.0
WI Cancer (diag.)	38.0	88.6	**30.0**	**30.0**	**30.0**	60.0
WI Cancer (prog.)	273.1	125.1	**32.0**	**32.0**	**32.0**	64.0
Wine	32.1	**14.7**	39.0	39.0	39.0	39.0
Yeast	3018.1	1246.4	1408.0	264.0	360.0	**80.0**

Table 3.20: Total number of features used by MBT and common multi-class extensions for real data sets

	MBT	FS-MBT	ABT	OVO	OVA	CART
Binary 1	3.5	3.1	**3.0**	**3.0**	6.0	8.0
Binary 2	3.7	3.1	**3.0**	**3.0**	6.0	9.5
Binary 3	7.9	4.1	**3.0**	**3.0**	6.0	14.2
Monk 1	19.3	**5.2**	6.0	6.0	12.0	29.6
Monk 2	22.7	8.0	**6.0**	**6.0**	12.0	38.6
Monk 3	18.9	11.9	**6.0**	**6.0**	12.0	19.0
Partially Dependent	3.4	2.7	**2.0**	**2.0**	4.0	5.7
Simple Dependent	4.1	3.2	**2.0**	**2.0**	4.0	6.9
2-way linear corr.	3.1	2.0	**2.0**	**2.0**	4.0	4.6
3-way linear corr.	5.4	**2.2**	3.0	3.0	6.0	6.0
3-way non-linear corr.	3.0	**2.0**	3.0	3.0	6.0	8.7
1 good feature + noise	8.0	**2.0**	10.6	24.0	16.0	12.0
Pure noise	17.2	9.2	**9.0**	18.0	12.0	18.1
Un-nested	6.0	**2.0**	6.0	18.0	12.0	9.7
Monotonic	8.0	**4.5**	10.1	24.0	16.0	14.2
Two U's	4.9	4.3	**2.0**	**2.0**	4.0	7.8
Multimodal	3.6	3.2	**2.0**	**2.0**	4.0	11.6
Linear sep. Gaussian	3.6	3.3	**2.0**	**2.0**	4.0	6.8
Redundant	8.0	**2.0**	8.0	24.0	16.0	12.0
Cluster per class	2.0	**1.0**	2.0	2.0	4.0	4.0

Table 3.21: Average number of features required to classify each point using MBT and common multi-class extensions for artificial data sets

	MBT	FS-MBT	ABT	OVO	OVA	CART
Car Evaluation	31.4	**16.5**	18.0	36.0	24.0	29.0
Congressional Voting	18.3	32.8	**16.0**	**16.0**	32.0	48.2
Contraceptive Choice	54.0	**16.4**	18.0	27.0	27.0	93.0
Credit Approval	55.1	17.9	**15.0**	**15.0**	30.0	96.5
Dermatology	95.3	**11.5**	150.8	510.0	204.0	158.6
E-coli	41.2	**21.2**	40.0	196.0	56.0	38.6
Flag	168.6	**114.0**	194.4	784.0	224.0	160.8
Glass	59.3	**41.7**	44.9	189.0	63.0	59.5
Haberman's Survival	12.0	5.1	**3.0**	**3.0**	6.0	21.1
Ionosphere	75.6	72.3	**34.0**	**34.0**	68.0	239.4
Iris	8.4	**4.4**	6.7	12.0	12.0	12.5
Page blocks	59.0	46.2	**39.5**	100.0	50.0	134.2
Post-op	35.3	**13.8**	14.4	24.0	24.0	43.2
Segmentation	77.2	**46.7**	81.0	378.0	126.0	128.7
Statlog (vehicle)	85.6	68.7	**54.0**	108.0	72.0	139.8
WI Cancer (orig.)	13.5	15.4	**9.0**	**9.0**	18.0	42.5
WI Cancer (diag.)	**30.0**	53.1	**30.0**	**30.0**	60.0	166.9
WI Cancer (prog.)	89.7	62.8	**32.0**	**32.0**	64.0	175.8
Wine	22.5	**12.4**	26.0	39.0	39.0	50.8
Yeast	96.6	**43.5**	68.6	360.0	80.0	91.5

Table 3.22: Average number of features required to classify each point using MBT and common multi-class extensions for real data sets

	accuracy		nodes		F_{tot}		F_{test}	
	0%	6%	0%	6%	0%	6%	0%	6%
Binary 1	**0.99**	**0.99**	1.8	**1.3**	3.8	**3.0**	3.8	**3.0**
Binary 2	0.98	**0.98**	1.5	**1.3**	3.5	**3.5**	3.5	**3.5**
Binary 3	0.96	**0.97**	**4.3**	4.5	**8.1**	**8.1**	**8.1**	**8.1**
Monk 1	**0.84**	**0.84**	10.4	**9.9**	15.3	**12.3**	15.3	**12.3**
Monk 2	**0.71**	0.66	14.4	**14.3**	37.7	**27.1**	37.7	**27.1**
Monk 3	**0.97**	0.97	9.4	**5.7**	23.1	**17.7**	23.1	**17.7**
Partially Dependent	**0.98**	0.97	4.3	**2.6**	5.8	**3.5**	5.8	**3.5**
Simple Dependent	0.93	**0.96**	5.3	**3.7**	7.1	**5.4**	7.1	**5.4**
2-way linear corr.	**0.98**	**0.98**	4.0	**1.4**	5.1	**1.0**	5.1	**1.0**
3-way linear corr.	**1.00**	0.99	2.5	**1.0**	3.3	**1.1**	3.3	**1.1**
3-way non-linear corr.	**1.00**	**1.00**	**1.1**	**1.1**	**2.0**	**2.0**	**2.0**	**2.0**
1 good feature + noise	**1.00**	**1.00**	**2.9**	**2.9**	**3.0**	**3.0**	**3.0**	**3.0**
Pure noise	0.28	**0.31**	**46.9**	47.1	77.1	**75.7**	77.1	**75.7**
Un-nested	**1.00**	1.00	5.9	**5.8**	3.1	**3.0**	3.1	**3.0**
Monotonic	**0.96**	0.91	**3.0**	5.9	**6.9**	7.0	**6.9**	7.0
Two U's	**0.99**	0.97	7.3	**1.3**	9.2	**2.0**	9.2	**2.0**
Multimodal	**1.00**	1.00	2.9	**2.3**	4.1	**3.8**	4.1	**3.8**
Linear sep. Gaussian	0.96	**0.96**	6.9	**1.4**	8.6	**2.4**	8.6	**2.4**
Redundant	**1.00**	**1.00**	3.3	**2.9**	**3.0**	**3.0**	**3.0**	**3.0**
Cluster per class	**1.00**	**1.00**	**1.1**	**1.1**	**1.0**	**1.0**	**1.0**	**1.0**

Table 3.23: Comparison FS-MBT with different r_{mc} values on artificial data sets. F_{tot} gives the total number of stored features and F_{test} gives the average number of features to classify a point

divisions are selected simply by clustering the class means.

3.2.3 MBT Summary

The MBT algorithm gives the best performance on a majority of the artificial data sets, and on a plurality of the real data sets. The performance of the MBT is significantly higher than other SVM-based methods on multi-modal data sets.

Adding feature selection for the individual classifiers not only improves the accuracy in many cases, but also reduces the memory footprint of the generated trees. Even with feature selection, the memory footprint of the classifier is large compared to the other algorithms. However, because not all the nodes in the tree need to be queried, the feature selected version of the algorithm has relatively good performance, requiring the fewest average number of features to classify a point on the majority of the real data sets. This is especially important if the intention is to use the classifier to classify new, incoming points in an online manner.

Lastly, the MBT can be used with different classifiers and different feature selection methods. MBT can also be used with multi-class classifiers by using a different K value to

	accuracy		nodes		F_{tot}		F_{test}	
	0%	6%	0%	6%	0%	6%	0%	6%
Car Evaluation	**0.96**	0.95	51.3	**35.1**	122.6	**68.8**	16.5	**7.1**
Congressional Voting	0.94	**0.95**		**6.3**	58.9	**5.2**	32.8	**1.6**
Contraceptive Choice	**0.52**	**0.52**	**50.5**	54.1	198.7	**108.5**	16.4	**7.7**
Credit Approval	0.85	**0.86**	14.7	**10.4**	58.9	**18.4**	17.9	**4.6**
Dermatology	**0.95**	0.94	34.2	**12.4**	25.3	**13.9**	11.5	**4.0**
E-coli	**0.79**	0.77	51.7	**45.5**	124.5	**88.7**	21.2	**9.0**
Flag	0.45	**0.51**	**20.3**	39.9	285.1	**139.0**	114.0	**17.3**
Glass	0.66	**0.68**		**39.2**	156.7	**96.9**	41.7	**13.5**
Haberman's Survival	**0.72**	**0.72**	14.4	**12.5**	19.3	**14.9**	5.1	**4.3**
Ionosphere	**0.89**	0.88	7.9	**7.8**	111.3	**11.7**	72.3	**5.0**
Iris	**0.95**	**0.95**	7.9	**5.0**	12.3	**5.2**	4.4	**2.2**
Page blocks	**0.96**	0.93	97.8	**70.9**	289.3	**128.9**	46.2	**9.7**
Post-op	0.60	**0.62**	**15.1**	21.4	47.1	**45.2**	13.8	**10.9**
Segmentation	**0.96**	0.94	**49.3**	53.6	307.5	**145.7**	46.7	**8.4**
Statlog (vehicle)	**0.80**	0.76	**39.9**	41.7	371.7	**148.6**	68.7	**13.4**
WI Cancer (orig.)	**0.96**	0.94	11.3	**7.5**	36.7	**6.2**	15.4	**2.8**
WI Cancer (diag.)	**0.97**	0.93	7.5	**6.0**	88.6	**8.0**	53.1	**3.4**
WI Cancer (prog.)	0.66	**0.70**	**7.3**	10.7	125.1	**68.7**	62.8	**15.2**
Wine	**0.94**	0.91	**2.5**	5.8	14.7	**8.5**	12.4	**3.5**
Yeast	0.48	**0.50**	389.9	**341.5**	1246.4	**944.8**	43.5	**18.9**

Table 3.24: Comparison FS-MBT with different r_{mc} values on real data sets. F_{tot} gives the total number of stored features and F_{test} gives the average number of features to classify a point

create the initial class clustering.

3.3 Summary

FSHC uses genetic algorithms to jointly optimize the tree structure and selected features. Tests with the multi-modal KNN classifier with artificial data sets show that FSHC is capable of outperforming a flat classifier on certain data sets. Experiments with binary SVM classifiers show that the accuracy of FSHC is comparable to other multi-class extensions, but with a shorter testing time.

However, FSHC does suffer from problems with robustness and may select a classifier structure and features that are too specific to the training set, leading to generalization problems.

The MBT algorithm aims to correct some of the generalization and robustness problems by performing the tree design and feature selection sequentially using non-stochastic methods. Additionally, MBT adds the ability to have classes appear at more than one node output, which is similar to the ABT and CART algorithms.

The MBT algorithm performs well compared to the other multi-class extensions, giving the highest accuracy on the majority of the artificial data sets and a plurality of the real data sets. The accuracy of the MBT is significantly higher than other classifiers on known multi-modal data sets.

Importantly, experiments with both the FSHC and MBT show that the feature selection is beneficial for both tree-based and non-tree-based extensions, and experiments with the FSHC show that the chosen feature selection measure affects the accuracy and size of the classifier.

Chapter 4

Evaluation of filter based feature selection measures

This chapter describes the testing and evaluation of commonly used filter-based feature subset evaluation measures. Experiments performed in Chapter 3 show that both the multi-modal binary tree algorithm and other multi-class extensions benefit from feature selection. However, experiments with the feature-selected hierarchical classifier show that the selected feature set evaluation measure is important, as it affects the accuracy. Although there are many common feature set evaluation measures, it is unclear which measures are appropriate to use with each classifier.

The goal of this chapter is to assess the ability of commonly used filter-based measures to correctly identify informative features and to assess their performance with respect to specific classifiers. As seen in Chapter 2, a large number of feature selection measures have been proposed in literature. However, there is little consensus on which measures are best for specific problems.

One of the most intuitive feature subset evaluation measures is a wrapper, where the feature subset is evaluated directly on the classifier being designed. While wrapper methods give direct feedback about the ability of the classifier to use the feature set, they tend to be computationally expensive, since they require training a new classifier for every feature set to be tested. Wrappers can also over-fit [143].

Filter measures are less computationally expensive, but may not as accurately reflect the ability of the classifier to use the selected features. The ideal feature set evaluation measure is a fast filter measure whose value closely reflects the accuracy of the classifier on that feature set. Numerous filter methods have been proposed, but there has been

little work directly comparing various filters. This is complicated by the fact that the "best" filter measure is likely classifier specific, since different classifiers perform best with different feature sets [81].

For a researcher looking to use feature selection as a tool for solving a specific problem, the large number of potential solutions and the lack of direct comparison currently available in the literature can make it difficult to determine which measures are most suitable for a particular application or classifier. Testing a large number of filter measures is impractical in many cases, especially when the feature selection method is not the only parameter to be tuned.

This work tests 16 common filter measures for use with K-nearest neighbours and support vector machine classifiers. The measures are tested on 20 real and 20 artificial data sets, which are designed to probe for specific challenges. The strengths and weaknesses of each measure are discussed with respect to the specific challenges and correlation with classifier accuracy. The results highlight several major problems with a number of common filter measures and give guidance about which filter measures are appropriate for different classifiers.

This information is then used to guide the selection of a feature evaluation measure for future tests with human motion data. A new feature measure is also proposed, which can overcome some of the specific deficiencies seen in other feature evaluation measures.

4.1 Filter measures

A detailed description of each of the tested filter measures is given in Chapter 2 Section 2.4.2. The tested measures can be categorized a number of different ways, as described in Table 2.2 in Chapter 2 Section 2.4.1. The set of tested measures includes both supervised and unsupervised measures as well as both univariate and multivariate measures. It also includes a variety of different types of multivariate measures, including both additive and grouped measures as well as both full set and pairwise measures.

Table 4.1 shows the filter measures and the parameters used for the tests in this chapter.

4.2 Experiments

The purpose of these experiments is to determine the ability of each filter measure to identify known informative features and also to determine how well each filter measure

measure	Category	Section	parameters	
			description	**values**
Fisher's interclass separability criterion	distance	2.4.2	–	–
RELIEF-F	distance	2.4.2	K: # neighbours	K={1, 3, 5, 10}
subset-RELIEF	distance	2.4.2		K={1, 3, 5, 10}
count-based RELIEF	distance	2.4.2		K={1}
KL divergence	probability	2.4.2	continuous / discrete	discrete q={2,3,4,5,10,e,u}
JS divergence	probability	2.4.2	q: quantization	continuous (single Gaussian)
Bhattacharyya	probability	2.4.2		
mutual information	information theoretic	2.4.2	q: quantization	q={2, 3, 5, 10}
symmetric uncertainty	information theoretic	2.4.2	(discrete only)	
CMIM	information theoretic	2.4.2		
mRMR	information theoretic	2.4.2		
FOU	information theoretic	2.4.2		
LS	neighbouhood graph	2.4.2	K: # neighbours	K={2, 3, 5, 10, 25, 50, N/10, N/4, N/2}
			Kern: kernel	Kern={heat}
			t: heat kernel param	t={0.1, 1, 10, 100}
MCFS	neighbouhood graph	2.4.2	K: # neighbours	K={25, 50, N/4, N/2}
			Kern: kernel	Kern={heat, dot, zero-one}
			t: heat kernel param	t={0.1, 1, 10, 100}
consistency	consistency	2.4.2	q: quantization	q={2,3,4,5,10,e,u}
CFS	correlation / information theoretic	2.4.2	q: quantization	q={2, 3, 5, 10}

Table 4.1: Tested filter measures and parameters

can predict the classification accuracy of different feature sets with a specific classifier. To determine the prediction ability of the measures, classifiers are first trained using different feature sets from within each data set. The filter measures are assessed based on the correlation between the calculated filter value and the classifier accuracy.

The randomized data set selection assesses the ability of the filter measure to correctly predict the benefit of feature sets selected using stochastic techniques such as genetic algorithms [85], simulated annealing [54] and dependency aware feature selection [143], all of which use randomly selected sets.

Sequential methods such as sequential forwards or backwards search (SFS/SBS) [130] add features to the selected set one at a time, selecting the feature that is most beneficial with respect to the other features in the set. These types of feature selection methods benefit from filter measures that rank the most informative features the most highly.

Ranked selection methods, where each feature is ranked individually and the best Q features or all features over a pre-set threshold are selected, require univariate measures that are able to identify the most informative features. These types of methods are most often used with data sets that have large number of features, where sequential or stochastic search techniques are impractical.

Two exemplar classifiers are used to demonstrate the difference between the filter measures: K-nearest neighbours (KNN) and support vector machine (SVM). Because SVM is a binary classifier, tests are run using two common multi-class extensions: one vs. one and one vs. rest. Both the KNN and SVM implementations are from Matlab. The SVM kernel used is a linear kernel. Although non-linear kernels can achieve better accuracy on many data sets [21], all non-linear kernels have parameters that need to be tuned. Poorly tuned parameters can affect results, and it is difficult to determine if differences in performance are the result of the different multi-class extension, or parameter tuning. Hence, a linear kernel is used for these experiments.

The accuracy of the classifier is assessed for each feature set using 5-fold cross validation, averaged three times. Each filter measure described in Section 2.4.2 is calculated on that feature set. For data sets with fewer than 10 features, the number of feature sets tested is $2^F - 1$, which is the number of unique feature sets. For data sets with more than 10 features, the number of possible feature sets becomes too large to handle easily, and the number of tests is set to $2 * F * (\log_2(F) + 1)$.

In a real feature selection problem, measures are used to compare different feature sets from within the same data set. Hence, correlation of filter measures and accuracy values across data sets is not as important as good correlation within each individual data set. The specific value of the filter measure is also less important than how the value changes for different feature sets. It is less important to know what the specific accuracy of a feature set will be than it is to know whether a different feature set will improve accuracy. Hence, these experiments assess the correlation of the classifier accuracy and filter measure value for each individual data set. Correlation across different data sets is not assessed.

Correlation is assessed using Pearson's linear correlation [74]. There are also several data sets where the accuracy is unchanged regardless of the feature set selected. In these cases, it is not possible to calculate a correlation. Instead, the tables record the amount the filter measure changes as more features are added, as a percentage of the single feature set score:

$$\frac{1}{N_{FS>1}} \sum_{i=1}^{N_{FS>1}} \left(\frac{J_i - \overline{J_{FS=1}}}{N_{Fi}} \right) \Big/ \overline{J_{FS=1}} \tag{4.1}$$

where $N_{FS>1}$ is the number of feature sets that use more than one feature, J_i is the filter measure value for feature set i with more than one feature, and $\overline{J_{FS=1}}$ is the average filter value for all the feature sets that use only one feature.

Tests are performed on a number of artificial and real data sets, as described in Section 4.2.1. The artificial data sets are designed to probe for specific problems with the filter

measures, and the real data sets are used to determine how well these filter measures behave in the real world.

There are several measures that require a probability distribution (the probability based measures, the mutual information-based measures and CFS). In this paper, a continuous (Gaussian) probability function is tested, and also a discrete form, using a number of quantization bins. Tests are also run where the number of bins is set to be equal to the number of clusters in the data. The number of clusters is determined using the Gap statistic [151].

4.2.1 Data Sets

The selected filter measures are tested on a number of real and artificial data sets. Table 3.14 in Chapter 3 gives a description of each of the real data sets, all taken from the UCI machine learning repository [8].

The real data sets have a sufficient number of features that in most cases exhaustive search and wrapper methods are too computationally expensive for real world applications, especially where the feature set is not the only parameter that needs to be adjusted. However, the sets do not contain so many features that univariate measures and ranked selection are preferable. The artificial sets contain fewer features in order to allow a full exploration and understanding of the set and the possible reasons for failure. Any filter measure that does not work well for sets with a relatively small number of features is unlikely to perform well on sets with a larger number of features.

The artificial data sets are described in detail in Chapter 3 Section 3.2.1. The artificial data sets are designed to test a number of commonly encountered feature selection challenges, outlined below. Table 4.2 gives the properties of all the tested artificial data sets and their challenges.

Data set challenges

- non-monotonic sets. In a non-monotonic set, adding a feature reduces the accuracy of the classifier. A filter measure can capture this reduction only if adding another feature to a set can also decrease the filter measure value.

- dependence. If two features are fully dependent, there is no accuracy improvement when one feature is used alone, but a large accuracy improvement when the features

Artificial data sets	N	F	C	NM	dep.	corr.	red.	noise
binary 1	80	3	2		✓			
binary 2	80	3	2		✓			
binary 3	80	3	2		✓			
monk 1	556	6	2	✓	✓			✓
monk 2	601	6	2	✓	✓			✓
monk 3	554	6	2	✓	✓			✓
partially dependent	200	2	2		✓			
simple dependent	100	2	2		✓			
two way linear	200	2	2		✓	✓		
three way linear	200	3	2	✓	✓	✓		
three way non-linear	200	3	2		✓	✓		
monotonic	100	4	4			✓		
noise with 1 good feature	100	4	4	✓				✓
redundant easy	100	4	4			✓	✓	
un-nested	100	3	4	✓	✓			
one cluster per class	200	2	2			✓	✓	
two u's	400	2	2					
multi-modal	200	2	2		✓			
linearly separable	200	2	2					

Table 4.2: Properties of artificial data sets. N is the number of samples, F is the number of features, and C is the number of classes. NM describes whether the data set is non-monotonic, dep. describes whether it contains dependent features, corr. describes whether it contains correlated features, red. describes if it contains redundant features, and noise indicates if it contains noisy features.

are used together. A filter measure that considers each feature in isolation will not be able to find sets of dependent, informative features.

- correlated features. Correlated features may be informative, but it is easy for a filter measure to overestimate the benefit of adding a correlated feature, since some of the information is already captured by the other features in the set.
- redundance. Truly redundant features (copies) do not add any information. A filter measure should be able to detect true redundance, and adding a redundant feature should not change the filter measure value.
- noise. Pure noise features do not help differentiate classes and can reduce accuracy. Fully random features should not improve filter measure values.

Artificial data sets

4.2.2 Combining univariate and additive multivariate measures

For univariate measures, the individual feature scores must be combined to give a performance measure for the set. Possible methods include summing and averaging the individual

feature values. Using the minimum or maximum has also been suggested [7].

The minimum and maximum method are theoretically poor because the overall filter value is dominated by the best or worst feature in the set. Averaging is also a poor choice, since once the individual best feature is included in the set, any additional features will only reduce the filter measure value. This method would only work with a data set that is strictly non-monotonic, which is very unusual. Conversely, summing assumes a strictly monotonic data set, unless the filter value is able to take negative values, in which case it is possible for summing to rank smaller sets more highly.

Tests using the RELIEF-F filter measure confirm the poor performance of the minimum, maximum and averaging methods on the artificial data sets, as seen in Table 4.3. The results for the real data show the best performance either from summing or from taking the max, as shown in Table 4.4. However, using the maximum value is theoretically unsound and will give the same ranking to a larger number of data sets, making it difficult to differentiate between sets. For these reasons, the remaining tests will find the correlation of univariate measures by summing the values of the individual features in the set.

When assessing additive multivariate measures, either all features are assessed with respect to all the other features in the set (grouped method), or the set is assessed using a sequential forward search method (SFS), where a single feature is selected to start, and additional features are assessed with respect to the selected features. The highest ranked feature is then added to the set, the summed filter value is updated, and the remaining features are re-assessed.

When combining additive multivariate measures, the grouped method is more computationally efficient, but the SFS method is theoretically better. Consider, for example, the *two-way linear correlated* data set. The individual features are both highly predictive of the class but are also highly correlated. With a grouped approach, neither feature is scored highly, because of the correlation with the other features in the set. With an SFS approach, one feature is initially scored highly, and only the other feature gets a lower score, due to the correlation with the feature that has already been added. The SFS method gives a more realistic view of the performance, where at least one of the features provides a baseline of accuracy for the classifier, but adding more features adds little. Experiments with the multivariate mutual information measures all show that the SFS method performs better, giving a higher correlation with the classifier accuracy for both classifiers.

	RELIEF-F (K=1)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	avg	sum	min	max	avg	sum	min	max	avg	sum	min	max
Binary 1	0.10	**0.58**	-0.15	0.25	0.07	**0.54**	-0.15	0.20	0.12	**0.93**	-0.39	0.56
Binary 2	0.06	**0.51**	-0.24	0.41	0.15	**0.49**	-0.05	0.40	-0.06	**0.56**	-0.48	0.42
Binary 3	**0.00%**	100.00%	-5.26%	5.08%	**0.00%**	100.00%	-5.26%	5.08%	0.19	**0.89**	-0.33	0.63
Monk 1	0.81	**0.98**	0.12	**0.98**	0.82	**0.98**	0.13	**0.98**	0.30	**0.58**	-0.07	**0.58**
Monk 2	-	-	-	-	-	-	-	-	**0.00**	**0.00**	**0.00**	**0.00**
Monk 3	-0.43	-0.59	-0.59	**-0.04**	-0.41	-0.56	-0.56	**-0.04**	-0.44	-0.64	-0.64	**-0.01**
Simple dependent	0.02	**0.93**	-0.48	0.52	0.00	**0.92**	-0.50	0.50	0.01	**0.93**	-0.49	0.51
Partially dependent	0.78	**0.99**	0.36	**0.99**	0.78	**0.99**	0.36	**0.99**	0.75	**0.98**	0.32	**0.98**
2-way linear corr.	-0.72	**0.67**	-0.97	-0.28	0.00	**1.00**	-0.50	0.50	-0.17	**0.98**	-0.64	0.35
3-way linear corr.	0.64	**0.76**	0.31	0.74	0.62	**0.77**	0.27	0.74	0.67	**0.75**	0.38	0.73
3-way non-linear corr.	0.12	**0.88**	-0.39	0.54	0.12	**0.88**	-0.40	0.53	0.04	**0.88**	-0.47	0.48
1 good and noise	0.82	**1.00**	0.25	**1.00**	0.87	0.97	0.41	**0.98**	0.83	**0.99**	0.26	**0.99**
Pure noise	0.58	-0.49	0.21	**0.77**	-0.70	**0.31**	-0.27	-0.82	0.64	-0.51	0.10	**0.91**
Un-nested	0.10	**0.75**	-0.31	0.43	0.27	**0.84**	-0.22	0.62	0.13	**0.78**	-0.29	0.48
Monotonic	0.16	**0.81**	-0.35	0.63	-0.09	**0.78**	-0.44	0.35	0.15	**0.85**	-0.36	0.63
Two U's	0.18	**0.76**	-0.34	0.65	0.09	**0.70**	-0.42	0.57	0.32	**0.85**	-0.19	0.75
Multimodal (XOR)	0.08	0.22	-0.43	**0.57**	-0.09	**0.05**	-0.58	0.42	-0.07	0.07	-0.56	**0.44**
linear sep. Guassian	0.39	**1.00**	-0.13	0.80	0.36	**1.00**	-0.16	0.78	0.29	**1.00**	-0.23	0.73
Redundant	**0.00%**	100.00%	**0.00%**	**0.00%**	**0.00**	-0.65	**0.00**	**0.00**	**0.00%**	100.00%	**0.00%**	**0.00%**
2 clusters	**0.00%**	100.00%	-2.84%	2.84%	**0.00%**	100.00%	-2.84%	2.84%	**0.00%**	100.00%	-2.84%	2.84%

Table 4.3: Correlation between the RELIEF-F filter values and SVM and 1-NN accuracy values using different methods for combining univariate filter measures on artificial data sets

	RELIEF-F (K=1)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	avg	sum	min	max	avg	sum	min	max	avg	sum	min	max
Abalone	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Car Evaluation	0.37	**0.77**	-0.01	0.31	0.32	**0.73**	-0.06	0.23	0.18	**0.48**	-0.05	0.13
Cardiotocography	**0.01**	-0.55	-0.51	0.51	0.03	-0.57	-0.50	**0.52**	-0.17	-0.56	-0.64	**0.44**
Congressional Voting	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Contraceptive Choice	-0.32	-0.55	-0.55	**0.06**	0.02	-0.21	-0.21	**0.13**	-0.02	-0.14	-0.14	**0.00**
Credit Approval	-0.03	**0.46**	-0.22	0.37	-0.02	**0.46**	-0.21	0.38	-0.04	**0.54**	-0.26	0.41
Dermatology	0.12	0.63	-0.41	**0.64**	0.09	0.61	-0.41	**0.63**	0.11	**0.66**	-0.43	0.64
E-coli	0.23	-0.35	-0.33	**0.60**	0.21	-0.36	-0.31	**0.55**	0.20	-0.36	-0.34	**0.58**
Flag	0.24	-0.46	-0.22	**0.53**	0.15	-0.70	-0.32	**0.49**	0.08	-0.32	-0.22	**0.33**
Glass	0.23	-0.16	-0.19	**0.41**	0.21	-0.17	-0.15	**0.40**	0.16	-0.16	-0.18	**0.38**
Haberman's Survival	**0.84**	0.81	0.78	0.58	0.55	0.52	**0.83**	0.07	-0.88	-0.74	-0.82	**-0.62**
Ionosphere	0.55	**0.77**	-0.41	0.77	0.54	**0.78**	-0.41	0.77	0.33	0.30	-0.23	**0.36**
Iris	0.70	0.74	0.22	**0.92**	0.67	0.81	0.18	**0.90**	0.70	0.82	0.17	**0.96**
Mushroom	0.32	**0.64**	-0.03	0.59	0.32	**0.64**	-0.02	0.59	0.35	0.58	-0.10	**0.63**
Page blocks	-0.01	**0.51**	-0.42	0.26	0.08	**0.63**	-0.34	0.31	0.09	**0.50**	-0.36	0.24
Post-op	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Segmentation	0.33	0.62	-0.22	**0.65**	0.23	**0.65**	-0.33	0.58	0.40	0.55	-0.18	**0.69**
Statlog (vehicle)	0.03	0.48	-0.50	**0.52**	0.03	0.49	-0.52	**0.53**	0.10	0.41	-0.44	**0.47**
WI Breast Cancer (orig.)	0.11	**0.58**	-0.32	0.52	0.11	**0.58**	-0.32	0.52	0.19	**0.62**	-0.26	0.61
WI Breast Cancer (diag.)	0.23	0.48	-0.17	**0.63**	0.23	0.48	-0.18	**0.63**	0.32	0.53	-0.11	**0.71**
WI Breast Cancer (prog.)	0.05	-0.79	**0.53**	-0.35	0.06	-0.78	**0.52**	-0.35	0.06	**0.20**	-0.08	0.08
Wine	0.62	0.76	-0.20	**0.78**	0.62	0.76	-0.20	**0.78**	0.61	**0.72**	-0.15	0.71
Yeast	0.29	-0.49	-0.22	**0.67**	0.42	-0.24	-0.03	**0.60**	0.17	-0.58	-0.34	**0.61**

Table 4.4: Correlation between the RELIEF filter values and SVM and 1-NN accuracy values using different methods for combining univariate filter measures on real data sets

4.3 Results and Discussion

This section presents the results from numerous filter measure tests. Section 4.3.1 describes whether or not each filter measure gives the highest score to the feature set with the highest accuracy, and whether or not it is capable of selecting the known informative features in the artificial data sets. Section 4.3.2 discusses the correlation between the filter measures and the accuracy of the classifier. Section presents conclusions and discusses the performance of each filter measure with respect to the specific data set challenges detailed in Section 4.2.1.

Appendix C discusses the performance of the measures on each artificial data set in detail, describing notable filter measure successes and failures.

4.3.1 Ability of filter measures to identify informative features

Table 4.5 describes whether or not each filter measure can correctly identify the best feature set. A filter measure that is able to select the best set, but is not otherwise well correlated with the accuracy may drive a sub-optimal search away from the best set, but will still select the best set if an exhaustive search is used.

Univariate measures do not work well for non-monotonic sets, but in many cases they do rate the informative features more highly. In the table, a "*" is used to identify cases where a univariate measure correctly rates the informative features more highly than the non-informative features. In these cases, the univariate measures would suffice if the researcher also had a way to determine the correct number of features in the set, though this itself is a challenging problem. For other data sets, the best feature set does include all the features. The univariate measures will select these full sets by default. A"*" is also used to identify additive multivariate measures that select the informative features first.

The most challenging data sets are the non-monotonic sets, especially the sets where some informative features are not included in the best set (*un-nested* and *three-way non-linear*). Another very difficult set is *Monk 1*, where two features are strongly dependent, and neither individual feature is informative about the class in any way. This data set is particularly difficult for the univariate measures. The unsupervised measures have trouble when the classes do not come from clusters in the underlying manifold, or when there are noisy features included in the set.

	FS	Fish.	RELIEF	subset-RELIEF	count-RELIEF	prob	MI	SU	mRMR	CMIM	FOU	LS	MCFS	consis-tency	CFS
Binary 1	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Binary 2	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Binary 3	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Monk 1				✓	✓						*			✓	
Monk 2		✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Monk 3				✓		*	*	*	*	*	*				
Simple dependent	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Partially dependent	y	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	
2-way linear corr.	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3-way linear corr.			*		✓	*	*	*	*	*	*				✓
3-way non-linear	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1 good and noise			*	✓	✓	*	*	*	*	*	*			✓	✓
Un-nested					✓			*	*	*	*			✓	
Monotonic	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Two U's	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Multi-modal (XOR)	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
linear sep. Gaussian	y	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 4.5: Ability of different measures to determine the informative features. A check mark indicates that the measure value was largest for the feature set that contained only the informative features. A "*" is a value used for univariate and additive multivaraite measures where the informative features are selected first. The column "FS" indicates a data set whose best feature set includes all the features (and hence should be selected by default by the univariate and additive multivariate measures).

4.3.2 Correlation of filter measures with classifier accuracy

The performance of classifiers is different on different data sets. Selecting informative features is not beneficial if a classifier is unable to properly use them. Therefore, it is also important to assess the ability of filter measures to predict the accuracy of different classifiers. It is important to have a filter measure that not only identifies the best set, but is well correlated with accuracy overall so the search can be guided correctly.

Tables describing the accuracies of the classifiers on the different artificial and real data sets are available in appendix A.

Tables 4.6 and 4.7 present the correlation between each filter measure and KNN accuracy using the best performing parameters. The correlations for SVM one vs. one are presented in Tables 4.8 and 4.9. The SVM results for the one vs. all are presented in Tables 4.10 and 4.11.

For the KNN classifier, the filter measures that have the highest correlation are the consistency measure, CFS, CMIM and subset RELIEF. CFS and CMIM both only consider pairwise interactions, but in practice they work well on the real data sets. The consistency measure matches the KNN results well, especially on the artificial data sets, and has the

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	consis-tency	CFS
Binary 1	0.91	0.93	0.92	0.97	0.94	0.94	0.93	0.94	0.96	0.94	0.94	0.94	0.94	**0.99**	0.93	0.90
Binary 2	0.95	0.92	0.98	0.92	0.97	0.97	0.97	0.97	**1.00**	0.97	0.97	0.97	0.97	0.99	0.97	0.94
Binary 3	0.79	0.90	0.90	0.94	0.93	0.92	0.69	0.91	0.95	0.93	0.93	0.93	0.90	0.95	0.92	0.78
Monk 1	0.59	0.58	0.90	0.51	0.60	0.60	0.59	0.36	0.79	0.60	0.60	0.60	0.75	0.25	**0.91**	0.47
Monk 2	-0.49	**0.00**	-0.23	-0.32	-0.47	-0.47	-0.45	-0.53	-0.49	-0.47	-0.47	-0.47	-0.35	-0.30	-0.19	-0.29
Monk 3	0.98	0.64	0.85	0.72	0.98	**0.99**	0.98	0.95	0.90	0.98	0.98	0.98	0.77	0.60	0.81	0.95
Simple dependent	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00	1.00	1.00	0.97	1.00	1.00	1.00
Partially dependent	0.99	0.99	1.00	0.87	1.00	1.00	0.99	**1.00**	1.00	1.00	1.00	0.99	0.89	0.60	1.00	0.91
2-way linear corr.	0.55	0.99	0.99	0.61	0.98	0.99	0.35	0.88	0.35	1.00	1.00	0.98	1.00	**1.00**	1.00	0.99
3-way linear corr.	0.97	0.78	0.90	0.86	0.78	0.78	0.94	0.98	0.97	0.78	0.78	0.78	0.59	0.27	**0.98**	0.84
3-way non-linear corr.	0.88	0.88	0.95	0.99	0.88	0.88	0.88	0.89	0.81	0.88	0.88	0.88	0.88	0.86	**1.00**	0.92
1 good and noise	0.99	0.99	0.97	0.80	0.98	0.98	**0.99**	0.98	0.96	0.99	0.99	0.99	0.48	0.69	0.99	0.96
Pure noise	0.93	-0.51	-0.46	0.85	0.97	**0.97**	0.95	0.79	0.74	0.96	0.96	0.96	0.49	0.69	0.63	0.95
Un-nested	0.84	0.81	0.88	0.95	0.84	0.81	0.92	**0.99**	0.88	0.83	0.84	0.83	0.75	0.67	0.98	0.80
Monotonic	0.94	0.85	0.92	0.90	0.84	0.84	0.82	0.88	0.92	0.84	0.84	0.84	0.84	0.79	**0.96**	0.93
Two U's	0.99	0.98	1.00	0.98	1.00	0.99	0.99	1.00	0.99	1.00	1.00	1.00	1.00	0.98	1.00	1.00
Multimodal (XOR)	0.98	0.07	1.00	1.00	0.97	0.97	0.83	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00	0.87
linear sep. Guassian	0.99	**1.00**	0.98	0.97	0.99	1.00	0.98	1.00	1.00	0.99	1.00	1.00	1.00	1.00	0.97	1.00
Redundant features	**0%**	100%	39%	**0%**	100%	100%	**0%**	**0%**	**0%**	100%	100%	100%	100%	-	**0%**	22%
One cluster per class	2%	100%	47%	**0%**	100%	100%	**0%**	**0%**	**0%**	100%	100%	100%	100%	328%	**0%**	26%
best	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
worst	0.55	0.07	0.85	0.51	0.60	0.60	0.35	0.36	0.35	0.60	0.60	0.60	0.48	0.25	0.81	0.47

Table 4.6: Correlation of different filter measures with KNN accuracy, using best parameter set. The "worst" category selects the lowest correlation value, not including *Monk 2*. The *Monk 2* data set also has low correlation and is discussed in detail in Section C.3.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	0.74	0.00	0.00	0.00	0.70	0.70	0.23	0.76	0.49	0.69	0.71	0.69	**0.77**	0.00	0.77	0.74
Car Evaluation	0.50	0.48	0.47	0.29	0.51	0.51	0.51	0.53	**0.61**	0.49	0.47	0.48	0.36	0.59	0.42	0.40
Cardiotocography	0.84	-0.56	0.77	0.73	0.75	0.75	0.74	0.80	0.64	0.75	0.75	0.75	0.58	0.00	**0.93**	0.84
Congressional Voting	**0.96**	0.82	0.84	0.70	0.74	0.74	0.82	0.87	0.71	0.84	0.74	0.79	0.56	0.00	0.71	0.88
Contraceptive Choice	0.52	0.57	0.19	0.50	0.51	0.47	0.10	0.60	0.38	0.50	0.46	0.47	**0.61**	0.00	0.46	0.50
Credit Approval	0.90	0.54	0.84	**0.94**	0.90	0.89	0.87	0.91	0.91	0.90	0.90	0.89	0.66	0.00	0.76	0.79
Dermatology	**0.94**	0.67	0.88	0.84	0.77	0.76	0.80	0.87	0.66	0.77	0.78	0.77	0.76	0.00	0.94	0.90
E-coli	0.75	-0.30	0.91	0.66	0.89	0.88	0.94	**0.95**	0.88	0.80	0.84	0.80	0.18	0.00	0.90	0.89
Flag	0.53	-0.25	0.45	0.65	0.59	0.56	0.71	0.57	0.38	0.59	0.58	0.58	0.33	0.00	0.58	**0.78**
Glass	0.63	-0.16	0.75	0.00	0.68	0.65	0.53	**0.76**	0.67	0.68	0.70	0.68	0.31	0.00	0.76	0.62
Haberman's Survival	0.49	-0.11	0.04	0.05	0.63	0.61	**0.82**	0.41	0.48	0.63	0.63	0.63	0.50	0.40	0.26	0.73
Ionosphere	0.40	0.30	0.32	0.44	0.33	0.34	0.20	0.35	0.37	0.34	0.32	0.31	0.00	0.00	**0.60**	0.48
Iris	0.91	0.82	0.94	0.95	0.81	0.80	0.99	**0.99**	0.98	0.81	0.81	0.82	0.40	0.56	0.99	0.93
Mushroom	0.70	0.58	0.72	0.20	0.63	0.63	0.48	**0.83**	0.59	0.61	0.64	0.46	0.00	0.00	0.72	0.69
Page blocks	0.70	0.50	0.49	0.68	0.62	0.69	0.58	0.64	0.64	0.63	0.63	0.62	0.56	0.00	0.51	**0.74**
Post-op	0.61	0.00	0.00	0.00	0.45	0.33	0.45	0.57	0.54	0.63	0.59	0.57	0.37	0.57	**0.66**	0.32
Segmentation	0.84	0.55	0.74	0.69	0.64	0.64	0.74	0.82	0.60	0.64	0.64	0.64	0.42	0.00	**0.97**	0.85
Statlog (vehicle)	0.77	0.41	**0.86**	0.76	0.67	0.68	0.08	0.76	0.58	0.66	0.67	0.66	0.60	0.00	0.74	0.71
WI Breast Cancer (orig.)	0.82	0.69	**0.83**	0.64	0.74	0.73	0.73	0.79	0.71	0.73	0.74	0.74	0.58	0.00	0.77	0.80
WI Breast Cancer (diag.)	**0.94**	0.54	0.71	0.70	0.55	0.55	0.64	0.76	0.60	0.55	0.55	0.55	0.50	0.00	0.84	0.72
WI Breast Cancer (prog.)	0.20	0.21	0.47	**0.47**	0.19	0.19	-0.10	0.23	0.26	0.19	0.20	0.19	0.22	0.00	0.19	0.17
Wine	0.92	0.72	0.84	0.83	0.72	0.72	0.77	0.82	0.64	0.71	0.71	0.71	0.64	0.00	**0.92**	0.87
Yeast	0.74	-0.58	0.48	0.33	0.96	0.95	0.95	**0.96**	0.93	0.83	0.79	0.82	0.74	0.00	0.90	0.90
Best	0.96	0.82	0.94	0.95	0.96	0.95	0.99	0.99	0.98	0.90	0.90	0.89	0.77	0.59	0.99	0.93
worst	0.20	-0.58	0.00	0.00	0.19	0.19	-0.10	0.23	0.26	0.19	0.20	0.19	0.00	0.00	0.19	0.17

Table 4.7: Correlation of different filter measures with KNN accuracy on real data sets, using best parameter set

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	consis-tency	CFS
Binary 1	0.57	0.58	0.54	**0.91**	0.58	0.58	0.57	0.59	0.68	0.58	0.58	0.58	0.59	0.81	0.57	0.50
Binary 2	0.68	0.70	0.71	0.59	0.70	0.70	0.70	0.70	0.79	0.70	0.70	0.70	0.71	**0.87**	0.70	0.62
Monk 1	0.98	0.98	0.48	0.45	0.99	0.99	0.99	0.60	0.64	0.99	0.99	0.99	**0.99**	0.16	0.59	0.95
Monk 3	0.95	0.59	0.78	0.64	0.95	**0.95**	0.95	0.89	0.82	0.95	0.95	0.95	0.71	0.56	0.70	0.94
Simple dependent	1.00	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00	1.00	1.00	0.97	1.00	1.00	1.00
Partially dependent	1.00	0.99	1.00	0.88	1.00	1.00	0.99	1.00	**1.00**	1.00	1.00	1.00	0.87	0.56	1.00	0.92
2-way linear corr.	-0.05	0.70	0.72	0.02	0.69	0.69	-0.28	0.43	-0.28	0.77	0.82	0.92	**1.00**	0.81	0.97	0.69
3-way linear corr.	**0.99**	0.80	0.90	0.81	0.80	0.80	0.97	0.99	0.98	0.80	0.80	0.80	0.63	0.27	0.98	0.87
3-way non-linear corr.	0.89	0.88	0.95	0.99	0.88	0.88	0.88	0.89	0.81	0.88	0.88	0.88	0.87	0.85	**1.00**	0.92
1 good and noise	1.00	1.00	0.97	0.80	1.00	1.00	1.00	0.99	0.98	**1.00**	1.00	1.00	0.50	0.72	**1.00**	0.96
Pure noise	0.73	-0.49	-0.44	0.70	0.75	0.76	0.69	0.58	0.62	0.78	0.75	0.78	0.45	0.71	0.51	0.80
Un-nested	0.82	0.78	0.85	0.93	0.83	0.78	0.91	**0.99**	0.87	0.81	0.82	0.80	0.76	0.64	0.99	0.77
Monotonic	0.91	0.81	0.88	0.87	0.80	0.81	0.80	0.87	0.94	0.81	0.80	0.80	0.81	0.75	**0.97**	0.92
Two U's	1.00	0.94	1.00	0.94	1.00	1.00	0.99	1.00	1.00	1.00	**1.00**	0.99	1.00	1.00	1.00	1.00
Multimodal (XOR)	0.94	0.22	1.00	1.00	1.00	0.99	0.73	1.00	1.00	0.99	1.00	**1.00**	0.99	1.00	1.00	0.93
linear sep. Guassian	0.96	1.00	0.96	0.94	1.00	1.00	1.00	1.00	0.99	1.00	1.00	**1.00**	1.00	1.00	0.99	1.00
Redundant features	**0%**	100%	39%	**0%**	100%	100%	**0%**	**0%**	**0%**	100%	100%	100%	100%	-	**0%**	22%
One cluster per class	2%	100%	47%	**0%**	100%	100%	**0%**	**0%**	**0%**	100%	100%	100%	100%	328%	**0%**	26%
Binary 3	97%	100%	542%	331%	100%	100%	**18%**	652%	30%	100%	100%	100%	100%	162%	902%	36%
Monk 2	102%	-	-	-	100%	100%	82%	460%	855%	100%	100%	100%	100%	221%	-	**36%**
best	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
worst	-0.05	0.22	0.48	0.02	0.58	0.58	-0.28	0.43	-0.28	0.58	0.58	0.58	0.50	0.16	0.57	0.36

Table 4.8: Correlation of different filter measures with SVM one vs. one accuracy, using best parameter set. The "worst" category selects the lowest correlation value, not including *Monk 2*. The *Monk 2* data set also has low correlation and is discussed in detail in Section C.3.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	0.60	0.00	0.00	0.00	0.46	0.45	0.07	0.60	0.42	0.45	0.44	0.45	**0.63**	0.00	0.48	0.61
Car Evaluation	0.77	0.77	0.72	0.25	0.79	0.79	0.79	0.81	**0.88**	0.74	0.71	0.74	0.58	0.81	0.68	0.66
Cardiotocography	**0.92**	-0.55	0.90	0.72	0.86	0.87	0.88	0.89	0.77	0.85	0.84	0.86	0.73	0.00	0.87	0.92
Congressional Voting	**0.99**	0.82	0.88	0.73	0.78	0.78	0.84	0.91	0.74	0.86	0.78	0.83	0.62	0.00	0.77	0.90
Contraceptive Choice	**0.87**	0.46	-0.16	0.70	0.82	0.76	0.27	0.87	0.70	0.80	0.80	0.80	0.75	0.00	0.80	0.72
Credit Approval	**0.98**	0.47	0.81	0.85	0.91	0.91	0.93	0.94	0.93	0.91	0.91	0.91	0.54	0.00	0.64	0.93
Dermatology	0.92	0.64	0.85	0.82	0.73	0.73	0.76	0.83	0.62	0.73	0.74	0.73	0.72	0.00	**0.94**	0.88
E-coli	0.75	-0.30	0.89	0.66	0.90	0.89	0.95	**0.97**	0.87	0.80	0.85	0.80	0.16	0.00	0.94	0.92
Flag	0.70	-0.44	0.30	0.58	0.73	0.72	0.79	0.73	0.53	0.66	0.71	0.71	0.51	0.00	0.67	**0.82**
Glass	**0.78**	-0.16	0.75	0.00	0.76	0.74	0.71	0.76	0.70	0.76	0.73	0.75	0.40	0.00	0.72	0.73
Haberman's Survival	-0.86	**0.81**	0.11	-0.09	-0.68	-0.80	-0.76	0.19	-0.01	-0.67	-0.67	-0.67	-0.80	0.40	0.00	-0.92
Ionosphere	**0.87**	0.77	0.74	0.72	0.74	0.73	0.01	0.77	0.72	0.75	0.77	0.77	0.00	0.00	0.63	0.84
Iris	0.90	0.74	0.88	0.91	0.74	0.73	0.98	0.98	0.99	0.74	0.73	0.74	0.32	0.52	**0.99**	0.90
Mushroom	**0.86**	0.64	0.73	0.24	0.70	0.71	0.49	0.81	0.65	0.68	0.70	0.55	0.00	0.00	0.61	0.67
Page blocks	0.74	0.51	0.60	0.69	0.68	0.78	0.57	0.68	0.72	0.72	0.71	0.72	0.58	0.00	0.59	**0.80**
Post-op	-0.34	0.00	0.00	0.00	-0.24	-0.34	-0.20	-0.10	-0.28	-0.08	**0.05**	0.00	-0.67	-0.07	0.00	-0.32
Segmentation	0.90	0.62	0.82	0.73	0.73	0.73	0.80	0.87	0.70	0.73	0.74	0.73	0.53	0.00	**0.95**	0.86
Statlog (vehicle)	**0.93**	0.48	0.89	0.56	0.82	0.83	0.19	0.89	0.77	0.81	0.82	0.81	0.83	0.00	0.90	0.77
WI Breast Cancer (orig.)	**0.93**	0.67	0.81	0.60	0.71	0.70	0.74	0.81	0.67	0.71	0.71	0.72	0.56	0.00	0.72	0.82
WI Breast Cancer (diag.)	**0.94**	0.49	0.67	0.64	0.51	0.51	0.62	0.72	0.60	0.51	0.51	0.51	0.49	0.00	0.83	0.66
WI Breast Cancer (prog.)	-0.83	-0.79	-0.65	0.14	-0.77	-0.78	**0.79**	-0.75	-0.29	0.00	0.00	0.00	-0.82	0.00	0.00	-0.53
Wine	**0.94**	0.76	0.86	0.84	0.75	0.74	0.79	0.83	0.68	0.75	0.75	0.74	0.66	0.00	0.93	0.89
Yeast	0.71	-0.49	0.56	0.29	0.94	0.92	0.94	**0.95**	0.93	0.75	0.72	0.74	0.84	0.00	0.82	0.85
Best	0.99	0.82	0.90	0.91	0.94	0.92	0.98	0.98	0.99	0.91	0.91	0.91	0.84	0.81	0.99	0.93
worst	-0.86	-0.79	-0.65	-0.09	-0.77	-0.80	-0.76	-0.75	-0.29	-0.67	-0.67	-0.67	-0.82	-0.07	0.00	-0.92

Table 4.9: Correlation of different filter measures with SVM one vs. one accuracy on real data sets, using best parameter set

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	consis-tency	CFS
Binary 1	0.54	0.54	0.51	**0.90**	0.55	0.55	0.55	0.55	0.65	0.55	0.55	0.54	0.55	0.81	0.54	0.47
Binary 2	0.51	0.56	0.55	0.40	0.55	0.55	0.59	0.56	0.65	0.56	0.55	0.56	0.56	**0.78**	0.54	0.48
Monk 1	0.99	0.98	0.47	0.46	0.99	**0.99**	0.99	0.60	0.64	0.99	0.99	0.99	0.99	0.15	0.58	0.95
Monk 3	**0.95**	0.56	0.77	0.63	0.94	0.94	0.94	0.88	0.80	0.95	0.94	0.94	0.71	0.57	0.69	0.93
Simple dependent	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	**1.00**	1.00
Partially dependent	1.00	0.99	1.00	0.88	1.00	1.00	0.99	1.00	**1.00**	1.00	1.00	1.00	0.87	0.57	1.00	0.92
2-way linear corr.	0.69	**1.00**	1.00	0.74	1.00	1.00	0.50	0.95	0.50	1.00	1.00	1.00	1.00	1.00	0.98	1.00
3-way linear corr.	**1.00**	0.81	0.90	0.79	0.81	0.81	0.97	0.99	0.98	0.81	0.81	0.81	0.65	0.28	0.98	0.88
3-way non-linear corr.	0.89	0.88	0.95	0.99	0.88	0.88	0.88	0.89	0.81	0.88	0.88	0.88	0.87	0.85	**1.00**	0.92
1 good and noise	0.97	0.97	0.94	0.82	0.97	0.97	0.97	0.97	0.96	0.98	0.98	0.98	0.48	0.70	**0.98**	0.97
Pure noise	-0.60	**0.42**	0.41	-0.78	-0.36	-0.37	0.28	-0.14	-0.24	-0.45	-0.25	-0.45	-0.16	0.00	0.20	-0.38
Un-nested	0.85	0.85	0.92	0.97	0.85	0.84	0.90	0.96	0.87	0.85	0.85	0.85	0.72	0.66	**0.98**	0.88
Monotonic	0.78	0.80	0.83	0.82	0.81	0.81	0.80	0.83	**0.91**	0.81	0.81	0.81	0.81	0.73	0.89	0.86
Two U's	0.99	0.90	0.99	0.91	1.00	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Multimodal (XOR)	0.99	0.05	1.00	1.00	0.96	0.96	0.84	1.00	1.00	1.00	1.00	1.00	**1.00**	1.00	1.00	0.86
linear sep. Guassian	0.97	1.00	0.96	0.95	1.00	1.00	0.99	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00
Redundant features	0%	-65%	-71%	0%	-65%	-65%	0%	0%	0%	-65%	-64%	-65%	65%	0%	0%	**-82%**
One cluster per class	2%	100%	47%	**0%**	100%	100%	**0%**	**0%**	**0%**	100%	100%	100%	100%	328%	**0%**	26%
Binary 3	97%	100%	542%	331%	100%	100%	**18%**	652%	30%	100%	100%	100%	100%	162%	902%	36%
Monk 2	102%	-	-	-	100%	100%	82%	460%	855%	100%	100%	100%	100%	221%	-	**36%**
best	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
worst	0.51	0.05	0.47	0.40	0.55	0.55	0.18	0.55	0.30	0.55	0.55	0.54	0.48	0.15	0.54	0.36

Table 4.10: Correlation of different filter measures with SVM one vs. all accuracy, using best parameter set. The "worst" category selects the lowest correlation value, not including *Monk 2*. The *Monk 2* data set also has low correlation and is discussed in detail in Section C.3.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	0.27	0.00	0.00	0.00	0.27	0.27	0.24	0.24	0.23	**0.31**	0.28	0.31	0.22	0.00	0.23	0.19
Car Evaluation	0.70	0.73	0.67	0.13	0.74	0.74	0.74	0.77	**0.82**	0.67	0.64	0.66	0.54	0.77	0.60	0.61
Cardiotocography	**0.93**	-0.57	0.91	0.72	0.88	0.89	0.90	0.90	0.79	0.87	0.86	0.87	0.76	0.00	0.87	0.92
Congressional Voting	**0.99**	0.82	0.88	0.73	0.77	0.78	0.84	0.91	0.74	0.85	0.77	0.82	0.61	0.00	0.76	0.90
Contraceptive Choice	**0.83**	0.48	-0.17	0.62	0.82	0.81	0.07	0.78	0.68	0.82	0.82	0.82	0.63	0.00	0.67	0.82
Credit Approval	**0.97**	0.47	0.80	0.85	0.90	0.91	0.93	0.94	0.92	0.91	0.90	0.91	0.54	0.00	0.64	0.93
Dermatology	0.91	0.62	0.82	0.79	0.71	0.71	0.74	0.82	0.60	0.71	0.72	0.71	0.70	0.00	**0.95**	0.87
E-coli	0.74	-0.33	0.85	0.64	0.87	0.87	0.93	**0.94**	0.85	0.78	0.82	0.78	0.15	0.00	0.91	0.89
Flag	0.87	-0.70	0.19	0.52	0.87	0.86	**0.89**	0.88	0.76	0.84	0.84	0.84	0.73	0.00	0.83	0.86
Glass	**0.87**	-0.17	0.79	0.00	0.80	0.79	0.78	0.82	0.74	0.80	0.78	0.80	0.47	0.00	0.75	0.80
Haberman's Survival	-0.96	**0.52**	-0.16	-0.59	-0.72	-0.86	-0.83	0.22	-0.08	-0.70	-0.70	-0.70	-0.93	0.23	0.00	-0.83
Ionosphere	**0.87**	0.78	0.75	0.71	0.75	0.74	0.00	0.78	0.72	0.75	0.77	0.77	0.00	0.00	0.63	0.84
Iris	0.96	0.81	0.95	0.93	0.80	0.79	0.96	0.98	**0.98**	0.80	0.79	0.80	0.48	0.59	0.98	0.90
Mushroom	**0.86**	0.64	0.72	0.24	0.70	0.71	0.49	0.81	0.65	0.68	0.70	0.55	0.00	0.00	0.61	0.67
Page blocks	**0.86**	0.63	0.73	0.64	0.82	0.86	0.73	0.82	0.85	0.82	0.82	0.81	0.68	0.00	0.71	0.79
Post-op	-0.15	0.00	0.00	0.00	-0.03	-0.15	0.04	0.11	-0.04	0.15	0.30	0.12	-0.63	0.20	**0.33**	-0.16
Segmentation	0.94	0.65	0.86	0.66	0.79	0.79	0.84	0.90	0.77	0.78	0.80	0.78	0.59	0.00	**0.96**	0.86
Statlog (vehicle)	**0.95**	0.49	0.87	0.53	0.84	0.85	0.18	0.90	0.79	0.84	0.84	0.83	0.85	0.00	0.90	0.79
WI Breast Cancer (orig.)	**0.93**	0.67	0.81	0.60	0.71	0.71	0.74	0.81	0.68	0.71	0.71	0.72	0.56	0.00	0.72	0.83
WI Breast Cancer (diag.)	**0.94**	0.50	0.67	0.64	0.52	0.52	0.62	0.72	0.61	0.51	0.52	0.51	0.50	0.00	0.84	0.66
WI Breast Cancer (prog.)	-0.81	-0.78	-0.65	0.12	-0.74	-0.74	**0.77**	-0.73	-0.27	0.00	0.00	0.00	-0.81	0.00	0.00	-0.49
Wine	**0.96**	0.76	0.87	0.85	0.74	0.74	0.79	0.84	0.68	0.74	0.74	0.73	0.67	0.00	0.93	0.87
Yeast	0.42	-0.24	0.53	0.15	0.66	0.64	0.68	0.68	0.71	0.45	0.45	0.46	**0.82**	0.00	0.59	0.56
Best	0.99	0.82	0.95	0.93	0.90	0.91	0.96	0.98	0.98	0.91	0.90	0.91	0.85	0.77	0.98	0.93
worst	-0.96	-0.78	-0.65	-0.59	-0.74	-0.86	-0.83	-0.73	-0.27	-0.70	-0.70	-0.70	-0.93	0.00	0.00	-0.83

Table 4.11: Correlation of different filter measures with SVM one vs. all accuracy on real data sets, using best parameter set

benefit of being relatively computationally inexpensive. However, the consistency measure does not separate noisy and non-informative features, and is also quite dependent on the N_b parameter, which can affect performance greatly. Subset RELIEF is the most closely related to the KNN classifier. However, it is almost as computationally expensive as a wrapper.

For SVM, CMIM performs well on the real data sets, and is relatively tolerant to changes of the parameter settings. CFS and Fisher's also perform relatively well, and Fisher's does not require parameter setting. The consistency measure performs relatively well on the artificial data sets, but the performance is much worse on the real sets, and the variance is large between different parameter settings.

None of the univariate measures perform well for either classifier. For a data set with a reasonably small number of features, a subset based filter measure with a good search technique would be preferable. The univariate measures have a particular problem with redundant features, adding multiple unnecessary features to a set.

None of the filter measures work very well for non-monotonic sets. This is unfortunate, because most of the real world data sets appear to be non-monotonic to some degree. Even the subset based measures, such as Fisher's or subset-RELIEF tend to prefer feature sets with a larger number of features because they are distance based.

RELIEF family

The RELIEF measures are closely related to the KNN classifier and are better correlated with KNN accuracy than SVM. The subset RELIEF is generally the best performing RELIEF family measure for both KNN and SVM.

The univariate RELIEF-F does have an advantage over some univariate measures because it can take negative values and can therefore be used for non-monotonic sets. In the *noise with one good feature* data set, the noise features all take negative values, indicating that these features are undesirable. This also correlates very well with the KNN classifier, where adding noisy features decreases accuracy.

In the *Monk 1* data set subset RELIEF identifies two informative features that the SVM is unable to use. The SVM one vs. rest formulation suffers from a similar problem with the *one good feature and noise*, *monotonic* and *redundant* data sets. The SVM one vs. rest is unable to make use of the informative feature and all of the RELIEF family measures overestimate the value of these features for this classifier.

The univariate measure fails completely on the *multi-modal* set. The count-based and the subset-based RELIEF are much better choices for this data set for both the KNN and

	SVM (one vs. one)			SVM (one vs. all)			KNN		
	univariate	subset	count	univariate	subset	count	univariate	subset	count
	K=10	K=1	K=1	K=10	K=1	K=1	K=10	K=1	K=1
Binary 1	0.57	0.54	**0.91**	0.54	0.50	**0.90**	0.92	0.92	**0.97**
Binary 2	0.70	**0.71**	0.59	**0.56**	0.55	0.40	0.92	**0.98**	0.92
Binary 3	**100%**	2521%	331%	**100%**	2521%	331%	0.86	0.90	**0.94**
Monk 1	**0.98**	0.48	0.45	**0.98**	0.47	0.46	0.58	**0.90**	0.51
Monk 2	-	-	-	-	-	-	**0.00**	-0.23	-0.32
Monk 3	0.59	**0.69**	0.64	0.56	**0.67**	0.63	0.64	**0.81**	0.72
Simple dependent	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Partially dependent	**0.99**	**0.99**	0.88	**0.99**	**0.99**	0.88	0.99	**1.00**	0.87
2-way linear corr.	**0.70**	0.64	0.02	**1.00**	**1.00**	0.74	**0.99**	0.97	0.61
3-way linear corr.	0.80	**0.90**	0.81	0.81	**0.90**	0.79	0.78	**0.90**	0.86
3-way non-linear corr.	0.88	0.95	**0.99**	0.88	0.95	**0.99**	0.88	0.95	**0.99**
1 good and noise	0.94	**0.97**	0.80	0.92	**0.94**	0.82	0.94	**0.97**	0.80
Pure noise	-0.50	-0.47	**0.70**	0.32	**0.33**	-0.78	-0.52	-0.50	**0.85**
Un-nested	0.78	0.83	**0.93**	0.85	0.90	**0.97**	0.81	0.86	**0.95**
Monotonic	0.81	**0.87**	**0.87**	0.80	**0.83**	0.82	0.84	**0.91**	0.90
Two U's	0.94	**1.00**	0.94	0.90	**0.98**	0.91	0.98	**1.00**	0.98
Multimodal (XOR)	-0.98	**1.00**	**1.00**	-0.93	**1.00**	**1.00**	-0.93	**1.00**	**1.00**
linear sep. Guassian	**0.98**	0.95	0.94	**0.98**	0.95	0.95	**1.00**	0.98	0.97
Redundant	100%	39%	**0%**	-0.65	-0.71	**0.00**	100.00%	38.93%	**0.00%**
2 clusters	100%	53%	**0%**	100%	53%	**0%**	100%	53%	**0%**

Table 4.12: Correlations between filter values and SVM and 1-NN accuracy for RELIEF-F, subset-based RELIEF and count-based RELIEF

the SVM classifiers. Similarly, the univariate measures fail to identify the two dependent features in *Monk 1* as informative. This will be the case for any dependent feature with a univariate measure.

Overall, subset-based RELIEF outperforms the count-based and univariate RELIEF measures for KNN classifiers, as shown in Tables 4.12 and 4.13. None of the RELIEF-based measures appear to be the best choice for the SVM classifiers.

Fishers

Fisher's interclass separability criterion theoretically works well for SVM because both prefer feature sets that are compact and have a large margin of separation. Fisher's achieves the highest correlation on 11 of the 20 real data sets for the SVM classifier.

Fisher's does, however, have a slight preference for larger feature sets, because adding another spatial dimension increases the possible between class distance.

Fisher's interclass separability criterion performs fairly well for the KNN classifier. The exceptions are the *two-way linear correlated*, *Monk 1*, and *Binary 3* data sets. In the *two-way linear correlated* set, the two-feature set is slightly better than either of the single

	SVM (one vs. one)			SVM (one vs. all)			KNN		
	univariate	subset	count	univariate	subset	count	univariate	subset	count
	K=10	K=1	K=1	K=10	K=1	K=1	K=10	K=1	K=1
Abalone	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Car Evaluation	**0.77**	0.70	0.25	**0.73**	0.66	0.13	**0.48**	0.43	0.29
Cardiotocography	-0.71	**0.87**	0.72	-0.73	**0.88**	0.72	-0.68	**0.77**	0.73
Congressional Voting	**0.81**	0.80	0.73	**0.81**	0.80	0.73	**0.80**	0.76	0.70
Contraceptive Choice	0.12	-0.16	**0.70**	0.20	-0.17	**0.62**	0.43	0.19	**0.50**
Credit Approval	0.46	0.71	**0.85**	0.46	0.70	**0.85**	0.54	0.79	**0.94**
Dermatology	0.63	**0.85**	0.82	0.61	**0.82**	0.79	0.65	**0.88**	0.84
E-coli	-0.31	**0.88**	0.66	-0.36	**0.83**	0.64	-0.31	**0.90**	0.66
Flag	-0.46	0.30	**0.58**	-0.72	0.19	**0.52**	-0.27	0.45	**0.65**
Glass	-0.55	**0.75**	0.00	-0.53	**0.79**	0.00	-0.54	**0.75**	0.00
Haberman's Survival	**0.44**	0.10	-0.09	**0.23**	-0.17	-0.59	-0.11	-0.02	**0.05**
Ionosphere	0.72	**0.74**	0.72	0.73	**0.75**	0.71	0.23	0.32	**0.44**
Iris	0.74	0.84	**0.91**	0.81	**0.93**	**0.93**	0.82	0.91	**0.95**
Mushroom	0.64	**0.72**	0.24	0.64	**0.72**	0.24	0.58	**0.69**	0.20
Page blocks	0.50	0.60	**0.69**	0.62	**0.73**	0.64	0.48	0.49	**0.68**
Post-op	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Segmentation	0.62	**0.81**	0.73	0.65	**0.86**	0.66	0.55	**0.73**	0.69
Statlog (vehicle)	-0.79	**0.89**	0.56	-0.80	**0.87**	0.53	-0.58	**0.84**	0.76
WI Breast Cancer (orig.)	0.67	**0.73**	0.60	0.67	**0.73**	0.60	0.69	**0.77**	0.64
WI Breast Cancer (diag.)	0.49	**0.67**	0.64	0.50	**0.67**	0.64	0.54	**0.71**	0.70
WI Breast Cancer (prog.)	-0.81	-0.65	**0.14**	-0.80	-0.65	**0.12**	0.20	**0.47**	**0.47**
Wine	0.72	**0.85**	0.84	0.71	**0.86**	0.85	0.69	**0.83**	**0.83**
Yeast	-0.69	**0.56**	0.29	-0.45	**0.53**	0.15	-0.78	**0.48**	0.33

Table 4.13: Correlations between filter values and SVM and 1-NN accuracy for RELIEF-F, subset-based RELIEF and count-based RELIEF

feature sets. The filter measure reflects this, but identifies the wrong single feature set as being better. This is not really a significant problem, but because the correlation is only between three points, this small error causes a large decrease in the correlation value.

The *Monk 1* data set is difficult for the Fisher's measure, because Fisher's assumes compact, unimodal clusters. Not only is the *Monk 1* data set multi-modal, but one of the clusters is elongated. Fisher's measure underestimates the separability of this data set for KNN but not for the SVM. A similar problem occurs with the multi-modal *binary 3* for KNN.

Overall, Fisher's is better for SVM and may not identify certain features that are usable by the KNN classifier. In particular, Fisher's will not rank multi-modal features well. This is the opposite problem of RELIEF, which does not work well for SVM because it does rank these features highly.

Probability-based measures

The Gaussian measures perform reasonably well for both KNN and SVM on most of the artificial data sets. However, they are outperformed by the discrete form on both the real and artificial data sets. This may be an indication that at least some of the classes are not well represented by a Gaussian.

The different divergence measures are fairly comparable for both the artificial and real data sets on both classifiers, as seen in Tables 4.14 and 4.15. The Jensen-Shannon measure is the easiest to use since it does not become undefined if there are bins with a probability of zero. Hence, Jensen-Shannon would be the best choice if a probability measure was desirable. Unfortunately, none of the probability-based measures work very well on the real data sets for either classifier.

The Gaussian measures perform quite well for both KNN and SVM on most of the artificial data sets. The two major exceptions are the *un-nested* and *three way linear correlated* sets, for both the KNN and SVM classifiers. For the real data sets, the Gaussian form has a lower correlation than the discrete form in most cases for both the KNN and the SVM classifiers (see Tables B.11, B.12, B.13 and B.14). This is likely because some classes are not well represented by a Gaussian. There are some data sets where the Gaussian performs slightly better, but in many cases the performance is much worse. The discrete form appears to be a better choice.

Since these measures are univariate, none work well with non-monotonic sets or sets with dependent features. The *three-way non-linear correlated* data set is non-monotonic

	Probability measures ($N_b = 10$)								
	SVM (one vs. one)			SVM (one vs. all)			KNN		
	KL	JS	Bh.	KL	JS	Bh.	KL	JS	Bh.
Binary 1	0.58	0.58	0.58	0.54	0.54	0.54	0.94	0.94	0.94
Binary 2	0.70	0.70	0.70	0.56	0.55	0.56	0.94	0.95	0.94
Binary 3	100%	100%	100%	100%	100%	100%	0.93	0.93	0.93
Monk 1	0.98	0.98	0.98	0.98	0.98	0.98	0.59	0.59	0.59
Monk 2	100%	100%	100%	100%	100%	100%	-0.56	-0.56	-0.56
Monk 3	0.91	0.95	0.94	0.90	0.94	0.92	0.95	0.98	0.97
Simple dependent	1.00	1.00	0.99	1.00	1.00	0.99	1.00	1.00	0.99
Partially dependent	1.00	1.00	0.99	1.00	1.00	0.99	1.00	1.00	0.99
2-way linear corr.	0.70	0.68	0.67	1.00	1.00	1.00	0.99	0.98	0.98
3-way linear corr.	0.80	0.80	0.79	0.81	0.81	0.80	0.77	0.77	0.77
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.97	0.94	0.98	0.94	0.92	0.95	0.96	0.93	0.96
Pure noise	0.62	0.53	0.62	-0.45	-0.33	-0.45	0.80	0.68	0.80
Un-nested	0.81	0.82	0.80	0.85	0.83	0.85	0.83	0.84	0.83
Monotonic	0.80	0.80	0.80	0.81	0.81	0.81	0.84	0.84	0.84
Two U's	0.98	1.00	0.97	0.99	1.00	0.95	0.94	0.99	1.00
Multimodal (XOR)	0.87	1.00	1.00	0.78	0.96	0.98	0.79	0.97	0.98
linear sep. Guassian	0.98	0.97	0.98	0.98	0.98	0.99	0.99	0.99	1.00
Redundant	100%	100%	100%	-0.65	-0.65	-0.65	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table 4.14: Correlation between filter measure and 1-NN and SVM accuracy for various probability divergence measures

for both the KNN and SVM classifier. The probability based measures successfully detect the more powerful feature, and show that the less predictive features are better when used in combination. However, the measures are unable to deal with the non-monotonic nature of the data set, and give larger values when all three features are used together, when in fact adding the two less predictive features decreases the accuracy. A similar problem occurs for the *un-nested* data set, where the probability measures overestimate the benefit of adding new features.

Overall, none of the probability measures can be recommended for either classifier. The closely related mutual information measures are better choices with only a small additional computational expense.

Mutual Information measures

There are five tested mutual information based measures: two univariate (mutual information and symmetric uncertainty) and three additive, pairwise multivariate (conditional mutual information maximization, minimal-redundancy maximal-relevancy and first order utility).

	Probability measures ($N_b = 10$)								
	SVM (one vs. one)			SVM (one vs. all)			KNN		
	KL	JS	Bh.	KL	JS	Bh.	KL	JS	Bh.
Abalone	0.39	0.44	0.39	0.26	0.26	0.26	0.67	0.71	0.68
Car Evaluation	0.71	0.66	0.70	0.63	0.57	0.62	0.47	0.45	0.47
Cardiotocography	0.85	0.84	0.86	0.87	0.86	0.87	0.75	0.75	0.75
Congressional Voting	0.80	0.77	0.82	0.80	0.77	0.82	0.77	0.73	0.79
Contraceptive Choice	0.80	0.80	0.80	0.82	0.82	0.82	0.50	0.46	0.47
Credit Approval	0.88	0.87	0.89	0.88	0.87	0.89	0.89	0.89	0.89
Dermatology	0.73	0.74	0.73	0.71	0.71	0.71	0.77	0.77	0.77
E-coli	0.77	0.85	0.76	0.75	0.82	0.74	0.77	0.84	0.76
Flag	0.61	0.63	0.62	0.82	0.81	0.82	0.45	0.51	0.45
Glass	0.71	0.73	0.71	0.77	0.78	0.77	0.68	0.70	0.68
Haberman's Survival	-0.73	-0.70	-0.73	-0.83	-0.86	-0.83	0.55	0.47	0.55
Ionosphere	0.66	0.65	0.44	0.66	0.66	0.44	0.19	0.18	0.13
Iris	0.70	0.69	0.71	0.77	0.77	0.77	0.77	0.76	0.78
Mushroom	0.68	0.70	0.45	0.68	0.70	0.45	0.61	0.64	0.32
Page blocks	0.72	0.71	0.72	0.82	0.82	0.81	0.63	0.63	0.62
Post-op	-0.12	-0.61	-0.69	-0.05	-0.64	-0.74	0.51	0.30	0.24
Segmentation	0.72	0.73	0.72	0.78	0.78	0.77	0.63	0.63	0.63
Statlog (vehicle)	0.81	0.82	0.81	0.84	0.84	0.83	0.66	0.66	0.66
WI Breast Cancer (orig.)	0.68	0.68	0.68	0.69	0.68	0.68	0.71	0.71	0.70
WI Breast Cancer (diag.)	0.51	0.51	0.51	0.51	0.52	0.51	0.55	0.55	0.55
WI Breast Cancer (prog.)	-0.83	-0.83	-0.83	-0.81	-0.81	-0.81	0.19	0.19	0.19
Wine	0.73	0.73	0.73	0.73	0.73	0.73	0.70	0.70	0.70
Yeast	0.74	0.69	0.73	0.43	0.38	0.41	0.83	0.79	0.82

Table 4.15: Correlation between filter measure and 1-NN and SVM accuracy for various probability divergence measures on real data sets

The univariate measures only outperform the multivariate measures on one real data set using KNN or one vs. one SVM. They outperform multivariate measures on three data sets using one vs. all SVM. The two univariate measures are comparable for all the classifiers, with the mutual information measure slightly outperforming the symmetric uncertainty measures. This is likely because all the features are quantized to the same values.

Mutual information tends to rank features with a larger number of nominal values more highly than symmetric uncertainty [83]. To illustrate the difference between mutual information and symmetric uncertainty, consider three different features that exactly divide a data set. The first feature can take values from one to ten, and the class is one when the value is six or higher. The second feature can take values of zero or one and the class is one when the feature is one. The third feature can take values from one to ten and the class is one when the value is even. The mutual information between the feature and the class for all of these features is exactly 1. However, the symmetric uncertainty for features one and three (the features with ten nominal values) is 0.46, whereas the symmetric uncertainty for the second feature is exactly 1. Symmetric uncertainty ranks the simpler feature more highly, despite the fact that any feature can be used to separate the data set perfectly when using a KNN classifier. This may or may not be a desirable trait for a filter measure, and using a simpler feature may be beneficial for computational reasons. It is important to note, however, that neither the mutual information nor symmetric uncertainty differentiates between the two ten valued features, despite the different structure. While KNN is able to perfectly separate the data set with either, a linear SVM would only work with feature one.

CMIM outperforms the other mutual information based measures on 18 of the 24 real data sets with the KNN classifier. CMIM also performs well with the SVM classifiers, outperforming the other mutual information based measures in 14 of the 24 data sets when using one vs. all, and in 17 of the 24 data sets when using one vs. one. However, CMIM cannot always detect dependent features, even though it uses the conditional mutual information. Although the conditional mutual information is higher for pairwise dependent features, in the CMIM filter measure the conditional mutual information is always calculated with respect to the selected set of features, with the goal of eliminating correlated features rather than selecting dependent features. Regardless, CMIM still outperforms mRMR and FOU.

Both the mRMR and FOU include a term that accounts for the mutual information between the features in the set. It is possible for this term to be larger than the mutual information between the feature and the class, giving a negative value for the measure. Consider, for example, the two way linear correlated data set. The mutual information between the two features is higher than the mutual information between each feature and

the class. For both FOU and mRMR, this gives a negative value for the second feature added to the set. A completely random feature would have zero mutual information with the class and with the other features. In these cases, the completely random feature would be selected over the correlated feature. This is extremely undesirable.

A different formulation, where the values are divided rather than subtracted, can avoid this problem. CFS, which is a mutual information-based measure of this form, works better than mRMR and FOU. It does not, however, contain a conditional mutual information term.

Consistency

Although the consistency measure is interesting, it has a number of faults that render it unsuitable for use in most data sets. The consistency measure does correctly reject redundant features, and is capable of differentiating between a truly redundant feature and features that are correlated but still improve accuracy. It is also one of the few measures that can be less computationally expensive than a KNN wrapper. However, it implicitly assumes a monotonic set and although the consistency measure can give a rough estimate of the accuracy, it has difficulty differentiating between similar features. For most data sets, the linear correlation is higher than the rank correlation, indicating that the general trends are correct, but small accuracy changes are not correctly distinguished. The quantization impacts the performance greatly and there is no theoretically sound way to set his parameter. Selecting the quantization to match the number of clusters in the set also does not work well, as in most cases N_b is too low to capture detail in the features.

Most importantly, the consistency measure does not work well with noisy features, as noisy features can easily separate samples without being informative about the class. Consider, for example, a feature that is different for every sample, such as an ID number. This feature will have a perfect consistency score, which is highly misleading about its utility for classification.

For these reasons, consistency cannot be recommended as a good filter measure choice.

Laplacian Score

Overall, the Laplacian score performs poorly. On the real data sets, the highest correlation it achieves is only 0.77 for KNN, and 0.85 for the SVMs, with most data sets having much lower correlation. This does not appear to be a good filter measure choice for KNN or

SVM classifiers as it is likely that the correlation between the features and the accuracy will be low.

This low correlation is likely because this is one of the few unsupervised feature selection measures and therefore the class differences are not accounted for when ranking the features. This is particularly problematic for the artificial sets, where in most cases the class division is not related to the underlying structure.

Consider, for example, the *Monk* data sets. Because the features are the same across all three *Monk* data sets, the features are ranked in the same order. This is clearly not desirable, since the informative features are different for the different sets. Additionally, discrete features with only two values will often have a Laplacian score of zero, regardless of the feature utility. For these features, points with different values are not neighbours and points that are neighbours have the same value and therefore also have a zero score. This indicates a potential problem with neighbourhood graph based techniques when using discrete features.

For both the KNN and SVM classifiers, the Laplacian score works best when neighbourhood size parameter k is relatively large. This may be because none of the tested artificial data sets really have an underlying structure. If the data set does have an underlying structure, using a neighbourhood that is too large can cause "short-circuiting" [9], where points that are distant on the underlying structure are considered neighbours. For data sets with no underlying structure, using a larger neighbourhood just gives a better representation of the entire data set. Larger k values also work well on the real data sets, which may be an indication that many of the real data sets also have no underlying structure. A similar observation is made for the Isomap algorithm [149] where short-circuiting is found to help performance on many real data sets [155].

Overall, the performance of the Laplacian score is poor. The supervised measures are better correlated with classifier accuracy and should be used when possible.

Multi-cluster feature selection

Overall, the performance of the MCFS filter measure is quite poor for both classifiers, for similar reasons as the Laplacian score. One major problem with the MCFS filter measure is the transformation to the underlying manifold, which requires solving an eigenvalue problem. The matrices used are not always well conditioned, which makes this filter measure unsuitable for some data sets. This is a particular problem for the real data sets, where the eigenvalue problem cannot be solved for a large number of the sets. It is also one of the more computationally complex filter measures. Overall, MCFS appears to

be a poor filter choice for either classifier, due to numerical problems, the computational expense and the low correlation between the filter scores and the classifier accuracy on both the artificial and real data sets.

Correlation Feature selection (CFS)

Although this measure is called correlation feature selection, it uses symmetric uncertainty as a measure of feature correlation. CFS is actually a slightly different formulation of the mRMR measure, instead of subtracting the feature-feature mutual information, the two terms are divided. It is also a grouped measure. This circumvents the problem of correlated features taking negative values, making this filter measure more stable and less prone to selecting random features.

On the real data sets, CFS outperforms symmetric uncertainty in most cases, and actually performs quite well overall. With a quantization of 10 bins, CFS gives the best overall correlation on the flag data set, and is one of the 10 best performing measures for 10 of the 24 data sets. CFS likely performs better on the real data sets because real data sets are more likely to have some correlation between the features, which is the major strength of CFS.

CFS does increase slightly as the redundant features are added, but the measure value is not doubled when a second feature is added, as it is when symmetric uncertainty is used alone. Because CFS accounts for feature-feature information, the measure value increases only slightly when a fully redundant feature is added. CFS also performs better than the simple symmetric uncertainty on the monotonic set and on the correlated sets, where the symmetric uncertainty overestimates the benefit of adding new features.

Overall, CFS is a fairly good choice, particularly for the KNN classifier.

4.3.3 A note on the *Monk 2* data set

None of the tested filter measures work well for the *Monk 2* data set. For the *Monk 2* data sets, the class is 1 when exactly two classes take the value one. This means that each cluster of class 1 values is surrounded by clusters of class 0 values. Because this set is multi-modal, SVM fails completely, achieving the same accuracy with any feature set. KNN also performs poorly even with all six features. This is due to sampling issues. Using all six informative features gives a KNN classification accuracy of just 84.1%.

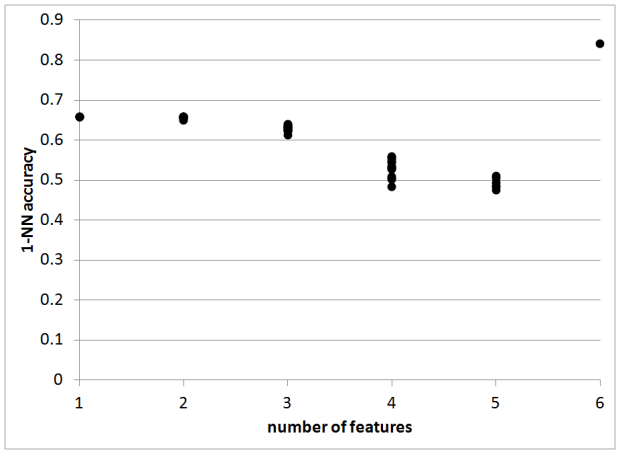


Figure 4.1: 1-NN accuracy on the *Monk 2* data set for feature sets with different numbers of features

The KNN classifier correlations are also exceptionally low. Although this is a fairly difficult data set, it is actually the classifier itself that struggles in this case. Because the classifier reacts in a strange way, the filter measures have difficulty predicting the outcome.

The KNN accuracy values are obtained using the Matlab version of the KNN function. This function uses the built-in min function to identify the nearest neighbour. If more than one point has the same minimum value, the function returns the first point in the list. Because the features in *Monk 2* do not include noise, more than one point will often share the same distance, especially when a reduced feature set is used. In these cases, the returned class will always be the same, as the list is in the same order. This may or may not be the majority class for that set of points. For the 1-NN classifier, the accuracy actually decreases as more features are added to the data set, until all six features are used (see Fig. 4.1). For this data set, even a wrapper measure would be misleading, as it would indicate that adding additional features to a small set would decrease the accuracy.

The KNN classifier accuracies would be slightly more predictable if the class of a tied point was selected stochastically. This can be simulated by adding a small amount of noise to each of the features in the data set. A new data set is tested using 1% uniform random noise to the features. As seen in Figure 4.2, when noise is added, adding features tends to increase the accuracy, which better matches with the predictions of the filter measures. Table 4.16 shows the correlation between KNN accuracy for the regular *Monk 2* data set, and a version of the *Monk 2* data set with 1% added noise. The filter measures in this data set are all designed to be positively correlated with the accuracy (i.e. the filter measure

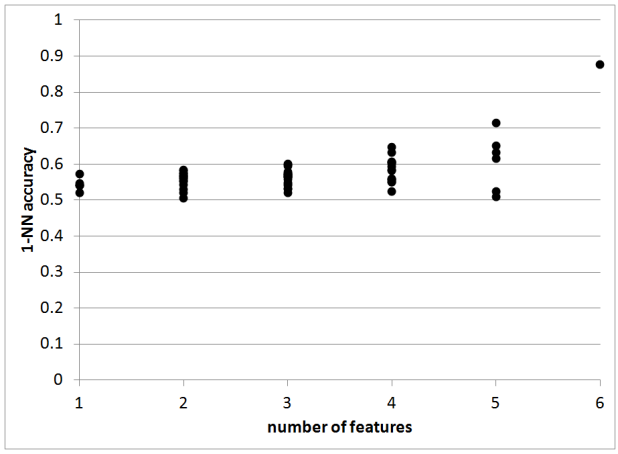


Figure 4.2: 1-NN accuracy on the *Monk 2* data set with 1% uniform random noise added to each feature

increases as the accuracy improves), but for the *Monk 2* data set, all the correlation values are negative. When the noise is added to the data set, the correlations become positive. The only exception is the Fisher measure, which does not perform well on any multi-modal data sets. Although the correlation values are still quite low, a positive correlation is still more desirable than a negative correlation.

4.3.4 Parameter sensitivity

The majority of the tested measures require one or more parameters to be tuned. The sensitivity to the parameter settings is an important practical consideration for selecting a measure. Even if a measure is able to achieve good performance under ideal parameter settings, if the measure is sensitive to parameter changes it may select a poor set of features if it is not well tuned. These parameters can be tuned using a validation set, but this adds an additional layer of complexity.

The ideal feature selection measure would return good results over a large range of parameter settings. This would allow researchers to set the parameters arbitrarily at first, in order to tune the classifier and determine what level of accuracy is reasonable for the problem. If necessary, the measure parameters could then be more finely tuned.

Each of the filter measures was tested over a range of parameter settings. Tables 4.17 and 4.18 show the variance of the correlation between the measures and KNN accuracy as the parameters are changed. Tables 4.19, 4.20, 4.21 and 4.22 show the variance of the

	Monk 2	
filter measure	without noise	with noise
Fishers Qb_new, J1	0.59	0.30
RELIEF-F (K=10)	0.00	0.42
subset RELIEF (K=1)	-0.23	0.85
count-relief	-0.32	0.76
KL-divergence ($N_b = 10$)	-0.56	0.39
JS-divergence ($N_b = 10$)	-0.56	0.31
Bhattacharyya ($N_b = 10$)	-0.56	0.39
Mutual information ($N_b = 10$)	-0.56	0.31
Symmetric uncertainty ($N_b = 10$)	-0.57	0.34
mRMR ($N_b = 10$)	-0.54	0.30
CMIM ($N_b = 10$)	-0.59	0.49
FOU ($N_b = 10$)	-0.55	0.61
consistency ($N_b = 10$)	-0.21	0.64

Table 4.16: Correlation between filter values and 1-NN accuracy values for the *Monk 2* data set with and without 1% added noise

correlation between the measures and the SVM one vs. one and one vs. all accuracy as the parameters are changed. Detailed tables showing the classifier correlation for each of the measures with each of the tested parameters are presented in Appendix B.

RELIEF family of measures

The RELIEF family of measures all use a single parameter, K, which controls the number of neighbouring points used to generate the nearest hit and miss values.

Different K values do not appear to have a large effect on the performance of the measure for either the univariate or subset-based RELIEF measures for either classifier (see Tables B.1, B.2 B.3 and B.4 in appendix B).

For the KNN classifier, increasing the K value of the univariate measure appears to slightly improve accuracy. This may be because averaging over K points reduces the effect of noise. For the subset-based RELIEF, in most cases the highest KNN correlations occur when $K = 1$ for the artificial data sets. However subset-RELIEF tends to work best with a large K value (K=10) on the real data sets.

For SVM, the results are more varied, with larger values of K improving the correlation for some data sets, but decreasing the correlation in others. Overall, however, the choice of K does not appear to have a large effect on the correlation.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Binary 1	-	0.00	0.02	-	0.02	0.02	0.86	0.02	0.18	0.03	0.01	0.03	0.03	0.07	0.03	0.05
Binary 2	-	0.17	0.01	-	0.01	0.01	0.26	0.00	0.07	0.01	0.01	0.02	0.04	0.03	0.03	0.04
Binary 3	-	0.02	0.01	-	0.43	0.44	0.28	0.01	0.10	0.44	0.44	0.44	0.06	0.08	0.04	0.36
Monk 1	-	0.00	0.07	-	0.00	0.00	0.00	0.12	0.03	0.00	0.00	0.00	0.07	0.13	0.02	0.00
Monk 2	-	0.00	0.13	-	0.05	0.05	0.05	0.03	0.03	0.04	0.04	0.04	0.06	0.01	0.05	0.06
Monk 3	-	0.64	0.02	-	0.00	0.00	0.00	0.01	0.01	0.08	0.02	0.08	0.06	0.17	0.03	0.00
Simple dependent	-	0.03	0.00	-	0.00	0.00	0.24	0.00	0.00	0.01	0.02	0.05	0.13	0.03	0.01	0.01
Partially dependent	-	0.00	0.00	-	0.01	0.01	0.00	0.00	0.01	0.61	0.46	0.46	0.11	0.24	0.37	0.01
2-way linear corr.	-	0.00	0.01	-	0.01	0.01	0.00	0.22	0.00	0.01	0.01	0.01	0.17	0.06	0.25	0.04
3-way linear corr.	-	0.01	0.01	-	0.04	0.03	0.08	0.11	0.15	0.33	0.31	0.32	0.08	0.00	0.46	0.02
3-way non-linear corr.	-	0.00	0.00	-	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.09	0.24	0.00	0.01
1 good and noise	-	0.02	0.02	-	0.01	0.01	0.01	0.15	0.05	0.01	0.02	0.01	0.12	0.13	0.10	0.00
Un-nested	-	0.01	0.01	-	0.02	0.01	0.07	0.09	0.05	0.25	0.25	0.24	0.11	0.21	0.25	0.03
Monotonic	-	0.00	0.00	-	0.01	0.01	0.03	0.03	0.03	0.00	0.01	0.01	0.05	0.07	0.45	0.02
Two U's	-	0.06	0.00	-	0.02	0.02	0.13	0.02	0.02	0.02	0.01	0.02	0.08	0.04	0.02	0.02
Multi-modal (XOR)	-	0.49	0.00	-	0.06	0.06	0.18	0.00	0.00	0.10	0.11	0.11	0.03	0.41	0.00	0.16
linear sep. Guassian	-	0.00	0.00	-	0.02	0.02	0.05	0.02	0.02	0.01	0.02	0.02	0.14	0.18	0.47	0.07

Table 4.17: Variance of the correlation of different filter measures with KNN accuracy as the parameter values are changed.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	-	0.00	0.00	-	0.17	0.16	0.33	0.18	0.07	0.34	0.34	0.34	0.11	0.00	0.33	0.29
Car Evaluation	-	0.00	0.03	-	0.00	0.00	0.00	0.00	0.01	0.05	0.03	0.03	0.10	0.09	0.01	0.00
Cardiotocography	-	0.05	0.00	-	0.03	0.03	0.03	0.03	0.01	0.33	0.33	0.33	0.03	0.00	0.41	0.08
Congressional Voting	-	0.40	0.04	-	0.00	0.00	0.00	0.01	0.02	0.36	0.33	0.36	0.07	0.00	0.34	0.00
Contraceptive Choice	-	0.34	0.04	-	0.17	0.16	0.14	0.19	0.12	0.17	0.17	0.17	0.17	0.00	0.17	0.22
Credit Approval	-	0.01	0.02	-	0.01	0.01	0.03	0.02	0.01	0.41	0.41	0.41	0.04	0.00	0.36	0.02
Dermatology	-	0.01	0.01	-	0.01	0.00	0.01	0.01	0.00	0.35	0.36	0.35	0.03	0.00	0.44	0.02
E-coli	-	0.03	0.01	-	0.01	0.01	0.02	0.02	0.02	0.38	0.35	0.35	0.05	0.00	0.41	0.02
Flag	-	0.03	0.10	-	0.02	0.02	0.05	0.02	0.00	0.23	0.25	0.23	0.02	0.00	0.27	0.03
Glass	-	0.16	0.04	-	0.06	0.05	0.14	0.08	0.06	0.32	0.31	0.32	0.05	0.00	0.31	0.10
Haberman's Survival	-	0.30	0.03	-	0.10	0.08	0.18	0.14	0.14	0.10	0.10	0.10	0.01	0.19	0.09	0.11
Ionosphere	-	0.03	0.01	-	0.06	0.06	0.12	0.09	0.05	0.12	0.12	0.10	0.00	0.00	0.25	0.03
Iris	-	0.00	0.01	-	0.02	0.02	0.11	0.03	0.17	0.02	0.02	0.01	0.04	0.06	0.02	0.01
Mushroom	-	0.00	0.01	-	0.04	0.04	0.15	0.12	0.04	0.27	0.28	0.16	0.00	0.00	0.33	0.11
Page blocks	-	0.01	0.00	-	0.06	0.08	0.15	0.08	0.07	0.27	0.27	0.26	0.09	0.00	0.23	0.13
Post-op	-	0.00	0.00	-	0.04	0.02	0.32	0.07	0.05	0.23	0.17	0.20	0.07	0.12	0.05	0.07
Segmentation	-	0.00	0.01	-	0.01	0.01	0.03	0.03	0.02	0.29	0.29	0.28	0.05	0.00	0.44	0.03
Statlog (vehicle)	-	0.45	0.01	-	0.03	0.03	0.28	0.06	0.10	0.29	0.30	0.30	0.03	0.00	0.33	0.11
WI Breast Cancer (orig.)	-	0.04	0.03	-	0.01	0.02	0.29	0.02	0.10	0.33	0.32	0.32	0.10	0.00	0.35	0.01
WI Breast Cancer (diag.)	-	0.01	0.00	-	0.01	0.01	0.42	0.06	0.01	0.25	0.25	0.24	0.01	0.00	0.37	0.07
WI Breast Cancer (prog.)	-	0.00	0.09	-	0.02	0.02	0.03	0.01	0.06	0.08	0.08	0.08	0.00	0.00	0.08	0.06
Wine	-	0.01	0.00	-	0.01	0.01	0.02	0.04	0.02	0.32	0.32	0.32	0.03	0.00	0.42	0.05
Yeast	-	0.09	0.18	-	0.14	0.13	0.13	0.13	0.09	0.37	0.34	0.35	0.27	0.00	0.40	0.17

Table 4.18: Variance of the correlation of different filter measures with KNN accuracy on real data sets as the parameter values are changed.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Binary 1	-	0.00	0.03	-	0.01	0.01	0.54	0.02	0.14	0.01	0.01	0.01	0.03	0.12	0.06	0.02
Binary 2	-	0.09	0.01	-	0.01	0.01	0.13	0.02	0.14	0.01	0.01	0.01	0.01	0.06	0.06	0.01
Monk 1	-	0.00	0.01	-	0.00	0.00	0.00	0.20	0.11	0.00	0.00	0.00	0.18	0.06	0.12	0.00
Monk 3	-	0.59	0.04	-	0.00	0.00	0.00	0.02	0.01	0.09	0.02	0.08	0.05	0.16	0.04	0.00
Simple dependent	-	0.03	0.00	-	0.00	0.00	0.24	0.00	0.00	0.01	0.02	0.04	0.13	0.03	0.01	0.01
Partially dependent	-	0.00	0.00	-	0.00	0.00	0.00	0.00	0.01	0.63	0.48	0.48	0.11	0.24	0.37	0.01
2-way linear corr.	-	0.01	0.03	-	0.03	0.03	0.00	0.30	0.00	0.05	0.06	0.11	0.13	0.14	0.52	0.10
3-way linear corr.	-	0.02	0.00	-	0.03	0.02	0.08	0.11	0.15	0.36	0.30	0.30	0.08	0.00	0.46	0.03
3-way non-linear corr.	-	0.00	0.00	-	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.11	0.23	0.00	0.00
1 good and noise	-	0.03	0.02	-	0.01	0.01	0.02	0.15	0.05	0.01	0.02	0.01	0.12	0.13	0.10	0.00
Un-nested	-	0.01	0.01	-	0.03	0.01	0.08	0.10	0.06	0.26	0.26	0.25	0.12	0.20	0.27	0.03
Monotonic	-	0.00	0.01	-	0.01	0.01	0.02	0.04	0.04	0.00	0.01	0.01	0.04	0.07	0.44	0.03
Two U's	-	0.07	0.00	-	0.01	0.01	0.19	0.01	0.03	0.01	0.00	0.02	0.05	0.02	0.04	0.03
Multi-modal (XOR)	-	0.57	0.00	-	0.13	0.13	0.12	0.00	0.00	0.08	0.11	0.12	0.04	0.41	0.00	0.25
linear sep. Guassian	-	0.01	0.00	-	0.05	0.04	0.05	0.04	0.04	0.03	0.03	0.03	0.17	0.21	0.46	0.06

Table 4.19: Variance of the correlation of different filter measures with SVM one vs. one accuracy as the parameter values are changed.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	-	0.00	0.00	-	0.14	0.13	0.22	0.18	0.14	0.21	0.21	0.21	0.10	0.00	0.20	0.25
Car Evaluation	-	0.00	0.03	-	0.00	0.00	0.00	0.00	0.02	0.07	0.04	0.05	0.18	0.11	0.02	0.00
Cardiotocography	-	0.07	0.02	-	0.06	0.06	0.08	0.07	0.03	0.37	0.37	0.37	0.03	0.00	0.40	0.13
Congressional Voting	-	0.40	0.04	-	0.00	0.00	0.01	0.01	0.02	0.38	0.35	0.37	0.07	0.00	0.37	0.00
Contraceptive Choice	-	0.47	0.08	-	0.12	0.11	0.16	0.10	0.05	0.31	0.32	0.32	0.13	0.00	0.33	0.15
Credit Approval	-	0.01	0.04	-	0.02	0.01	0.02	0.01	0.02	0.42	0.41	0.42	0.03	0.00	0.30	0.02
Dermatology	-	0.01	0.01	-	0.01	0.01	0.01	0.01	0.00	0.34	0.34	0.34	0.02	0.00	0.44	0.02
E-coli	-	0.02	0.01	-	0.01	0.01	0.03	0.02	0.03	0.39	0.35	0.35	0.04	0.00	0.43	0.03
Flag	-	0.02	0.09	-	0.02	0.02	0.02	0.03	0.00	0.29	0.30	0.30	0.04	0.00	0.32	0.01
Glass	-	0.17	0.05	-	0.07	0.04	0.14	0.06	0.03	0.36	0.36	0.36	0.04	0.00	0.32	0.10
Haberman's Survival	-	0.15	0.04	-	0.05	0.02	0.06	0.37	0.34	0.06	0.06	0.06	0.02	0.11	0.15	0.01
Ionosphere	-	0.02	0.01	-	0.04	0.02	0.23	0.08	0.02	0.32	0.32	0.25	0.00	0.00	0.29	0.07
Iris	-	0.00	0.02	-	0.02	0.02	0.11	0.03	0.18	0.01	0.02	0.01	0.04	0.06	0.03	0.01
Mushroom	-	0.00	0.00	-	0.03	0.03	0.16	0.09	0.03	0.31	0.31	0.21	0.00	0.00	0.28	0.09
Page blocks	-	0.00	0.01	-	0.07	0.10	0.10	0.08	0.07	0.29	0.29	0.29	0.08	0.00	0.27	0.17
Post-op	-	0.00	0.00	-	0.06	0.09	0.12	0.09	0.04	0.27	0.35	0.28	0.04	0.25	0.02	0.15
Segmentation	-	0.00	0.00	-	0.02	0.02	0.02	0.02	0.02	0.33	0.33	0.32	0.05	0.00	0.45	0.03
Statlog (vehicle)	-	0.57	0.02	-	0.02	0.02	0.32	0.04	0.08	0.36	0.37	0.37	0.02	0.00	0.42	0.08
WI Breast Cancer (orig.)	-	0.05	0.04	-	0.01	0.02	0.24	0.03	0.05	0.32	0.31	0.31	0.09	0.00	0.33	0.01
WI Breast Cancer (diag.)	-	0.01	0.01	-	0.02	0.01	0.40	0.07	0.03	0.23	0.23	0.23	0.01	0.00	0.36	0.07
WI Breast Cancer (prog.)	-	0.01	0.05	-	0.03	0.03	0.05	0.03	0.20	0.37	0.38	0.37	0.01	0.00	0.31	0.05
Wine	-	0.02	0.01	-	0.01	0.01	0.02	0.03	0.02	0.33	0.34	0.33	0.04	0.00	0.43	0.04
Yeast	-	0.09	0.16	-	0.16	0.15	0.15	0.15	0.11	0.33	0.30	0.31	0.31	0.00	0.37	0.20

Table 4.20: Variance of the correlation of different filter measures with SVM one vs. one accuracy on real data sets as the parameter values are changed.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Binary 1	-	0.00	0.03	-	0.00	0.00	0.51	0.02	0.17	0.01	0.00	0.01	0.03	0.12	0.06	0.02
Binary 2	-	0.03	0.01	-	0.02	0.02	0.08	0.03	0.13	0.02	0.02	0.02	0.01	0.08	0.07	0.02
Monk 1	-	0.00	0.01	-	0.00	0.00	0.00	0.20	0.11	0.00	0.00	0.00	0.18	0.06	0.12	0.00
Monk 3	-	0.56	0.04	-	0.00	0.00	0.00	0.02	0.01	0.09	0.03	0.09	0.05	0.16	0.04	0.00
Simple dependent	-	0.03	0.00	-	0.00	0.00	0.25	0.00	0.00	0.01	0.02	0.05	0.12	0.03	0.01	0.01
Partially dependent	-	0.00	0.00	-	0.00	0.00	0.00	0.00	0.01	0.63	0.48	0.48	0.11	0.24	0.37	0.01
2-way linear corr.	-	0.00	0.00	-	0.00	0.00	0.00	0.18	0.00	0.00	0.01	0.03	0.23	0.03	0.21	0.02
3-way linear corr.	-	0.02	0.00	-	0.03	0.02	0.08	0.11	0.15	0.36	0.30	0.30	0.07	0.00	0.46	0.03
3-way non-linear corr.	-	0.00	0.00	-	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.11	0.23	0.00	0.01
1 good and noise	-	0.02	0.01	-	0.01	0.01	0.02	0.14	0.04	0.01	0.02	0.01	0.12	0.13	0.10	0.00
Un-nested	-	0.00	0.01	-	0.02	0.02	0.06	0.08	0.05	0.25	0.25	0.25	0.14	0.20	0.20	0.03
Monotonic	-	0.01	0.00	-	0.00	0.00	0.02	0.02	0.08	0.01	0.01	0.01	0.06	0.04	0.42	0.01
Two U's	-	0.09	0.00	-	0.01	0.00	0.23	0.01	0.05	0.01	0.01	0.03	0.04	0.01	0.06	0.04
Multi-modal (XOR)	-	0.48	0.00	-	0.05	0.06	0.20	0.00	0.00	0.11	0.11	0.11	0.02	0.40	0.00	0.15
linear sep. Guassian	-	0.01	0.00	-	0.04	0.04	0.05	0.04	0.03	0.02	0.03	0.03	0.16	0.20	0.46	0.06

Table 4.21: Variance of the correlation of different filter measures with SVM one vs. all accuracy as the parameter values are changed.

	Fish.	RELIEF	subset RELIEF	count RELIEF	MI	SU	mRMR	CMIM	FOU	KL	JS	Bhat.	LS	MCFS	Consis-tency	CFS
Abalone	-	0.00	0.00	-	0.03	0.02	0.21	0.04	0.11	0.14	0.13	0.14	0.04	0.00	0.09	0.05
Car Evaluation	-	0.00	0.02	-	0.01	0.00	0.01	0.01	0.02	0.07	0.04	0.05	0.19	0.11	0.02	0.01
Cardiotocography	-	0.07	0.01	-	0.06	0.06	0.08	0.07	0.03	0.38	0.38	0.38	0.03	0.00	0.41	0.13
Congressional Voting	-	0.40	0.04	-	0.00	0.00	0.01	0.01	0.02	0.37	0.35	0.37	0.07	0.00	0.37	0.00
Contraceptive Choice	-	0.33	0.10	-	0.09	0.09	0.16	0.05	0.11	0.34	0.34	0.34	0.12	0.00	0.29	0.12
Credit Approval	-	0.01	0.04	-	0.02	0.01	0.01	0.01	0.02	0.42	0.41	0.41	0.03	0.00	0.30	0.02
Dermatology	-	0.01	0.01	-	0.01	0.01	0.01	0.01	0.00	0.33	0.33	0.33	0.02	0.00	0.45	0.02
E-coli	-	0.02	0.01	-	0.01	0.01	0.02	0.01	0.02	0.38	0.34	0.34	0.04	0.00	0.41	0.02
Flag	-	0.01	0.09	-	0.01	0.01	0.01	0.03	0.00	0.37	0.38	0.38	0.07	0.00	0.37	0.02
Glass	-	0.15	0.07	-	0.05	0.04	0.13	0.05	0.02	0.40	0.38	0.39	0.03	0.00	0.35	0.09
Haberman's Survival	-	0.15	0.05	-	0.11	0.04	0.06	0.50	0.43	0.09	0.09	0.09	0.01	0.14	0.16	0.04
Ionosphere	-	0.02	0.01	-	0.04	0.02	0.23	0.08	0.02	0.32	0.33	0.25	0.00	0.00	0.29	0.07
Iris	-	0.00	0.01	-	0.02	0.02	0.07	0.01	0.13	0.02	0.01	0.02	0.05	0.05	0.05	0.01
Mushroom	-	0.00	0.00	-	0.03	0.03	0.16	0.09	0.03	0.31	0.31	0.21	0.00	0.00	0.28	0.09
Page blocks	-	0.00	0.01	-	0.07	0.07	0.14	0.08	0.06	0.34	0.34	0.34	0.09	0.00	0.32	0.11
Post-op	-	0.00	0.00	-	0.07	0.11	0.17	0.09	0.05	0.36	0.49	0.33	0.07	0.32	0.04	0.21
Segmentation	-	0.00	0.00	-	0.02	0.02	0.01	0.01	0.02	0.35	0.35	0.34	0.06	0.00	0.45	0.03
Statlog (vehicle)	-	0.58	0.03	-	0.02	0.02	0.33	0.05	0.08	0.37	0.38	0.38	0.02	0.00	0.42	0.08
WI Breast Cancer (orig.)	-	0.05	0.03	-	0.01	0.02	0.24	0.03	0.05	0.32	0.31	0.31	0.09	0.00	0.33	0.01
WI Breast Cancer (diag.)	-	0.01	0.01	-	0.02	0.01	0.40	0.07	0.03	0.23	0.23	0.23	0.01	0.00	0.36	0.07
WI Breast Cancer (prog.)	-	0.01	0.05	-	0.03	0.03	0.05	0.04	0.20	0.36	0.36	0.36	0.01	0.00	0.31	0.06
Wine	-	0.02	0.00	-	0.01	0.01	0.02	0.03	0.02	0.33	0.33	0.33	0.03	0.00	0.43	0.04
Yeast	-	0.10	0.14	-	0.17	0.15	0.16	0.17	0.11	0.19	0.17	0.18	0.35	0.00	0.24	0.21

Table 4.22: Variance of the correlation of different filter measures with SVM one vs. all accuracy on real data sets as the parameter values are changed.

Probability measures

Three different probability measures are tested, and each are tested using a continuous Gaussian distribution, discrete distributions with different numbers of bins and discrete distributions where the number of bins is set to be the estimated number of clusters in the data. When using the estimated number of clusters to set the bins, the bin centers can either be evenly spaced ('e') or spaced to match the cluster centers ('u').

The correlations for the different parameter values for the artificial data sets are presented in Tables B.5,B.6, B.7, B.8, B.9and B.10 in appendix B. The correlations for the different parameter values for the real data sets are presented in Tables B.11, B.12, B.13 and B.14 in appendix B.

The quantization parameter does not appear to have a large effect on the performance of the measures for either KNN or SVM, as long as there are a sufficient number of bins to separate the relevant data. Unfortunately, neither the 'e' or 'u' bin number estimation methods achieve good results and neither one of these quantization methods reliably outperforms any of the other quantization methods, so cannot be recommended.

Mutual Information measures

For the univariate mutual information measures, the quantization value does not seem to greatly affect the correlation value for either the KNN or SVM classifier (see Table B.15, B.16, B.17, B.18 in appendix B). On the real data sets, there is a slight increase in correlation for the larger quantization values, but the increase is not large. Any quantization value that is sufficiently large to separate the clusters in the data will likely work. Using a fairly large value such as 10 appears to be a good choice for most data sets.

Neither mRMR nor FOU perform very well on the real data sets. There are no clear trends with respect to the best quantization value. The best quantization value appears to be data-set specific (see Tables B.19, B.21, B.22 and B.24 in appendix B).

CMIM, however, does perform quite well on both the artificial and real data sets. For most data sets, the difference in correlation between the different quantization values is relatively small, but increases slightly as the quantization value is increased (see Tables B.20 and B.23 in appendix B). This is not the case for every data set, but as with the univariate measures, selecting a quantization value that is relatively large should give reasonable results in most cases.

Consistency

The correlation values for the correlation filter measure using different quantizations are presented in Tables B.25 and B.26 in appendix B.

For continuous variables, the quantization of the data is important. If the data is quantized into a small number of bins, many dissimilar points get grouped together. For example, in the artificial *noise with one good feature* data set, one feature directly corresponds to the class. However, since there are four classes, if the data is quantized into fewer than four bins, the performance is severely compromised, and the correlation between the accuracy and the filter measure is quite low (0.67).

Even when the quantization is as large or larger than the number of clusters, the correlation is not perfect. In a KNN classifier, a small amount of noise normally results in a small change in accuracy. When the values are quantized, these noisy values are all placed in the same bin, and the small accuracy changes are disregarded. The filter value gets the correct general area, but not the detail. For this reason, selecting the quantization to match the number of clusters in the set also does not work well, as in most cases N_b is too low to capture detail in the features.

In general, the consistency measure appears to work best with larger N_b for the KNN classifier, but the results are much less predictable with the SVM. Overall, the measure appears to be quite sensitive to the quantization on many data sets, and there is no theoretically sound way to determine the proper value for this parameter. Therefore, if the consistency measure is used, the quantization value should be set with care, perhaps through trial and error or previous knowledge of the system.

Laplacian score

The Laplacian score measure is controlled by two separate parameters. The number of points that are included in the neighbourhood of each point is controlled by the parameter k. The parameter t controls the form of the heat kernel that is used to create the similarity matrix. The exponent of the kernel is the squared Euclidean distance divided by t. Hence, a large t value drives the exponent towards zero, and the similarity value of the neighbouring points to 1. Hence, as t becomes larger, the heat kernel approaches the 0-1 neighbourhood graph form. A low t value increases the exponent, emphasizing the distance between the neighbouring points. The actual value of t, however, appears to have little effect on the performance of the filter measure for either the SVM or KNN classifiers. The correlation values for different values of t are shown in Tables B.27 and B.30 in appendix B.

The Laplacian score correlations for different values of k are given in Tables B.28, B.29 B.31 and B.32 in appendix B. For both the KNN and SVM classifiers, the Laplacian score appears to work best when the number of neighbours (k) is relatively large. This is the case both for the artificial sets, which are very simple and do not have an underlying structure, and for the real data sets where the presence of an underlying structure is unknown.

If a data set does have an underlying structure, using a neighbourhood that is too large can cause "short-circuiting' [9]', where points that are distant on the underlying structure can be considered neighbours. However, for data sets with no underlying structure, using a larger neighbourhood basically just gives better sampling and a better representation of the entire data set. In some cases, the neighbourhood value is to set half the number of samples, which would almost certainly cause short-circuiting.

Although short-circuiting is normally considered to be a problem with neighbourhood-graph based algorithms [9], Van der Maaten [155] finds that short-circuiting can be beneficial for Isomap, which is a neighbourhood graph based feature extraction method. Short-circuiting may be beneficial for the Laplacian score filter measure for similar reasons, namely that there may not actually be an underlying manifold, or the technique used may not be able to properly extract it.

The neighbourhood size also affects the absolute value of the filter measure, which is not an issue if the neighbourhood size is fixed, but is something to be aware of when comparing between different neighbourhood sizes.

MCFS

MCFS has parameters controlling the neighbourhood size, k, the kernel type and the heat kernel parameter t. Similar to the Laplacian score, the best t value is data set dependent, and no single value of t outperforms the others for either the KNN or SVM classifier (see Tables B.33 and B.37). Three different kernels are also tested, as shown in Tables B.34, B.35, B.36, B.38, B.39 and B.40. The performance of the three kernels is fairly comparable for both classifiers. If the MCFS filter measure is being used, the zero-one kernel is likely a good choice for the kernel as it is the simplest to calculate and also appears to create eigenvalue problems that are more easily solved.

Because the Laplacian score performed much better with larger k values, the MCFS filter is only tested with k values of 25 and higher. The best k value is data set dependent, with no single value of k outperforming the others for any data set or kernel on either of the classifiers.

CFS

The correlation of CFS and classifier accuracy for different values of N_b are presented in Tables B.41 and B.42. A relatively large N_b works well for all the classifiers on both the artificial and real data sets. Overall, however, the different quantizations do not have a large effect on the performance. As with the mutual information based measures, any quantization that is sufficiently large to capture the behaviour of the data will likely work well.

4.3.5 Summary of filter measure evaluation

While univariate measures are faster to use, they are unable to detect dependencies between features, and are also unable to identify non-monotonic sets. This is a particular problem for the real data sets as all the tested data sets are, in fact, non-monotonic to varying degrees.

For KNN classifiers, the best filter measures appear to be the consistency measure, CFS, CMIM and subset RELIEF. However, the consistency measure is unable to identify non-monotonic sets, and is highly sensitive to the quantization parameter. CFS, CMIM or subset-RELIEF would be a more practical choice.

For SVM classifiers, the best filter measure appears to be Fisher's interclass separability criterion, or CMIM if an additive multivariate measure is desirable.

The results for each of these classifiers is summarized in Figures 4.3, 4.4 and 4.5

Table 4.23 summarizes the ability of each filter measure to correctly handle each data set challenge outlined in Section 4.2.1

Non-monotonic sets

Univariate measures do not work well for non-monotonic sets since the summed values increase as more features are added. The only exception is RELIEF-F, which can take negative values.

Additive multivariate measures also have difficulty with non-monotonic sets if the measures can only take positive values. Because mRMR and FOU can take negative values, they can theoretically identify non-monotonic sets whereas CMIM, Laplacian score and MCFS cannot.

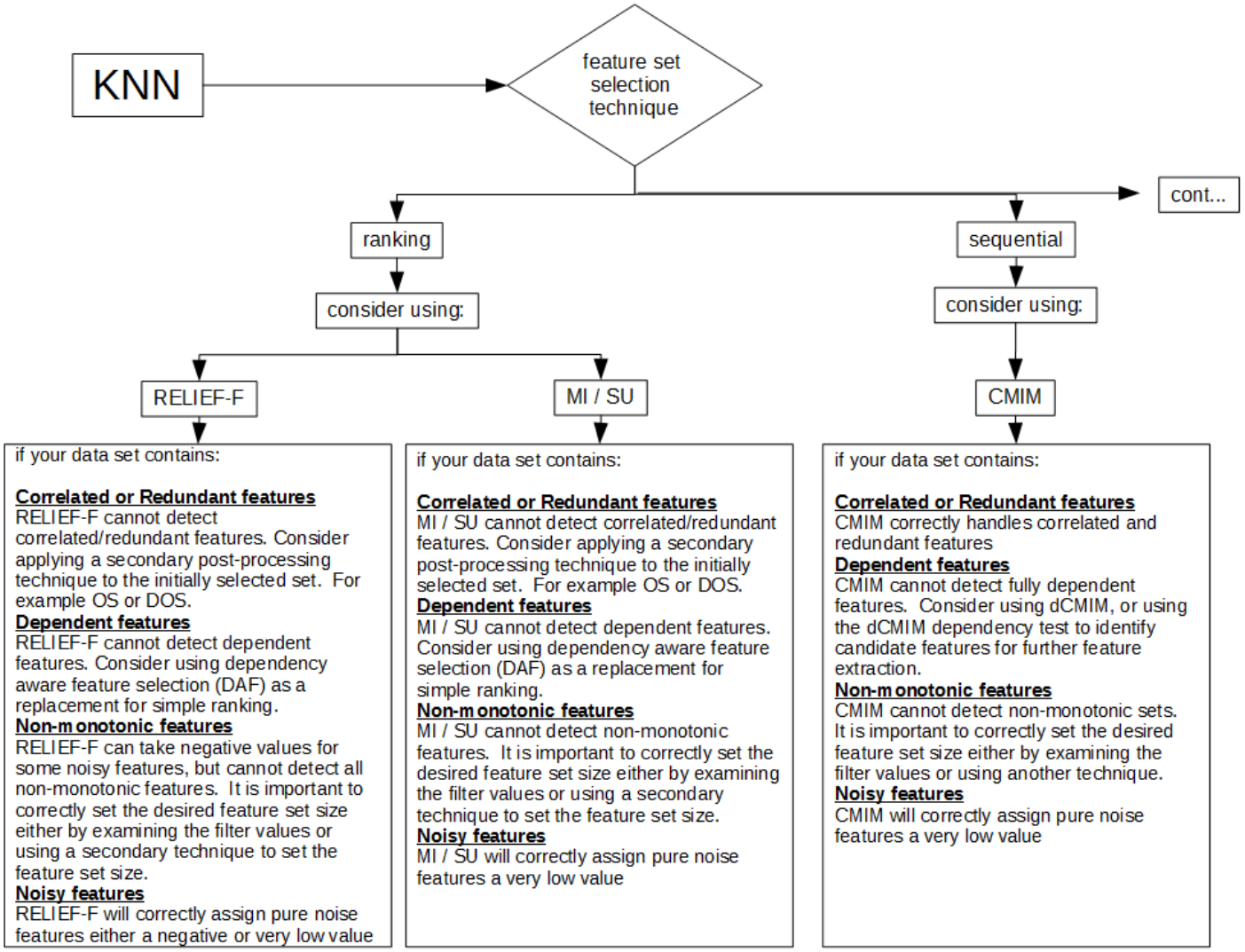


Figure 4.3: Selecting filter measures for use with KNN. Continued in Figure 4.4

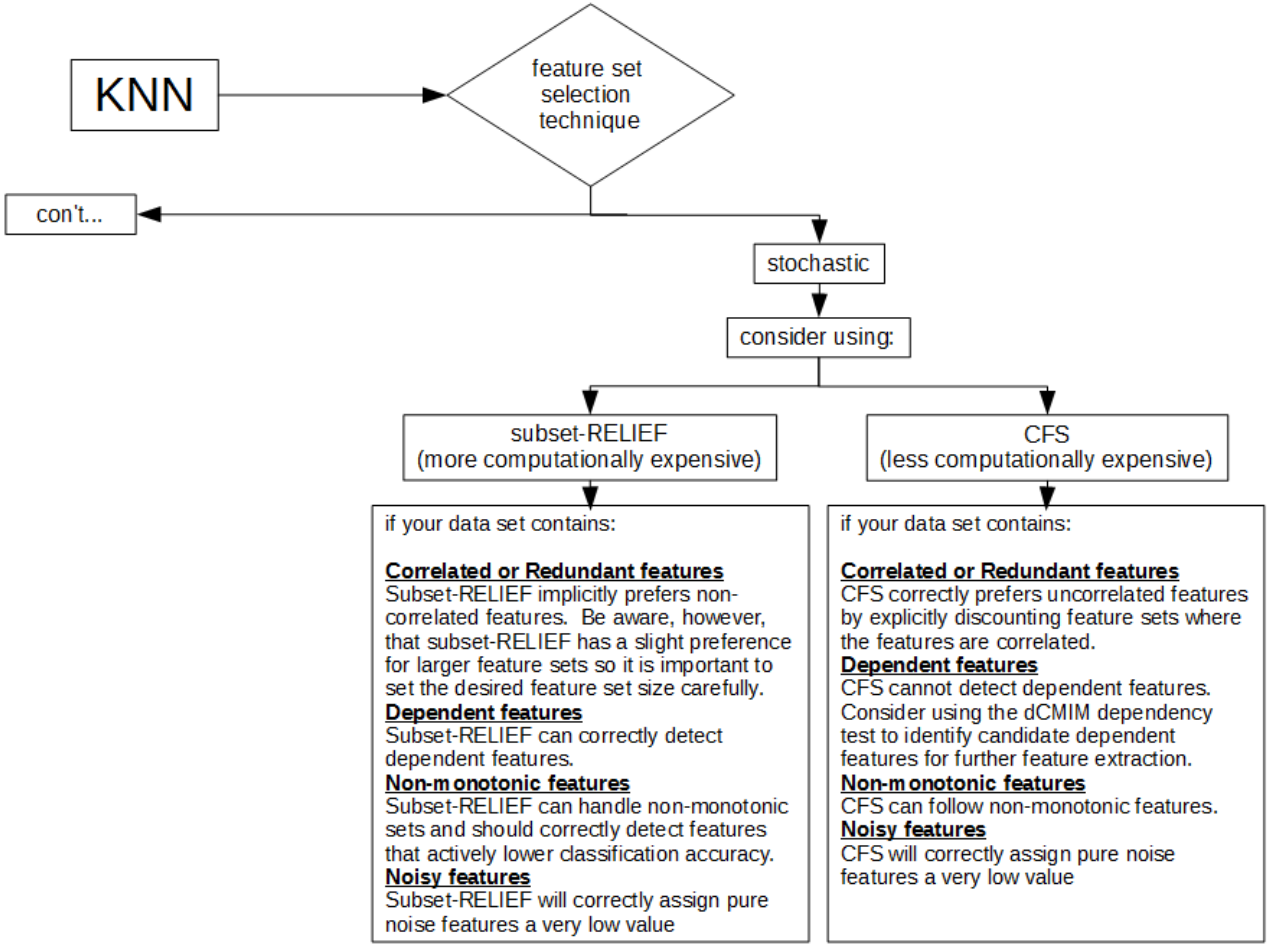


Figure 4.4: Selecting filter measures for use with KNN (continued). Continued from Figure 4.3.

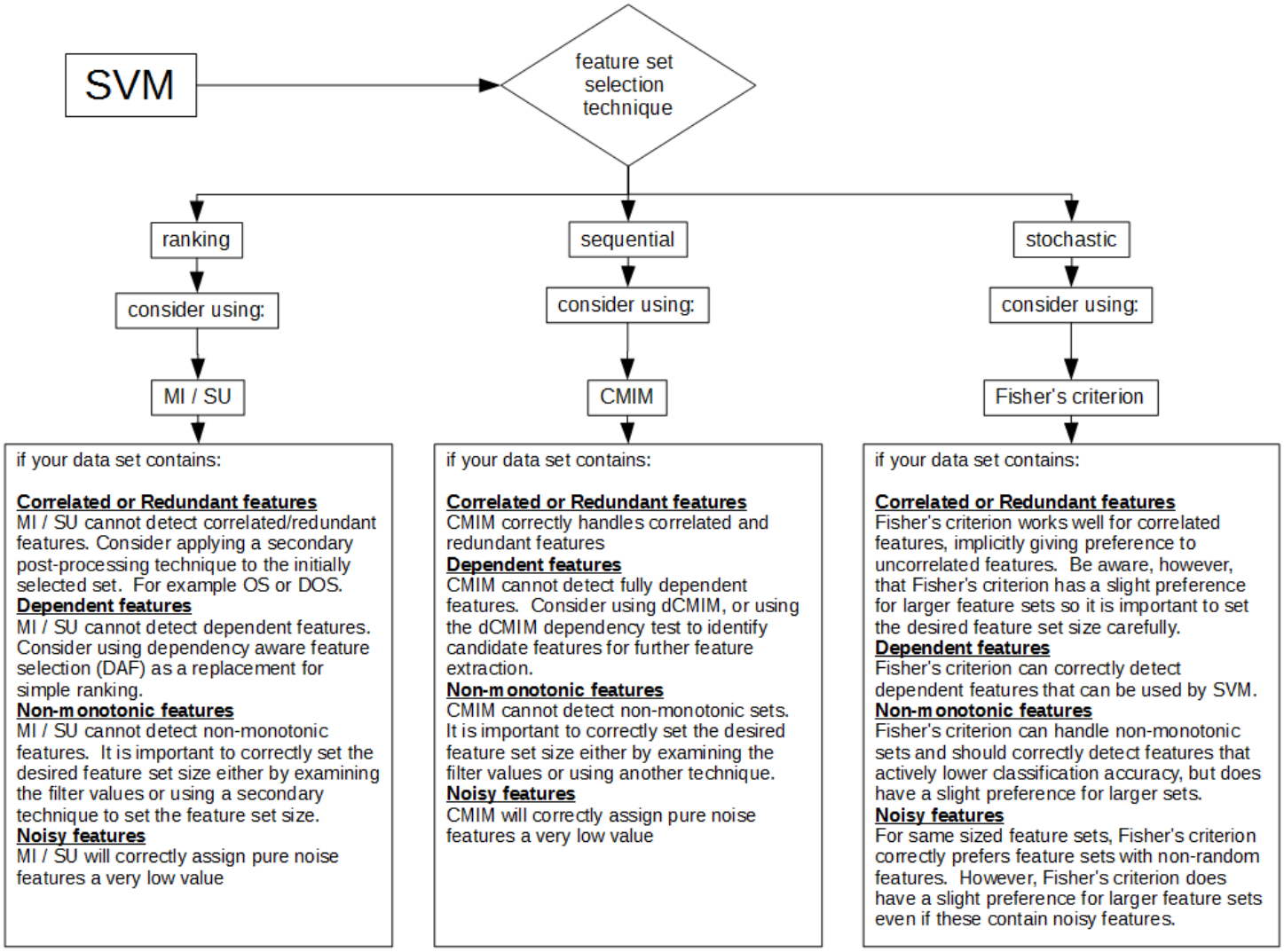


Figure 4.5: Selecting filter measures for use with SVM

measure	non-monotonic	dependence	correlation	redundance	noise
Fisher	yes	yes	yes	no	yes
RELIEF-F	yes	no	no	no	yes
subset RELIEF	yes	yes	yes	no	yes
count-RELIEF	yes	yes	yes	yes	yes
KL divergence	no	no	no	no	yes
JS divergence	no	no	no	no	yes
Bhattacharyya	no	no	no	no	yes
mutual information	no	no	no	no	yes
symmetric uncertainty	no	no	no	no	yes
CMIM	no	yes*	yes	no	yes
mRMR	yes	no	no	no	yes
FOU	yes	yes*	no	no	yes
LS	no	no	yes	no	no
MCFS	no	no	yes	no	no
consistency	no	yes	yes	yes	no
CFS	yes	no	yes	no	yes

Table 4.23: Common data set challenges for feature selection

The consistency filter also fails for non-monotonic sets, since adding features can only separate previously inconsistent points.

Dependency

The univariate measures cannot detect dependency because each feature is considered independently of the other features. The unsupervised filter measures also cannot detect dependent features because they do not consider the class at all when selecting the features.

CMIM and FOU both include a conditional mutual information (CMI) term that appears to be able to select dependent features. However, they are designed to exclude correlated features rather than to select dependent features. The CMI term is high for dependent features. However, because both filter measures measure the CMI with respect to the selected feature set, these filter measures can only select features that have a high conditional mutual information with a feature that is already selected. Neither, therefore, can select features that have a low individual mutual information but a high conditional mutual information (fully dependent). They can, however, select partially dependent features if one of the features is selected on its own merits.

Subset RELIEF, count-RELIEF and Fisher's are all able to select dependent features.

Correlation

Univariate measures are unable to detect correlation between features because each feature is considered independently.

The supervised multivariate filter measures all consider correlated features either explicitly or implicitly. Fisher's interclass separability criterion, for example, implicitly discounts correlated features because adding a linearly correlated feature causes a smaller distance change. Conversely, CFS, FOU and mRMR explicitly discount correlated features by including feature-feature correlation as a term in the measure.

FOU and mRMR can actually over-correct if the correlation between the features is higher than the correlation between the feature and the class. These features have negative filter measure values, indicating that a correlated feature is highly undesirable even if the correlated feature can improve classification accuracy. In these cases, the filter measures will actually prefer a feature that is completely random and has zero mutual information with the class and with the other feature.

Redundancy

Univariate filter measures always indicate that adding more features will be helpful, which is incorrect for redundant features. This can cause helpful features to be excluded in favour of redundant features that contain no additional information.

The multivariate mutual information-based filter measures work well for the redundant data sets in these tests. The conditional mutual information for a redundant feature is zero, so CMIM will work well for any truly redundant features. As with the correlated features, FOU and mRMR can over-correct. Redundant features have a mutual information value of 1. FOU and mRMR will therefore have a negative value if the correlation between the feature and the class is not also 1. In these tests, the features in the redundant set perfectly predict the class and therefore have a mutual information of 1 and FOU and mRMR mappear to work well. However, as with the correlated features, FOU and mRMR do not work in the general case.

For the distance-based measures, adding redundant features affects the distances between classes. The subset RELIEF measure prefers sets that include the redundant features, but count-based RELIEF does not.

The consistency measure also performs correctly for redundant features, since adding a fully redundant feature will never separate any new samples.

Noise

The only filter measures that do not properly identify noisy features are the two unsupervised filter measures (LS and MCFS) and consistency. The unsupervised measures have

no way to determine if features are correlated with the class, and therefore have difficulty removing them. The consistency measure also has difficulty with noisy features. Because the measure only considers separation of pairwise samples and not an overall pattern in the features, noisy features appear to be good because they separate samples in different classes, even if the separation is arbitrary.

FOU and mRMR can select noisy features if there are correlated features that are have a lower mutual information with the class than with the other features in the set.

4.4 Proposal for a new filter measure

It is clear from the results presented in Section 4.3 that some of the tested measures have difficulty with specific data set challenges. There is no single filter measure that performs well for every data set. However, there are several candidate filter measures that work well in many cases, and could be adapted to better handle specific challenges.

One promising candidate is the CMIM measure. In most cases, this measure can properly discount pairwise correlated features, can identify noisy features and has a relatively low increase for redundant features. However, it cannot detect all dependent features and assumes monotonic sets.

As discussed in Section 4.3.5 and 4.3.2, the conditional mutual information can detect dependent features, but CMIM will not add dependent features unless one of the dependent features is not already included in the set. This stems from the fact that the original goal of CMIM was to exclude correlated features rather than include dependent ones.

Consider, for example, the *Monk 1* data set. In this data set, the class is 1 when $(x_1 == x_2)|(x_5 == 1)$. The feature-class mutual information and conditional mutual information values for the features in this data set are given in Table 4.24. In a sequential forward search, feature 5 would be selected first. However, $I(x_1; Y|x_5)$ and $I(x_2; Y|x_5)$ are both low, despite the fact that $I(x_1; Y|x_2)$ and $I(x_2; Y|x_1)$ are both high. Hence, features 1 and 2 are not the next selected features despite the fact that they collectively bring a large amount of information.

One possible way to prevent this problem is to first examine the table of CMI values and find pairs of features that have high CMI. If the CMI of two features is more than their summed mutual information then they bring more as a pair than they would as two individuals and should therefore be added as a pair. The new proposed measure uses this criterion to identify dependent pairs of features, which are then used to create a new

Monk 1 data set					
mutual information: $I(f_i;C)$					
0.00	0.00	0.00	0.00	0.31	0.00
conditional mutual information: $I(f_i;C\|f_j)$					
-	0.46	0.00	0.00	0.00	0.00
0.46	-	0.00	0.00	0.00	0.00
0.00	0.00	-	0.00	0.00	0.00
0.00	0.00	0.00	-	0.00	0.00
0.31	0.31	0.31	0.31	-	0.31
0.00	0.00	0.00	0.00	0.00	-

Table 4.24: Mutual information and conditional mutual information for *Monk 1* data set

combined feature. CMI is then performed normally on the new feature set, adding both dependent features when the combined feature is selected.

The other difficulty with CMIM is that it assumes monotonic sets. First order utility [20] can behave non-monotonically because it includes a negative term for correlated features. However, it often overestimates the detrimental effect of correlated features, and although the terms of the filter measure can be parameterized, the correct parameterization is data set dependent [20].

Although it is called Correlation Feature Selection [58], CFS is actually based on the symmetric uncertainty, which is an information-theoretic measure. Like mRMR and FOU, CFS includes a term that explicitly discounts features that are pairwise correlated with other selected features (as measured using their symmetric uncertainty). However, rather than subtracting the pairwise correlation term, CFS divides the terms. Because CFS is applied group-wise, it can be monotonic, but using this division term on an additive multivariate measure would create a positive monotonic measure.

This section proposes two new filter measures, dCMIM and dFOU, based on CMIM and CFS/FOU. Both begin by using the matrix of feature-feature CMI values to identify dependent features, then apply the measures on a set of transformed features. A flow chart describing this procedure is given in Figure 4.6.

A dependent pair of features is defined as a pair of features where the conditional mutual information is higher than the sum of the mutual information values for the independent features. These features bring more information as a pair than as a set of individual features.

$$DEP_{i,j} = (I(x_i;Y|x_j) > I(x_i;Y)+I(x_j;Y)+t)|(I(x_j;Y|x_i) > I(x_i;Y)+I(x_j;Y)+t) \quad (4.2)$$

where $DEP_{i,j}$ is a binary value that determines if a pair of features is dependent, $I(x_i;Y|x_j)$ is the conditional mutual information of feature x_i and the class labels Y given x_j is already

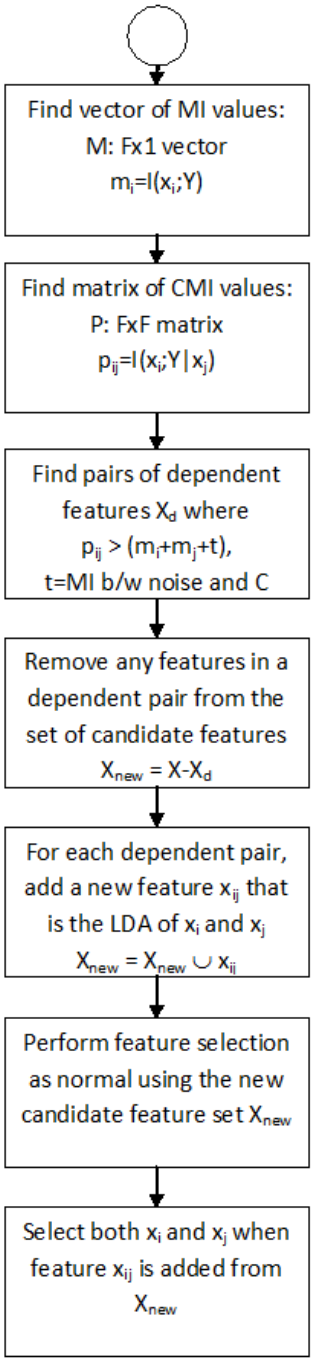


Figure 4.6: Flow chart for using dCMIM or dFOU filter measure

included in the set of selected features, $I(x_i;Y)$ is the mutual information between feature x_i and the class labels and t is a tolerance factor that is found using the mean mutual information of N samples of uniform random noise and the class labels.

The set of dependent features X_d consists of all the features that are included in a dependent pair ($DEP_{i,j} == TRUE$). The set of dependent features is removed from the new set of input features $X_{new} = X - X_d$. For each pair of dependent features, a new feature (x_{ij}) is added to the new feature set. The new feature is the first dimension of the FDA transform of the two input features.

$$x_{ij} = FDA_1(x_i, x_j, Y) \tag{4.3}$$

After creating the new set of features X_{new}, feature selection is performed normally, selecting features one at a time from X_{new} using SFS and the standard CMIM measures as described in 2.4.2. When a dependent pair feature x_{ij} is selected, both x_i and x_j are added to the selected feature set, but the remaining CMI values are still calculated with respect to x_{ij}.

The feature transform is used in place of the pair of features to avoid sampling issues. The CMI for a single feature with respect to each feature in the selected set $I(X_i;Y|X_f)$ requires calculating a three-way joint probability between the classes and the two features. The CMI for a pair of dependent features $I(X_p;Y|X_f)$ would require calculating a four-way joint probability. This problem is exacerbated if pairs of dependent features are already present in the selected set. FDA is used to create a new feature because it is a relatively simple supervised feature extraction technique. More complex feature extraction methods may also yield good results.

A second measure is also proposed, which uses SFS and a new measure that is similar to FOU or CFS. As with the first measure, a new set of input features, X_{new} is constructed by removing the set of dependent features X_d and replacing them with the set of FDA features constructed from each dependent pair. The set of features is selected using SFS and a measure based on CMIM and FOU/CFS where the value of adding each new feature x_m to the set of $M-1$ previously selected features S_{m-1} is given as:

$$J_m = \frac{\min_{x_i \in S_{m-1}} I(x_m;Y|x_i)}{1 + \frac{1}{M-1}\sum_{i=1}^{M-1} I(x_m;x_i)} \tag{4.4}$$

The numerator of this measure is the CMIM value for feature m. The denominator includes a term to account for the average feature-feature mutual information between x_m and the previously selected features.

4.4.1 Experiments

Tests were run using the two new proposed measures: dCMIM, which uses SFS and CMIM on the transformed input set X_{new} and dFOU, which uses SFS and the measures described in 4.4 on the transfomed input set X_{new}. The test environment and data sets are as described in Section 4.2.

4.4.2 Results and discussion

The results from tests using the two new proposed measures are presented Tables 4.25, 4.26 and 4.27.

On the real data sets, the performance of the measures is fairly comparable to CMIM. This may be an indication that the real data sets do not contain a large number of dependent features, or that the tested classifiers do not work better with dependent features.

However, the two measures do perform particularly well on the artificial data sets, which are known to contain dependent features. Table 4.25 shows the ability of dCMIM and dFOU to find dependent features in a set and to select the known informative features from the artificial data sets. It is clear from this table that the new measures are able to find dependent features where the other tested measures fail. In particular, the new measures are able to correctly identify dependent features in all the *Monk* data sets, *Binary 3*, *multi-modal* and *simple dependent*.

The new measures do not, however, correctly identify the features in the *partially dependent* set. This is because one of the features is actually relatively powerful, and the second feature only brings a small amount of additional information. Because the mutual information for one feature is so large, the features are not flagged as dependent even though they work better as a pair. Similarly, the two dependent features in the nested data set are not flagged because the information from the individual features is too high.

These data sets illustrate the problem of defining dependence in features. The newly proposed measures use a binary condition for dependence. This is necessary to generate the new feature values. However, feature dependence is likely a more fuzzy concept, with some features being fully dependent, and others that bring a large amount of information themselves, but still work somewhat better as a pair. It may be worthwhile to explore this idea as a future work, using a fuzzy measure of dependence based on the ratio of the CMI to the mutual information values. This could be particularly beneficial if these measures were being used as a way of exploring the data set rather than strictly for feature selection.

The noise features suffer from the opposite problem. Because the mutual information values for the individual noise features are so low, they are mistakenly flagged as being dependent when in fact they are just noise. This does not affect the final outcome as the mutual information values for the newly created features are also quite low, but it does make the procedure more computationally expensive.

In fact, the computational expense is the largest single drawback to these new measures. The standard SFS/CMIM procedure requires the calculation of $F \times (Q-1)$ CMI values and F mutual information values, where Q is the desired number of features. The new measures require the calculation of $F \times F$ CMI values and F mutual information values, and then an additional calculation of $D \times (Q-1)$ CMI values and D mutual information values, where D is the number of dependent features. Not only is the calculation of the initial CMI matrix quite expensive, there is an additional expense for re-calculating the CMI for the dependent features. In some cases, D can be larger than the initial number of features if there are large number of dependent pairs.

There several potential ways to reduce this computational cost. Firstly, the CMI calculations are completely independent and can easily be parallelized with very little additional programming effort. The mutual information calculations can be similarly parallelized.

A second option is to examine the dependent feature set after each new feature is added. Any feature x_{ij} can be removed if features i and j are already included in the set. This effectively reduces the size of D as more features are added to the set.

A final option is to reduce the size of D preemptively by including only the best performing dependent pairs. The feature pair with the highest CMI value is added to the set as feature x_{ij}. Any dependent pairs containing either feature i or feature j are then eliminated before their dependent feature is calculated. This would greatly reduce the number of dependent features, but may also slightly reduce the effectiveness of the measure by considering only the best initial dependent feature pairs.

4.5 Summary

This chapter presented an extensive empirical evaluation of common filter measures. The measures are evaluated with respect to their ability to select known informative features, their correlation with classifier accuracy and their ability to deal with specific feature selection challenges, including non-monotonic data sets, correlated features, redundant features, dependent features and noisy features. Based on these tests, several filter measures are

	# dependent features		Informative features identified	
	data set	dCMIM / dFOU	dCMIM	dFOU
Binary 1	3	0	✓	✓
Binary 2	3	0	✓	✓
Binary 3	3	3	✓	✓
Monk 1	2	2	*	*
Monk 2	6	6	✓	✓
Monk 3	3	3	*	*
Simple dependent	2	2	✓	✓
Partially dependent	2	0	✓	✓
2-way linear corr.	2	0	✓	✓
3-way linear corr.	2	0	*	*
3-way non-linear corr.	3	0	✓	✓
1 good and noise	0	3 (noise)	*	*
Un-nested	2	0	–	–
Monotonic	0	0	✓	✓
Two U's	2	0	✓	✓
Multi-modal (XOR)	2	2	✓	✓
linear sep. Gaussian	2	0	✓	✓

Table 4.25: Ability of dCMIM and dFOU to find dependent features and to select informative features from the set. A ✓ indicates that the measure was highest when the best performing set was selected. A * indicates that the measures was highest for the set that contained all the features, but that the informative features were selected first. A - indicates that the measure was unable to recover the best feature set.

	KNN			SVM (OVO)			SVM (OVA)		
	dCMIM	dFOU	CMIM	dCMIM	dFOU	CMIM	dCMIM	dFOU	CMIM
Binary 1	0.94	0.94	**0.97**	**0.59**	**0.59**	**0.59**	**0.55**	**0.55**	**0.55**
Binary 2	0.97	0.97	**1.00**	**0.70**	**0.70**	**0.70**	**0.56**	**0.56**	**0.56**
Binary 3	0.95	**0.96**	0.94	4.63	3.88	6.52	4.63	3.88	6.52
Monk 1	**0.87**	**0.87**	0.82	**0.98**	**0.98**	0.60	**0.98**	**0.98**	0.60
Monk 2	**-0.43**	**-0.43**	-0.50	2.83	2.64	4.60	2.83	2.64	4.60
Monk 3	**0.95**	**0.95**	0.86	0.89	**0.90**	0.89	0.87	**0.89**	0.88
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Partially dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
2-way linear corr.	0.88	0.73	**0.99**	**0.43**	0.18	**0.43**	**0.95**	0.83	**0.95**
3-way linear corr.	**0.98**	0.97	0.88	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**
3-way non-linear corr.	0.89	0.89	**0.95**	**0.89**	**0.89**	**0.89**	**0.89**	**0.89**	**0.89**
1 good and noise	0.98	**0.99**	0.93	0.99	**1.00**	0.99	**0.97**	**0.97**	**0.97**
Pure noise	**0.83**	**0.83**	0.68	0.94	**0.95**	0.58	-0.20	-0.23	**-0.14**
Un-nested	**0.99**	0.98	0.88	**0.99**	0.98	**0.99**	**0.96**	0.95	**0.96**
Monotonic	0.88	0.88	**0.92**	0.87	**0.88**	0.87	**0.83**	0.82	**0.83**
Two U's	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Multimodal (XOR)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
linear sep. Guassian	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Redundant	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2 clusters	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
best	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
worst	0.87	0.73	0.82	0.43	0.18	0.43	0.55	0.55	0.55

Table 4.26: Correlation of dCMIM and dFOU with classifier accuracy on artificial data sets, using best parameter set

	KNN			SVM (OVO)			SVM (OVA)		
	dCMIM	dFOU	CMIM	dCMIM	dFOU	CMIM	dCMIM	dFOU	CMIM
Car Evaluation	**0.53**	**0.53**	**0.53**	**0.81**	**0.81**	**0.81**	**0.77**	**0.77**	**0.77**
Congressional Voting	0.87	**0.90**	0.87	0.91	**0.93**	0.91	0.91	**0.93**	0.91
Contraceptive Choice	**0.60**	0.56	**0.60**	**0.87**	0.85	**0.87**	**0.78**	**0.78**	**0.78**
Credit Approval	0.90	**0.91**	**0.91**	0.94	**0.95**	0.94	**0.94**	**0.94**	**0.94**
Dermatology	**0.87**	**0.87**	**0.87**	**0.83**	**0.83**	**0.83**	**0.82**	**0.82**	**0.82**
E-coli	**0.95**	**0.95**	**0.95**	**0.97**	**0.97**	**0.97**	0.94	**0.95**	0.94
Flag	**0.58**	**0.58**	0.57	**0.75**	**0.75**	0.73	0.87	0.87	**0.88**
Glass	**0.76**	0.74	**0.76**	**0.76**	**0.76**	**0.76**	**0.82**	**0.82**	**0.82**
Haberman's Survival	0.44	**0.47**	0.41	-0.50	-0.50	**0.19**	-0.40	-0.40	**0.22**
Ionosphere	0.14	0.15	**0.35**	0.62	0.62	**0.77**	0.62	0.63	**0.78**
Iris	**0.99**	**0.99**	**0.99**	0.98	**0.99**	0.98	**0.98**	**0.98**	**0.98**
Page blocks	**0.64**	**0.64**	**0.64**	**0.68**	**0.68**	**0.68**	**0.82**	**0.82**	**0.82**
Post-op	0.43	0.42	**0.57**	0.24	**0.26**	-0.10	**0.42**	**0.42**	0.11
Segmentation	0.82	**0.83**	0.82	0.87	**0.88**	0.87	**0.90**	**0.90**	**0.90**
Statlog (vehicle)	0.76	**0.77**	0.76	0.87	0.87	**0.89**	0.87	0.87	**0.90**
WI Breast Cancer (orig.)	**0.79**	**0.79**	**0.79**	0.81	**0.84**	0.81	0.81	**0.84**	0.81
WI Breast Cancer (diag.)	0.66	0.70	**0.76**	0.63	0.68	**0.72**	0.64	0.69	**0.72**
WI Breast Cancer (prog.)	0.23	**0.24**	0.23	**0.00**	**0.00**	-0.75	**0.00**	**0.00**	-0.73
Wine	**0.76**	**0.76**	0.75	0.83	**0.84**	0.83	0.84	**0.85**	0.84
Yeast	**0.70**	**0.70**	**0.70**	**0.95**	**0.95**	**0.95**	**0.68**	**0.68**	**0.68**
best	0.99	0.99	0.99	0.98	0.99	0.98	0.98	0.98	0.98
worst	0.14	0.15	0.23	-0.50	-0.50	-0.75	-0.40	-0.40	-0.73

Table 4.27: Correlation of dCMIM and dFOU with classifier accuracy on real data sets, using best parameter set

recommended for the different classifiers. For KNN, subset-RELIEF, CFS or CMIM are recommended. For SVM, Fisher's interclass separability criterion or CMIM is recommended. CMIM was found to be a good general feature selection measure.

Although the performance of CMIM is generally quite good, it is unable to select fully dependent features. Two new filter measures are proposed to overcome this problem. The dCMIM filter measure is based on the CMIM measure, but uses an additional preprocessing step to explicitly look for dependent features. The dFOU measure also adds a term to account for feature correlation.

The dependency test is based on a comparison of the pairwise conditional mutual information terms of the two features to their summed mutual information values. Results indicate that this test is able to identify a number of strongly dependent features and the dCMIM and dFOU filter measures are capable of identifying the known informative features in 19 of the 20 tested artificial data sets.

Chapter 5

Feature Selection for time-series human-motion data

The majority of the commonly used classification algorithms do not explicitly deal with time-series data. Most classifiers use a single set of input variables, corresponding either to a single time step, or a set of features computed over a-priori defined windows.

A non-temporal classifier can be extended to work with time-series data by adding the various time-delay components as additional inputs to the system. This is illustrated in Fig. 5.1b. This type of time-delay system has been used in many applications, including human motion recognition [138].

The inclusion of time-delay components adds additional parameters that need to be considered. It is not clear that simply adding a set number of time-delayed inputs of the selected features will give good results. Adding time-delay components also adds more features, which can make the classifier slower during both training and testing and also increase the amount of memory required to store the classifier. When considering time-delay inputs it is important to determine systematically which features should be included. In many cases, the entire time-series is considered as a single feature and is included or excluded accordingly [25]. However, because these time-delay components enter the classifier as separate inputs, it should be possible to evaluate each time-delay as a separate feature (see Figure 5.1c). Time delay selection has also been used in prediction problems [67], which use a single feature as an input.

When classifying motions, it may be possible to differentiate between motions using only a portion of the available time-series. For example, when classifying a punching motion using position or joint data, the classifier may not require the entire arm trajectory. It

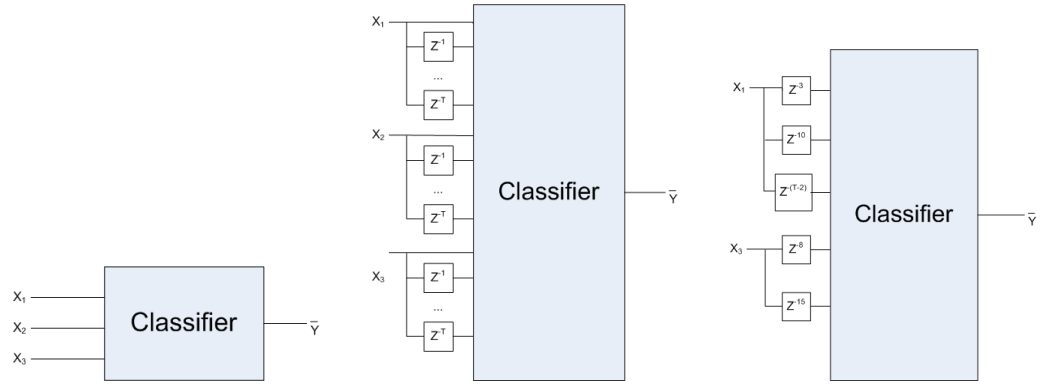
may be sufficient to know that the arm starts near the body, and becomes fully extended approximately halfway through the motion. A kick motion should be differentiable from a punch motion by comparing the arm and foot positions midway through the motion, without knowing the full motion trajectory.

Selection of time delay components is challenging, however. Adding delayed versions of each feature increases the search space greatly. The time-delayed versions of the features are also likely to be correlated. In human motion data, for example, the position of any body part at time t will be similar to the position at time $t + 1$ because people are only capable of moving with a finite speed. The features themselves may also be correlated because of the kinematic and dynamic constraints of the body.

This chapter examines the selection of specific time-delay values for human motion data. The aim of this chapter is to determine whether time-delay feature selection can be used to generate classifiers that have a smaller memory footprint and shorter test time, but maintain a comparable accuracy to classifiers that use the entire time-series of a feature.

Time-delay feature selection is then used with the multi-modal binary tree classifier outlined in Chapter 3 Section 3.2 to generate a tree-based classifier of human motions. These techniques are tested on two human motion data sets. The first data set classifies full-body human motions from joint angle data. The second data set examines hand gesture recognition from EMG data.

MBT is an intuitively good choice for human motion recognition, as human motions fall in a naturally hierarchical structure, as shown in [86]. Additionally, humans may use different movement strategies to perform the same motion. These different strategies can create multiple clusters of movement types within the same motion class. As demonstrated in Chapter 3, MBT performs well on multi-modal data sets. Lastly, the MBT requires fewer multiplications to classify incoming points, resulting in a fast test time. Because the eventual intention is to create a classifier that is trained offline, but is used to classify incoming samples in an online manner, test time is an important consideration. The test time directly affects the time between the receipt of a sample and the output of a classification result. If the result of the classification is used to trigger another event, as in gesture control systems, the classification test time directly affects the system lag and the perceived performance. A computationally efficient classifier is also important for systems that have little processing power, such as body-worn devices, in order to complete the classification in a reasonable time and to allow other processes to share the available resources.



(a) classifier with three non-time-series inputs

(b) classifier with time-delay inputs

(c) classifier with time-delay inputs, using feature selection

Figure 5.1: Classifier extensions for time-series data

5.1 Full body motion recognition from joint angles

Tests with the full body human motion recognition data set are designed to explore the effectiveness of time-series window selection and the performance of the MBT algorithm on human motion data.

First, the effectiveness of time-window selection is evaluated. The purpose of these tests is to determine if time-delay feature selection can be used to reduce classifier memory footprint and test time, while maintaining classifier accuracy.

In many cases, human motion recognition is performed using time-series modeling that uses the entire time sequence of each feature. For example, [87] models human motions using hidden Markov models. However, for motion recognition, using the entire time sequence may be unnecessary. In many cases, for human motion data there is a large amount of correlation both between different features and between the time windows of the same feature due to the kinematic constraints of the human body. Additionally, there may be uninformative features since not all body parts are involved in every motion. For example, the position of the arms during a kicking motion may not be informative. Using only a subset of the windows in a feature can reduce classifier memory requirement and classification time, and may also improve accuracy. Feature selection is a good way to select among these windows and features because many feature selection methods are designed specifically to eliminate noisy, uninformative and correlated features.

In order to separate the effect of the time-window feature selection from the effect of the MBT classifier design, the time-window feature selection is first evaluated on a two-class problem using a single SVM classifier. Selection of an entire time-series feature is compared to the selection of individual time windows.

Based on the initial tests using feature selection to select individual time windows, a two-stage selection process is proposed. A two-stage feature selection method is used to reduce the time required for feature selection. Because each of the time windows of each feature is included as a separate input to the feature selection process, using feature selection methods for time-series data can be quite time consuming.

Most of the selected windows occur at a local minimum or maximum of either the average or variance/difference. This suggests a way to reduce the feature selection time by passing only the local extrema points to the feature selection algorithm. The first stage of the two-stage selection finds local minimums and maximums in the time series. These windows are passed to the feature selection algorithm and the remaining windows are eliminated. This two-stage selection process is investigated using two-class human motion classification problem as well as several artificial data sets.

Lastly, the MBT algorithm developed in Chapter 3 Section 3.2 is applied to this human motion data set. Using a tree-based classifier reduces classification time, as discussed in Section 3.2.2. Additionally, the structure of the tree-based classifier may also be informative about the motions themselves, indicating which motions are more similar.

5.1.1 Experiments

The first experiments explore whether selecting individual time windows can be as effective as using the entire time series for human motion recognition. Selection of individual time windows is compared to selection of the entire feature, where all the time windows are included as inputs.

Two-class data sets and a single SVM classifier are used for these experiments so the effect of the time-series selection can be evaluated separately from the effect of the MBT algorithm. Two two-class human motion recognitions problems are used: classifying kick vs. punch and throw vs. punch motions from joint angle data. Kick vs. punch is selected because these motions are intuitively quite different and involve different major limbs. The features should be fairly distinctive because the problem is relatively simple. Throw vs. punch is used because these motions use the same limb and are quite similar. This problem is harder than the kick vs. punch problem.

The motions are extracted from the human motion recognition data set, which is a 21 feature data set of joint angles. The time-series data from each joint is divided into 20 windows and the average and variance or difference of each window is included as an input feature for the feature selection process. Tests are run using joint angle features from the whole body or from the upper body only. The data set is discussed in detail in Section 5.1.1.

Feature selection is performed using sequential forward search[12] with either CMIM [48] or FOU [20] as the feature set evaluation measure. Ties are broken using the highest individual mutual information then by Fisher's interclass separability on each individual feature [43].

CMIM is selected as the feature set measure because CMIM with SFS is a good predictor of linear SVM accuracy, as found in Chapter 4. FOU is used because it explicitly accounts for feature-feature correlation, which is likely to be high in this data set. Fisher's interclass separability criterion is also a good predictor of linear SVM accuracy and hence is used to break ties. While it is possible for CMIM and MI to generate a number of ties, particularly if there are multiple highly predictive features, Fisher's is distance based and hence more easily ranks the features.

Based on the initial tests, a two-stage feature selection process is proposed where only the local extrema of the time series are passed to the feature selection method. The two-stage feature selection is tested on the two-class kick vs. punch and throw vs. punch motion recognition problems used in the first set of experiments. It is also tested on artificial data sets of Gaussian curves that are designed to simulate the type of movement seen by one joint. The artificial data sets are used to explore how well the proposed two-stage feature selection method works for curves that are very similar. The artificial data sets are described in detail in Section 5.1.1.

Lastly, the two-stage local min/max feature selection is used with the MBT to generate a tree of motions. Experiments are run using both a three-class kick vs. punch vs. throw problem and using the entire set of motions. For the multi-class problem, two similar feature selection methods are tested. Because each SVM node in the tree creates a binary separation of the classes, the features can either be evaluated with respect to the original class labels (multi-class selection) or with respect to the binary class labels for each node (binary selection). Both methods are tested.

In all tests, accuracy values are calculated using the average of three runs of five folds cross validation. The average number of features required to classify each point is also reported. Because the classifiers are linear SVM classifiers, a point can be classified by projecting the point onto the hyperplane. Hence, the number of features, or dimensionality

of the hyperplane, directly corresponds to the number of multiplications required to classify a new point. The total number of multiplications required depends on the number of nodes used for classification as well as dimensionality, or number of features, at each node.

Data sets

The *human motion data set* consists of nine movements captured by a motion capture system. The data set is provided by Kulić, Takano and Nakamura and was used in their 2008 paper in the Journal of Robotics Research [89]. The movements are manually segmented and labeled. The motion capture system captures 3D positions of passive markers attached to body landmarks; an inverse kinematics solver is used to compute the corresponding joint angles, using a 20 degree of freedom kinematic model. The resulting data set includes 20 joint angle features and one unchanging feature showing the sample rate. Table 5.1 lists the resulting features and indicates the figure numbers illustrating the extracted features. The sampling time feature is not plotted because the sampling time is the same for every time delay value and hence the minimum, maximum, average and all the window values are the same.

The system takes one sample every 33 ms (30 Hz). The segmented motions range in duration from 63 to 139 samples (2.1 to 4.6 seconds) depending on the motion and the trial. Although there are time sequence alignment methods, such as dynamic time warping [136], that would guarantee that key portions of the movement occur in the same time-delay window, these require a-priori knowledge of the class to generate a template for each motion. Using the minimum, maximum and average value of a feature is an extremely rough type of landmark alignment that does not require class knowledge. The lengths of the motions are normalized by dividing each time series into an equal number of windows. This ensures that the start and end of the motions are aligned.

For this data set, 20 non-overlapping windows are used. From inspection, the joint angles of most movements form smooth curves with one or two peaks. Twenty windows was estimated to be sufficient to distinguish the shape of the curves. Two different tests are run, either using the average and variance of the samples in the window or the average of the samples and the difference between two adjacent windows. Both the variance and the difference give an indication of how much the signal is changing in a single window, but the variance is more affected by noise. The absolute minimum, maximum and average are also optionally included.

The *time-series artificial data set* consists of six different Gaussian curves. The curves are described in Table 5.2, and each curve is illustrated in Appendix D. Each curve consists

Features	Figure references Max/min/avg	windowed
Sampling Time	–	–
Right Leg Yaw	D.8	D.28
Right Leg Roll	D.9	D.29
Right Leg Pitch	D.10	D.30
Right Knee	D.11	D.31
Right Ankle Pitch	D.12	D.32
Right Ankle Roll	D.13	D.33
Right Shoulder Pitch	D.14	D.34
Right Shoulder Yaw	D.15	D.35
Right Elbow Roll	D.16	D.36
Right Elbow Pitch	D.17	D.37
Left Leg Yaw	D.18	D.38
Left Leg Roll	D.19	D.39
Left Leg Pitch	D.20	D.40
Left Knee	D.21	D.41
Left Ankle Pitch	D.22	D.42
Left Ankle Roll	D.23	D.43
Left Shoulder Pitch	D.24	D.44
Left Shoulder Yaw	D.25	D.45
Left Elbow Roll	D.26	D.46
Left Elbow Pitch	D.27	D.47

Table 5.1: Measured body features used in the real data sets in this work

name	# curves	mean	std	amp
reg	1	50	20	1
low	1	50	20	0.5
moved	1	70	20	1
skinny	1	50	10	1
stacked	2	50	10	0.75
		50	30	0.25
extra	2	50	20	1
		70	10	0.5

Table 5.2: Description of artificial data set curves

of 100 time samples and each includes 10% noise. The skinny and stacked are also tested with no noise. A set of binary problems is used, comparing each transformed curve to the original curve, as described in Table 5.3.

5.1.2 Results and Discussion

Selecting individual time windows

Tables 5.4 and 5.5 show the accuracy of the SVM classifier on the binary human motion data sets. Tests are run using both CMIM and FOU either selecting the entire time-series feature (all 20 windows) or selecting individual time windows. Tests selecting the

Problem #			
1	reg	vs	low
2	reg	vs	moved
3	reg	vs	skinny
4	reg	vs	stacked
5	reg	vs	extra

Table 5.3: Binary problems with artificial data set

entire feature include all windows and hence include up to 20x the number of inputs as the classifiers using individual time window selection. It is clear from the results in both tables that individual time window selection can be used to drastically reduce the classifier memory requirements and test time, without greatly affecting the classifier accuracy.

It is also interesting to note that the minimum, maximum and average features perform well when used alone, and can improve the accuracy when included with the windowed features.

In general, the feature set that uses the difference between adjacent windows performs better than the feature set that uses the variance within the window. This is likely due to the effect of noise. With the variance features, any small changes within the window contribute to the variance. The difference between windows measures the change in the window averages, so small changes in the signal are averaged out before the difference is calculated. Additionally, the feature set that includes joint angles from the entire body performs slightly better than the feature set that includes only the upper body features. These two differences underscore the importance of having a good set of candidate features to provide to the feature selection measure.

Tables 5.6, 5.7, 5.8, 5.9 and 5.10 show the most commonly selected features when using only minimum, maximum and average features, when selecting the entire feature both with and without minimum, maximum and average features included and selecting the individual time windows both with and without minimum, maximum and average features included.

In most cases, using a larger number of features slightly improves the accuracy (see Tables 5.4 and 5.5). Depending on the test case, two or three single window features are sufficient to match the entire time series using 20 inputs. Estimating the proper number of features is a difficult problem by itself. In many cases, the first feature selected has perfect mutual information with the class, but adding a second or third feature improves the accuracy slightly. This is likely because the effects of noise are reduced as more features are added. However, CMIM does not provide an intuitive stopping criterion or a good way to add extra features as a protection against noise. If the first feature selected has a

				CMIM					
				min-max-	selecting full feature		selecting windows		two-stage
			# features	avg only	w/ Mma	no Mma	w/ Mma	no Mma	no Mma
average & variance	whole body	K / P	1	1.00	1.00	1.00	0.97	0.97	0.97
			2	1.00	1.00	1.00	1.00	1.00	0.99
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T /P	1	0.97	0.97	1.00	0.94	0.94	1.00
			2	0.97	0.97	1.00	0.98	0.98	1.00
			5 and up	1.00	1.00	1.00	0.99	0.99	1.00
	upper body	K / P	1	1.00	1.00	1.00	0.99	0.97	1.00
			2	1.00	1.00	1.00	0.99	0.99	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	0.99	0.99	1.00	0.92	0.90	0.96
			2	1.00	1.00	1.00	0.96	0.96	1.00
			5 and up	1.00	1.00	1.00	1.00	0.99	1.00
average & difference	whole body	K / P	1	1.00	1.00	1.00	1.00	1.00	1.00
			2	1.00	1.00	1.00	1.00	1.00	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	1.00	1.00	1.00	1.00	1.00	1.00
			2	1.00	1.00	1.00	1.00	1.00	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
	upper body	K / P	1	1.00	1.00	1.00	1.00	1.00	1.00
			2	1.00	1.00	1.00	1.00	1.00	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	1.00	1.00	1.00	0.98	0.98	0.99
			2	1.00	1.00	1.00	0.99	0.99	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00

Table 5.4: SVM classification accuracy on kick vs. punch and throw vs. punch data sets selecting either the entire feature, individual time windows from the feature and using a two-stage selection selected using SFS and CMIM. Tests selecting the entire feature include all windows and hence include up to 20x the number of inputs as the classifiers using individual time window selection.

				FOU					
				min-max-	selecting full feature		selecting window		two-stage
			# features	avg only	w/ Mma	no Mma	w/ Mma	no Mma	no Mma
average & variance	whole body	K / P	1	1.00	1.00	1.00	0.97	0.97	0.97
			2	1.00	0.99	1.00	0.97	0.97	0.97
			5	1.00	0.99	1.00	1.00	1.00	1.00
			10 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	0.97	0.97	1.00	0.94	0.94	0.97
			2	0.97	0.97	1.00	0.94	0.94	0.97
			5	1.00	1.00	1.00	0.93	0.93	1.00
			10	1.00	1.00	1.00	0.97	0.97	1.00
			15	1.00	1.00	1.00	0.97	0.97	1.00
			20	1.00	1.00	1.00	0.99	0.99	1.00
			30 and up	1.00	1.00	1.00	1.00	1.00	1.00
	upper body	K / P	1	1.00	1.00	1.00	0.99	0.97	1.00
			2	1.00	0.98	1.00	0.99	0.99	1.00
			5	1.00	0.98	1.00	1.00	1.00	1.00
			10 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	0.99	0.99	1.00	0.92	0.90	0.96
			2	1.00	0.97	1.00	0.89	0.88	0.96
			5	1.00	1.00	1.00	0.95	0.94	1.00
			10	1.00	1.00	1.00	0.95	0.93	0.99
			15	1.00	1.00	1.00	0.97	0.97	1.00
			all	1.00	1.00	1.00	1.00	1.00	1.00
average & difference	whole body	K / P	1	1.00	1.00	1.00	1.00	1.00	1.00
			2	1.00	1.00	1.00	1.00	1.00	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	0.97	0.97	1.00	0.99	0.99	1.00
			2	0.97	0.97	1.00	0.99	0.99	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
	upper body	K / P	1	1.00	1.00	1.00	1.00	1.00	1.00
			2	1.00	1.00	1.00	1.00	1.00	1.00
			5 and up	1.00	1.00	1.00	1.00	1.00	1.00
		T / P	1	0.98	0.98	1.00	0.98	0.98	0.99
			2	0.99	0.99	1.00	0.98	0.98	1.00
			5	1.00	1.00	1.00	1.00	1.00	1.00
			10	1.00	1.00	1.00	0.99	0.97	0.99
			15	1.00	1.00	1.00	1.00	1.00	0.99
			all	1.00	1.00	1.00	1.00	1.00	1.00

Table 5.5: SVM classification accuracy on kick vs. punch and throw vs. punch data sets selecting either the entire feature, individual time windows from the feature and using a two-stage selection selected using SFS and FOU. Tests selecting the entire feature include all windows and hence include up to 20x the number of inputs as the classifiers using individual time window selection.

		All features						Upper body only					
		kick vs. punch			throw vs. punch			kick vs. punch			throw vs. punch		
		feature	type	fig.	feature	type	fig.	feature	type	fig.	feature	type	fig.
avg. & var.	CMIM	R Leg Pitch	avg	D.10	R Elbow Roll	max	D.16	R Elbow Pitch	min	D.17	R Elbow Roll	max	D.16
			min		R Leg Roll	min	D.9	R Elbow Pitch	max	D.17	L Elbow Roll	avg	D.26
		R Leg Yaw	max	D.8	L Leg Pitch	min	D.20	L Elbow Pitch	avg	D.27	R Elbow Pitch	min	D.17
		L Leg Pitch	avg	D.20		avg		L Elbow Pitch	max	D.27	L Elbow Pitch	avg	D.27
					R Leg Pitch	max		L Elbow Roll	max	D.26	R Elbow Pitch	avg	D.17
	FOU	Sampling Time	max		Sampling Time	max		R Elbow Pitch	max	D.17	R Elbow Roll	max	D.16
			min			min			min			min	
			avg			avg		R Elbow Roll	min	D.16	L Shoulder Pitch	max	D.24
		R Leg Pitch	min	D.10	R Elbow Roll	max	D.16	L Shoulder Pitch	min	D.24	L Elbow Pitch	min	D.27
		R Elbow Pitch	max	D.17	R Leg Pitch	max	D.10	L Elbow Pitch	avg	D.27	R Elbow Pitch	max	D.17
		R Leg Yaw	max	D.8	R Leg Roll	min	D.9						
avg. & diff.	CMIM	R Leg Pitch	min	D.10	R Elbow Roll	max	D.16	R Elbow Pitch	max	D.17	R Elbow Roll	max	D.16
			avg		R Leg Roll	min	D.9		min		L Elbow Roll	avg	D.26
		R Leg Yaw	max	D.8	L Leg Pitch	min	D.20	L Elbow Pitch	avg	D.27	L Elbow Pitch	avg	D.27
		L Leg Pitch	avg	D.20		avg			max		R Elbow Pitch	avg	D.17
		R Elbow Pitch	max	D.17	R Leg Pitch	max	D.10	L Shoulder Pitch	min	D.24		min	
	FOU	Sampling Time	max		Sampling Time	max		R Elbow Pitch	max	D.17	R Elbow Roll	max	D.16
			min			min		R Elbow Roll	min	D.16	L Shoulder Pitch	max	D.24
			avg			avg			max		L Elbow Pitch	min	D.27
		R Elbow Pitch	max	D.17	R Elbow Roll	max	D.16	L Shoulder Pitch	min	D.24	R Shoulder Pitch	min	D.14
		L Elbow Pitch	max	D.27	R Leg Pitch	max	D.10	R Shoulder Yaw	max	D.15			

Table 5.6: Commonly selected features when selecting among minimum, maximum and average features

		All Features						Upper body only					
		kick vs. punch			throw vs. punch			kick vs. punch			throw vs. punch		
		feature	type	fig.	feature	type	fig.	feature	type	fig.	feature	type	fig.
avg. & var.	CMIM	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47
		R Elbow Pitch	win avg	D.37	R Leg Roll	win avg	D.29	R Elbow Pitch	win avg	D.37	L Elbow Roll	win avg	D.46
		R Knee	win var	D.31	L Elbow Roll	win avg	D.46	L Shoulder Pitch	win avg	D.44	R Elbow Pitch	win avg	D.37
		R Ankle Pitch	win avg	D.32	R Elbow Pitch	win avg	D.37	L Shoulder Yaw	win var	D.45	R Elbow Roll	win avg	D.36
		L Shoulder Yaw	win var	D.45				R Shoulder Yaw	win avg	D.35	L Shoulder Yaw	win avg	D.45
	FOU	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Roll	win var	D.46	L Elbow Pitch	win avg	D.47
		L Knee	win var	D.41	R Leg Roll	win avg	D.29	R Elbow Pitch	win avg	D.37	L Elbow Roll	win avg	D.46
		R Leg Roll	win avg	D.29	L Elbow Roll	win avg	D.46		win var		R Elbow Roll	win var	D.36
		L Ankle Pitch	win avg	D.42	R Leg Roll	win var	D.29	L Shoulder Yaw	win var	D.45		win avg	
		R Knee	win var	D.31	R Leg Yaw	win var	D.28	R Shoulder Yaw	win var	D.35	L Shoulder Yaw	win var	D.45
		Sampling Time	win avg										
avg. & diff.	CMIM	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47
		R Knee	win diff	D.31	L Elbow Roll	win avg	D.46	L Shoulder Pitch	win avg	D.44	L Elbow Roll	win avg	D.46
		R Leg Pitch	win diff	D.30	R Leg Roll	win avg	D.29		win diff		R Elbow Roll	win avg	D.36
		R Elbow Pitch	win avg	D.37		win diff		R Elbow Pitch	win avg	D.37	R Elbow Pitch	win avg	D.37
		R Ankle Pitch	win avg	D.32	R Elbow Pitch	win avg	D.37	L Shoulder Yaw	win diff	D.45	L Shoulder Yaw	win avg	D.45
	FOU	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47
		Sampling Time	win avg		R Leg Roll	win avg	D.29	R Elbow Pitch	win diff	D.37	L Elbow Roll	win avg	D.46
		R Leg Roll	win avg	D.29		win diff			win avg		R Elbow Roll	win avg	D.36
		L Knee	win diff	D.41	L Elbow Roll	win avg	D.46	L Elbow Roll	win diff	D.46		win diff	
		L Shoulder Yaw	win diff	D.45	Sampling Time	win avg		L Shoulder Pitch	win avg	D.44	L Elbow Pitch	win diff	D.47

Table 5.7: Commonly selected features when selecting entire feature (not including minimum, maximum and average)

		All features						Upper body only					
		kick vs. punch			throw vs. punch			kick vs. punch			throw vs. punch		
		feature	type	fig.	feature	type	fig.	feature	type	fig.	feature	type	fig.
avg. & var.	CMIM	R Leg Pitch	abs avg	D.30	R Elbow Roll	abs max	D.36	R Elbow Pitch	abs min	D.37	R Elbow Roll	abs max	D.36
			abs min		R Leg Roll	abs min	D.29	R Elbow Pitch	abs max	D.37	L Elbow Roll	abs avg	D.46
		R Leg Yaw	abs max	D.28	R Leg Pitch	abs max	D.30	L Elbow Pitch	abs avg	D.47	R Elbow Pitch	abs min	D.37
		L Leg Pitch	abs avg	D.40	L Leg Pitch	abs min	D.40	L Elbow Pitch	abs max	D.47	L Elbow Pitch	abs avg	D.47
						abs avg		L Elbow Roll	abs max	D.46	R Elbow Pitch	abs avg	D.37
	FOU	Sampling Time	win avg		Sampling Time	win avg		R Elbow Pitch	abs max	D.37	R Elbow Roll	abs max	D.36
			win var			win var			win var			win var	
			abs max		R Elbow Roll	abs max	D.36	L Shoulder Pitch	abs min	D.44	L Shoulder Pitch	win var	D.44
		L Elbow Pitch	win avg	D.47	L Elbow Pitch	win avg	D.47	L Elbow Pitch	win var	D.47	L Elbow Roll	win avg	D.46
		R Leg Pitch	abs min	D.30	R Leg Roll	abs min	D.29		abs avg		R Elbow Pitch	win avg	D.37
					R Leg Pitch	abs max	D.30				L Elbow Pitch	win avg	D.47
avg. & diff.	CMIM	R Leg Pitch	abs min	D.30	R Elbow Roll	abs max	D.36	R Elbow Pitch	abs max	D.37	R Elbow Roll	abs max	D.36
			abs avg		R Leg Roll	abs min	D.29		abs min		L Elbow Roll	abs avg	D.46
		R Leg Yaw	abs max	D.28	L Leg Pitch	abs min	D.40	L Elbow Pitch	abs avg	D.47	L Elbow Pitch	abs avg	D.47
		L Leg Pitch	abs avg	D.40		abs avg			abs max		R Elbow Pitch	abs avg	D.37
		R Elbow Pitch	abs max	D.37	R Leg Pitch	abs max	D.30	L Shoulder Pitch	abs min	D.44		abs min	
	FOU	Sampling Time	win avg		Sampling Time	win avg		R Elbow Pitch	abs max	D.37	R Elbow Roll	abs max	D.36
			win diff			win diff			win diff		L Elbow Roll	win avg	D.46
			abs max		R Elbow Roll	abs max	D.36		win avg		L Elbow Pitch	win avg	D.47
		R Elbow Pitch	abs max	D.37	L Elbow Pitch	win avg	D.47	L Elbow Roll	win diff	D.46	R Elbow Pitch	win avg	D.37
		L Elbow Pitch	abs max	D.47	R Leg Pitch	abs max	D.30	L Elbow Pitch	win diff	D.47	R Shoulder Pitch	win diff	D.34

Table 5.8: Commonly selected features when selecting entire feature and the minimum, maximum and average are included as separate features in the set

		All features								Upper body only							
		kick vs. punch				throw vs. punch				kick vs. punch				throw vs. punch			
		feature	win	type	fig.	feature	win	type	fig.	feature	win	type	fig.	feature	win	type	fig.
avg. & var.	CMIM	R Leg Yaw	11	avg	D.28	R Elbow Pitch	6	var	D.37	R Shoulder Pitch	10	avg	D.34	R Elbow Roll	5	avg	D.36
			12	avg		L Leg Pitch	13	var	D.40		11	avg			6	avg	
			10	avg		R Elbow Roll	20	var	D.36		12	avg			4	avg	
		R Shoulder Pitch	10	avg	D.34	R Leg Yaw	4	var	D.28	L Shoulder Pitch	13	avg	D.44	R Elbow Pitch	6	var	D.37
		L Leg Yaw	15	avg	D.38	L Leg Yaw	4	var	D.38		12	avg			8	avg	
		R Ankle Pitch	11	avg	D.32												
	FOU	Sampling Time	1,2	avg		Sampling Time	1	avg		L Shoulder Pitch	2	var	D.44	L Elbow Pitch	18	var	D.47
			1	var			2	avg		R Shoulder Pitch	1	var	D.34	R Elbow Pitch	6	var	D.37
		R Leg Yaw	10	avg	D.28		1	var			10	avg			10	var	
			11	avg		L Leg Pitch	13	var	D.40	R Elbow Roll	20	var	D.36	R Shoulder Pitch	1	var	D.34
						R Elbow Pitch	6	var	D.37	L Elbow Roll	12	avg	D.46	R Elbow Roll	2	var	D.36
avg. & diff.	CMIM	R Leg Yaw	11	avg	D.28	L Leg Pitch	13	avg	D.40	L Shoulder Pitch	12	avg	D.44	R Elbow Roll	5	avg	D.36
			12	avg			14	avg			13	avg			6	avg	
			10	avg			15	avg		R Shoulder Pitch	11	avg	D.34		4	avg	
		L Leg Pitch	11	avg	D.40	L Leg Yaw	15	avg	D.38		12	avg		R Elbow Pitch	6	diff	D.37
		R Leg Pitch	10	avg	D.30		12	diff			10	avg			7	diff	
	FOU	Sampling Time	1	avg		Sampling Time	1	avg		R Shoulder Pitch	1	diff	D.34	R Shoulder Pitch	1	diff	D.34
			2	avg			1	diff			9	avg		R Shoulder Yaw	1	diff	D.35
			1	diff		R Leg Yaw	1	diff	D.28	R Shoulder Yaw	1	diff	D.35	R Elbow Roll	1	diff	D.36
		L Leg Pitch	12	avg	D.40	L Leg Pitch	14	avg	D.40	R Elbow Roll	1	diff	D.36	R Elbow Pitch	6	diff	D.37
		R Leg Yaw	11	avg	D.28	R Elbow Pitch	6	diff	D.37	L Elbow Roll	11	avg	D.46		13	avg	

Table 5.9: Commonly selected features when selecting individual time samples (not including minimum, maximum and average)

		All features								Upper body only							
		kick vs. punch				throw vs. punch				kick vs. punch				throw vs. punch			
		feature	win	type	fig.	feature	win	type	fig.	feature	win	type	fig.	feature	win	type	fig.
avg. & var.	CMIM	L Leg Yaw	15	avg	D.38	R Elbow Pitch	6	var	D.37	R Shoulder Pitch	10	avg	D.34	R Elbow Roll	4,5,6	avg	D.36
		R Shoulder Pitch	10	avg	D.34	L Leg Pitch	13	var	D.40	L Shoulder Pitch	12	avg	D.44		5	avg	
		R Ankle Pitch	11	avg	D.32	R Elbow Roll	20	var	D.36	R Shoulder Pitch	11	avg	D.34		6	avg	
		R Leg Yaw	11	avg	D.28	R Leg Yaw	4	var	D.28	R Shoulder Pitch	12	avg	D.34		abs	max	
			10	avg		L Leg Yaw	4	var	D.38					R Elbow Pitch	6	var	D.37
	FOU	Sampling Time	1	avg		Sampling Time	1	avg		L Shoulder Pitch	2	var	D.44	L Elbow Pitch	18	var	D.47
			2	avg			2	avg		R Shoulder Pitch	1	var	D.34	R Elbow Pitch	6	var	D.37
			1	var			1	var			10	avg		R Elbow Roll	20	var	D.36
		R Leg Yaw	10	avg	D.28	L Leg Pitch	13	var	D.40	L Elbow Pitch	11	var	D.47		2	var	
		R Ankle Pitch	11	avg	D.32	R Elbow Pitch	6	var	D.37	L Elbow Roll	12	avg	D.46	R Shoulder Pitch	1	var	D.34
avg. & diff.	CMIM	R Leg Yaw	11	avg	D.28	L Leg Pitch	14	avg	D.40	R Shoulder Pitch	11	avg	D.34	R Elbow Roll	5	avg	D.36
			12	avg			13	avg		L Shoulder Pitch	12	avg	D.44		21	avg	
			10	avg			15	avg			13	avg			6	avg	
		L Leg Pitch	11	avg	D.40	L Leg Yaw	15	avg	D.38	R Shoulder Pitch	12	avg	D.34		4	avg	
		R Leg Pitch	10	avg	D.30		12	diff			10	avg		R Elbow Pitch	6	diff	D.37
	FOU	Sampling Time	1	avg		Sampling Time	1	avg		R Shoulder Pitch	1	diff	D.34	R Shoulder Pitch	1	diff	D.34
			1	diff			1	diff			9	avg		R Shoulder Yaw	1	diff	D.35
		R Leg Yaw	1	diff	D.28	R Leg Yaw	1	diff	D.28	R Shoulder Yaw	1	diff	D.35	R Elbow Roll	1	diff	D.36
			10	avg		L Leg Pitch	14	avg	D.40	R Elbow Roll	1	diff	D.36	L Shoulder Pitch	7	diff	D.44
		L Leg Pitch	12	avg	D.40	L Leg Yaw	15	avg	D.38	L Elbow Roll	11	avg	D.46	R Elbow Pitch	6	diff	D.37

Table 5.10: Commonly selected features when selecting individual time samples and the minimum, maximum and average are included as separate features in the set

mutual information of 1, the CMI with any additional features will be zero, as adding a new feature will never improve the mutual information. In these cases, the feature selection defaults to univariate mutual information and Fisher's interclass separability criterion as a tie-breaking mechanism, neither of which discourage correlated features.

FOU does include a term to discourage the inclusion of correlated features. However, as discussed in Chapter 4, the feature-feature correlation term undervalues the correlated features. As seen in Table 5.9 and 5.10, the most commonly selected second feature is the sampling time. However, the sampling time feature is actually a very poor choice, as it is the same value for all time values and classes. This is because the conditional mutual information term is zero, due to the high mutual information of the first feature. The feature-feature correlation is greater than zero, making the FOU value negative, whereas the FOU value for the sampling time is zero.

The poor feature selection of FOU affects the accuracy. Although the accuracies of the FOU and CMIM are similar in most cases, there are tests where CMIM outperforms FOU. For example, in the throw vs. punch classification problem, CMIM gives a better performance with fewer features on both the upper only and whole body problems, especially on the average/variance data sets (see Table 5.4 5.5).

Two-stage feature selection

As seen in Tables 5.9 and 5.10, many of the selected windows occur near the local minimum or maximum of the time signal. Examples of this are illustrated in Figures D.30 and D.40. Intuitively, this makes sense as the most distinguishing windows are likely to occur where the values are the most different or are changing the most. This observation suggests a possible method to simplify the feature selection process.

In the two-stage selection, the pool of candidate features is first narrowed by only including windows that are the local minimum or maximum. This two-stage selection is applied to the binary human motion data and outperforms the full time-series selection in some cases (see Tables 5.4 and 5.5).

The two-stage selection is also tested on several single-feature time series artificial data sets, detailed in Tables 5.2 and 5.3. Each curve has a slightly different distinguishing feature and the artificial data sets are designed to test the conditions under which the two stage feature selection will work.

The results from the artificial data sets are given in Table 5.11. They illustrate the ability of the two-stage feature selection to find good features even on difficult data sets.

	# features	with noise					no noise	
		reg. vs. low	reg. vs. moved	reg. vs. skinny	reg. vs. stacked	reg. vs. extra	reg. vs. skinny	reg. vs. stacked
using variance	1	1.00	1.00	1.00	0.73	1.00	1.00	1.00
	2	1.00	1.00	1.00	0.80	1.00	1.00	1.00
	3	1.00	1.00	1.00	0.82	1.00	1.00	1.00
	4	1.00	1.00	1.00	0.81	1.00	1.00	1.00
	5	1.00	1.00	1.00	0.83	1.00	–	1.00
	6	1.00	1.00	1.00	0.85	1.00	–	–
	7	1.00	1.00	1.00	0.87	1.00	–	–
	8	1.00	1.00	1.00	0.88	1.00	–	–
	9	1.00	1.00	1.00	0.88	1.00	–	–
using difference	1	1.00	1.00	1.00	0.93	1.00	1.00	1.00
	2	1.00	1.00	1.00	0.99	1.00	1.00	1.00
	3	1.00	1.00	1.00	0.99	1.00	1.00	1.00
	4	1.00	1.00	1.00	0.99	1.00	1.00	1.00
	5	–	1.00	1.00	0.98	1.00	–	1.00
	6	–	1.00	1.00	0.99	1.00	–	–
	7	–	1.00	1.00	0.98	1.00	–	–
	8	–	1.00	1.00	0.99	–	–	–
	9	–	–	1.00	–	–	–	–

Table 5.11: SVM accuracy on artificial data sets using two-stage feature selection. Dashes indicate feature sets with fewer than the requested number of local minimums.

The two data sets that are inseparable are the regular vs. skinny, where the two curves are the same height and shape, but with one having a slightly lower standard deviation, and the regular vs. stacked, where the second curve consists of two summed Gaussians that have the same mean and different standard deviations. As seen in Figures D.3 and D.4, these two curves are quite difficult to distinguish. However, when the 10% added noise is removed, the two-stage feature selection is able to achieve 100% separation.

As with binary human motion data sets, CMIM tends to select correlated features when the mutual information of the first feature is high. For example, please see Figures D.3, D.6 and D.7 where the selected features are the mirrored features on either side of the symmetric curve. These features are highly correlated, but equally predictive. In the noise free data sets, the features from either side of the curve are fully redundant. Although this type of perfect symmetry is not often seen in real data sets, the artificial data sets clearly illustrate the problem with feature selection measures that do not take feature-feature correlation into account.

Tree-based classification

Tables 5.12 and 5.13 show the accuracy of the MBT classifier on a three-class and a nine-class human motion classification problem. The tables also give the average number of

	feature selection			
	multi-class		binary	
	accuracy	# features	accuracy	# features
avg. & var.	1.00	7.9	1.00	7.9
avg. & diff.	1.00	7.9	1.00	7.9

Table 5.12: Accuracy of multi-modal tree-based classification on three class human motion problem. Multi-class feature selection evaluates each feature with respect to the original class labels, binary feature selection evaluates each feature with respect to the binary division at each node. Avg. & var. features use the window average and variance as the original candidate features, Avg. & diff. features use the window average and difference between adjacent windows as the candidate features. Accuracy is calculated using an average of three runs of five-fold cross validation. # features gives the average number of features required to classify a new point.

	feature selection			
	multi-class		binary	
	accuracy	# features	accuracy	# features
avg. & var.	1.00	16.2	1.00	16.1
avg. & diff.	1.00	15.9	1.00	15.8

Table 5.13: Accuracy of multi-modal tree-based classification on nine class human motion problem. Multi-class feature selection evaluates each feature with respect to the original class labels, binary feature selection evaluates each feature with respect to the binary division at each node. Avg. & var. features use the window average and variance as the original candidate features, Avg. & diff. features use the window average and difference between adjacent windows as the candidate features. Accuracy is calculated using an average of three runs of five-fold cross validation. # features gives the average number of features required to classify a new point.

features required to classify a new point.

It is clear from the table that the MBT is capable of accurately classifying the human motion data. The computational complexity is quite low, requiring approximately 16 multiplications to classify a new point.

Figure 5.2 shows the tree generated for the three-class motion classification problem using multi-class feature selection and window average and difference features. The generated tree mimics the intuitive separation of the motions, with the kick motion separated first from the more similar throw and punch motions.

Figure 5.3 shows the tree generated for the nine-class motion classification problem using multi-class feature selection and window average and difference features. As with the three-class problem, the tree matches the intuitive class separation where the kick

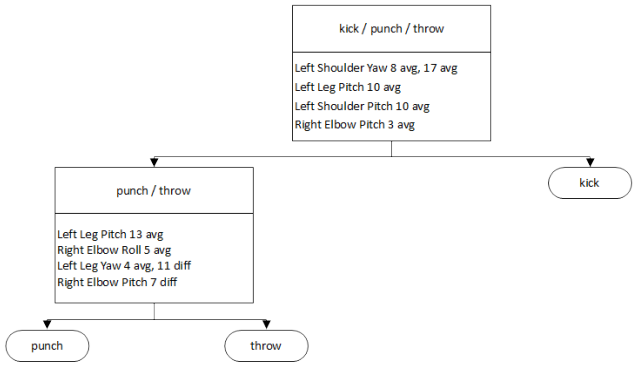


Figure 5.2: Tree-based classifier structure for three-class human motion classification problem

motion is separated from the punch and throw motions at the highest node of the tree and the similar punch and throw motions are separated in the lower nodes.

5.2 Hand gesture recognition from Electromyography (EMG) sensor data

Tests with the EMG hand gesture recognition data sets are designed to explore whether feature selection is beneficial for features that have both temporal and spatial components and whether the MBT classifier can be used successfully for this type of difficult motion recognition problem.

The EMG hand gesture recognition data set is provided by Thalmic Labs Inc. Thalmic Labs uses an armband with eight EMG sensors to recognize hand gestures from muscle activation. The challenge is to recognize hand gestures based on samples from other sub-

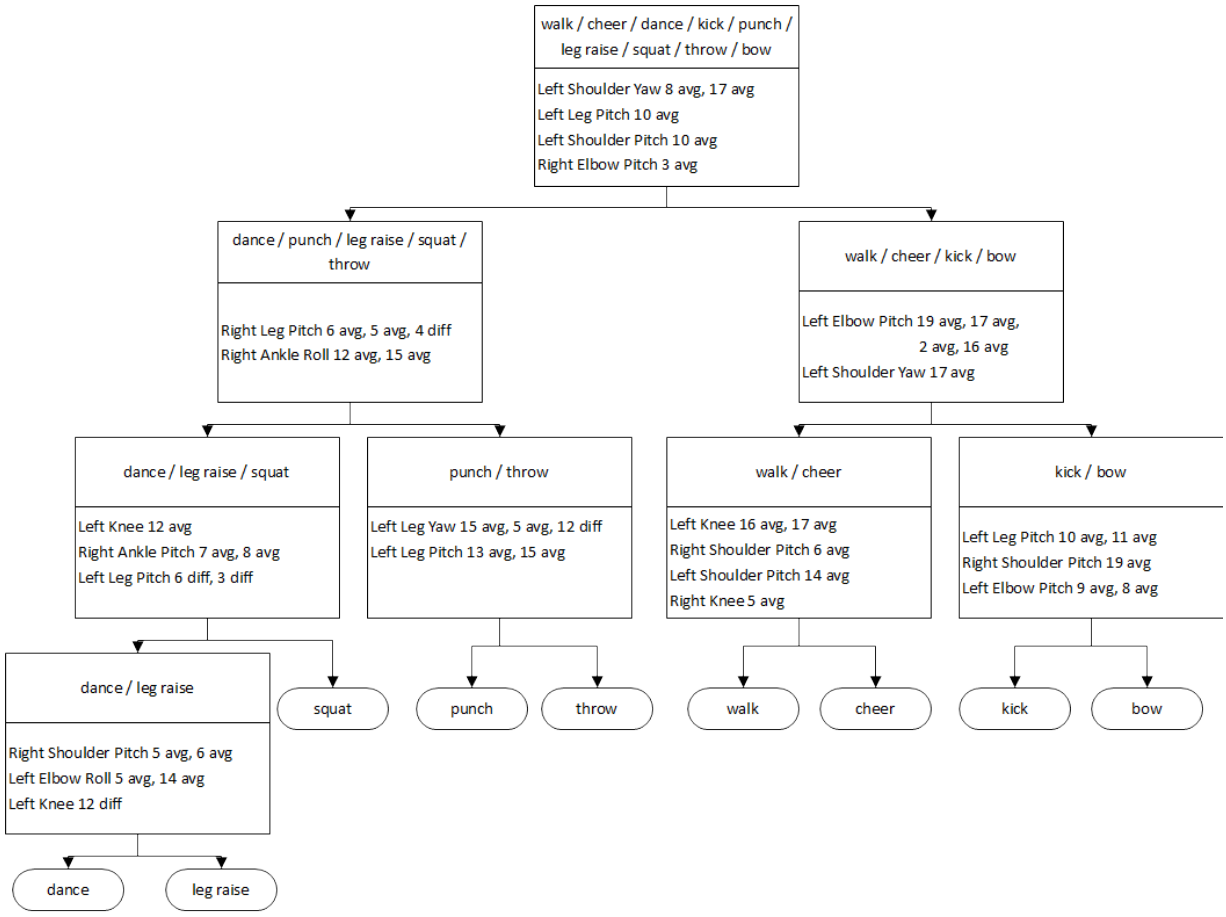


Figure 5.3: Tree-based classifier structure for nine-class human motion classification problem

jects. The data sets in this section were generated using two different prototype versions of the Thalmic Labs' MYO armband.

This data set is challenging for a number of reasons. The signal itself is noisy because the EMG sensors are skin mounted. The sensors are detecting electrical activation of the muscle through a layer of fat and skin. Because the sensors are spatially removed from the muscles, they pick up electrical activations from a number of different muscles at a time. Additionally, the sensors are not guaranteed to be placed directly over the desired arm muscle.

The magnitude of the signal varies from person to person and trial to trial based on a number of different factors [31]. A larger amount of fat on the arm can reduce the signal strength because the signal has to travel further though the fat layer. Conversely, an arm that has a small diameter can cause the sensors to lie slack on the skin, reducing the signal amplitude and increasing noise due to lack of fixed contact with the arm. Similarly, a large amount of hair on the arm can also cause contact problems, as can an excessive amount of sweat. This is particularly problematic as this can change during a session wearing the band. These problems affect the ability to normalize the signal and perform segmentation as the threshold for signal activation differs from person to person.

The gesture recognition itself is also difficult as people employ multiple movement strategies for each gesture, and the strategy can vary from person to person or even from trial to trial within the same subject.

The noisiness of the signal can be overcome by windowing the signal and applying a filter or calculating a feature value. Thalmic labs have suggested a set of exemplary features that can be applied to windowed data. The exemplar features are detailed in Section 5.2.1. Using these features creates another challenge, however. Each of the 52 features is calculated for 8 sensors and W time windows, where W is variable. Values from 5 to 40 are tested. This gives an $S \times W$ matrix of feature values for each feature. This is an extremely large number of features.

The challenges related to the large number of input features and feature normalization can be overcome by calculating meta-features from the initial set of candidate features.

In Section 5.1, it was shown that the majority of the selected features appear at or near to a local maximum of the average or variance of a time-series feature. The features used in this dataset are slightly different, as they include not only a time component, but also a spatial component relating to the different sensors around the arm. This is illustrated in Figure 5.4.

In the feature selection proposed in Section 5.1, the input to the classifier is the feature value at a certain window. The meta-features tested for EMG signal recognition include

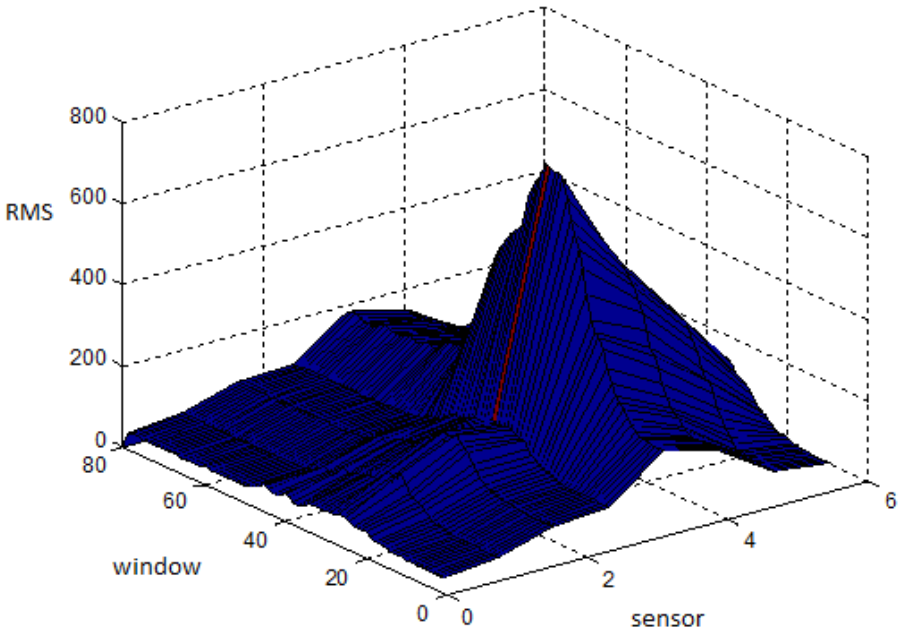


Figure 5.4: RMS values for a single trial of the "snap" motion

a normalized version of the feature value at a certain sensor and window, but also use a transposed version of that method, such that the meta-feature describe the time window and sensor value where the feature 2-D peak occurs. Meta-features describing the width of the peaks are also included. These meta-feature values naturally have a similar range of values because they are related to peak location in a fixed size grid, rather than signal magnitude.

Various sets of features and meta-features are tested with the different forms of the MBT algorithm. The single level MBT described in Chapter 3 Section 3.2 is compared to a two-level tree and a one vs. one formulation. Because the MBT uses only the class centers as a way to generate the output classes for each node, classes with a large spread can be divided early in the tree. In theory this is not a problem, since the class mean will change in the lower nodes, and the class will eventually be separated. However, when the number of samples is small, lower nodes will have fewer samples, which can affect the generalization capability of the classifier. A two-stage tree first designs the tree-structure, then uses MBT at each binary node. Because the MBT guarantees the training samples are correctly classified, each class appears only in one high level output. More easily separable classes can be removed higher in the tree, to reduce the chance of misclassification for the

easier classes and reduce the number of classes and samples to be considered in the lower nodes of the tree.

The separability of two classes can be estimated using Fisher's interclass separability criterion. Calculating this value for each pair of glasses creates a $C \times C$ matrix of Fisher values. This matrix is used to create a undirected graph, where each node is a class and the weight between each class is the negative Fisher score. The minimum graph cut is then used to recursively separate the graph and define the top level tree. An MBT is used to fully separate each level of the tree. This effectively separates the tree design from the piecewise linear separation of classes. The same technique can be applied directly to other multi-class techniques, using a binary MBT in place of a single level classifier. All three methods are compared.

5.2.1 Experiments

Three sets of experiments are performed on two different versions of the EMG hand gesture recognition data set. Each successive group of experiments adjusts the candidate set of features as more features were added.

For every test, the accuracy of the classifier is assessed using leave-one-subject-out (LOSO) cross validation, where the data from a single subject is removed from the data set and used for testing. The classifier is trained on the data from the remaining subjects. This more closely reflects the real use of the armband, where the classifier will be used to identify hand gestures on new subjects not present in the training data.

For all tests, the data is windowed, and the features described in Section 5.2.1 are calculated within each window for each sensor. Meta-features are then calculated as described in Section 5.2.1. The meta-features form the inputs to the MBT algorithm and feature selection is performed in each node.

The first set of experiments use the features originally suggested as exemplar features by Thalmic Labs (features 1-51 in Table 5.16) and a subset of the meta-features (meta-features 1-4 and 17-20 in Table 5.17). Tests were run using single and two-level MBT. These experiments use an older, eight gesture EMG data set, *EMG 1*. The data set is described further in Section 5.2.1. The motions are pre-segmented, using the segmentation algorithm provided by Thalmic, described in Section 5.2.1. The features are calculated in 20 overlapping windows.

The second set of experiments is performed on a newer version of the data set, *EMG 2*, which is generated using a newer prototype armband. The features used include the original

features as well as the envelope and polar features (features 52-54 in Table 5.16) and the entire set of meta-features. The polar feature matrix is also included as individual features. Features are calculated over a set of either 5 or 40 overlapping windows. Additionally, each feature in Table 5.16 is calculated for each sensor using a window that includes all the time samples. These features were included because there are several candidate features that work well to characterize the time-series as a whole. For example, skewness and kurtosis over the entire signal should be able to describe the behaviour of the peak in time and across the sensors.

A final test is performed using the simple square integral, wavelet and Hjorth parameter features (features 5, 16-31 and 33-35 from Table 5.16) calculated over the entire window, as well as all of the cosine features (features 53 and 54), which are identified in earlier tests as strong candidate features. The pairwise sensor comparison feature (f_{coss}) is calculated using window size (W_{sz}) of 20 and a set window shift (W_{sh}) of 5. The pairwise time comparison cosine feature (f_{cosw}) is calculated using ten overlapping windows. Segmentation is performed only for feature f_{cosw}. Tests are run using a single stage multi-modal SVM tree, a two-stage multi-modal SVM tree and a one vs. one formulation where each individual pairwise classifier is a multi-modal SVM tree.

Data sets

The data sets used in this section were provided and collected by Thamic Labs Inc., in collaboration with the Adaptive Systems Laboratory at the University of Waterloo. The data sets used in this section are generated using two early prototypes of the MYO armband from Thalmic labs. The prototype MYO armbands use eight EMG sensors around the arm. It is placed just below the elbow on the arm and records the eight EMG sensor signals at 300 Hz.

The first data set, *EMG 1* uses an earlier prototype version of the hardware. The data set was collected by Thalmic labs and consists of ten trials each of eight gestures being performed by nine different subjects. The gestures are described in Table 5.14. The original data set consists of 800 samples per trial, but includes a short rest period at the start and end of the gestures. Thalmic labs also provided an exemplary segmentation algorithm, which labels the active part of the gesture. The segmentation is based on the root mean squared (RMS) value of the signal. A baseline signal value is established at the start of the gesture, where the person is assumed to be at rest. The active portion of the gesture starts when at least two channels pass a threshold and the gesture is considered to be complete once those two channels fall below a second threshold.

#	gesture
1	rest
2	fist
3	left
4	right
5	snap
6	rock on
7	gun
8	finger tap

Table 5.14: Hand gestures from first MYO armband data set

#	gesture
1	rest
2	fist
3	paddle in
4	paddle out
5	snap
6	spider man
7	gun
8	thumb index finger tap
9	thumb middle finger tap
10	thumb ring finger tap
11	thumb pinkie finger tap
12	point with index
13	point with inde and middle
14	point with index, middle and ring
15	thumbs up
16	talking hand

Table 5.15: Hand gestures from second MYO armband data set

The second data set *EMG 2* uses a more recent prototype of the hardware. The data set includes 5 trials each of 16 gestures by 25 subjects. The gestures are described in Table 5.15. A subset of the gestures is selected for testing as many of the gestures are quite similar. The second data set uses a slightly improved segmentation algorithm, also provided by Thalmic labs. The segmentation uses the entire rest gesture to establish a baseline for the gestures. The activation thresholds are set dynamically based on the maximum signal value for each gesture.

Features

Thalmic labs provided a set of exemplary features, as suggested in [123], described in table 5.16. Each EMG signal is segmented and divided into overlapping windows. Each feature is calculated within each window, giving a matrix of values for each feature. Some features suggested by Thalmic are described in Section 5.2.1. Later tests also include the signal Envelope (suggested by Ali-Akbar Samadani), polar co-ordinates (suggested by Thalmic

#	Feature
1	Root mean squared
2	Integrated EMG
3	Mean absolute value
4	Mean absolute value slope
5	Simple square integral
6	Variance
7	Waveform change
8	Zero crossings
9	Slope sign change
10	Willson Amplitude
11-15	Autoregressive coefficient
16-31	Wavelet features
32	Ratio of range to standard deviation
33-35	Hjorth paramters
36	Mean
37	Skewness
38	Kurtosis
39	Accumuated energy
40	Average non-linear energy
41-44	Power in 4 frequency bands
45	Mean power frequency
46	Mid power frequency
47	Peak power frequency
48-50	Spectrum features
51	Frequency Ratio
52	Envelope
53-54	Polar Coordinates
55-56	Cosine of sensor or time vectors

Table 5.16: Features calculated from EMG data for hand gesture recognition

labs) and two transposed versions of the cosine angle (suggested by Ali-Akbar Samadani and Thalmic labs).

Root mean squared values (RMS) RMS values are calculated in a window as:

$$f_{rms}(s,w) = \sqrt{\frac{1}{W_n}\sum_{t=1}^{W_n} EMG(t)^2} \tag{5.1}$$

where $f_{rms}(s,w)$ is the RMS value for sensor s in time window w, W_n is the number of samples in the window w and $EMG(t)$ is the EMG value for sample t of the window.

Integrated EMG The integrated EMG is calculated as:

$$f_{int}(s,w) = \sum_{t=1}^{W_n} |EMG(t)| \tag{5.2}$$

Mean absolute value and slope of the mean absolute value The mean absolute value is calculated as:

$$f_{mav}(s,w) = \frac{1}{W_n}\sum_{t=1}^{W_n} |EMG(t)| \tag{5.3}$$

The slope of the mean absolute value is calculated as

$$f_{smav}(s,w) = f_{mav}(s,w) - f_{mav}(s,w-1) \tag{5.4}$$

Simple square integral (SSI) The simple square integral is calculated as:

$$f_{ssi}(s,w) = \sum_{t=1}^{W_n} EMG(t)^2 \tag{5.5}$$

Variance The variance is calculated with an assumed mean of zero.

$$f_{var}(s,w) = \frac{1}{W_n}\sum_{t=1}^{W_n} EMG(t)^2 \tag{5.6}$$

Waveform change The waveform change measures how much the signal changes between samples.

$$f_{wc} = \sum_{t=1}^{W_n-1} |EMG(t+1) - EMG(t)| \tag{5.7}$$

Zero crossings The zero crossing feature measures how many times the EMG signal changes from positive to negative or negative to positive, with a small threshold to prevent fluctuations around zero from increasing the feature value.

Slope sign change The slope sign change measures the number of times the slope between two adjacent EMG signals changes from positive to negative or negative to positive.

Willson Amplitude The Wilson amplitude counts the number of times the EMG signal change is greater than a preset threshold.

Autoregressive coefficients The signal is modeled as an autoregressive model such that

$$EMG(t) = \sum_{\tau=1}^{m} \varphi_i EMG(t-\tau) + \varepsilon_t \tag{5.8}$$

where $EMG(t)$ is the EMG signal at time t and ε_t is a noise parameter.

The features are the model parameters Φ, which are found by solving the Yule-Walker equations for Φ

$$R\Phi = r \tag{5.9}$$

where R is the autocorrelation matrix of size $m \times m$ where m is the model order

$$\begin{aligned} R &= E \begin{bmatrix} u(t)^2 & u(t)u(t-1) & \cdots & u(t)u(t-m+1) \\ u(t-1)u(t) & u(t-1)^2 & \cdots & u(t-1)u(t-m+2) \\ \vdots & \vdots & \ddots & \vdots \\ u(t-m+1)u(t) & u(t-m+1)u(t-1) & \cdots & u(t-m+1)^2 \end{bmatrix} \\ &= \begin{bmatrix} r(0) & r(1) & \cdots & r(m-1) \\ r(1) & r(0) & \cdots & r(m-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(m-1) & r(m-2) & \cdots & r(0) \end{bmatrix} \end{aligned} \tag{5.10}$$

where $u(t)$ is the signal at time t (here, the EMG signal) and $r(t)$ is the autocorrelation for a time lag of t. $\mathbf{r}$ is an $m \times 1$ vector of autocorrelation values:

$$\mathbf{r} = E \begin{bmatrix} u(t)u(t-1) \\ u(t)u(t-2) \\ \vdots \\ u(t)u(t-m) \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(m) \end{bmatrix} \tag{5.11}$$

The model order used is 4, which returns 5 coefficients: the four Φ values and a constant offset.

Wavelet decomposition The wavelet decomposition of a signal is similar to short time Fourier analysis, but instead of using equally sized windows for each frequency component, wavelets match the size of the window to the frequency being analyzed. This is accomplished through successive convolution of the signal with scaled and shifted versions of a

mother wavelet [38]. For a discrete signal, this corresponds to dividing the signal by passing it through a high and a low pass filter, downsampling each side and successively applying high and low pass filters to the low frequencies. The coefficients from these recursively applied filters are used to generate the wavelet features.

The wavelet decomposition used in this chapter is a four level decomposition and a Daubechies wavelet, as recommended in [124]. The sixteen wavelet features are the minimum, maximum, mean and standard deviation of the coefficients at each level.

Ratio of range to standard deviation

$$f_{rst}(s,w) = \log\left(\frac{\max_{t\in w} EMG(t) - \min_{t\in w} EMG(t)}{\sqrt{\frac{1}{W_n}\sum_{t=1}^{W_n} EMG(t) - \mu}}\right) / \log W_n \tag{5.12}$$

Hjorth Parameters The three Hjorth parameters are the signal activity, mobility and complexity [64]. The activity is the variance of the signal, calculated as [64]:

$$f_{hj1}(s,w) = \frac{1}{W_n}\sum_{t=1}^{W_n}(EMG(t) - \mu)^2 \tag{5.13}$$

The mobility is the standard deviation of the slope divided by the standard deviation of the signal [64]. The slope or first derivative of the signal is found as

$$\Delta EMG(t) = EMG(t) - EMG(t-1) \tag{5.14}$$

This is then used to calculate the mobility as [64]:

$$f_{hj2}(s,w) = \frac{\sigma_d^2}{\sigma^2} \tag{5.15}$$

where σ_d^2 is the variance of the slope of the EMG signal ΔEMG and σ^2 is the variance of the EMG signal or f_{hj1}.

The complexity is the mobility of the first derivative of the signal and the mobility of the signal itself [64]:

$$f_{hj3}(s,w) = \frac{\sigma_{dd}^2/\sigma_d^2}{\sigma_d^2/\sigma^2} \tag{5.16}$$

where σ_{dd}^2 is the variance of the second derivative of the signal, or the slope of ΔEMG, calculated as in equation 5.14.

Mean The mean is calculated as:

$$f_{mean}(s,w) = \frac{1}{W_N}\sum_{t=1}^{W_n} EMG(t) \tag{5.17}$$

Skewness The skewness is a measure of how asymmetric a distribution is. The skewness is:

$$f_{sk}(s,w) = E\left[\left(\frac{EMG-\mu}{\sigma}\right)^3\right] \tag{5.18}$$

Kurtosis The kurtosis measures the"peakedness" of a distribution:

$$f_{kt}(s,w) = \frac{E[(EMG-\mu)^4]}{(E[(EMG-\mu)^4])^2} \tag{5.19}$$

Accumulated energy The accumulated energy is calculated as:

$$f_{ae}(s,w) = \frac{1}{W_n}\sum_{t=1}^{W_n} EMG(t)^2 \tag{5.20}$$

Average non-linear energy The non-linear energy of a signal at time i is:

$$NE(t) = EMG(t)^2 - EMG(t-1)EMG(t+1) \tag{5.21}$$

The normalized average non-linear energy is:

$$f_{nle}(s,w) = \frac{\frac{1}{W_n-2}\sum_{t=2}^{W_n-1} NE(t)}{\max_{t\in w} NE(t)} \tag{5.22}$$

Frequency There are a number of features related to the frequency components of the signals. The features f_{VLF}, f_{LF}, f_{HF} and f_{VHF} correspond to the power of the signal in 4 ranges, from 0 to the Nyquist frequency.

The mid and peak power frequency features describe the frequencies at which the signal achieves half the total power and at which the power peaks.

Envelope The signal envelope is found first by rectifying the signal and then applying a low pass filter. The low pass filter used is a second order Butterworth filter [60].

Polar co-ordinates The polar co-ordinates of the signal indicate the angular direction of each sensor at each time step. There are 9 polar co-ordinate features for each time window, corresponding to each of the sensor values and the radius. There is one redundant value, but this is included as an additional feature since feature selection is being performed. The polar co-ordinates are calculated using the RMS values. The radius of the sensor vector at time window w is

$$f_{pcr}(w) = \sqrt{\sum_{s=1}^{8} f_{rms}(s,w)^2} \tag{5.23}$$

The angles for each of the sensors s are calculated as:

$$f_{pcs}(w) = arccos\left(\frac{f_{rms}(s,w)}{f_{pcr}(w)}\right) \tag{5.24}$$

Cosine of the vector angle There are two versions of this feature, which are either used to compare the readings of two sensors over all the time values, or to compare two time windows across all the sensor readings.

The cosine of the vector angle features are based on the RMS feature, which is calculated for each of the eight sensors (S) in W time windows. This gives an $S \times W$ matrix of RMS values. For the feature that compares two sensors (f_{coss}), two row vectors are extracted from the matrix and the feature is computed as

$$f_{coss}(S_i, S_j) = \frac{R_{Si} \cdot R_{Sj}}{|R_{Si}||R_{Sj}|} \tag{5.25}$$

This generates 28 features from the pairwise combination of 8 sensor vectors.

RMS values are taken with a set window size (W_{sz}) of 20 and a set window shift (W_{sh}) of 5. No segmentation is performed. Because the window sizes are set to absolute values, the number of windows varies from trial to trial. There are approximately 90 windows per trial. This is not a problem, however, because the feature compares two sensors from within the same trial, which will always have the same number of features. Because the dot product is normalized by the magnitude of the two vectors, the values are comparable across trials.

#	Meta Feature
1	Number of Peaks
2-4	Peak width in windows, samples and sensors to 20% of range
5-7	Peak width in windows, samples and sensors to 30% of range
8-10	Peak width in windows, samples and sensors to 50% of range
11-13	Peak width in windows, samples and sensors to 70% of range
14-16	Peak width in windows, samples and sensors to 90% of range
17-18	Peak distance in from start in windows and samples
19	Peak angular distance from sensor 0
20	Normalized Peak value-average
21	non-normalized peak value
22	range
23	Normalized average
24	Normalized median

Table 5.17: Meta-features used for hand gesture recognition

This feature also does not require segmentation. Because the sensor values are very low in the rest period, they do not contribute greatly to the dot product value. The performance degradation from improper segmentation is more detrimental than the small contribution of the rest period values.

For feature f_{cosw} each gesture is pre-segmented. The window size and shift are set to generate 10 equally sized, overlapping windows. The RMS value is taken in each window, as given in equation 5.1. The feature is calculated column-wise on the resulting RMS matrix as:

$$f_{cosw}(W_i, W_j) = \frac{C_{Wi} \cdot C_{Wj}}{|C_{Wi}||C_{Wj}|} \tag{5.26}$$

This generates 45 features from the pairwise combination of 10 window vectors.

Meta-features

Table 5.17 describes the meta-features extracted from the original set of features described in Section 5.2.1. They are described in detail below.

Number of peaks This feature describes the total number of peaks or valleys found in the 2-D feature matrix

Position of maximum peak Three meta-features are used, describing the position of the maximum peak with respect to the gesture start time either in number of windows or in absolute number of samples. A third meta-feature describes the position of the peak as

an angular spatial distance from sensor 0. Angular distance is used because the sensors form a circle around the arm, so sensor 7 is spatially quite close to sensor 0.

Peak widths These features describe the 1-D width of the peak either across the sensors or time. The width of the peak is measured starting from the peak position and finding the first feature value where the feature value drops below x% of the feature value range, where x can take the value of 20%, 30%, 50%, 70% or 90%. As with the position values, this is described in terms of the number of time windows, the number of time samples an the angular sensor distance.

Normalized Peak - average This feature describes how much higher the peak value is than the average. It is normalized by dividing by the range.

Non-normalized peak value This feature is simply the feature value at the peak.

Range This feature describes the feature value range.

Normalized Average or Median This feature is the average or median of the feature matrix divided by the range.

5.2.2 Results and Discussion

The first set of experiments is performed on data set *EMG 1* using both a single level MBT and a two-level MBT.

Table 5.18 shows the confusion matrix of the single-stage MBT on *EMG 1* with 8 gestures. Table 5.19 shows the precision and recall for each individual person for each gesture. The overall accuracy is only 35%, which is higher than chance, but not sufficient for use in a real-world situation. Additionally, the precision and recall vary greatly from person to person.

The top level of the two-stage tree is shown in Figure 5.5. It is tested to the fourth level (rest vs. right vs. snap vs. finger tap vs. other). The confusion matrix is shown in Table 5.20. Table 5.21 show the precision and recall for each individual person for each gesture. The accuracy is 71.0%. The comparable accuracy from the single stage tree is 59.3%, indicating that the two-stage tree may be beneficial.

		classifier result							
		rest	fist	left	right	snap	rock on	gun	finger tap
true class	rest	72	0	1	6	3	2	2	4
	fist	2	14	12	24	7	11	9	11
	left	1	10	25	5	3	19	19	8
	right	8	14	6	46	3	4	4	5
	snap	3	9	4	5	45	7	6	11
	rock on	3	14	15	1	6	13	14	24
	gun	1	17	17	4	7	12	22	10
	finger tap	2	21	11	8	7	10	10	21

Table 5.18: Confusion matrix for *EMG 1* using a single stage MBT

	Precision gesture							
person	rest	fist	left	right	snap	rock on	gun	finger tap
1	0.78	0.00	0.17	0.40	0.29	0.17	0.50	0.18
2	1.00	0.17	0.33	0.62	0.75	0.11	0.27	0.32
3	0.83	0.20	0.08	0.25	0.67	0.00	0.17	0.00
4	0.83	0.00	0.20	1.00	0.80	0.00	0.24	0.25
5	1.00	0.07	0.38	0.47	0.71	0.43	0.00	0.25
6	1.00	0.23	0.36	0.47	0.50	0.00	0.14	0.67
7	0.88	0.29	0.25	0.64	1.00	0.50	0.36	0.40
8	0.37	0.19	0.43	0.50	0.36	0.40	0.20	0.10
9	0.88	0.09	0.40	0.18	0.38	0.14	0.00	0.00
	Recall gesture							
person	rest	fist	left	right	snap	rock on	gun	finger tap
1	0.70	0.00	0.10	0.20	0.20	0.20	0.60	0.40
2	0.90	0.10	0.30	0.80	0.30	0.10	0.30	0.60
3	1.00	0.30	0.10	0.30	0.20	0.00	0.10	0.00
4	1.00	0.00	0.30	0.60	0.40	0.00	0.40	0.20
5	1.00	0.10	0.30	0.70	1.00	0.30	0.00	0.20
6	0.50	0.30	0.40	0.70	0.70	0.00	0.10	0.20
7	0.70	0.20	0.30	0.90	0.70	0.40	0.50	0.40
8	0.70	0.30	0.30	0.10	0.40	0.20	0.20	0.10
9	0.70	0.10	0.40	0.30	0.60	0.10	0.00	0.00

Table 5.19: Precision and recall for each individual person on *EMG 1* using a single stage MBT

		classifier result				
		rest	right	snap	finger tap	other
true class	rest	79	2	2	3	4
	right	1	52	8	8	21
	snap	0	4	65	1	20
	finger tap	5	7	5	20	53
	other	2	16	21	26	295

Table 5.20: Confusion matrix for *EMG 1* using a two-stage MBT

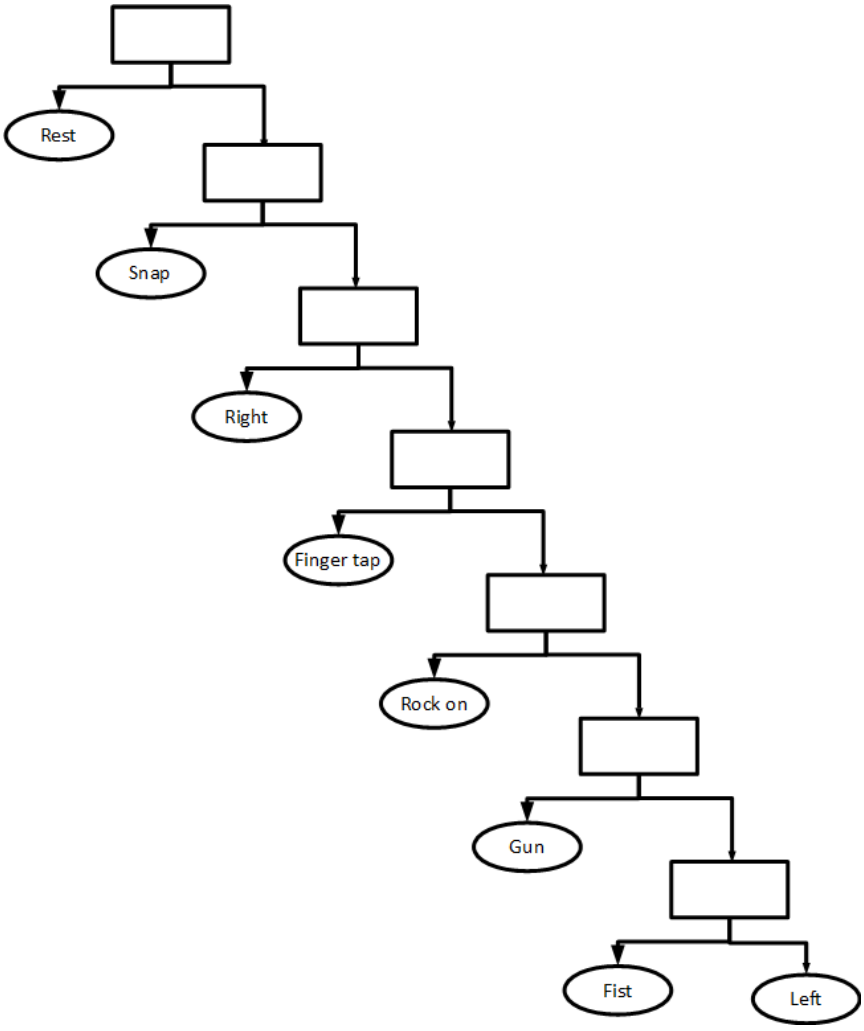


Figure 5.5: Top level of the two-stage tree designed using *EMG 1*

	Precision gesture				Recall gesture			
person	rest	snap	right	finger tap	rest	snap	right	finger tap
1	1.00	0.89	0.57	0.39	0.70	0.80	0.40	0.70
2	1.00	0.62	1.00	0.63	1.00	0.80	0.50	0.50
3	1.00	1.00	0.33	0.00	1.00	0.20	0.10	0.00
4	1.00	1.00	0.91	0.25	1.00	1.00	0.90	0.10
5	1.00	0.89	0.42	0.40	1.00	0.80	0.50	0.20
6	0.70	0.43	0.82	1.00	0.70	0.90	0.90	0.10
7	0.73	0.89	1.00	0.44	0.80	0.80	0.80	0.40
8	0.90	0.88	0.71	0.00	0.90	0.50	0.70	0.00
9	0.89	0.39	0.15	0.00	0.80	0.90	0.20	0.00

Table 5.21: Precision and recall for each individual person on *EMG 1* using a two-stage MBT

	5 windows		40 windows	
	5 features/ node	10 features/ node	5 features/ node	10 features/ node
LOSO accuracy	0.77	0.78	0.77	0.75
10 folds accuracy	0.82	0.79	0.80	0.79

Table 5.22: Accuracy of two-stage MBT on *EMG 2*

The first set of tests on *EMG 2* include only four of the gestures. The gestures are fist, paddle in, paddle out and thumb/ring finger tap. All were tested using a two-stage MBT tree, shown in Figure 5.6. Tests were run using either 5 or 40 overlapping windows and using either 5 or 10 features at each node. Table 5.22 shows the overall accuracy for the various test cases.

Table 5.23 shows the five most commonly selected features for the different classifier levels. It is clear from these tables that the wavelet features are quite strong. The Hjorth parameters are also common in many of the trees, but are selected less frequently than the wavelet features.

The most commonly selected features are almost all calculated over the entire signal. There are two reasons for this. Firstly, the feature selection measure used is information based. Features that are calculated over a larger window can theoretically have a larger amount of information and therefore should be selected. For this reason, it may be unfair to compare features with different window lengths using this type of feature selection. Selecting the proper window length may be a separate problem from feature selection.

Second, it appears that many of the windowed features are dependent. Chapter 4 Section 4.4 proposes a new feature selection measure that first identifies dependent features using the pairwise conditional mutual information $I(x_i; Y|x_j)$ and the two feature-class mutual information values $I(x_i; Y)$. The dependence criterion is outlined in equation 4.2.

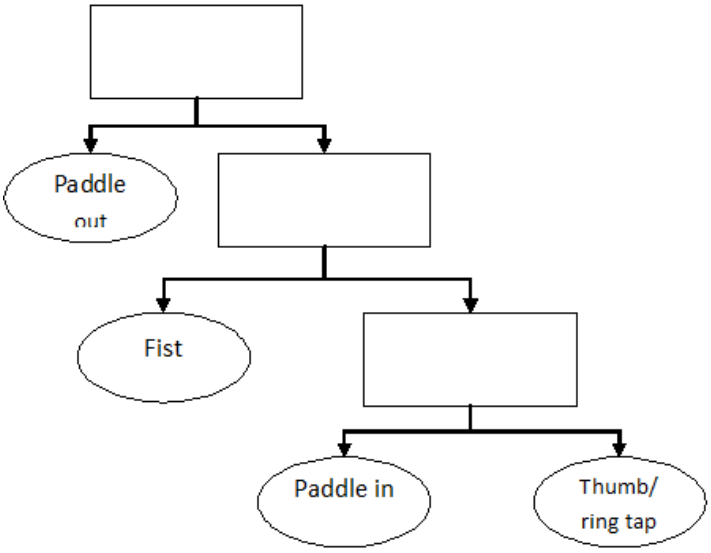


Figure 5.6: Top level of the two-stage tree designed using *EMG 2* and four class problem

	feature	meta-feature
level 1	Wavelet level 1 min	whole signal sensor 2
	Wavelet level 1 std	whole signal sensor 2
	RMS	Median / range
	Wavelet level 1 max	whole signal sensor 2
	Willson amplitude	whole signal sensor 6
level 2	Willson amplitude	whole signal sensor 4
	Wavelet level 1 max	whole signal sensor 4
	Waveform change	whole signal sensor 4
	Wavelet level 4 std	whole signal sensor 4
	Mean abs value	whole signal sensor 4
level 3	Wavelet level 3 min	whole signal sensor 3
	Wavelet level 4 std	whole signal sensor 3
	Wavelet level 4 max	whole signal sensor 1
	Wavelet level 3 std	whole signal sensor 3
	Wavelet level 1 min	whole signal sensor 3

Table 5.23: Commonly selected features for two-stage MBT on *EMG 2*

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	0.83	0.80	0.72	0.68	0.70	0.77
10	0.88	0.84	0.73	0.69	0.71	0.79
15	0.86	0.83	0.76	0.74	0.75	0.81
20	0.88	0.85	0.76	0.71	0.75	0.79
25	0.86	0.84	0.74	0.75	–	0.82

Table 5.24: Accuracy of single stage MBT classifier using cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap.

To test this hypothesis, a smaller data set consisting of the RMS meta-features and the polar features was tested for dependence. There are a total of 118 features and 13924 pairwise comparisons. Using the definition in 4.2, over 5000 out of the 13924 feature pairs were identified as dependent. This is a large number of dependent features that will not be selected by CMIM and which the SVM may be unable to use.

Unfortunately, training the classifier using the extracted dependent features does not improve accuracy. An MBT generated using the four-gesture data set and no dependent features yields an accuracy of 56.2%. The accuracy with dependent features is only 53.4%. This implies that SVM may be unable to properly use all the dependent features, as seen in Chapter 4 Section 4.4. Additionally, a linear projection of the dependent features using LDA may be insufficient to capture the information in the two dependent features. A non-linear projection may be necessary.

A final test on this data compares a single stage MBT to a two-stage MBT to a one vs. one formulation that uses MBT as the base classifier. The set of candidate features includes both sets of cosine features (f_{coss} and f_{cosw}) as well as the simple squared integral, wavelet and Hjorth parameter features calculated over the entire feature window. These are included because they are identified as potentially good candidate features in the first set of tests on the newer data set. These three configurations are tested with a variety of different gesture sets.

Tables 5.24, 5.25 and 5.26 show the leave-one-subject-out accuracy for the single MBT, the two-stage MBT and the one vs. one MBT selecting various numbers of features at each node. Tables 5.27, 5.28 and 5.29 show the average number of features required to classify a new point.

It is clear from these tables that the new feature set gives a much better performance than the original set. The one vs. one formulation appears to give the best accuracy at the expense of a longer test time. The single stage MBT gives a good balance of accuracy

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	0.83	0.82	0.76	0.71	0.73	0.79
10	0.86	0.85	0.73	0.70	–	0.81
15	0.86	0.83	0.75	–	0.75	0.83
20	0.87	0.85	0.77	–	0.76	0.83
25	0.85	0.86	0.75	0.73	0.76	0.83

Table 5.25: Accuracy of two-stage MBT classifier using cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap.

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	0.85	0.82	0.76	0.73	0.76	0.81
10	0.87	0.86	0.78	0.76	0.77	0.82
15	0.88	0.84	0.77	0.77	0.77	0.85
20	0.88	0.85	0.80	0.78	0.78	0.85
25	0.89	0.87	0.80	0.78	0.79	0.86

Table 5.26: Accuracy of MBT one vs. one classifier using cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	32	33	42	46	44	38
10	53	57	78	82	82	67
15	69	76	98	116	105	88
20	87	93	115	138	130	109
25	98	110	146	164	–	132

Table 5.27: Average number of features used to classify a new point using a single stage MBT classifier with cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap.

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	40	38	60	73	67	53
10	66	71	107	131	–	92
15	86	89	147	–	157	125
20	115	107	168	–	203	140
25	126	121	186	237	233	167

Table 5.28: Average number of features used to classify a new point using a two-stage MBT classifier with cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap.

	classes					
	5	4	6	7	7	6
# feats	R,F,PI,PO,Sn	F,PI,PO,TR	R,F,PI,PO,Sp,TR	R,F,PI,PO,Sn,Sp,TR	R,F,PI,PO,Sn,Sp,TP	R,F,PI,PO,Sn,TP
5	107	76	176	265	281	182
10	182	131	318	468	471	296
15	237	165	419	600	599	377
20	284	201	511	715	717	454
25	317	217	584	827	910	515

Table 5.29: Average number of features used to classify a new point using a MBT one vs. one classifier with cosine and full window features. Gestures are as follows: R = Rest, F = Fist, PI=Paddle In, PO = Paddle Out, Sn = Snap, Sp = Spiderman, TR = Thumb/Ring finger tap, TP = Thumb/Pinky tap.

and number of features. The two-stage MBT has a lower accuracy and longer test time than the single MBT and is therefore not a good choice.

The performance of all three methods is quite good compared to earlier performance and compared to other problems seen in literature. For example, [110] quote a 73.0% accuracy on a five gesture classification problem using EMG signals. Although these are not directly comparable as the hardware and EMG placement is different, the performance of the one vs. one MBT appears to be quite good.

5.3 Summary

This chapter examined the use of feature selection for time-series data classification tasks. Time-series feature selection and the MBT algorithm were then tested on two time-series human motion recognition problems.

Experiments with the human motion data set demonstrate the benefit of using feature selection to select individual time windows for time-series classification. Classifiers trained using only a small selected subset of windows are able to achieve the same accuracy as classifiers using the entire time-series of a feature. This type of feature selection reduces the number of inputs from 20 time-series to only two or three features with no loss in accuracy. The reduced number of inputs directly reduces the memory footprint, the classification complexity and the time required to classify new points.

In a second set of experiments, time-series feature selection and the MBT classifier were tested for use with two time-series human motion data sets: full body human motion recognition from joint angles and hand gesture recognition from EMG. On the full body motion data set, the feature-selected MBT was able to achieve 100% accuracy on both a three motion and a nine motion problem. The MBT classifier requires only eight features to classify the three motion data set and only 16 features to classify the nine motion data set. On the EMG hand gesture recognition data set the single level MBT achieves 88% accuracy on a five gesture recognition problem.

Chapter 6

Conclusions

This thesis analyzes feature selection and class division for classification problems. The first part of this thesis examines feature selection and class division for the creation of feature-selected tree-based classifiers. The second part evaluates the performance of different filter-based feature set evaluation measures with respect to specific classifiers. The last part of the thesis examines feature selection for time-series data and applies the selected filter measure and tree-based classifier to two time-series human motion recognition problems.

The first part of this thesis explores the joint contribution of class division and feature selection to classifier memory requirements, test time and accuracy in tree-based classifiers. Two algorithms are developed to create feature selected tree-based classifiers. The first algorithm, feature selected hierarchical classification (FSHC), jointly optimizes the tree structure and features selected at each node of the tree using a genetic algorithm. This algorithm shows good results with artificial data sets containing partially informative features. On real data sets the performance is comparable to other multi-class extensions, but with a shorter test time.

A second feature-selected tree-based classification algorithm, multi-modal binary tree (MBT), is proposed as a way to overcome some of the challenges with the FSHC algorithm. Because FSHC uses a genetic algorithm, it has some issues with robustness and generalization to the test set. MBT uses a non-stochastic sequential method for tree design and adds the ability to have classes appear at more than one node output by reclassifying initially misclassified samples. This allows MBT to separate multi-modal and non-linearly separable data. MBT performs well on both real and artificial data sets and the accuracy of the MBT is significantly higher on known multi-modal data sets. The test time of the MBT is short compared to other multi-class extensions, which is important for applications that

classify new, incoming samples in an online manner and require low delay. MBT also has a relatively low computational complexity, which is beneficial for applications that require low power.

The MBT algorithm can also be implemented as a generic framework, and can easily be used with different classifiers and different feature selection methods. The MBT can also be used with multi-class classifiers by using a different K value in the clustering algorithm.

These experiments also show the benefits of feature selection both for reducing classifier memory requirements and test time, and improving accuracy. However, the performance of the different feature set evaluation measures is varied and this indicates the importance of selecting the feature set evaluation measure carefully. The second part of the thesis presents an empirical evaluation of filter-based feature set evaluation measures. Based on these findings, a new measure of feature dependence is proposed and evaluated.

Sixteen different univariate and multi-variate filter measures are evaluated with respect to their ability to identify known informative features and their correlation with classifier accuracy. Based on the results of these tests, CFS, CMIM and subset-RELIEF are identifieda as good choices for KNN, and Fisher's interclass separability criterion or CMIM are recommended for SVM.

Despite the relatively good performance of the CMIM measure, it is unable to identify dependent, informative features. Two new measures of feature dependence are proposed to overcome these issues. The measures are based on CMIM and FOU/CFS, but include an explicit test for feature dependence. These measures are capable of finding informative dependent features that the other filter measures cannot detect.

The final part of the thesis examines the application of feature selection and the MBT algorithm to time-series human-motion recognition data. Two different human motion classification problems are considered: recognition of full-body human motions from joint angles, and the recognition of hand gestures from EMG data.

The first experiments with the full-body human motion data set use binary classification problems to assess whether feature selection can be used to select individual windows from within a time-series input instead of including the entire time series. Results show that classifiers using a subset of the windowed inputs can achieve the same accuracy as the full time-series, with a shorter test time and smaller memory footprint. The MBT algorithm is also applied to the full eight class full-body motion recognition problem. The feature selected MBT algorithm achieves 100% accuracy on this data set using only two to three classifiers and ten to fifteen features per classification. This is a significant reduction in classification time over a standard full -feature multi-class extension with no loss of accuracy.

The MBT algorithm is also applied to an EMG-based gesture recognition set. Several variations of the MBT algorithm are tested, including a single-level tree, a two-level tree and a one vs. one formulation with MBT used as the base classifier. The one vs. one MBT algorithm is able to achieve 85% accuracy on a six gesture set and 88% accuracy on a five gesture set using leave-one-subject-out testing. These test results show the potential of the MBT algorithm for such difficult problems.

This thesis makes the following contributions:

- The development of the multi-modal binary tree algorithm that can be used to classify multi-modal and non-linear classes using piecewise separation
- An extensive empirical evaluation of current filter-based feature set evaluation measures with respect to specific classifiers
- The development of an information-theoretic based test for feature dependence and the development of a new filter-based feature selection measure that can account for dependent features
- The application of feature selection and the MBT algorithm to two different time-series human motion recognition problems

6.1 Future Work

6.1.1 Tree-based classifiers

The initial set of experiments with FSHC compares FSHC to both one vs. rest with no tie breaking, and fuzzy one vs. rest, which uses the distance to the hyperplane as a way to classify points that are claimed by multiple classifiers or that are not claimed. Interestingly, the performance of the fuzzy one vs. rest is much better than the one vs. rest formulation with no ties. This indicates that there are a number of points that are misclassified by one or more classifiers in the one vs. rest formulation. The fuzzy one vs. rest can recover a correct classification even if one or more of the base classifiers is incorrect. This can also occur in some cases with a one vs. one formulation, but appears to be less common based on the similar performance of the one. vs. one and the DAG-SVM.

Although the MBT algorithm allows points to be classified into either output of a node, a misclassification of a test point at any node will result in a misclassification of the point.

It may be possible to incorporate a fuzzy component into the tree, with each classifier returning a fuzzy measure of class membership for each output. For example, for an SVM, this may be distance to the hyperplane, as described in [70]. Such fuzzy membership functions can also be defined for other classifiers, such as ANN with a linear output layer [166].

Following each branch in the tree would be quite expensive, but the fuzzy membership functions can be defined in such a way that a point a certain distance from the hyperplane or linear output value is considered to be a member of only a single class. In this case, only a single branch would need to be followed from that node. A fuzzy tree would still be more computationally expensive than a standard tree, but could be used to give a measure of confidence in the output.

6.1.2 Dependent features

The dCMIM and dFOU measures proposed in Chapter 4 identify dependent features in the set. However, not every classifier is capable of using dependent features, or may only be able to use certain types of dependent features. Even if a classifier is not able to use the identified dependent features to improve classification accuracy directly, identifying dependent features is still important. Dependent features carry information that may not be directly usable by the classifier. Feature extraction techniques may be able to generate new features from the dependent features.

This is a slightly different way of using feature extraction and feature selection together. While feature extraction techniques can be used to generate a larger set of candidate features for feature selection, for dependent features, the dependent feature selection measure would be used to generate a candidate set of features that would benefit from feature extraction techniques. The best feature extraction measure to use for such features remains an open question.

6.1.3 EMG hand gesture classification

There are several possible improvements to the EMG-based hand gesture classifier to improve accuracy, computational complexity and test time and include new data points through online training.

Options for improving accuracy

There are two options for improving the accuracy of the MBT one vs. one classifier. The first option is to identify the pairwise classifiers with low accuracy on the two-class problem and use either a more complex kernel, a larger number of features or specifically constructed features to achieve better accuracy. Using a more computationally expensive classifier on a small subset of the data may allow the accuracy to be improved, without unnecessarily increasing the classifier complexity on classes that are already well separated.

Another option is to change the selected outputs. The EMG hand gesture recognition data set includes a large number of gestures, but not all of the gestures in the set may be required for a particular application. It may be possible to optimize the set of gestures to improve the accuracy of the classifier.

One other option, which is slightly less convenient for the user, is to ask for a single sample of each gesture from the current user and then re-train the classifier using a over-weighted version of this sample. There are, however, two major issues with this solution. Firstly, the sample provided by the user may not be representative of all their gestures. It is possible for a single use to use multiple strategies for a single gesture. Secondly, this requires re-training the classifier at run-time, which is inconvenient for the user.

Options for reducing computational complexity

Because the base classifiers are linear SVMs, classification at each node is performed by finding the distance to the hyperplane. This means that the complexity is directly proportional to the number of features at each base classifier. The number of multiplications can be reduced by reducing the number of features at each node, by reducing the number of nodes in the tree, or by reducing the number of trees queried.

The easiest method for reducing the number of required multiplications is to reduce the number of pairwise classifiers that need to be queried. The numbers quoted in Table 5.29 come from a classification method that uses a voting scheme. This scheme trains a set of pairwise classifier trees and each is queried and casts a vote for its winning class. An alternate method is DAG-SVM [126]. In DAG-SVM, each queried classifier eliminates one class. Using DAG-SVM would reduce the number of trees queried from C(C-1)/2 to C-1. For a 4 class problem, this reduces the number of trees queried from 6 to 3, and for a 5 class problem this reduces the number of trees queried from 10 to 4. For the 7 class problem, the number of trees is reduced from 21 to 6. This is a significant reduction that requires little effort and almost no reduction in accuracy, as found in [65] and Chapter 3.

The second method is to reduce the average number of features used for each node. In many cases, increasing the number of features improves the accuracy. However, this improvement is not consistent across all pairwise classifiers. For some pairs of classes, 5 features is sufficient. Other pairwise classifiers require more features. Hence, one simple method to reduce the average number of features is to set the number of features differently for each pairwise classifier, using a validation set. This allows the use of smaller feature sets for the more easily separated classes, while maintaining the accuracy gain from using a larger number of features on the more difficult pairs.

The last option is to reduce the number of nodes of the tree. This can be accomplished by adjusting the misclassification error rate r_{mc}, as described in Chapter 3. As with the number of features, this is a parameter that can be set using a validation set.

Options for decreasing test time

The ideal classifier would be able to determine the gesture being performed before the gesture is completed. The current system uses features generated over the entire length of the gesture.

A simple method would be to use a segmentation program to find the start of the action and then extract the desired feature over a set window size that is shorter than the length of the average gesture. This would require re-training the classifier on these new features. This may also adversely affect the accuracy.

Because some gestures are much longer than others, using a standard window length may adversely affect the accuracy and/or test time. Instead, it may be possible to use an adjustable gesture length by setting the window length based on the signal peak.

Options for incorporating large amounts of incoming data

Thalmic labs is also pursuing the option of collecting data at run-time from the users. Incorporating this data back into the classifier can be challenging as it requires continually re-training classifiers on a progressively larger amount of data. In tests of the commonly used sequential minimal optimization training method, Platt [126] finds empirically that the training time scales as N^{γ} where $\gamma \approx 2$. Hence, as the number of samples gets large, the training time becomes impractically large.

As the collection of this data is ongoing, the training should be performed online, and should begin with a trained classifier. The new points then need to be incorporated to adjust the SVM boudaries, or to add new nodes to the tree.

For the linear tree SVM, mis-classified points are the most interesting because they indicate that the current hyperplane is incorrect, or that additional nodes need to be added. Hence, the amount of data in the training set can be reduced by selecting only points that are misclassified. Identification of misclassified points may be the next largest challenge.

Similarly, points that are classified correctly but are very close to the hyperplane may also alter the classifier training. However, identifying these points would require finding a distance to the hyperplane rather than just the classification result.

These new misclassified points can be added to the tree online by adding new nodes for misclassified samples. Each sample is classified by the existing tree. If the classification is correct, no further nodes need to be added. If the classification is incorect, a new node is added to properly separate the new samples, as is done in the original tree. Eventually, this will increase the number of nodes and layers of the tree. If the misclassified points are maintained in a list, the tree can be re-trained periodically using these new samples in an effort to reduce the number of nodes and layers by adjusting the nodes higher in the tree to accommodate the misclassified points.

6.1.4 Application to other domains

One of the benefits of the MBT is its short test time and small computational expense. This is particularly important for domains where the objective is to perform classification of incoming data samples in an online manner, even if the initial training is offline. The MBT may be a good fit for other human motion recognition problems or problems in other domains. For example, MBT could be applied to motion recognition problems in the area of rehabilitation. One interesting application examines the performance of rehabilitation exercises to distinguish between correctly and incorrectly performed motions. This information can be used to provide feedback to the patient in real-time. Time-series feature selection could be beneficial in such a task, allowing the classifier to focus on portion of motion that may be incorrectly performed. The relatively small computational expense of the MBT would allow the classifications to be performed with low delay, giving feedback to the patient in a timely manner.

The MBT could also be beneficial to domains outside of human motion recognition. One example is in the area of audio-environment recognition for hearing aid background noise reduction. Because of the extremely small size and low power of hearing aids, any algorithm used for classification must be extremely computationally efficient. MBT is well

suited for such a task and may be beneficial to this domain or other domains that require efficient, low-delay classification of incoming samples.

APPENDICES

Appendix A

Classifier accuracy on tested data sets

This appendix presents the different accuracy values of the various classifiers on the artificial and real data sets.

	1-NN					
	Best		Worst		All Features	
	acc.	FS size	acc.	FS size	acc.	FS size
Binary 1	0.99	3	0.80	1	0.99	3
Binary 2	1.00	3	0.55	1	1.00	3
Binary 3	1.00	3	0.59	1	1.00	3
Monk 1	1.00	4	0.46	5	0.81	6
Monk 2	0.84	6	0.47	5	0.84	6
Monk 3	0.99	4	0.51	3	0.93	6
Simple dependent	0.92	2	0.62	1	0.92	2
Partially dependent	0.97	2	0.60	1	0.97	2
Two way linear correlated	0.99	2	0.98	1	0.99	2
Three way linear correlated	1.00	1	0.69	1	0.97	3
Three was non-linear correlated	1.00	2	0.72	1	1.00	3
Noise with one good feature	1.00	2	0.14	1	0.93	4
Pure noise	0.29	2	0.18	1	0.28	3
Un-nested	1.00	2	0.59	1	1.00	3
Monotonic	1.00	4	0.84	1	1.00	4
Two U's	1.00	2	0.75	1	1.00	2
Multimodal (XOR)	1.00	2	0.51	1	1.00	2
Single cluster linearly separable	0.95	2	0.66	1	0.95	2
Redundant features	1.00	3	1.00	3	1.00	4
One cluster per class	1.00	2	1.00	2	1.00	2

Table A.1: Summary of 1-NN accuracy on different artificial data sets

	1-NN					
	Best		Worst		All Features	
	acc.	FS size	acc.	FS size	acc.	FS size
Abalone	0.21	3	0.12	1	0.20	8
Car Evaluation	0.93	6	0.58	5	0.93	6
Cardiotocography	0.80	12	0.12	2	0.76	22
Congressional Voting	0.95	9	0.69	3	0.93	16
Contraceptive Choice	0.48	6	0.37	5	0.44	9
Credit Approval	0.84	9	0.45	2	0.82	15
Dermatology	0.97	24	0.24	1	0.96	34
E-coli	0.82	6	0.43	1	0.81	7
Flag	0.56	7	0.20	4	0.44	28
Glass	0.77	4	0.28	2	0.69	9
Haberman's Survival	0.73	1	0.36	1	0.66	3
Ionosphere	0.91	15	0.74	1	0.86	34
Iris	0.96	2	0.49	1	0.94	4
Mushroom	1.00	18	0.43	2	1.00	22
Page blocks	0.96	8	0.79	1	0.96	10
Post-op	0.70	5	0.41	3	0.62	8
Segmentation	0.97	10	0.14	1	0.96	18
Statlog (vehicle)	0.73	10	0.34	1	0.69	18
Wisconsin Breast Cancer (original)	0.96	7	0.73	1	0.95	9
Wisconsin Breast Cancer (diagnostic)	0.96	13	0.65	1	0.95	30
Wine	0.97	9	0.37	1	0.95	13
Wisconsin Breast Cancer (prognostic)	0.72	9	0.58	6	0.67	32
Yeast	0.53	7	0.16	1	0.53	8

Table A.2: Summary of 1-NN accuracy on different real data sets

	SVM (OVO)						SVM (OVA)					
	Best		Worst		All Features		Best		Worst		All Features	
	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size
Binary 1	0.99	3	0.83	2	0.99	3	0.99	3	0.84	2	0.99	3
Binary 2	0.98	3	0.70	2	0.98	3	0.98	3	0.64	2	0.98	3
Binary 3	0.75	1	0.75	1	0.75	3	0.75	1	0.75	1	0.75	3
Monk 1	0.68	4	0.45	2	0.67	6	0.68	3	0.45	2	0.67	6
Monk 2	0.66	5	0.66	5	0.66	6	0.66	5	0.66	5	0.66	6
Monk 3	0.80	5	0.47	3	0.80	6	0.80	4	0.48	3	0.80	6
Simple dependent	0.93	2	0.70	1	0.93	2	0.95	2	0.71	1	0.95	2
Partially dependent	0.98	2	0.57	1	0.98	2	0.98	2	0.57	1	0.98	2
Two way linear correlated	0.99	1	0.98	1	0.99	2	0.99	2	0.99	1	0.99	2
Three way linear correlated	0.99	1	0.72	1	0.98	3	0.98	3	0.73	1	0.98	3
Three was non-linear correlated	1.00	2	0.70	1	1.00	3	1.00	2	0.71	1	1.00	3
Noise with one good feature	1.00	2	0.21	3	1.00	4	0.83	1	0.21	2	0.70	4
Pure noise	0.36	2	0.19	1	0.35	3	0.29	1	0.20	2	0.24	3
Un-nested	1.00	2	0.60	1	1.00	3	0.87	2	0.37	1	0.83	3
Monotonic	1.00	4	0.84	1	1.00	4	0.66	3	0.54	1	0.66	4
Two U's	0.97	2	0.81	1	0.97	2	0.97	2	0.83	1	0.97	2
Multimodal (XOR)	0.63	2	0.41	1	0.63	2	0.59	2	0.42	1	0.59	2
Single cluster linearly separable	0.96	2	0.73	1	0.96	2	0.97	2	0.73	1	0.97	2
Redundant features	1.00	3	1.00	3	1.00	4	0.89	1	0.61	2	0.63	4
One cluster per class	1.00	2	1.00	2	1.00	2	1.00	2	1.00	2	1.00	2

Table A.3: Summary of SVM accuracy on different artificial data sets

	SVM (OVO)						SVM (OVA)					
	Best		Worst		All Features		Best		Worst		All Features	
	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size	acc.	FS size
Abalone	0.27	4	0.18	1	0.26	8	0.14	2	0.03	3	0.08	8
Car Evaluation	0.85	6	0.70	4	0.85	6	0.81	6	0.66	1	0.81	6
Cardiotocography	0.84	12	0.27	2	0.84	22	0.79	17	0.11	2	0.78	22
Congressional Voting	0.96	12	0.67	3	0.96	16	0.97	12	0.67	3	0.96	16
Contraceptive Choice	0.53	4	0.41	4	0.51	9	0.49	7	0.33	2	0.47	9
Credit Approval	0.86	6	0.54	4	0.86	15	0.86	6	0.54	4	0.86	15
Dermatology	0.99	22	0.31	1	0.96	34	0.98	22	0.20	1	0.96	34
E-coli	0.86	6	0.43	1	0.86	7	0.86	6	0.18	2	0.85	7
Flag	0.55	11	0.24	2	0.45	28	0.51	15	0.12	6	0.47	28
Glass	0.66	6	0.30	2	0.63	9	0.61	6	0.15	1	0.58	9
Haberman's Survival	0.74	2	0.72	1	0.73	3	0.74	2	0.72	2	0.73	3
Ionosphere	0.90	24	0.64	1	0.87	34	0.90	25	0.64	1	0.88	34
Iris	0.97	3	0.53	1	0.96	4	0.93	3	0.55	1	0.91	4
Mushroom	0.99	19	0.54	6	0.99	22	0.99	19	0.54	6	0.99	22
Page blocks	0.96	10	0.90	1	0.96	10	0.96	9	0.89	1	0.96	10
Post-op	0.72	6	0.69	8	0.69	8	0.71	1	0.69	3	0.69	8
Segmentation	0.96	12	0.14	1	0.95	18	0.92	15	0.14	4	0.92	18
Statlog (vehicle)	0.80	15	0.31	2	0.79	18	0.78	18	0.24	2	0.78	18
Wisconsin Breast Cancer (original)	0.97	6	0.77	1	0.96	9	0.97	5	0.77	1	0.96	9
Wisconsin Breast Cancer (diagnostic)	0.98	19	0.64	1	0.97	30	0.98	18	0.66	1	0.97	30
Wine	0.98	10	0.50	1	0.97	13	0.99	10	0.42	1	0.98	13
Wisconsin Breast Cancer (prognostic)	0.76	16	0.71	28	0.74	32	0.76	3	0.71	22	0.73	32
Yeast	0.58	8	0.31	1	0.58	8	0.51	8	0.05	2	0.51	8

Table A.4: Summary of SVM accuracy on different real data sets

Appendix B

Filter measure correlation values

This appendix presents the correlation values between different filter measures and classifier accuracy for different parameter settings.

B.1 RELIEF-based measures

	RELIEF-F (summed)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10
Binary 1	0.58	0.57	0.57	0.57	0.54	0.54	0.54	0.54	0.93	0.93	0.92	0.92
Binary 2	0.51	0.65	0.69	0.70	0.49	0.56	0.56	0.56	0.56	0.81	0.91	0.92
Binary 3	100%	100%	100%	100%	100%	100%	100%	100%	0.89	0.90	0.89	0.86
Monk 1	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.58	0.58	0.58	0.58
Monk 2	-	-	-	-	-	-	-	-	0.00	0.00	0.00	0.00
Monk 3	-0.59	0.59	0.59	0.59	-0.56	0.56	0.56	0.56	-0.64	0.64	0.64	0.64
Simple dependent	0.93	0.99	0.97	1.00	0.92	0.98	0.97	0.99	0.93	0.99	0.97	1.00
Partially dependent	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.98	0.99
2-way linear corr.	0.67	0.69	0.70	0.70	1.00	1.00	1.00	1.00	0.98	0.99	0.99	0.99
3-way linear corr.	0.76	0.78	0.79	0.80	0.77	0.79	0.80	0.81	0.75	0.77	0.77	0.78
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	1.00	0.99	0.98	0.94	0.97	0.97	0.96	0.92	0.99	0.99	0.98	0.94
Pure noise	-0.49	-0.58	-0.54	-0.50	0.31	0.42	0.38	0.32	-0.51	-0.62	-0.58	-0.52
Un-nested	0.75	0.76	0.76	0.78	0.84	0.84	0.85	0.85	0.78	0.79	0.80	0.81
Monotonic	0.81	0.81	0.81	0.81	0.78	0.79	0.80	0.80	0.85	0.85	0.85	0.84
Two U's	0.76	0.84	0.89	0.94	0.70	0.79	0.84	0.90	0.85	0.91	0.95	0.98
Multimodal (XOR)	0.22	0.21	-0.41	-0.98	0.05	0.04	-0.56	-0.93	0.07	0.06	-0.54	-0.93
linear sep. Guassian	1.00	0.99	0.98	0.98	1.00	1.00	0.99	0.98	1.00	1.00	1.00	1.00
Redundant	100%	100%	100%	100%	-0.65	-0.65	-0.65	-0.65	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table B.1: Correlation between SVM and 1-NN accuracies and RELIEF-F filter values using different values for K on artificial data sets

	RELIEF-subset											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10
Binary 1	0.54	0.54	0.54	0.47	0.50	0.51	0.51	0.44	0.92	0.92	0.91	0.88
Binary 2	0.71	0.70	0.70	0.68	0.55	0.54	0.54	0.51	0.98	0.98	0.98	0.97
Binary 3	2521%	1482%	1029%	542%	2521%	1482%	1029%	542%	0.90	0.89	0.89	0.88
Monk 1	0.48	0.46	0.45	0.46	0.47	0.46	0.45	0.46	0.90	0.86	0.83	0.75
Monk 2	-	-	-	-	-	-	-	-	-0.23	-0.45	-0.49	-0.52
Monk 3	0.69	0.72	0.73	0.78	0.67	0.70	0.72	0.77	0.81	0.82	0.83	0.85
Simple dependent	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Partially dependent	0.99	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2-way linear corr.	0.64	0.69	0.71	0.72	1.00	1.00	1.00	1.00	0.97	0.99	0.99	0.99
3-way linear corr.	0.90	0.90	0.89	0.89	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.88
3-way non-linear corr.	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.94	0.94	0.95
1 good and noise	0.97	0.95	0.94	0.93	0.94	0.93	0.92	0.92	0.97	0.95	0.94	0.93
Pure noise	-0.47	-0.48	-0.48	-0.44	0.33	0.41	0.41	0.38	-0.50	-0.50	-0.50	-0.46
Un-nested	0.83	0.84	0.84	0.85	0.90	0.90	0.91	0.92	0.86	0.87	0.87	0.88
Monotonic	0.87	0.88	0.88	0.88	0.83	0.83	0.83	0.83	0.91	0.92	0.92	0.92
Two U's	1.00	1.00	1.00	1.00	0.98	0.99	0.99	0.99	1.00	1.00	1.00	0.99
Multimodal (XOR)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
linear sep. Guassian	0.95	0.95	0.95	0.96	0.95	0.96	0.96	0.96	0.98	0.98	0.98	0.98
Redundant	39%	39%	39%	39%	-0.71	-0.71	-0.71	-0.71	39%	39%	39%	39%
2 clusters	53%	49%	48%	47%	53%	49%	48%	47%	53%	49%	48%	47%

Table B.2: Correlation between SVM and 1-NN accuracies and subset-based RELIEF filter values using different values for K on artificial data sets

	RELIEF-F (summed)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.77	0.77	0.77	0.77	0.73	0.73	0.73	0.73	0.48	0.48	0.48	0.48
Cardiotocography	-0.55	-0.60	-0.64	-0.71	-0.57	-0.60	-0.65	-0.73	-0.56	-0.61	-0.64	-0.68
Congressional Voting	0.00	0.82	0.78	0.81	0.00	0.82	0.78	0.81	0.00	0.82	0.77	0.80
Contraceptive Choice	-0.55	0.46	0.46	0.12	-0.21	0.48	0.48	0.20	-0.14	0.56	0.57	0.43
Credit Approval	0.46	0.46	0.47	0.46	0.46	0.47	0.47	0.46	0.54	0.51	0.53	0.54
Dermatology	0.63	0.64	0.64	0.63	0.61	0.62	0.62	0.61	0.66	0.66	0.67	0.65
E-coli	-0.35	-0.31	-0.30	-0.31	-0.36	-0.33	-0.33	-0.36	-0.36	-0.31	-0.30	-0.31
Flag	-0.46	-0.48	-0.44	-0.46	-0.70	-0.74	-0.71	-0.72	-0.32	-0.31	-0.25	-0.27
Glass	-0.16	-0.33	-0.43	-0.55	-0.17	-0.29	-0.39	-0.53	-0.16	-0.31	-0.41	-0.54
Haberman's Survival	0.81	0.59	0.57	0.44	0.52	0.18	0.22	0.23	-0.74	-0.73	-0.42	-0.11
Ionosphere	0.77	0.76	0.75	0.72	0.78	0.77	0.75	0.73	0.30	0.27	0.25	0.23
Iris	0.74	0.74	0.74	0.74	0.81	0.81	0.81	0.81	0.82	0.82	0.82	0.82
Mushroom	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.58	0.58	0.58	0.58
Page blocks	0.51	0.50	0.50	0.50	0.63	0.63	0.63	0.62	0.50	0.49	0.48	0.48
Post-op	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Segmentation	0.62	0.62	0.62	0.62	0.65	0.65	0.65	0.65	0.55	0.55	0.55	0.55
Statlog (vehicle)	0.48	0.13	-0.45	-0.79	0.49	0.14	-0.46	-0.80	0.41	0.18	-0.29	-0.58
WI Breast Cancer (orig.)	0.58	0.56	0.61	0.67	0.58	0.56	0.61	0.67	0.62	0.60	0.63	0.69
WI Breast Cancer (diag.)	0.48	0.48	0.49	0.49	0.48	0.49	0.50	0.50	0.53	0.53	0.54	0.54
WI Breast Cancer (prog.)	-0.79	-0.80	-0.81	-0.81	-0.78	-0.79	-0.79	-0.80	0.20	0.21	0.20	0.20
Wine	0.76	0.76	0.75	0.72	0.76	0.75	0.74	0.71	0.72	0.72	0.71	0.69
Yeast	-0.49	-0.66	-0.67	-0.69	-0.24	-0.42	-0.46	-0.45	-0.58	-0.74	-0.75	-0.78

Table B.3: Correlation of SVM and 1-NN accuracies and RELIEF-F filter values using different values of K on real data sets

	RELIEF-subset											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10	K=1	K=3	K=5	K=10
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.70	0.72	0.69	0.66	0.66	0.67	0.65	0.62	0.43	0.47	0.43	0.39
Cardiotocography	0.87	0.88	0.89	0.90	0.88	0.90	0.91	0.91	0.77	0.77	0.77	0.76
Congressional Voting	0.80	0.85	0.86	0.88	0.80	0.85	0.86	0.88	0.76	0.81	0.82	0.84
Contraceptive Choice	-0.16	-0.31	-0.34	-0.33	-0.17	-0.33	-0.36	-0.40	0.19	0.10	0.10	0.12
Credit Approval	0.71	0.77	0.79	0.81	0.70	0.77	0.79	0.80	0.79	0.82	0.83	0.84
Dermatology	0.85	0.84	0.84	0.83	0.82	0.81	0.81	0.81	0.88	0.87	0.87	0.86
E-coli	0.88	0.89	0.89	0.88	0.83	0.85	0.84	0.83	0.90	0.91	0.91	0.91
Flag	0.30	0.16	0.11	0.09	0.19	0.04	0.00	0.03	0.45	0.32	0.26	0.21
Glass	0.75	0.72	0.68	0.62	0.79	0.76	0.72	0.64	0.75	0.73	0.70	0.65
Haberman's Survival	0.10	0.01	0.06	0.11	-0.17	-0.27	-0.23	-0.16	-0.02	-0.01	-0.02	0.04
Ionosphere	0.74	0.74	0.73	0.72	0.75	0.74	0.73	0.72	0.32	0.31	0.29	0.29
Iris	0.84	0.87	0.87	0.88	0.93	0.94	0.95	0.95	0.91	0.93	0.94	0.94
Mushroom	0.72	0.72	0.73	0.73	0.72	0.72	0.72	0.72	0.69	0.70	0.71	0.72
Page blocks	0.60	0.60	0.60	0.59	0.73	0.73	0.72	0.72	0.49	0.49	0.49	0.49
Post-op	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Segmentation	0.81	0.82	0.82	0.82	0.86	0.86	0.86	0.86	0.73	0.73	0.74	0.74
Statlog (vehicle)	0.89	0.88	0.87	0.84	0.87	0.86	0.85	0.80	0.84	0.85	0.85	0.86
WI Breast Cancer (orig.)	0.73	0.77	0.79	0.81	0.73	0.77	0.79	0.81	0.77	0.81	0.83	0.83
WI Breast Cancer (diag.)	0.67	0.66	0.66	0.65	0.67	0.67	0.67	0.66	0.71	0.70	0.70	0.70
WI Breast Cancer (prog.)	-0.65	-0.69	-0.73	-0.77	-0.65	-0.69	-0.73	-0.77	0.47	0.38	0.33	0.26
Wine	0.85	0.86	0.86	0.86	0.86	0.86	0.87	0.86	0.83	0.84	0.84	0.84
Yeast	0.56	0.32	0.23	0.21	0.53	0.34	0.25	0.23	0.48	0.22	0.12	0.08

Table B.4: Correlation of SVM and 1-NN accuracies and subset RELIEF filter values using different values of K on real data sets

B.2 Probability-based measures

	Kullback-Leibler divergence							
	SVM (one vs. one)							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.57	0.55	0.54	0.57	0.58	0.57	0.57	0.58
Binary 2	0.69	0.68	0.70	0.70	0.70	0.69	0.69	0.70
Binary 3	-	100%	100%	100%	100%	-	-	100%
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99	0.98
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.95	0.91	0.91	0.91	0.91	0.68	0.91	0.95
Simple dependent	1.00	0.99	0.98	1.00	1.00	0.96	1.00	1.00
Partially dependent	0.99	0.99	1.00	0.99	1.00	-0.36	-0.36	0.99
2-way linear corr.	0.64	0.59	0.66	0.68	0.70	0.77	0.73	0.68
3-way linear corr.	0.79	0.74	0.80	0.78	0.80	0.02	0.02	0.80
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.99	0.99	0.99	0.99	0.97	0.98	0.98	1.00
Pure noise	0.67	0.75	0.70	0.63	0.62	0.78	0.77	0.74
Un-nested	0.77	0.78	0.80	0.80	0.81	0.47	0.44	0.12
Monotonic	0.80	0.80	0.80	0.79	0.80	0.80	0.80	0.81
Two U's	0.99	1.00	1.00	1.00	0.98	0.99	0.99	1.00
Multimodal (XOR)	0.79	0.95	0.75	0.86	0.87	0.79	0.79	0.99
linear sep. Guassian	1.00	1.00	0.99	0.91	0.98	0.98	0.98	0.97
Redundant	100%	100%	100%	100%	100%	100%	100%	101%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%
	Kullback-Leibler divergence							
	SVM (one vs. all)							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.54	0.53	0.51	0.54	0.54	0.54	0.54	0.55
Binary 2	0.53	0.51	0.55	0.56	0.56	0.53	0.52	0.54
Binary 3	-	100%	100%	100%	100%	-	-	100%
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99	0.98
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.94	0.90	0.90	0.90	0.90	0.65	0.90	0.95
Simple dependent	1.00	1.00	0.98	1.00	1.00	0.96	1.00	1.00
Partially dependent	0.99	0.99	1.00	0.99	1.00	-0.36	-0.36	0.99
2-way linear corr.	1.00	0.99	1.00	1.00	1.00	0.99	1.00	1.00
3-way linear corr.	0.80	0.76	0.81	0.80	0.81	0.02	0.02	0.81
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.97	0.97	0.97	0.97	0.94	0.95	0.96	0.98
Pure noise	-0.53	-0.66	-0.56	-0.47	-0.45	-0.72	-0.70	-0.64
Un-nested	0.83	0.85	0.85	0.85	0.85	0.65	0.62	0.13
Monotonic	0.80	0.81	0.80	0.80	0.81	0.81	0.81	0.79
Two U's	1.00	1.00	1.00	1.00	0.99	0.97	0.97	0.99
Multimodal (XOR)	0.88	0.88	0.63	0.93	0.78	0.88	0.88	1.00
linear sep. Guassian	0.99	1.00	0.99	0.93	0.98	0.98	0.98	0.98
Redundant	-0.65	-0.65	-0.65	-0.65	-0.65	-0.66	-0.67	-0.66
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%

Table B.5: Correlation between SVM accuracies and Kullback-Leibler divergence filter values using different quantizations on artificial data sets

	Kullback-Leibler divergence							
	KNN							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.93	0.86	0.90	0.92	0.94	0.93	0.93	0.92
Binary 2	0.97	0.97	0.95	0.94	0.94	0.97	0.97	0.96
Binary 3	0.00	0.73	0.84	0.89	0.93	0.00	0.00	0.82
Monk 1	0.60	0.59	0.59	0.59	0.59	0.59	0.60	0.59
Monk 2	-0.47	-0.56	-0.56	-0.56	-0.56	-0.56	-0.48	-0.53
Monk 3	0.98	0.95	0.95	0.95	0.95	0.73	0.95	0.98
Simple dependent	1.00	1.00	0.98	1.00	1.00	0.96	1.00	1.00
Partially dependent	0.98	0.99	0.99	0.99	1.00	-0.32	-0.32	0.98
2-way linear corr.	0.97	0.95	0.98	0.98	0.99	1.00	0.99	0.98
3-way linear corr.	0.77	0.70	0.78	0.75	0.77	0.04	0.04	0.78
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.98	0.98	0.98	0.98	0.96	0.97	0.97	0.99
Pure noise	0.78	0.96	0.85	0.72	0.80	0.87	0.87	0.90
Un-nested	0.80	0.81	0.82	0.82	0.83	0.53	0.48	0.15
Monotonic	0.83	0.84	0.84	0.83	0.84	0.84	0.84	0.84
Two U's	0.95	0.99	0.99	0.98	0.94	1.00	1.00	0.99
Multimodal (XOR)	0.87	0.89	0.64	0.92	0.79	0.87	0.87	1.00
linear sep. Guassian	0.98	0.99	0.98	0.95	0.99	0.99	0.99	0.99
Redundant	100%	100%	100%	100%	100%	100%	100%	101%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%

Table B.6: Correlation between 1-NN accuracies and Kullback-Leibler divergence filter values using different quantizations on artificial data sets

	Jensen-Shannon divergence						
	SVM (one vs. one)						
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$
Binary 1	0.57	0.57	0.57	0.58	0.58	0.57	0.57
Binary 2	0.69	0.68	0.70	0.70	0.70	0.69	0.69
Binary 3	-	100%	100%	100%	100%	-	-
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99
Monk 2	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.94	0.95	0.95	0.95	0.95	0.95	0.95
Simple dependent	1.00	0.99	0.99	1.00	1.00	0.97	0.95
Partially dependent	0.99	1.00	1.00	0.99	1.00	-0.36	0.99
2-way linear corr.	0.66	0.61	0.67	0.69	0.68	0.61	0.82
3-way linear corr.	0.80	0.73	0.80	0.79	0.80	0.12	0.12
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.99	0.99	0.99	0.99	0.94	0.96	0.96
Pure noise	0.67	0.75	0.69	0.72	0.53	0.53	0.45
Un-nested	0.79	0.79	0.81	0.82	0.82	0.47	0.44
Monotonic	0.80	0.80	0.80	0.79	0.80	0.80	0.80
Two U's	0.99	1.00	1.00	1.00	1.00	1.00	1.00
Multimodal (XOR)	0.79	0.95	0.75	0.73	1.00	0.79	0.79
linear sep. Guassian	1.00	1.00	1.00	0.91	0.97	0.97	0.97
Redundant	100%	100%	100%	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%
	Jensen-Shannon divergence						
	SVM (one vs. all)						
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$
Binary 1	0.54	0.54	0.54	0.55	0.54	0.54	0.54
Binary 2	0.53	0.52	0.54	0.55	0.55	0.53	0.52
Binary 3	-	100%	100%	100%	100%	-	-
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99
Monk 2	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.94	0.94	0.94	0.94	0.94	0.94	0.94
Simple dependent	1.00	1.00	0.99	1.00	1.00	0.96	0.95
Partially dependent	0.99	1.00	1.00	0.99	1.00	-0.36	0.99
2-way linear corr.	1.00	0.99	1.00	1.00	1.00	0.99	0.98
3-way linear corr.	0.81	0.75	0.81	0.80	0.81	0.15	0.15
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.97	0.96	0.96	0.96	0.92	0.93	0.93
Pure noise	-0.53	-0.65	-0.55	-0.61	-0.33	-0.33	-0.25
Un-nested	0.83	0.85	0.85	0.85	0.83	0.65	0.62
Monotonic	0.80	0.81	0.80	0.80	0.81	0.81	0.81
Two U's	1.00	0.99	1.00	1.00	1.00	1.00	0.99
Multimodal (XOR)	0.88	0.88	0.63	0.83	0.96	0.88	0.88
linear sep. Guassian	1.00	1.00	1.00	0.92	0.98	0.98	0.98
Redundant	-0.65	-0.65	-0.65	-0.65	-0.65	-0.66	-0.64
2 clusters	100%	100%	100%	100%	100%	100%	100%

Table B.7: Correlation between SVM accuracies and Jensen-Shannon divergence filter values using different quantizations on artificial data sets

	Jensen-Shannon divergence						
	KNN						
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$
Binary 1	0.93	0.90	0.93	0.93	0.94	0.93	0.93
Binary 2	0.97	0.97	0.96	0.95	0.95	0.97	0.97
Binary 3	0.00	0.73	0.84	0.90	0.93	0.00	0.00
Monk 1	0.60	0.59	0.59	0.59	0.59	0.59	0.60
Monk 2	-0.47	-0.56	-0.56	-0.56	-0.56	-0.56	-0.48
Monk 3	0.98	0.98	0.98	0.98	0.98	0.98	0.98
Simple dependent	1.00	1.00	0.99	1.00	1.00	0.97	0.95
Partially dependent	0.99	0.99	0.99	0.99	1.00	-0.32	0.99
2-way linear corr.	0.98	0.96	0.98	0.98	0.98	0.96	1.00
3-way linear corr.	0.78	0.69	0.78	0.75	0.77	0.07	0.07
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.98	0.98	0.98	0.97	0.93	0.95	0.95
Pure noise	0.78	0.96	0.84	0.83	0.68	0.68	0.73
Un-nested	0.82	0.82	0.83	0.84	0.84	0.53	0.48
Monotonic	0.84	0.84	0.83	0.83	0.84	0.84	0.84
Two U's	0.95	1.00	0.98	0.98	0.99	0.99	1.00
Multimodal (XOR)	0.87	0.89	0.65	0.82	0.97	0.87	0.87
linear sep. Guassian	0.98	0.99	0.99	0.95	0.99	0.99	0.99
Redundant	100%	100%	100%	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%

Table B.8: Correlation between 1-NN accuracies and Jensen-Shannon divergence filter values using different quantizations on artificial data sets

	Bhattacharyya divergence							
	SVM (one vs. one)							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.57	0.55	0.54	0.57	0.58	0.57	0.57	0.58
Binary 2	0.69	0.68	0.70	0.70	0.70	0.69	0.69	0.69
Binary 3	-	100%	100%	100%	100%	-	-	100%
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99	0.98
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.95	0.94	0.94	0.94	0.94	0.70	0.94	0.88
Simple dependent	1.00	0.99	0.98	1.00	0.99	0.91	0.89	0.99
Partially dependent	0.99	0.99	0.99	0.99	0.99	-0.36	0.99	1.00
2-way linear corr.	0.61	0.57	0.63	0.67	0.67	0.57	0.92	0.69
3-way linear corr.	0.78	0.79	0.79	0.80	0.79	0.12	0.12	0.53
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.87
1 good and noise	1.00	0.99	1.00	0.99	0.98	0.98	0.99	1.00
Pure noise	0.67	0.75	0.69	0.63	0.62	0.78	0.77	0.72
Un-nested	0.75	0.75	0.78	0.78	0.80	0.47	0.44	0.12
Monotonic	0.80	0.80	0.80	0.79	0.80	0.80	0.80	0.76
Two U's	0.99	0.95	0.99	0.99	0.97	0.99	0.94	0.99
Multimodal (XOR)	0.79	0.95	0.75	0.68	1.00	0.79	0.79	1.00
linear sep. Guassian	0.99	1.00	0.99	0.91	0.98	0.98	0.98	1.00
Redundant	100%	100%	100%	100%	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%
	Bhattacharyya divergence							
	SVM (one vs. all)							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.54	0.53	0.51	0.54	0.54	0.54	0.54	0.54
Binary 2	0.53	0.51	0.55	0.56	0.56	0.53	0.52	0.51
Binary 3	-	100%	100%	100%	100%	-	-	100%
Monk 1	0.99	0.98	0.98	0.98	0.98	0.98	0.99	0.98
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.94	0.92	0.92	0.92	0.92	0.67	0.92	0.86
Simple dependent	1.00	1.00	0.99	1.00	0.99	0.90	0.88	0.99
Partially dependent	0.99	0.99	0.99	0.99	0.99	-0.36	0.99	1.00
2-way linear corr.	0.99	0.99	1.00	1.00	1.00	0.99	0.92	1.00
3-way linear corr.	0.79	0.81	0.80	0.81	0.80	0.15	0.15	0.56
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.87
1 good and noise	0.97	0.97	0.97	0.97	0.95	0.96	0.96	0.98
Pure noise	-0.53	-0.66	-0.56	-0.47	-0.45	-0.72	-0.70	-0.66
Un-nested	0.83	0.84	0.85	0.85	0.85	0.65	0.62	0.13
Monotonic	0.80	0.80	0.80	0.80	0.81	0.81	0.81	0.77
Two U's	1.00	0.92	0.97	0.98	0.95	0.97	0.90	0.98
Multimodal (XOR)	0.88	0.88	0.63	0.80	0.98	0.88	0.88	1.00
linear sep. Guassian	0.99	1.00	0.99	0.92	0.99	0.99	0.99	1.00
Redundant	-0.65	-0.65	-0.65	-0.65	-0.65	-0.65	-0.67	-0.65
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%

Table B.9: Correlation between SVM accuracies and Bhattacharyya divergence filter values using different quantizations on artificial data sets

	Bhattacharyya divergence							
	KNN							
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = e$	$N_b = u$	Gaussian
Binary 1	0.93	0.86	0.90	0.92	0.94	0.93	0.93	0.93
Binary 2	0.97	0.97	0.96	0.94	0.94	0.97	0.97	0.97
Binary 3	0.00	0.73	0.84	0.90	0.93	0.00	0.00	0.85
Monk 1	0.60	0.59	0.59	0.59	0.59	0.59	0.60	0.59
Monk 2	-0.47	-0.56	-0.56	-0.56	-0.56	-0.56	-0.48	-0.57
Monk 3	0.98	0.98	0.97	0.97	0.97	0.75	0.97	0.92
Simple dependent	1.00	1.00	0.98	1.00	0.99	0.91	0.88	0.99
Partially dependent	0.98	0.99	0.99	0.98	0.99	-0.32	0.98	0.99
2-way linear corr.	0.96	0.95	0.97	0.98	0.98	0.95	0.97	0.98
3-way linear corr.	0.76	0.75	0.77	0.78	0.77	0.07	0.07	0.49
3-way non-linear corr.	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
1 good and noise	0.99	0.98	0.98	0.98	0.96	0.97	0.98	0.99
Pure noise	0.79	0.96	0.85	0.71	0.80	0.87	0.87	0.69
Un-nested	0.79	0.79	0.81	0.81	0.83	0.53	0.48	0.15
Monotonic	0.83	0.84	0.84	0.83	0.84	0.84	0.84	0.79
Two U's	0.96	0.99	1.00	1.00	1.00	1.00	0.98	1.00
Multimodal (XOR)	0.87	0.89	0.64	0.78	0.98	0.87	0.87	1.00
linear sep. Guassian	0.97	0.99	0.97	0.95	1.00	1.00	1.00	1.00
Redundant	100%	100%	100.01%	100%	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%

Table B.10: Correlation between 1-NN accuracies and Bhattacharyya divergence filter values using different quantizations on artificial data sets

	Kullback-Leibler divergence											
	SVM (one vs. one)						SVM (one vs. all)					
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaus.	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaus.
Abalone	0.37	0.45	0.43	0.42	0.39	0.00	0.31	0.28	0.26	0.27	0.26	0.00
Car Evaluation	0.64	0.74	0.71	0.71	0.71	0.55	0.57	0.67	0.63	0.63	0.63	0.46
Cardiotocography	0.71	0.80	0.82	0.83	0.85	0.42	0.73	0.82	0.84	0.85	0.87	0.45
Congressional Voting	0.80	0.81	0.81	0.81	0.80	0.86	0.80	0.80	0.80	0.80	0.80	0.85
Contraceptive Choice	0.56	0.58	0.70	0.73	0.80	0.51	0.65	0.66	0.76	0.78	0.82	0.57
Credit Approval	0.91	0.91	0.91	0.91	0.88	0.35	0.90	0.90	0.90	0.91	0.88	0.36
Dermatology	0.72	0.73	0.73	0.73	0.73	0.72	0.69	0.71	0.71	0.71	0.71	0.70
E-coli	0.74	0.71	0.77	0.80	0.77	0.02	0.73	0.67	0.76	0.78	0.75	0.01
Flag	0.65	0.63	0.62	0.62	0.61	0.66	0.84	0.83	0.83	0.83	0.82	0.65
Glass	0.68	0.66	0.71	0.76	0.71	0.00	0.75	0.72	0.77	0.80	0.77	0.00
Haberman's Survival	-0.67	-0.78	-0.71	-0.77	-0.73	-0.87	-0.70	-0.96	-0.93	-0.92	-0.83	-0.95
Ionosphere	0.75	0.69	0.67	0.67	0.66	0.69	0.75	0.70	0.68	0.67	0.66	0.69
Iris	0.70	0.74	0.72	0.72	0.70	0.70	0.77	0.80	0.78	0.79	0.77	0.73
Mushroom	0.63	0.67	0.67	0.68	0.68	0.32	0.63	0.67	0.67	0.68	0.68	0.32
Page blocks	0.52	0.56	0.65	0.66	0.72	0.55	0.67	0.71	0.77	0.78	0.82	0.69
Post-op	-0.52	-0.63	-0.08	-0.48	-0.12	-0.73	-0.56	-0.56	0.15	-0.47	-0.05	-0.80
Segmentation	0.73	0.69	0.71	0.73	0.72	0.32	0.78	0.74	0.76	0.78	0.78	0.36
Statlog (vehicle)	0.74	0.79	0.78	0.81	0.81	0.77	0.76	0.81	0.80	0.83	0.84	0.79
WI Breast Cancer (orig.)	0.71	0.69	0.69	0.69	0.68	0.62	0.71	0.69	0.69	0.69	0.69	0.61
WI Breast Cancer (diag.)	0.47	0.50	0.50	0.51	0.51	0.47	0.47	0.51	0.51	0.51	0.51	0.47
WI Breast Cancer (prog.)	-0.77	-0.79	-0.81	-0.82	-0.83	-0.81	-0.74	-0.76	-0.79	-0.80	-0.81	-0.77
Wine	0.72	0.70	0.75	0.72	0.73	0.69	0.71	0.69	0.74	0.71	0.73	0.67
Yeast	0.34	0.75	0.56	0.62	0.74	0.02	0.07	0.45	0.26	0.30	0.43	0.01

Table B.11: Correlation between SVM accuracies and KL divergence filter values using different quantizations on real data sets

	Kullback-Leibler divergence					
	KNN					
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaus.
Abalone	0.56	0.69	0.69	0.68	0.67	0.00
Car Evaluation	0.44	0.49	0.47	0.47	0.47	0.35
Cardiotocography	0.67	0.71	0.73	0.74	0.75	0.33
Congressional Voting	0.77	0.77	0.77	0.77	0.77	0.84
Contraceptive Choice	0.14	0.17	0.31	0.34	0.50	0.14
Credit Approval	0.87	0.88	0.89	0.90	0.89	0.37
Dermatology	0.76	0.77	0.77	0.77	0.77	0.75
E-coli	0.75	0.70	0.77	0.80	0.77	0.05
Flag	0.50	0.48	0.47	0.46	0.45	0.59
Glass	0.56	0.58	0.61	0.65	0.68	0.00
Haberman's Survival	0.63	0.37	0.35	0.47	0.55	0.50
Ionosphere	0.34	0.23	0.18	0.20	0.19	0.29
Iris	0.77	0.81	0.79	0.79	0.77	0.75
Mushroom	0.54	0.59	0.60	0.61	0.61	0.26
Page blocks	0.50	0.54	0.59	0.60	0.63	0.60
Post-op	0.00	-0.01	0.63	0.41	0.51	0.20
Segmentation	0.64	0.61	0.63	0.64	0.63	0.25
Statlog (vehicle)	0.57	0.66	0.63	0.65	0.66	0.63
WI Breast Cancer (orig.)	0.73	0.72	0.71	0.71	0.71	0.63
WI Breast Cancer (diag.)	0.51	0.55	0.55	0.55	0.55	0.48
WI Breast Cancer (prog.)	0.16	0.14	0.17	0.17	0.19	0.16
Wine	0.67	0.69	0.71	0.69	0.70	0.66
Yeast	0.45	0.81	0.66	0.73	0.83	0.02

Table B.12: Correlation between 1-NN accuracies and KL divergence filter values using different quantizations on real data sets

	Jensen-Shannon divergence									
	SVM (one vs. one)					SVM (one vs. all)				
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$
Abalone	0.32	0.34	0.42	0.42	0.44	0.28	0.23	0.25	0.26	0.26
Car Evaluation	0.60	0.71	0.66	0.66	0.66	0.53	0.64	0.57	0.57	0.57
Cardiotocography	0.73	0.80	0.82	0.83	0.84	0.74	0.82	0.84	0.85	0.86
Congressional Voting	0.78	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
Contraceptive Choice	0.56	0.58	0.70	0.73	0.80	0.65	0.66	0.77	0.78	0.82
Credit Approval	0.91	0.91	0.90	0.90	0.87	0.90	0.90	0.90	0.90	0.87
Dermatology	0.72	0.74	0.74	0.74	0.74	0.70	0.72	0.72	0.72	0.71
E-coli	0.69	0.65	0.75	0.80	0.85	0.69	0.62	0.74	0.78	0.82
Flag	0.66	0.64	0.64	0.64	0.63	0.84	0.83	0.83	0.82	0.81
Glass	0.64	0.61	0.69	0.73	0.73	0.70	0.67	0.76	0.78	0.78
Haberman's Survival	-0.67	-0.77	-0.71	-0.79	-0.70	-0.70	-0.96	-0.93	-0.92	-0.86
Ionosphere	0.74	0.68	0.67	0.66	0.65	0.75	0.69	0.67	0.67	0.66
Iris	0.69	0.73	0.71	0.71	0.69	0.76	0.79	0.77	0.78	0.77
Mushroom	0.63	0.70	0.70	0.70	0.70	0.63	0.70	0.70	0.70	0.70
Page blocks	0.52	0.56	0.63	0.65	0.71	0.67	0.71	0.76	0.77	0.82
Post-op	0.05	-0.01	-0.29	-0.63	-0.61	0.30	0.25	-0.08	-0.64	-0.64
Segmentation	0.74	0.69	0.71	0.73	0.73	0.80	0.75	0.77	0.79	0.78
Statlog (vehicle)	0.77	0.80	0.80	0.81	0.82	0.79	0.83	0.82	0.84	0.84
WI Breast Cancer (orig.)	0.71	0.69	0.68	0.68	0.68	0.71	0.69	0.68	0.68	0.68
WI Breast Cancer (diag.)	0.48	0.50	0.51	0.51	0.51	0.48	0.51	0.51	0.52	0.52
WI Breast Cancer (prog.)	-0.78	-0.83	-0.83	-0.83	-0.83	-0.74	-0.81	-0.81	-0.81	-0.81
Wine	0.73	0.71	0.75	0.73	0.73	0.72	0.71	0.74	0.72	0.73
Yeast	0.35	0.72	0.54	0.55	0.69	0.09	0.45	0.24	0.24	0.38

	Jensen-Shannon divergence				
	KNN				
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$
Abalone	0.48	0.64	0.69	0.70	0.71
Car Evaluation	0.42	0.47	0.45	0.45	0.45
Cardiotocography	0.68	0.71	0.73	0.74	0.75
Congressional Voting	0.74	0.73	0.73	0.73	0.73
Contraceptive Choice	0.14	0.17	0.31	0.34	0.46
Credit Approval	0.87	0.88	0.90	0.90	0.89
Dermatology	0.76	0.78	0.77	0.77	0.77
E-coli	0.70	0.65	0.76	0.80	0.84
Flag	0.51	0.52	0.53	0.53	0.51
Glass	0.51	0.51	0.56	0.62	0.70
Haberman's Survival	0.63	0.36	0.35	0.49	0.47
Ionosphere	0.32	0.23	0.18	0.20	0.18
Iris	0.76	0.81	0.78	0.78	0.76
Mushroom	0.54	0.62	0.63	0.63	0.64
Page blocks	0.50	0.54	0.58	0.59	0.63
Post-op	0.59	0.58	0.40	0.35	0.30
Segmentation	0.64	0.61	0.63	0.64	0.63
Statlog (vehicle)	0.60	0.67	0.64	0.65	0.66
WI Breast Cancer (orig.)	0.74	0.71	0.71	0.71	0.71
WI Breast Cancer (diag.)	0.52	0.55	0.55	0.55	0.55
WI Breast Cancer (prog.)	0.15	0.20	0.18	0.19	0.19
Wine	0.68	0.70	0.71	0.69	0.70
Yeast	0.46	0.77	0.63	0.66	0.79

Table B.13: Correlation between SVM and 1-NN accuracies and JS divergence filter values using different quantizations on real data sets

	Bhattacharyya divergence											
	SVM (one vs. one)						SVM (one vs. all)					
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaussian	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaussian
Abalone	0.33	0.45	0.44	0.42	0.39	0.00	0.31	0.27	0.26	0.27	0.26	0.00
Car Evaluation	0.62	0.73	0.70	0.70	0.70	0.59	0.55	0.66	0.61	0.61	0.62	0.50
Cardiotocography	0.71	0.79	0.82	0.83	0.86	0.74	0.72	0.81	0.84	0.85	0.87	0.77
Congressional Voting	0.82	0.83	0.83	0.83	0.82	0.64	0.82	0.82	0.82	0.82	0.82	0.63
Contraceptive Choice	0.56	0.58	0.70	0.73	0.80	0.34	0.65	0.66	0.76	0.78	0.82	0.37
Credit Approval	0.91	0.91	0.91	0.91	0.89	0.42	0.90	0.90	0.90	0.91	0.89	0.43
Dermatology	0.71	0.73	0.73	0.73	0.73	0.73	0.69	0.71	0.71	0.71	0.71	0.71
E-coli	0.75	0.68	0.77	0.80	0.76	0.32	0.74	0.65	0.75	0.78	0.74	0.31
Flag	0.66	0.63	0.63	0.62	0.62	0.71	0.84	0.83	0.83	0.83	0.82	0.80
Glass	0.66	0.64	0.70	0.75	0.71	0.00	0.73	0.71	0.77	0.80	0.77	0.00
Haberman's Survival	-0.67	-0.78	-0.72	-0.78	-0.73	-0.88	-0.70	-0.96	-0.93	-0.92	-0.83	-0.94
Ionosphere	0.28	0.32	0.33	0.37	0.44	0.77	0.28	0.32	0.33	0.38	0.44	0.77
Iris	0.72	0.74	0.73	0.73	0.71	0.72	0.78	0.80	0.79	0.79	0.77	0.75
Mushroom	0.34	0.41	0.42	0.43	0.45	0.55	0.35	0.41	0.42	0.43	0.45	0.55
Page blocks	0.51	0.55	0.62	0.64	0.72	0.61	0.66	0.70	0.75	0.76	0.81	0.70
Post-op	-0.53	-0.62	-0.18	-0.65	-0.69	-0.76	-0.58	-0.55	-0.20	-0.65	-0.74	-0.83
Segmentation	0.71	0.66	0.70	0.73	0.72	0.45	0.76	0.71	0.74	0.78	0.77	0.50
Statlog (vehicle)	0.75	0.79	0.79	0.81	0.81	0.80	0.77	0.81	0.81	0.83	0.83	0.82
WI Breast Cancer (orig.)	0.72	0.69	0.68	0.69	0.68	0.50	0.72	0.70	0.69	0.69	0.68	0.50
WI Breast Cancer (diag.)	0.46	0.50	0.50	0.50	0.51	0.44	0.47	0.50	0.51	0.51	0.51	0.44
WI Breast Cancer (prog.)	-0.77	-0.79	-0.82	-0.82	-0.83	-0.77	-0.74	-0.76	-0.79	-0.80	-0.81	-0.74
Wine	0.71	0.69	0.74	0.71	0.73	0.71	0.70	0.68	0.73	0.71	0.73	0.71
Yeast	0.33	0.74	0.53	0.59	0.73	0.15	0.07	0.46	0.23	0.27	0.41	0.08
	KNN											
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	Gaussian						
Abalone	0.52	0.69	0.69	0.69	0.68	0.00						
Car Evaluation	0.43	0.48	0.47	0.47	0.47	0.38						
Cardiotocography	0.67	0.71	0.73	0.74	0.75	0.58						
Congressional Voting	0.79	0.79	0.79	0.79	0.79	0.59						
Contraceptive Choice	0.14	0.17	0.31	0.34	0.47	0.07						
Credit Approval	0.87	0.87	0.89	0.89	0.89	0.44						
Dermatology	0.75	0.77	0.77	0.77	0.77	0.77						
E-coli	0.76	0.68	0.77	0.80	0.76	0.33						
Flag	0.51	0.48	0.47	0.46	0.45	0.58						
Glass	0.55	0.57	0.61	0.65	0.68	0.00						
Haberman's Survival	0.63	0.37	0.35	0.48	0.55	0.51						
Ionosphere	0.09	0.10	0.10	0.11	0.13	0.31						
Iris	0.79	0.82	0.80	0.80	0.78	0.77						
Mushroom	0.21	0.27	0.29	0.29	0.32	0.46						
Page blocks	0.49	0.53	0.57	0.58	0.62	0.59						
Post-op	-0.01	-0.01	0.41	0.34	0.24	0.17						
Segmentation	0.63	0.59	0.62	0.64	0.63	0.35						
Statlog (vehicle)	0.58	0.66	0.63	0.65	0.66	0.65						
WI Breast Cancer (orig.)	0.74	0.72	0.71	0.71	0.70	0.51						
WI Breast Cancer (diag.)	0.50	0.55	0.54	0.55	0.55	0.45						
WI Breast Cancer (prog.)	0.16	0.15	0.17	0.17	0.19	0.17						
Wine	0.66	0.68	0.71	0.68	0.70	0.68						
Yeast	0.44	0.80	0.62	0.70	0.82	0.14						

Table B.14: Correlation between SVM and 1-NN accuracies and Bhattacharyya divergence filter values using different quantizations on real data sets

B.3 Mutual Information measures

B.3.1 Univariate measures

	Mutual information														
	SVM (one vs. one)					SVM (one vs. all)					KNN				
	$N_b = 2$	3	5	10		$N_b = 2$	3	5	10		$N_b = 2$	3	5	10	
Binary 1	0.57	0.57	0.58	**0.58**		0.54	0.54	**0.55**	0.54		0.93	0.89	0.93	**0.94**	
Binary 2	0.69	0.68	**0.70**	**0.70**		0.53	0.52	**0.55**	**0.55**		**0.97**	**0.97**	0.95	0.95	
Binary 3	-	100%	100%	100%		-	100%	100%	100%		0.00	0.73	0.89	**0.93**	
Monk 1	**0.99**	0.98	0.98	0.98		**0.99**	0.98	0.98	0.98		**0.60**	0.59	0.59	0.59	
Monk 2	100%	100%	100%	100%		100%	100%	100%	100%		**-0.47**	-0.56	-0.56	-0.56	
Monk 3	0.94	**0.95**	**0.95**	**0.95**		**0.94**	**0.94**	**0.94**	**0.94**		**0.98**	**0.98**	**0.98**	**0.98**	
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**		**1.00**	**1.00**	**1.00**	**1.00**		**1.00**	**1.00**	**1.00**	**1.00**	
Partially dependent	0.99	**1.00**	0.99	**1.00**		0.99	**1.00**	0.99	**1.00**		0.99	0.99	0.99	**1.00**	
2-way linear corr.	0.66	0.61	**0.69**	0.68		**1.00**	0.99	**1.00**	**1.00**		**0.98**	0.96	**0.98**	**0.98**	
3-way linear corr.	**0.80**	0.73	0.79	**0.80**		**0.81**	0.75	0.80	**0.81**		**0.78**	0.69	0.75	0.77	
3-way non-linear corr.	**0.88**	**0.88**	**0.88**	**0.88**		**0.88**	**0.88**	**0.88**	**0.88**		**0.88**	**0.88**	**0.88**	**0.88**	
1 good and noise	**1.00**	0.99	0.99	0.97		**0.97**	**0.97**	**0.97**	0.94		**0.98**	**0.98**	**0.98**	0.96	
Pure noise	0.71	**0.75**	0.73	0.55		-0.58	-0.67	-0.63	**-0.36**		0.82	**0.97**	0.84	0.72	
Un-nested	0.76	0.78	0.81	**0.83**		0.81	**0.85**	**0.85**	0.83		0.79	0.81	0.83	**0.84**	
Monotonic	**0.80**	**0.80**	0.79	**0.80**		0.80	**0.81**	0.80	**0.81**		0.83	**0.84**	0.82	**0.84**	
Two U's	0.99	**1.00**	**1.00**	**1.00**		**1.00**	0.99	**1.00**	**1.00**		0.95	**1.00**	0.98	0.99	
Multimodal (XOR)	0.79	0.95	0.73	**1.00**		0.88	0.88	0.83	**0.96**		0.87	0.89	0.82	**0.97**	
linear sep. Guassian	**1.00**	**1.00**	0.90	0.97		0.99	**1.00**	0.91	0.98		0.98	**0.99**	0.94	**0.99**	
Redundant	100%	100%	100%	100%		**-0.65**	**-0.65**	**-0.65**	**-0.65**		100%	100%	100%	100%	
2 clusters	100%	100%	100%	100%		100%	100%	100%	100%		100%	100%	100%	100%	

Table B.15: Correlation between SVM and 1-NN accuracies and mutual information filter values using different quantizations on artificial data sets

	Symmetric uncertainty											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Binary 1	0.57	0.57	**0.58**	**0.58**	0.54	0.54	**0.55**	0.54	0.93	0.89	0.93	**0.94**
Binary 2	0.69	0.68	**0.70**	**0.70**	0.53	0.52	**0.55**	0.54	**0.97**	**0.97**	0.95	0.96
Binary 3	-	100%	100%	100%	-	100%	100%	100%	0	0.73	0.91	**0.92**
Monk 1	**0.99**	0.98	0.98	0.98	**0.99**	0.98	0.98	0.98	**0.6**	0.59	0.59	0.59
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%	**-0.47**	-0.57	-0.57	-0.57
Monk 3	**0.95**	0.94	**0.95**	**0.95**	**0.94**	0.93	**0.94**	**0.94**	0.98	0.98	**0.99**	**0.99**
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Partially dependent	0.99	**1.00**	0.99	**1.00**	0.99	**1.00**	0.99	**1.00**	0.99	0.99	0.99	**1.00**
2-way linear corr.	0.66	0.63	0.68	**0.69**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	0.97	0.98	**0.99**
3-way linear corr.	**0.80**	0.76	0.79	**0.80**	**0.81**	0.78	**0.81**	**0.81**	**0.78**	0.72	0.76	0.77
3-way non-linear corr.	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**	**0.88**
1 good and noise	**1.00**	0.99	0.99	0.98	**0.97**	**0.97**	**0.97**	**0.96**	**0.98**	0.98	**0.98**	0.97
Pure noise	0.71	**0.76**	0.74	0.56	-0.58	-0.68	-0.63	**-0.37**	0.82	**0.97**	0.84	0.72
Un-nested	0.75	0.77	0.77	**0.78**	0.80	**0.84**	0.83	**0.84**	0.78	0.80	0.80	**0.81**
Monotonic	0.80	**0.81**	0.79	0.80	0.80	**0.81**	0.80	**0.81**	0.83	**0.84**	0.82	**0.84**
Two U's	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	0.95	**0.99**	0.97	0.98
Multimodal (XOR)	0.79	0.93	0.72	**0.99**	0.88	0.85	0.83	**0.96**	0.87	0.87	0.81	**0.97**
linear sep. Guassian	**1.00**	**1.00**	0.91	0.97	0.99	**1.00**	0.92	0.98	0.98	**1.00**	0.95	0.99
Redundant	100%	100%	100%	100%	**-0.65**	**-0.65**	**-0.65**	**-0.65**	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	99.99%	100%

Table B.16: Correlation between SVM and 1-NN accuracies and symmetric uncertainty filter values using different quantizations on artificial data sets

	Mutual information											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	0.16	0.4	0.44	**0.46**	**0.27**	0.22	**0.27**	0.25	0.35	0.69	0.69	**0.7**
Car Evaluation	**0.79**	**0.79**	**0.79**	**0.79**	**0.74**	**0.74**	0.73	0.73	**0.51**	**0.51**	**0.51**	**0.51**
Cardiotocography	0.73	0.81	0.84	**0.86**	0.74	0.83	0.86	**0.88**	0.67	0.71	0.74	**0.75**
Congressional Voting	**0.78**	**0.78**	**0.78**	**0.78**	**0.77**	**0.77**	**0.77**	**0.77**	**0.74**	**0.74**	**0.74**	**0.74**
Contraceptive Choice	0.57	0.6	0.77	**0.82**	0.65	0.66	0.79	**0.82**	0.15	0.18	0.39	**0.51**
Credit Approval	**0.91**	**0.91**	0.9	0.87	**0.9**	**0.9**	**0.9**	0.87	0.87	0.88	**0.9**	0.89
Dermatology	0.72	**0.73**	**0.73**	**0.73**	0.7	**0.71**	**0.71**	**0.71**	0.76	**0.77**	**0.77**	**0.77**
E-coli	0.88	0.89	**0.9**	**0.9**	0.86	0.86	**0.87**	**0.87**	0.88	0.88	**0.89**	**0.89**
Flag	**0.73**	0.69	0.7	0.69	**0.87**	0.86	0.86	0.84	0.57	0.55	**0.59**	0.58
Glass	0.69	0.62	**0.76**	**0.76**	0.74	0.69	**0.8**	**0.8**	0.53	0.58	0.59	**0.68**
Haberman's Survival	**-0.68**	-0.79	-0.8	-0.72	**-0.72**	-0.96	-0.92	-0.87	**0.63**	0.38	0.49	0.48
Ionosphere	**0.74**	0.68	0.66	0.65	**0.75**	0.69	0.67	0.66	**0.33**	0.23	0.2	0.18
Iris	0.69	**0.74**	0.72	0.7	0.76	**0.8**	0.79	0.78	0.76	**0.81**	0.79	0.78
Mushroom	0.63	**0.7**	**0.7**	**0.7**	0.63	**0.7**	**0.7**	**0.7**	0.54	0.62	**0.63**	**0.63**
Page blocks	0.51	0.58	0.62	**0.68**	0.66	0.75	0.78	**0.82**	0.48	0.58	0.59	**0.62**
Post-op	**-0.24**	-0.36	-0.34	-0.34	**-0.03**	-0.17	-0.17	-0.17	**0.45**	0.4	0.36	0.36
Segmentation	**0.73**	0.69	**0.73**	0.72	0.78	0.74	**0.79**	0.78	**0.64**	0.61	**0.64**	0.63
Statlog (vehicle)	0.78	0.81	**0.82**	**0.82**	0.8	0.83	**0.84**	**0.84**	0.61	**0.67**	0.65	**0.67**
WI Breast Cancer (orig.)	**0.71**	0.69	0.68	0.68	**0.71**	0.69	0.68	0.68	**0.74**	0.71	0.71	0.71
WI Breast Cancer (diag.)	0.48	0.5	**0.51**	**0.51**	0.48	0.51	**0.52**	**0.52**	0.52	**0.55**	**0.55**	**0.55**
WI Breast Cancer (prog.)	**-0.77**	-0.82	-0.83	-0.83	**-0.74**	-0.8	-0.81	-0.8	0.15	**0.19**	0.18	**0.19**
Wine	0.74	0.73	0.74	**0.75**	**0.74**	0.72	**0.74**	**0.74**	0.69	0.71	0.71	**0.72**
Yeast	0.57	**0.94**	0.8	0.88	0.26	**0.66**	0.46	0.57	0.66	**0.96**	0.87	0.94
avg	0.52	0.54	0.55	0.56	0.54	0.54	0.55	0.56	0.58	0.60	0.62	0.63

Table B.17: Correlation between SVM and 1-NN accuracies and mutual information filter values using different quantizations on real data sets

	Symmetric uncertainty											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	0.18	0.41	0.44	**0.45**	**0.27**	0.22	**0.27**	0.25	0.36	0.69	0.69	**0.7**
Car Evaluation	**0.79**	**0.79**	**0.79**	**0.79**	**0.74**	0.73	0.73	0.73	**0.51**	**0.51**	**0.51**	**0.51**
Cardiotocography	0.73	0.81	0.84	**0.87**	0.75	0.84	0.87	**0.89**	0.67	0.71	0.74	**0.75**
Congressional Voting	**0.78**	**0.78**	**0.78**	**0.78**	0.77	**0.78**	**0.78**	**0.78**	**0.74**	**0.74**	**0.74**	**0.74**
Contraceptive Choice	0.56	0.59	0.75	**0.76**	0.65	0.65	0.79	**0.81**	0.15	0.17	0.39	**0.47**
Credit Approval	**0.91**	**0.91**	**0.91**	0.89	**0.91**	0.9	0.9	0.88	0.88	**0.89**	**0.89**	0.88
Dermatology	0.72	**0.73**	**0.73**	**0.73**	0.7	**0.71**	**0.71**	**0.71**	0.75	**0.76**	**0.76**	**0.76**
E-coli	0.87	0.88	**0.89**	**0.89**	0.85	0.85	0.86	**0.87**	0.87	0.87	**0.88**	**0.88**
Flag	**0.72**	0.68	0.69	0.69	**0.86**	0.85	**0.86**	0.85	**0.56**	0.53	**0.56**	**0.56**
Glass	0.71	0.64	**0.74**	0.73	0.75	0.72	**0.79**	**0.79**	0.54	0.59	0.58	**0.65**
Haberman's Survival	-0.81	-0.82	-0.83	**-0.8**	**-0.86**	-0.96	-0.94	-0.92	**0.61**	0.43	0.49	0.5
Ionosphere	**0.73**	0.7	0.69	0.69	**0.74**	0.71	0.7	0.69	**0.34**	0.24	0.22	0.21
Iris	0.69	**0.73**	0.71	0.7	0.76	**0.79**	0.78	0.77	0.76	**0.8**	0.78	0.78
Mushroom	0.65	**0.71**	**0.71**	**0.71**	0.65	**0.71**	**0.71**	**0.71**	0.56	0.62	**0.63**	**0.63**
Page blocks	0.54	0.62	0.69	**0.78**	0.69	0.79	0.83	**0.86**	0.51	0.64	0.66	**0.69**
Post-op	**-0.34**	-0.53	-0.51	-0.51	**-0.15**	-0.38	-0.37	-0.37	**0.33**	0.32	0.28	0.28
Segmentation	0.72	0.69	**0.73**	0.72	0.78	0.75	**0.79**	0.77	0.63	0.61	**0.64**	0.63
Statlog (vehicle)	0.78	0.81	0.82	**0.83**	0.8	0.83	0.84	**0.85**	0.61	**0.68**	0.66	0.67
WI Breast Cancer (orig.)	**0.7**	0.68	0.67	0.67	**0.71**	0.68	0.67	0.67	0.73	**0.7**	**0.7**	0.69
WI Breast Cancer (diag.)	0.49	0.5	**0.51**	**0.51**	0.49	0.51	**0.52**	0.51	0.53	0.55	**0.54**	**0.54**
WI Breast Cancer (prog.)	**-0.78**	-0.83	-0.83	-0.83	**-0.74**	-0.81	-0.81	-0.81	0.15	**0.19**	0.18	**0.19**
Wine	**0.74**	0.73	**0.74**	**0.74**	**0.74**	0.72	**0.74**	**0.74**	0.69	0.71	0.71	**0.72**
Yeast	0.6	**0.92**	0.82	0.89	0.29	**0.64**	0.49	0.59	0.68	**0.95**	0.88	0.94
avg	0.51	0.53	0.54	0.55	0.53	0.53	0.54	0.55	0.57	0.60	0.61	0.62

Table B.18: Correlation between SVM and 1-NN accuracies and symmetric uncertainty filter values using different quantizations on real data sets

B.3.2 Multivariate measures

	mRMR														
	SVM (one vs. one)					SVM (one vs. all)					KNN				
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$		$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$		$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	
Binary 1	**0.57**	**0.57**	0.11	-0.57		0.54	**0.55**	0.15	-0.54		**0.93**	0.85	0.47	-0.93	
Binary 2	0.69	**0.70**	0.63	0.42		0.53	0.53	**0.59**	0.41		**0.97**	**0.97**	0.83	0.42	
Binary 3	-	51%	18%	27%		-	51%	18%	27%		0.00	0.39	0.37	**0.69**	
Monk 1	**0.99**	0.98	0.98	0.98		**0.99**	0.98	0.98	0.98		**0.59**	0.58	0.58	0.58	
Monk 2	94%	86%	82%	82%		94%	86%	82%	82%		**-0.45**	-0.55	-0.54	-0.54	
Monk 3	0.94	**0.95**	**0.95**	**0.95**		0.93	**0.94**	**0.94**	**0.94**		**0.98**	**0.98**	**0.98**	**0.98**	
Simple dependent	**1.00**	0.99	0.99	0.52		**1.00**	**1.00**	0.99	0.50		**1.00**	0.99	0.99	0.51	
Partially dependent	**0.99**	**0.99**	**0.99**	**0.99**		**0.99**	**0.99**	**0.99**	**0.99**		**0.99**	**0.99**	0.98	0.98	
2-way linear corr.	**-0.28**	**-0.28**	**-0.28**	**-0.28**		**0.50**	**0.50**	**0.50**	**0.50**		**0.35**	**0.35**	**0.35**	**0.35**	
3-way linear corr.	0.86	0.87	**0.97**	0.77		0.86	0.87	**0.97**	0.77		0.84	0.83	**0.94**	0.75	
3-way non-linear corr.	**0.88**	**0.88**	**0.88**	0.86		**0.88**	**0.88**	**0.88**	0.86		**0.88**	**0.88**	**0.88**	0.85	
1 good and noise	**1.00**	**1.00**	**1.00**	0.97		**0.97**	**0.97**	**0.97**	0.93		**0.99**	**0.99**	**0.99**	0.96	
Pure noise	**0.69**	0.56	0.26	-0.44		-0.66	-0.49	-0.44	**0.28**		**0.95**	0.91	0.41	-0.42	
Un-nested	0.71	0.81	0.85	**0.91**		0.76	0.87	0.89	**0.90**		0.75	0.84	0.88	**0.92**	
Monotonic	0.76	**0.80**	0.75	0.76		0.78	0.76	**0.80**	0.79		0.79	**0.82**	0.78	0.75	
Two U's	**0.99**	0.97	0.65	0.65		**1.00**	0.94	0.57	0.57		0.96	**0.99**	0.75	0.75	
Multimodal (XOR)	**0.73**	0.57	0.43	0.57		**0.84**	0.42	0.58	0.42		**0.83**	0.44	0.56	0.44	
linear sep. Guassian	**1.00**	0.99	0.90	0.92		**0.99**	0.98	0.91	0.90		**0.98**	0.97	0.94	0.87	
Redundant	0%	0%	0%	0%		**0.00**	**0.00**	**0.00**	**0.00**		0%	0%	0%	0%	
2 clusters	0%	6%	-1%	-5%		0%	6%	-1%	-5.25%		0.00%	6%	-1%	-5%	

Table B.19: Correlation between SVM and 1-NN accuracies and mRMR filter values using different quantizations on artificial data sets

	CMIM											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Binary 1	0.57	0.54	**0.59**	0.56	0.54	0.51	**0.55**	0.52	0.93	0.90	**0.94**	0.93
Binary 2	0.69	0.65	0.68	**0.70**	0.53	0.49	0.52	**0.56**	**0.97**	**0.97**	0.96	0.96
Binary 3	-	2843%	818%	651%	-	2843%	818%	651%	0.88	0.91	**0.89**	0.87
Monk 1	0.21	**0.60**	**0.60**	**0.60**	0.21	**0.60**	**0.60**	**0.60**	0.12	**0.36**	**0.36**	**0.36**
Monk 2	521%	465%	460%	460%	521%	465%	456%	460%	**-0.53**	-0.59	-0.59	-0.59
Monk 3	0.86	**0.89**	0.88	0.88	0.84	**0.88**	0.87	0.87	0.93	**0.95**	**0.95**	**0.95**
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Partially dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
2-way linear corr.	0.08	-0.21	**0.43**	0.37	0.77	0.56	**0.95**	0.93	0.65	0.41	**0.88**	0.85
3-way linear corr.	0.94	0.74	0.94	**0.99**	0.94	0.75	0.95	**0.99**	0.92	0.72	0.92	**0.98**
3-way non-linear corr.	**0.89**	**0.89**	0.88	**0.89**	**0.89**	**0.89**	0.88	**0.89**	**0.89**	**0.89**	**0.89**	**0.89**
1 good and noise	**0.99**	**0.99**	0.96	0.67	**0.97**	**0.97**	0.95	0.68	**0.98**	**0.98**	0.96	0.67
Pure noise	0.27	0.50	0.53	**0.58**	**-0.14**	-0.36	-0.37	-0.43	0.26	**0.79**	0.62	0.62
Un-nested	0.74	0.84	0.91	**0.99**	0.79	0.92	0.95	**0.96**	0.77	0.88	0.93	**0.99**
Monotonic	0.80	**0.87**	0.78	0.84	0.78	0.82	0.81	**0.83**	0.82	**0.88**	0.82	0.87
Two U's	0.99	0.99	**1.00**	**1.00**	**1.00**	0.98	**1.00**	0.99	0.96	**1.00**	0.99	0.99
Multimodal (XOR)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
linear sep. Guassian	**1.00**	**1.00**	0.91	0.96	**1.00**	**1.00**	0.92	0.97	**1.00**	0.99	0.95	0.99
Redundant	0%	0%	0%	0%	**0.00**	**0.00**	**0.00**	**0.00**	0%	0%	0%	0%
2 clusters	0%	6%	0%	0%	0%	6%	0%	0%	0%	6%	0%	0%

Table B.20: Correlation between SVM and 1-NN accuracies and CMIM filter values using different quantizations on artificial data sets

	FOU											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Binary 1	**0.68**	0.67	0.60	0.37	**0.65**	**0.65**	0.58	0.29	**0.96**	0.94	0.94	0.59
Binary 2	**0.79**	0.77	0.77	0.50	**0.65**	0.63	0.64	0.39	0.99	**1.00**	0.99	0.85
Binary 3	-	2216%	729%	30%	-	2216%	729%	30%	0.92	0.91	**0.95**	0.73
Monk 1	0.41	**0.64**	**0.64**	**0.64**	0.41	**0.64**	**0.64**	**0.64**	0.73	**0.79**	**0.79**	**0.79**
Monk 2	888%	869%	855%	855%	888%	869%	855%	855%	**-0.49**	-0.55	-0.55	-0.55
Monk 3	0.80	**0.82**	0.81	0.81	0.78	**0.80**	**0.80**	**0.80**	0.89	**0.90**	**0.90**	**0.90**
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Partially dependent	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.98
2-way linear corr.	**-0.28**	**-0.28**	**-0.28**	**-0.28**	**0.50**	**0.50**	**0.50**	**0.50**	**0.35**	**0.35**	**0.35**	**0.35**
3-way linear corr.	0.90	0.63	0.82	**0.98**	0.91	0.64	0.82	**0.98**	0.89	0.63	0.80	**0.97**
3-way non-linear corr.	0.79	0.80	0.80	**0.81**	0.79	0.80	0.80	**0.81**	0.79	0.80	0.80	**0.81**
1 good and noise	0.97	**0.98**	0.95	0.88	0.95	**0.96**	0.94	0.87	**0.96**	**0.96**	0.93	0.85
Pure noise	0.49	0.44	0.55	**0.62**	**-0.24**	-0.25	-0.32	-0.41	0.50	**0.74**	0.59	0.64
Un-nested	0.73	0.79	0.82	**0.87**	0.75	0.82	0.84	**0.87**	0.76	0.82	0.85	**0.88**
Monotonic	0.86	0.87	0.87	**0.94**	0.78	0.74	0.88	**0.91**	0.85	0.87	0.90	**0.92**
Two U's	0.99	**1.00**	**1.00**	0.94	**1.00**	0.99	**1.00**	0.90	0.95	**0.99**	**0.99**	0.98
Multimodal (XOR)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
linear sep. Guassian	**0.99**	**0.99**	0.91	0.96	0.99	**1.00**	0.92	0.97	**1.00**	**1.00**	0.95	0.99
Redundant	0%	0%	0%	0%	**0.00**	**0.00**	**0.00**	**0.00**	0%	0%	0%	0%
2 clusters	0%	10%	0%	0%	0%	10%	0%	0%	0%	10%	0%	0%

Table B.21: Correlation between SVM and 1-NN accuracies and FOU filter values using different quantizations on artificial data sets

	mRMR											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	**0.07**	-0.26	-0.40	-0.36	**0.24**	0.07	-0.20	-0.17	**0.23**	-0.24	-0.45	-0.47
Car Evaluation	**0.79**	**0.79**	**0.79**	**0.79**	**0.74**	**0.74**	0.73	0.73	**0.51**	**0.51**	**0.51**	**0.51**
Cardiotocography	0.71	0.83	0.86	**0.88**	0.73	0.85	0.88	**0.90**	**0.67**	0.72	0.74	0.74
Congressional Voting	**0.84**	0.83	0.83	0.83	**0.84**	0.83	0.83	0.83	**0.82**	0.81	0.81	0.81
Contraceptive Choice	**0.27**	-0.12	0.04	0.10	**0.07**	-0.30	-0.14	-0.03	-0.09	-0.22	-0.01	**0.10**
Credit Approval	**0.93**	0.92	0.92	0.89	**0.93**	0.92	0.92	0.90	**0.87**	0.83	0.85	0.80
Dermatology	0.74	**0.76**	0.75	0.75	0.72	**0.74**	0.73	0.73	0.78	**0.80**	0.79	0.79
E-coli	0.89	0.92	**0.95**	0.94	0.88	0.90	0.92	**0.93**	0.89	0.90	**0.94**	0.93
Flag	0.75	0.74	0.77	**0.79**	0.87	**0.89**	**0.89**	0.88	0.60	0.61	0.66	**0.71**
Glass	0.69	0.56	**0.71**	0.40	0.76	0.64	**0.78**	0.50	**0.53**	0.49	0.44	0.21
Haberman's Survival	**-0.76**	-0.83	-0.89	-0.88	**-0.83**	-0.95	-0.94	-0.94	**0.82**	0.40	0.52	0.57
Ionosphere	**0.01**	-0.24	-0.43	-0.50	**0.00**	-0.25	-0.44	-0.51	**0.20**	0.05	-0.04	-0.07
Iris	0.74	0.97	**0.98**	0.97	0.82	**0.96**	0.95	0.92	0.76	**0.99**	**0.99**	0.98
Mushroom	0.34	**0.49**	0.28	0.11	0.35	**0.49**	0.28	0.11	0.32	**0.48**	0.27	0.11
Page blocks	0.50	**0.57**	0.56	0.36	0.65	**0.73**	0.71	0.43	0.48	**0.58**	0.55	0.25
Post-op	**-0.20**	-0.39	-0.44	-0.44	**0.04**	-0.19	-0.32	-0.32	**0.45**	-0.07	-0.23	-0.23
Segmentation	0.76	0.76	0.78	**0.80**	0.81	0.81	**0.84**	0.83	0.67	0.68	0.70	**0.74**
Statlog (vehicle)	**-0.12**	0.19	-0.39	-0.54	-0.15	**0.18**	-0.42	-0.56	-0.27	**0.08**	-0.47	-0.55
WI Breast Cancer (orig.)	0.72	**0.74**	0.72	0.25	0.72	**0.74**	0.72	0.25	**0.73**	**0.73**	0.70	0.14
WI Breast Cancer (diag.)	0.54	**0.62**	0.58	-0.21	0.54	0.62	**0.58**	-0.22	0.58	**0.64**	0.58	-0.24
WI Breast Cancer (prog.)	0.67	0.71	0.74	**0.79**	0.66	0.69	0.72	**0.77**	**-0.10**	**-0.10**	-0.12	-0.16
Wine	0.76	0.77	**0.79**	0.75	0.76	0.77	**0.79**	0.74	0.72	0.76	**0.77**	0.75
Yeast	0.61	**0.94**	0.83	0.90	0.30	**0.68**	0.49	0.59	0.68	**0.95**	0.90	**0.95**
avg	0.49	0.49	0.45	0.36	0.50	0.50	0.45	0.36	0.52	0.49	0.45	0.36

Table B.22: Correlation between SVM and 1-NN accuracies and mRMR filter values using different quantizations on real data sets

	CMIM											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	0.19	0.32	0.48	**0.60**	**0.24**	0.16	0.20	0.17	0.34	0.62	0.69	**0.76**
Car Evaluation	**0.81**	**0.81**	**0.81**	**0.81**	**0.77**	0.76	0.75	0.75	**0.53**	0.52	**0.53**	**0.53**
Cardiotocography	0.75	0.85	0.88	**0.89**	0.76	0.87	0.89	**0.90**	0.73	0.77	0.79	**0.80**
Congressional Voting	**0.91**	0.88	0.88	0.88	**0.91**	0.88	0.88	0.88	**0.87**	0.84	0.84	0.84
Contraceptive Choice	0.65	0.74	0.84	**0.87**	0.66	0.74	**0.78**	0.76	0.18	0.27	0.47	**0.60**
Credit Approval	0.93	0.93	**0.94**	0.91	0.93	0.93	**0.94**	0.91	0.87	0.88	**0.91**	**0.91**
Dermatology	**0.83**	**0.83**	0.82	0.82	**0.82**	0.81	0.80	0.80	**0.87**	0.86	0.85	0.85
E-coli	0.92	0.94	**0.97**	0.94	0.91	0.91	**0.94**	0.92	0.92	0.92	**0.95**	0.92
Flag	**0.73**	0.68	0.68	0.65	**0.88**	0.86	0.85	0.82	**0.57**	0.56	0.56	0.53
Glass	0.69	0.63	**0.76**	0.72	0.76	0.69	**0.82**	0.75	0.56	0.64	0.69	**0.76**
Haberman's Survival	**0.19**	-0.59	-0.59	-0.32	**0.22**	-0.87	-0.80	-0.59	0.08	0.33	**0.41**	0.31
Ionosphere	**0.77**	0.59	0.61	0.63	**0.78**	0.60	0.61	0.63	**0.35**	0.20	0.16	0.14
Iris	0.91	**0.98**	**0.98**	**0.98**	0.95	**0.98**	**0.98**	**0.98**	0.92	**0.99**	**0.99**	0.98
Mushroom	0.61	**0.81**	0.79	0.76	0.61	**0.81**	0.79	0.76	0.58	0.79	0.81	**0.83**
Page blocks	0.49	0.57	0.61	**0.68**	0.63	0.73	0.76	**0.82**	0.45	0.55	0.58	**0.64**
Post-op	-0.29	**-0.10**	-0.13	-0.13	-0.11	**0.11**	0.03	0.03	**0.57**	0.50	0.42	0.42
Segmentation	0.83	**0.87**	0.83	0.84	0.87	**0.90**	0.88	0.87	0.75	**0.82**	0.75	0.76
Statlog (vehicle)	0.81	0.87	**0.89**	0.88	0.79	0.89	**0.90**	0.89	0.63	**0.76**	0.73	0.73
WI Breast Cancer (orig.)	0.75	**0.81**	0.80	0.78	0.75	**0.81**	0.80	0.78	0.76	**0.79**	0.78	0.76
WI Breast Cancer (diag.)	0.57	**0.72**	**0.72**	0.71	0.58	**0.72**	**0.72**	**0.72**	0.62	**0.76**	0.74	0.72
WI Breast Cancer (prog.)	**-0.75**	-0.81	-0.81	-0.81	**-0.73**	-0.80	-0.80	-0.81	0.22	**0.23**	0.20	0.22
Wine	0.79	**0.83**	**0.83**	0.78	0.80	0.83	**0.84**	0.79	0.75	**0.82**	0.81	0.75
Yeast	0.62	**0.95**	0.86	0.92	0.30	**0.68**	0.53	0.61	0.70	**0.96**	0.92	**0.96**
avg	0.60	0.61	0.63	0.64	0.61	0.61	0.61	0.61	0.60	0.67	0.68	0.68

Table B.23: Correlation between SVM and 1-NN accuracies and CMIM filter values using different quantizations on real data sets

	FOU											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	0.13	0.11	0.19	**0.42**	**0.23**	0.15	0.04	-0.03	0.34	0.44	0.37	**0.49**
Car Evaluation	0.87	**0.88**	0.85	0.85	0.81	**0.82**	0.78	0.78	**0.61**	**0.61**	0.60	0.60
Cardiotocography	0.71	0.75	0.76	**0.77**	0.73	0.78	0.78	**0.79**	0.61	0.62	0.63	**0.64**
Congressional Voting	**0.74**	0.70	0.70	0.70	**0.74**	0.70	0.70	0.70	**0.71**	0.68	0.68	0.68
Contraceptive Choice	0.62	0.58	0.63	**0.70**	0.48	0.45	0.63	**0.68**	0.15	0.12	0.28	**0.38**
Credit Approval	**0.93**	0.92	0.91	0.87	**0.92**	**0.92**	0.91	0.87	0.89	0.89	**0.91**	0.88
Dermatology	0.61	**0.62**	0.61	0.61	0.59	**0.60**	0.59	0.59	**0.66**	**0.66**	**0.66**	**0.66**
E-coli	0.81	0.82	**0.87**	0.85	0.81	0.81	**0.85**	0.84	0.83	0.83	**0.88**	0.86
Flag	**0.53**	**0.53**	0.52	0.52	0.75	**0.76**	0.75	0.75	**0.38**	**0.38**	0.37	**0.38**
Glass	**0.70**	0.64	**0.70**	0.68	0.73	0.69	**0.74**	0.71	0.54	0.60	0.56	**0.67**
Haberman's Survival	**-0.01**	-0.63	-0.65	-0.76	**-0.08**	-0.96	-0.86	-0.98	0.16	0.31	0.42	**0.48**
Ionosphere	0.68	0.70	**0.72**	**0.72**	0.69	0.71	**0.72**	**0.72**	0.30	0.26	0.35	**0.37**
Iris	0.61	0.94	0.97	**0.99**	0.72	**0.98**	**0.98**	0.97	0.64	0.97	0.97	**0.98**
Mushroom	0.58	**0.65**	0.63	0.62	0.58	**0.65**	0.63	0.62	0.49	0.54	0.55	**0.59**
Page blocks	0.55	0.60	0.62	**0.72**	0.71	0.78	0.80	**0.85**	0.48	0.59	0.58	**0.64**
Post-op	-0.35	**-0.28**	-0.35	-0.35	-0.11	**-0.04**	-0.16	-0.16	**0.54**	0.45	0.42	0.42
Segmentation	0.66	0.68	**0.70**	0.68	0.74	0.75	**0.77**	0.75	0.56	0.58	**0.60**	0.59
Statlog (vehicle)	0.67	**0.77**	0.66	0.59	0.65	**0.79**	0.67	0.61	0.43	**0.58**	0.42	0.34
WI Breast Cancer (orig.)	**0.67**	0.59	0.56	0.61	**0.68**	0.59	0.56	0.60	**0.71**	0.56	0.49	0.50
WI Breast Cancer (diag.)	0.54	0.57	**0.60**	0.57	0.54	0.57	**0.61**	0.57	0.57	0.58	**0.60**	0.58
WI Breast Cancer (prog.)	-0.72	-0.66	-0.67	**-0.29**	-0.70	-0.68	-0.64	**-0.27**	0.20	**0.26**	0.16	0.11
Wine	**0.68**	0.65	0.65	0.62	**0.68**	0.66	0.67	0.63	0.63	**0.64**	0.63	0.60
Yeast	0.69	**0.93**	0.87	0.90	0.44	**0.71**	0.60	0.65	0.73	**0.93**	0.90	0.92
avg	0.52	0.52	0.52	0.55	0.54	0.53	0.53	0.53	0.53	0.57	0.57	0.58

Table B.24: Correlation between SVM and 1-NN accuracies and FOU filter values using different quantizations on real data sets

B.4 Consistency

	consistency																	
	SVM (one vs. one)						SVM (one vs. all)						KNN					
	$N_b = 2$	3	4	5	10		$N_b = 2$	3	4	5	10		$N_b = 2$	3	4	5	10	
Binary 1	**0.57**	0.49	0.56	0.55	0.42		**0.54**	0.45	0.53	0.52	0.38		**0.93**	0.87	**0.93**	**0.93**	0.86	
Binary 2	**0.70**	0.65	0.65	0.64	0.52		**0.54**	0.49	0.48	0.47	0.34		**0.97**	**0.97**	0.96	0.95	0.89	
Binary 3	-	-	-	-	-		-	-	-	-	-		-	-	-	-	-	
Monk 1	0.28	**0.59**	**0.59**	**0.59**	**0.59**		0.27	**0.58**	**0.58**	**0.58**	**0.58**		0.85	**0.91**	**0.91**	**0.91**	**0.91**	
Monk 2	-	-	-	-	-		-	-	-	-	-		**-0.19**	-0.21	-0.21	-0.21	-0.21	
Monk 3	0.61	**0.70**	**0.70**	**0.70**	**0.70**		0.59	**0.69**	**0.69**	**0.69**	**0.69**		0.73	**0.81**	**0.81**	**0.81**	**0.81**	
Simple dependent	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**		**1.00**	**1.00**	**1.00**	**1.00**	0.99		**1.00**	**1.00**	**1.00**	**1.00**	0.99	
Partially dependent	0.63	0.99	0.93	**1.00**	0.99		0.63	0.99	0.93	**1.00**	0.99		0.66	0.98	0.95	**1.00**	**1.00**	
2-way linear corr.	0.05	-0.21	0.82	0.32	**0.87**		0.76	0.55	**0.98**	0.90	0.96		0.63	0.40	**1.00**	0.82	0.99	
3-way linear corr.	0.93	0.85	0.95	0.96	**0.98**		0.93	0.86	0.95	0.96	**0.98**		0.93	0.86	0.95	0.96	**0.98**	
3-way non-linear corr.	**1.00**	0.99	**1.00**	**1.00**	**1.00**		**1.00**	0.99	**1.00**	**1.00**	0.99		**1.00**	0.99	**1.00**	**1.00**	**1.00**	
1 good and noise	**0.99**	0.98	0.96	0.92	0.71		**0.97**	0.96	0.95	0.91	0.70		**0.98**	0.97	0.95	0.91	0.71	
Pure noise	0.49	0.42	0.46	**0.51**	0.50		**-0.16**	-0.18	-0.22	-0.31	-0.44		0.39	0.63	0.55	**0.57**	**0.57**	
Un-nested	0.61	0.54	0.78	0.95	**0.99**		0.73	0.71	0.88	**0.98**	0.97		0.66	0.60	0.82	0.97	**0.98**	
Monotonic	0.89	0.83	0.88	0.89	**0.97**		**0.89**	0.83	0.86	0.86	0.87		0.90	0.89	0.92	0.91	**0.96**	
Two U's	0.98	0.88	0.97	**1.00**	0.93		**1.00**	0.83	0.94	0.99	0.89		0.95	0.94	**1.00**	**1.00**	0.97	
Multimodal (XOR)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**		**1.00**	**1.00**	**1.00**	**1.00**	**1.00**		**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	
linear sep. Guassian	0.92	**0.99**	0.93	0.92	0.93		0.93	**0.99**	0.94	0.94	0.94		0.96	**0.97**	0.96	0.96	**0.97**	
Redundant	0%	0%	0%	0%	0%		**0.00**	**0.00**	**0.00**	**0.00**	**0.00**		0%	0%	0%	0%	0%	
2 clusters	0%	7%	0%	0%	0%		0%	7%	0%	0%	0%		0%	7%	0%	0%	0%	

Table B.25: Correlation between SVM and 1-NN accuracies and consistency filter values using different quantizations on artificial data sets

	consistency														
	SVM (one vs. one)					SVM (one vs. all)					KNN				
	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 4$	$N_b = 5$	$N_b = 10$
Abalone	0.32	0.25	0.44	0.42	**0.48**	**0.23**	0.16	0.17	0.16	0.15	0.48	0.61	0.69	0.72	**0.77**
Car Evaluation	0.64	**0.67**	0.64	0.64	0.64	0.59	**0.60**	0.57	0.57	0.57	0.40	**0.41**	0.40	0.40	0.40
Cardiotocography	0.74	0.83	0.86	**0.87**	0.80	0.75	0.85	**0.87**	**0.87**	0.78	0.72	0.76	0.84	0.89	**0.93**
Congressional Voting	0.77	**0.76**	**0.76**	**0.76**	**0.76**	**0.76**	0.75	0.75	0.75	0.75	**0.71**	0.70	0.70	0.70	0.70
Contraceptive Choice	0.46	0.63	0.66	0.70	**0.80**	0.41	0.55	0.58	0.61	**0.67**	0.16	0.26	0.29	0.34	**0.46**
Credit Approval	0.56	**0.64**	**0.64**	0.63	0.61	0.56	**0.64**	0.63	0.63	0.61	0.65	0.74	0.74	0.74	**0.76**
Dermatology	0.92	**0.94**	0.89	0.89	0.88	0.92	**0.95**	0.91	0.91	0.90	0.93	**0.94**	0.88	0.87	0.86
E-coli	0.80	0.82	0.87	0.93	**0.94**	0.72	0.79	0.80	**0.91**	0.90	0.77	0.83	0.83	**0.90**	0.87
Flag	**0.67**	**0.67**	0.65	0.64	0.60	**0.83**	0.79	0.75	0.73	0.67	0.51	0.56	0.57	**0.58**	0.57
Glass	0.67	0.54	0.64	0.67	**0.72**	0.70	0.62	0.71	**0.75**	0.74	0.57	0.51	0.55	0.69	**0.76**
Haberman's Survival	-0.45	-0.14	**-0.09**	-0.18	-0.18	-0.35	-0.34	**-0.17**	-0.28	-0.34	**0.26**	0.16	0.15	0.18	0.15
Ionosphere	0.54	0.61	**0.63**	0.61	0.52	0.54	0.62	**0.63**	0.60	0.51	0.21	0.35	0.51	0.51	**0.60**
Iris	0.94	0.96	**0.99**	0.94	0.95	0.96	0.95	0.97	**0.98**	0.97	0.95	0.98	**0.99**	0.97	0.96
Mushroom	0.54	**0.61**	0.60	0.56	0.53	0.55	0.61	0.60	**0.56**	0.54	0.45	0.66	0.69	0.71	**0.72**
Page blocks	0.50	**0.59**	0.54	0.56	0.57	0.55	0.71	0.64	0.65	**0.68**	0.45	**0.51**	0.43	0.45	0.48
Post-op	-0.01	**0.00**	**0.00**	**0.00**	**0.00**	**0.33**	0.23	0.23	0.23	0.23	**0.66**	0.54	0.54	0.54	0.54
Segmentation	0.88	0.90	0.93	0.94	**0.95**	0.91	0.92	0.93	**0.96**	0.91	0.83	0.86	0.89	0.92	**0.97**
Statlog (vehicle)	0.80	0.85	0.88	**0.90**	0.83	0.79	0.86	0.89	**0.90**	0.83	0.64	0.64	0.67	0.69	**0.74**
WI Breast Cancer (orig.)	0.61	**0.72**	0.65	0.68	0.71	0.61	**0.72**	0.65	0.69	**0.72**	0.63	**0.77**	0.70	0.74	0.73
WI Breast Cancer (diag.)	0.59	0.65	0.71	0.77	**0.83**	0.60	0.66	0.72	0.77	**0.84**	0.64	0.67	0.74	0.80	**0.84**
WI Breast Cancer (prog.)	-0.79	-0.66	-0.52	-0.42	**-0.20**	-0.78	-0.64	-0.50	-0.41	**-0.19**	**0.19**	0.16	0.13	0.10	0.01
Wine	0.81	0.84	0.91	**0.93**	0.86	0.82	0.83	0.91	**0.93**	0.86	0.77	0.82	0.88	**0.92**	0.87
Yeast	0.60	0.78	0.75	0.77	**0.82**	0.35	**0.59**	0.46	0.46	0.52	0.68	0.81	0.82	0.85	**0.90**

Table B.26: Correlation between SVM and 1-NN accuracies and consistency filter values using different quantizations on real data sets

B.5 Laplacian score

	Laplacian Score ($k = 25$)															
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$
Binary 1	**0.59**	0.58	0.58	0.58	**0.54**	0.53	0.53	0.53	**0.94**	**0.94**	**0.94**	**0.94**
Binary 2	**0.70**	**0.70**	**0.70**	**0.70**	**0.56**	**0.56**	**0.56**	**0.56**	0.93	**0.94**	**0.94**	**0.94**
Binary 3	100%	100%	100%	100%	100%	100%	100%	100%	0.84	**0.89**	**0.89**	**0.89**
Monk 1	**0.75**	0.50	0.50	0.50	**0.74**	0.49	0.48	0.48	**0.70**	0.64	0.64	0.64
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%	**-0.54**	-0.55	-0.55	-0.55
Monk 3	**0.70**	0.64	0.64	0.64	**0.71**	0.66	0.66	0.66	**0.76**	0.69	0.69	0.69
Simple dependent	**0.89**	0.85	0.84	0.84	**0.90**	0.86	0.86	0.86	**0.89**	0.86	0.85	0.85
Partially dependent	0.79	**0.80**	**0.80**	**0.80**	0.79	**0.80**	**0.80**	**0.80**	0.81	**0.82**	**0.82**	**0.82**
2-way linear corr.	0.79	**0.80**	**0.80**	**0.80**	**0.99**	**0.99**	**0.99**	**0.99**	**1.00**	**1.00**	**1.00**	**1.00**
3-way linear corr.	0.38	0.40	**0.41**	**0.41**	0.41	0.43	**0.44**	**0.44**	0.34	**0.36**	**0.36**	**0.36**
3-way non-linear corr.	**0.69**	0.68	0.68	0.68	**0.70**	0.69	0.69	0.69	**0.74**	0.73	0.73	0.73
1 good and noise	**0.42**	**0.42**	0.41	0.41	**0.40**	**0.40**	0.39	0.39	**0.40**	**0.40**	0.39	0.39
Pure noise	**0.43**	0.38	0.38	0.38	-0.24	-0.19	**-0.18**	**-0.18**	**0.44**	0.38	0.38	0.38
Un-nested	0.70	**0.76**	**0.76**	**0.76**	0.60	**0.72**	**0.72**	**0.72**	0.67	**0.75**	**0.75**	**0.75**
Monotonic	**0.80**	**0.80**	**0.80**	**0.80**	**0.81**	**0.81**	**0.81**	**0.81**	0.83	**0.84**	**0.84**	**0.84**
Two U's	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.99**	0.98	0.98	0.98
Multimodal (XOR)	**0.99**	**0.99**	**0.99**	**0.99**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
linear sep. Guassian	**0.99**	**0.99**	**0.99**	**0.99**	**1.00**	0.99	0.99	0.99	**1.00**	**1.00**	**1.00**	**1.00**
Redundant	100%	100%	100%	100%	**-0.65**	**-0.65**	**-0.65**	**-0.65**	100%	100%	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table B.27: Correlation between SVM and 1-NN accuracies and Laplacian score using different t values and $k = 25$ on artificial data sets

	Laplacian Score ($t = 1$)								
	SVM (one vs. one)								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Binary 1	0.51	0.57	**0.59**	**0.59**	0.58	0.57	**0.59**	0.58	0.57
Binary 2	0.68	**0.70**	**0.70**	**0.70**	**0.70**	0.69	0.68	**0.70**	0.69
Binary 3	100%	100%	100%	100%	100%	100%	100%	100%	100%
Monk 1	**0.89**	0.63	0.40	0.48	0.50	0.53	0.54	0.48	0.43
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.61	0.57	0.56	0.64	0.64	**0.66**	**0.66**	0.64	0.62
Simple dependent	0.60	0.66	0.95	**0.96**	0.85	0.90	**0.96**	0.85	0.90
Partially dependent	0.71	0.58	0.52	**0.86**	0.80	0.83	0.82	0.83	0.75
2-way linear corr.	**1.00**	0.99	0.90	**1.00**	0.80	0.73	0.69	0.73	0.71
3-way linear corr.	**0.63**	0.49	0.38	0.42	0.40	0.40	0.42	0.40	0.40
3-way non-linear corr.	0.49	0.66	0.62	0.60	0.68	0.77	0.62	0.77	**0.87**
1 good and noise	0.15	0.26	0.41	0.39	0.42	**0.49**	0.39	0.42	**0.49**
Pure noise	0.39	0.38	0.41	**0.42**	0.38	0.38	**0.42**	0.38	0.38
Un-nested	0.54	0.54	0.54	0.54	**0.76**	0.45	0.54	0.76	0.45
Monotonic	0.72	0.77	0.66	0.75	0.80	**0.81**	0.75	0.80	**0.81**
Two U's	0.82	0.95	0.98	**1.00**	**1.00**	0.99	**1.00**	0.98	0.99
Multimodal (XOR)	0.96	0.84	0.92	0.97	**0.99**	0.95	**0.99**	0.95	0.93
linear sep. Guassian	0.45	0.86	0.92	0.90	0.99	**1.00**	0.99	**1.00**	0.99
Redundant	100%	100%	-	-	100%	100%	-	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%
	Laplacian Score ($t = 1$)								
	SVM (one vs. all)								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Binary 1	0.45	0.51	0.54	0.54	0.53	0.54	**0.55**	0.53	0.54
Binary 2	0.54	0.55	**0.56**	**0.56**	**0.56**	0.53	**0.56**	**0.56**	0.54
Binary 3	100%	100%	100%	100%	100%	100%	100%	100%	100%
Monk 1	**0.89**	0.62	0.39	0.46	0.49	0.52	0.53	0.46	0.42
Monk 2	100%	100%	100%	100%	100%	100%	100%	100%	100%
Monk 3	0.60	0.58	0.59	0.65	0.66	**0.68**	0.67	0.65	0.64
Simple dependent	0.62	0.68	0.96	**0.97**	0.86	0.92	**0.97**	0.86	0.92
Partially dependent	0.71	0.59	0.52	**0.87**	0.80	0.83	0.82	0.83	0.75
2-way linear corr.	0.74	0.81	0.30	0.69	0.99	**1.00**	**1.00**	**1.00**	**1.00**
3-way linear corr.	**0.65**	0.52	0.41	0.45	0.43	0.43	0.45	0.43	0.43
3-way non-linear corr.	0.51	0.67	0.62	0.61	0.69	0.78	0.63	0.78	**0.87**
1 good and noise	0.13	0.24	0.38	0.37	0.40	**0.47**	0.37	0.40	**0.47**
Pure noise	-0.25	**-0.16**	-0.19	-0.23	-0.19	-0.18	-0.23	-0.19	-0.18
Un-nested	0.37	0.37	0.37	0.37	**0.72**	0.43	0.37	**0.72**	0.43
Monotonic	0.63	0.71	0.73	**0.81**	**0.81**	0.80	**0.81**	**0.81**	0.80
Two U's	0.87	0.97	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.95	**1.00**
Multimodal (XOR)	**1.00**	0.92	0.97	**1.00**	**1.00**	0.99	**1.00**	0.99	0.98
linear sep. Guassian	0.47	0.87	0.94	0.92	0.99	**1.00**	0.99	**1.00**	**1.00**
Redundant	-0.65	-0.65	**0.66**	0.65	-0.65	-0.65	0.65	-0.65	-0.65
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table B.28: Correlation between SVM accuracies and Laplacian score using different k values and $t = 1$ on artificial data sets

	Laplacian Score ($t = 1$)								
	KNN								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Binary 1	0.84	0.91	**0.94**	0.93	**0.94**	0.93	0.92	**0.94**	**0.94**
Binary 2	0.83	0.87	0.90	0.94	0.94	**0.97**	0.90	0.92	**0.97**
Binary 3	0.74	0.78	0.86	0.75	0.89	**0.90**	0.80	0.88	**0.90**
Monk 1	0.73	**0.74**	0.63	0.64	0.64	0.64	0.64	0.57	0.52
Monk 2	**-0.43**	-0.50	-0.53	-0.55	-0.55	-0.57	-0.58	-0.63	-0.63
Monk 3	0.66	0.61	0.60	0.69	0.69	**0.72**	0.71	0.69	0.67
Simple dependent	0.61	0.68	**0.96**	**0.96**	0.86	0.91	**0.96**	0.86	0.91
Partially dependent	0.73	0.62	0.56	**0.88**	0.82	0.85	0.85	0.85	0.78
2-way linear corr.	0.84	0.90	0.46	0.80	**1.00**	0.99	0.99	0.99	0.99
3-way linear corr.	**0.59**	0.45	0.33	0.38	0.36	0.35	0.37	0.35	0.36
3-way non-linear corr.	0.56	0.71	0.66	0.66	0.73	0.81	0.68	0.81	**0.88**
1 good and noise	0.13	0.24	0.39	0.38	0.40	**0.47**	0.38	0.40	**0.47**
Pure noise	0.28	0.45	**0.48**	0.45	0.38	0.39	0.45	0.38	0.39
Un-nested	0.50	0.50	0.50	0.50	**0.75**	0.48	0.50	0.75	0.48
Monotonic	0.76	0.82	0.69	0.77	**0.84**	**0.84**	0.77	**0.84**	**0.84**
Two U's	0.72	0.89	0.94	0.98	0.98	0.96	0.97	**1.00**	0.97
Multimodal (XOR)	0.99	0.91	0.97	0.99	**1.00**	0.99	**1.00**	0.99	0.97
linear sep. Guassian	0.54	0.91	0.96	0.95	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Redundant	100%	100%	-	-	100%	100%	-	100%	100%
2 clusters	100%	100%	100%	100%	100%	100%	100%	100%	100%

Table B.29: Correlation between 1-NN accuracies and Laplacian score using different k values and $t = 1$ on artificial data sets

	Laplacian Score ($k = 25$)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$
Abalone	**0.63**	0.61	0.58	0.58	0.16	0.18	**0.19**	**0.19**	**0.70**	0.68	0.65	0.65
Car Evaluation	0.33	0.46	**0.47**	**0.47**	0.28	**0.42**	**0.42**	**0.42**	0.21	0.29	**0.30**	**0.30**
Cardiotocography	**0.67**	0.66	0.65	0.65	**0.70**	0.69	0.68	0.68	**0.54**	0.52	0.51	0.50
Congressional Voting	0.41	0.50	**0.56**	**0.56**	0.40	0.49	0.55	**0.56**	0.37	0.44	**0.50**	**0.50**
Contraceptive Choice	0.70	**0.72**	0.69	0.68	**0.59**	0.57	0.55	0.54	**0.59**	0.53	0.49	0.48
Credit Approval	0.44	0.48	**0.49**	**0.49**	0.44	0.48	0.48	**0.49**	0.54	0.61	**0.62**	**0.62**
Dermatology	0.65	0.68	0.68	**0.69**	0.63	0.66	**0.67**	**0.67**	0.68	**0.72**	**0.72**	**0.72**
E-coli	0.02	**0.04**	**0.04**	**0.04**	0.02	**0.04**	**0.04**	**0.04**	0.03	**0.06**	**0.06**	**0.06**
Flag	0.42	**0.51**	**0.51**	**0.51**	0.56	0.71	**0.72**	**0.72**	0.29	**0.33**	**0.33**	**0.33**
Glass	0.33	**0.37**	**0.37**	**0.37**	0.41	**0.44**	**0.44**	**0.44**	0.22	0.27	**0.28**	**0.28**
Haberman's Survival	**-0.85**	**-0.85**	**-0.85**	**-0.85**	**-0.94**	**-0.94**	**-0.94**	**-0.94**	**0.50**	**0.50**	**0.50**	**0.50**
Ionosphere	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Iris	**0.26**	0.23	0.23	0.23	**0.35**	0.32	0.31	0.31	**0.30**	0.27	0.27	0.27
Mushroom	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Page blocks	**0.48**	0.44	0.44	0.44	**0.54**	0.51	0.51	0.51	**0.41**	0.39	0.39	0.39
Post-op	-0.76	-0.73	**-0.72**	**-0.72**	-0.80	-0.75	**-0.72**	**-0.72**	0.28	0.27	**0.29**	**0.29**
Segmentation	**0.35**	**0.35**	0.34	0.34	0.39	**0.40**	0.39	0.39	**0.25**	0.24	0.24	0.24
Statlog (vehicle)	0.76	0.78	**0.79**	0.78	0.79	0.80	**0.81**	**0.81**	0.49	0.51	**0.54**	**0.54**
WI Breast Cancer (orig.)	**0.43**	0.33	0.33	0.33	**0.43**	0.32	0.32	0.32	**0.44**	0.34	0.33	0.33
WI Breast Cancer (diag.)	**0.47**	**0.47**	**0.47**	**0.47**	**0.48**	**0.48**	0.47	0.47	**0.47**	**0.47**	**0.47**	**0.47**
WI Breast Cancer (prog.)	**-0.84**	**-0.84**	**-0.84**	**-0.84**	-0.83	-0.83	**-0.82**	**-0.82**	**0.21**	**0.21**	**0.21**	**0.21**
Wine	**0.60**	0.56	0.56	0.55	**0.61**	0.58	0.57	0.57	**0.58**	0.55	0.55	0.55
Yeast	**0.80**	0.21	0.16	0.15	**0.74**	0.13	0.07	0.07	**0.71**	0.19	0.14	0.14

Table B.30: Correlation between SVM and 1-NN accuracies and Laplacian score using different t values and $k = 25$ on real data sets

	Laplacian Score ($t = 1$)								
	SVM (one vs. one)								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Abalone	0.44	0.48	0.52	0.57	0.61	0.62	0.58	0.54	0.30
Car Evaluation	0.15	0.13	0.14	0.26	0.46	0.52	0.56	0.56	0.57
Cardiotocography	0.68	0.67	0.68	0.66	0.66	0.67	0.69	0.71	0.72
Congressional Voting	0.57	0.57	0.55	0.52	0.50	0.49	0.49	0.49	0.50
Contraceptive Choice	0.74	0.74	0.75	0.74	0.72	0.69	0.49	0.43	0.41
Credit Approval	0.45	0.46	0.46	0.47	0.48	0.48	0.49	0.50	0.52
Dermatology	0.69	0.68	0.68	0.68	0.68	0.68	0.68	0.69	0.70
E-coli	0.01	0.01	0.01	0.02	0.04	0.05	0.05	0.08	0.14
Flag	0.49	0.49	0.50	0.50	0.51	0.51	0.50	0.51	0.51
Glass	0.37	0.38	0.37	0.36	0.37	0.37	0.36	0.37	0.33
Haberman's Survival	-0.87	-0.86	-0.86	-0.86	-0.85	-0.84	-0.85	-0.83	-0.81
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.25	0.26	0.29	0.24	0.23	0.29	0.26	0.27	0.16
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.32	0.33	0.37	0.41	0.44	0.47	0.54	0.56	0.57
Post-op	-0.70	-0.77	-0.78	-0.76	-0.73	-0.71	-0.76	-0.74	-0.72
Segmentation	0.34	0.34	0.34	0.35	0.35	0.35	0.36	0.43	0.48
Statlog (vehicle)	0.77	0.77	0.77	0.77	0.78	0.79	0.79	0.80	0.82
WI Breast Cancer (orig.)	0.28	0.30	0.30	0.30	0.33	0.36	0.38	0.45	0.53
WI Breast Cancer (diag.)	0.46	0.46	0.46	0.46	0.47	0.47	0.47	0.48	0.49
WI Breast Cancer (prog.)	-0.83	-0.83	-0.84	-0.84	-0.84	-0.84	-0.84	-0.84	-0.84
Wine	0.57	0.54	0.55	0.53	0.56	0.59	0.56	0.58	0.65
Yeast	0.80	0.83	0.75	0.75	0.21	0.15	0.14	0.17	0.19
	SVM (one vs. all)								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Abalone	0.21	0.21	0.21	0.20	0.18	0.16	0.14	0.14	0.10
Car Evaluation	0.09	0.09	0.09	0.22	0.42	0.48	0.52	0.52	0.53
Cardiotocography	0.70	0.70	0.71	0.69	0.69	0.70	0.73	0.74	0.75
Congressional Voting	0.56	0.56	0.54	0.51	0.49	0.48	0.48	0.48	0.49
Contraceptive Choice	0.62	0.62	0.62	0.60	0.57	0.55	0.39	0.33	0.33
Credit Approval	0.46	0.46	0.46	0.47	0.48	0.48	0.48	0.50	0.52
Dermatology	0.67	0.67	0.66	0.66	0.66	0.67	0.67	0.68	0.68
E-coli	0.01	0.01	0.01	0.02	0.04	0.05	0.04	0.07	0.14
Flag	0.68	0.69	0.70	0.70	0.71	0.72	0.71	0.72	0.72
Glass	0.44	0.44	0.43	0.43	0.44	0.45	0.44	0.44	0.41
Haberman's Survival	-0.95	-0.95	-0.95	-0.95	-0.94	-0.94	-0.94	-0.94	-0.93
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.38	0.40	0.45	0.35	0.32	0.40	0.34	0.37	0.31
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.42	0.43	0.46	0.49	0.51	0.55	0.64	0.66	0.67
Post-op	-0.76	-0.84	-0.85	-0.83	-0.75	-0.70	-0.84	-0.76	-0.71
Segmentation	0.38	0.38	0.38	0.39	0.40	0.40	0.42	0.49	0.54
Statlog (vehicle)	0.79	0.79	0.79	0.80	0.80	0.81	0.81	0.81	0.84
WI Breast Cancer (orig.)	0.27	0.29	0.29	0.30	0.32	0.35	0.37	0.44	0.52
WI Breast Cancer (diag.)	0.47	0.47	0.47	0.47	0.48	0.48	0.48	0.49	0.50
WI Breast Cancer (prog.)	-0.82	-0.82	-0.82	-0.83	-0.83	-0.83	-0.83	-0.83	-0.83
Wine	0.58	0.55	0.56	0.55	0.58	0.60	0.57	0.59	0.66
Yeast	0.80	0.81	0.77	0.78	0.13	0.07	0.06	0.08	0.09

Table B.31: Correlation between SVM accuracies and Laplacian score using different k values and $t = 1$ on real data sets

	Laplacian Score ($t = 1$)								
	KNN								
	$k = 2$	$k = 3$	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = N/10$	$k = N/4$	$k = N/2$
Abalone	0.49	0.53	0.58	0.65	0.68	0.72	0.76	0.77	0.67
Car Evaluation	0.13	0.09	0.11	0.18	0.29	0.33	0.35	0.35	0.35
Cardiotocography	0.53	0.54	0.54	0.52	0.52	0.53	0.55	0.56	0.57
Congressional Voting	0.52	0.51	0.49	0.47	0.44	0.43	0.43	0.43	0.44
Contraceptive Choice	0.60	0.60	0.59	0.58	0.53	0.48	0.26	0.18	0.16
Credit Approval	0.55	0.56	0.57	0.59	0.61	0.62	0.62	0.63	0.64
Dermatology	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.73	0.74
E-coli	0.02	0.02	0.02	0.03	0.06	0.08	0.06	0.10	0.17
Flag	0.32	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Glass	0.24	0.26	0.26	0.27	0.27	0.24	0.28	0.23	0.18
Haberman's Survival	0.48	0.49	0.50	0.50	0.50	0.49	0.50	0.49	0.49
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.31	0.32	0.37	0.29	0.27	0.34	0.30	0.32	0.23
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.29	0.30	0.33	0.36	0.39	0.43	0.51	0.54	0.55
Post-op	0.24	0.18	0.17	0.18	0.27	0.33	0.17	0.26	0.32
Segmentation	0.24	0.23	0.24	0.24	0.24	0.24	0.25	0.32	0.36
Statlog (vehicle)	0.49	0.49	0.49	0.50	0.51	0.52	0.52	0.52	0.56
WI Breast Cancer (orig.)	0.28	0.30	0.30	0.31	0.34	0.37	0.38	0.46	0.54
WI Breast Cancer (diag.)	0.46	0.46	0.46	0.47	0.47	0.48	0.48	0.49	0.50
WI Breast Cancer (prog.)	0.22	0.22	0.22	0.21	0.21	0.21	0.22	0.21	0.21
Wine	0.55	0.53	0.53	0.52	0.55	0.57	0.55	0.57	0.63
Yeast	0.70	0.74	0.67	0.67	0.19	0.13	0.13	0.15	0.18

Table B.32: Correlation between 1-NN accuracies and Laplacian score using different k values and $t = 1$ on real data sets

	MCFS (heat kernel k=25)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$t = 0.1$	1	10	100	$t = 0.1$	1	10	100	$t = 0.1$	1	10	100
Binary 1	0.60	0.58	0.80	0.70	0.57	0.57	0.76	0.67	0.93	0.82	0.99	0.88
Binary 2	0.71	0.70	0.76	0.75	0.55	0.54	0.61	0.60	0.98	0.95	0.99	0.97
Binary 3	248%	319%	195%	173%	248%	319%	195%	173%	0.85	0.75	0.69	0.71
Monk 1	0.01	0.01	0.01	0.01	0.02	0.02	0.02	0.02	-0.06	-0.06	-0.06	-0.06
Monk 2	-	-	-	-	-	-	-	-	-0.33	-0.33	-0.33	-0.33
Monk 3	0.17	0.22	0.23	0.18	0.19	0.24	0.24	0.20	0.18	0.23	0.24	0.20
Simple dependent	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Partially dependent	0.33	0.52	0.32	0.30	0.33	0.53	0.32	0.30	0.37	0.56	0.36	0.34
2-way linear corr.	0.67	0.68	0.68	0.66	1.00	1.00	1.00	1.00	0.98	0.98	0.98	0.98
3-way linear corr.	0.26	0.26	0.26	0.26	0.28	0.28	0.28	0.28	0.27	0.27	0.27	0.27
3-way non-linear corr.	0.41	0.41	0.41	0.41	0.41	0.41	0.41	0.41	0.41	0.42	0.42	0.42
1 good and noise	0.26	0.64	0.71	0.34	0.26	0.62	0.69	0.36	0.23	0.61	0.69	0.31
Pure noise	0.49	0.48	0.54	0.60	-0.28	-0.25	-0.25	-0.41	0.54	0.53	0.51	0.69
Un-nested	0.00	0.46	0.00	0.39	0.00	0.41	0.00	0.37	0.00	0.48	0.00	0.41
Monotonic	0.59	0.56	0.64	0.64	0.60	0.67	0.64	0.63	0.64	0.60	0.68	0.69
Two U's	0.99	0.97	1.00	0.99	1.00	0.99	1.00	1.00	0.95	0.92	0.98	0.95
Multimodal (XOR)	1.00	0.00	1.00	1.00	0.99	0.00	0.99	0.99	1.00	0.00	1.00	1.00
linear sep. Guassian	0.53	0.79	0.75	0.47	0.55	0.81	0.77	0.49	0.62	0.85	0.82	0.56
Redundant	-	-	-	-	0.00	0.00	0.00	0.00	-	-	-	-
2 clusters	1835%	1835%	1835%	1835%	1835%	1835%	1835%	1835%	1835%	1835%	1835%	1835%

Table B.33: Correlation between SVM and 1-NN accuracies and MCFS filter values using different values of t and $k = 25$ on artificial data sets

	SVM (one vs. one)											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Binary 1	0.58	0.76	0.80	0.48	0.66	0.71	0.51	0.55	0.60	0.55	0.49	0.66
Binary 2	0.70	0.87	0.78	0.68	0.73	0.75	0.74	0.79	0.70	0.80	0.75	0.63
Binary 3	319%	389%	162%	285%	167%	298%	196%	293%	231%	338%	173%	418%
Monk 1	0.01	0.00	0.03	0.13	0.01	0.00	0.02	0.14	0.01	-0.01	0.10	0.10
Monk 2	-	-	-	-	-	-	-	-	-	-	-	-
Monk 3	0.22	0.23	0.34	0.56	0.20	0.23	0.32	0.54	0.11	0.03	0.54	0.33
Simple dependent	1.00	0.99	1.00	0.99	1.00	1.00	0.98	0.98	0.99	1.00	0.90	1.00
Partially dependent	0.52	0.15	0.14	-0.07	0.36	-0.11	0.03	0.56	0.45	0.22	0.32	0.32
2-way linear corr.	0.68	0.52	0.60	0.81	0.66	0.63	0.50	0.37	0.68	0.64	0.65	0.80
3-way linear corr.	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
3-way non-linear corr.	0.41	0.67	0.69	0.77	0.41	0.00	0.00	0.85	0.41	0.71	0.73	0.78
1 good and noise	0.64	0.40	0.49	0.55	0.41	0.63	0.72	0.43	0.48	0.58	0.66	0.42
Pure noise	0.48	0.46	0.54	0.52	0.60	0.43	0.40	0.41	0.50	0.16	0.42	0.35
Un-nested	0.46	0.60	0.42	0.37	0.41	0.49	0.48	0.47	0.40	0.45	0.25	0.40
Monotonic	0.56	0.53	0.58	0.69	0.65	0.59	0.72	0.58	0.59	0.57	0.62	0.74
Two U's	0.97	1.00	0.99	0.91	0.91	1.00	0.99	1.00	0.99	0.99	0.98	0.99
Multimodal (XOR)	0.00	1.00	1.00	1.00	1.00	0.00	0.00	0.99	0.00	1.00	1.00	0.93
linear sep. Guassian	0.79	0.95	0.67	0.42	0.99	0.73	0.64	0.24	0.67	0.88	0.91	1.00
Redundant	-	-	-	-	-	-	-	-	-	-	-	-
2 clusters	1835%	1835%	1835%	2623%	1835%	1835%	1835%	1261%	1835%	1835%	1835%	517%

Table B.34: Correlation between SVM one vs. one accuracies and MCFS filter values using different kernels and values of k on artificial data sets. For the heat kernel, $t = 1$.

	SVM (one vs. all)											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Binary 1	0.57	0.75	0.76	0.41	0.63	0.68	0.50	0.57	0.57	0.52	0.45	0.63
Binary 2	0.54	0.78	0.67	0.59	0.59	0.62	0.60	0.63	0.52	0.66	0.66	0.46
Binary 3	319%	389%	162%	285%	167%	298%	196%	293%	231%	338%	173%	418%
Monk 1	0.02	0.00	0.03	0.13	0.02	0.00	0.02	0.14	0.02	0.00	0.11	0.10
Monk 2	-	-	-	-	-	-	-	-	-	-	-	-
Monk 3	0.24	0.25	0.36	0.57	0.22	0.25	0.34	0.55	0.13	0.03	0.54	0.34
Simple dependent	1.00	0.99	1.00	0.98	1.00	1.00	0.99	0.99	0.99	1.00	0.88	1.00
Partially dependent	0.53	0.15	0.14	-0.07	0.36	-0.11	0.03	0.57	0.45	0.23	0.32	0.32
2-way linear corr.	1.00	0.98	0.99	0.99	1.00	1.00	0.97	0.93	1.00	1.00	1.00	0.99
3-way linear corr.	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28	0.28
3-way non-linear corr.	0.41	0.68	0.70	0.77	0.41	0.00	0.00	0.85	0.41	0.71	0.73	0.77
1 good and noise	0.62	0.42	0.47	0.55	0.39	0.61	0.70	0.40	0.46	0.57	0.64	0.41
Pure noise	-0.25	-0.26	-0.32	-0.35	-0.34	-0.14	-0.24	-0.13	-0.22	-0.01	-0.20	-0.18
Un-nested	0.41	0.61	0.37	0.38	0.40	0.48	0.42	0.51	0.39	0.44	0.26	0.42
Monotonic	0.67	0.65	0.68	0.67	0.73	0.62	0.60	0.68	0.64	0.64	0.65	0.72
Two U's	0.99	1.00	1.00	0.95	0.95	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Multimodal (XOR)	0.00	0.96	0.97	0.99	0.99	0.00	0.00	0.95	0.00	0.97	0.97	0.85
linear sep. Guassian	0.81	0.94	0.69	0.45	0.98	0.75	0.66	0.27	0.69	0.89	0.92	1.00
Redundant	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2 clusters	1835%	1835%	1835%	2623%	1835%	1835%	1835%	1261%	1835%	1835%	1835%	517%

Table B.35: Correlation between SVM one vs. all accuracies and MCFS filter values using different kernels and values of k on artificial data sets. For the heat kernel, $t = 1$.

	KNN											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Binary 1	0.82	0.89	0.98	0.79	0.95	0.92	0.85	0.76	0.94	0.91	0.89	0.89
Binary 2	0.95	0.93	0.95	0.92	0.99	0.97	0.96	0.98	0.88	0.98	0.91	0.93
Binary 3	0.75	0.85	0.90	0.78	0.61	0.95	0.79	0.81	0.92	0.73	0.87	0.83
Monk 1	-0.06	-0.06	-0.03	0.23	-0.06	-0.06	-0.04	0.24	-0.06	-0.06	0.06	0.10
Monk 2	-0.33	-0.30	-0.32	-0.33	-0.33	-0.30	-0.32	-0.33	-0.33	-0.31	-0.31	-0.34
Monk 3	0.23	0.24	0.35	0.60	0.22	0.25	0.33	0.58	0.13	0.03	0.56	0.35
Simple dependent	1.00	0.99	1.00	0.98	1.00	1.00	0.98	0.99	0.99	1.00	0.89	1.00
Partially dependent	0.56	0.19	0.18	-0.03	0.40	-0.07	0.07	0.60	0.48	0.26	0.36	0.36
2-way linear corr.	0.98	0.93	0.96	1.00	0.98	0.97	0.91	0.85	0.98	0.97	0.97	1.00
3-way linear corr.	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
3-way non-linear corr.	0.42	0.70	0.72	0.80	0.42	0.00	0.00	0.86	0.42	0.72	0.75	0.74
1 good and noise	0.61	0.38	0.46	0.52	0.38	0.60	0.69	0.41	0.46	0.55	0.64	0.39
Pure noise	0.53	0.50	0.59	0.56	0.55	0.31	0.37	0.30	0.41	0.11	0.63	0.23
Un-nested	0.48	0.63	0.44	0.40	0.44	0.52	0.49	0.50	0.43	0.47	0.28	0.43
Monotonic	0.60	0.57	0.60	0.72	0.68	0.63	0.77	0.61	0.64	0.59	0.69	0.78
Two U's	0.92	0.97	0.96	0.84	0.84	0.98	0.96	0.98	0.96	0.97	0.95	0.96
Multimodal (XOR)	0.00	0.97	0.98	0.99	1.00	0.00	0.00	0.96	0.00	0.97	0.97	0.86
linear sep. Guassian	0.85	0.91	0.74	0.51	0.96	0.80	0.72	0.35	0.75	0.92	0.95	1.00
Redundant	-	-	-	-	-	-	-	-	-	-	-	-
2 clusters	1835%	1835%	1835%	2623%	1835%	1835%	1835%	1261%	1835%	1835.60%	1835.60%	517.16%

Table B.36: Correlation between 1-NN accuracies and MCFS filter values using different kernels and values of k on artificial data sets. For the heat kernel, $t = 1$.

	MCFS (heat kernel k=25)											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$	$t = 0.1$	$t = 1$	$t = 10$	$t = 100$
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.67	0.77	0.61	0.73	0.70	0.74	0.63	0.74	0.40	0.53	0.36	0.45
Cardiotocography	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Congressional Voting	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contraceptive Choice	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Credit Approval	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dermatology	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E-coli	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Flag	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Glass	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Haberman's Survival	0.31	0.27	0.33	0.34	0.21	0.21	0.18	0.15	0.10	0.13	0.03	-0.01
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.38	0.38	0.41	0.40	0.45	0.47	0.49	0.49	0.42	0.43	0.46	0.46
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Post-op	-0.81	-0.11	-0.17	-0.11	-0.70	0.18	0.14	0.20	0.22	0.57	0.46	0.55
Segmentation	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Statlog (vehicle)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (orig.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (diag.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (prog.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Wine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Yeast	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table B.37: Correlation between SVM and 1-NN accuracies and MCFS filter values using different values of t and $k = 25$ on real data sets

	SVM (one vs. one)											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.77	0.40	0.71	0.77	0.61	0.62	0.57	0.52	0.62	0.61	0.81	0.59
Cardiotocography	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Congressional Voting	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contraceptive Choice	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Credit Approval	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dermatology	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E-coli	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Flag	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Glass	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Haberman's Survival	0.27	0.16	0.38	0.40	0.29	0.34	0.14	0.12	0.11	0.18	0.25	0.04
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.38	0.43	0.42	0.47	0.35	0.42	0.45	0.26	0.51	0.43	0.41	0.36
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Post-op	-0.11	-0.12	-0.16	-0.11	-0.18	-0.17	-0.15	-0.25	-0.16	-0.24	-0.40	-0.38
Segmentation	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Statlog (vehicle)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (orig.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (diag.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (prog.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Wine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Yeast	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table B.38: Correlation between SVM one vs. one accuracies and MCFS filter values using different kernels and k values on real data sets

	SVM (one vs. all)											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.74	0.40	0.73	0.77	0.65	0.59	0.59	0.53	0.54	0.55	0.75	0.52
Cardiotocography	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Congressional Voting	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contraceptive Choice	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Credit Approval	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dermatology	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E-coli	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Flag	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Glass	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Haberman's Survival	0.21	0.20	0.16	0.15	0.17	0.15	0.21	-0.41	0.20	0.23	0.13	-0.14
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.47	0.55	0.49	0.56	0.45	0.50	0.54	0.37	0.59	0.49	0.52	0.46
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Post-op	0.18	0.15	0.14	0.16	0.11	0.12	0.15	0.02	0.15	-0.02	-0.14	-0.27
Segmentation	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Statlog (vehicle)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (orig.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (diag.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (prog.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Wine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Yeast	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table B.39: Correlation between SVM one vs. all accuracies and MCFS filter values using different kernels and k values on real data sets

	KNN											
	MCFS (heat kernel t=1)				MCFS (dot kernel)				MCFS (zero-one kernel)			
	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$	$k = 25$	$k = 50$	$k = N/4$	$k = N/2$
Abalone	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Car Evaluation	0.53	0.22	0.43	0.47	0.37	0.38	0.33	0.30	0.47	0.38	0.59	0.41
Cardiotocography	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Congressional Voting	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contraceptive Choice	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Credit Approval	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dermatology	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E-coli	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Flag	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Glass	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Haberman's Survival	0.13	0.31	-0.10	-0.11	0.09	-0.01	0.38	-0.29	0.39	0.33	0.13	0.25
Ionosphere	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Iris	0.43	0.51	0.46	0.53	0.41	0.47	0.51	0.32	0.56	0.47	0.47	0.42
Mushroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Page blocks	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Post-op	0.57	0.53	0.52	0.55	0.52	0.45	0.46	0.55	0.37	0.41	0.35	0.32
Segmentation	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Statlog (vehicle)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (orig.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (diag.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WI Breast Cancer (prog.)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Wine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Yeast	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table B.40: Correlation between 1-NN accuracies and MCFS filter values using different kernels and k values on real data sets

B.7 CFS

	CFS											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	3	5	10	$N_b = 2$	3	5	10	$N_b = 2$	3	5	10
Binary 1	**0.50**	0.46	**0.50**	0.47	**0.47**	0.44	0.46	0.42	**0.90**	0.78	0.88	0.89
Binary 2	**0.62**	0.60	0.60	0.60	0.47	0.43	**0.48**	0.44	**0.94**	**0.94**	0.86	0.88
Binary 3	-	40%	39%	36%	-	40%	39%	36%	0.00	0.44	**0.78**	0.72
Monk 1	**0.95**	0.94	0.94	0.94	**0.95**	**0.95**	**0.95**	**0.95**	**0.47**	0.46	0.46	0.46
Monk 2	36%	36%	36%	36%	36%	36%	36%	36%	**-0.29**	-0.43	-0.42	-0.42
Monk 3	**0.94**	0.93	**0.94**	**0.94**	**0.93**	0.92	**0.93**	**0.93**	**0.95**	0.94	**0.95**	**0.95**
Simple dependent	**1.00**	0.98	**1.00**	**1.00**	**1.00**	0.98	0.99	**1.00**	**1.00**	0.98	0.99	**1.00**
Partially dependent	0.91	**0.92**	0.91	**0.92**	0.91	**0.92**	0.91	**0.92**	0.89	0.90	0.90	**0.91**
2-way linear corr.	0.57	0.47	0.66	**0.69**	0.99	0.96	**1.00**	**1.00**	0.95	0.90	0.98	**0.99**
3-way linear corr.	0.80	**0.87**	0.86	0.84	0.80	**0.88**	0.86	0.84	0.80	0.83	**0.84**	0.83
3-way non-linear corr.	**0.92**	0.91	**0.92**	**0.92**	**0.92**	0.90	**0.92**	**0.92**	0.91	0.89	**0.92**	0.91
1 good and noise	**0.96**	**0.96**	**0.96**	**0.96**	**0.97**	0.96	0.96	0.96	**0.96**	**0.96**	**0.96**	**0.96**
Pure noise	0.79	0.74	**0.80**	0.55	-0.73	-0.72	-0.78	**-0.38**	0.94	**0.95**	0.94	0.78
Un-nested	0.70	0.74	0.73	**0.77**	0.80	0.86	0.85	**0.88**	0.74	0.78	0.77	**0.80**
Monotonic	0.88	**0.92**	0.85	0.91	0.84	0.83	0.85	**0.86**	0.90	0.93	0.87	**0.92**
Two U's	0.99	0.94	**1.00**	0.98	**1.00**	0.90	0.99	0.96	0.96	0.98	0.99	**1.00**
Multimodal (XOR)	0.46	0.70	0.37	**0.93**	0.61	0.57	0.53	**0.86**	0.59	0.59	0.51	**0.87**
linear sep. Guassian	0.88	0.94	0.88	**1.00**	0.87	0.93	0.90	**1.00**	0.83	0.90	0.93	**1.00**
Redundant	22%	22%	22%	22%	**-0.82**	**-0.82**	**-0.82**	**-0.82**	22%	22%	22%	22%
2 clusters	26%	31%	33%	34%	26%	31%	33%	34%	26%	31%	33%	34%

Table B.41: Correlation between SVM and 1-NN accuracies and CFS filter values using different quantizations on artificial data sets

	CFS											
	SVM (one vs. one)				SVM (one vs. all)				KNN			
	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$	$N_b = 2$	$N_b = 3$	$N_b = 5$	$N_b = 10$
Abalone	0.07	0.41	0.60	**0.61**	**0.19**	0.08	0.14	0.10	0.12	0.62	**0.74**	**0.74**
Car Evaluation	**0.66**	**0.66**	**0.66**	**0.66**	**0.61**	0.60	0.60	0.60	**0.40**	0.39	**0.40**	**0.40**
Cardiotocography	0.63	0.80	0.87	**0.92**	0.62	0.82	0.88	**0.92**	0.67	0.77	0.82	**0.84**
Congressional Voting	**0.90**	**0.90**	**0.90**	**0.90**	**0.90**	**0.90**	**0.90**	**0.90**	0.87	**0.88**	**0.88**	**0.88**
Contraceptive Choice	0.42	0.48	**0.72**	0.71	0.61	0.62	**0.82**	0.82	0.07	0.10	0.41	**0.50**
Credit Approval	0.88	0.89	0.92	**0.93**	0.88	0.89	0.92	**0.93**	0.74	0.75	0.78	**0.79**
Dermatology	0.84	**0.88**	**0.88**	**0.88**	0.82	**0.87**	**0.87**	**0.87**	0.87	**0.90**	**0.90**	**0.90**
E-coli	0.86	**0.92**	0.91	**0.92**	0.85	**0.89**	0.87	**0.89**	0.85	**0.89**	0.88	**0.89**
Flag	**0.82**	0.80	0.81	0.79	0.83	**0.86**	0.85	0.81	0.71	0.73	**0.78**	**0.78**
Glass	0.61	0.50	0.67	**0.73**	0.66	0.59	0.74	**0.80**	0.40	0.46	0.44	**0.62**
Haberman's Survival	-0.93	**-0.92**	-0.94	-0.94	**-0.83**	-0.91	-0.90	-0.90	**0.73**	0.46	0.55	0.59
Ionosphere	0.68	0.77	0.80	**0.84**	0.68	0.76	0.81	**0.84**	0.42	0.42	0.45	**0.48**
Iris	**0.90**	0.88	0.88	0.88	**0.90**	0.87	0.89	0.89	**0.93**	0.90	0.92	0.92
Mushroom	0.47	0.66	**0.67**	0.66	0.48	0.66	**0.67**	0.66	0.45	0.64	**0.69**	**0.69**
Page blocks	0.39	0.50	0.59	**0.80**	0.52	0.66	0.71	**0.79**	0.42	0.59	0.63	**0.74**
Post-op	**-0.32**	-0.65	-0.62	-0.62	**-0.16**	-0.58	-0.58	-0.58	**0.32**	0.25	0.18	0.18
Segmentation	0.84	0.80	**0.86**	**0.86**	0.84	0.79	**0.86**	0.84	0.82	0.79	0.84	**0.85**
Statlog (vehicle)	0.58	0.70	0.73	**0.77**	0.59	0.71	0.74	**0.79**	0.46	0.67	0.62	**0.71**
WI Breast Cancer (orig.)	**0.82**	0.80	0.81	0.80	**0.83**	0.81	0.81	0.80	**0.80**	0.79	0.79	0.77
WI Breast Cancer (diag.)	0.50	0.62	**0.66**	0.64	0.50	0.62	**0.66**	0.65	0.57	0.69	**0.72**	0.71
WI Breast Cancer (prog.)	**-0.53**	-0.64	-0.61	-0.64	**-0.49**	-0.62	-0.59	-0.61	0.05	**0.17**	0.15	0.16
Wine	0.81	0.82	0.86	**0.89**	0.80	0.81	0.85	**0.87**	0.74	0.83	0.84	**0.87**
Yeast	0.42	**0.85**	0.73	0.84	0.09	**0.56**	0.36	0.51	0.52	0.86	0.81	**0.90**
average	0.49	0.54	0.58	**0.60**	0.51	0.53	0.56	**0.57**	0.56	0.63	0.66	**0.69**

Table B.42: Correlation between SVM and 1-NN accuracies and CFS filter values using different quantizations on real data sets

Appendix C

Performance of filter measures on the artificial data sets

C.1 Binary data sets

The *binary* data sets are all two class data sets with three noisy binary features. All of the *binary* data sets require all three features to fully separate the data set.

For *binary 1*, the class is one when all three features are one. Each individual feature carries some information about the class, since the class cannot be one if any feature is zero. KNN achieves an accuracy of 99.17% when all three features are used. For single feature sets, the KNN achieves accuracies of 80% to 82.5% and adding the second feature improves the accuracy to an average of 86.0%. The SVM classifiers achieve almost perfect separation when using all three features (98.8%), but perform better with the single feature sets then the two feature sets.

Univariate RELIEF-F scores each feature similarly and each has a score greater than zero. The difference between the nearest hit and nearest miss for points in class zero will be quite large, and these form the majority of the points in the set. Summing the value of the individual features mimics the accuracy increase of the KNN, but does not match well with the non-monotonic SVM. This is similar for all the univariate measures (probability measures, mutual information and symmetric uncertainty), which all identify each feature as being somewhat valuable. Fisher's interclass separability criterion also prefers the two feature sets over the single feature sets. Subset and count-based RELIEF both perform well for the KNN classifier, but count-based RELIEF outperforms the subset and univariate measures for the SVM as there is a large increase in its value for the three feature set.

In *binary 2*, points are in class 1 if exactly two features are 1. Each individual feature carries some information about the class, as the class is more likely to be zero when a feature is zero, and vice versa. The KNN can achieve an accuracy of 100% when all three features are used. However, this set appears to be slightly harder than the *binary 1* data set. When only a single feature is used, the average accuracy is only 56.1%. Similar to *binary 1*, the SVM performs better with the single feature sets than the two feature sets while the KNN accuracy is better on the two feature sets. Hence, all the univariate measures perform better for the KNN, where the accuracy steadily improves are more features are added. This is the same for the subset and count RELIEF measures as well as Fisher's, all of which rank the two feature set more highly.

In *binary 3*, points are in class one if all three features take the same value. The KNN can achieve 100% accuracy when all three features are used, but the SVM is unable to separate the set because it is multi-modal. The SVM achieves 75% accuracy with any feature set.

Each of the individual features is ranked as being slightly informative by each of the univariate measures, mostly due to the class imbalance. All of the univariate measures match more closely with the KNN.

Fisher's interclass separability criterion does not work well for *binary 3*. Fisher's assumes compact, unimodal clusters, but *binary 3* is a multi-modal set. For this data set, Fisher's identifies the two class centers as being very close, and detects the class width as being quite large, since the class width encompasses points from two different clusters. The effect is that the feature set ranking is fairly random.

The additive multi-variate mutual information measures do not appear to work any better than the univariate mutual information measures for the *binary* data sets. mRMR and FOU both include terms that discount features that have a high mutual information with features that have already been selected. However, the features in the *binary* data sets share almost no mutual information and hence these extra terms do not greatly contribute to the scores. CFS also includes terms to reduce the score of correlated features. For the *binary* data sets, it performs no better than the simple symmetric uncertainty on which it is based.

Because CMIM always takes positive values, it naturally ranks the feature sets containing a larger number of features more highly. This is similar to the univariate measures and hence CMIM also matches more closely with the KNN than the SVM accuracies for the *binary* data sets.

The unsupervised Laplacian score and MCFS both appear to work well for these data sets for similar reasons as the univariate measures. Each feature is given a similar value,

and the sum of the feature values ranks feature sets containing a larger number of features more highly. As with the univariate measures, the correlation is higher for KNN than SVM.

C.2 Monk 1 and Monk 3

The *Monk* data sets are two-class sets with six discrete, non-noisy features with varying numbers of discrete values. All are all non-monotonic. The are also prone to errors from sampling due to the relatively large number of clusters and small number of samples. None are linearly separable.

The *Monk 3* data set is one when $(x_5 = 3 \& x_4 = 1) | (x_5 \neq 4 \& x_2 \neq 3)$. There are three informative features (x_5, x_4 and x_3). The KNN classifier is able to use all three of these features to properly separate the set, with the highest accuracy achieved when only the three informative features are used. Sets that contain the three highest ranked features perform well, and are also ranked most highly by the mutual information measures. The SVM classifier accuracy does not improve when f_4 is included in the set.

Because there is no noise in the set, the univariate RELIEF-F values are all zero because the nearest hit and miss for an individual feature are all in the same location. The subset-based RELIEF performs better for this data set.

The mutual information and symmetric uncertainty measures work well for the *Monk 3* data set for both classifiers. This data set requires more than one feature for full separation, but some information about the class is contained in each of the informative features. Features x_5 and x_2 have a high mutual information, and feature x_4 has a much lower mutual information, but is still much higher than the non-informative features. Similarly, the probability measures are quite high for features 2 and 5, but are much lower for feature 4. This matches well for both the KNN, which is able to use all three informative features, and also for the SVM, since the filter measure values for the unhelpful x_4 feature are much lower.

For *Monk 1*, the class is one when $(x_1 = x_2) | (x_5 = 1)$. The KNN is able to use all three informative features to improve accuracy. However, the SVM is only able to use feature 5. Both the KNN and SVM classifiers are non-monotonic.

Feature 5 is the only feature with a RELIEF-F value that is not zero. This is the same for all the univariate measures (probability, mutual information and symmetric uncertainty). This is correct for the SVM, but incorrect for the KNN, resulting in a lower correlation.

The subset-based RELIEF measure performs much better for the KNN, where including features 1 and 2 causes the measure value to increase. For SVM, this decreases the correlation, because the value of these feature sets is overestimated.

The *Monk 1* data set is difficult for the Fisher's measure, because Fisher's assumes compact, unimodal clusters. Not only is the *Monk 1* data set multi-modal, but one of the clusters is elongated. Fisher's measure underestimates the separability of this data set for the KNN.

CMIM also struggles with *Monk 1*, despite the fact that the conditional mutual information between x_1 and x_2 is high. However, their individual mutual information is low, and the conditional mutual information with x_5 is low. The first feature selected is feature five, and since the conditional mutual information is taken with respect to the features already in the set, adding features one and two does not greatly improve the score. In fact, the score for the set with only the three informative features is fairly low, which does not match well with the KNN accuracies.

Neither mRMR or FOU perform better than the univariate mutual information measures for either of these sets, as the features are not correlated.

The Laplacian score appears to be particularly unsuitable for the *Monk* data sets for both the KNN and the SVM classifier. Because the features themselves are the same across all three *Monk* data sets, the features are ranked in the same order for all the data sets. This is clearly not desirable, since the informative features are different for the different sets. A similar problem occurs for MCFS. Interestingly, the Laplacian score ranks each of the features differently. The Laplacian score fails for features three and six, both of which take only 2 values. Points with different values are not neighbours, and not considered, and points that are neighbours have the same value and therefore contribute nothing. Only when the number of samples is large do points with different values get included in the neighbourhood graph, and only then do features three and six get filter values greater than zero. Hence, Laplacian score may be problematic for data sets with nominal valued features, where features with too few values are not captured by the neighbourhood graph.

C.3 Monk 2

The *Monk 2* data set is discussed in detail in Section 4.3.3.

C.4 Simple dependent, partially dependent and linearly separable Gaussian

All three data sets are two feature, two class sets and all three are monotonic. They theoretically require both features for full separation, but each feature is moderately informative about the data set since points with lower feature values are more likely to come from class zero. Because these are only two feature sets, and all work best using both features, the expectation is that the correlation will be high by default for any filter measure that naturally tends to prefer larger numbers of features. This is the case for the univariate measures, LS and MCFS and consistency.

On the KNN classifier, using both features gives an accuracy of 92.0% for the *simple dependent*, 97.3% for the *partially dependent* and 95% for the *linearly separable Gaussian*. Points that are along the separating line may be misclassified, as the nearest neighbour may easily be from the other side. The accuracy is directly related to the number of points near the line, with the shorter line of the *partially dependent* set yielding better results. For the *simple dependent* and *linearly separable Gaussian*, the two features are approximately equally informative and yield similar results. For the *partially dependent* set, one feature is more powerful and yields a better accuracy.

For the SVM, the *simple dependent* and *partially dependent* sets are, in theory, fully linearly separable. However, sampling issues along the training line mean that some of the trained SVMs might be slightly off, and some of the test points along the line may be misclassified. Using both features, the *simple dependent* set is only 93.3% accurate. As with the KNN, the *partially dependent* is more accurate, as the separating line is shorter.

For *simple dependent* and *linearly separable Gaussian*, there is a larger concentration of points from the same class at the extremes of each feature. This means that the probability and mutual information measures are able to determine that each individual feature is informative. It also makes it more likely that the closest point will be in the same class. Hence, univariate RELIEF-F also determines that both features are moderately informative. The *partially dependent* data set is similar, except that this effect is less pronounced with one feature, and more pronounced with the other. The subset-based and count-based RELIEF measures also work well for these data sets for both classifiers.

Fisher's also works well, indicating that the two feature sets are better, since the class centers are further apart even though the within class scatter is large for all data sets.

The multivariate mutual information measures and CFS do not add much on these data sets since the features are not correlated and the univariate measures already perform quite well.

C.5 Two-way linear, three-way linear and three-way non-linear correlated data sets

The *two-way linear correlated* set is monotonic, but adding the second feature does not improve the accuracy as much as would be expected for non-correlated features. The individual features give accuracies of 97.5% and 97.8% on the KNN. Including the second feature increases the accuracy to 99.3%.

The *three-way linear correlated* data set is non-monotonic on the KNN classifier. The best performance is achieved when only the third feature is used (100%). The SVM is able to achieve almost 100% accuracy on any data set that includes the third feature.

The *three-way non-linear correlated* data set is monotonic, but using more than two features does not improve the accuracy on either the KNN or the SVM.

Fisher's interclass separability has a low correlation with the *two-way linear* accuracy on both classifiers. In this data set, the two-feature set is slightly better than either of the single feature sets. The filter measure reflects this, but identifies the wrong single feature set as being better. This is not really a large problem, but because the correlation is only between three points, this small error causes a large decrease in the correlation value.

The FOU and mRMR measures both severely undervalue the two-feature set, due to the high correlation between the features. For this data set, the CMIM and univariate measures are better. The multivariate measures outperform the univariate measures for the other correlated data sets. CFS is able to outperform the simple symmetric uncertainty on these data sets.

The consistency measure works quite well for these data sets at larger quantization values and is able to correctly determine that the addition of the third feature in the *three-way non-linear* set adds nothing. It does not, however, identify the *three-way linear correlated* as a non-monotonic set.

The MCFS and Laplacian score filters also perform poorly on the *three-way correlated* data sets because the features are ranked almost equally for their ability to preserve the structure. Because the features are scored individually and summed, the value of adding more features to the set is overestimated. Neither is capable of determining that only one feature is required for class separation in the three way linear set, because the labels are not used.

C.6 One good feature and noise

The KNN can achieve 100% accuracy on this data set when only the good feature is used. Adding noise features decreases the accuracy. This data set is non-monotonic for the KNN. The SVM one vs. one achieves 100% accuracy using any feature set that includes the good feature. The one vs. all performs best when only the good feature is used, and achieves a maximum accuracy of 82.7%.

For the data set with a single informative feature and noise, the RELIEF value for the informative feature is much higher than the noise values, all of which are negative. Hence, the summing method correctly predicts the degradation of performance of the KNN classifier that comes from adding noise features. For the SVM classifier, there is no decrease in accuracy from the noise features, but the accuracy of the all noise set is significantly lower, as predicted by the measures.

None of the other univariate measures match the non-monotonic response of the KNN classifier. Similarly, the consistency measure also does not decrease when the noise features are added, but closely matches the response of the SVM since the single feature is fully consistent and adding the noise features does not affect the filter value.

Neither MCFS nor Laplacian score work well for this data set, because most of the neighbouring points are defined by the noise features, rather than the one informative feature.

C.7 Un-nested

The *un-nested* set uses three features. Features two and three can be used together to perfectly separate the set. Feature one matches the class, but contains a large amount of noise. The best single feature set is feature one, but the best two feature set contains features two and three.

KNN achieves 100% accuracy as long as both features two and three are included in the set. Using only feature one gives an accuracy of 82.7%. The SVM one vs. one achieves 100% accuracy when features two and three are both included. The one vs. rest configuration is not able to separate the data set, even using the two good features.

None of the univariate measures are able to correctly rank the two-feature set as highly as the three feature set. Fisher's also has a slight preference for feature sets with a larger number of features, because adding another spacial dimension increases the possible between class distance. It also ranks the three feature set more highly than the two feature

set. The multivariate mutual information measures outperform the univariate measures on this set.

Consistency performs quite well on this set, and correctly determines that the two feature set is as powerful as the three feature set. It also correctly identifies the best single feature set.

C.8 Monotonic

KNN and SVM one vs. one are able to achieve 100% accuracy on this data set when all four features are used. The accuracy using data sets with fewer features is lower.

For this data set, although adding more features does improve the accuracy on both classifiers, most of the univariate measures overestimate the benefit of adding new features. This is similar to the problems seen in the redundant set. Count-based and subset RELIEF both perform slightly better than the univariate RELEIF measures for this data set, as do the multivariate mutual information measures.

CFS also performs better than the simple symmetric uncertainty on this set, where the symmetric uncertainty overestimates the benefit of adding new features.

C.9 Two U's

Using both features on the KNN gives an accuracy of 99.75%. This is slightly better than the SVM, which gives an accuracy of 97.25% using both features.

This is a relatively easy data set for all the filter measures because despite the fact that it is not linearly separable, both classifiers work better with both features. Since the U shapes are interlocked and more spread in one direction, one of the features is significantly more powerful than the others. Since there are a small number of sample points, the correlation will be high for any filter measure that ranks the two feature set more highly. This happens for all the filter measures. Most of the filter measures are also able to find the more powerful feature. This is even the case for LS and MCFS for most parameterizations.

C.10 Multi-modal (XOR)

For the *multi-modal* data set, using a single feature and a KNN classifier gives an accuracy of 51.1% to 54.8%, which is just over chance, while using both features gives an accuracy of 100%. The SVM achieves an accuracy of just 62.7% using two features, but is able to achieve a higher accuracy with the two feature set because the two-feature set allows the SVM to use a diagonal separating line that divides one of the clusters. The data set is monotonic and uses two fully dependent features. Neither feature is informative individually, but together they are quite powerful for the KNN.

The univariate RELIEF measures fail completely for this data set. Neither feature is individually informative about the class and the nearest hits and misses occur very close together. The univariate RELIEF measure decreases for the two feature set, since the RELIEF value for one of the features is negative. The count-based and the subset-based RELIEF are much better choices for this data set for both the KNN and the SVM classifier, where using both features together improves the accuracy.

The univariate mutual information measures have a fairly high correlation with the accuracy, but this is likely due to the low number of sample points. Even a small amount of mutual information on either feature would indicate that the two feature set is a better choice. However, they are both outperformed by the multivariate conditional mutual information measures, which correctly assess the combined power of the two features. mRMR, however, is not a good choice for this data set. With mRMR, the two feature set value is even lower than the univariate measures due to the very small amount of noise-induced mutual information between the two features.

C.11 Redundant and one cluster per class

Any combination of features in the one cluster per class data set gives 100% accuracy on both the KNN and SVM classifiers. The data set is technically monotonic, but adding the second feature does not improve accuracy. The KNN classifier can achieve 100% accuracy using any set of features in the redundant set. For the redundant set, the one vs. one SVM achieves 100% accuracy with any feature set. The performance of the one vs. all actually degrades as more features are added.

For these data sets, all the summed univariate measures drastically overestimate the benefit of adding more features. As with the univariate measures, Laplacian score and

MCFS do not work well for redundant features, because each of the features is ranked individually and then summed.

Subset-based RELIEF score also increases as more features are added. The additional dimensions increase the distance between the clusters, and imply that more features will give a higher accuracy. The value of the count-based RELIEF, however, does not change as more features are added. The one vs. all SVM cannot separate this data set, since the "rest" portions for classes two and three are multi-modal. For this classifier, the correlation is actually negative, as adding more features decreases the accuracy of the classifier.

Fisher's measure performs well on the redundant sets, maintaining the same value for the redundant set, and increasing only very slightly for the 2 clusters data set. Consistency also works extremely well for the redundant sets since a duplicate feature can never separate points that its original does not.

The multivariate mutual information measures perform well on these redundant sets, but will not necessarily perform well in the general case. FOU and mRMR both include terms that decrease the filter measure value proportionally to the mutual information between the features. For these data sets, the mutual information between the features is 1, and the mutual information between each feature and the class is 1, so these terms perfectly cancel. However, this will not be the case for every data set with redundant features. If the mutual information between the redundant features and the class is less than 1, these measure would indicate that adding a redundant feature would hinder performance, which is incorrect. Hence, although the multivariate mutual information measures perform well in these tests, they will not perform well on every set with redundant features.

CFS does increase slightly as the redundant features are added, but the measure value is not doubled when a second feature is added, as it is when symmetric uncertainty is used alone. Because CFS accounts for feature-feature information, the measure value increases only slightly when a fully redundant feature is added.

Appendix D

Graphs of artificial time series and human motion joint angle features

D.1 Artificial data sets

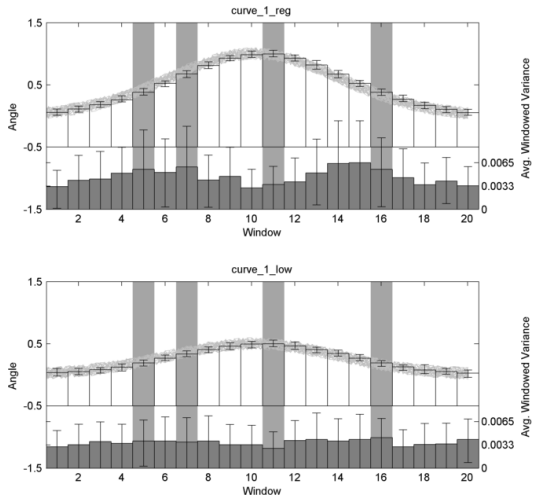


Figure D.1: Regular vs. low curve

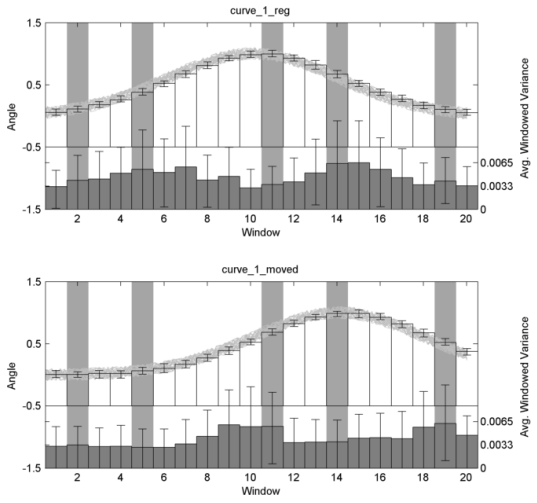


Figure D.2: Regular vs. moved curve

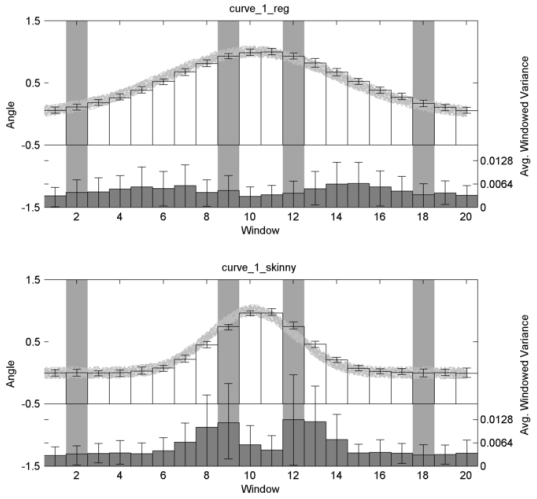


Figure D.3: Regular vs. skinny curve

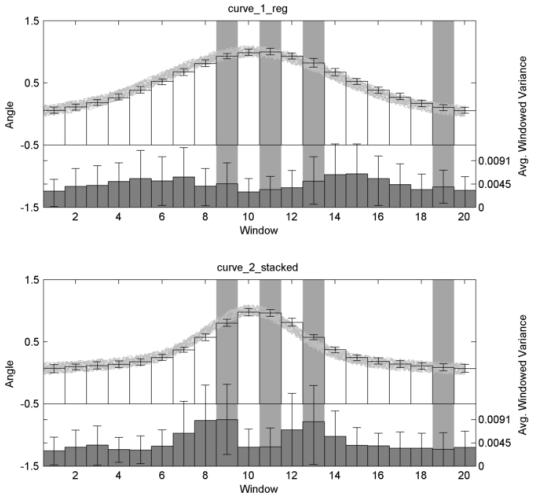


Figure D.4: Regular vs. stacked curve

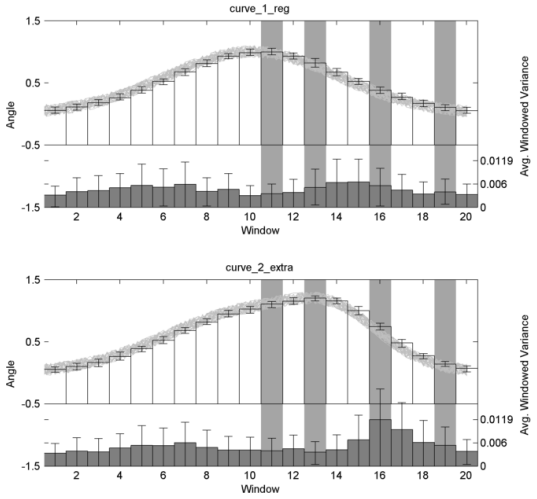


Figure D.5: Regular vs. extra curve

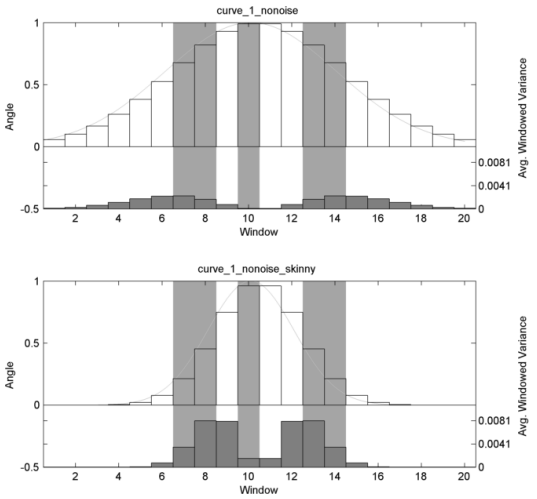


Figure D.6: Regular vs. skinny curve with no added noise

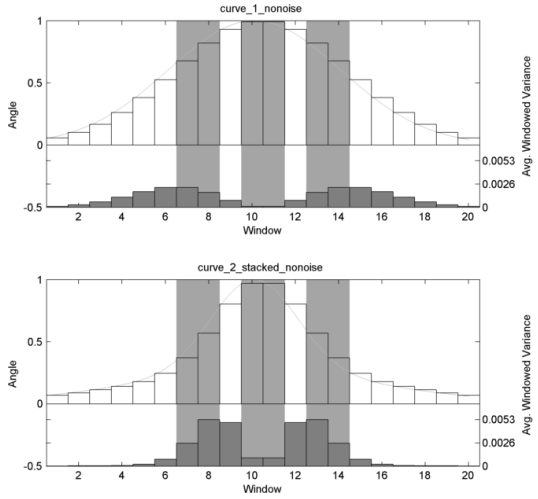


Figure D.7: Regular vs. stacked curve with no added noise

D.2 Human motion data set

D.2.1 Minimum, maximum and average features

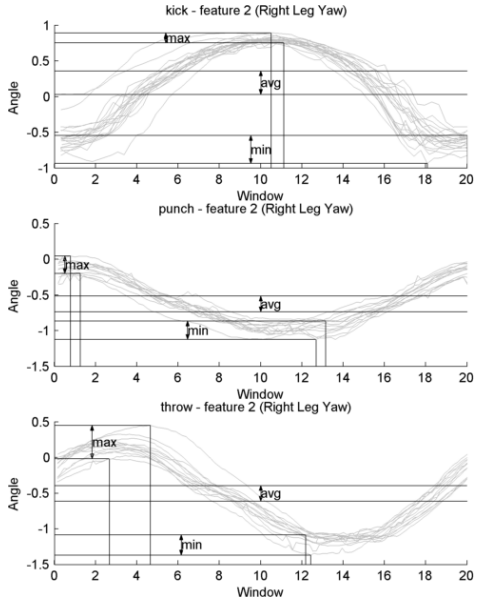


Figure D.8: Minimum, maximum and average for right leg yaw

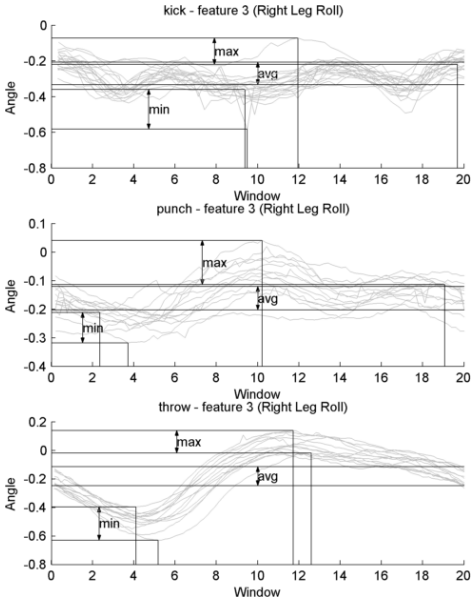


Figure D.9: Minimum, maximum and average for right leg roll

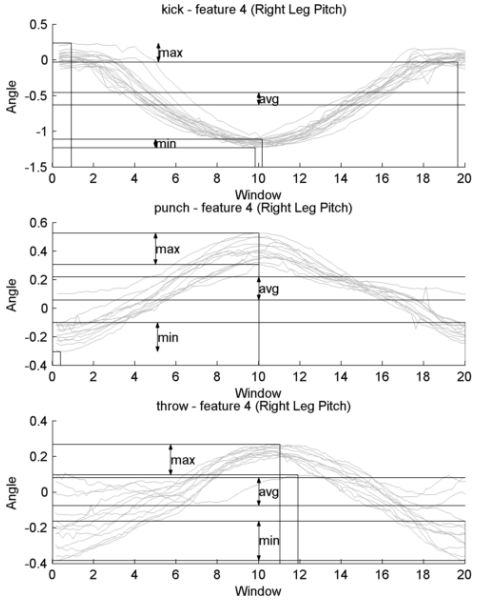


Figure D.10: Minimum, maximum and average for right leg pitch

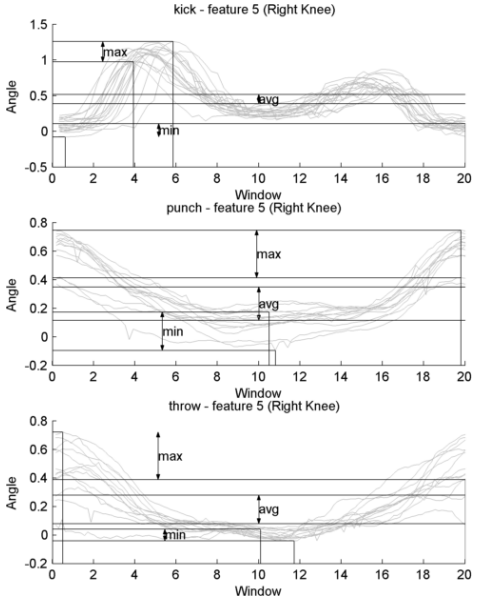


Figure D.11: Minimum, maximum and average for right knee

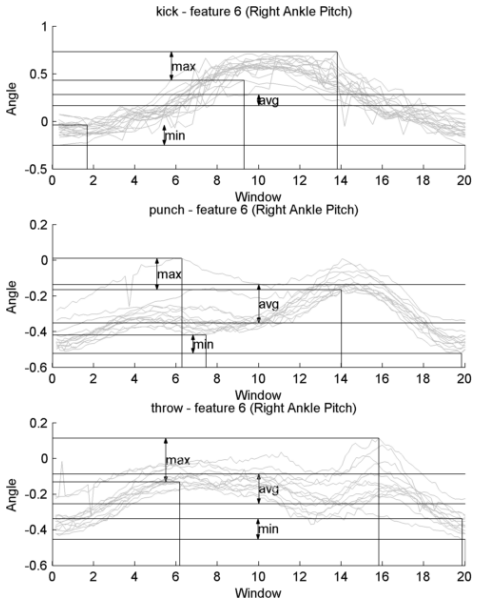


Figure D.12: Minimum, maximum and average for right ankle pitch

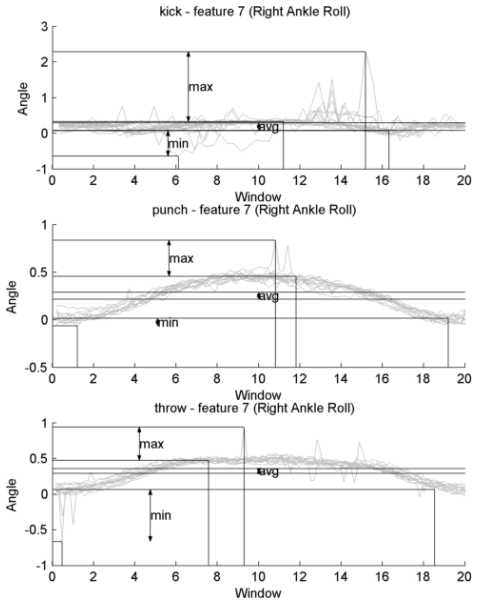


Figure D.13: Minimum, maximum and average for right ankle yaw

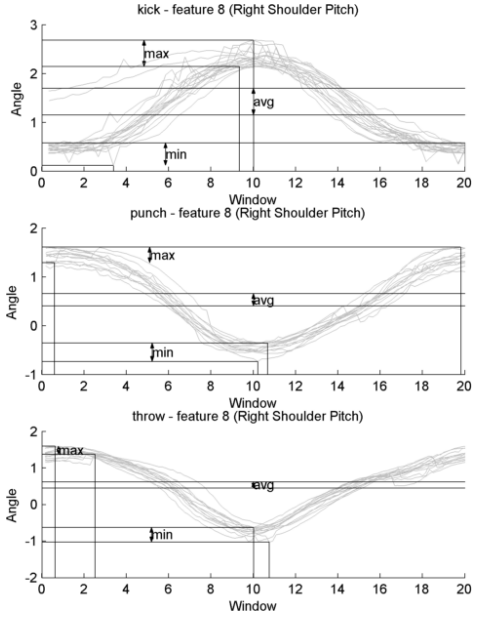


Figure D.14: Minimum, maximum and average for right shoulder pitch

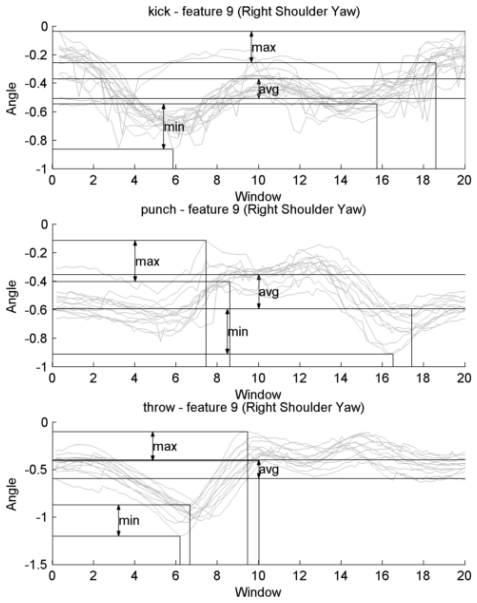


Figure D.15: Minimum, maximum and average for right shoulder yaw

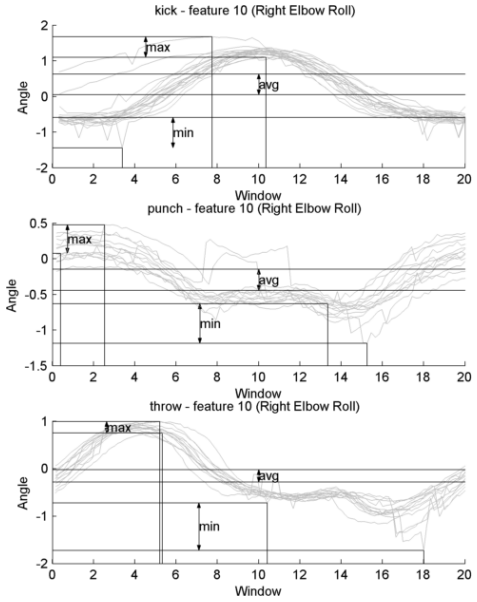


Figure D.16: Minimum, maximum and average for right elbow roll

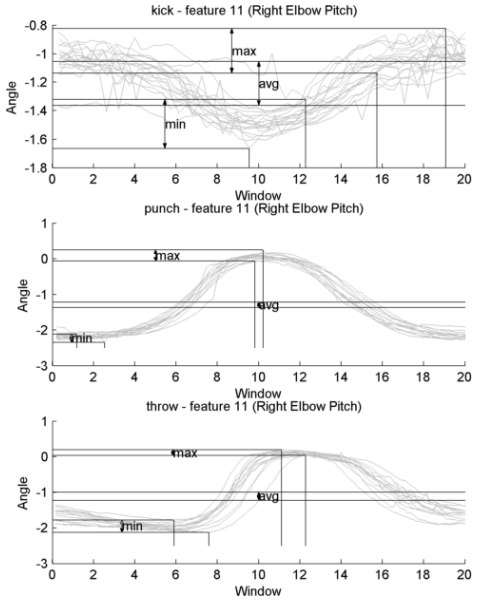


Figure D.17: Minimum, maximum and average for right elbow pitch

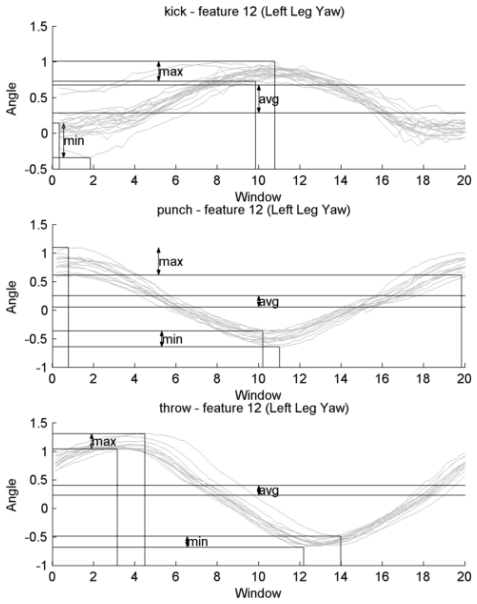


Figure D.18: Minimum, maximum and average for left leg yaw

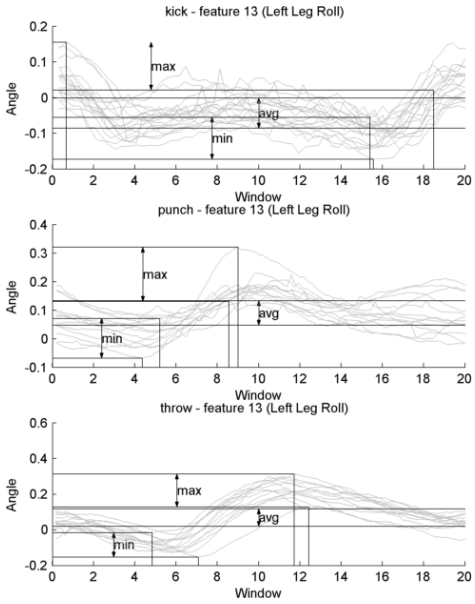


Figure D.19: Minimum, maximum and average for left leg roll

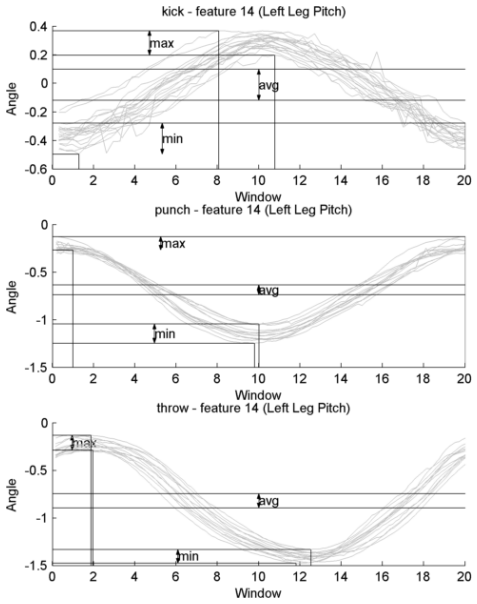


Figure D.20: Minimum, maximum and average for left leg pitch

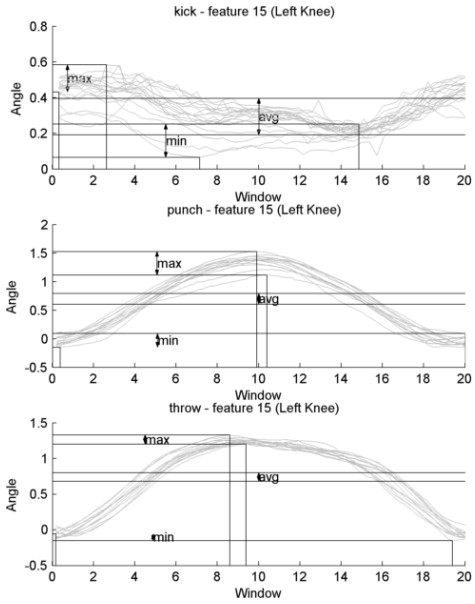


Figure D.21: Minimum, maximum and average for left knee

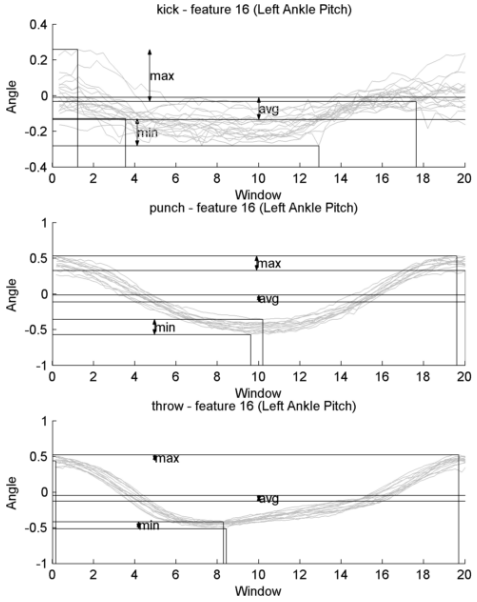


Figure D.22: Minimum, maximum and average for left ankle pitch

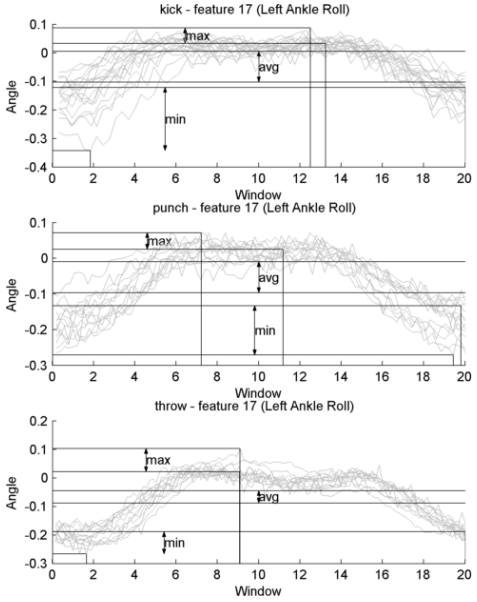


Figure D.23: Minimum, maximum and average for left ankle roll

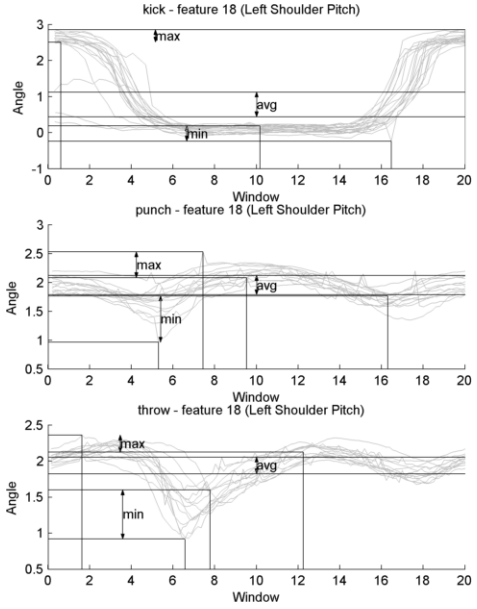


Figure D.24: Minimum, maximum and average for left shoulder pitch

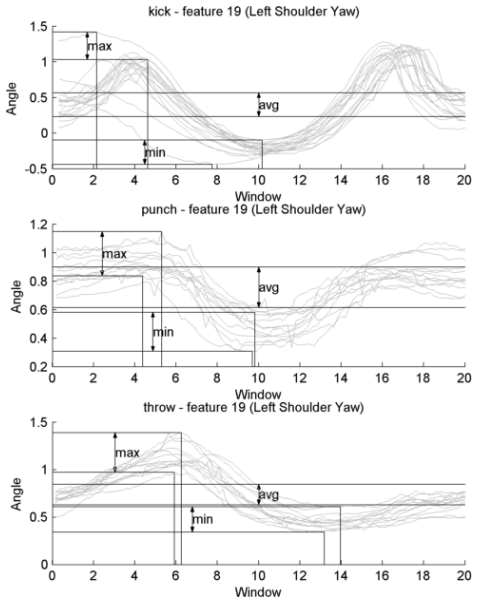


Figure D.25: Minimum, maximum and average for left shoulder pitch

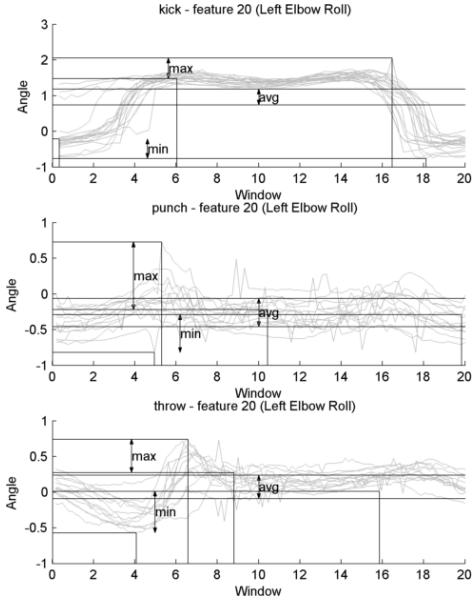


Figure D.26: Minimum, maximum and average for left elbow roll

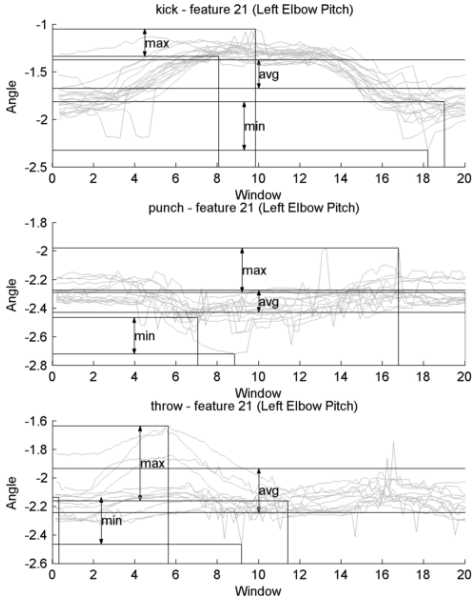


Figure D.27: Minimum, maximum and average for left elbow pitch

D.2.2 Window average and variance features

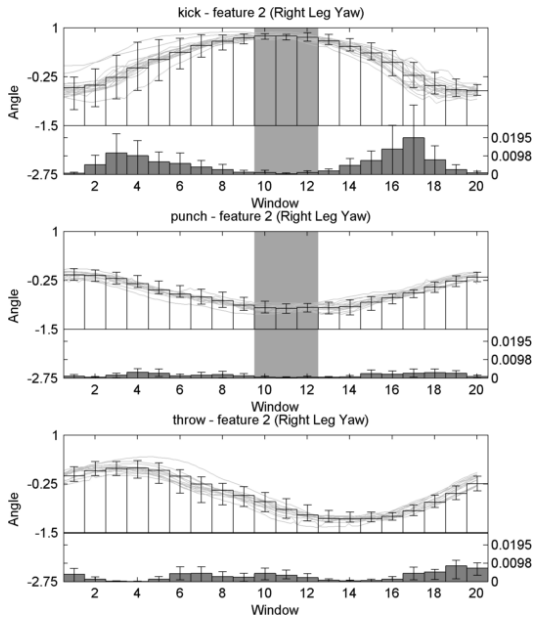


Figure D.28: Window average and variance features for right leg yaw

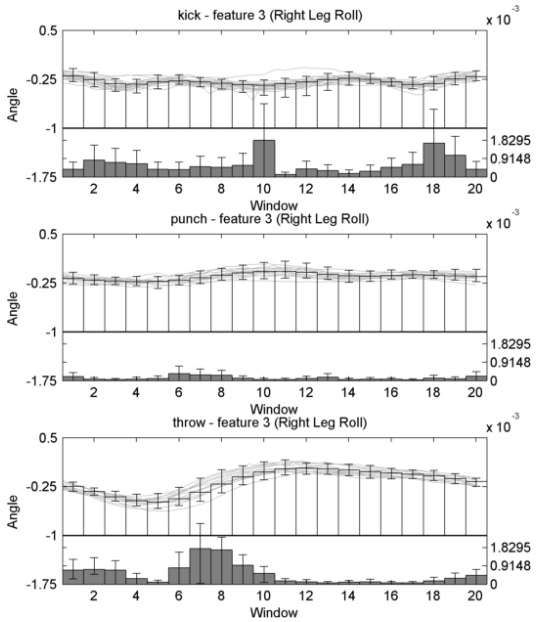


Figure D.29: Window average and variance features for right leg roll

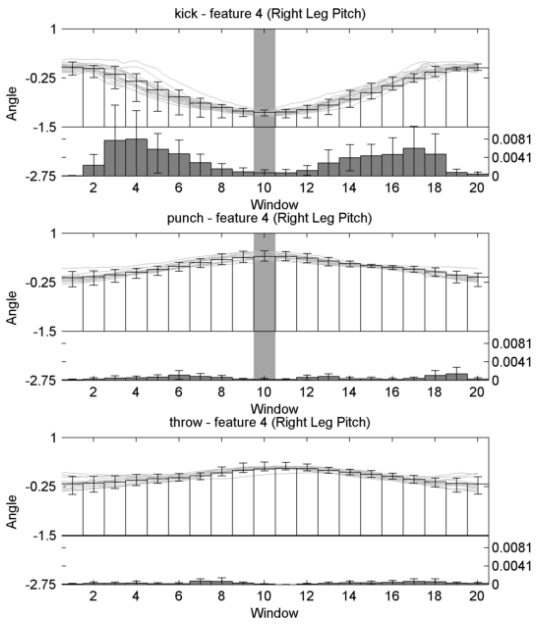


Figure D.30: Window average and variance features for right leg pitch

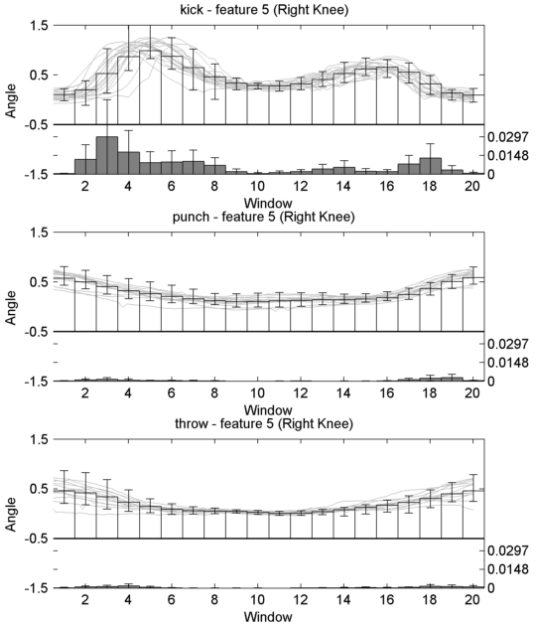


Figure D.31: Window average and variance features for right knee

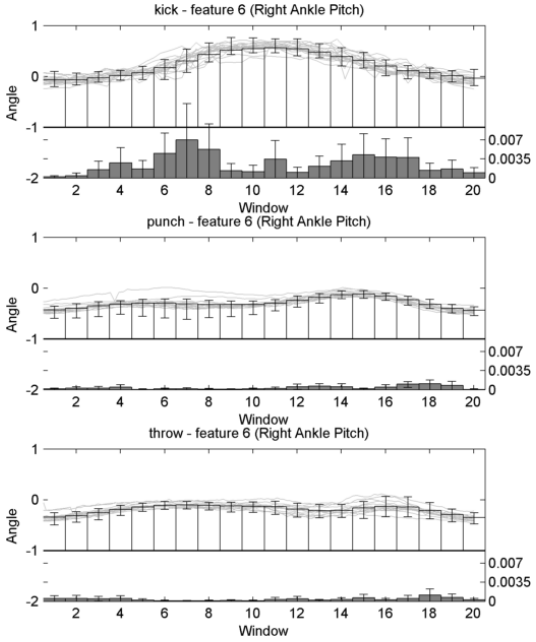


Figure D.32: Window average and variance features for right ankle pitch

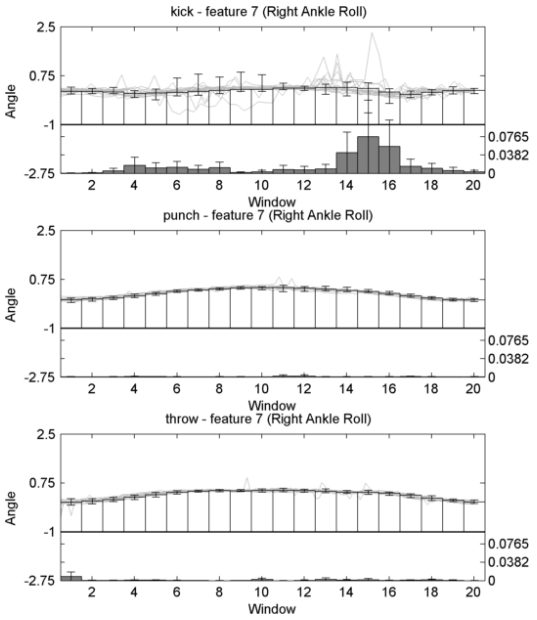


Figure D.33: Window average and variance features for right ankle roll

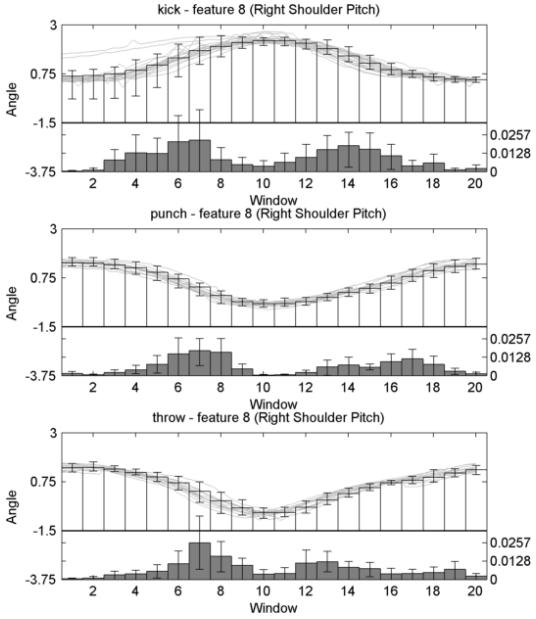


Figure D.34: Window average and variance features for right shoulder pitch

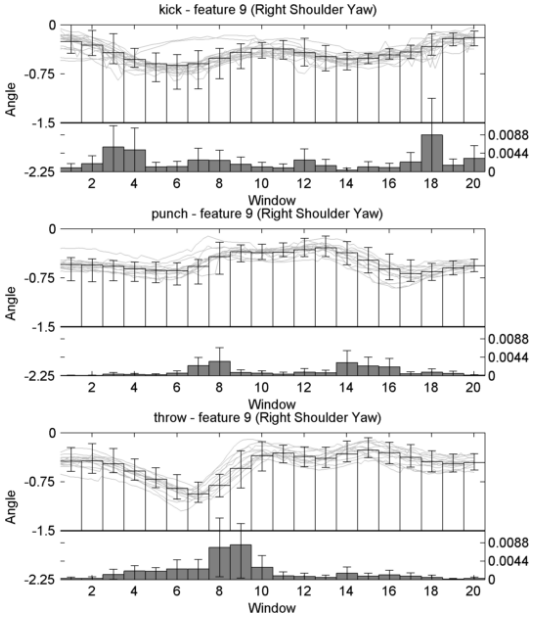


Figure D.35: Window average and variance features for right shoulder yaw

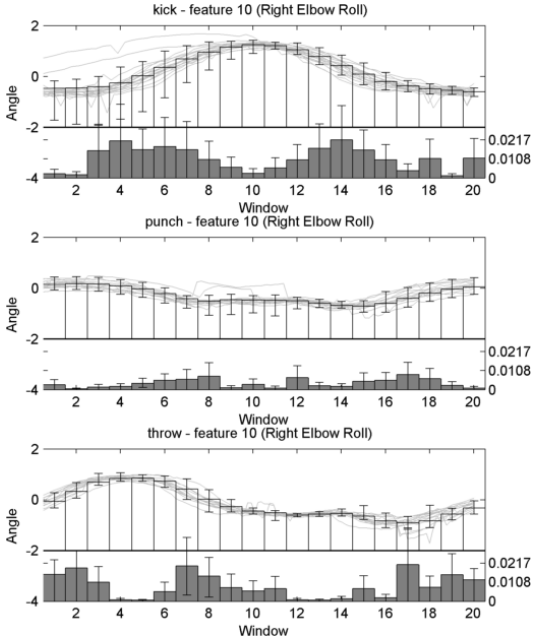


Figure D.36: Window average and variance features for right elbow roll

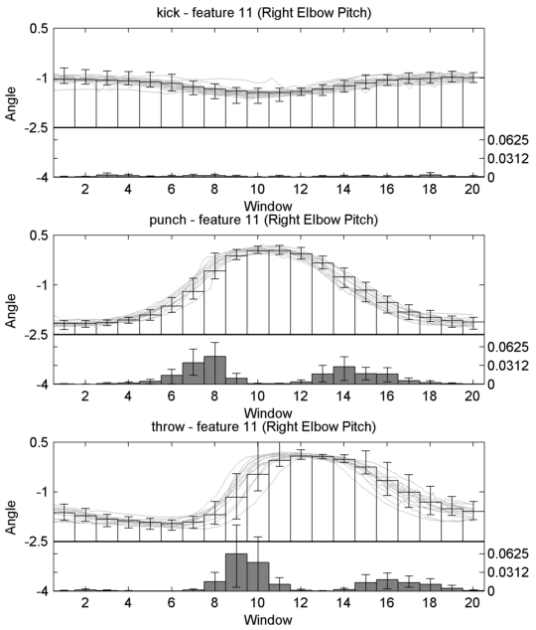


Figure D.37: Window average and variance features for right elbow pitch

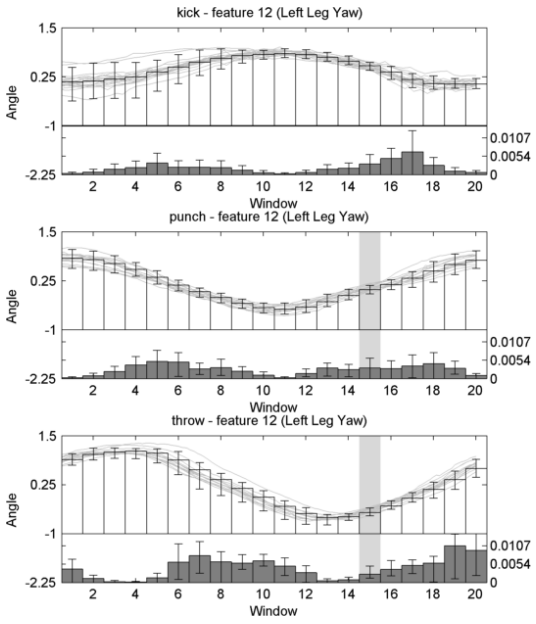


Figure D.38: Window average and variance features for left leg yaw

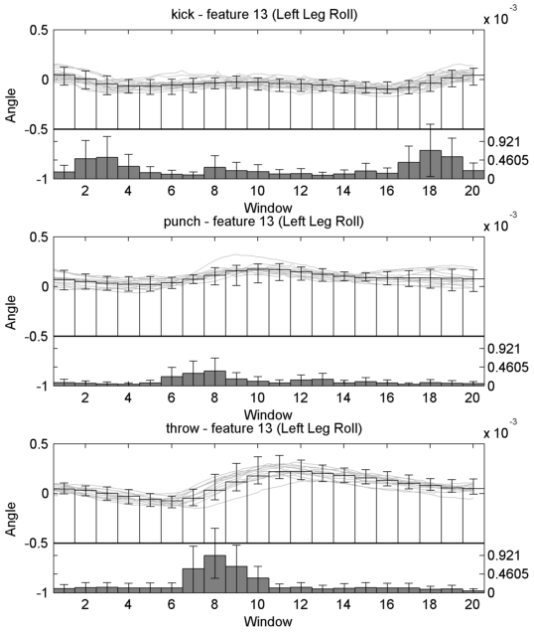


Figure D.39: Window average and variance features for left leg roll

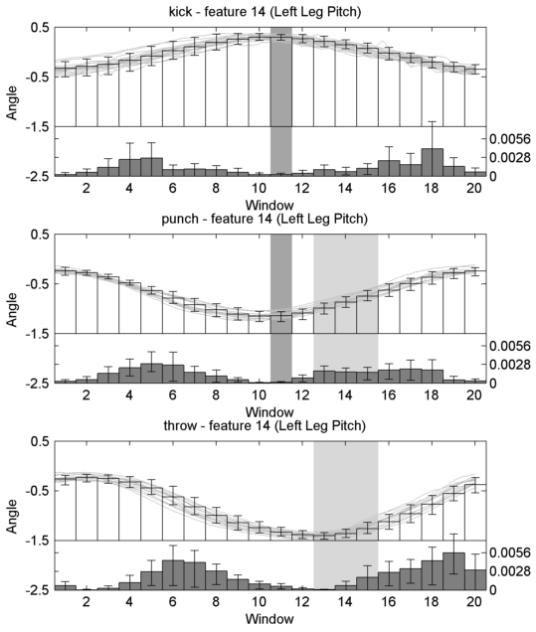


Figure D.40: Window average and variance features for left leg pitch

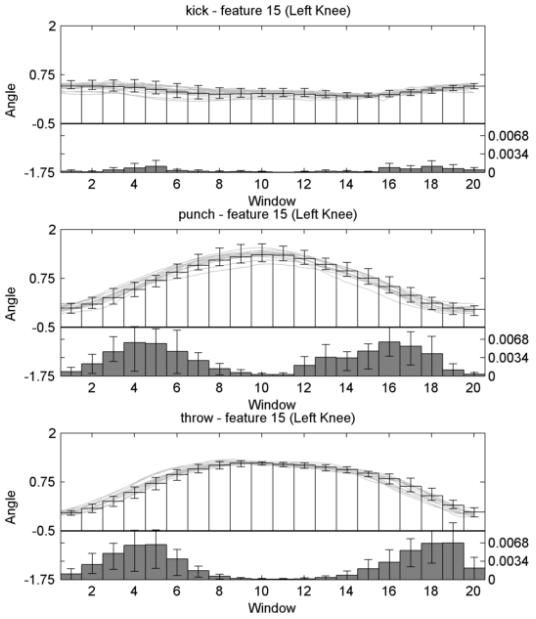


Figure D.41: Window average and variance features for left knee

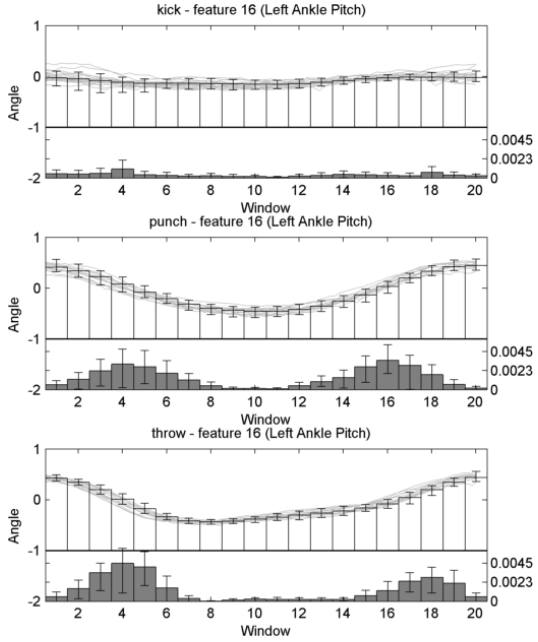


Figure D.42: Window average and variance features for left ankle pitch

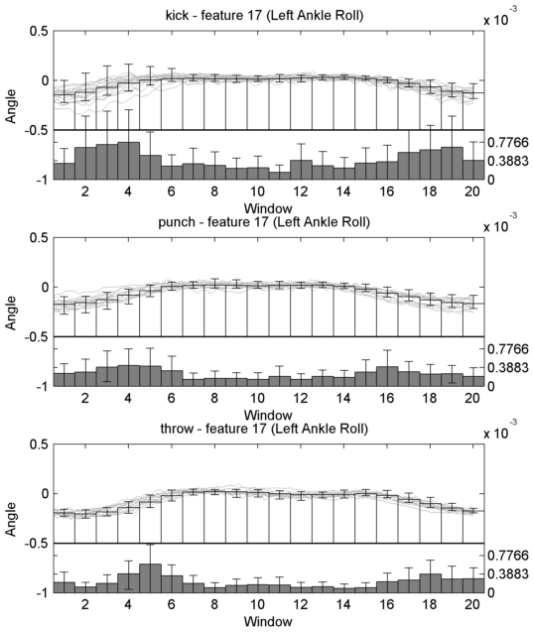


Figure D.43: Window average and variance features for left ankle roll

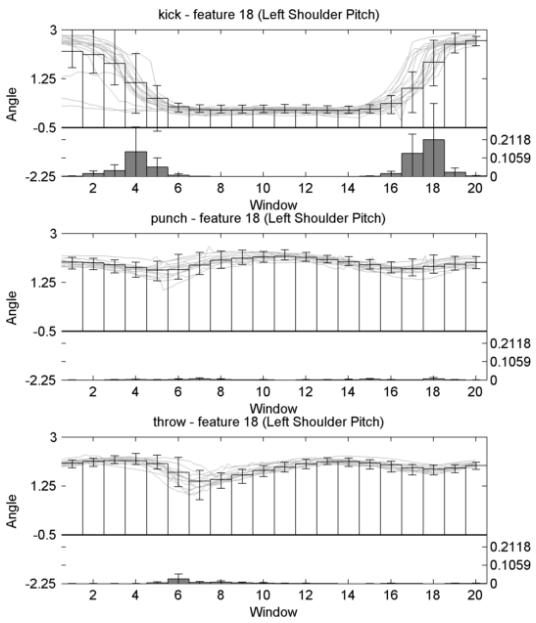


Figure D.44: Window average and variance features for left shoulder pitch

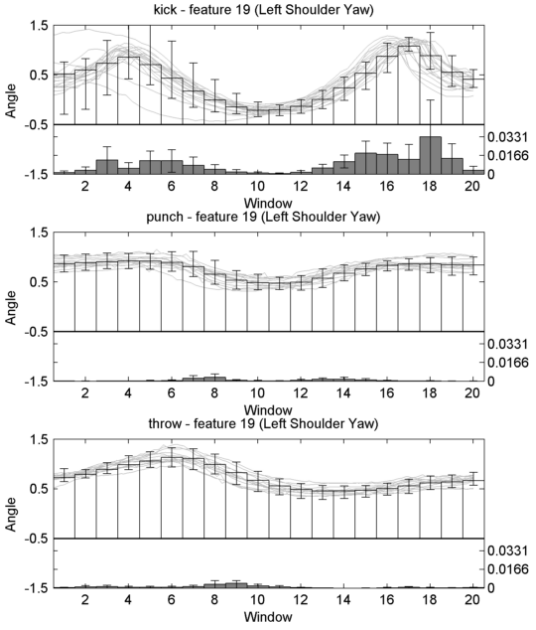


Figure D.45: Window average and variance features for left shoulder yaw

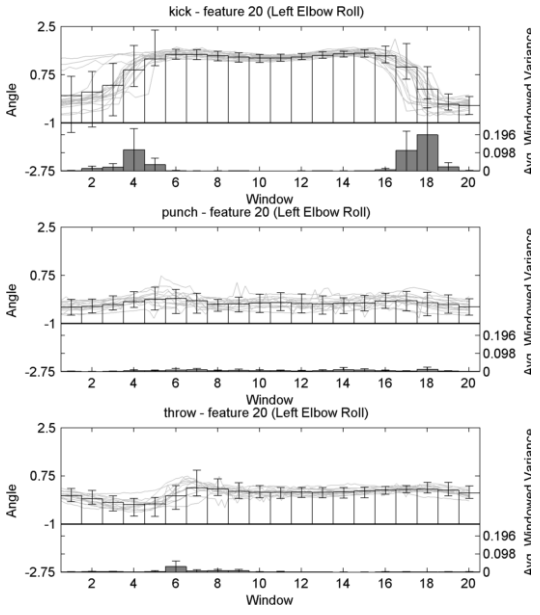


Figure D.46: Window average and variance features for left elbow roll

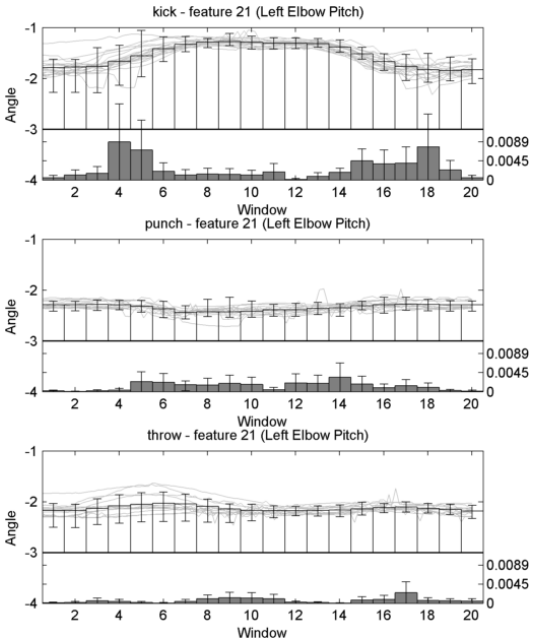


Figure D.47: Window average and variance features for left elbow pitch

References

[1] Margareta Ackerman, Shai Ben-David, and David Loker. Towards property-based classification of clustering paradigms. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 10–18. 2010.

[2] J.K. Aggarwal and Qin Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(13):4(13)28–440, 1999.

[3] Mehdi Hosseinzadeh Aghdam, Masser Ghasem-Aghaee, and Mohammad Ehsan Basiri. Text feature selection using ant colony optimization. *Expert systems with applications*, 36:6843–6853, 2009.

[4] Hussein Almuallim and Thomas G. Dietterich. Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 547–552, 1991.

[5] Ethem Alpaydin. Combined 5 x 2 cv f test for comparing supervised classification learning algorithms. *Neural Computation*, 11:1185–1892, 1999.

[6] Wilker Altidor, Taghi M. Khoshgoftaar, and Jason Van Hulse. An empirical study on wrapper-based feature ranking. In *Proceedings of the International Conference on Tools with Artificial Intelligence, ICTAI*, pages 75 – 82, Newark, NJ, United states, 2009.

[7] Antonio Arauzo-Azofra, José Manuel Benítez, and Juan Luis Castro. A feature set measure based on RELIEF. In *Proceedings of the 5th International Conference on Recent Advances in Soft Computing*, pages 104–109, 2004.

[8] Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013.

[9] Mukund Balasubramanian and Eric L. Schwartz. The Isomap algorithm and topological stability. *Science*, 295(7):7a, 2002.

[10] Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357 – 1371, 2011.

[11] Roberto Battiti. Using the mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.

[12] Lluís A. Belanche and Félix Fernando González. Review and evaluation of feature selection algorithms in synthetic problems. *CoRR*, abs/1101:2320, 2011.

[13] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[14] Anil Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943.

[15] Aude Billard, Yann Epars, Sylvain Calinon, Stefan Schaal, and Gordon Cheng. Discovering optimal imitation strategies. *Robotics and Autonomous Systems*, 47:69–77, 2004.

[16] Sebastian Bitzer and Sethu Vijayakumar. Latent spaces for dynamic movement primitives. In *Proceedings of the 9th IEEE-RAS International Conference on Humanoid Robots, HUMANOIDS09*, pages 574 – 581, Paris, France, 2009.

[17] Randolph Blake and Maggie Shiffrar. Perception of human motion. *Annual Review of Psychology*, 58(1):47–73, 2007.

[18] Ingwer Borg and Patrick J.F. Groenen. *Modern multidimensional scaling: theory and applications*. Springer series in statistics. Springer, New York, USA, 2005.

[19] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[20] Gavin Brown. A new perspective for information theoretic feature selection. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 49–56, Clearwater Beach, Florida, USA, 2009.

[21] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[22] Christopher J.C. Burges. Dimension reduction: A guided tour. *Foundations and Trends in Machine Learning*, 2(4):275–365, 2010.

[23] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 333–342, 2010.

[24] Erick Cantú-paz. Feature subset selection, class separability, and genetic algorithms. In *Genetic and Evolutionary Computation Conference*, pages 959–970. Springer-Verlag, 2004.

[25] Shahzad Cheema, Timo Henne, Uwe Koeckemann, and Erwin Prassler. Applicability of feature selection on multivariate time series data for robotic discovery. In *Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, volume 2, pages 592–597, 2010.

[26] Jin Chen, Cheng Wang, and Runsheng Wang. Combining support vector machines with a pairwise decision tree. *IEEE Geoscience and Remote Sensing Letters*, 5(3):409 –413, July 2008.

[27] Jin Chen, Cheng Wang, and Runsheng Wang. Adaptive binary tree for fast svm multiclass classification. *Neurocomputing*, 72(1315):3370 – 3375, 2009.

[28] Xiang Chen, Xu Zhang, Zhang-Yan Zhao, Ji-Hai Yang, Vuokko Lantz, and Kong-Qiao Wang. Hand gesture recognition research based on surface EMG sensors and 2D-accelerometers. In *Proceedings of the 11th IEEE International Symposium on-Wearable Computers*, pages 11–14, Oct. 2007.

[29] Sungmoon Cheong, Sang Hoon Oh, and Soo-Young Lee. Support vector machines with binary tree architecture for multi-class classification. *Neural Information Processing*, 2(3):47–51, 2004.

[30] Jongmoo Choi and Juneho Yi. Low-dimensional facial image representation using FLD and MDS. In *Advances in intelligent computing: international conference on intelligent computing, ICIC 2005*, Hefei, China, 2005.

[31] Anirban Chowdhury, Rithvik Ramadas, and Sougata Karmakar. Muscle computer interface: A review. In Amaresh Chakrabarti and Raghu V. Prakash, editors, *ICoRD'13*, Lecture Notes in Mechanical Engineering, pages 411–421. 2013.

[32] Li-Yeh Chuang, Sheng-Wei Tsai, and Cheng-Hong Yang. Improved binary particle swarm optimization using catfish effect for feature selection. *Expert Systems with Applications*, 38(10):12699 – 12707, 2011.

[33] Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

[34] Thomas M. Cover and Peter E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Throry*, IT-13(1):21 – 27, 1967.

[35] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. John Wiley & Sons Inc., Hoboken, NJ, USA, 2 edition, 2006.

[36] Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

[37] Manoranjan Dash and Huan Liu. Consistency-based search in feature selection. *Artificial Intelligence*, 151(12):155 – 176, 2003.

[38] Ingrid Daubechies. Ten lectures on wavelets. In *Proceedings of the CBMS-NSF Regional Conference Series in Applied Mathematics*, volume 61. Society for Industrial and Applied Mathematics, June 1992.

[39] Vin de Silva and Joshua B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems*, 15:705–712, 2002.

[40] Vin de Silva and Joshua B. Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, Stanford University, 2004.

[41] Kresimir Delac, Mislav Grgic, and Sonja Grgic. Independent comparative study of PCA, ICA, and LDA on the FERET data set. *International Journal of Imaging Systems and Technology*, 15(5):252–260, 2005.

[42] Li Deng and Douglas O'Shaughnessy. *Speech Processing: A dynamic and optimization-oriented approach*. Signal processing and communications series. Marcel Dekker, Inc., New York, USA, 2003.

[43] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: a statistical approach*. Prentice-Hall International, 1982.

[44] Chandra Shekhar Dhir and Soo Young Lee. Hybrid feature selection: Combining Fisher criterion and mutual information for efficient feature selection. *Lecture Notes in Computer Science*, 5506(1):613 – 620, 2009.

[45] Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorihms. *Neural Computation*, 10:18951923, 1998.

[46] Sebastian Thrun. et. al. The monk's problems - a performance comparison of different learning algorithms. Technical Report CS-CMU-91-197, Carnegie Mellon University, Dec. 1991.

[47] Chris Felton and Chia-Jiu Wang. Intelligent music classification with support vector machines. *Intelligent Engineering Systems Through Artificial Neural Networks*, 13:963 – 968, 2003.

[48] François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

[49] Ajo Fod, Maja J. Mataric, and Odest Chadwicke Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39 – 54, 2002.

[50] George Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

[51] Cecille Freeman, Dana Kulić, and Otman Basir. Joint feature selection and hierarchical classifier design. In *Proceedings of the 2011 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 1728–1734, Anchorage, AK, 2011.

[52] Cecille Freeman, Dana Kulić, and Otman Basir. Feature-selected tree-based classification. *Cybernetics, IEEE Transactions on*, 43(6):1990–2004, Dec. 2013.

[53] Cecille Freeman, Dana Kulić, and Otman Basir. Multi-modal tree-based SVM classification. In *Proceedings of the International Conference on Machine Learning and Applications*, pages 65–71, Miami Beach, FL, 2013.

[54] Iffat A. Gheyas and Leslie S. Smith. Feature subset selection in large dimensionality domains. *Pattern Recognition*, 43(1):5–13, 2010.

[55] Yue Guan and Jennifer G. Dy. Sparse probabilistic principal component analysis. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5, pages 185–192, Clearwater Beach, Florida, 2009.

[56] Pei-Fang Guo, Prabir Bhattacharya, and Nawwaf N. Kharma. Automated synthesis of feature functions for pattern detection. In *Proceedings of the 23rd Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4, 2010.

[57] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[58] Mark A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, University of Waikato, 1999.

[59] Caner Hamarat and Kemal Kilic. A genetic algorithm based feature weighting methodology. In *Proceedings of the 40th International Conference on Computers and Industrial Engineering (CIE)*, pages 1–6, 2010.

[60] Simon Haykin and Barry Van Veen. *Signals and Systems*. John Wiley and Sons, Inc., 1999.

[61] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 507–514, 2006.

[62] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems 15*, pages 833–840. MIT Press, 2003.

[63] Geoffrey E. Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[64] Bo Hjorth. EEG analysis based on time domain properties. *Electroencephalography and Clinical Neurophysiology*, 29(3):306–310, 1970.

[65] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

[66] Qi Huang. Assisted analysis of ultrasound tendon images based on neural network texture segmentation. Master's thesis, University of Guelph, 2004.

[67] Wei Huang, Shouyang Wang, Lean Yu, Yukun Bao, and Lin Wang. A new computational method of input selection for stock market forecasting with neural networks. In *Proceedings of the 6th international conference on Computational Science - Volume IV*, ICCS'06, pages 308–315, Berlin, Heidelberg, 2006.

[68] Aapo Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

[69] Aapo Hyvärinen and Erkki Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4):411–430, 2000.

[70] Takyua Inoue and Shigeo Abe. Fuzzy support vector machines for pattern classification. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2, pages 1449 – 1454, Washington, DC, 2001.

[71] Anil Jain and Douglas Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:153–158, 1997.

[72] Odest Chadwicke Jenkins and Maja J. Mataric. A spatio-temporal extension to Isomap nonlinear dimension reduction. In *Proceedings of the 21st International Conference on Machine Learning, ICML'04*, pages 441 – 448, Banff, AB, Canada, 2004.

[73] Thorsten Joachims. Making large-scale SVM learning practical. In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

[74] Richard A. Johnson. *Miller and Freund's Probability and Statistics for Engineers*, pages 337,369. Prentice-Hall Inc., 5 edition, 1994.

[75] Ian T. Jolliffe. *Principal Component Analysis*. Springer, New York, USA, 2 edition, 2002.

[76] Fakhreddine O. Karray and Clarence DeSilva. *Soft Computing and Intelligent Systems Design*. Pearson Education, 2004.

[77] James M. Kates. Classification of background noises for hearing-aid applications. *Journal of the Acoustical Society of America*, 97(1):461 – 470, 1995.

[78] Mahdi Khezri and Mehran Jahed. A novel approach to recognize hand movements via sEMG patterns. In *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBS'07*, pages 4907–4910, 2007.

[79] Kenji Kira and Larry A. Rendell. The feature selection problem: traditional methods and a new algorithm. In *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI-92)*, pages 129 – 34, Menlo Park, CA, 1992.

[80] Josef Kittler. *Feature set search Algorithms*, chapter 3, pages 41–60. Sijthoff and Noordhoff, 1978.

[81] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273 – 324, 1997.

[82] Igor Kononenko. Estimating attributes. analysis and extensions of relief. In *Proceedings of the European Conference on Machine Learning*, volume 784, pages 171 – 171, Catania, Italy, 1994.

[83] Igor Kononenko. On biases in estimating multi-valued attributes. In *Proceedings of the 14th International Joint Conference on Artificial intelligence*, volume 2 of *IJCAI'95*, pages 1034–1040, 1995.

[84] Volker Krüeger, Danica Kragic, Aleš Ude, and Christopher Geib. The meaning of action: A review on action recognition and mapping. *Advanced Robotics*, 21(13):1473–1501, 2007.

[85] Mineichi Kudo and Jack Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(1):25 – 41, 2000.

[86] D. Kulić, W. Takano, and Y. Nakamura. On-line segmentation and clustering from continuous observation of whole body motions. *IEEE Transactions on Robotics*, 25(5):1158–1166, 2009.

[87] Dana Kulić and Yoshihiko Nakamura. Incremental learning of human behaviors using hierarchical hidden markov models. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 2010.

[88] Dana Kulić, Wataru Takano, and Yoshihiko Nakamura. Incremental on-line hierarchical clustering of whole body motion patterns. In *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, pages 1016 – 1021, Jeju, Korea, 2007.

[89] Dana Kulić, Wataru Takano, and Yoshihiko Nakamura. Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. *International Journal of Robotics Research*, 27(7):761 – 784, 2008.

[90] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[91] Ludmila I. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications*, AIAP'07, pages 390–395, 2007.

[92] Nojun Kwak, Chong-Ho Choi, and Jin Choi. Feature extraction using ICA. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks ICANN 2001*, volume 2130 of *Lecture Notes in Computer Science*, pages 568–573. Springer Berlin / Heidelberg, 2001.

[93] Neil Lawrence. Gaussian process latent variable models for visualization of high dimensional data. In *Advances in Neural Information Processing Systems*, volume 15, 2004.

[94] Neil Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

[95] Feng Li, Yu Zhang, and Kening Gao. Pattern recognition of finger motion's EMG signal based on improved BP neural networks. In *Proceedings of the 2011 International Conference on Computer Science and Network Technology (ICCSNT)*, volume 2, pages 1266–1269, 2011.

[96] Tao Li, Chengliang Zhang, and Mitsunori Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20:2429–2437, 2004.

[97] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[98] Huawen Liu, Lei Liu, and Huijie Zhang. Boosting feature selection using information metric for classification. *Neurocomputing*, 73(1-3):295 – 303, 2009.

[99] Huiqing Liu, Jinyan Li, and Limsoon Wong. A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics*, 13:51–60, 2002.

[100] Yonggang Liu, Robert H. Weisberg, and Christopher N. K. Mooers. Performance evaluation of the self-organizing map for feature extraction. *Journal of Geophysical Research*, 111:C05018, 2006.

[101] Martin Lösch, Sven Schmidt-Rohr, Steffen Knoop, Stefan Vacek, and Rüdiger Dillmann. Feature set selection and optimal classifier for human activity recognition. In *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, pages 1022–1027, Jeju, Korea, Republic of, 2007.

[102] David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 7.2 edition, 2003.

[103] Gjorgji Madzarov, Dejan Gjorgjevikj, and Ivan Chorbev. A multi-class SVM classifier utilizing binary decision tree. *Informatica*, 33:233–241, 2009.

[104] Robert G. Malkin and Alex Waibel. Classifying user environment for mobile applications using linear autoencoding of ambient audio. In *Proceedings of the 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 509 – 512, 2005.

[105] Olvi L. Mangasarian and William H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1–18, 1990.

[106] Jianchang Mao, K. Moidin Mohiuddin, and Anil K. Jain. Parsimonious network design and feature selection through node pruning. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, volume 2, pages 622 –624, 1994.

[107] Thomas Marill and David M. Green. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, 9(1):11–17, 1963.

[108] Aleix M. Martínez and Avinash C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, 2001.

[109] Mahdokht Masaeli, Yan Yan, Ying Cui, Glenn Fung, and Jennifer G. Dy. Convex principal feature selection. In *Proceedings of the SIAM Internationsal Conference on Data Mining*, pages 619–628, 2010.

[110] Takamitsu Matsubara and Jun Morimoto. Bilinear modeling of emg signals to extract user-independent features for multiuser myoelectric interface. *IEEE Transactions on Biomedical Engineering*, 60(8):2205–2213, Aug. 2013.

[111] Andrew N. Melzoff and M. Keith Moore. Imitation of facial and manual festures by human neonates. *Science*, 198(4312):75–78, 1977.

[112] Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, and Klaus-Robert Müller. Fisher discriminant analysis with kernels. In *Proceedings of the 9th IEEE Signal Processing Society Workshop: Neural Networks for Signal Processing*, pages 41–48, 1999.

[113] Luis Carlos Molina, Lluís Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, pages 306–324, 2002.

[114] Pete Mowforth and Barry Shepherd. Statlog (vehicle silhouettes) data set, 1993.

[115] Songyot Nakariyakul and David P. Casasent. Improved forward floating selection algorithm for feature subset selection. In *Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR '08)*, volume 2, pages 793 –798, 2008.

[116] Songyot Nakariyakul and David P. Casasent. An improvement on floating search algorithms for feature subset selection. *Pattern Recognition*, 42:1932 – 1940, 2009.

[117] Gypsy Nandi. An enhanced approach to Las Vegas filter (LVF) feature selection algorithm. In *Proceedings of the 2nd National Conference on Emerging Trends and Applications in Computer Science (NCETACS)*, pages 1 –3, 2011.

[118] Minh Hoai Nguyen and Fernando de la Torre. Optimal feature selection for support vector machines. *Pattern Recognition*, 43(3):584–591, 2010.

[119] Feiping Nie, Heng Huang, Xiao Cai, and Chris Ding. Efficient and Robust Feature Selection via Joint L2,1-Norms Minimization. In *Advances in Neural Information Processing Systems 23*, 2010.

[120] Mark Nixon and Alberto S. Aquado. *Feature extraction and image processing*. Elsevier, 2002.

[121] Vesa Peltonen, Juha Tuomi, Anssi Klapuri, Jyri Huopaniemi, and Timo Sorsa. Computational auditory scene recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2:1941–1944, 2002.

[122] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226 – 1238, 2005.

[123] Angkoon Phinyomark, Chusak Limsakul, and Pornchai Phukpattaranont. A novel feature extraction for robust EMG pattern recognition. *Journal of Computing*, 1(1):71–80, 2009.

[124] Angkoon Phinyomark, Chusak Limsakul, and Pornchai Phukpattaranont. Application of wavelet analysis in EMG feature extraction for pattern classification. *Measurement Science Review*, 11:45–52, Jan. 2011.

[125] John C. Platt. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods*, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

[126] John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large margin DAGs for multiclass classification. In *Advances in Neural Information Processing Systems*, volume 12, page 547553. MIT-Press, Cambridge, MA, 2000.

[127] Pavel Pudil, Jana Novovicŏvá, and Josef Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119 – 1125, 1994.

[128] Sourabh Ravindran and David V. Anderson. Audio classification and scene recognition and for hearing aids. *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, 2:860 – 863, 2005.

[129] David B. Redpath and Katia Lebart. Observations on boosting feature selection. *Lecture Notes in Computer Science*, 3541:32 – 41, 2005.

[130] Juha Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.

[131] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the International Conference on Machine Learning*, 2011.

[132] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[133] Juan Diego Rodriguez, Aritz Perez, and Jose Antonio Lozano. Sensitivity analysis of K-fold cross validation in prediction error estimation. *IEEE Transasctions on Pattern Analysis and Machine Intelligence*, 32(3):569–575, 2010.

[134] Volker Roth. The generalized LASSO. *IEEE Transactions on Neural Networks*, 15(1):16 –28, 2004.

[135] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec. 2000.

[136] Stan Salvador and Philip Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

[137] Noelia Sanchez-Marono, Amparo Alonso-Betanzos, and Rosa M. Calvo-Estevez. A wrapper method for feature selection in multiple classes datasets. *Proceedings of the 10th International Work-Conference on Artificial Neural Networks*, 1:456–463, 2009.

[138] Stefan Schaal, Auke Ijspeert, and Aude Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 358:537–547, 2003.

[139] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 583–588, 1997.

[140] Xi Shao, Changsheng Xu, and Mohan S. Kankanhalli. Unsupervised classification of music genre using hidden Markov model. *Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME)*, 3:2023 – 2026, 2004.

[141] Jonathon Shlens. A tutorial on principle component analysis. Technical report, Salk Institute Systems Neurobiology Lab, New York University, 2009.

[142] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[143] Petr Somol, Jiří Grim, and Pavel Pudil. Fast dependence-aware feature selection in very-high-dimensional pattern recognition. In *Proceedings of the 2011 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, Anchorage, Alaska, USA, 2011.

[144] Petr Somol, Jana Novovicŏvá, Jiří Grim, and Pavel Pudil. Dynamic oscillating search algorithm for feature selection. In *Proceedings of the 19th International Conference on Pattern Recognition (ICPR 2008)*, pages 1–4, 2008.

[145] Petr Somol and Pavel Pudil. Oscillating search algorithms for feature selection. In *Proceedings of the International Conference on Pattern Recognition*, volume 2, pages 2406–2409, 2000.

[146] Samuel D. Stearns. On selecting features for pattern classifiers. In *Proceedings of the 3rd International Conference on Pattern Recognition*, 1976.

[147] Johan A.K. Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9:293–300, 1999.

[148] Wataru Takano and Yoshihiko Nakamura. Humanoid robot's autonomous acquisition of proto-symbols through motion segmentation. In *Proceedings of the 6th IEEE-RAS International Conference on Humanoid Robots*, pages 425–431, 2006.

[149] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[150] Robert Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[151] Robert Tibshirani, Walther Guenther, and Trevor Ha. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society, Series B*, 6(2):411–423, 2001.

[152] Michael E. Tipping and Christopher M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[153] Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[154] Naoyuki Tsuruta, Saleh K. H. Aly, Sakashi Maeda, Shin ya Takahashi, and Tsuyoshi Morimoto. Self-organizing map vs. spectral clustering on visual feature extraction for human interface. In *Proceedings of the 1st International Forum on Strategic Technology*, pages 55 –58, 2006.

[155] Laurens van der Maaten, Eric O. Postma, and H.J van den Herik. Dimensionality reduction : A comparative review. Technical report, Tilburg University, 2008.

[156] Antanas Verikas and Marija Bacauskiene. Feature selection with neural networks. *Pattern Recognition Letters*, 23(11):1323 – 1335, 2002.

[157] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395 – 416, 2007.

[158] Jing Wang and Chein-I Chang. Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 44(6):1586–1600, 2006.

[159] Xi-Zhao Wang, Qiang He, De-Gang Chen, and Daniel Yeung. A genetic algorithm for solving the inverse problem of support vector machines. *Neurocomputing*, 68:225 – 238, 2005.

[160] Xi-Zhao Wang, Shu-Xia Lu, and Jun-Hai Zhai. Fast fuzzy multicategory SVM based on support vector domain description. *International Journal of Pattern Recognition and Artificial Intelligence*, 22(1):109–120, 2008.

[161] Kilian Q. Weinberger and Lawrence K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Boston, MA, USA, 2006.

[162] Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the 21st International Conference on Machine Learning (ICML-04)*, pages 839–846. ACM Press, 2004.

[163] Jason Weston and Chris Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, University of London, 1998.

[164] Kevin R. Wheeler. Device control using gestures sensed from EMG. In *Proceedings of the 2003 IEEE International Workshop on Soft Computing in Industrial Applications (SMCia/03)*, pages 21–26, 2003.

[165] A.W. Whitney. A direct method of non-parametric measurement selection. *IEEE Transactions on Computers*, 20(9):1100–1103, 1971.

[166] Jian-Bo Yang, Kai-Quan Shen, Chong-Jin Ong, and Xiao-Ping Li. Feature selection for MLP neural network: the use of random permutation of probabilistic outputs. *IEEE Transactions on Neural Networks*, 20(12):1911–1922, 2009.

[167] Ming-Hsuan Yang. Extended Isomap for pattern classification. In *Proceedings of the 18th national conference on Artificial intelligence*, pages 224–229. American Association for Artificial Intelligence, 2002.

[168] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, pages 412–420, San Francisco, CA, 1997.

[169] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the International Conference on Machine Learning*, volume 2, pages 856–863, Washington DC, 2003.

[170] Shipeng Yu, Kai Yu, Volker Tresp, Hans peter Kriegel, and Mingrui Wu. Supervised probabilistic principal component analysis. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 464–473. ACM Press, 2006.

[171] Biying Zhang. A joint evolutionary method based on neural network for feature selection. In *Proceedings of the 2nd International Conference on Intelligent Computing Technology and Automation (ICICTA)*, volume 1, pages 7–10, 2009.

[172] Hongbin Zhang and Guangyu Sun. Feature selection using tabu search method. *Pattern Recognition*, 35(3):701 – 711, 2002.

[173] Zhenyue Zhang and Hongyuan Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26:313–338, 2002.

[174] Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. In *Neural Information Processing Systems*, page 16. MIT Press, 2003.

[175] Douglas Zongker and Anil Jain. Algorithms for feature selection: An evaluation. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 2, pages 18 – 22, 1996.

[176] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[177] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15:262–286, 2006.