TECHNISCHE UNIVERSITÄT DRESDEN

FAKULTÄT ELEKTROTECHNIK UND INFORMATIONSTECHNIK

Diplomarbeit

Thema: Examination of the applicability of Support Vector Machines in the context of ischaemia detection

Vorgelegt von: Gemma Guillaumes Sanchez

Geboren am: 22.09.1983 in Barcelona

Betreuer: Dipl.-Ing. Sebastian Zaunseder
Prof. Dr. med. habil. Dipl.-Ing. R. Poll
Verantwortlicher Hochschullehrer: Prof. Dr.-Ing. H. Malberg

Tag der Einreichung: 30. November 2010

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage eingereichte Diplomarbeit zum Thema

Examination of the applicability of Support Vector Machines in the context of ischaemia detection

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie Zitate kenntlich gemacht habe.

Dresden, den 30.11.2010

Gemma Guillaumes Sanchez

Contents

List of Tables

List of Figures

1. Introduction

1.1. Motivation

The electrocardiogram (ECG) provides useful information about the functional status of the heart. Therefore, the analysis of the ECG is of great importance in the detection of cardiac anomalies [46].

Regarding the detection of ischaemia the long-term ECG exhibits some advantages compared to classical techniques which are used to detect ischaemic modifications:

1. The ECG constitutes a non-invasive, easy and cheap applicable way to draw conclusions on the existence of ischaemia
2. Based on the ECG a monitoring is possible by which endangered persons can be continuously recorded and warnings can be generated in case of immediate problems
3. The long term ECG allows the study of people during their daily activities, since factors like mental stress are thought to be a major cause of ischaemia [16]

Owing to these advantages the detection of ischaemia from the long-term ECG is thought to be of great importance in the future [13, 15, 25, 29, 68].

Precondition is the existence of methods which are able to provide meaningful results on ischaemic events based on the long-term ECG. As the detection of ischaemic events is one of the toughest detection challenges in the field of ECG processing [66] the development of suited processing methods constitutes an interesting field. Different studies propose detectors of ischaemic episodes (e.g. [34, 65, 67]). However, the results which have been obtained until today are not satisfactory and therefore an accurate detection of ischaemic events considering the long-term ECG still constitutes a challenging task [21].

At the Institute of Biomedical Engineering (IBMT) topics related to the detection of ischaemia are also studied. The research is carried out in cooperation with the Fraunhofer Institute for Photonic Microsystems (IPMS) [7, 17, 50, 75]. The goal of the research is to extend the applicability of ambulatory monitoring devices as developed by the IPMS regarding the possibility to detect ischaemic episodes. In the feature this detection should be carried out online and in real-time in order to allow medical intervention in the case of emergencies.

However, the complexity and the variety of factors that produce modifications which resemble ischaemic beats are striking problems from which arises the need for proper classification methods.

The Support Vector Machine (SVM) is nowadays the most successful statistical pattern classifier [63]. This classifier has been used in multiple classification tasks with highly accurate results

[46, 38]. Since the SVM can manage high-dimensional and large datasets this classifier constitutes a suited choice for many tasks which are related to biomedical classification problems [11]. Facial expression classification [27], text classification [58], beat detection [46] and QRS complex classification [23] are only a few examples for successful applications of the SVM.

The idea which underlies the SVM is to find a hyperplane that divides the samples in two classes with a maximum margin between them. The SVM uses previous labeled data to find the most suitable classification border. This border has a strong dependence on just few samples of these training data, also called *support vectors*. Analyzing these *support vectors* one can extract some characteristics of the data that is going to be classified. This renders SVM not only a powerful classifier, but also allows the interpretation of its application in many cases. When the input data is not linearly separable, Kernel Methods (KMs) are used to transform the input data in a higher dimensional space where linear classification can be successfully used.

The SVM constitutes in its basic theory a supervised learning classifier. Nevertheless, some recent studies have used this classifier to develop an unsupervised learning classifying method [10, 74]. This recently proposed approach line can lead to further successful applications which are related to SVMs accounting for the importance of this method.

1.2. Specification of the topic

As stated before the detection of ischaemia was topic of previous investigations at the IBMT. Regarding the presented work the studies which have been carried out by Nauber [50] are of particular interest, aiming at detecting ischaemia based on the morphology of the ventricular repolarization (VR). Thereby, Nauber focuses on preprocessing techniques and their capacity to provide a suited basis for the detection of ischaemia (details in section 2.1.4).

The presented work now builds up directly on Nauber's work. The goal of this thesis is to take advantage of SVMs' classifying characteristics to develop an automatic detector of ischaemic events. Therefore, the European ST-T-database is used.

As there have been no previous works on SVMs in the working group, at first, some basic research on

- applications
- functioning
- proposed realizations

of SVMs has to be carried out within this work.

Based on the findings a method for beat classification regarding ischaemic beats based on the SVMs has to be implemented.

Finally, not only an evaluation on the level of beat classes but the detection of ischaemic episodes is required. However, the SVM originally only provides a classification on beat level. Therefore a scheme must be developed which is able to detect episodes based on the results of the beat classification.

Nomenclature

α_i : Lagrange multiplier

γ: Kernel parameter

γ_{opt}: Optimal γ parameter

λ_i: i^{th} basis function

Φ: Basis functon of KLT

$\phi(s)$: Mapping function

ν: Upper bound on the fraction of training errors and a lower bound of the fraction of support vectors

ξ_i : Slack variable associated with x_i

$+P$: Positive predictivity

acc_{global}: global accuracy

acc_i: Accuracy for class i

$\boldsymbol{B}$: Set of support vectors

b : Bias term of the hyperplane

bac: Balanced accuracy

b_i: ith KLT coefficient

$\boldsymbol{C}$: Covariance matrix

C: Penalizing parameter

C_{opt}: Optimal C parameter

d: margin of the SVM

d_i: distance between the hyperplane and the nearest observation of class i

FN_i: False Negatives labels for class i

FP_i: False Positives labels for class i

FP_{QRS}: Fiducial point of the QRS complex

$K(\boldsymbol{x_i}, \boldsymbol{x_j})$: Kernel function

d_i: Distance between the hyperplane and the nearest observation of class i

H: Hyperplane

H_i: Secondary hyperplanes

$\boldsymbol{H}$: Constructed matrix for QP solving problem

k: Number of classes

L: Number of binary classification problems constructed in a multi-class case

L_P: Minimizing goal function in dual formulation

L_D: Maximizing goal function in dual formulation

$l_{VR}(RR)$: Lengt h of $onset_{VR}$

$\boldsymbol{M}$: Matrix containing mean of the samplers of pattern vectors

mac: Mean accuracy

N: Number of support vectors in hard margin case

N_S: Number of support vectors in soft margin case

N_{beats}: Total number of beats

N_{sig}: Length of patter vector

N_{train}: Number of records of the training data

$onset_{VR}$: Segment of a beat which contains the complete VR

p: Kernel parameter

$\boldsymbol{P}$: Matrix containing pattern vectors

q: The dimension of the input vectors

r: Kernel parameter

RR: Distance to the next QRS-complex

$\boldsymbol{S}$: Matrix with all the support vectors

Se: Sensitivity

Sp: Specificity

$SDSe$: Duration based sensitivity

$SD+P$: Duration based positive predictive value

$SESe$: Episode based sensitivity

$SE+P$: Episdoe based positive predictive value

TN_i: True Negatives Lables for class i

TP_i: True Positives Lables for class i

TP_i: True Positives Lables of class i

$threshold1$: Upper threshold

$threshold2$: Lower threshold

$\boldsymbol{w}$: Normal to the hyperplane

W^r: rth formula to calculate w_i values

w_i^r: Weight value of the i^{th} class and r method

w_{opt}: Optimal w parameter

$\boldsymbol{X}$: Training set matrix

x: Pattern vector

$\boldsymbol{x_i}$: Training attribute vector

$\boldsymbol{x_i^s}$: Support vectors

$\boldsymbol{x'}$: Testing vector

x_{rec}: Reconstructed patter vector

$\boldsymbol{Y}$: Training set labels

y_i : Training label vector

$\boldsymbol{y'}$: Predicted label

Abbreviations

ART: Adaptive Resonance Theory

B: Balanced

BAC: Balanced Accuracy

DPC: Detector Performance Characteristic

ECG: Electrocardiogram

EDB: European ST-T Database

DAG: Directed Acyclic Graph

DWT: Discrete Wavelet Transform

IBMT: Institute of Biomedical Engineering

ICA: Independent Component Analisys

IPMS: Fraunhofer Institute of Photonic Microsystems

KLT: Karhunen-Loève-Transformation

KM: Kernel Method

L: Linear Kernel

LDA: Linear Discriminant Analysis

LLSF: Linear Least Squares Fit

LSSVM: Last Square SVM

MAC: Mean Accuracy

MC: Multiclass Classification

MLP: Multilayer Perceptron

NB: Naive Bayes

OvA: One versus All

OvO: One versus One

PCA: Principal Component Analysis

QDA: Quadratic Discriminant Analysis

QP:Quadratic Programming

ROC: Receiver-Operator-Characteristic

RBF: Radial Basis Function

SMO: Sequential Minimal Optimization

SOM: Self-Organizing Map

SVM: Support Vector Machine

VR: Ventricular Repolarization

UB: Unbalanced

2C: Binary classification

2. State of Art

2.1. Ischaemia detection from the long-term ECG

2.1.1. Underlying ideas - definition of episode

The detection of ischaemia from the long-term ECG typically is based on the assumption that an ischaemic event has to be considered as relevant when it causes a modification which is characterized by a certain duration and intensity. This leads to the definition of so-called *episodes*. An episode defines a signal segment which exhibits a relevant event. Occurrence and characteristics of such episodes are used to assess the patient state regarding the existence of an ischaemic disease [15, 69].

To define episodes typically the deviation from the normal state of one person is used (baseline-ECG). Figure 2.1 gives an overview of the proceeding which is used to define an episode based on the ST-deviation[1]. An upper threshold $thres1$ must be crossed at least for a time T_{min}. When this criterion is fulfilled an episode is existent. To delineate this episode, i.e. find onset and end of this episode, another threshold $trhes2$ is applied. This threshold must be undercut for at least 30 s to define the border of an episode. This scheme is applied in the databases which are provided to support the development of algorithms in the context of ischaemia detection.

The scheme outlined before is applicable not only for the ST-deviation, but also for other ECG based parameters. Indeed, different parameters which can be extracted from the ECG are subject to ischaemic modifications. Amongst others, the ventricular depolarization [57] and the heart rate variability [19] showed a characteristic behavior during transient ischaemic episodes. Thus, it would be possible to define episodes based on these parameters. However, the VR is considered to be the most sensitive signal portion when ischaemia is to be detected within the long-term ECG. Thus, parameters of the VR as the ST-deviation and the T-wave amplitude are

[1] The ST-deviation is the deviation of the ST-segment from the isoelectric level under consideration of some baseline deviation which can be fixed or even temporally changing

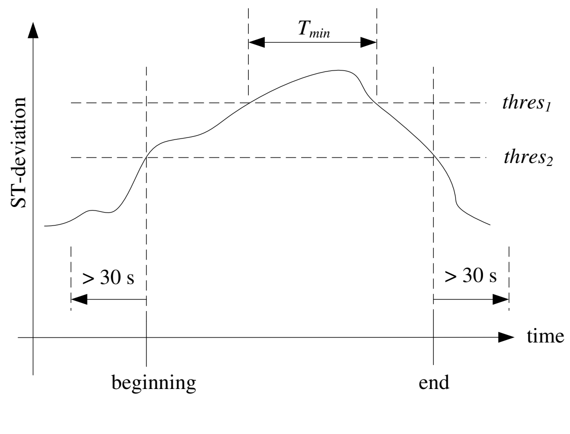


Figure 2.1.: Scheme to annotate episodes based on the registered ST-deviation

most often used in the detection of ischaemia. Even within this work, following consideration only focus on the VR-based detection of ischaemia.

2.1.2. Subtasks in ischaemia detection

When one tries to detect ischaemia from the long-term ECG, different influence factors have to be taken into account. These influence factors are able to modify the morphology of the VR which may lead to erroneous classifications when they are not correctly identified as non-ischaemic modifications. Based on the different types of influence factors, the detection of ischaemia today can be seen as three step process [75]. Each step tackles a certain aspect and produces, in conjunction with the previously given definition of an episode, its belonging output.

1. Handling of randomly occurring artifacts and noise → detection of episodes with modified ST-(T)-segment[2]
2. Consideration of slow-drifts and axis-shift → ST(-T)-episodes
3. Discrimination between ischaemic and heart-rate related ST(-T)-episodes → ischaemic episodes

Strictly speaking, only approaches which execute all three steps really try to detect ischaemia[3]. Nevertheless, to accomplish just the first two or even the first step are important tasks as they constitute the basis for the detection of ischaemia.

Further on, ischaemic episodes can be seen as a subgroup of the ST(-T)-episodes and the ST(-T)-modifications, respectively. A processing regarding these events already achieves an aggregation of information based on which the medical personal can draw conclusions on the patient state. Compared to the examination of the whole ECG such a processing already means a simplification and is of high interest. More details on the subtasks and their execution can be found in [75].

2.1.3. Methodological considerations

In the literature a huge amount of methods to detect ischaemia and related modifications, i.e. ST(-T)-modifications and ST(-T)-episodes, respectively, from the long-term ECG can be found. Even different methods to categorize the approaches have been devised. Most of them are based on the computational paradigm underlying each of the methods. Differing from this, an own partitioning separates the methods based on the feature on which the occurrence of ischaemia is detected [75]. This idea leads to three approaches:

1. Univariate detection: the ST-deviation is used
2. Discrete multivariate detection: different discrete features from the VR are used (e.g. ST-deviation, ST-slope, T-amplitude,...)
3. (Continuous) multivariate detection: entire signal segments (parts of the VR or the whole VR) are used (most often in conjunction with a transform for dimension reduction)

[2] From here when the term "ST(-T)-modifications" is used always episodes are meant

[3] In the literature typically there is not differentiation between these aspects; especially in the case of works which consider more than the first step this can lead to confusions

[75] gives a comprehensive overview on methods and results of approaches which have been devised in the literature.

2.1.4. Previous own works

Different studies accomplished within the joint research activities of IPMS and IBMT already tackled problems related to the detection of ischaemia from the long-term ECG. These studies include methods belonging to the group of univariate algorithms, i.e. methods which try to detect ischaemia based on the ST-deviation [7, 17, 75] as well as approaches which rely on the multivariate detection [30, 50, 51].

Recently, Nauber [50] focused on different methods to extract and scale the VR in dependency to varying heart rates. The focus of this work was the preparation of suited feature vectors based on which the detection of ST-T-modifications can be done. More details regarding the method of Nauber which serves as basis for this work are contained in chapter 4.2.2. The detection/classification itself, however, was not a central aspect of Nauber's work. Thus, a comparatively elementary approach was used for the classification. This approach does the classification based on the euclidean distance between the beat under examination and a reference template.

According to the topic of this work of major interest are the methods which rely on a multivariate detection. However, details on signal segmentation and transformation which are of great importance in the process of multivariate ischaemia detection are not topic of this work, as they have been tackled by own previous works. Moreover this, speaking in the introduced nomenclature, this work focuses exclusively on the detection of ST-T-modifications. However, owing to this limitation and in order to simplify the nomenclature from here the expression *ischaemic episodes* is used even when only ST-T-modifications are detected.

2.2. Classification Methods and SVM in ECG processing

2.2.1. Classification Methods

Classification methods use a set of *features* or *parameters* to characterize given objects. Classification methods can be grouped in two main categories: *supervised* and *unsupervised*. Methods for *supervised* classification are those where a set of objects with known class membership is used for training. *Unsupervised* classifications have no a priori knowledge of the memberships[9, 53].

2.2.1.1. Methods for supervised classification

In supervised classification methods the set of known objects is called *training set*. Each object of the training set consists of an feature vector and a belonging class value. Based on the training data the supervised learning algorithm extracts a decision function to classify unknown input data. If the output of the decision function is a discrete value this function is known as *inferred function*. Otherwise, if the output is continuous the function is called *regression function* [9, 53].

Some widely used examples of supervised classification methods are:

- Neural networks
- Nearest-neighbor classifiers

- Linear least squares fit (LLSF)
- Naive Bayes (NB) Cassifiers
- SVMs

2.2.1.2. Methods for unsupervised classification

In unsupervised classification methods one seeks to determine how the data is organized. This type of method is usually related with density estimation [9, 53].

Some examples of unsupervised classification methods are:

- Neural networks
- Partitioning clustering techniques
- Hierarchical clustering techniques
- Linear discriminant analysis (LDA)
- Quadratic discriminant analysis (QDA)
- Self-organizing maps (SOMs)

2.2.2. SVMs in ECG processing

Although there are approaches in which the ideas which underlies the SVM are used in the context of unsupervised learning [10, 74], the SVM in its original version is a supervised classification method.

In the field of ECG processing, the usage of SVMs becomes more and more popular in recent times. The main reasons, first and foremost the convincing results, already have been outlined in chapter 1. Table 2.1 summarizes the application of SVM by giving some selected examples.

By analyzing the given examples the actual relevance of SVMs becomes obvious. Moreover this, table 2.1 gives details on the most important facts of the applied SVMs. Those facts include

- the type of SVM formulation used
- the Kernels which are used
- the multiclass classification strategy
- the feature vector extraction

For information on the theoretical basis of the depicted facts readers are referred to chapter 3.

Table 2.1.: Selected examples regarding SVM usage in the field of ECG processing

Reference	Aim of the study	SVM Details	Remarks
Acir [6]	Classification of ECG beats	Last Square SVM (LSSVM), Radial Basis Function (RBF) Kernel, compares 5 different feature extraction	Standard backpropagation Multilayer Perceptron (MLP) networks
Diaz [24]	Predicting Spontaneous Termination of Paroxysmal Atrial Fibrillation Episodes	Polynomial Kernel, compare different feature combination	
Ghosh [28]	Cardiac Abnormality Detection	Compare different Kernels: Linear, RBF, Polynomial and Neural	RBF best performance Kernel, Optimal parameters found by trial and error
Jankowski [35]	Morphological Analysis of ECG	RBF and Directed Acyclic Graph SVM (DAGSVM) for multiclass classification	Use a faster algorithm to solve the Quadratic Problem (QP) of SVM: Sequential Minimal Optimization (SMO)
Joshi [37]	Arrhythmia Classification	RBF, Unbalanced data, Hybrid arrangement of binary and multiclass SVMs	LibSVM toolbox used
Khadtare [38]	Arrhythmia Classification	RBF kernel, compare One versus One (OvO) and DAGSVM strategies	DAGSVM most suitable algorithm
Khandoker [39]	Recognition of Obstructive Sleep Apnea Syndrome	Compare Linear, Polynomial and RBF Kernels	Polynomial Kernel best performance, MatLab SVM Toolbox used
Kontilla [40]	Classification of Acute Myocardial Ischaemia	Linear Kernel	Reduced Sets of Body Surface Potential Map Electrodes
Lannoy [23]	Feature relevance in heart beat classification	OvO, Several Kernels evaluated, weighted SVM	Linear Kernel best performance
Lei [42]	Automatic ECG Interpretation	RBF Kernel, One versus All (OvA) strategy	Optimal parameters found by trial and error
Mehta [45]	QRS complex, P and T waves detection	Sigmoid Kernel, SMO	LIBSVM software, optimal parameters found by cross validation
Mehta [46]	Beat Detection	Sigmoid kernel, modified SMO algorithm	LIBSVM software
Melgani [47]	Classification of ECG Signals	Several Kernel tried: Linear, RBF and Polynomial, OvA	Most suitable kernel: RBF
Mohebbi [48]	Detection of Ischaemic ECG Beats	RBF Kernel	
Nasiri [49]	Arrhythmia Classification	OvO	Principal Component Analysis (PCA) method for reducing the dimensions of the data
Osowski [52]	Heartbeat Recognition	OvO Multiclass SVM, RBF	Combining the SVM network with these preprocessing methods yields two neural classifiers
Park [56]	Heartbeat Classification	Hierarchical classification method using SVM	
Rojo-Alvarez [59]	Heart Rate Turbulence Denoising	SVM interpolation algorithm	
Soman [66]	Ischaemia classification	Proximal SVMs based decision trees and Polynomial Kernel	
Übeyli [12]	Heartbeat classification	RBF Kernel	Feature extraction by discrete wavelet transform (DWT), SVM get better performance than MLP Neural Network
Uyar [71]	Arrhythmia Classification	Several Kernel tried: Linear, RBF, Polynomial and Sigmoid, Fusion of SVM and Logistic Regression	Most suitable Kernel: RBF, LIBSVM software used
Zhang [77]	ECG Analysis	RBF Kernel	Use PCA to simplifying complex data sets

2.3. SVM implementations

Regarding the usage of SVMs there is some useful software available. Famous implementations are:

- SVM-Toolbox by Albrecht [8]
- SVM-Light by Joachims [36]
- MySVM by Rüping [60]
- LibSVM by Chang and Lin [20]
- MatLab-SVM-Toolbox by Gunn [31]
- LaSVM by Bordes [14]

Besides SVM specific software there are libraries/toolboxes which are dedicated to signal processing in a more general sense but include some SVM routines. Amongst others these are:

- Gait-CAD by Burmeister [18]
- MatLab Math Works [1]
- DTREG by Sherrod[62]
- PRTools [2]
- Tiberius [3]

In table 2.2 some of the toolboxes and their characteristics are summarized.

Table 2.2.: Examples of SVM software

Reference/Software Name	**Characteristics of the Implementation**
Albrecht [8] SVM Toolbox	MatLab functions for transductive and inductive learning 2D-Visualization Accuracy determination Heuristic parameter search OvA and OvO Multiclassification Kernels: linear, polynomial, RBF, Sigmoidal
Bordes [14] LaSVM	C library functions for approximate SVM solver Uses online approximation Kernel classifier that modifies its hypothesis as new training instances become available Requires considerably less memory Soft margin SVM formulation Reorganization of SMO
Burmeister [18] Gait-CAD	MatLab classification Toolbox Supported methods: fuzzy systems, artificial neural networks, SVMs, statistical methods, feature extraction methods
Chang and Lin [20] LibSVM	C++ and Java implementation C-SVC, ν-SVC, One-class SVM, ϵ-SVR and ν-SVR Kernels: linear, polynomial, RBF, Sigmoidal cross-validation and automatic model selection
Gunn [31] MatLab SVM Toolbox	SVM classification and regression Kernels: linear, polynomial, RBF, exponential RBF, Sigmoidal, Fourier series, Spline, B spline, Additive Kernels and Tensor Product
Joachims [36] SVM light	C implementation Solves classification and regression problems α-estimates of the error rate, the precision and the recall Kernels: linear, polynomial, RBF, Sigmoidal
Rüping [60] MySVM	C++ implementation SVM classification and regression Kernels: linear, polynomial, RBF and Sigmoidal
MatLab Math Works [1]	Own MatLab Toolbox Classification and statistical learning tools Cross-validation Compare different classification methods SVM and K-nearest neighbor classifiers Selecting diversity and discriminating features
[2] PRTools	MatLab based toolbox for pattern recognition Methods for: data generation, training classifiers, combining classifiers, features selection, linear and non-linear feature extraction, density estimation, cluster analysis, evaluation and visualization.
Sherrod [62] DTREG	Software for predictive modeling and forecasting Methods: SVM, Decision Trees, Boosted Decision Trees, Decision Tree Forests, MLP Neural Networks, RBF Neural Networks, Polynomial Neural Networks, Cascade Correlation Neural Networks, Probabilistic Neural Networks, K-Means Clustering, LDA, Linear and Logistic Regression
[3] Tiberius	Predicting modeling software Methods: SVM, Neural Networks, Decision Tree, Logic Regression, Regression Splines, Automatic scorecard building algorithms

3. Principles of SVMs

3.1. Introduction to SVMs

SVM is a training algorithm for learning classification and regression rules from data. It was first introduced in 1990s by Vapnik and Chervonenkis in the book: *The Nature of Statistical Learning Theory* [72]. SVMs, if necessary combined with KMs, have become popular to solve classification problems due to their high accuracy, their ability to deal with high-dimensional and large datasets as well as their flexibility in modeling diverse sources of data [11].

The main idea of applying SVMs is to find the separating hyperplane between two classes that maximizes the margin between them. This border is found by using previously labeled data. After finding this border, a classification function is extracted. Depending on the value that this function takes, previously unknown data then can be assigned to a certain class. This formulation results in a *linear hard margin SVM*.

To use the SVM as generalized classifier two considerations have to be taken into account beside the given basic principle:

1. It must be possible to apply the SVM even when data is not linearly separable in the input space

2. It must be possible to apply the SVM to multi-class problems

Regarding the first aspect there are two strategies. On the one hand, if the training data is not linearly separable in the input space erroneous classifications during the training are tolerated under certain circumstances. This expansion allows to construct a separating hyperplane even in such case where linear separability is not existing (called *soft margin SVM*). On the other hand, a nonlinear mapping onto a high dimensional space to achieve separability can be applied. This can be done with the help of a Kernel Method (called *non-linear SVM*).

Regarding the second aspect the it multi-class problem is built up directly on binary SVMs. Multi-class classification thereby is reached by combining single binary SVMs (called *multiclass SVM*).

The easiest example that one can formulate using a SVM is a binary, linear classification. The following considerations which are based on [5] will start from using this assumption. Afterwards, the basic ideas are extrapolated to the non-trivial cases of nonlinear classification and multi-class classification of non-separable data.

3.2. Binary Classification

3.2.1. Linear Classification

In a SVM formulation, a set of examples $\boldsymbol{X} = \left\{\boldsymbol{x_i} | i \in \{1, \ldots, n\} \wedge \boldsymbol{x_i} \in \mathbb{R}^q\right\}$ with known class membership $\boldsymbol{Y} = \left\{\boldsymbol{y_i} | i \in \{1, \ldots, n\} \wedge \boldsymbol{y_i} \in \mathbb{R}\right\}$ is used to train the classifier. Thereby, q is the dimension of the input vectors. In a dichotomous case there are only two classes, i.e. Class 1 and Class 2, respectively. Then, the classes can be labeled, for instance, as $y_i = 1$ for Class 1 and $y_i = -1$ for Class 2.

Consider the case where $\boldsymbol{X}$ is linearly separable and $\boldsymbol{x_i}$ has only two feature components. One can place these examples in a two dimensional plane as is shown in figure 3.1. The best boundary that one can intuitively trace will be placed as far as possible of both classes. This is formally called *Hard Margin*.

3.2.1.1. Hard Margin

If the data is linearly separable, a hyperplane that maximizes the margin between two classes can be defined as (see figure 3.1):

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{3.1}$$

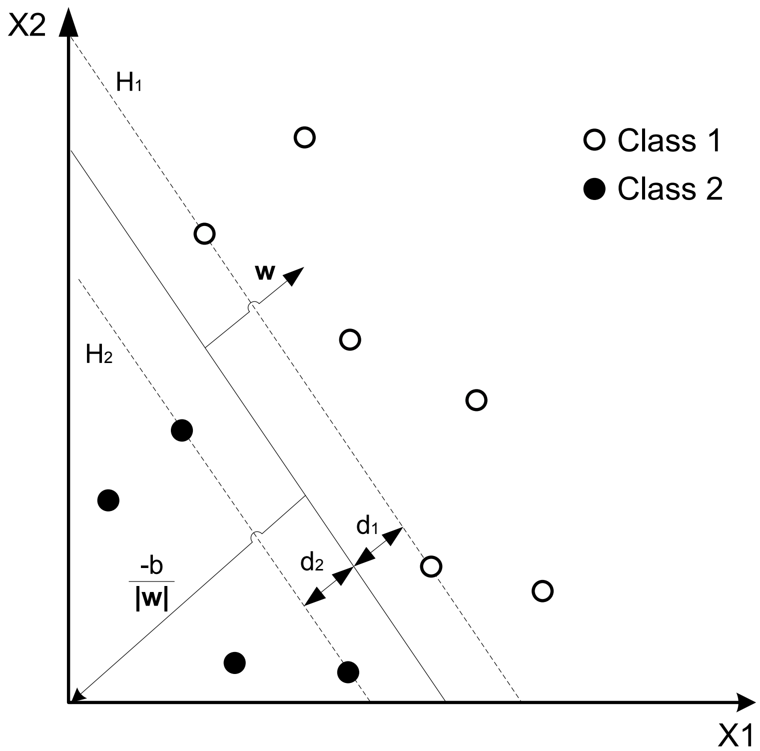


Figure 3.1.: Hard Margin classification. Based on a illustration in [5]

Where $\boldsymbol{w}, \boldsymbol{x} \in \mathbb{R}^d$ and $b \in \mathbb{R}$. $\boldsymbol{w}$ is the normal to the hyperplane, $b / \|\boldsymbol{w}\|$ is the perpendicular distance from the hyperplane to the origin, $\|\boldsymbol{w}\|$ is the Euclidean norm of $\boldsymbol{w}$ and d_1 and d_2 are the distance between the hyperplane and the nearest observation of each class.

To obtain the optimal separating hyperplane H, it has to be considered that the training data satisfies:

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1 \qquad \forall i \tag{3.2}$$

Then, the optimal separating hyperplane can be obtained by minimizing the norm of $\boldsymbol{w}$ by adding the condition of equation (3.2):

$$\min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2 \tag{3.3}$$

Two secondary hyperplanes (H_1 and H_2) can be defined. These hyperplanes are parallel and between them there are no training points. The training points that are on the hyperplane $H_1 = \boldsymbol{w} \cdot \boldsymbol{x_i} + b = 1$ and on the hyperplane $H_2 = \boldsymbol{w} \cdot \boldsymbol{x_i} + b = -1$ are called *support vectors* $\boldsymbol{x_i^s}$. All the $\boldsymbol{x_i^s}$ are grouped in the matrix S. This set of *support vectors* are the training examples situated nearest to the optimal hyperplane. Finally one can take off all the other training points and leave only the *support vectors* and the optimal hyperplane will not change.

The margin d of the SVM is calculated as the distance between both secondary hyperplane H_1 and H_2. One can identify that the distance from a positive support vector to the origin is $\frac{|1-b|}{\|\boldsymbol{w}\|}$. On the other hand, the distance from a negative support vector to the origin is $\frac{|-1-b|}{\|\boldsymbol{w}\|}$. Consequently, the distance from a positive support vector to the optimal hyperplane is $d_1 = \frac{1}{\|\boldsymbol{w}\|}$ as the distance from a negative support vector to the optimal hyperplane. The margin of the SVM will be the sum of this both distances $d = \frac{2}{\|\boldsymbol{w}\|}$ as is represented in figure 3.1.

After determining the most suitable hyperplane, new data $\boldsymbol{x'}$ will be classified by evaluating $y' = sgn(\boldsymbol{w} \cdot \boldsymbol{x'} + b)$ as:

$\boldsymbol{w} \cdot \boldsymbol{x'} + b > 0$, $\boldsymbol{x'}$ is classified as Class 1 $\Rightarrow y' = +1$.

$\boldsymbol{w} \cdot \boldsymbol{x'} + b < 0$, $\boldsymbol{x'}$ is classified as Class 2 $\Rightarrow y' = -1$.

$\boldsymbol{w} \cdot \boldsymbol{x'} + b = 0$, $\boldsymbol{x'}$ is data unclassifiable.

The optimization task which was depicted in equation 3.3 constraint to equation 3.2 can be formulated in terms of Lagrange multipliers. This is very useful to reformulate the restriction equation, equation 3.2, to be easily manageable. Also, using Lagrange multipliers, the training points will only appear as a scalar product of vectors. This characteristic will be useful in the nonlinear classification case. This formulation is called *Dual Formulation*.

Dual Formulation

The function of this new formulation [5] is build up by adding a positive Lagrange multiplier α_i for each restriction 3.2 and subtracting them from the goal function 3.3. With N as the number of support vectors this results in:

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i(\boldsymbol{x_i} \cdot \boldsymbol{w} + b) + \sum_{i=1}^{N} \alpha_i \tag{3.4}$$

Now it is needed to minimize L_P in respect of $\boldsymbol{w}$ and b, assuring that the gradient of L_P will be 0, and the Lagrange multipliers are positive defined, $\alpha_i \geq 0$.

$$\frac{\partial L_P}{\partial \boldsymbol{w}} = 0 \tag{3.5}$$

$$\frac{\partial L_P}{\partial b} = 0 \tag{3.6}$$

This leads to two conditions:

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x_i} \tag{3.7}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.8}$$

Substituting these conditions in 3.4 results in:

$$L_D \equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i} \cdot \boldsymbol{x_j} \qquad \text{s.t} \qquad \alpha_i \geq 0 \ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.9}$$

$$\equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \boldsymbol{H}_{ij} \alpha_j \qquad \text{where} \qquad \boldsymbol{H_{ij}} \equiv y_i y_j \boldsymbol{x}_i \cdot \boldsymbol{x}_j \tag{3.10}$$

$$\equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\boldsymbol{\alpha^T H \alpha} \qquad \text{s.t} \qquad \alpha_i \geq 0 \ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.11}$$

Having moved from minimizing L_P to maximizing L_D, the convex QP that is needed to solve is:

$$\max_{\alpha} \left[\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\boldsymbol{\alpha^T H \alpha}\right] \qquad \text{s.t} \qquad \alpha_i \geq 0 \forall i \qquad \text{and} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.12}$$

The solution of 3.12 is a vector with all the Lagrange multipliers $\boldsymbol{\alpha}$. Each Lagrange multiplier corresponds to a training vector. Those which are greater than 0 ($\alpha_i > 0$) are *support vectors* and are situated on the secondary hyperplanes H_1 and H_2, respectively. The other Lagrange multipliers will be 0.

Finally, $\boldsymbol{w}$ is expressed as a combination of these support vectors 3.7. Parameter b will be calculated as the average of all the equations:

$$b = \frac{1}{N}\sum_{s \in \boldsymbol{S}}(y_s - \sum_{m \in \boldsymbol{S}} \alpha_m y_m \boldsymbol{x_m} \cdot \boldsymbol{x_i^s}) \tag{3.13}$$

Application Example

Consider the case where the input data has only one feature component and 2 defined classes (see figure 3.2) [5]. Take as training observations the vectors:

$$x_1 = -1, y_1 = 1 \tag{3.14}$$
$$x_2 = 0, y_2 = 1 \tag{3.15}$$
$$x_3 = 1, y_3 = 1 \tag{3.16}$$

Substituting this training points in equation 3.2 the restriction equations follow:

$$-w + b \geq 1 \tag{3.17}$$
$$-b \geq 1 \tag{3.18}$$
$$-w - b \geq 1 \tag{3.19}$$

The solution that minimizes $\|w\|^2$ is given by:

$$b = -1 \tag{3.20}$$
$$w = -2 \tag{3.21}$$

If those values are substituted in equation 3.1, the decision function that is obtained is:

$$y' = -2x' - 1 \tag{3.22}$$

In this example, if the decision function is equal to 0, the decision boundary which is obtained is fixed in $x = -1/2$. One has to take into account that as the support vectors of this example are x_1 and x_2, x_3 can be removed from the formulation and the decision boundary will not change.

This example can be rewritten in a dual formulation. Now, the equation that has to be maximized is:

$$L_D = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(\alpha_1 + \alpha_3)^2 \tag{3.23}$$

Subject to:

$$\alpha_1 - \alpha_2 - \alpha_3 = 0 \text{ and } \alpha_i \geq 0 \tag{3.24}$$

The solution is:

$$\alpha_1 = 2 \ \alpha_2 = 2 \ \alpha_3 = 0 \tag{3.25}$$

Substituting these values in the equation 3.7, one obtains the same solutions as the primary problem.

3.2.1.2. Soft Margin

When the input data is not linear separable the *Hard Margin* SVM is unsolvable. In these cases it is possible to introduce a non-negative slack variable $\xi_i (\geq 0)$. This results in a *Soft Margin* SVM for linear classification (see Figure 3.3). For this type of margin one assumes that some points of the training data will be wrongly classified which allows to find a solution without using a mapping function.

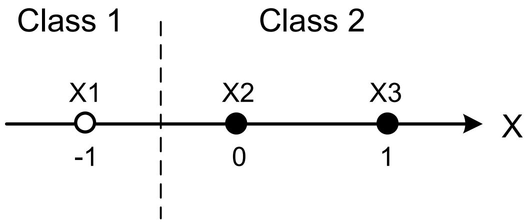


Figure 3.2.: Example of SVM Hard Margin classification. Based on an illustration in [5]

This modifies the formula 3.2 to:

$$y_i(\boldsymbol{x_i} \cdot \boldsymbol{w} + b) \geq 1 - \xi_i \qquad \xi_i \geq 0 \ \forall i \tag{3.26}$$

Using a *Soft Margin* the points that are wrongly classified are penalized using a linear combination in the goal function. The parameter C is used to trade-off between the maximization of the margin and minimization of the classification error. Now, one can obtain the optimal separating hyperplane by minimizing [5]:

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i \qquad \text{s.t} \qquad y_i(\boldsymbol{x_i} \cdot \boldsymbol{w} + b) - 1 + \xi_i \geq 0 \qquad \forall i \tag{3.27}$$

C can be adjusted to penalize more or less the errors which are allowed. If C takes a big value, the penalization is higher. One must take into account that $\xi_i = 0$ for those cases in which the classification is correct. Consequently, just these points that are misclassified will be affected by the parameter C, where ξ_i will take a value grater than 1.

Now the support vectors used in the formulation include the training points situated on the hyperplanes H_1 and H_2 and the training points between them. Where N_S is the number of the support vectors.

Dual Formulation

The *Soft Margin* formulation also can be rewritten in a *Dual Formulation* [22]. In this formulation it does not appear neither ξ_i or the Lagrange multipliers associated with ξ_i. Consequently, as in the hard margin case, the dual formulation is easier to handle than the primary formulation. Analogue to formula equation 3.4 in the hard margin case, the primary formulation in the soft margin case is:

$$L_P = \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{N_S} \xi_i - \sum_{i=1}^{N_S} \alpha_i \left[y_i(\boldsymbol{x_i} \cdot \boldsymbol{w} + b) - 1 + \xi_i \right] - \sum_{i=1}^{N_S} \mu_i \xi_i \tag{3.28}$$

This formula is to be minimized with respect to $\boldsymbol{w}$, b and ξ_i with $\mu_i \geq 0$:

$$\frac{\partial L_P}{\partial \boldsymbol{w}} = 0 \tag{3.29}$$

$$\frac{\partial L_P}{\partial b} = 0 \tag{3.30}$$

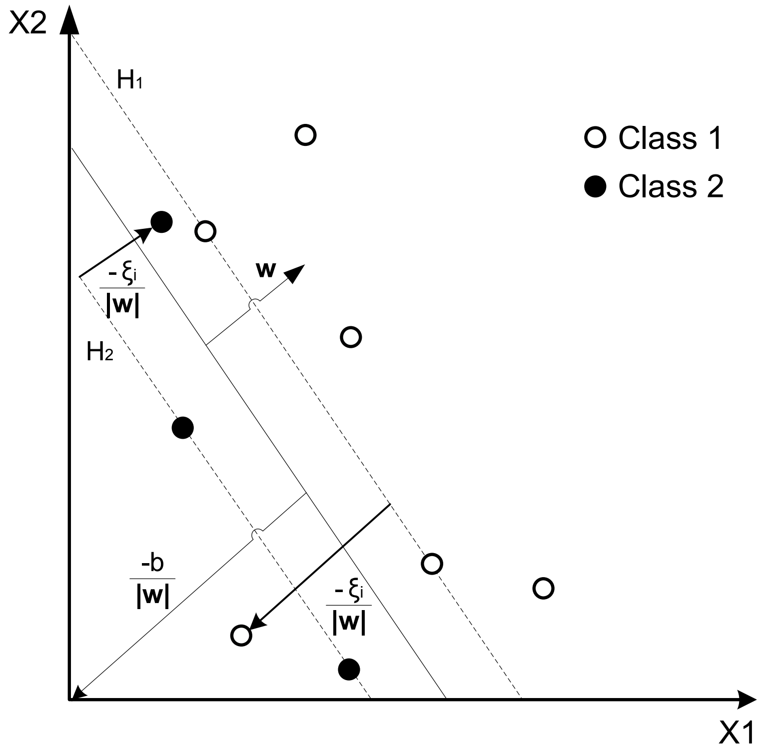


Figure 3.3.: Soft Margin classification. Based on a illustration in [5]

$$\frac{\partial L_P}{\partial \xi_i} = 0 \tag{3.31}$$

This leads to three conditions:

$$\boldsymbol{w} = \sum_{i=1}^{N_S} \alpha_i y_i \boldsymbol{x_i} \tag{3.32}$$

$$\sum_{i=1}^{N_S} \alpha_i y_i = 0 \tag{3.33}$$

$$C = \alpha_i + \mu_i \tag{3.34}$$

Moving from minimizing L_P to maximizing L_D, the convex QP that is needed to solve in soft margin case is:

$$\max_{\alpha} \left[\sum_{i=1}^{N_S} \alpha_i - \frac{1}{2} \boldsymbol{\alpha^T H \alpha} \right] \qquad \text{s.t} \qquad 0 \leq \alpha_i \leq C \ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N_S} \alpha_i y_i = 0 \tag{3.35}$$

The solution of 3.35 is also a vector with all Lagrange multipliers $\boldsymbol{\alpha}$. Each Lagrange multiplier corresponds to a training vector. Those α_i which take values $0 < \alpha < C$ are *support vectors.*

In the soft margin case $\boldsymbol{w}$ is calculated as in equation3.7 and b as in equation3.13 where N is now N_S.

3.2.2. Non-linear Classification

In many applications, the data is not linearly separable and it is not possible to obtain proper classification results using a linear SVM. In these cases the definition of a suitable mapping function $\phi : \mathbb{R}^n \longrightarrow \mathbb{R}^{n^+}$ (where $n^+ > n$) to transform the input data to a high dimensional feature space where the data is linearly separable is needed. Afterwards, a linear SVM is applied in this new space to find the separating hyperplane (see figure 3.4).

Those methods, referred to as Kernel Methods (KMs), are applied by an inner product of a nonlinear mapping function $\phi(x)$ that exports the data from an input space X to a high dimensional feature space Z. This new space is commonly known as *Hilbert* space.

The Kernel functions are statistically seen as covariance. They must be symmetric functions and positive defined. One can use any operation of transformation as a Kernel if it obey the *Cover* Theorem [22] about separability of patterns. Furthermore, this transformation must be able to express it as a scalar product of vectors in the output space ($K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \phi(\boldsymbol{x_i}) \cdot \phi(\boldsymbol{x_j})$). The *Mercer*'s condition [5] is used to prove it. Any Kernel that obey those conditions can be used in the training by substitute all the $\boldsymbol{x_i} \cdot \boldsymbol{x_j}$ for $K(\boldsymbol{x_i}, \boldsymbol{x_j})$.

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \phi(\boldsymbol{x_i}) \cdot \phi(\boldsymbol{x_j}) \tag{3.36}$$

The convex QP which must be solved in this case is:

$$\max_{\alpha} \left[\sum_{i=1}^{N_S} \alpha_i - \frac{1}{2} \boldsymbol{\alpha^T H \alpha} \right] \qquad \text{s.t} \qquad 0 \leq \alpha_i \leq C \ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N_S} \alpha_i y_i = 0 \tag{3.37}$$

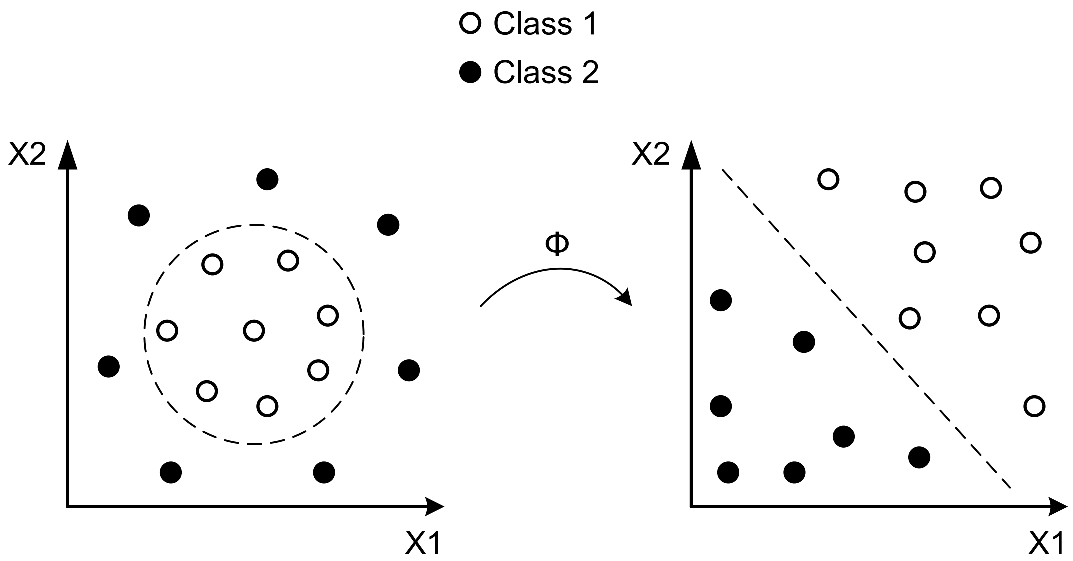


Figure 3.4.: Mapping data. Based on a illustration in [5]

Where $\boldsymbol{H}$ is now:

$$\boldsymbol{H_{ij}} = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) = y_i y_j \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j) \tag{3.38}$$

Different Kernel functions have been depicted in the literature. The most widely used Kernel functions are :

Linear:

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = (\boldsymbol{x_i}) \cdot (\boldsymbol{x_j})$$

Polynomial:

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = ((\gamma(\boldsymbol{x_i}) \cdot (\boldsymbol{x_j})) + a)^p, \qquad \gamma > 0$$

RBF:

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = e^{(-\gamma \|\boldsymbol{x_i} - \boldsymbol{x_j})\|^2} \qquad \gamma > 0$$

Perceptron:

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \|\boldsymbol{x_i} - \boldsymbol{x_j}\|$$

Sigmoidal:

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = tanh(\gamma \boldsymbol{x_i} \cdot \boldsymbol{x_j} + r) \qquad \gamma > 0$$

Where γ, r and p are Kernel parameters.

3.2.3. Types of SVM Classification

In the literature different types of SVMs are depicted which differ in their characteristics. Essentially, the proposed methods are expansions of the basic algorithm C-SVM. Those expansions aim at rendering the SVM useful for specific cases regarding the input data or the desired output. There are also different types of SVM Regression as ϵ-SVR and ν-SVR which are not contemplated within this work. Following some important expansions for classification propose are introduced [20].

3.2.3.1. $Weight$-Classification

$Weight$-Classification is used when the class distribution in the trainings examples is unbalanced. To reduce the contribution of dominating classes in the training process one has to weight the parameter C by adding a new parameter w_i for each class [23].

$$\min_{\alpha}\left[\frac{1}{2}\boldsymbol{\alpha^T H \alpha} - \sum_{i=1}^{N_S}\alpha_i\right] \qquad \text{s.t} \qquad 0 \leq \alpha_i \leq Cw_i\ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N_S}\alpha_i y_i = 0 \tag{3.39}$$

Where $\boldsymbol{H}$ is now:

$$\boldsymbol{H_{ij}} = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) = y_i y_j K(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \tag{3.40}$$

3.2.3.2. ν-Classification

This type of SVM is used when the number of support vectors and training errors is to be controlled. A new parameter $\nu \in (0, 1]$ is introduced in the formulation to define the upper bound on the fraction of training errors and a lower bound of the fraction of support vectors [20].

$$\min_{\alpha}\left[\frac{1}{2}\boldsymbol{\alpha^T H \alpha}\right] \qquad \text{s.t} \qquad 0 \leq \alpha_i \leq 1/l\ \forall i \qquad \text{and} \qquad \sum_{i=1}^{N_S}\alpha_i \geq \nu,\ \sum_{i=1}^{N_S} y_i\alpha_i = 0 \tag{3.41}$$

Where $\boldsymbol{H}$ is now:

$$\boldsymbol{H_{ij}} = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) = y_i y_j K(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \tag{3.42}$$

3.3. Multi-class Support Vector Machines formulation

The classification in more than two classes is based on a combination of binary classification. Now the possible labels of each training data will be $y_1, y_2 ... y_k$ with $k > 2$.

There are three typical strategies to solve multi-class SVM based on the construction on L binary classification problems [33]:

- One versus One (OvO): this technique divided the hole formulation in $k \cdot (k-1)/2$ binary SVMs, one for each pair of classes. Each testing example is classified by all the binary SVMs. The decision function assigns the class which has the largest number of votes.
- One versus All (OvA): this technique divides the hole formulation into k binary SVMs. The ith SVM is trained with all of the examples in ith class labeled as positive and the rest labeled as negative. The class which is finally assigned is the class which has the largest value of the decision function.

- Directed Acyclic Graph SVM (DAGSVM): this technique also use a set of binary SVM to solve the main problem. In DAGSVM the decision function is constructed as a binary tree. In each state of the tree the data is compared with two classes. The class that is less similar to the input example is discarded (see figure 3.5).

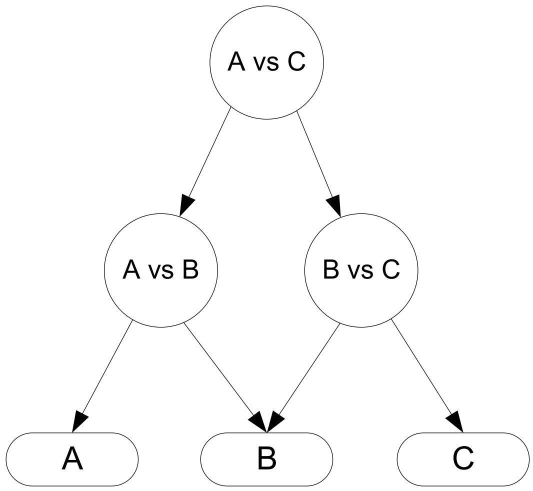


Figure 3.5.: DAGSVM diagram. Illustration based on [38]

4. Implementation

4.1. Overview

The processing which is proposed within this thesis finally aims at the detection of ischaemic episodes. The required processing steps to fulfill this task are shown in figure 4.1.

Preprocessed and segmented beats which are obtained from the European ST-T-database (EDB) serve as input to further processing methods. These beats are labeled using a trained SVM. Afterwards, the labeled beats are processed by a function to locate the episodes.

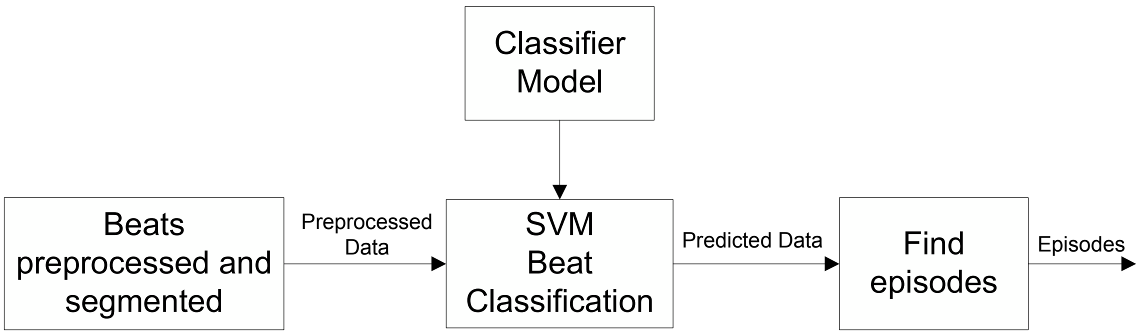


Figure 4.1.: Implementation of ischaemia episode detection

In the following sections the three main aspects of the presented thesis

- Input data: preprocessing and segmentation (see section 4.2)
- Beat classification: SVM model[1] generation (training) and SVM application (see section 4.3)
- Episode detection: method and parameters of the algorithm (see section 4.4)

are explained in detail.

4.2. Input data and preprocessing

4.2.1. The European ST-T database

Within this work the records of the EDB are used as data material. These records are referred to as *Raw Data*.

[1]With model from here all the information which is related to the SVM is meant; amongst others this includes the support vectors; for details see section 4.3.7

The EDB was created for the evaluation of algorithms dedicated to the analysis of ST and T-wave changes. It consists of 90 annotated excerpts of ambulatory ECG recording from 79 subjects. The database includes 367 episodes of ST segment changes, and 401 episodes of T-wave changes, with durations ranging from 30 seconds to several minutes, and peak displacements ranging from $100\,\mu$V to $1\,$mV. Each record is two hours in duration and contains two ECG channels, each sampled at 250 samples per second with 12-bit resolution [69].

Two cardiologists worked independently to annotate each record beat-by-beat (QRS classes). Further on, changes in ST segment, T-wave morphology and signal quality were annotated. ST segment and T-wave changes were identified in both leads, and their onsets, extrema, and ends were annotated. Annotations made by the two cardiologists were compared, disagreements were resolved by the coordinating group in Pisa, and the reference annotation files were prepared; altogether, these files contain 802,866 annotations [69].

4.2.2. Beat Preprocessing

4.2.2.1. Preprocessing Overview

The preprocessing summarizes all steps which lead to the Karhunen-Loève-Transformation-coefficients (KLT-coefficients) which serve as basis in this work. The used preprocessing scheme originates from the ideas described in Nauber [50]. Essentially, it consists of four steps.

1. Signal filtering
2. Beat segmentation
3. Temporal alignment
4. Application of the KLT

In the following sections the most important details on each step are outlined. For more details readers are referred to [50, 76].

4.2.2.2. Signal filtering

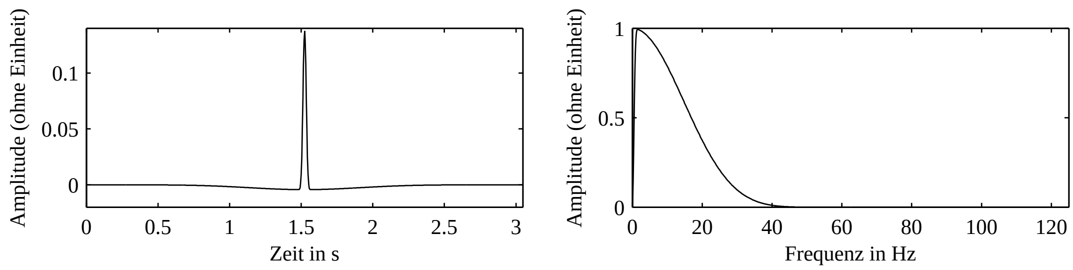
To reduce the influence of typical distortions on the ST-T-complex a linear filter is applied to the ECG. The applied FIR filter is implemented through the Wavelet Transform and its inverse. Thereby the quadratic spline wavelet [44] is used. The upper and lower -3 dB frequencies of the band pass filter are 0,48 Hz and 15,4 Hz, respectively. Figure 4.2 contains the filter characteristics.

4.2.2.3. Beat segmentation

The beat segmentation aims at cutting out a signal segment which contains the complete VR. In this regard, a scheme based on [41] is chosen. Onset $onset_{VR}$ and length $l_{VR}(RR)$ of the signal segments are given by

$$onset_{VR} = FP_{QRS} + 72\,\text{ms} \tag{4.1}$$

$$l_{VR}(RR) = \begin{cases} RR - 240\,\text{ms} - 72\,\text{ms} & RR \geq 720\,\text{ms} \\ \frac{2}{3}RR - 72\,\text{ms} & RR < 720\,\text{ms}. \end{cases} \tag{4.2}$$



(a) Impulse response

(b) Frequency response

Figure 4.2.: Filters

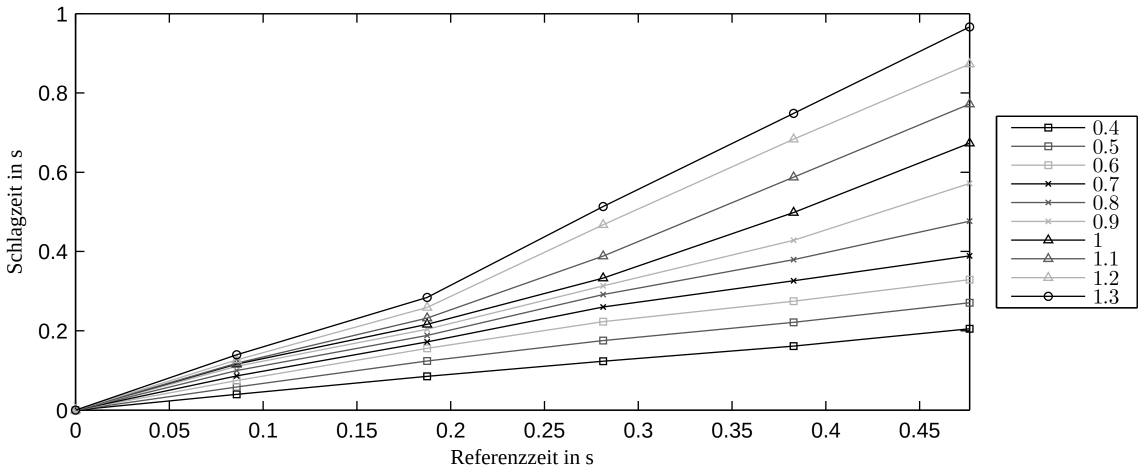


Figure 4.3.: Warping paths for different RR-categories [75]

Thereby, FP_{QRS} constitutes the fiducial point of the QRS complex whose VR is under investigation. RR is the distance to the next QRS-complex. Within this work all *regular beats* of the EDB are used. The regular beats comprise all beats except those which are annotated as non-normal, neighboring beats of the non-normal beats as well as beats which are marked as noisy.

4.2.2.4. Temporal alignment

The temporal alignment aims at reducing the influence of the varying lengths of the segmented VRs. In [50] a proceeding is described which does this alignment based on data based warping paths. The warping paths do a piecewise linear temporal scaling, i.e. a stretching and compression respectively is done. The rule which is applied to a segmented VR depends on the VR's foregoing RR interval. Figure 4.3 contains the warping paths which resulted in [50]. The application of the warping paths ensures that the resulting signal segments are of equal length. This is the precondition to apply the KLT as next step.

4.2.2.5. Application of the KLT

KLT basics

The KLT is a signal dependent linear transform. Applying the KLT ensures the minimization of the resulting square error between an original signal $\boldsymbol{x}$, also called pattern vector, of length N_{sig} and its reconstruction $\boldsymbol{x}_{rec}$ which is calculated from a feature vector consisting of n KLT coefficients $b_1, b_2 \dots b_n$ with $n < N_{sig}$.

The transformation rests upon the basis functions Φ. The basis functions are the eigenvectors of the covariance matrix $\boldsymbol{C}$ established by all training patterns

$$\boldsymbol{C} = E\left\{(\boldsymbol{P} - \boldsymbol{M})^{\mathrm{T}} (\boldsymbol{P} - \boldsymbol{M})\right\} \tag{4.3}$$

Thereby, $\boldsymbol{P}$ is a matrix containing all pattern vectors and $\boldsymbol{M}$ is a same-sized matrix as $\boldsymbol{P}$ containing copies of the mean $\boldsymbol{m}$ of the samples of all pattern vectors. The eigenvector with the i-largest eigenvalues λ_i constitutes the i^{th} basis function.

KLT application

To calculate the KLT-coefficients an inner product between the signal segment under consideration and the first six basis functions is calculated. Within this works the basis functions of [50] are used. These basis functions are constructed using the temporal aligned VRs. As result of applying the KLT one gets 6 coefficients by which the VR of each beat is described. The further processing in this work relies on these coefficients.

4.2.3. Generation of SVM Input

To apply the SVM a vector of the features of each beat and a label that determines the belonging class is needed. Within this work this data is referred to as *Preprocessed Data*. The steps to obtain proper preprocessed data are shown in figure 4.4. A detailed description is given in the next sections.

4.2.3.1. Feature vectors

The feature vectors of the preprocessed data are composed of the first 6 coefficients of the KLT of each beat. These coefficients are obtained by the method which was previously explained. However, in the ST-T database the criteria which is applied to identify and annotate ST-episodes and T-episodes involves a reference. This reference was originally selected from the first 30 seconds of each record and channel. ST-segment deviations were always measured relative to this reference. To identify a T-wave episode a similar criteria was applied: T-deviations were measured relative to the same reference waveform which was used for measuring ST deviations. [69]

Within this work, to use an equivalent criteria to identify ST-episodes and T-episodes, a reference of each record and channel is extracted. This reference is obtained by calculating the median of the first 200 beats' coefficients. As a result one gets a pattern consisting of 6 KLT coefficients for each channel and record. These pattern serve as reference. Finally this reference is extracted from the original KLT coefficients.

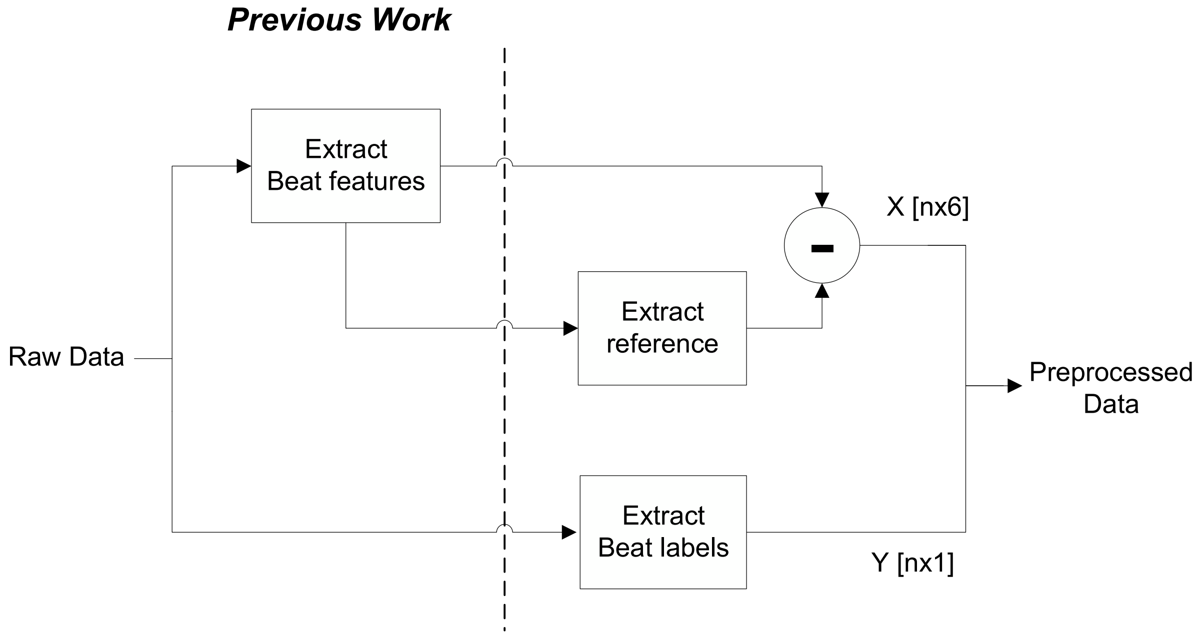


Figure 4.4.: Preprocessing Raw Data

4.2.3.2. Beat labels

By using the annotations of ST-episodes and T-episodes each beat is labeled with a specific number[2]. The labeling is done based on the annotations for the single channels. The labels assigned to the different annotations are:

Labels

- **0** : Non ischaemic beat
- **1** : ST segment elevation (s+)
- **2** : ST segment depression (s-)
- **4** : T wave elevation (T+)
- **8** : T wave depression (T-)

It has to be considered that each beat can have more than one annotation if ST- and T-episodes occur at the same time. Considering this possibility and adding all the values assigned to each beat, there are 9 label types:

Combined Labels

- **0** : Non ischaemic beat
- **1** : ST segment elevation (s+)
- **2** : ST segment depression (s-)

[2]Note that the EDB does distinguish between ischaemia and normal on a beat level, but just defines episodes

- **4** : T wave elevation (T+)
- **5** : T elevation (T+) and ST elevation (s+)
- **6** : T elevation (T+) and ST depression (s-)
- **8** : T wave depression (T-)
- **9** : T depression (T-) and ST elevation (s+)
- **10** : T depression (T-) and ST depression (s-)

Considering the annotation scheme, some combinations are not feasible. As an example, label 3 can never be assigned to a beat because that means that a ST-segment elevation and depression is annotated at the same time.

The number of labels that is obtained for each record is contained in table 4.1:

Table 4.1.: Number of Labels in each record

Record	Label 0	Label 1	Label 2	Label 4	Label 5	Label 6	Label 8	Label 9	Label 10
e0103	13276	656	0	0	0	0	0	0	0
e0104	10633	288	138	378	71	0	2585	0	87
e0105	9495	270	793	448	1708	0	0	0	0
e0106	11777	47	875	211	0	34	635	0	279
e0107	8298	29	417	416	353	0	805	0	116
e0108	10234	0	130	1423	303	0	35	0	267
e0110	12507	1	0	39	57	0	0	0	0
e0111	13555	9	3	98	99	0	158	0	152
e0112	6948	167	90	0	148	0	938	55	0
e0113	13637	96	0	235	543	0	339	0	0
e0114	8786	98	179	84	71	0	57	1	144
e0115	18077	0	603	0	0	0	118	0	0
e0116	7352	1	178	41	209	0	173	0	0
e0118	9345	166	191	355	841	0	110	0	0
e0119	5994	248	324	413	803	0	1159	33	38
e0121	14955	0	168	1471	0	0	769	543	356
e0122	18027	176	179	1598	0	0	2232	40	0
e0123	16527	0	579	23	0	841	0	0	0
e0124	14488	0	1041	305	0	576	136	0	0
e0125	15566	0	59	1270	0	369	0	0	0
e0126	15132	0	298	183	0	0	187	0	0
e0127	16380	136	224	250	298	0	0	0	0
e0129	8267	10	233	676	552	0	0	0	200
e0133	9202	0	0	558	0	0	200	0	0
e0136	11147	25	0	1298	158	28	0	0	0
e0139	16364	334	764	0	100	0	0	0	0
e0147	11235	0	430	0	0	273	0	0	0
e0148	16	0	0	0	0	0	0	0	0
e0151	11589	34	0	1625	258	0	0	0	0
e0154	12127	0	98	0	0	0	131	0	0
e0155	7467	0	0	13	0	0	0	0	0
e0159	8262	0	390	0	0	0	0	0	0
e0161	12511	0	1193	2046	0	0	0	0	0
e0162	10356	0	0	419	0	0	947	0	7066
e0163	13842	0	17	0	0	0	47	0	0
e0166	9668	45	335	678	784	0	614	0	72
e0170	11179	0	87	7	67	0	0	0	0
e0202	13546	0	2043	0	0	974	86	0	879

Continued on next page

continued from previous page

Record	Label 0	Label 1	Label 2	Label 4	Label 5	Label 6	Label 8	Label 9	Label 10
e0203	13984	2584	2450	0	0	0	0	0	0
e0204	19901	0	1388	0	0	0	79	0	76
e0205	15643	984	323	0	0	0	1214	0	1156
e0206	12823	3080	1148	0	0	0	1526	0	1091
e0207	11224	0	2162	0	0	0	246	0	0
e0208	14886	0	790	0	0	0	0	0	0
e0210	13043	0	3347	0	0	0	0	0	0
e0211	23322	0	2332	1915	0	275	0	0	0
e0212	19225	0	899	0	0	0	0	0	0
e0213	14393	0	1527	0	0	0	0	0	0
e0302	18999	0	591	0	0	0	0	0	0
e0303	15709	0	1213	108	0	0	0	0	0
e0304	15259	0	125	0	0	0	550	0	0
e0305	9999	0	1724	0	0	0	2424	0	3015
e0306	8380	0	3318	0	0	0	1304	0	0
e0403	17251	254	0	0	0	0	386	74	249
e0404	11878	0	735	622	77	0	0	0	0
e0405	11899	0	2165	1543	0	0	3653	0	1428
e0406	9612	0	4748	290	0	0	0	0	0
e0408	15726	0	1666	0	0	0	0	0	0
e0409	25572	0	0	0	0	0	0	0	0
e0410	13916	0	0	125	203	0	0	0	0
e0411	13965	96	200	84	98	0	212	0	269
e0413	10753	0	150	272	0	407	0	0	0
e0415	8435	0	4020	0	0	0	640	0	2571
e0417	16882	0	1380	0	0	0	0	0	0
e0418	21090	0	1922	0	0	0	0	0	0
e0501	11518	0	2276	0	0	0	0	0	0
e0509	14704	0	0	0	0	0	0	0	0
e0515	11106	421	18	0	0	0	4802	0	427
e0601	14299	55	0	0	25	0	351	0	0
e0602	14904	0	3736	0	0	0	0	0	0
e0603	11465	9	1491	2056	455	0	0	0	0
e0604	10850	22	1309	315	789	0	569	0	14
e0605	17278	0	1086	0	0	0	0	0	0
e0606	12821	0	156	0	0	0	1639	0	3434
e0607	7922	0	9350	0	0	0	0	0	0
e0609	14701	0	2825	0	0	0	0	0	0
e0610	12864	0	1725	680	0	41	0	0	0
e0611	9790	0	0	0	0	0	0	0	0
e0612	8173	0	1185	0	0	0	45	45	0
e0613	4157	32	427	2955	156	137	0	0	0
e0614	4444	0	8352	0	0	0	0	0	0
e0615	11897	0	0	112	0	5	0	0	0
e0704	13608	0	346	1490	0	0	278	0	2146
e0801	12572	0	391	0	0	0	66	0	25
e0808	7072	227	0	0	0	0	126	0	293
e0817	5378	0	120	428	0	0	0	0	0
e0818	12730	0	2073	0	0	0	0	0	23
e1301	15237	0	1123	0	0	0	610	0	0
e1302	13622	0	526	692	115	0	189	0	46
e1304	12078	0	823	0	0	0	24	0	1733
Total	1126726	10600	91720	30248	9341	3960	33394	791	27652
%	84.43 %	0.7943 %	6.8733 %	2.2667 %	0.7000 %	0.2968 %	2.5025 %	0.0593 %	2.0722 %

4.3. Beat classification

4.3.1. Used software

Software selection: The SVM implementation in this work is done by building up on the LibSVM software [20]. This software has been chosen because it can be applied to large datasets (implementation in C++), supports automatic model selection, weighted SVM for unbalanced data and multi-class classification. Furthermore, LibSVM is an integrated software package that supports vector classification, regression and distribution estimation. Thus, it offers a wider range than just the classification which may become important for future works.

Software details: To avoid solving a time-consuming numerical QP optimization problem, LibSVM uses a modified SMO algorithm to perform SVM training. This algorithm breaks the QP problem into a series of smallest possible QP problems that are solved analytically. In table 4.2 the main characteristics of LibSVM are detailed. Through it wide usage in the scientific community LibSVM can be seen in table 2.1.

Software usage: Within this work, MatLab is used as interface to invoke the LibSVM software package. All the C++, Java and Python sources included in LibSVM use data stored in a special format. This format is known as LibSVM format. To transform this data files into MatLab format one can use the function `libsvmread.m`. To write data in a LibSVM the function `libsvmwrite.m` can be used. When needed the functions provided by LibSVM are supplemented by own functions which are programmed in MatLab.

4.3.2. Classification proceeding

The method to extract the classifier model and the ischaemic episodes can be divided in different stages which are characterized by different variants of their implementation. An overview is shown in figure 4.5.

Details on each of the steps are outlined in the following sections.

4.3.3. Data separation

In order to obtain a realistic estimate of the real world performance of trained classifiers the used data is separated into two sets. One set is used to train the SVM and the other to test it. In particular when the SVM is used, limitations arise from the computational complexity which is expected to increase with a growing number of training patterns (see details in 6.1). However, a too small number of training patterns could result in an incomplete training set and thus reduced classification results.

Within this work, the separation is quasi randomly done by choosing a number of records N_{train} from the 90 records which are available in the EDB. The only requirement to be fulfilled was that the record which are used for training purposes must contain a minimum number of beats from each class [3]. In this work $N_{train} = 10$ records have been selected. These records are referred to as *Training Data*, whereas the rest of the records are referred to as *Testing Data* (see figure 4.5).

[3] Therefore the selection is only quasi randomly

Table 4.2.: Characteristics of LibSVM

Characteristics	**LibSVM**
Sources	C++ Java
Interfaces	MatLab Python Others
Kernels	Linear RBF Polynomial Sigmoid Your own kernel by modifying `svm.cpp`
Different SVM formulations	C-SVC ν-SVC One-class SVM ϵ-SVR ν-SVR
One-class SVMs/Regression	ϵ-SVR ν-SVC Probability estimates for SVR One-class SVM using a hyperplane
Binary classification SVMs	C-SVM ν-SVC Probability estimates for C-SVC Probability estimates for ν-SVC
Multiclass algorithms	OvO OvA
Weighted SVM	Different weight of C for each class for unbalanced data
Cross validation	With multiple criteria for evaluation
Grid search	Via python script
Subset of data with same distribution	Via python script
Feature scaling	Via external program *svm-scale*
Other useful tools	Read and write files in LibSVM format with MatLab functions

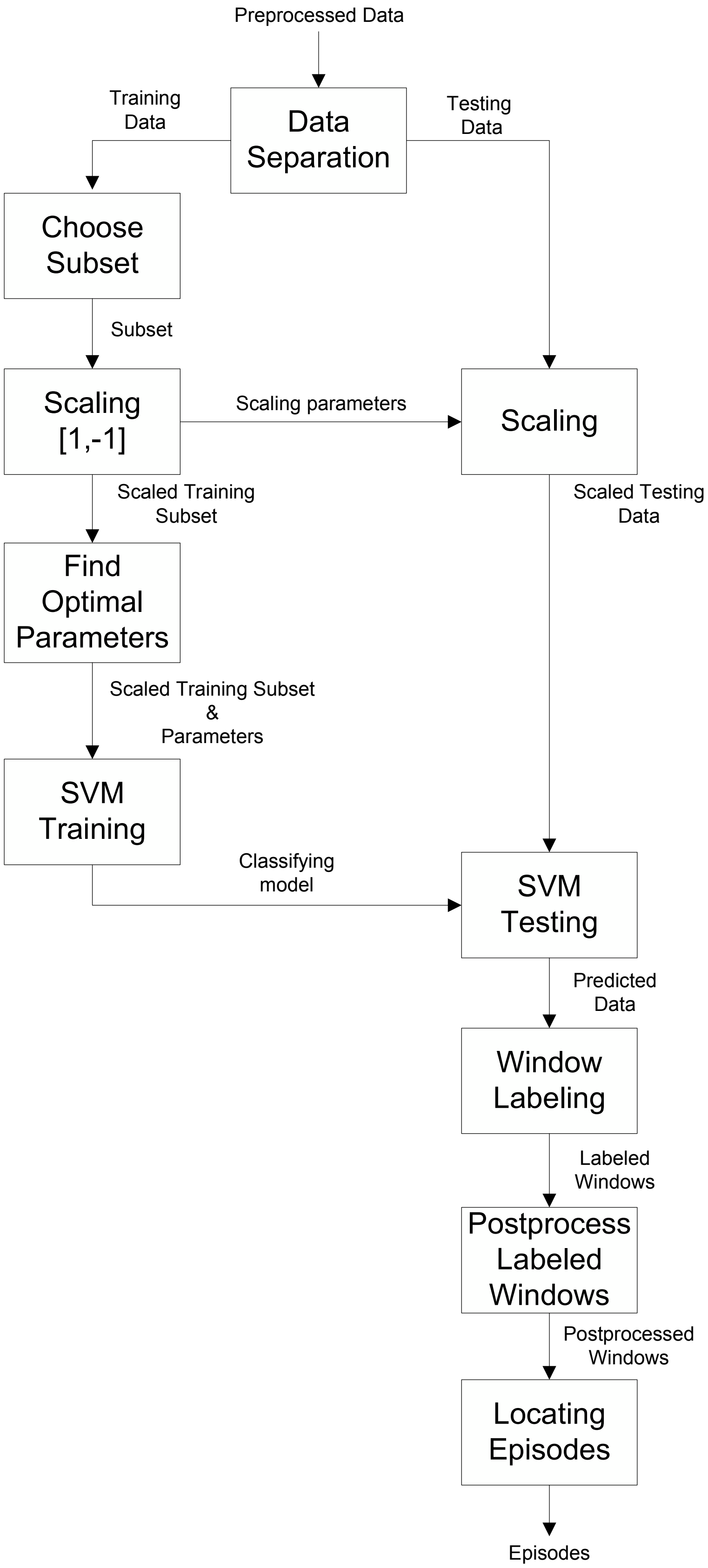


Figure 4.5.: Training and testing of an algorithm to detect ischaemic episodes

The 10 records which build up the training data and their characteristics are described in detail in table 4.3.

Table 4.3.: Number of labels in the training data

Record	Label 0	Label 1	Label 2	Label 4	Label 5	Label 6	Label 8	Label 9	Label 10
e0113	13637	96	0	235	543	0	339	0	0
e0121	14955	0	168	1471	0	0	769	543	356
e0166	9668	45	335	678	784	0	614	0	72
e0202	13546	0	2043	0	0	974	86	0	879
e0206	12823	3080	1148	0	0	0	1526	0	1091
e0210	13043	0	3347	0	0	0	0	0	0
e0211	23322	0	2332	1915	0	275	0	0	0
e0302	18999	0	591	0	0	0	0	0	0
e0417	16882	0	1380	0	0	0	0	0	0
e0605	17278	0	1086	0	0	0	0	0	0
Total	154153	3221	12430	4299	1327	1249	3334	543	2398
%	84.2578 %	1.7605 %	6.7940 %	2.3498 %	0.7253 %	0.6826 %	1.8223 %	0.2968 %	1.3107 %

In total, the training consists of 182954 beats (16.2377% of the preprocessed data). To reduce the training time consumption just a subset of the whole training data is selected. The selected data is referred to as *Training Subset* (see figure 4.5) and has 8500 beats.

In the experiments on different training strategies, two distributions for the training subset (unbalanced and balanced) are contemplated. With these two different distributions one expects different SVM models and consequently different beat classification results.

4.3.3.1. Unbalanced

The first distribution choice is to keep the same data distribution as it is found in the original training data. Having the same data distribution is supposed to have a positive contribution to the training process [70]. LibSVM offers a python script (`subset.py`) that extracts a subset with the same class distribution as in the original training data.

The number of beats belonging to each class and the percentage of each class from the total number of beats in the unbalanced subset are detailed in table 4.4:

Table 4.4.: Number of Labels in Unbalanced Subset

Record	**Label 0**	**Label 1**	**Label 2**	**Label 4**	**Label 5**	**Label 6**	**Label 8**	**Label 9**	**Label 10**
Number	7161	150	578	199	62	58	155	25	112
%	84.24 %	1.77 %	6.80 %	2.34 %	0.73 %	0.68 %	1.82 %	0.29 %	1.43

4.3.3.2. Balanced

A balanced distribution, i.e. a training subset with same number of beats of each class is also chosen. This distribution has the advantage of including more examples of ischaemic beats than in the unbalanced distribution. In this case an own MatLab script (`InputRandomData.m`) is used to select these beats. This script extracts randomly from the training subset, 1000 beats of each class (except in class 9 were only 500 beats are selected; this exception is debited to the distribution which is found in the preprocessed data where only 791 beats were labeled as class 9) (see table: 4.1). Furthermore, there is just one record (e0121), that has more than 500 beats labeled as class 9. For this reason, this record must be included in the training data (therewith it is the only record that is previously known to be part of the training subset).

4.3.4. Scaling

Scaling the feature vectors before applying the SVM is needed to get a faster and a better performance of the SVM training and testing [32, 61]. Scaling each feature of training subset to a specific range [-1,1] avoids that greater attributes dominate smaller ones. Furthermore, scaling features results in simpler calculations.

In order to scale the data it is important to use the same scaling method in training and testing processes. The scaling factors must be derived from the training data in order to avoid *biasing* the classifier and thus modifying the results which are expected for real world data.

To scale data in a proper way, LibSVM includes a script (`svm-scale.c`) that scales each feature into the range [-1,1].

The data that one gets after scaling the training subset is referred to as *Scaled Training Subset*. Analogously, the scaled testing data is referred to as *Scaled Testing Data* as it is shown in figure 4.5.

4.3.5. Classification - binary and multi-class approach

The experiments which are carried out within this work are divided in binary classification and multi-class classification. Whereas the binary classification just distinguishes between normal and ischaemic beats, the multi-class classification distinguishes all the classes which have been outlined in section 4.2.3.2.

By applying a binary classification the computational time is lower than in multi-class classification. However, by applying multi-class classification one could expect a better results of the classification. By applying both strategies one allows to draw conclusions on the effect of the multi-class classification compared to its higher computational costs in the end.

4.3.5.1. Binary Classification

To apply the binary classification at first the label vector of the scaled training subset and the scaled testing data has to be adopted. This modification is made by changing all the ischaemic labels to 1 and the normal labels to -1. The usage of -1 allows the application of an automatic parameter finding tool. A detailed description of the tool is contained in the next section 4.3.6.

4.3.5.2. Multi-class classification

To apply the multi-class classification the OvO strategy is chosen. This strategy is chosen regarding better classification results [33] than OvA and DAGSVM. Considering that there are 9 classes, applying the OvO strategy one will get $\frac{k(k-1)}{2} = 36$ binary classifiers.

4.3.6. Kernels and optimal parameters

From the most common Kernels which have been mentioned in section 3.2.2, Linear and RBF Kernels are chosen to be used within this work. For each of these Kernels one must find the optimal Kernel parameters as well as the penalty parameter C.

4.3.6.1. Linear

The linear Kernel has been chosen because it is computationally easiest to apply. Furthermore, applying this Kernel one just needs to find the optimal parameter C. Moreover this, the linear Kernel turned out to be even more powerful compared to more complex Kernels in other investigations [23].

4.3.6.2. RBF

The RBF Kernel is used in case that the relation between class labels and features coefficients is not linear. This Kernel has been chosen because it has fewer numerical difficulties than sigmoid Kernel or polynomial Kernel [32]. Furthermore, RBF Kernel has less parameters to optimize than polynomial Kernel. To characterize RBF Kernels one has only to find C and γ parameters.

4.3.6.3. Parameter selection

LibSVM contains the script `grid.py` to find the best C and γ parameters for a C-SVM classification using the RBF kernel. This script uses the program `gnuplot` [73] to visualize the results in a graphic way. `grid.py` uses the global accuracy acc_{global} to evaluate the results. acc_{global} is defined by

$$acc_{global} = \frac{\sum_{i=1}^{k} TP_i}{N_{beats}} \tag{4.4}$$

where TP_i is the number of true positive labels for class i, k is the number of classes and N_{beats} is the total number of beats.

However, the global accuracy is not a suited criterion for binary classification of unbalanced data [70]. One achieves high values of accuracy by predicting all beats as belonging to the class. As an example, the preprocessed data has 84.43% of normal beats and 15.57% ischaemic beats. By classifying all beats as normal one gets a value of 84.43% accuracy. For this reason, an own sricpt `Grid.m` is used to find the best C and γ parameters with a more suitable evaluating criteria.

Therefore, in this work the mean accuracy (MAC) *mac* is used [23].

$$mac = \frac{\sum_{i=1}^{k} acc_i}{k} \tag{4.5}$$

where acc_i is the accuracy for class i and k is the number of classes. For acc_i holds

$$acc_i = \frac{TP_i}{TP_i + FN_i} \tag{4.6}$$

where TP_i and FN_i are the true positives and false negatives, respectively.

In binary classification, the definition of MAC results in the balanced accuracy (BAC) *bac* which is defined as

$$bac = \frac{Se + Sp}{2} \tag{4.7}$$

Where Se is sensitivity and Sp is specificity.

Sensitivity: ratio of correct detected events to the total number of events

$$Se = \frac{TP}{TP + FN} \tag{4.8}$$

Specificity: ratio of correctly rejected nonevents to the total number of nonevents

$$Sp = \frac{TN}{TN + FP} \tag{4.9}$$

The method used by `Grid.m` to estimate the best parameter combination (C, γ) is *cross validation* [20]. To perform cross validation one has to divide the scaled training subset in a certain number of folds of equal size. In each iteration one of the folds is used to test the classifier. The rest of the folds are used to train the SVM with a certain C and γ parameter. In each iteration the fold used to test the found parameters is different, in order to extract the criterion value for the selected parameters. This process is repeated with all determined combinations of C and γ. Finally, the parameter combination (C, γ) that gets the best test results is extracted. The method which underlies the parameter search is explained in figure 4.6.

The range of C and γ must be defined considering that the performance of the SVM depends on them. One has to take into account that small values of C indicate low penalization of misclassified training samples. Thus, the empirical error of the learning machine is considerably large and the hyperplane is simple. However, large values of C lead to a complex hyperplane as no misclassified training samples are allowed [43]. The range of C used is between 10^{-5} and 10^{15} with a step size of 10^2 as it is defined in the default values of `grid.py`.

The range of γ must be also defined. In this case, one must to take into consideration that big values of γ lead to all the samples become support vectors. This causes that testing time increases and over-fitting problem. However, small values of γ cause worse SVM performance [43]. The range of γ used is between 10^{-15} and 10^5 with a step size of 10^2, as it is defined also in the default values of `grid.py`.

In the case of unbalanced data one should use a weighted SVM to avoid possibly occurring disproportionate influence of the majority class on the margin [26, 23]. This requires to weigh the parameter C by defining a new parameter w_i for each class ($w_i C$). This parameter is not automatically found by the grid function. In the literature there are some suggested formulas to

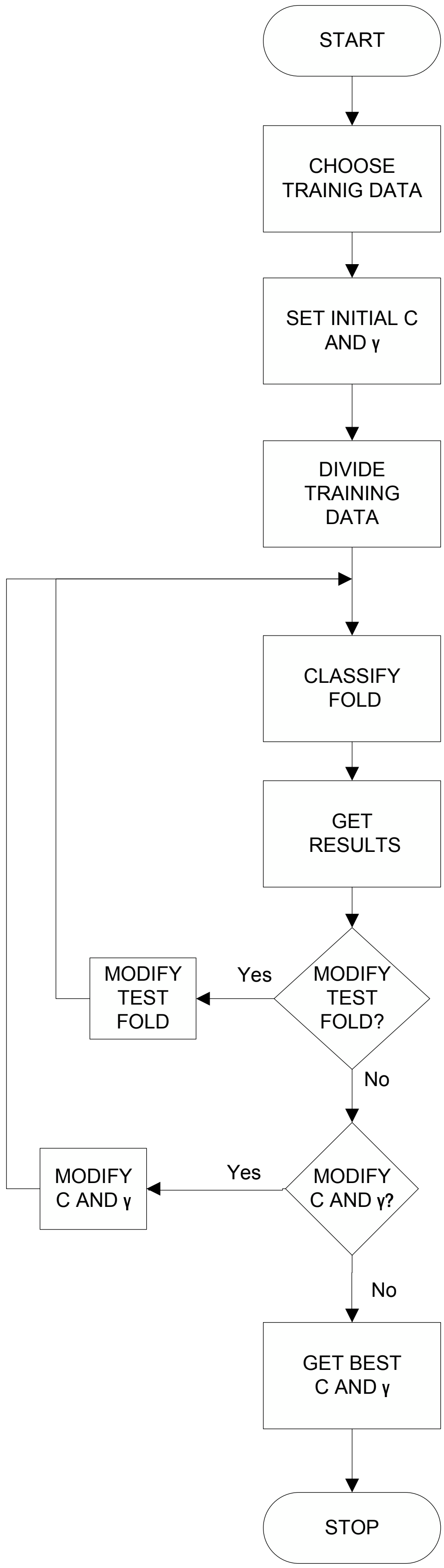


Figure 4.6.: Cross validation algorithm

define these values. Within this work, 3 of them are evaluated for each Kernel and the different distributions of training data.

The first formula suggested in [23] is to set w_i values according to the prior probabilities of each class. This formula is referred to as W^1:

$$w_i^1 = \frac{number\ of\ instances\ of\ i\ class}{total\ number\ of\ instances}$$

The second formula [4] to define w_i is referred to as W^2:

$$w_i^2 = \frac{number\ of\ instances\ of\ the\ biggest\ class}{number\ of\ instances\ of\ class_i}$$

The third formula suggested in [26] to define w_i is referred to as W^3:

$$w_i^3 = \frac{total\ number\ of\ instances - number\ of\ instances\ of\ i\ class}{total\ number\ of\ instances}$$

The w_i values that results of each formula and distribution are detailed in table 5.1:

Finally, the process made to find the optimal parameters for each method which is used to set the w_i (W^1, W^2, W^3) is an improved grid-search algorithm [43]:

1. Define w_i
2. Conduct a coarse grid to find optimal parameters C_1 (and γ_1 in RBF Kernel)
3. Choose the w_i with the best evaluation results
4. Only in RBF: Conduct a fine grid between $\gamma_1 + 10^2$ and $\gamma_1 - 10^2$ with a step size of $10^{0.25}$ and a fixed C_1 to extract γ_{opt}.
5. Conduct a fine grid between $C_1 + 10^2$ and $C_1 - 10^2$ with a step size of $10^{0.25}$ to extract C_{opt} (and a fixed γ_{opt} in RBF Kernel)
6. Select the optimal parameters w_{opt} and C_{opt} (and γ_{opt} in RBF Kernel)

4.3.7. Training

To train the SVM the MatLab interface is used. The training function which is included in LibSVM is called `svmtrain.m`. This function allows to chose the type of SVM formulation which is to be used, the Kernel and define the parameters which are needed for the respective Kernel. Furthermore, the C parameter and a weight for each class w_i can be used. As output one gets a model structure which contains the following information:

Table 4.5.: Weight values

			W^1	W^2	W^3
Binary	**Unbalanced**	Normal	0.8425	1	0.1575
		Ischaemic	0.1575	5.348	0.8425
	Balanced	Normal	0.1176	7.5	0.8824
		Ischaemic	0.8824	1	0.1176
Multiclass	**Unbalanced**	Label 0	0.84	1	0.15
		Label 1	0.02	47	1
		Label 2	0.07	12	1
		Label 4	0.02	36	1
		Label 5	0.01	115	1
		Label 6	0.01	124	1
		Label 8	0.02	46	1
		Label 9	0.003	287	1
		Label 10	0.01	64	1
	Balanced	Label 0	0.12	1	0.88
		Label 1	0.12	1	0.88
		Label 2	0.12	1	0.88
		Label 4	0.12	1	0.88
		Label 5	0.12	1	0.88
		Label 6	0.12	1	0.88
		Label 8	0.12	1	0.88
		Label 9	0.06	2	0.94
		Label 10	0.12	1	0.88

1. $[1 \times 6]$ Vector with the training parameters options selected
2. Number of classes: k
3. Total number of support vectors: N
4. Parameter b of the decision function
5. $[l \times 1]$ Vector with the value of labels
6. $[k \times 1]$ Vector with the number of support vectors of each class
7. $[N \times 1]$ Vector with the coefficients assigned to each support vector $(C_i w_i)$
8. $[N \times 6]$ Matrix with support vectors : $\boldsymbol{S}$

This model is directly used to test new data. Thereto one can easily extract the decision function by calculating:

$$\boldsymbol{w} = \boldsymbol{S}^T \cdot \boldsymbol{C_i w_i}$$
$$y' = \boldsymbol{w}^T \cdot \boldsymbol{x_i} + b$$

4.3.8. Model application

To apply the SVM also the MatLab interface is used. The respective function included in LibSVM package is called `svmtest.m`. The output of this function consists in a vector of predicted labels, accuracy value and a matrix of decision values or probability estimates. This third output is only completed if probability is specified in the options. In this work only the predicted labels vector is used.

To evaluate the performance of the classifier an own MatLab script (`accuracy.m`) is used to build up a confusion matrix. In each row of this matrix the predicted class is represented, while each column represents the truth class of each tested beat. By this matrix one can easily extract some conclusions of the predicted results.

4.3.9. Experiments

So far different training settings regarding methodology and training data have been described. Overall, the combination of investigated selections leads to a set of experiments which are carried out within this work regarding the beat classification. The performed experiments are itemized in figure 4.7. Each experiment is a different combination of subset distribution, binary or multi-class classification and linear of RBF Kernel. The experiments are referred to as:

Experiment 1: Unbalanced Subset, binary classification and Linear Kernel: **UB2CL**

Experiment 2: Balanced Subset, binary classification and Linear Kernel: **B2CL**

Experiment 3: Unbalanced Subset, binary classification and RBF Kernel: **UB2CRBF**

Experiment 4: Balanced Subset, binary classification and RBF Kernel: **B2CRBF**

Experiment 5: Unbalanced Subset, multi classification and Linear Kernel: **UBMCL**

Experiment 6: Balanced Subset, multi classification and Linear Kernel: **BMCL**

Experiment 7: Unbalanced Subset, multi classification and RBF Kernel: **UBMCRBF**

Experiment 8: Balanced Subset, multi classification and RBF Kernel: **BMCRBF**

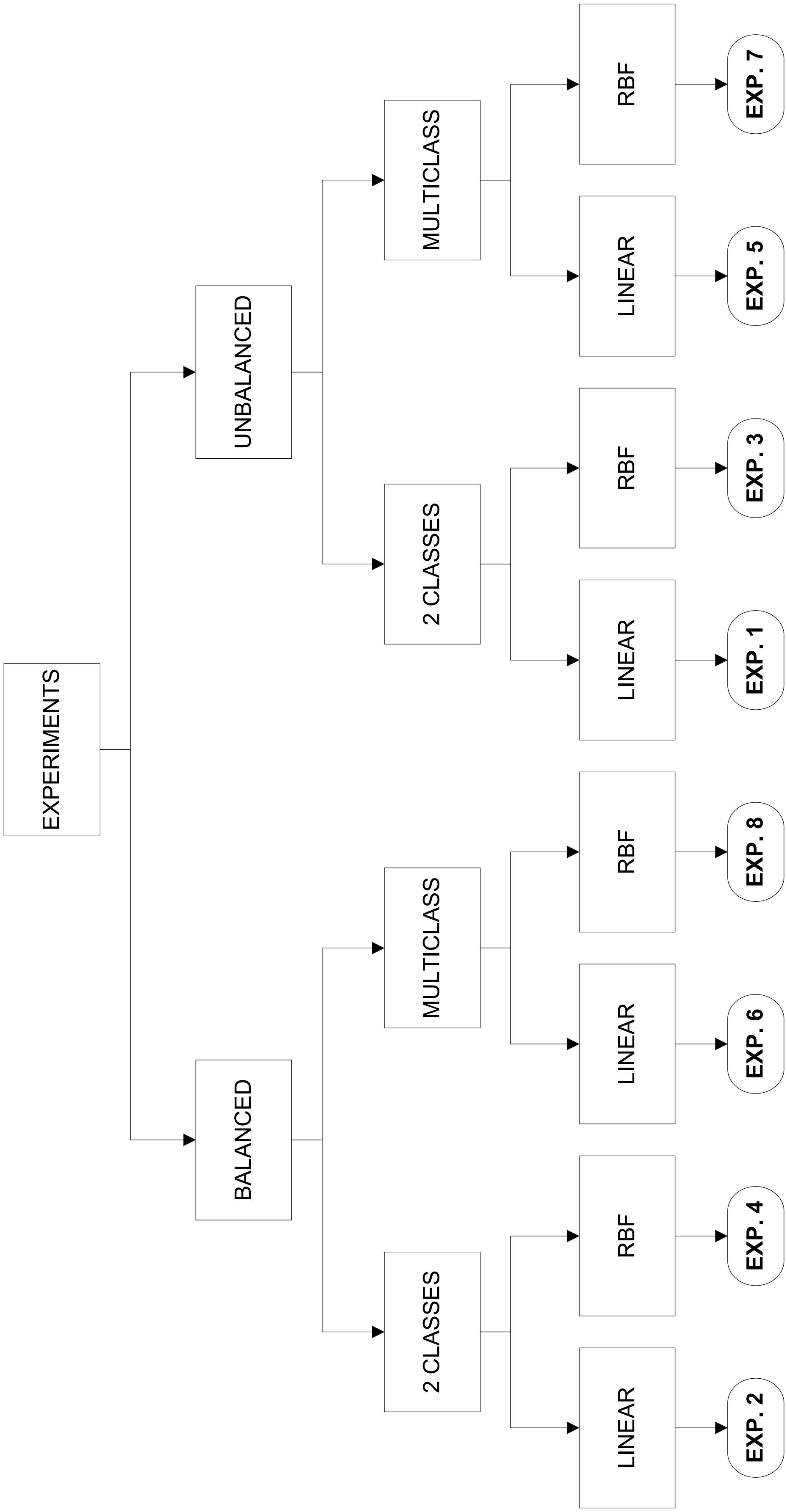


Figure 4.7.: Experiments

4.4. Ischaemic Episode Detection

After classifying the beats of a record using the SVM one gets as output the predicted label vector. The transformation of this vector into episode annotations is done by applying a sliding window technique [48, 54, 55]: Based on the predicted labels windows of a certain time duration are identified as normal or ischaemic. By using a sliding window the whole record becomes classified on the basis of single windows. The classification of each window is based on the percentage of beats which are predicted as ischaemic. Afterwards, a postprocessing of the labeled windows is done in order to improve the quality of the window classification. Based on the output of this postprocessing ischaemic episodes are found and delineated.

4.4.1. Window Labeling

The criteria used to label each time window is based on the number of the ischaemic predicted beats related to the total number of beats in the respective window and channel (the channels are considered seperately). This percentage is compared with two thresholds which divide the possible range in 3 intervals. Depending on the interval to which the percentage of a window belongs, a certain label is assigned to the window under consideration (see figure 4.8). The interval between 100% and $threshold_1$ is labeled as *High: 3*. The interval between $threshold_1$ and $threshold_2$ is labeled as *2: Median*. The last interval is labeled as *1: Low*.

To reduce erroneous labellings, there must be a minimum number of beats in a window to be classified. If this minimum number is not reached because some of the beats have been discarded in the preprocessing process, the whole window is labeled as *-1 :Unlabeled*.

The next window to be labeled is displaced a defined time from the beginning of the previous window. This displacement is done until the end of the record is achieved. The parameters chosen in this process to make the experiments are:

- Size of the window: 30 seconds
- Displacement: 10 seconds
- Minimum number of beats for each window: 5
- $Threshold_1$: variable (see section 4.4.5)
- $Threshold_2$: variable (see section 4.4.5)

After labeling all the windows of a record one obtains a vector of *Labeled Windows* for each channel. Each labeled window is related to his initial time, his end time, the percentage of ischaemic beats and the label which was assigned. The label vector constitutes a symbolic representation regarding the occurrence of ischaemia which must be turned to episodes of ischaemia. Thereto, in order to allow a more robust episode detection, the label vector undergoes a postprocessing before episodes are detected.

4.4.2. Postprocessing Labeled Windows

The postprocessing of labeled windows consists in analyzing and filtering the labeled window vector. First, *Unlabeled* windows are labeled with the same label as the anterior window. Then,

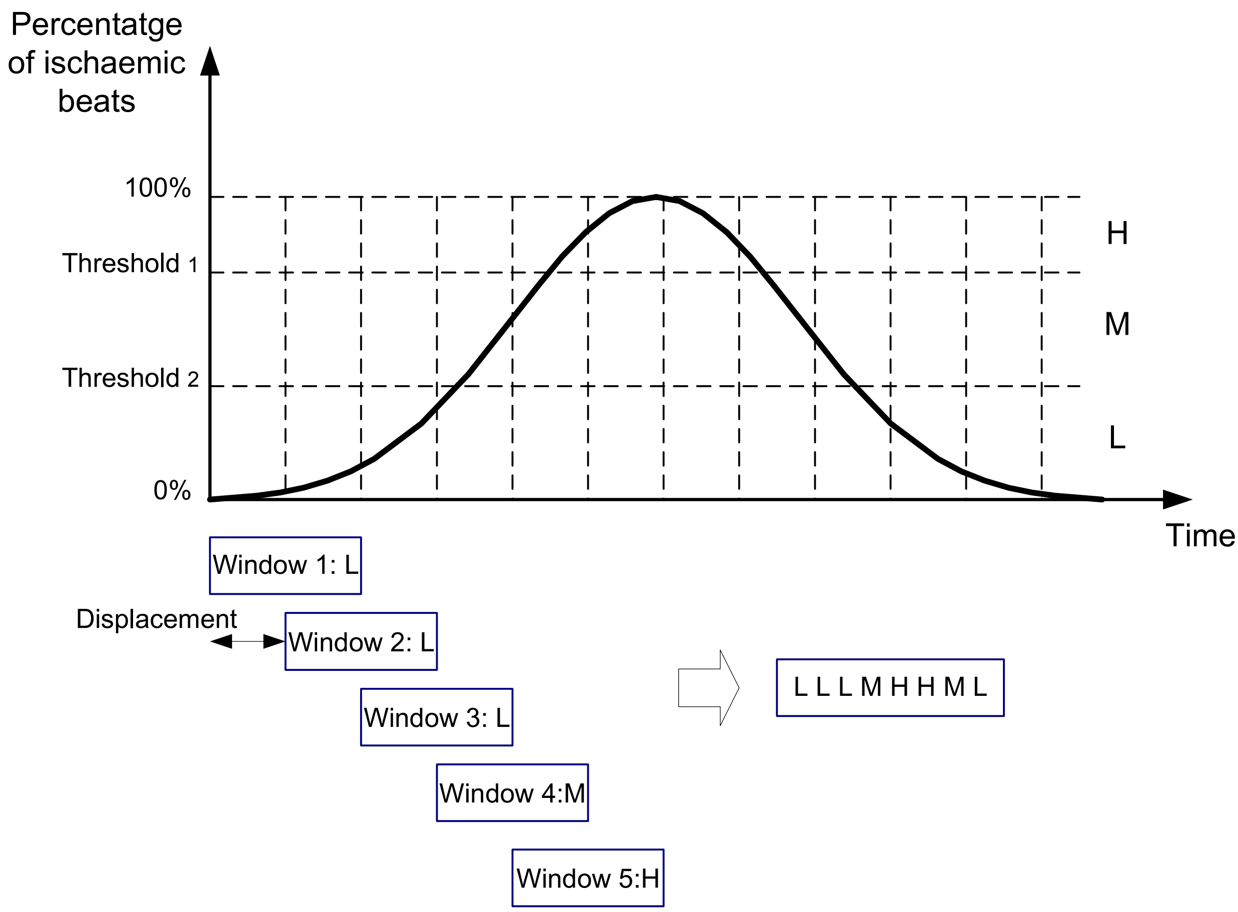


Figure 4.8.: Label windows episodes

each group of three consecutive labels is analyzed to assign a postprocessed label. This post-processed label will be assigned following this criteria:

1. 2 or 3 labels are equal $\longrightarrow$ the postprocessed label is the majority label
2. All the labels are different $\longrightarrow$ the postprocessed label is *2: Median*

4.4.3. Locating Episodes

4.4.3.1. Detection

Compared to the annotation scheme of ST and T-episodes in the EDB, the basis for finding episodes in the present work differs significantly: instead of a numeric value which describes the ST-deviation and T-deviation, respectively, the output of SVM and the following processing steps is a symbolic string. Thus, the own episode detection scheme must operate on such a string. However, apart from the difference of the input a detection method was implemented which is similar to the annotation scheme: following this scheme.

A detection of an episode is assumed when three consecutive *High: 3* postprocessed labels are found. This means that in an interval of 50 seconds the percentage of ischaemic predicted labels is above $threshold_1$.

4.4.3.2. Delineation

Analogously to the detection, even for the delineation similar criterion as explained in section 2.1.1 are used to delineate ischaemic episodes, i.e. find onset and end of the episode. The location of the beginning of an episode is annotated by analyzing the postprocessed labels which occur previous to the detected episode. To find the beginning of the episode at least two consecutive *Low: 1* postprocessed labels are required. The annotation of the episode beginning corresponds with the end of the first *Low: 1* label which was found.

The criterion which is used to annotate the episode end is, analogously, based on the analysis of the postprocessed labels after each detected episode. The episode end is annotated when two consecutive *Low: 1* are found. The time which is annotated as episode end corresponds with the initial time of the first *Low: 1* label which was found.

Furthermore, the maximum of each found episode is annotated. The maximum of an episode is located by comparing the percentages of ischaemic beats between the begin and end of an episode. If more than one consecutive windows exhibit the same percentage, the maximum is placed in the middle of all found the maxima. If more than one non-consecutive windows exhibit the maximum percentage level, the maximum is placed in the middle of the first window of maximum percentage.

An example of locating an episode is shown in figure 4.9.

4.4.4. Creating annotations

The procedures to label windows and locate episodes is carried out separately for each channel. The annotations which are finally written constitute the disjunction of the single channel results for each record. The further characterization of the performance is base on these annotations. The single channel results are not evaluated within this work.

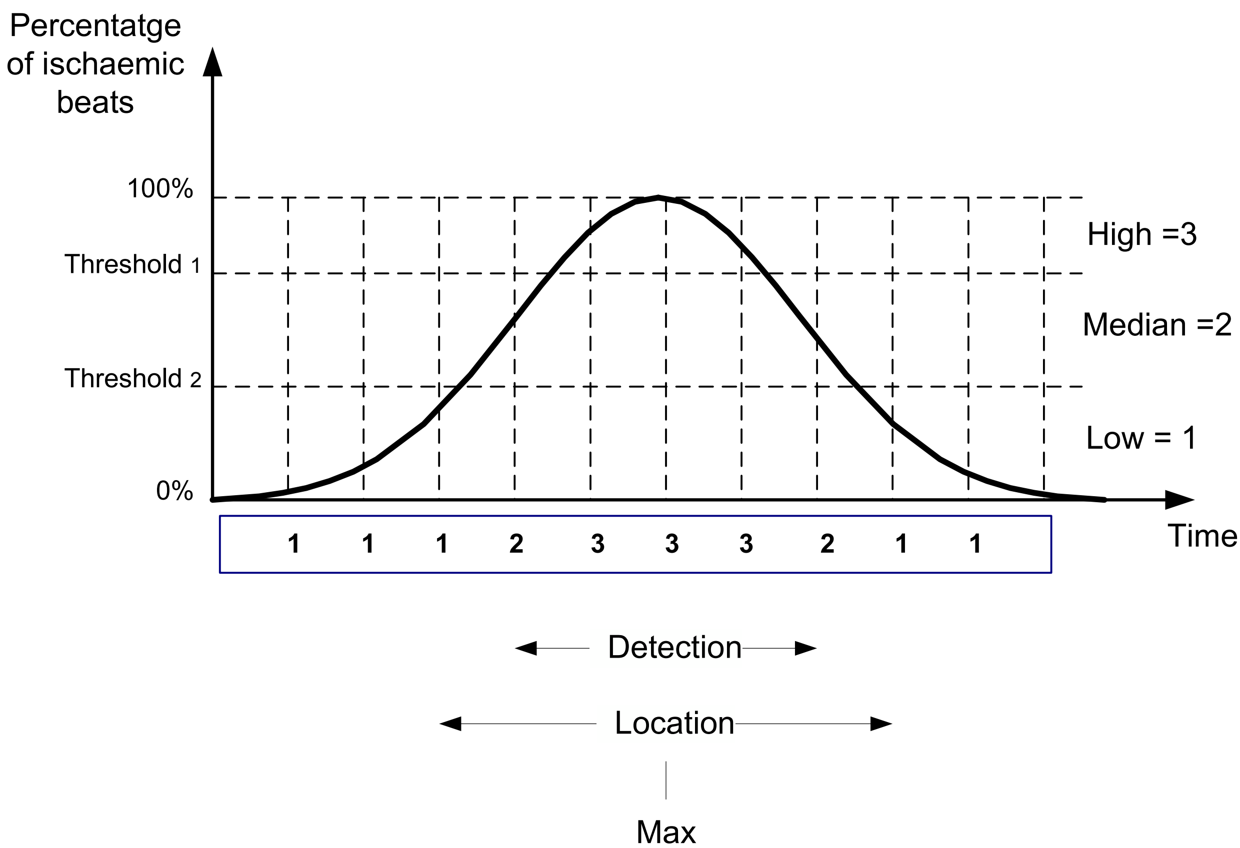


Figure 4.9.: Locating ischaemic episodes

Table 4.6.: Settings of the DPC analysis

Parameter	Type	Wert
Window size	fixed	30 s
Window displacement	fixed	10 s
Minimum number of beats	fixed	5
$threshold_1$	variable	From 80 % to 100 %; step width 5 %
$threshold_2$	variable	From 55 % to 77,5 %; step width 2.5 %

4.4.5. Performance Analysis and DPC Analysis

To characterize the episode detection the presented work uses the parameters which are recommended in this context [64]. Those parameters comprise

- SE Se: episode based sensitivity
- SE +P: episode based positive predictive value
- SD Se: duration based sensitivity
- SD +P: duration based positive predictive value

For all these values the average ([a]) and the gross ([g]) statistic are used. For detailed information on the calculation of these parameters readers are referred to [64, 75].

The results of the episode detection vary with respect to the variables which have been introduced in section 4.4.1. To find optimal settings for each experiment a detector performance characteristic (DPC) analysis is carried out [64]. The DPC, which was introduced in the context of ischaemia detection, constitutes an implementation of a receiver-operator-characteristic (ROC) analysis. Differing from the classical ROC analysis which is based on sensitivity and specificity the DPC uses the sensitivity and positive prediction, respectively.

During the own DPC analysis the thresholds $threshold_1$ and $threshold_2$ are varied according to the values which are outlined in table 4.6. This range was identified as most suitable in extensive pretests. Variations of the window length and window translation, respectively, are not in the scope of this investigation and remain fixed during the DPC analysis.

As outlined before different parameters are used to characterize the detection results. This makes it difficult to find the best results of the DPC. However, to find the most suited result and its belonging parameters the optimal criterion which is proposed in [75] is applied. For *optcrit* holds

$$\begin{aligned} optcrit = [&\mathrm{SE\,Se}_{[g]} \cdot \mathrm{SE\,{+P}}_{[g]} \cdot \mathrm{SD\,Se}_{[g]} \cdot \mathrm{SD\,{+P}}_{[g]} \cdot \\ &\mathrm{SE\,Se}_{[a]} \cdot \mathrm{SE\,{+P}}_{[a]} \cdot \mathrm{SD\,Se}_{[a]} \cdot \mathrm{SD\,{+P}}_{[a]} \end{aligned} \tag{4.10}$$

optcrit thus constitutes a merged measure for the detector's quality.

Furthermore, in [75] the constraint

$$\frac{\mathrm{SD\,{+P}}_{[g]} + \mathrm{SD\,{+P}}_{[a]}}{2} > 60\,\% \tag{4.11}$$

must be fulfilled by a solution to be regarded as valid. Equation 4.11 tackles the problem of globally improving results for decreasing values of SD +P. However, according to the results, in this work this condition is not necessarily to be fulfilled as a meaningful discussion would not be possible if all results which not fulfill the criterion would be disregarded. For this reason in the results it is distinguished between *unconstrained* and *constrained* results. Only in the second case equation 4.11 is considered.

5. Results

5.1. SVM application

5.1.1. Parameter search

In table 5.1 the optimal parameters for each weight formulation are detailed. Thereby, the parameters are varied as outlined in section 4.3.6.3. The evaluation is done based on the percentage which results from the cross-validation tests.

The highlighted parameters are the ones which belong to the settings which are finally selected for each experiment.

5.1.2. Beat Classification

5.1.2.1. Binary Classification

In table 5.2 the MACs which result for each experiments that relies on the binary classification are outlined. Classification results from training as well as testing are given.

5.1.2.2. Multiclass Classification

In table 5.3 the MACs which result for each experiments that relies on the multi-class classification are outlined. Classification results from training as well as testing are given.

The given results summarize the contents of the confusion matrices which are built up each experiment. Table 5.4 shows the confusion matrix which belongs to experiment 6. The remaining ones are contained in the appendix (see figure A.1,figure A.2,figure A.3).

Table 5.1.: Weight values

		Coarse grid			γ Fine grid		C Fine grid	
		C_1	γ_1	%	γ_{opt}	%	C_{opt}	%
Experiment 1								
UB	W^1	2^{-5}		55.30%				
2C	W^2	2^{15}		73.61%				
L	W^3	2^{13}		73.74%			$2^{11.25}$	73.89%
Experiment 2								
B	W^1	2^{-5}		51.46%				
2C	W^2	2^{15}		66.80%				
L	W^3	2^{5}		67.02%			$2^{6.75}$	67.04%
Experiment 3								
UB	W^1	2^{15}	2^3	78.87%				
2C	W^2	2^{7}	2^3	84.85%				
RBF	W^3	2^{11}	2^3	84.89%	$2^{3.5}$	85.01%	2^{9}	85.29%
Experiment 4								
B	W^1	2^{13}	2^3	88.03%	$2^{3.5}$	88.10%	$2^{11.25}$	88.78%
2C	W^2	2^{15}	2^3	65.99%				
RBF	W^3	2^{15}	2^3	61.55%				
Experiment 5								
UB	W^1	2^{13}		54.16%				
MC	W^2	2^{15}		70.46%			$2^{12.75}$	71.74%
L	W^3	2^{5}		55.21%				
Experiment 6								
B	W^1	2^{7}		71.37%				
MC	W^2	2^{13}		71.51%			2^{13}	71.51%
L	W^3	2^{7}		71.47%				
Experiment 7								
UB	W^1	2^{15}	2^3	67.38%				
MC	W^2	2^{-3}	2^3	77.83%	$2^{3.5}$	78.01%	$2^{-4.25}$	78.49%
RBF	W^3	2^{-5}	2^3	70.77%				
Experiment 8								
B	W^1	2^{15}	2^3	83.60%	$2^{3.25}$	83.67%	2^{15}	83.67%
MC	W^2	2^{11}	2^3	83.36%				
RBF	W^3	2^{11}	2^3	82.96%				

Table 5.2.: Binary Classification BAC Results

	Training Results	**Testing Results**
Experiment 1	73.89%	71.01%
Experiment 2	67.04%	61.82%
Experiment 3	85.29%	68.53%
Experiment 4	88.78%	73.3%

Table 5.3.: Multiclass Classification MAC Results

		Label 0	Label 1	Label 2	Label 4	Label 5	Label 6	Label 8	Label 9	Label 10	Total
Experiment 5	**Training**	58.55%	86.67%	41.18%	76.38%	87.10%	62.07%	71.61%	100%	66.07%	72.18%
	Testing	62.47%	43.60%	39.43%	38.07%	64.80%	62.06%	61.50%	18.15%	33.31%	47.04%
Experiment 6	**Training**	57.20%	80.90%	41.00%	69.30%	89.40%	67.90%	74.20%	100%	66.60%	71.83%
	Testing	60.58%	50.73%	36.80%	43.34%	63.31%	69.73%	57.45%	26.62%	46.98%	50.61%
Experiment 7	**Training**	61.70 %	90.67%	66.27%	92.97%	100%	94.83%	87.10%	100%	88.39%	86.88%
	Testing	60.83%	52.70%	46.62%	44.24%	41.84%	33.52%	54.97%	0%	21.04%	33.42%
Experiment 8	**Training**	85.7%	94.50%	89.00%	99.30%	99.90%	95.50%	93.70%	100%	93.60%	94.58%
	Testing	61.07%	27.04%	44.32 %	20.62%	57.30%	31.41%	17.74%	4.83%	15.35%	31.08%

Table 5.4.: Confusion Matrix Experiment 6

		Truth Labels								
		Label 0	**Label 1**	**Label 2**	**Label 4**	**Label 5**	**Label 6**	**Label 8**	**Label 9**	**Label 10**
Predicted Labels	**Label 0**	603328	1078	8173	5004	110	44	3665	7	56
	Label 1	93636	3743	541	4531	731	91	619	66	128
	Label 2	45771	2	30039	282	0	251	1840	0	5426
	Label 4	31010	87	2935	12076	250	386	368	0	14
	Label 5	4106	1068	97	3917	5074	10	6	0	1
	Label 6	46393	8	18171	1827	4	2082	3840	0	1462
	Label 8	131014	513	11652	92	25	58	17269	109	6263
	Label 9	1802	880	294	127	1817	0	190	66	39
	Label 10	38835	0	9720	8	3	64	2263	0	11865
Accuracy		60.58%	50.73%	36.80%	43.34%	63.31%	69.73%	57.45%	26.62%	46.98%

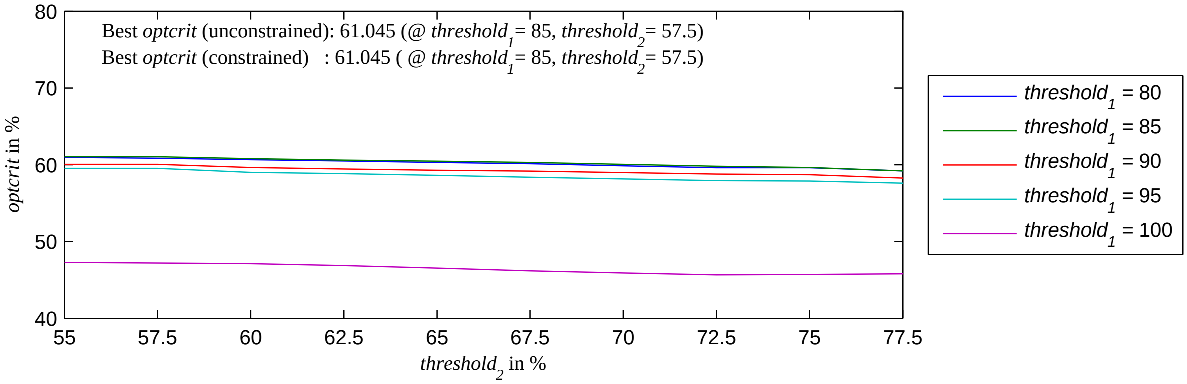
5.2. Episode detection results

The results which are outlined in this section are the ones which were obtained during the DPC. For each of the experiments a DPC with equal parameters was done. Figure 5.1 and figure 5.2 contain the results regarding *optcrit* which are obtained by applying the binary and the multi-class classification, respectively.

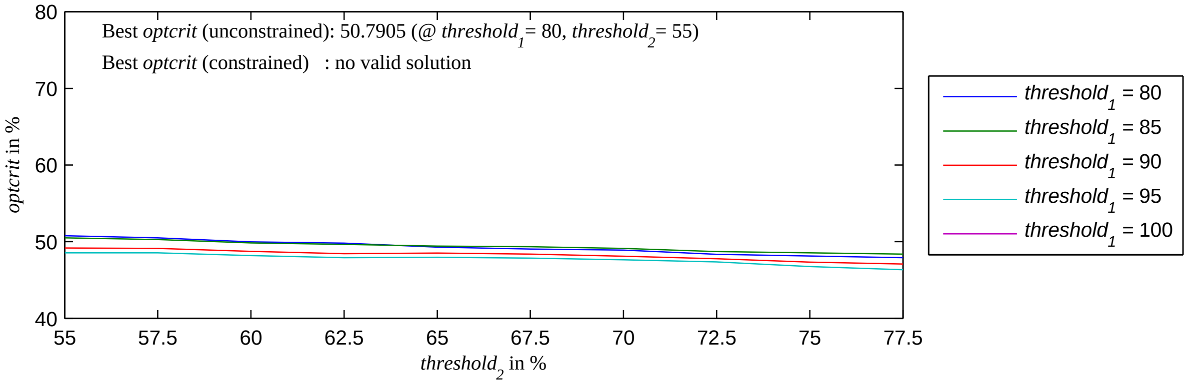
In each of the graphics each variation of the detection threshold $threshold_1$ constitutes a different test for the belonging experiment. The variations in the delineation threshold $threshold_1$ build up the abscissa over which *optcrit* is shown. Above each set of curves the best found *optcrit* and its belonging parameters are shown. Thereby, it is distinguished between unconstrained (i.e. equation 4.11 is not considered) and the constrained results (i.e. equation 4.11 is considered).

Table 5.5 and table 5.6 summarizes the single performance measures belonging to the best *optcrit* for the unconstrained and the constrained case, respectively.

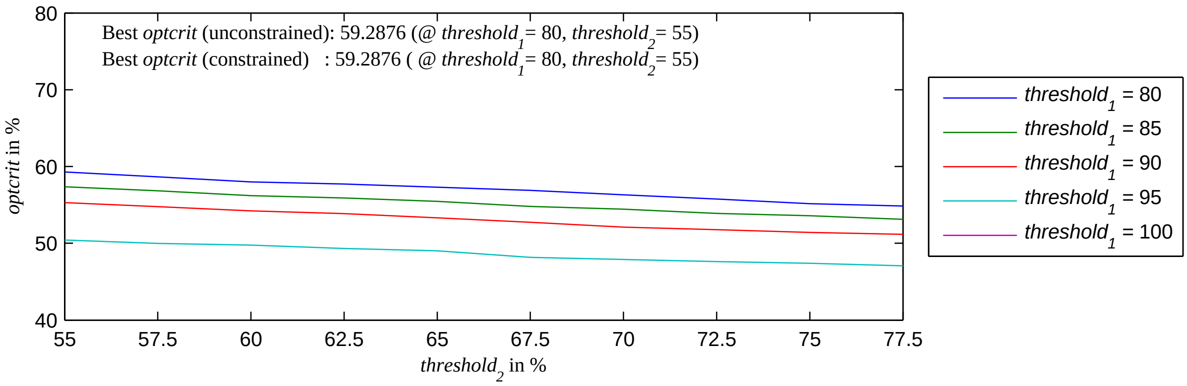
An even more detailed description of the behavior of these measures can be obtained by showing the resulting DPC curves. Figure 5.3 and figure 5.4 contain these curves for experiment 4. If *optcrit* exists for the constrained case and differs from the unconstrained case, as observed in experiment 4, the DPC curves for the constrained and the unconstrained case are given. Each of the curve shows the values which have been obtained by varying $threshold_2$. $threshold_1$ remains fixed. Thereby, the chosen value of $threshold_1$ is the one which belongs to the best found *optcrit* for the respective experiment. For Experiment 4 this is $threshold_1 = 85\,\%$ and $threshold_1 = 95\,\%$ for the unconstrained and for the constrained case, respectively. DPC curves of the remaining experiments are given in the appendix A (figure B.1 to B.9).



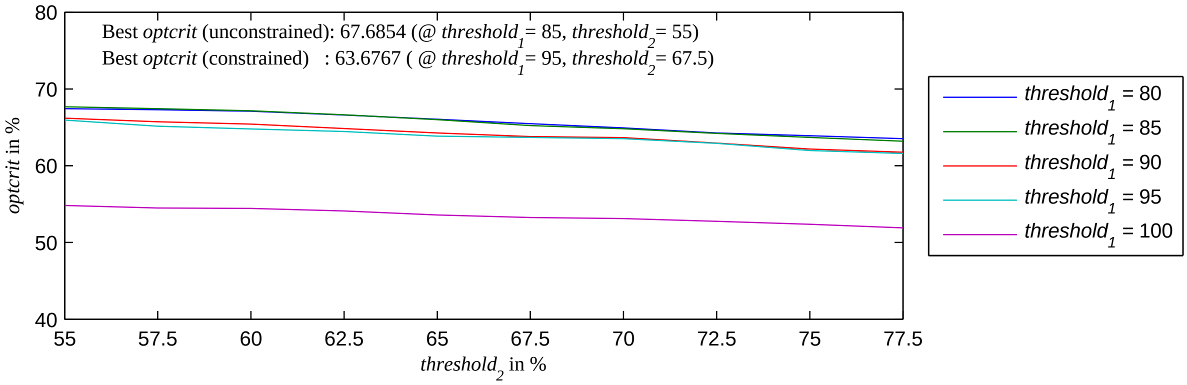
(a) Experiment 1



(b) Experiment 2

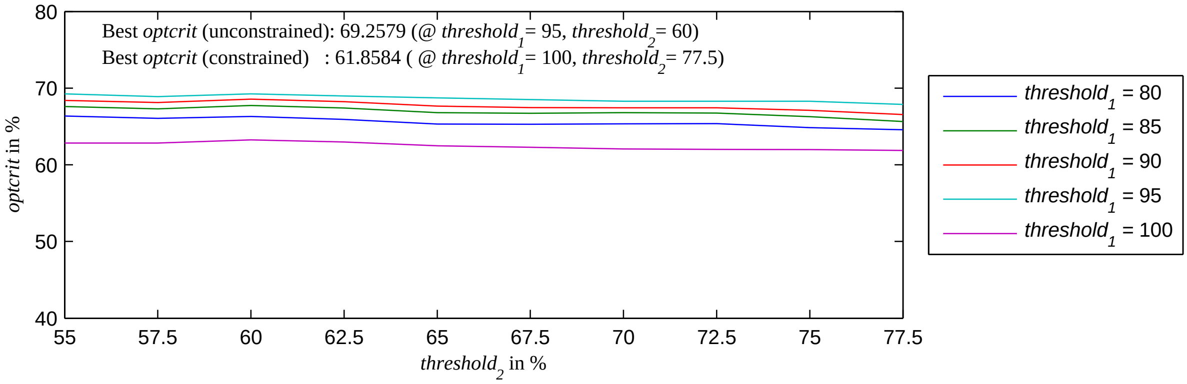


(c) Experiment 3

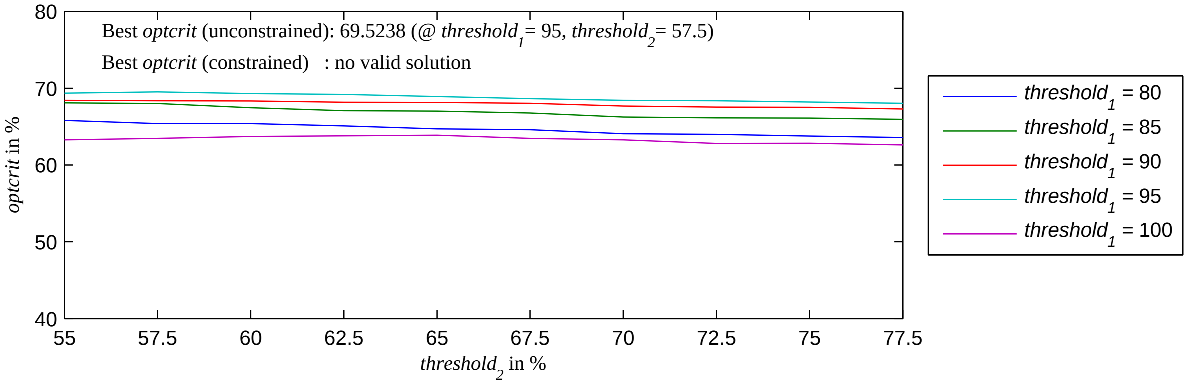


(d) Experiment 4

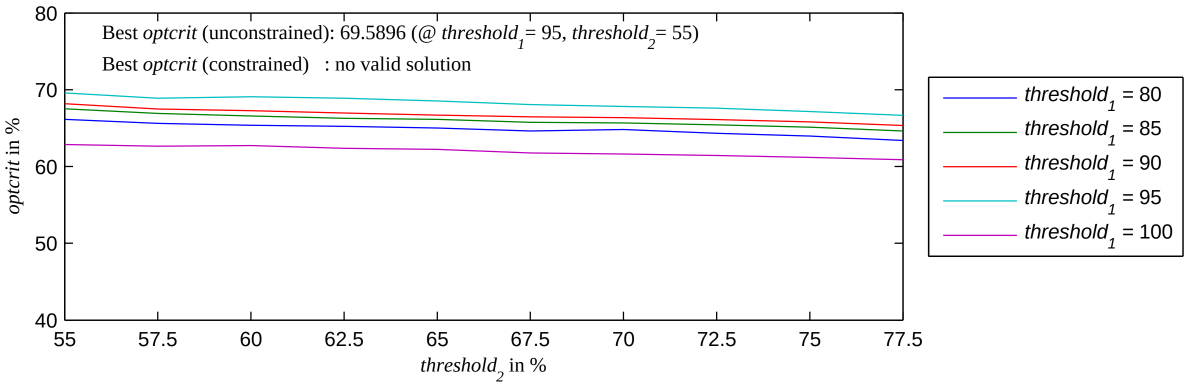
Figure 5.1.: Behaviour of *optcrit* for binary SVMs (threshold and optcrit values in %)



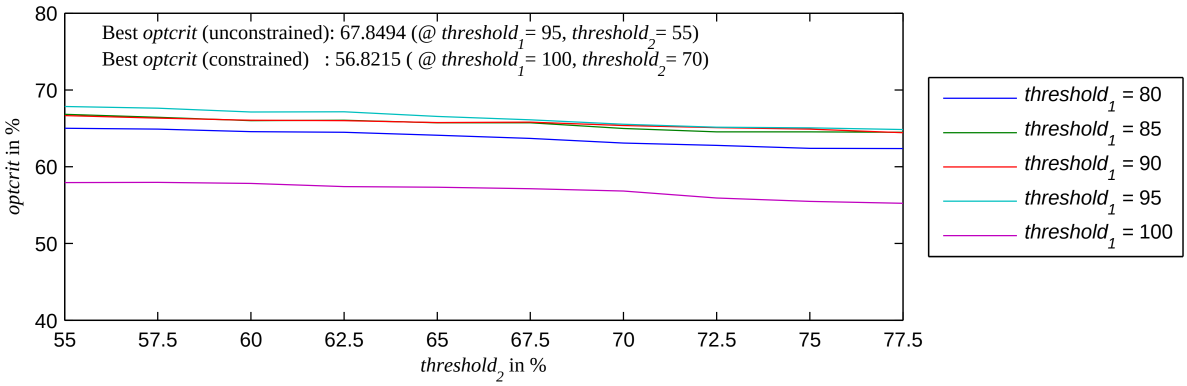
(a) Experiment 5



(b) Experiment 6



(c) Experiment 7



(d) Experiment 8

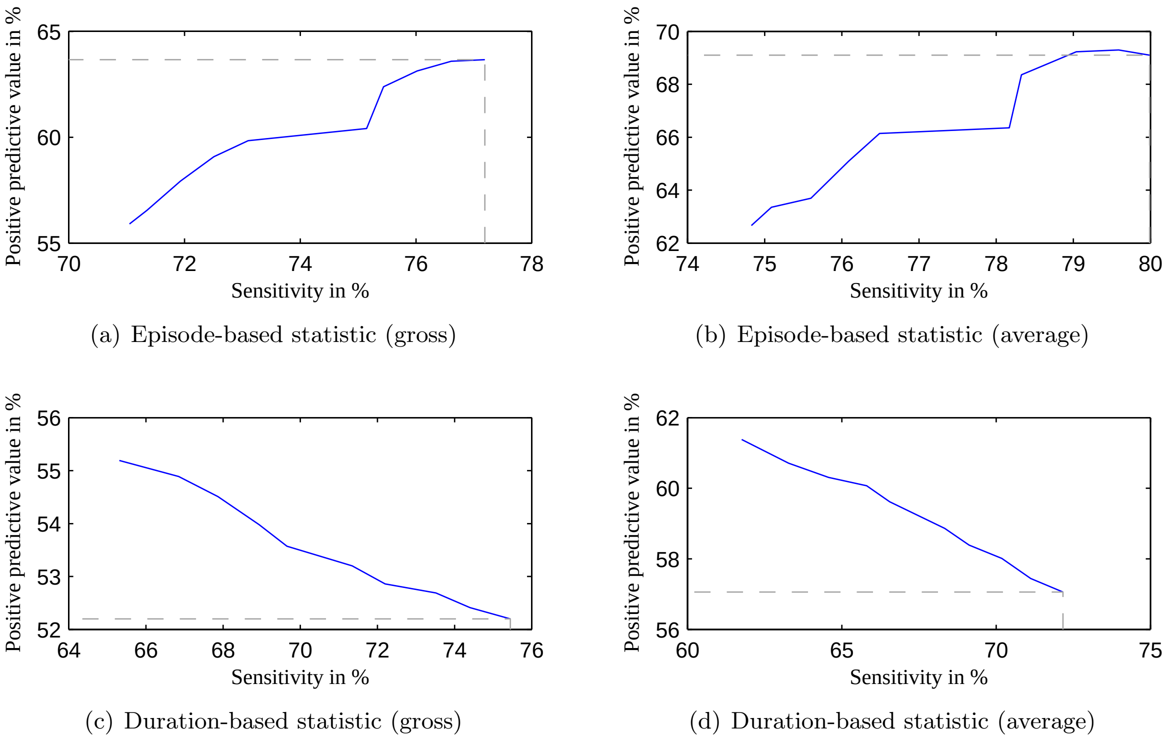
Figure 5.2.: Behaviour of *optcrit* for multi-class SVMs (threshold and optcrit values in %)

Table 5.5.: Unconstrained results for the detection of ST-T episodes (shown are the best values found in the DPC analysis selected based on the unconstrained *optcrit*)

Test run	**Results (in %)**				
	SE Se	SE +P	SD Se	SD +P	OPKRIT
Experiment 1	55.56 [g] 59.17 [a]	70.42 [g] 73.64 [a]	54.77 [g] 50.01 [a]	60.92 [g] 67.79 [a]	61.05
Experiment 2	54.39 [g] 53.50 [a]	54.75 [g] 59.15 [a]	48.41 [g] 40.75 [a]	46.72 [g] 50.99 [a]	50.79
Experiment 3	51.46 [g] 57.78 [a]	72.85 [g] 74.38 [a]	49.35 [g] 44.47 [a]	63.12 [g] 68.4 [a]	59.29
Experiment 4	77.19 [g] 79.99 [a]	63.66 [g] 69.1 [a]	75.45 [g] 72.17 [a]	52.2 [g] 57.06 [a]	67.69
Experiment 5	85.38 [g] 86.86 [a]	61.1 [g] 66.51 [a]	85.85 [g] 82.66 [a]	49.18 [g] 50.33 [a]	69.26
Experiment 6	88.01 [g] 88.59 [a]	62.15 [g] 65.59 [a]	89.29 [g] 86.19 [a]	46.05 [g] 48.46 [a]	69.52
Experiment 7	84.21 [g] 86.58 [a]	64.53 [g] 70.15 [a]	85.95 [g] 83.8 [a]	45.58 [g] 50.76 [a]	69.5896
Experiment 8	74.27 [g] 75.49 [a]	71.83 [g] 75.93 [a]	77.3 [g] 73.02 [a]	48.82 [g] 53.3 [a]	67.85

Table 5.6.: Constrained results for the detection of ST-T episodes (shown are the best values found in the DPC analysis selected based on the constrained *optcrit*)

Test run	**Results (in %)**				
	SE Se	SE +P	SD Se	SD +P	OPKRIT
Experiment 1	55.56 [g] 59.17 [a]	70.42 [g] 73.64 [a]	54.77 [g] 50.01 [a]	60.92 [g] 67.79 [a]	61.05
Experiment 2	No valid solution				
Experiment 3	51.46 [g] 57.78 [a]	72.85 [g] 74.38 [a]	49.35 [g] 44.47 [a]	63.12 [g] 68.4 [a]	59.29
Experiment 4	60.23 [g] 64.77 [a]	70.14 [g] 74.49 [a]	62.54 [g] 58.08 [a]	56.16 [g] 65.01 [a]	63.68
Experiment 5	61.4 [g] 63.48 [a]	63.69 [g] 68.91 [a]	62.18 [g] 55.95 [a]	57.85 [g] 62.27 [a]	61.86
Experiment 6	No valid solution				
Experiment 7	No valid solution				
Experiment 8	47.08 [g] 52.24 [a]	74.73 [g] 76.14 [a]	48.94 [g] 43.81 [a]	57.92 [g] 62.53 [a]	56.82



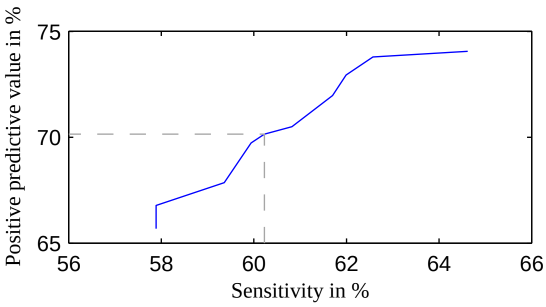
(a) Episode-based statistic (gross)

(b) Episode-based statistic (average)

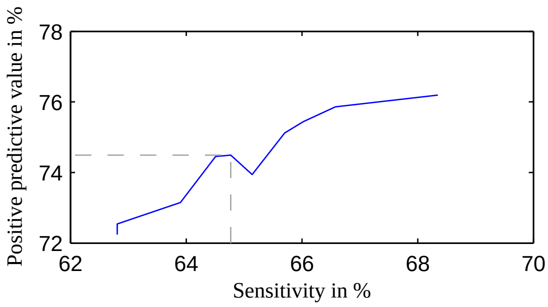
(c) Duration-based statistic (gross)

(d) Duration-based statistic (average)

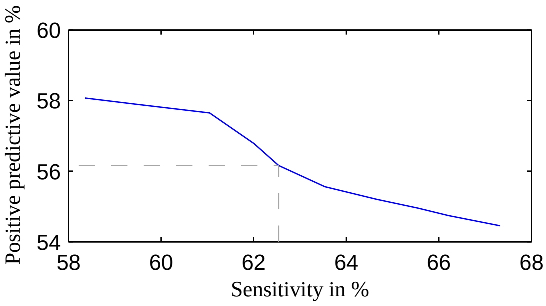
Figure 5.3.: DPC analysis for Experiment 4 (carried out @ best *optcrit*, i.e. $threshold_1 = 85\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*); the gray colored lines indicate the location of *optcrit*



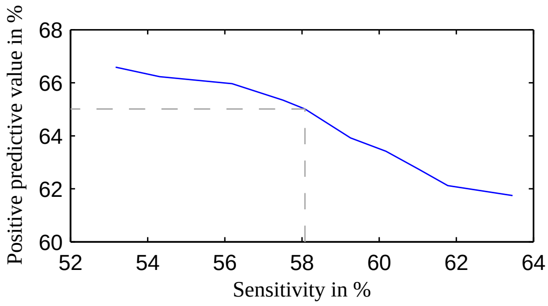
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)



(c) Duration-based statistic (gross)



(d) Duration-based statistic (average)

Figure 5.4.: DPC analysis for Experiment 4 (carried out @ best *optcrit*, i.e. $threshold_1 = 95\,\%$, $threshold_2$ is varied; results for constrained *optcrit*); the gray colored lines indicate the location of *optcrit*

6. Discussion

6.1. SVM application

6.1.1. Beat classification results

As can be seen in table 5.2 and table 5.3 the best classification results by means of MAC are obtained by Experiment 4. From there, one can conclude that the combination of a balanced training subset, binary classification and RBF Kernel is the most suited choice.

Thereby, the results which are obtained during the training are very promising (up to 88.78 % obtained in Experiment 4). However, between the results of training and testing a significant drop occurs. This holds for the binary classification, but even more for the multi-class classification. A possible reason for this behaviour can be found in the limited number of records which are used for training (just 10 records have been used). By such a small number of records just a limited number of realization are provided to the training algorithm. Considering the high variability of possible modifications regarding the ST-T-segment this issue becomes an comprehensible problem. The assumption of a too small number of training examples is supported by the behaviour which is examined in the case of multi-class classification: here the drop is even more destinctive.

However, owing to the computational complexity the elimination of this problem is not an easy task. Figure 6.1 contains the results of some tests on the computational costs of using the SVM. These tests measure the time amount which is required for training/testing a SVM with RBF kernel on an increasing number of patterns. The exponential increase as well as the large absolute time amounts which occur already at small numbers of pattern clarify the problem. In spite of using LibSVM as a software package for large datasets this behaviour constitutes a strong limitation concerning the training points that can be used. The selection of just 10 records and 8000 beats for training met these concerns, but apparently at the expense of the classification results.

6.1.2. Parameter search

The results which are obtained in this thesis clarify the general importance of choosing adequate parameters. This can be seen from table 5.1 when comparing the results which are obtained at after applying the coarse grid (first column) to the final results (last column). Significant changes are observed for all nearly all experiments thus pointing out the importance of a suited parameter selection method. Within this work a combined method for finding suited kernel parameters and class weights has been used.

To find the best parameters regarding the used kernel a method which builds up on the ideas of *grid.py* (which is provided by LibSVM) was chosen. As the obtained results are in accordance with the expected ones no further discussion of this method and its results will be carried out.

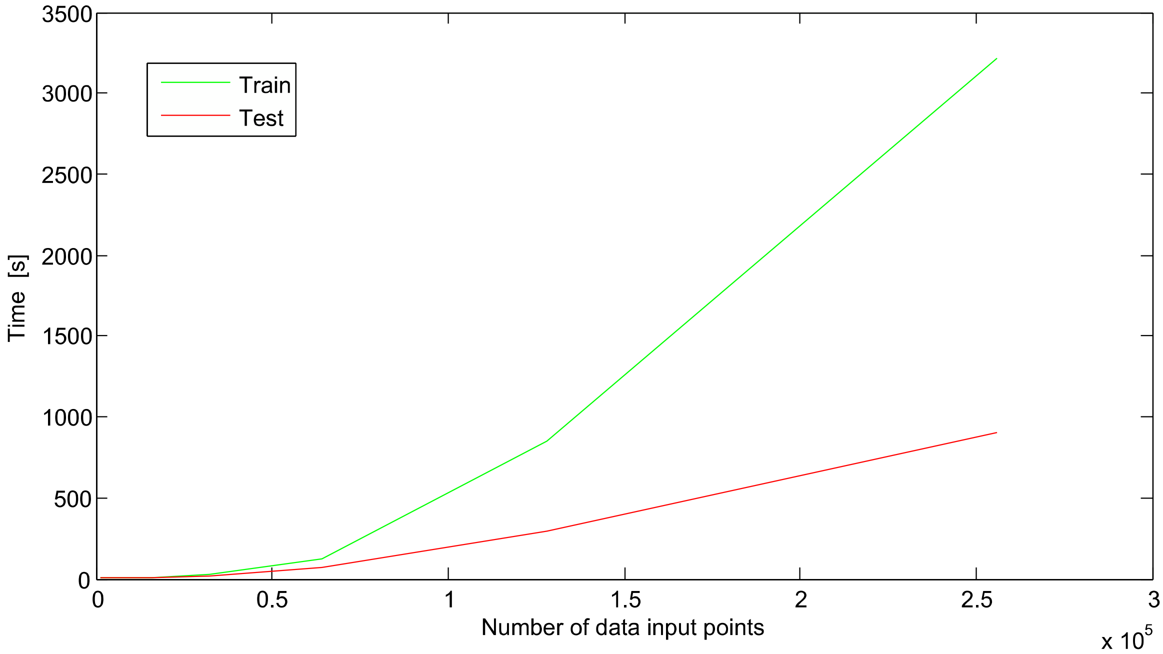


Figure 6.1.: Computational cost by means of time consumption for different number of pattern for training/testing

Regarding the class weights three different methods have been evaluated. The evaluated methods originate from ideas which have been depicted in the literature. By analyzing the results one must conclude that an optimal selection can not be given.

In a closer analysis one can identify that in binary classification the third criterion provides in 3 of 4 experiments the best evaluation results. Furthermore in these 3 experiments the second criterion provides a similar evaluation results. Just in the experiment 4 the first criterion is outperforms the others. In multi-class classification a similar behavior is observed. In this case the second criterion provides in 3 of 4 experiments the best evaluation results. Furthermore, just in experiment 8 the first criterion is the best one. However, as in the case of the Kernel parameters the class weights affect the obtainable results significantly. The consideration of different choices as it is done in this work thus constitutes an important step even for future works.

6.2. Episode detection

To estimate the results which were obtained regarding the episode detection one must distinguish between the unconstrained and the constrained results. With respect to the latter, one must conclude that no satisfactory grade was obtained. This rating is based on a comparison between the actual results and those reported in [75]. Table 6.1 summarizes the results.

Considering unconstrained results the proposed method obtains similar values to [75]. However, as this is reached at the expense of fulfilling the additional criterion which is defined by equation 4.11 those results cannot be compared directly to the ones depicted in [75]. But even

Table 6.1.: Comparison of obtained results for the unconstrained case, the constrained case and the results reported in [75]

	Results (in %)				
	SE Se	SE +P	SD Se	SD +P	OPKRIT
Unconstrained (Experiment 6)	88.01 [g] 88.59 [a]	62.15 [g] 65.59 [a]	89.29 [g] 86.19 [a]	46.05 [g] 48.46 [a]	69.52
Constrained (Experiment 4)	60.23 [g] 64.77 [a]	70.14 [g] 74.49 [a]	62.54 [g] 58.08 [a]	56.16 [g] 65.01 [a]	63.68
Literature [75]	72,50 [g] 72,9 [a]	72,5 [g] 77,9 [a]	72,8 [g] 65,7 [a]	55,2 [g] 66,9 [a]	69,2

if not directly comparable the results show that even on the basis of beat labels an episode detection of similar performance is possible. Thus, to increase the positive predictive should be the goal for future research.

However, considering all observed results this is not an easy task; thereby, the problem is not only specifically to increase the positive predictive value, but the ways to take influence on the results in general: it turned out, that the behaviour regarding the episode detection based on the beat labels is very complex. The DPC curves on the episode based performance measure clarify this. Two major observations can be found:

1. Inverse behaviour: the DPC curves show, at least in some ranges, an unexpected behaviour, i.e. SE Se and SE +P increase at the same time while SD Se and SD +P show the expected behaviour (see for example figure 5.3).
2. Inactive behaviour: In general the results are influenced only to a very limited extend by varying the thresholds. This can be observed by examining the given DPC curves, but even by the *optcrit* (see figure 5.1 and figure 5.2). Owing to its integrative character *optcrit* indeed compensates the variations in the single performance measures to some extend; nevertheless there should be typically found a clear maximum when all values are arranged in a mean range [75]. Such a maximum does not occur in the presented evaluation which accounts for the difficulty to influence the results in a systematic manner.

A very interesting aspect is the minor difference between the results obtained for binary and multi-class classification, respectively. As the results on beat level differ significantly one could expect distinctive differences even for the episode classification. However, before applying the episode detection to the multi-class label, these labels where transformed to a binary representation; i.e. the higher output space is only used to train and apply the beat classifier. Afterwards the gained information is broken down to the binary representation. Considering this, as well as the medical point of view in which all the non-normal beats are of interest, it becomes evident that not only the number of wrongly classified beats but also the *kind of missclassification* matters. These considerations imply that the confusion matrices which serve as basis for the MAC are evaluated considering there ability to distinguish between normal and ischaemic beats. Table 6.2 to table 6.5 shows the results of this binary examination of the confusion matrices. thereby "Old MAC" specifies the value obtained from the multi-class examination, "New MAC" the value obtained after breaking the confusion matrix down to normal and ischaemic.

This examination clarifies the found behaviour in the episode detection results. As there is no difference between the type of episodes the binary consideration of the multi-class results

Table 6.2.: Binary Confusion Matrix Experiment 5

	Truth Labels	
	Normal	**Ischaemic**
Predicted Labels	622195	21600
	373700	161827
Accuracy	62.78%	88.22%
New MAC	75.50 %	
Old MAC	47.04 %	

Table 6.3.: Binary Confusion Matrix Experiment 6

	Truth Labels	
	Normal	**Ischaemic**
Predicted Labels	603328	18137
	995895	165290
Accuracy	89.15%	90.11%
New MAC	89.63 %	
Old MAC	50.61 %	

Table 6.4.: Binary Confusion Matrix Experiment 7

	Truth Labels	
	Normal	**Ischaemic**
Predicted Labels	605883	21749
	390012	161678
Accuracy	60.84%	88.14%
New MAC	74,49 %	
Old MAC	33,42 %	

Table 6.5.: Binary Confusion Matrix Experiment 8

	Truth Labels	
	Normal	**Ischaemic**
Predicted Labels	608162	37072
	387733	146355
Accuracy	61.07%	79.79%
New MAC	70.43 %	
Old MAC	31.08 %	

is decisive for the episode detection results. These results even outperfom the ”native” binary results which accounts for the usage of multi-class SVM in the given context even when the original MAC does not support this choice.

7. Summary and forecast

The presented work constitutes an approach to detect ST-T-episodes based on beat classes. To classify the beats of the EDB SVMs have been applied. Different experiments regarding variants of SVMs (binary SVM - multi-class SVM, linear Kernel - RBF Kernel, balanced training data - unbalanced training data) have been carried out in order to extract information on suitable SVM models.

The obtained results show the applicability of SVMs in the given context. However, the comparison of these results to previously obtained results on the detection of ST-T-episodes even clarifies the need for further investigation on the topic.

The observations which may be deduced from this thesis give cause for the assumption that the relatively small number of training pattern is the main reason of the limitations which are examined within the results. However, expanding the training set is as shown before not an easy task.

Future works should address this issue. Thereby, two strategies may be pursued

1. The training subset could be expanded by including more morphologies but maintaining the same size; this would imply that the number of training records should be increased, from each record a smaller number of beats is selected and the applied methods (LibSVM) could be maintained

2. The training subset could be expanded by including more morphologies and increasing the size; this would imply that the number of training records should be increased, the overall number of beats is increased and new training methods must be applied.

The latter strategy obviously constitutes the more comprehensive approach. This strategy should be pursued if the application of SVMs is planned even in other contexts within the working group. Thereto, a possible approach could evaluate the usability and efficiency, respectively, of the software package LaSVM [14]. LaSVM especially is intended to be used in the case of large datasets (in the sense of a big number of training pattern). As LaSVM supports the same data formats as LibSVM most of the experience and some methods which have been devised in this work could be transfered thus allowing an easy integration of LaSVM functions.

Bibliography

[1] *Bioinformatics Toolbox*. http://www.mathworks.com, Abruf: 12.10.2010

[2] *PRTools*. Version: 2004. http://www.prtools.org/about.html, Abruf: 12.10.2010

[3] *Tiberius Data Mining*. Version: 2009. http://www.tiberius.biz/, Abruf: 12.10.2010

[4] *Kernel Machines Blackboard*. Version: 2010. www.kernel-machines.org/phpbb, Abruf: 12.10.2010

[5] Abe, Dr S. ; Professor Sameer Singh, PhD (Hrsg.): *Support Vector Machines for pattern Classification*. Springer, 2005

[6] Acir, Nurettin: Classification of ECG beats by using a fast least square support vector machines with a dynamic programming feature selection algorithm. In: *Neural Comput. Appl.* 14 (2005), Nr. 4, S. 299–309. http://dx.doi.org/http://dx.doi.org/10.1007/s00521-005-0466-z. – DOI http://dx.doi.org/10.1007/s00521–005–0466–z. – ISSN 0941–0643

[7] Aipperspach, Wolfgang: *Untersuchung der Herzratenvariabilität im Hinblick auf die Ischämiedetektion*, Institut für Biomedizinische Technik, TU Dresden, Diplomarbeit, 2009

[8] Albrecht: *SVM Toolbox*. Version: 2010. http://www.highgames.com/svm_toolbox/, Abruf: 12.10.2010

[9] Alegre, E. ; Gonzalez-Castro, V. ; Suarez, S. ; Castejon, M.: Comparison of supervised and unsupervised methods to classify boar acrosomes using texture descriptors. In: *Proc. Int. Symp. ELMAR ELMAR '09*, 2009, S. 65–70

[10] Ben-Hur, Asa ; Horn, David ; Siegelmann, Hava T. ; Vapnik, Vladimir: Support Vector Clustering. In: *Journal of Machine Learning Research* 2 (2001), S. 125–137

[11] Ben-Hur, Asa ; Ong, Cheng S. ; Sonnenburg, Sören ; Schölkopf, Bernhard ; Rätsch, Gunnar: Support vector machines and kernels for computational biology. In: *PLoS Comput Biol* 4 (2008), Oct, Nr. 10, e1000173. http://dx.doi.org/10.1371/journal.pcbi.1000173. – DOI 10.1371/journal.pcbi.1000173

[12] Übeyli, E.D.: ECG beats classification using multiclass support vector machines with error correcting output codes. In: *Digital Signal Processing* 17 (2007), Nr. 3, 675 - 684. http://dx.doi.org/DOI:10.1016/j.dsp.2006.11.009. – DOI DOI: 10.1016/j.dsp.2006.11.009. – ISSN 1051–2004

[13] Bjerregaard, Preben ; El-Shafei, Amr ; Kotar, Susan L. ; Labovitz, Arthur J.: ST segment analysis by Holter Monitoring: methodological considerations. In: *Ann Noninvasive Electrocardiol* 8 (2003), Jul, Nr. 3, S. 200–207

[14] Bordes, Antoine ; Ertekin, Seyda ; Weston, Jason ; Bottou, Léon: Fast Kernel Classifiers With Online And Active Learning. In: *JOURNAL OF MACHINE LEARNING RESEARCH* 6 (2005), S. 1579–1619

[15] Brandes, Axel ; Bethge, Klaus-Peter: [Long term electrocardiography (Holter monitoring)]. In: *Herzschrittmacherther Elektrophysiol* 19 (2008), Sep, Nr. 3, 107–129. http://dx.doi.org/10.1007/s00399-008-0010-9. – DOI 10.1007/s00399–008–0010–9

[16] Braunwald, Eugene ; Antman, Elliott M. ; Beasley, John W. ; Califf, Robert M. ; Cheitlin, Melvin D. ; Hochman, Judith S. ; Jones, Robert H. ; Kereiakes, Dean ; Kupersmith, Joel ; Levin, Thomas N. ; Pepine, Carl J. ; Schaeffer, John W. ; Smith, Earl E. ; Steward, David E. ; Theroux, Pierre ; Gibbons, Raymond J. ; Alpert, Joseph S. ; Faxon, David P. ; Fuster, Valentin ; Gregoratos, Gabriel ; Hiratzka, Loren F. ; Jacobs, Alice K. ; Smith, Sidney C. ; Patients With Unstable Angina), American C. o.: ACC/AHA guideline update for the management of patients with unstable angina and non-ST-segment elevation myocardial infarction–2002: summary article: a report of the American College of Cardiology/American Heart Association Task Force on Practice Guidelines (Committee on the Management of Patients With Unstable Angina). In: *Circulation* 106 (2002), Oct, Nr. 14, S. 1893–1900

[17] Burmeister, Jens: EKG-Verarbeitung / Fraunhofer IPMS. 2009. – Forschungsbericht

[18] Burmeister, Ole ; Reischl, Markus ; Bretthauer, Georg ; Mikut, Ralf: Data-Mining-Analysen mit der Matlab-Toolbox Gait-CAD (Data Mining Analyses with the Matlab Toolbox Gait-CAD). In: *Automatisierungstechnik* 56 (2008), Nr. 7, S. 381–389

[19] Canales, M. ; Mateo, J. ; Bailon, R. ; Laguna, P. ; Serrano, P.: Coherence between heart rate and dominant frequency of the time-frequency heart rate variability spectrum as ischemic marker in the exercise test. In: *Computers in Cardiology, 2003*, 2003, S. 805–808

[20] Chang, Chih-Chung ; Lin, Chih-Jen: *LIBSVM: a library for support vector machines.* Version: 2001. http://www.csie.ntu.edu.tw/~cjlin/libsvm, Abruf: 12.10.2010

[21] Clifford, G. D. (Hrsg.) ; Azuaje, F. (Hrsg.) ; McSharry, P. E. (Hrsg.): *Engineering in Medicine and Biology.* Bd. 1: *Advanced Methods and Tools for ECG Analysis*. 1. Norwood, MA, USA : Artech House, 2006. – ISBN 1-58053-966-1

[22] Cristianini, Nello ; Shawe-Taylor, John: *An introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, 2000

[23] De Lannoy, G. ; Francois, D. ; Delbeke, J ; Verleysen, M.: Feature Relevance Assessment in Automatic Inter-Patient Heart Beat Classification. In: *International Joint Conference on Biomedical Engineering Systems and Technologies*, 2010

[24] Diaz, J.D. ; Gonzalez, C. ; Escalona, O.: A Support Vector Machine for Predicting Spontaneous Termination of Paroxysmal Atrial Fibrillation Episodes. In: *Computers in Cardiology, 2006*, 2006, S. 942–952

[25] Enseleit, Frank ; Duru, Firat: Long-term continuous external electrocardiographic recording: a review. In: *Europace* 8 (2006), Apr, Nr. 4, 255–266. http://dx.doi.org/10.1093/europace/euj054. – DOI 10.1093/europace/euj054

[26] Fan, Wei ; Ma, Jie ; Zhu, He: A Thunderstorm Forecast Model Based on Weighted SVM and Data Field. In: *Proc. Sixth Int. Conf. Fuzzy Systems and Knowledge Discovery FSKD '09* Bd. 5, 2009, S. 160–164

[27] Faruqe, M. O. ; Al Mehedi Hasan, M.: Face recognition using PCA and SVM. In: *Proc. 3rd Int. Conf. Anti-counterfeiting, Security, and Identification in Communication ASID 2009*, 2009, S. 97–101

[28] Ghosh, D. ; Midya, B.L. ; Koley, C. ; Purkait, P.: Wavelet Aided SVM Analysis of ECG Signals for Cardiac Abnormality Detection. In: *INDICON, 2005 Annual IEEE*, 2005, S. 9 – 13

[29] Gibson, C. M. ; Ciaglo, Lauren N. ; Southard, Matthew C. ; Takao, Shaun ; Harrigan, Caitlin ; Lewis, Jason ; Filopei, Jason ; Lew, Michelle ; Murphy, Sabina A. ; Buros, Jacqueline: Diagnostic and prognostic value of ambulatory ECG (Holter) monitoring in patients with coronary heart disease: a review. In: *J Thromb Thrombolysis* 23 (2007), Apr, Nr. 2, 135–145. `http://dx.doi.org/10.1007/s11239-006-9015-6`. – DOI 10.1007/s11239–006–9015–6

[30] Grüning, Tina: *Formklassifikation der ventrikulären Repolarisation mittels neuronaler Netze*, Institut für Biomedizinische Technik, TU Dresden, Diplomarbeit, 2009

[31] Gunn: *MATLAB SVM Toolbox*. Version: 2004. `http://www.isis.ecs.soton.ac.uk/resources/svminfo/`, Abruf: 12.10.2010

[32] Hsu, Chih-Wei ; Chang, Chih-Chung ; Lin, Chih-Jen: A Practical Guide to Support Vector Classification / Department of Computer Science, National Taiwan University. Version: 2003. `http://www.csie.ntu.edu.tw/~cjlin/papers.html`. 2003. – Forschungsbericht

[33] Hsu, Chih-Wei ; Lin, Chih-Jen: A comparison of methods for multiclass support vector machines. 13 (2002), S. 415–425. `http://dx.doi.org/10.1109/72.991427`. – DOI 10.1109/72.991427

[34] Jager, F. ; Moody, G. B. ; Mark, R. G.: Detection of transient ST segment episodes during ambulatory ECG monitoring. In: *Computers and Biomedical Research* 31 (1998), Oct, Nr. 5, S. 305–322

[35] Jankowski, S. ; Tijink, J. ; Vumbaca, G. ; Balsi, M. ; Karpinski, G.: Morphological Analysis of ECG Holter Recordings by Support Vector Machines. In: Colosimo, Alfredo (Hrsg.) ; Sirabella, Paolo (Hrsg.) ; Giuliani, Alessandro (Hrsg.): *Medical Data Analysis* Bd. 2526, Springer Berlin / Heidelberg, 2002 (Lecture Notes in Computer Science), 134-143. – 10.1007/3-540-36104-9 15

[36] Joachims, Thorsten: *SVM light*. Version: 2008. `http://svmlight.joachims.org/`, Abruf: 12.10.2010

[37] Joshi, A.J. ; Chandran, S. ; Jayaraman, V.K. ; Kulkarni, B.D.: Hybrid SVM for Multiclass Arrhythmia Classification. In: *Bioinformatics and Biomedicine, 2009. BIBM '09. IEEE International Conference on*, 2009, S. 287 –290

[38] Khadtare, Mahesh S. ; Sahambi, J. S.: ECG Arrhythmia Analysis by Multicategory Support Vector Machine. In: *AACC*, 2004, S. 100–107

[39] Khandoker, A. H. ; Palaniswami, M. ; Karmakar, C. K.: Support Vector Machines for Automated Recognition of Obstructive Sleep Apnea Syndrome From ECG Recordings. 13 (2009), Nr. 1, S. 37–48. `http://dx.doi.org/10.1109/TITB.2008.2004495`. – DOI 10.1109/TITB.2008.2004495

[40] Konttila, T. ; Stenroos, M. ; Vaananen, H. ; Hanninen, H. ; Lindholm, M. ; Tierala, I. ; Katila, T.: Support vector classification of acute myocardial ischemia with reduced sets of body surface potential map electrodes. In: *Proc. Computers in Cardiology*, 2005, S. 869–872

[41] Laguna, P. ; Moody, G. B. ; García, J. ; Goldberger, A. L. ; Mark, R. G.: Analysis of the ST-T complex of the electrocardiogram using the Karhunen–Loève transform: adaptive monitoring and alternans detection. In: *Medical and Biological Engineering And Computing* 37 (1999), Mar, Nr. 2, S. 175–189

[42] Lei, Wai K. ; Dong, Ming C. ; Shi, Jun ; Fu, Bin B.: Automatic ECG interpretation via morphological feature extraction and SVM inference nets. In: *Proc. IEEE Asia Pacific Conf. Circuits and Systems APCCAS 2008*, 2008, S. 254–258

[43] Li, Jinghua ; Zhang, Congying ; Li, Zhenning: Battlefield Target Identification Based on Improved Grid-Search SVM Classifier. In: *Proc. Int. Conf. Computational Intelligence and Software Engineering CiSE 2009*, 2009, S. 1–4

[44] Mallat, S. ; Zhong, S.: Characterization of signals from multiscale edges. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 14 (1992), July, Nr. 7, S. 710–732. http://dx.doi.org/10.1109/34.142909. – DOI 10.1109/34.142909

[45] Mehta, S. S. ; Lingayat, N. S.: Biomedical signal processing using SVM. In: *Proc. ICTES Information and Communication Technology in Electrical Sciences (ICTES 2007) IET-UK Int. Conf*, 2007, S. 527–532

[46] Mehta, S. S. ; Lingayat, N.S: Support Vector Machine for Cardiac Beat Detection in Single Lead Electrocardiogram. In: *IAENG International Journal of Applied Mathematics* 36 (2007), S. 2

[47] Melgani, F. ; Bazi, Y.: Classification of Electrocardiogram Signals With Support Vector Machines and Particle Swarm Optimization. In: *Information Technology in Biomedicine, IEEE Transactions on* 12 (2008), sept., Nr. 5, S. 667 –677. http://dx.doi.org/10.1109/TITB.2008.923147. – DOI 10.1109/TITB.2008.923147. – ISSN 1089–7771

[48] Mohebbi, M. ; Moghadam, H.A.: An Algorithm for Automated Detection of Ischemic ECG Beats Using Support Vector Machines. In: *Proc. IEEE 15th Signal Processing and Communications Applications SIU 2007*, 2007, S. 1–4

[49] Nasiri, J. A. ; Naghibzadeh, M. ; Yazdi, H. S. ; Naghibzadeh, B.: ECG Arrhythmia Classification with Support Vector Machines and Genetic Algorithm. In: *Proc. Third UKSim European Symp. Computer Modeling and Simulation EMS '09*, 2009, S. 187–192

[50] Nauber, Richard: *Mikrocontroller-basierte Erstellung von Zeitreihen zur Beschreibung der ventrikul¨aren Repolarisation*, Institut für Biomedizinische Technik, TU Dresden, Diplomarbeit, 2010

[51] Noack, Alexander: *Charakterisierung von Episoden veränderter ventrikulärer Repolarisation unter Nutzung morphologiebasierter Trajektorien*, Institut für Biomedizinische Technik, TU Dresden, Diplomarbeit, 2009

[52] Osowski, S. ; Hoai, L. T. ; Markiewicz, T.: Support vector machine-based expert system for reliable heartbeat recognition. 51 (2004), Nr. 4, S. 582–589. http://dx.doi.org/10.1109/TBME.2004.824138. – DOI 10.1109/TBME.2004.824138

[53] Ozgur, Arzucan ; Alpaydin, Prof E. ; Prof, Assist ; Gingor, Tunga: Supervised and Unsupervised Machine Learning Techniques for Text Document Categorization. 2004. – Forschungsbericht

[54] Papaloukas, C. ; Fotiadis, D. I. ; Liavas, A. P. ; Likas, A. ; Michalis, L. K.: A knowledge-based technique for automated detection of ischaemic episodes in long duration electrocardiograms. In: *Med Biol Eng Comput* 39 (2001), Jan, Nr. 1, S. 105–112

[55] Papaloukas, Costas ; Fotiadis, Dimitrios I. ; Likas, Aristidis ; Stroumbis, Christos S. ; Michalis, Lampros K.: Use of a novel rule-based expert system in the detection of changes in the ST segment and the T wave in long duration ECGs. In: *J Electrocardiol* 35 (2002), Jan, Nr. 1, S. 27–34

[56] Park, K. S. ; Cho, B. H. ; Lee, D. H. ; Song, S. H. ; Lee, J. S. ; Chee, Y. J. ; Kim, I. Y. ; Kim, S. I.: Hierarchical support vector machine based heartbeat classification using higher order statistics and hermite basis function. In: *Proc. Computers in Cardiology*, 2008, S. 229–232

[57] Pueyo, E. ; Sornmo, L. ; Laguna, P.: QRS Slopes for Detection and Characterization of Myocardial Ischemia. In: *IEEE Transactions on Biomedical Engineering* 55 (2008), Februar, S. 468–477. `http://dx.doi.org/10.1109/TBME.2007.902228`. – DOI 10.1109/TBME.2007.902228. – ISSN 0018–9294

[58] Qin, Yu ping ; Wang, Xiu kun: Study on Multi-label Text Classification Based on SVM. In: *Proc. Sixth Int. Conf. Fuzzy Systems and Knowledge Discovery FSKD '09* Bd. 1, 2009, S. 300–304

[59] Rojo-Alvarez, José L. ; Barquero-Pérez, Oscar ; Mora-Jiménez, Inmaculada ; Everss, Estrella ; Rodríguez-González, Ana B. ; García-Alberola, Arcadi: Heart rate turbulence denoising using support vector machines. In: *IEEE Trans Biomed Eng* 56 (2009), Feb, Nr. 2, 310–319. `http://dx.doi.org/10.1109/TBME.2008.2003146`. – DOI 10.1109/TBME.2008.2003146

[60] Rüping, Stefan: *mySVM*. Version: 2004. `http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/index.html`, Abruf: 12.10.2010

[61] Sarle, W. S.: *Neural Network FAQ*. Version: 1997. `ftp://ftp.sas.com/pub/neural/FAQ.html`, Abruf: 12.10.2010. – Periodic posting to the Usenet newsgroup comp.ai.neural-nets

[62] Sherrod, Phil: *DTREG*. `http://www.dtreg.com/index.htm`, Abruf: 12.10.2010

[63] Shimodaira, Hiroshi ; Noma, Ken ichi ; Science, School Of I. ; Science, School Of I. ; Nakai, Mitsuru ; Sagayama, Shigeki: Dynamic Time-Alignment Kernel in Support Vector Machine. In: *School Of Information Science and Mitsuru Nakai and Ken-ichi Noma and Shigeki Sagayama* x (2001), S. x

[64] Smrdel, A. ; Jager, F.: Automated detection of transient ST-segment episodes in 24 h electrocardiograms. In: *Medical and Biological Engineering And Computing* 42 (2004), May, Nr. 3, S. 303–311

[65] Smrdel, A. ; Jager, F.: An Algorithm to Estimate the ST Segment Level in 24-Hour Ambulatory ECG Records. In: *Computers in Cardiology 2008*, 2008, S. 701–704

[66] Soman, K. P. ; Shyam, D. M. ; Madhavdas, P.: Efficient classification and analysis of ischemic heart disease using proximal support vector machines based decision trees. In: *Proc. TENCON 2003. Conf. Convergent Technologies for Asia-Pacific Region* Bd. 1, 2003, S. 214–217

[67] Stadler, R. W. ; Lu, S. N. ; Nelson, S. D. ; Stylos, L.: A real-time ST-segment monitoring algorithm for implantable devices. In: *J Electrocardiol* 34 Suppl (2001), S. 119–126

[68] Stern, Shlomo: State of the art in stress testing and ischaemia monitoring. In: *Card Electrophysiol Rev* 6 (2002), Sep, Nr. 3, S. 204–208

[69] Taddei, A. ; Distante, G. ; Emdin, M. ; Pisani, P. ; Moody, G. B. ; Zeelenberg, C. ; Marchesi, C.: The European ST-T database: standard for evaluating systems for the analysis of ST-T changes in ambulatory electrocardiography. In: *European Heart Journal* 13 (1992), Sep, Nr. 9, S. 1164–1172

[70] Tang, Yuchun ; Zhang, Yan-Qing ; Chawla, N. V. ; Krasser, S.: SVMs Modeling for Highly Imbalanced Classification. 39 (2009), Nr. 1, S. 281–288. `http://dx.doi.org/10.1109/TSMCB.2008.2002909`. – DOI 10.1109/TSMCB.2008.2002909

[71] Uyar, A. ; Gurgen, F.: Arrhythmia Classification Using Serial Fusion of Support Vector Machines and Logistic Regression. In: *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2007. IDAACS 2007. 4th IEEE Workshop on*, 2007, S. 560 –565

[72] Vapnik, Vladimir N.: *Statistical Learning Theory*. Wiley-Interscience, 1998

[73] Williams, T. ; Kelley, C.: *Gnuplot*. Version: 2010. `http://www.gnuplot.info`, Abruf: 12.10.2010

[74] Winters-Hilt, Stephen ; Merat, Sam: SVM clustering. In: *BMC Bioinformatics* 8 Suppl 7 (2007), S18. `http://dx.doi.org/10.1186/1471-2105-8-S7-S18`. – DOI 10.1186/1471–2105–8–S7–S18

[75] Zaunseder, S.: Ischämiedetektion aus dem Langzeit-EKG / Fraunhofer IPMS. 2010. – Forschungsbericht

[76] Zaunseder, S. ; Aipperspach, W. ; Poll, R: Discrimination between ischaemic and heart-rate related ST-episodes - Non-linear Classification for an Online Capable Approach. In: *International Joint Conference on Biomedical Engineering Systems and Technologies*, 2010, S. 245–251

[77] Zhang, Hao ; Zhang, Li-Qing: ECG analysis based on PCA and Support Vector Machines. In: *Proc. Int. Conf. Neural Networks and Brain ICNN&B '05* Bd. 2, 2005, S. 743–747

Appendix

A. Confusion Matrices

Table A.1.: Confusion Matrix Experiment 5

		Truth Labels								
		Label 0	**Label 1**	**Label 2**	**Label 4**	**Label 5**	**Label 6**	**Label 8**	**Label 9**	**Label 10**
Predicted Labels	**Label 0**	622195	1799	9642	6128	171	40	3739	4	77
	Label 1	72191	3217	413	3821	667	23	458	72	73
	Label 2	44857	1	32180	260	4	644	2954	0	7244
	Label 4	28160	60	2761	10607	196	293	534	0	19
	Label 5	4855	1209	141	4712	5193	11	4	0	2
	Label 6	36797	3	13249	2076	5	1853	2106	0	304
	Label 8	155318	592	13586	157	73	103	18487	1266	9003
	Label 9	523	498	6118	83	1704	0	148	45	120
	Label 10	30999	0	9532	20	1	19	1630	1	8412
Accuracy		62.47%	43.60%	39.43%	38.07%	64.80%	62.06%	61.50%	18.15%	33.31%

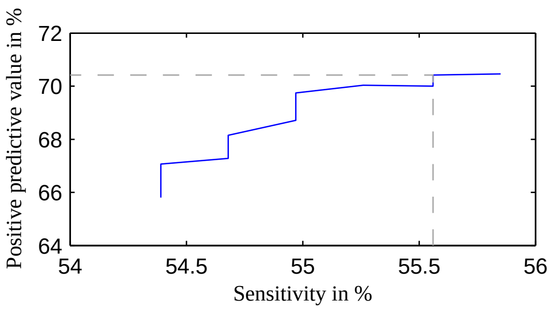
Table A.2.: Confusion Matrix Experiment 7

		Truth Labels								
		Label 0	**Label 1**	**Label 2**	**Label 4**	**Label 5**	**Label 6**	**Label 8**	**Label 9**	**Label 10**
Predicted Labels	**Label 0**	605883	1547	9514	6627	334	62	3572	33	60
	Label 1	54796	3889	208	2150	876	1	230	43	4
	Label 2	83298	29	38054	1307	112	1206	3225	3	8391
	Label 4	52329	925	5166	12327	2376	458	1765	79	4777
	Label 5	2854	440	98	2403	3353	12	23	0	0
	Label 6	26188	41	7865	2754	24	1001	1397	4	433
	Label 8	151439	282	13388	236	45	102	16525	75	6273
	Label 9	94	225	2	2	894	0	0	0	2
	Label 10	19014	1	7327	58	0	144	3323	11	5314
Accuracy		60.83%	52.70%	46.62%	44.24%	41.84%	33.52%	54.97%	0%	21.04%

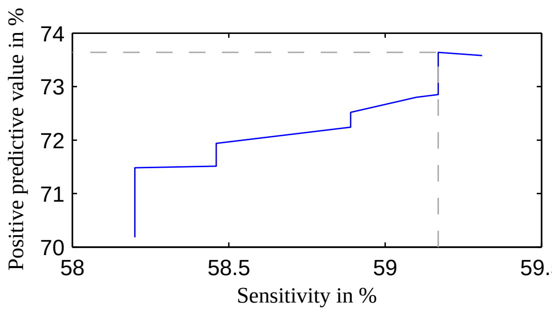
Table A.3.: Confusion Matrix Experiment 8

		Truth Labels								
		Label 0	**Label 1**	**Label 2**	**Label 4**	**Label 5**	**Label 6**	**Label 8**	**Label 9**	**Label 10**
Predicted Labels	**Label 0**	608162	2577	12719	7883	575	145	11734	102	1337
	Label 1	55887	1995	2127	1433	619	20	216	15	7
	Label 2	98905	162	36176	4251	196	1108	3348	0	8099
	Label 4	24001	229	2642	5745	821	111	416	0	232
	Label 5	29993	1209	2416	4342	4592	130	1633	62	4646
	Label 6	55412	162	11153	3957	26	938	5601	34	2896
	Label 8	106533	289	7936	138	32	62	5333	23	4150
	Label 9	287	748	91	7	1153	0	1	12	10
	Label 10	16715	8	6362	108	0	472	1778	0	3877
Accuracy		61.07%	27.04%	44.32 %	20.62%	57.30%	31.41%	17.74%	4.83%	15.35%

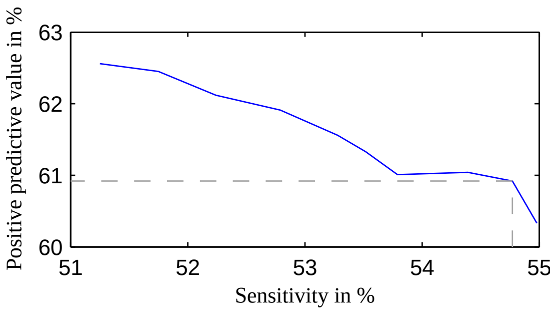
B. DPC Analysis



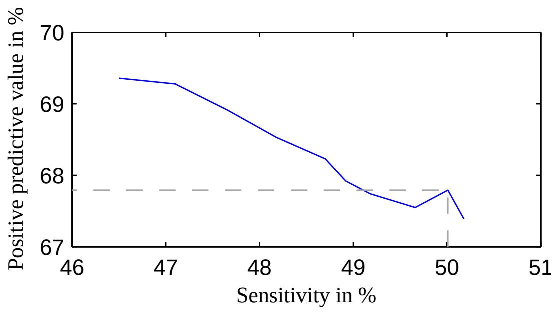
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)

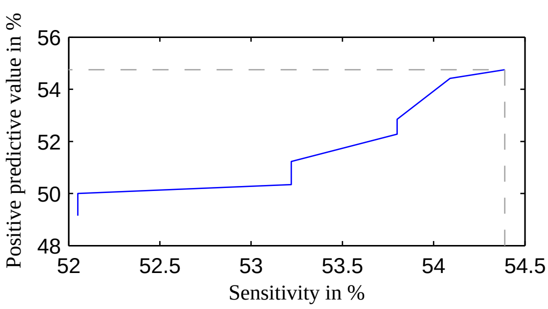


(c) Duration-based statistic (gross)

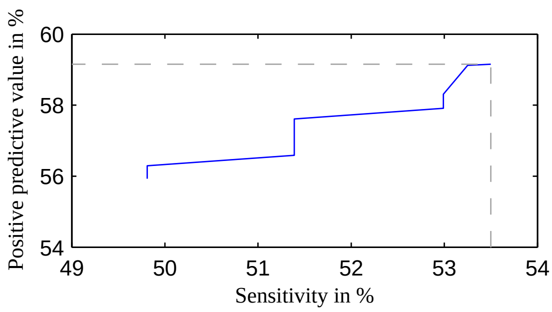


(d) Duration-based statistic (average)

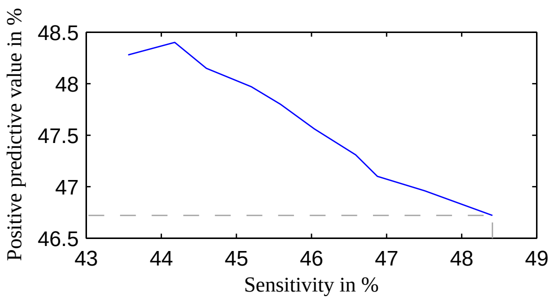
Figure B.1.: DPC analysis for Experiment 1 (carried out @ best *optcrit*, i.e. $threshold_1 = 85\,\%$, $threshold_2$ is varied; equal results for unconstrained and constrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



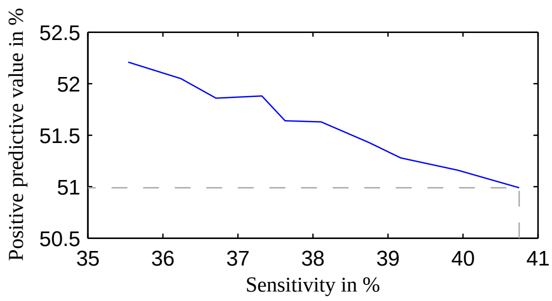
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)

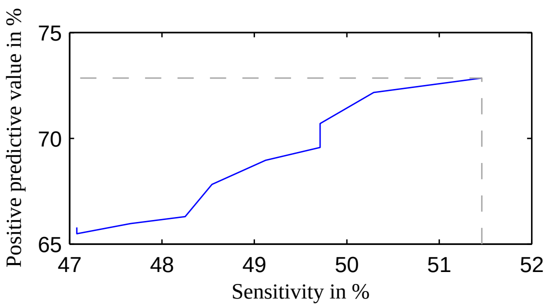


(c) Duration-based statistic (gross)

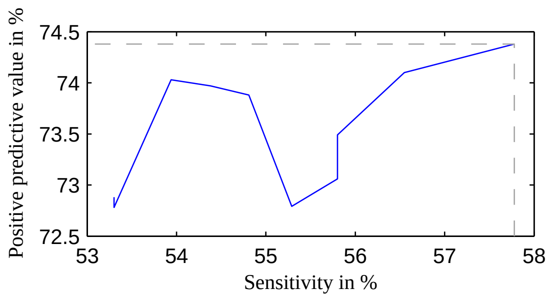


(d) Duration-based statistic (average)

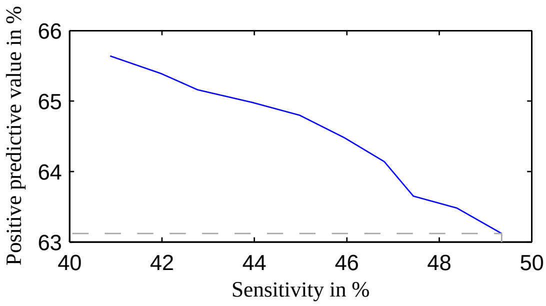
Figure B.2.: DPC analysis for Experiment 2 (carried out @ best *optcrit*, i.e. $threshold_1 = 80\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



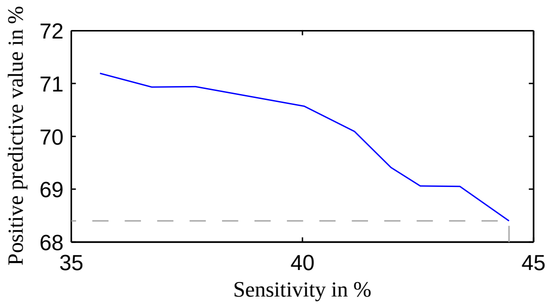
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)

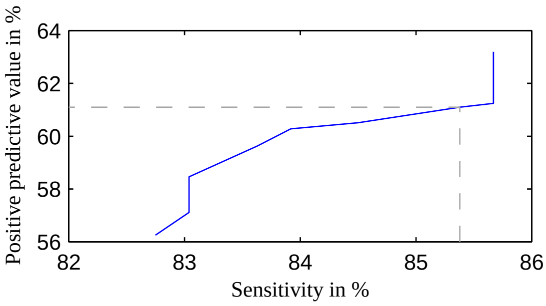


(c) Duration-based statistic (gross)

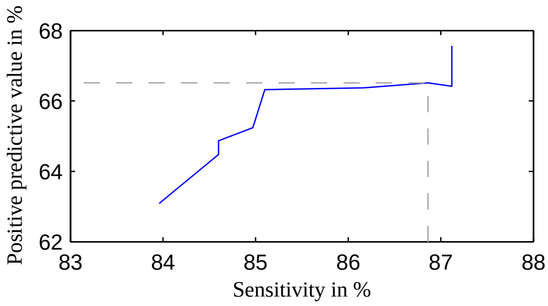


(d) Duration-based statistic (average)

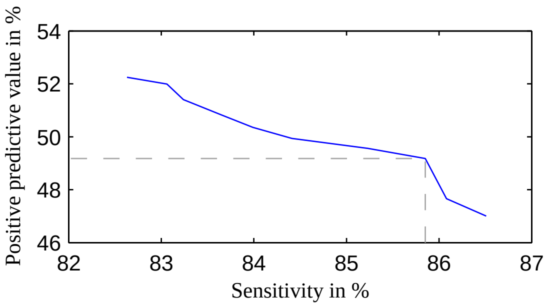
Figure B.3.: DPC analysis for Experiment 3 (carried out @ best *optcrit*, i.e. $threshold_1 = 80\,\%$, $threshold_2$ is varied; equal results for unconstrained and constrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



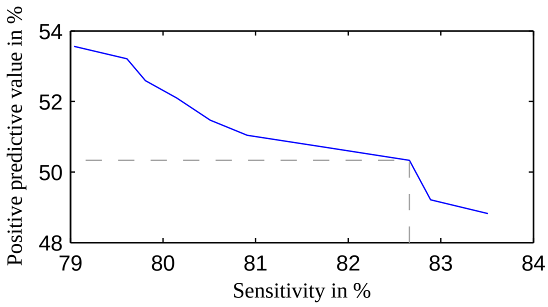
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)

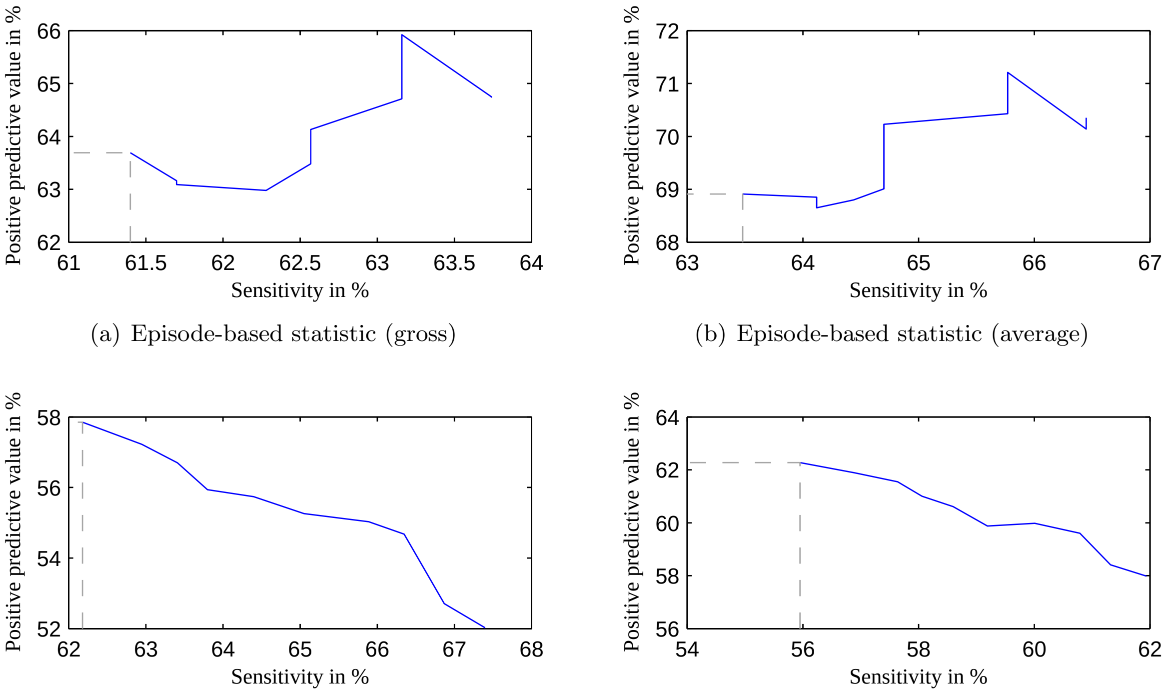


(c) Duration-based statistic (gross)



(d) Duration-based statistic (average)

Figure B.4.: DPC analysis for Experiment 5 (carried out @ best *optcrit*, i.e. $threshold_1 = 95\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



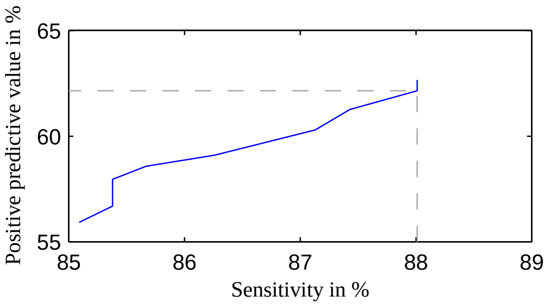
(a) Episode-based statistic (gross)

(b) Episode-based statistic (average)

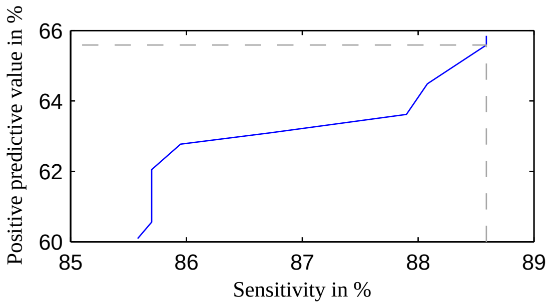
(c) Duration-based statistic (gross)

(d) Duration-based statistic (average)

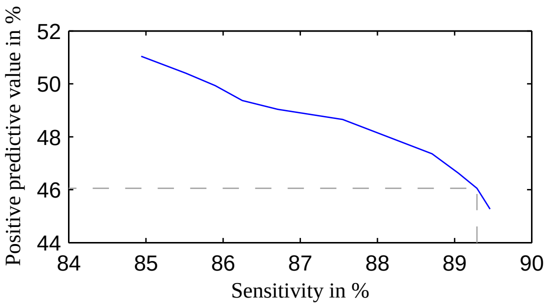
Figure B.5.: DPC analysis for Experiment 5 (carried out @ best $optcrit$, i.e. $threshold_1 = 100\,\%$, $threshold_2$ is varied; results for constrained $optcrit$; the gray colored lines indicate the location of $optcrit$)



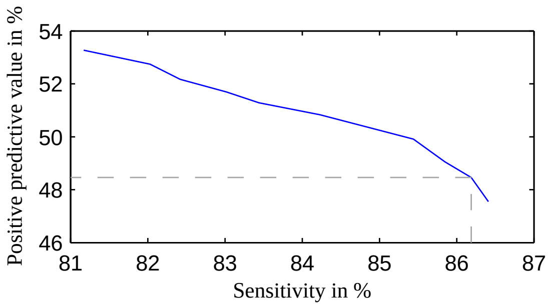
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)

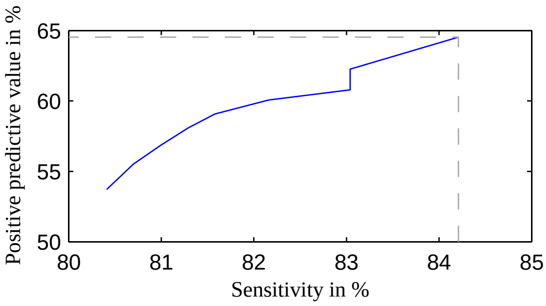


(c) Duration-based statistic (gross)

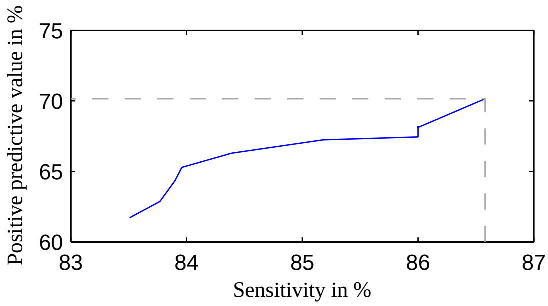


(d) Duration-based statistic (average)

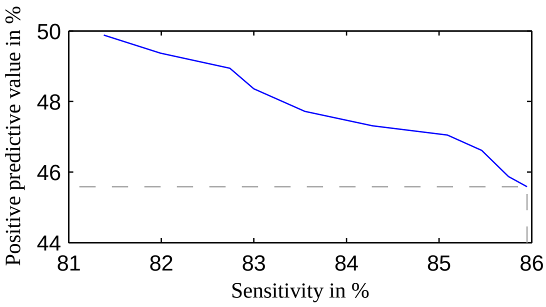
Figure B.6.: DPC analysis for Experiment 6 (carried out @ best *optcrit*, i.e. $threshold_1 = 95\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



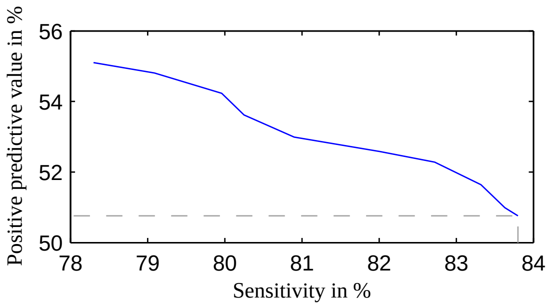
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)



(c) Duration-based statistic (gross)



(d) Duration-based statistic (average)

Figure B.7.: DPC analysis for Experiment 7 (carried out @ best *optcrit*, i.e. $threshold_1 = 95\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*; the gray colored lines indicate the location of *optcrit*)

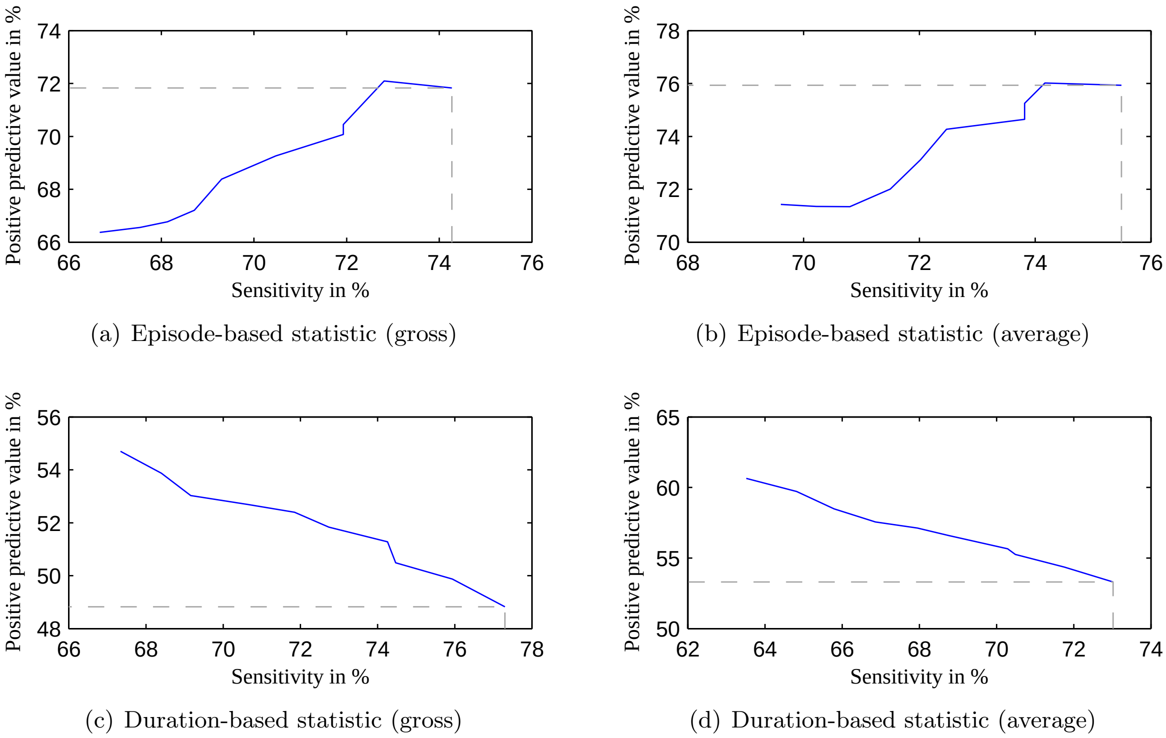
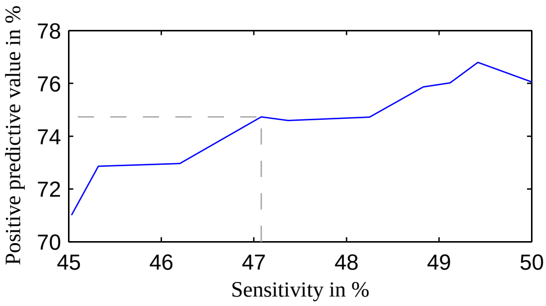
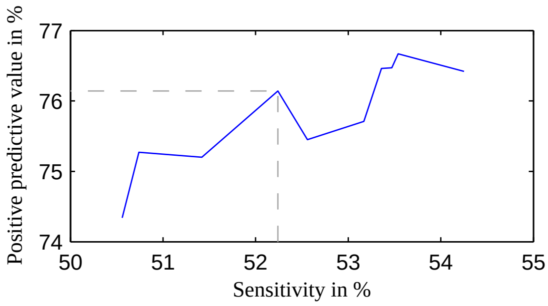


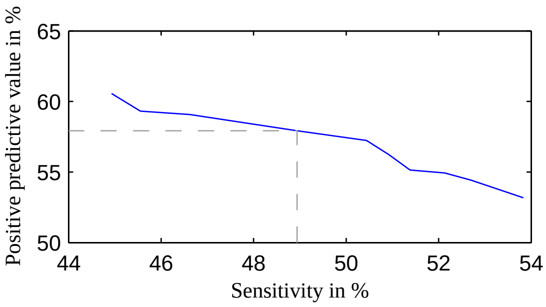
Figure B.8.: DPC analysis for Experiment 8 (carried out @ best *optcrit*, i.e. $threshold_1 = 95\,\%$, $threshold_2$ is varied; results for unconstrained *optcrit*; the gray colored lines indicate the location of *optcrit*)



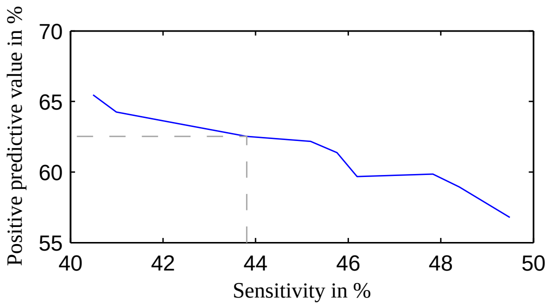
(a) Episode-based statistic (gross)



(b) Episode-based statistic (average)



(c) Duration-based statistic (gross)



(d) Duration-based statistic (average)

Figure B.9.: DPC analysis for Experiment 8 (carried out @ best *optcrit*, i.e. $threshold_1 = 100\,\%$, $threshold_2$ is varied; results for constrained *optcrit*; the gray colored lines indicate the location of *optcrit*)

C. Data Disc

Contents:

1. Thesis (pdf format)
2. Thesis (latex source code)
3. Matlab source Code (including LibSVM implementation)