Large-Scale Dimensionality Reduction Using Perturbation Theory and Singular Vectors

by

© *Majid Afshar*

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Doctor of Philosophy

Department of *Computer Science*

Memorial University of Newfoundland

March 2021

St. John's Newfoundland

Abstract

Massive volumes of high-dimensional data have become pervasive, with the number of features significantly exceeding the number of samples in many applications. This has resulted in a bottleneck for data mining applications and amplified the computational burden of machine learning algorithms that perform classification or pattern recognition. Dimensionality reduction can handle this problem in two ways, i.e. feature selection (FS) and feature extraction. In this thesis, we focus on FS, because, in many applications like bioinformatics, the domain experts need to validate a set of original features to corroborate the hypothesis of the prediction models. In processing the high-dimensional data, FS mainly involves detecting a limited number of important features among tens/hundreds of thousands of irrelevant and redundant features.

We start with filtering the irrelevant features using our proposed Sparse Least Squares (SLS) method, where a score is assigned to each feature, and the low-scoring features are removed using a soft threshold. To demonstrate the effectiveness of SLS, we used it to augment the well-known FS methods, thereby achieving substantially reduced running times while improving or at least maintaining the prediction accuracy of the models.

We developed a linear FS method (DRPT) which, upon data reduction by SLS, clusters the reduced data using the perturbation theory to detect correlations between the remaining features. Important features are ultimately selected from each cluster,

discarding the redundant features.

To extend the clustering applicability in grouping the redundant features, we proposed a new Singular Vectors FS (SVFS) method that is capable of both removing the irrelevant features and effectively clustering the remaining features. As such, the features in each cluster solely exhibit inner correlations with each other. The independently selected important features from different clusters comprise the final rank. Devising thresholds for filtering irrelevant and redundant features has facilitated the adaptability of our model to the particular needs of various applications.

A comprehensive evaluation based on benchmark biological and image datasets shows the superiority of our proposed methods compared to the state-of-the-art FS methods in terms of classification accuracy, running time, and memory usage.

Acknowledgements

First and foremost, I praise and thank God, the almighty, for his showers of blessings throughout my research work to complete this thesis successfully.

I would like to express my deep and sincere gratitude to my research supervisors Dr. Hamid Usefi, who showed me the right path when I was almost lost, and Dr. Saeed Samet, who taught me to be ethical and patient. Their expertise was invaluable in formulating the research questions and methodology. Their insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I would like to say a continuous and very loud thank you to my beloved wife, Haleh. Without her, I would not have been able to complete this thesis, and I would not have made it through my PhD degree.

I am extremely grateful to my mother and my father for their love, prayers, caring, and sacrifices from the other side of the world. I would also like to extend my gratitude to my brothers, sisters and my parents-in-law for their support and valuable prayers.

To my friends, I would like to say thanks for their constant encouragement and nonstop help.

Finally, my thanks go to all the people who have directly or indirectly supported me to complete this research work.

List of Acronyms

PSO Particle Swarm Optimization

ACO Ant Colony Optimization

GA Genetic Algorithm

FS Feature Selection

SLS Sparse Least-Squares

SVD Singular Value Decomposition

DRPT Dimension Reduction based on Perturbation Theory

SVFS Singular Vectors Feature Selection

GWAS Genome-Wide Association Study

SNP Single Nucleotide Polymorphism

RF Random Forest

SVM Support Vector Machines

mRMR Minimum Redundancy Maximum Relevance

SVMRFE Support Vector Machines Recursive Feature Elimination

MI Mutual Information

SOFT Simple Omnibus Format in Text

kNN k-Nearest Neighbors

CV Cross-Validation

CA Classification Accuracy

SD Standard Deviation

LFS Localized Feature Selection

LASSO Least Absolute Shrinkage and Selection Operator

LARS Least Angle Regression

HSIC-Lasso Hilbert-Schmidt Independence Criterion Least absolute shrinkage and selection operator

LAND Least Angle Nonlinear Distributed

LOFS Library of Online Feature Selection

LFI Learning Features added Individually

LGF Learning Grouped Features added sequentially

COA Coyote Optimization Algorithm

BCOA Binary Coyote Optimization Algorithm

PD Parkinson's Disease

CCM Conditional Covariance Minimization

BSF Breadth-First Search

CIFE Conditional Infomax Feature Extraction

JMI Joint Mutual Information

SMNB Sparse Multinomial Naive Bayes

GEO Gene Expression Omnibus

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

In nature, selection is one of the principal procedures of evolutionary change and is the primary mechanism responsible for the complexity and adaptive intricacy of living beings. Natural selection typically happens when there are limited resources for a large number of creatures. Understanding this type of selection has been becoming increasingly relevant in practical contexts using nature-inspired evolutionary algorithms [12, 17] such as Particle Swarm Optimization (PSO) [13], Ant Colony Optimization (ACO) [15], and Genetic Algorithm (GA) [33].

A feature is an individual measurable property of the data; it is also known as an attribute or variable. Feature Selection (FS) is an artificial selection or nature-inspired selection process in which a subset of optimal features is selected to help reduce the cost of computation requirement, reduce the effect of the curse of dimensionality, and reduce data acquisition in the future by identifying the minimum number of essential

features that can achieve competitive prediction accuracy [27, 28]. In the past years, in real-world machine learning or pattern recognition applications, the number of features has expanded from hundreds to thousands of features [28].

For example, one of the applications is gene microarray analysis, where gene expression data usually includes a small number of samples with high dimensions and noise [42, 3, 38]. A single gene chip is able to identify around tens of thousands of genes for one sample, while in some diseases or biological processes, only a few collections of genes are essential [23]. Moreover, testing numerous redundant genes requires tremendous memory space and significantly decreases data mining performance even for a few samples. Therefore, selecting the disease-related genes from the original gene expression will facilitate the design of proper remedial treatments [35, 49, 4].

In this thesis, we have demonstrated that the irrelevant features can be removed by assigning a weight to each feature, and filtering the features with low weights using a Sparse Least-Squares (SLS) method based on Singular Value Decomposition (SVD). We have shown that augmenting SLS to the well-known feature selection methods significantly reduces the running time while maintaining or even improving the prediction accuracy.

To achieve a holistic FS that, in addition to removing the irrelevant features using SLS, selects the important features, we proposed a Dimension Reduction based on Perturbation Theory (DRPT) method. This method takes advantage of SLS and defines a threshold based on the local maxima of the assigned weight and removes

those features whose weights are smaller than the threshold. To detect the correlations in the resulting reduced data, we applied a nested clustering approach, where features were clustered based on the perturbation theory. Each cluster was then turned into sub-clusters using the entropy of features. Finally, a feature was selected from each sub-cluster based on its assigned weight and entropy.

We extended our investigation in dimensionality reduction by proposing a Singular Vectors-based Feature Selection (SVFS). In this approach, we introduced a signature matrix in which the correlations between the features were encoded. The signature matrix is used for both removing irrelevant features and clustering the remaining features. To identify the important features among numerous redundant features, we clustered the correlated features, with each cluster containing features that only correlate with the features of the same cluster. Features with high mutual information with the class label are selected from the clusters as important features. Our comprehensive assessment over the benchmark and real-world genomic datasets has shown the overall superior performance of the developed methods in comparison with the state-of-the-art feature selection methods in terms of accuracy, running time, and memory usage.

1.1 Definitions

This section describes the frequently used terms in this thesis.

- **Class label**: The term class label is ordinarily used in supervised machine

learning, particularly in classification, where the goal is to learn a behavior that estimates the class label from the values of features. The class label always falls in a range of a limited number of distinct values.

- **Classification accuracy**: It refers to the accuracy of a classifier which is calculated based on the percentage of total correct classifier outcome divided by the total number of samples.

- **Important features**: Independent more informative features that are highly correlated to the class label. Indeed, finding the most important or relevant features is the goal of feature selection.

- **Redundant features**: In presence of important features, the redundant features provide less or no information about the class label or outcome

- **Irrelevant features**: These non-informative features do not correlate with the other features, including important and redundant features and the class label.

- **Feature selection**: refers to choose features to build feature vector according to the insights provided by the field experts or the literature.

- **Feature reduction**: refers to reduce the number of features using techniques similar to those presented in this dissertation, or according to insights from the field experts.

- **Feature extraction**: refers to extract new components as the combination of

the existing features, for instance, using principal component analysis.

- **Feature engineering**: refers to a combined set of feature reduction and feature extraction techniques that produce a new set of significant features to be used in building predictive and descriptive models.

1.2 Dimensionality Reduction

Dealing with large volumes of high-dimensional data has become common in various domains, such as bioinformatics [32], social media [7], and healthcare [14]. The rapid growth of high-dimensional data has introduced fruitful challenges to the effective and efficient data mining and machine learning approaches to discover knowledge from raw data. In some domains like bioinformatics, the number of samples is considerably smaller than the number of features. In such cases, applying data mining and machine learning algorithms may cause the curse of dimensionality, which typically arises when the data become sparser in the high-dimensional space [47], adversely affecting those approaches designed for low-dimensional spaces [10]. Moreover, considering a large number of features for learning models can result in overfitting, which may negatively impact the model generalizations when dealing with the unseen data [45]. In addition to these problems, high-dimensional data analytics require high volumes of memory storage and high-performance computing resources [8, 9]. Many studies [40, 6, 34, 5] have shown that dimensionality reduction overcomes the curse of

dimensionality, improves the learning performance, increases computational efficiency, reduces memory usage, and builds more reliable generalization models.

Dimensionality reduction is mainly performed in two ways: (1) feature extraction; (2) feature selection. In contrast to feature extraction techniques, like those based on projection (e.g., principal component analysis and neural networks), feature selection techniques do not modify the original features but merely select a subset of them [25]. Hence, feature selection approaches preserve the original semantics of the features and provide the advantage of interpretability by domain experts [29]. Feature selection is often preferred in many applications such as microarray data analysis [22, 24, 52], fraud detection [39, 46], and text mining [48]. Also, feature selection can be applied to both supervised and unsupervised learning methods. Supervised feature selection approaches only use labeled data for feature selection and rank feature importance values by calculating the correlation of feature with the class label [44]. Unsupervised feature selection approaches evaluate feature importance values by measuring the particular properties of the data, such as the variance or the locality preserving ability [43]. With adequate labeled data that are expensive to obtain, supervised feature selection approaches usually outperform the unsupervised feature selection methods due to the utilization of labeled information. Throughout this thesis, we focus on the problem of supervised learning or classification as the class labels are identified beforehand in our investigation.

1.3 Feature Selection Taxonomy

Various methods have been proposed for the appropriate selection of feature subsets for classification. FS mainly involves a searching method, with the size of the search space exponentially growing as the number of original features increases [50]. Applying a brute-force approach and exhaustive search for the most informative features is impractical in most situations, particularly when the number of features is more than tens of thousands. High dimensional data consists of irrelevant, misleading, or redundant features that significantly increase the search space size, resulting in inefficient data processing, hence no positive contribution to the learning process.

Feature selection methods are generally categorized into filter, wrapper, and embedded methods [27]. Filter methods utilize the intrinsic properties of features to measure the relevance of features. The selection of features is independent of any machine learning algorithms, and filter methods are ordinarily utilized as a preprocessing step. Instead, features are ranked based on their scores in various statistical tests to correlate with the class label. ReliefF [41] and mRMR [37] are popular examples of filter methods. In contrast, wrapper methods measure the weight of features according to the performance of a classifier. Since the relationship between the features and the class labels is examined through a trained classifier, wrapper methods usually yield better accuracy [21]. However, wrapper methods have high computational costs because of the ample feature space. Each selected subset must be evaluated with a classifier that eventually makes the process slow. Some common

examples of wrapper methods are forward feature selection [31], backward feature elimination [30] , recursive feature elimination [19]. Embedded approaches use the combination of an independent test and performance evaluation using a classifier for a subset of features. They perform feature selection during the modeling process and have lower costs compared to wrapper methods. Some of the most popular examples of embedded methods are LASSO [18] and RIDGE regression [36] which have inbuilt penalization functions to reduce overfitting. Filter methods are fast and have low computational costs; therefore, they are better suited for high dimensional data [51, 16].

1.4 Contribution

This thesis started by removing irrelevant features using SLS and then clustering the remaining features using perturbation theory. In the next phase, we proposed the SVFS model to promote both irrelevant feature removal and clustering correlated features. The outcomes of this study were published in two papers [1, 2], which have been presented in Chapters 3 and 4, respectively. Moreover, the content of Chapter 2 was submitted as a journal paper in March 2020. The structure of this dissertation is based on the format of thesis-by-article, where a series of papers, including published papers or papers submitted or accepted for publication, describes a coherent research study. This format also contains a short introductory chapter, explanation of the research question, relevant literature and methodology, and a concluding chapter.

The SLS paper that is incorporated in Chapter 2 has the following contributions:

- A sparse method (SLS) based on least squares to reduce dimensionality is proposed.
- Irrelevant features can be detected and removed by SLS.
- A soft threshold can be furthered tuned to reduce the given dataset properly.
- SLS can be augmented to any feature selection algorithm.
- SLS optimizes the performance of feature selection algorithms.

The contributions for the DRPT paper that is appeared in Chapter 3 are:

- Features Correlations are encoded in $\Delta\mathbf{x} = |\mathbf{x} - \tilde{\mathbf{x}}|$.
- Features with low $|x_i|$ should be filtered and features with corresponding high $|x_i|$ should be considered as more informative features.
- Clustering using $\Delta\mathbf{x}$ and entropy of features help to select important features.
- Prove DRPT is robust against noise.
- Prove the performance of DRPT is insensitive to permutation of rows or columns of the data.

The contributions for the FVFS paper that is emerged in Chapter 4 are:

- SVFS detects irrelevant features.

- Prove the correlations of features encoded in signature matrix S.
- SVFS can be used for clustering.
- Two thresholds in the process of filtering irrelevant and redundant features are introduced.
- SVFS can be applied in a wide range of high dimensional datasets by tuning the thresholds.
- SVFS can turn into an unsupervised feature selection model.

1.5 Methodology

Irrelevant feature elimination accommodates a more solid understanding of data, reduces computation costs, and enhances classification performance. To exclude irrelevant features, a filter-based feature selection method needs at least a metric to measure the weight of each feature. Several techniques have been developed to tackle the problem of filtering irrelevant and redundant features from high-dimensional datasets [11]. Fisher score [20] and ReliefF [26] methods are two well-known examples of this category. We take advantage of the least-square solutions method and assign a score to each feature, and the features with low scores are filtered using a soft threshold. We show that our method can optimize the performance of feature selection algorithms in terms of both running time and classification accuracy. More details and results are presented in Chapter 2.

During the last decade, the advent of high-dimensional microarray datasets led to a new line of data mining research in bioinformatics. Microarray data analysis is considered a great challenge for computational techniques because of the dimensionality of these data which are typically composed of tens of thousands of genes, despite the small number of samples. We provide a new feature selection method (DRPT) [1] that involves removing the irrelevant features using the least square method and then detecting correlations between the remaining features. Using the perturbation theory, we cluster the reduced data in a nested clustering process, where a feature is selected from each sub-cluster based on its weight and entropy. The effectiveness of DRPT has been verified by performing a series of comparisons with seven state-of-the-art feature selection methods over ten genetic datasets ranging up from 9,117 to 267,604 features. The results show that the overall performance of DRPT is favorable in several aspects compared to each feature selection algorithm. Furthermore, additional experimental complications like the noise in microarray data have been considered to be handled by the DRPT method, and we also prove that this method is robust against noise. Mathematical concepts and details are presented in Chapter 3.

We have further extended the scope of our research by designing and developing a new feature selection method based on singular vectors [2] in which the most informative features are selected in a two-step process. Let $D = [A \mid \mathbf{b}]$ be a labeled dataset, where $\mathbf{b}$ is the class label and the features (attributes) are the columns of matrix A. We show that the signature matrix $S_A = I - A^{\dagger}A$ can be used to determine

correlations between the columns of A, where $A^{\dagger}$ is the pseudo-inverse of A. To do this, we represent the matrix S_A by a graph with the vertices being the columns of A, and the columns $\mathbf{F}_i$ and $\mathbf{F}_j$ being connected if $S_{i,j} \neq 0$. We show that the connected components of this graph are the clusters of columns of A so that the columns in a cluster correlate only with the columns in the same cluster. In the first step, SVFS uses S_D to find the cluster that contains $\mathbf{b}$. We reduce the size of A by discarding features in the other clusters as irrelevant features. In the next step, SVFS uses S_A to partition the remaining features into clusters and choose the most important features from each cluster. Even though SVFS works perfectly on synthetic datasets, comprehensive experiments on real-world benchmark and genomic datasets show that SVFS exhibits overall superior performance compared to the state-of-the-art feature selection methods in terms of accuracy, running time, and memory usage. Mathematical concepts and details through some examples, as well as comprehensive experimental results, are presented in Chapter 4.

Bibliography

[1] Afshar, M., and Usefi, H. High-dimensional feature selection for genomic datasets. *Knowledge-Based Systems 206* (2020), 106370.

[2] Afshar, M., and Usefi, H. Dimensionality reduction using singular vectors. *Scientific Reports-Nature 11*, 1 (2021), 1–13.

[3] Almugren, N., and Alshamlan, H. A survey on hybrid feature selection methods in microarray gene expression data for cancer classification. *IEEE Access 7* (2019), 78533–78548.

[4] Alqudah, A. M., Sallam, A., Baenziger, P. S., and Börner, A. Gwas: Fast-forwarding gene identification and characterization in temperate cereals: lessons from barley–a review. *Journal of advanced research 22* (2020), 119–135.

[5] Ayesha, S., Hanif, M. K., and Talib, R. Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion 59* (2020), 44–58.

[6] Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I. W., Ng, L. G., Ginhoux, F., and Newell, E. W. Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology 37*, 1 (2019), 38–44.

[7] Bello-Orgaz, G., Jung, J. J., and Camacho, D. Social big data: Recent achievements and new challenges. *Information Fusion 28* (2016), 45–59.

[8] Böhm, C. A cost model for query processing in high dimensional data spaces. *ACM Transactions on Database Systems (TODS) 25*, 2 (2000), 129–178.

[9] Bühlmann, P., and Van De Geer, S. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

[10] Chen, S., Montgomery, J., and Bolufé-Röhler, A. Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution. *Applied Intelligence 42*, 3 (2015), 514–526.

[11] Cherrington, M., Thabtah, F., Lu, J., and Xu, Q. Feature selection: filter methods performance challenges. In *2019 International Conference on Computer and Information Sciences (ICCIS)* (2019), IEEE, pp. 1–4.

[12] Chiong, R. *Nature-inspired algorithms for optimisation*, vol. 193. Springer, 2009.

[13] Clerc, M. *Particle swarm optimization*, vol. 93. John Wiley & Sons, 2010.

[14] Dimitrov, D. V. Medical internet of things and big data in healthcare. *Healthcare informatics research 22*, 3 (2016), 156.

[15] Dorigo, M., Birattari, M., and Stutzle, T. Ant colony optimization. *IEEE computational intelligence magazine 1*, 4 (2006), 28–39.

[16] Ferreira, A. J., and Figueiredo, M. A. Efficient feature selection filters for high-dimensional data. *Pattern recognition letters 33*, 13 (2012), 1794–1804.

[17] Fister Jr, I., Yang, X.-S., Fister, I., Brest, J., and Fister, D. A brief review of nature-inspired algorithms for optimization. *arXiv preprint arXiv:1307.4186* (2013).

[18] Fonti, V., and Belitser, E. Feature selection using lasso. *VU Amsterdam Research Paper in Business Analytics 30* (2017), 1–25.

[19] Granitto, P. M., Furlanello, C., Biasioli, F., and Gasperi, F. Recursive feature elimination with random forest for ptr-ms analysis of agroindustrial products. *Chemometrics and intelligent laboratory systems 83*, 2 (2006), 83–90.

[20] Gu, Q., Li, Z., and Han, J. Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725* (2012).

[21] Gutlein, M., Frank, E., Hall, M., and Karwath, A. Large-scale attribute selection using wrappers. In *2009 IEEE symposium on computational intelligence and data mining* (2009), IEEE, pp. 332–339.

[22] Jirapech-Umpai, T., and Aitken, S. Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes. *BMC bioinformatics 6*, 1 (2005), 1–11.

[23] Jones, N. C., Pevzner, P. A., and Pevzner, P. *An introduction to bioinformatics algorithms*. MIT press, 2004.

[24] Kang, C., Huo, Y., Xin, L., Tian, B., and Yu, B. Feature selection and tumor classification for microarray data using relaxed lasso and generalized multi-class support vector machine. *Journal of theoretical biology 463* (2019), 77–91.

[25] Khalid, S., Khalil, T., and Nasreen, S. A survey of feature selection and feature extraction techniques in machine learning. In *2014 science and information conference* (2014), IEEE, pp. 372–378.

[26] Kononenko, I. Estimating attributes: Analysis and extensions of relief. In *European conference on machine learning* (1994), Springer, pp. 171–182.

[27] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR) 50*, 6 (2017), 1–45.

[28] Li, J., and Liu, H. Challenges of feature selection for big data analytics. *IEEE Intelligent Systems 32*, 2 (2017), 9–15.

[29] Liu, H., Dougherty, E. R., Dy, J. G., Torkkola, K., Tuv, E., Peng, H., Ding, C., Long, F., Berens, M., Parsons, L., et al. Evolving feature selection. *IEEE Intelligent systems 20*, 6 (2005), 64–76.

[30] Maldonado, S., and Weber, R. A wrapper method for feature selection using support vector machines. *Information Sciences 179*, 13 (2009), 2208–2217.

[31] Mao, K. Fast orthogonal forward selection algorithm for feature subset selection. *IEEE Transactions on Neural Networks 13*, 5 (2002), 1218–1224.

[32] Merelli, I., Pérez-Sánchez, H., Gesing, S., and D'Agostino, D. Managing, analysing, and integrating big data in medical bioinformatics: open problems and future perspectives. *BioMed research international 2014* (2014).

[33] Mirjalili, S. Genetic algorithm. In *Evolutionary algorithms and neural networks*. Springer, 2019, pp. 43–55.

[34] Nguyen, L. H., and Holmes, S. Ten quick tips for effective dimensionality reduction. *PLoS computational biology 15*, 6 (2019), e1006907.

[35] Patel, S. J., Sanjana, N. E., Kishton, R. J., Eidizadeh, A., Vodnala, S. K., Cam, M., Gartner, J. J., Jia, L., Steinberg, S. M., Yamamoto, T. N., et al. Identification of essential genes for cancer immunotherapy. *Nature 548*, 7669 (2017), 537–542.

[36] Paul, S., and Drineas, P. Feature selection for ridge regression with provable guarantees. *Neural computation 28*, 4 (2016), 716–742.

[37] Peng, H., Long, F., and Ding, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence 27*, 8 (2005), 1226–1238.

[38] Radovic, M., Ghalwash, M., Filipovic, N., and Obradovic, Z. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC bioinformatics 18*, 1 (2017), 1–14.

[39] Ravisankar, P., Ravi, V., Rao, G. R., and Bose, I. Detection of financial statement fraud and feature selection using data mining techniques. *Decision support systems 50*, 2 (2011), 491–500.

[40] Reddy, G. T., Reddy, M. P. K., Lakshmanna, K., Kaluri, R., Rajput, D. S., Srivastava, G., and Baker, T. Analysis of dimensionality reduction techniques on big data. *IEEE Access 8* (2020), 54776–54788.

[41] Robnik-Šikonja, M., and Kononenko, I. Theoretical and empirical analysis of relieff and rrelieff. *Machine learning 53*, 1 (2003), 23–69.

[42] Sayed, S., Nassef, M., Badr, A., and Farag, I. A nested genetic algorithm for feature selection in high-dimensional cancer microarray datasets. *Expert Systems with Applications 121* (2019), 233–243.

[43] Solorio-Fernández, S., Carrasco-Ochoa, J. A., and Martínez-Trinidad, J. F. A review of unsupervised feature selection methods. *Artificial Intelligence Review 53*, 2 (2020), 907–948.

[44] SpolaôR, N., Cherman, E. A., Monard, M. C., and Lee, H. D. A comparison of multi-label feature selection methods using the problem transformation approach. *Electronic Notes in Theoretical Computer Science 292* (2013), 135–151.

[45] Subramanian, J., and Simon, R. Overfitting in prediction models–is it a problem only in high dimensions? *Contemporary clinical trials 36*, 2 (2013), 636–641.

[46] Sun, J., Li, Y., Chen, C., Lee, J., Liu, X., Zhang, Z., Huang, L., Shi, L., and Xu, W. Fdhelper: Assist unsupervised fraud detection experts with interactive feature selection and evaluation. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–12.

[47] Verleysen, M., and François, D. The curse of dimensionality in data mining and time series prediction. In *International work-conference on artificial neural networks* (2005), Springer, pp. 758–770.

[48] Vora, S., and Yang, H. A comprehensive study of eleven feature selection algorithms and their impact on text classification. In *2017 Computing Conference* (2017), IEEE, pp. 440–449.

[49] Wang, T., Birsoy, K., Hughes, N. W., Krupczak, K. M., Post, Y., Wei, J. J., Lander, E. S., and Sabatini, D. M. Identification and characterization of essential genes in the human genome. *Science 350*, 6264 (2015), 1096–1101.

[50] Xue, B., Zhang, M., Browne, W. N., and Yao, X. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation 20*, 4 (2015), 606–626.

[51] Yu, L., and Liu, H. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning (ICML-03)* (2003), pp. 856–863.

[52] Zhang, G., Hou, J., Wang, J., Yan, C., and Luo, J. Feature selection for microarray data classification using hybrid information gain and a modified binary krill herd algorithm. *Interdisciplinary Sciences: Computational Life Sciences 12* (2020), 288–301.

Chapter 2

Optimizing Feature Selection Methods by Removing Irrelevant Features Using Sparse Least Squares

(This chapter is submitted as a paper to journal of Expert Systems with Applications, March 2020)

2.1 Introduction

Gene expression datasets usually consist of tens or hundreds of samples compared to thousands or tens of thousands of features. This property of gene datasets impacts the performance of the classifier [22] and also can cause data overfitting [10]. The purpose of Feature Selection (FS) is to find a subset of features that are more informative and

relevant to class labels [15]. So, in addition to improving the performance of the classifier, feature selection avoids over-fitting [5, 12].

In genome-wide association study (GWAS), partial or all of the human genome is genotyped for discovering the associations between genetic factors and a disease or a phenotypic trait. In GWAS, the genetic variants under consideration are single nucleotide polymorphisms (SNPs), the most common type of variation among people. The number of SNPs in a disease dataset varies from tens of thousands to more than a million. As such, one of the bottlenecks of working with these genome datasets is their large-scale size that makes it difficult to render the data for meaningful analysis. In these datasets, there are many embedded noisy SNPs; these are SNPs that have minimal effect on the disease.

One of the customary methods to determine whether an SNP is associated with a disease or not is using p-values and statistical significance. For example, in [25] a two-step feature selection strategy was used on a dataset containing 17,000 Crohn's disease cases, 13,000 Ulcerative Colitis cases, and 22,000 controls with 178,822 SNPs. In that study, Wei et al. reduced the number of features by filtering out SNPs with p-values greater than 10^{-4} and then applied a penalized feature selection with L_1 penalty to select a subset of SNPs. However, researchers are strongly advised against the use of p-values and statistical significance in relation to the null-hypothesis [1, 24].

In this chapter, we propose a sparse method to remove irrelevant features. We take advantage of the least-square solutions method and assign a score to each feature, and

the features with low scores are filtered using a soft threshold. We show features whose weights are very small do not affect the class label and consider them as irrelevant features. In other words, we propose a sparse method called Sparse Least Squares (SLS), in which we shrink the wights of irrelevant features to zero to reduce the size of a dataset. Any feature selection algorithm can be augmented with the SLS . Of particular interest to us are feature selection algorithms that have great prediction power, however, suffer from high computational cost. Among these algorithms are the wrapper methods such as SVMRFE [10] and methods based on information gain such as mRMR [20].

To show the effectiveness of our approach, we experiment with six genomic datasets. After reducing the size of datasets by our SLS method, we apply a feature selection algorithm to the reduced dataset. We examine three well-known feature selection algorithms: mRMR [20], SVMRFE [10], and ReliefF [13] ; also more description and experimental results are presented in Sections 2.2 and 2.4 respectively. Augmentation of FS methods with SLS to classify gene expression datasets, shows significant improvement to accuracy. Meanwhile, the running times are considerably reduced because apply SLS results in a much smaller dataset. We shall also experiment on image and text datasets and see that augmenting FS algorithms with our SLS, reduces the computational cost by orders of magnitude while maintaining or improving the prediction accuracy of the models.

The remaining of this chapter is organized as follows. In Section 2.2, we review

related work. We explain our methodology in Section 2.3 and report experimental results in Section 2.4. Finally, we summarize our work and conclude the chapter in Section 2.5.

2.2 Related work

Feature selection algorithms fall into three different categories: filter, wrappers, and embedded methods. Filter methods are independent of classifiers and select a subset before any classification. Relief-based methods [16] such as Minimum Redundancy Maximum Relevance (mRMR) [20] and Relief [14], are well-known filter feature selection methods. ReliefF model [13] is another widely used filter-based approach wherein features are scored using feature value differences between nearest-neighbor instance pairs. Wrapper methods [15] select a subset and estimate the score of the subset by employing the performance of the classifier. Wrapper methods have been proven to be useful but have a high computational complexity since the induction algorithm is called repeatedly. A well-known wrapper method is Support Vector Machines Recursive Feature Elimination (SVMRFE) [10] algorithm. This algorithm repeatedly constructs the model and eliminate features with low ranks. Filter methods are faster than wrappers and computationally suitable to be applied to large datasets. In embedded methods, feature selection is strongly coupled with the classifier design. In terms of computational complexity, this approach falls between filter and wrappers methods.

Irrelevant feature removal enables a more solid understanding of data, reduces computation costs, and improves classification performance. To identify and remove the irrelevant features, one needs a criteria to measure the relevancy of features to the output class. Two main ranking methods that help understand the relevance of a feature are Correlation metric and Mutual Information. The most practical and simple metric is the Pearson Correlation Coefficient [9, 19] which is defined as:

$$R(i) = \frac{cov(\mathbf{F}_i, \mathbf{b})}{\sqrt{var(\mathbf{F}_i) \times var(\mathbf{b})}}$$

where $\mathbf{F}_i$ is the i-th feature, $\mathbf{b}$ is the class label, cov is the covariance and var is the variance.

Ranking methods based on Information-theory [9, 21, 11] calculate the dependency between features. Mutual Information (MI) between features X and Y is zero if X and Y are independent and MI greater than zero reflects they are dependent. MI is defined as follows:

$$I(Y, X) = H(Y) - H(Y|X).$$

Here $H(Y)$ represents Shannon's definition for entropy given by:

$$H(Y) = -\sum_{y} p(y) \log(p(y))$$

where $p(y)$ is the probability of occurrence of y. Also, $H(Y|X)$ is the conditional entropy given as :

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

In [6], the author proposed a feature ranking based on conditional MI in the application of image classification and molecular bio-activity. This metric is used to score features using the following equation:

$$s[n] = \min_{\ell \leq k} \widehat{I}(Y; X_n | X_{v(\ell)})$$

where $s[n]$ is the score updated at each iteration, k contains the index of the last picked feature, X_n is the current evaluated feature, $X_{v(l)}$ is the set of already selected features.

2.3 Proposed Approach

Consider a dataset $D = [A \mid \mathbf{b}]$, consisting of m samples where each sample has $n+1$ features. The class label of D is denoted by $\mathbf{b}$. We consider the linear system $A\mathbf{x} = \mathbf{b}$, where $\mathbf{x} = [x_1, \ldots, x_n]^T$ is the vector of unknowns. Since the system $A\mathbf{x} = \mathbf{b}$ may not have exact solutions, instead we find the unique solution with the smallest 2-norm that satisfy the least squares problem

$$||A\mathbf{x} - \mathbf{b}||_2, \tag{2.1}$$

over all $\mathbf{x}$. This minimization problem is known as the method of least squares and its solutions are defined via singular value decomposition (SVD) of A. Recall that the SVD of an $m \times n$ matrix A is of the form $A = USV^T$, where U is an $m \times m$ orthogonal matrix, V is an $n \times n$ orthogonal matrix, and $S = \text{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$ is an $m \times n$ diagonal matrix. Also, recall that the Moore-Penrose inverse of A is the

$n \times m$ matrix $A^\dagger = VS^{-1}U^T$, where $S^{-1} = \text{diag}(\sigma_1^{-1}, \ldots, \sigma_r^{-1}, 0, \ldots, 0)$. It is well-known that $\mathbf{x} = A^\dagger \mathbf{b}$ is the least squares with the smallest 2-norm, see [7].

We view the solution $\mathbf{x} = \begin{bmatrix} x_1, \ldots, x_n \end{bmatrix}^t$ to the least squares problem as a weight vector. In other words, we can approximate the label column $\mathbf{b}$ as a linear combination $x_1\mathbf{F}_1 + \cdots + x_i\mathbf{F}_i + \cdots + x_n\mathbf{F}_n$, where $\mathbf{F}_i$ is the i-th column of A. Intuitively, the larger $|x_i|$ the more impact $\mathbf{F}_i$ has on $\mathbf{b}$. As such, we filter out those features whose corresponding weight is less than a threshold as irrelevant features. In other words, we shrink the wights of irrelevant features to zero. This process yields a sparse method to reduce the size of datasets.

Our aim is to show that identifying irrelevant features using their weights actually makes sense. We prove this for matrices that are full-row rank. So, for the rest of this section, we assume that $\text{rank}(A) = m$. Let us denote by A_j the matrix obtained from A by adding a (random) column vector $\mathbf{c} \in \mathbb{R}^m$ to $\mathbf{F}_j$. We realize that this kind of perturbation of A can be expressed in terms of a rank-1 update of A. Consider the column vector $\mathbf{e}_j \in \mathbb{R}^n$ as the j-th standard basis vector. It is easy to verify that $A_j = A + \mathbf{c}\mathbf{e}_j^t$. To solve $A_j\mathbf{x} = \mathbf{b}$, we need to find the pseudo-inverse of $A + \mathbf{c}\mathbf{e}_j^t$.

Let $A \in \mathbb{R}^{m \times n}$ be a matrix of full row-rank, $\mathbf{c} \in \mathbb{R}^m$ and $\mathbf{d} \in \mathbb{R}^n$. Then it is known (see for example [4, Theorem 3.3.2]) that

$$(A + \mathbf{c}\mathbf{d}^t)^\dagger = A^\dagger + \frac{1}{\beta}\mathbf{s}^t\mathbf{k}^t A^\dagger - \frac{\beta}{\alpha}(\frac{\| \mathbf{k} \|^2}{\beta}\mathbf{s}^t + \mathbf{k})(\frac{\| \mathbf{s} \|^2}{\beta}\mathbf{k}^t A^\dagger + \mathbf{h}),$$

where $\mathbf{k} = A^\dagger\mathbf{c}, \mathbf{h} = \mathbf{d}^t A^\dagger, \mathbf{s} = \mathbf{d}^t(I - A^\dagger A)$, $\beta = 1 + \mathbf{d}^t A^\dagger \mathbf{c}$, and $\alpha = \| \mathbf{k} \|^2 \| \mathbf{s} \|^2 + | \beta |^2$.

We denote column j of V by $\mathbf{v}_j$ and row j of V by $\mathbf{v}^j$. We partition $\mathbf{v}^j$ as $\mathbf{v}^j = \left[\begin{array}{c|c} \mathbf{v}^{j,1} & \mathbf{v}^{j,2} \end{array}\right]$, where $\mathbf{v}^{j,1}$ consists of the first m entries of $\mathbf{v}^j$ and $\mathbf{v}^{j,2}$ is the remaining $n-m$ entries.

Lemma 2.3.1. *The column $\mathbf{F}_j$ of A is independent of the rest of columns of A if and only if $\mathbf{v}^{j,2} = 0$.*

Proof. Note that $A\mathbf{v}_i = 0$, for all $m+1 \leq i \leq n$. Let k be in the range $m+1 \leq k \leq n$. Note that $A\mathbf{v}_k = 0$ yields a dependence relation between the columns of A. So if $\mathbf{F}_j$ is independent of the rest of columns of A, we deduce that the entry in the j-th position of $\mathbf{v}_k$ must be zero, that is $v_{j,k} = 0$. So the j-th row of V is of the form $\mathbf{v}^j = [v_{j,1} \cdots v_{j,m}\, 0 \cdots 0]$. Hence, $\mathbf{v}^{j,2} = 0$. Conversely, a dependence relation between $\mathbf{F}_j$ and the other columns, yields a vector $\mathbf{z}$ whose j-th position is non-zero and $A\mathbf{z} = 0$. So, $\mathbf{z}$ is in the $\ker(A)$ which is kernel of matrix A and can be expressed in terms of $\mathbf{v}_{m+1}, \ldots, \mathbf{v}_n$. So, the j-th component of at least one of the $\mathbf{v}_{m+1}, \ldots, \mathbf{v}_n$ must be non-zero. Hence, $\mathbf{v}^{j,2} \neq 0$. □

Lemma 2.3.2. *Suppose that column $\mathbf{F}_j$ of A is independent of the rest of columns of A. Then $\mathbf{s} = \mathbf{e}_j^t(I - A^\dagger A) = 0$.*

Proof. Note that, by Lemma 2.3.1, $\mathbf{v}^{j,2} = 0$. So, we have

$$\mathbf{v}^j \left[\begin{array}{c|c} I_m & 0 \\ \hline 0 & 0 \end{array}\right] = \mathbf{v}^j.$$

We have

$$\mathbf{e}_j^t A^\dagger A = \mathbf{e}_j^t V S^{-1} U^t U S V^t = \mathbf{e}_j^t V S^{-1} S V^t$$
$$= \mathbf{v}^j \left[\begin{array}{c|c} I_m & 0 \\ \hline 0 & 0 \end{array}\right] V^t = \mathbf{v}^j V^t = \mathbf{e}_j^t I.$$

Hence, $\mathbf{s} = \mathbf{e}_j^t (I - A^\dagger A) = 0$. □

Theorem 2.3.3. *Suppose that column $\mathbf{F}_j$ of A is independent of the rest of columns of A. Let $\mathbf{x} = A^\dagger \mathbf{b}$ and $\tilde{\mathbf{x}} = (A + \mathbf{c}\mathbf{e}_j^t)^\dagger \mathbf{b}$. Then $||\mathbf{x} - \tilde{\mathbf{x}}|| \leq \frac{|x_j|}{\sigma_m}$.*

Proof. Note that, by Lemma 2.3.2, we have $\mathbf{s} = 0$. Also, we have

$$\mathbf{khb} = A^\dagger \mathbf{c}\mathbf{e}_j^t A^\dagger \mathbf{b}$$
$$= A^\dagger \mathbf{c}\mathbf{e}_j^t \mathbf{x} = A^\dagger \mathbf{c} x_j$$

Hence, by [4, Theorem 3.3.2], we get

$$||\mathbf{x} - \tilde{\mathbf{x}}|| = ||A^\dagger \mathbf{b} - (A + \mathbf{c}\mathbf{e}_j^t)^\dagger \mathbf{b}||$$
$$= ||\mathbf{khb}|| = ||A^\dagger \mathbf{c} x_j|| \leq ||A^\dagger||\,|x_j| = \frac{|x_j|}{\sigma_m}$$

□

In Theorem 2.3.3, we can choose $\mathbf{c}$ so that $\Delta\mathbf{x} = \frac{|x_j|}{\sigma_m}$; this way $\Delta\mathbf{x}$ is a vector that directly correlates with x_j. To do so, we just need to choose $\mathbf{c}$ so that $||A^\dagger \mathbf{c}|| = ||A^\dagger||$. So, we take $\mathbf{c}$ that is a solution to the optimization problem:

$$\max_{||\mathbf{x}||=1} ||A^\dagger \mathbf{x}|| \tag{2.2}$$

It is well-known that the solution to optimization problem (2.2) is a unit eigenvector of $(A^\dagger)^t A^\dagger$ corresponding to eigenvalue $1/\sigma_m^2$.

We may think of $\mathbf{b}$ along with the features $\mathbf{F}_1, \cdots, \mathbf{F}_n$ as a many body problem so that $\mathbf{b}$ may interact (related) with some of the $\mathbf{F}_j$s and the features $\mathbf{F}_1, \cdots, \mathbf{F}_n$ might interact with each other due to their correlations with each other. The least squares solution to $A\mathbf{x} = \mathbf{b}$ determines the weight of interactions between $\mathbf{b}$ and the $\mathbf{F}_i$s. It follows from Theorem 2.3.3 that perturbing irrelevant features will not affect the equilibrium state of the whole system.

SLS works especially well when the number of samples is much less than the number of features, that is $m << n$. Of special interest to us are genomic datasets where there are usually tens or hundreds of samples compared to tens of thousands of genes. The matrix A in these datasets has full row-rank because gene expression of different samples are independent of each other. Intuitively, it makes sense to eliminate the columns that are less important. Of course the definition of relevancy is not quantitative and one has to set a threshold for the degree of relevancy. We can tune the threshold parameter, however, our experiments show that even a soft threshold is enough to reduce the computational times of feature selection algorithms and increase the prediction power of classifier on selected features. Next, we apply the feature selection algorithm on the reduced dataset. In other words, we augment the existing feature selection methods with our SLS.

The complexity of our proposed method is dominated by the complexity of the

Algorithm 1: Augmenting SLS to feature selection algorithms

Data: $D = [A \mid \mathbf{b}]_{m\times(n+1)}$

Result: Subset of features, CA

1 $\mathbf{x} = A^{\dagger}\mathbf{b}$, where $A^{\dagger}$ is the Moore-Penrose inverse of A ;

2 Threshold $=0.1 * \max(|\mathbf{x}|)$;

3 Irrelevant $=\{i \mid |x_i| <$ Threshold$\}$;

4 Index $=\{1, \ldots, n]\} \setminus$ Irrelevant;

5 $\hat{D} = [A_{\text{Index}} \mid \mathbf{b}]$;

6 Apply feature selection algorithm to the reduced dataset $\hat{D}$;

7 Classify D based on the selected features and return CA ;

SVD, since the inverse of perturbed $\tilde{A}$ is calculated using SVD. The complexity of computing SVD of $A_{m\times n}$ is $\mathcal{O}(\min(mn^2, m^2n))$.

2.4 Experimental Results

We examine three well-known feature selection algorithms, that are mRMR, SVMRFE and ReliefF as we describe below.

mRMR This method selects features with the highest relevance to the class labels and lowest redundancy among candidate features. Both maximum-relevance and minimum-redundancy criteria on this method are based on mutual information.

SVMRFE This algorithm utilizes an estimator to assign weights to features. These weights generated by estimator are used as ranking criteria. In each step of feature elimination, the lowest-ranked features are removed from the current subset of features. Feature elimination is a recursive procedure and is repeated until the specified number of features is selected.

ReliefF ReliefF assigns a score for each feature based on the identification of feature value differences between nearest-neighbor instance pairs. The main advantage of this algorithm is to measure feature interactions without performing a comprehensive inspection of every pairwise interaction, consequently taking significantly less time than a comprehensive pairwise search.

2.4.1 Datasets and Pre-processing

We select a variety of datasets, including genomic, image, and text datasets, which are considered high-dimensional. Three genomic datasets, described in Table 2.1, are publicly available from NCBI dataset browser [1]. Genomic datasets are not cleaned and for pre-processing the data, we develop an R code to clean and convert any full Simple Omnibus Format in Text (SOFT) dataset in NCBI to CSV format [2]. We use GEO2R [2] to find the mapping between prob IDs and gene samples. Probe IDs without a gene mapping were removed. Next, expression values of each gene

[1] https://www.ncbi.nlm.nih.gov/sites/GDSbrowser

[2] http://github.com/majid1292/NCBIdataPrep

is considered as the average of expression values of all mapped prob IDs to that gene. In the output dataset, we also refine null cells with k-Nearest Neighbors (kNN) imputation method.

Table 2.1: Summary of genomic datasets

Dataset	# **Samples**	# **Original features**	# **Cleaned features**	# **Classes**
GDS3268	202	44,290	29,916	2
GDS1615	127	22,282	13,649	3
GDS3929	183	24,526	19,334	2
GDS2545	171	12,625	9,391	4
GDS531	173	12,625	9,392	2
GDS1962	180	54,675	29,185	4

Table 2.2 shows a summary of selected image and text dataset which are accessible from the open-source feature selection repository at Arizona State University [3].

[3] http://featureselection.asu.edu/datasets.php

Table 2.2: Summary of image and text datasets

Dataset	**# Samples**	**# Features**	**# Classes**	**Type**
pixraw10P	100	10,000	10	Image
orlraws10P	100	10,304	10	Image
warpPIE10P	210	2,420	10	Image
warpAR10P	130	2,400	10	Image
BASEHOCK	1,993	4,862	2	Text
PCMAC	1,943	3,289	2	Text

2.4.2 Hardware and Software

In all experiments, all codes are implemented with Python 3.6 and also we have used the Python software packages of mRMR, SVMRFE and ReliefF available in scikit-rebate machine learning library [23]. Note that the basic ReliefF requires to specify the number of nearest neighbors to consider in the scoring algorithm, and we use MultiSURF version implementation of ReliefF, which is an extension to the original algorithm that automatically ascertain the ideal number of neighbors to consider for scoring the features. In addition, all experiments have been run on an IBM®LSF 10.1.0.6 machine (Suite Edition: IBM Spectrum LSF Suite for HPC 10.2.0) with

requested 8 nodes, 8 GB of RAM , and 4 GB swap memory using Python 3.6.5.

2.4.3 Validation and Evaluation

To evaluate how selected features can help differentiate between samples with different labels, having a test dataset that has not been seen by the machine learning models is essential. Thus, we use 5-fold cross-validation (CV) techniques, and to avoid the dataset shift, which is one of the drawbacks of using cross-validation, the stratified version was used [3]. The stratification ensures both training and testing sets contain the same distribution of class labels as in the original dataset. Stratified CV ensures that no value is over/under-represented in the training and test sets, leading to a more accurate estimate of the classification performance [17].

The number of features k to be selected by a feature selection algorithm should be given as an input parameter because the computational complexity is affected by k. So, we set $k = 50$ for all algorithms.

To evaluate the performance of each feature selection method or its augmentation with SLS, we report classification accuracy (CA) over 15 iterations (three times repeating stratified 5-fold CV). Similar studies [26, 8, 18] suggest considering the top 50 features for feature selection. However, those top selected features may not necessarily yield the highest accuracy. Therefore, in each of the 15 iterations, we set k = 50 to select a subset of 50 features using FS algorithms, and we take advantage of inner iteration from $t = 1$ to k to feed the first t features to the Random Forest

(RF) classier to find an optimal number of top feature where the subset of the first t features yields the highest accuracy. Then, the average CA along with size of the corresponding t optimal subset of features are reported. Also, we report the total running time over 15 iterations which includes the running time of the classifier as well. This setting is applied across all FS methods.

2.4.4 Experiments on text datasets

Fig. 2.1 represents the running time, classification accuracy, and the number of selected features of each FS method on the text datasets. We choose a range of threshold, starting from 0 to 0.23 or 0.24, where threshold 0 indicates the original datasets and increasing the threshold turns the given dataset into a lower dimension.

As we can see from Fig. 2.1, SVMRFE maintains the same CA on 0 threshold (original dataset) and 0.1 threshold while the running time of SVMRFE is reduced by a magnitude of order on the reduced datasets.

On the other hand, mRMR and ReliefF see an increasing trend in CA as we increase the threshold. The running time of mRMR on the original text datasets is around $4 \cdot 10^5$ seconds and around 400 seconds at the last threshold. This performance illustrates how selecting a soft threshold affects upon computation cost and classification accuracy of all FS methods. Moreover, reducing the size of datasets by increasing the threshold does not significantly change the number of selected features and reflects the characteristics of the original datasets are preserved in the reduction

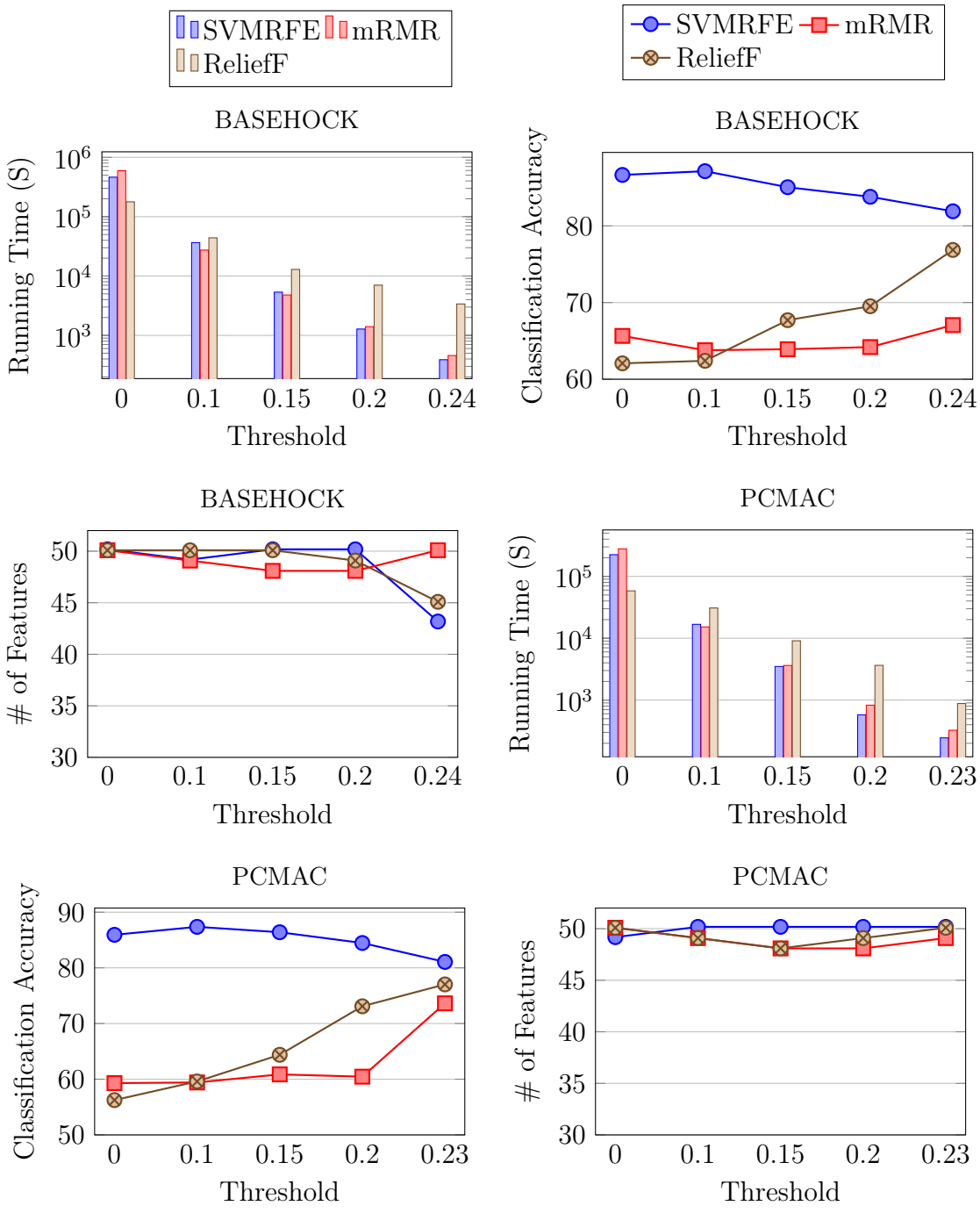
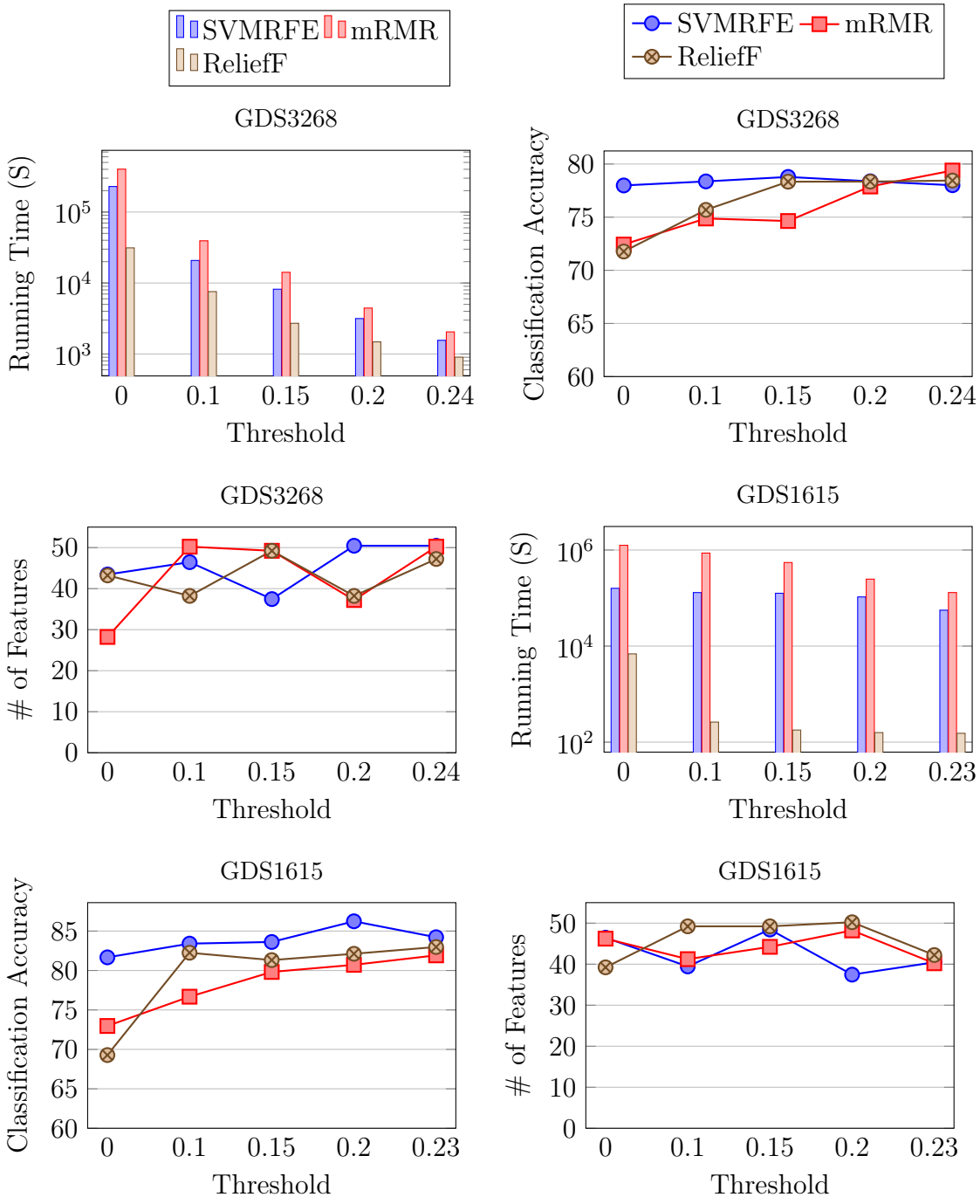


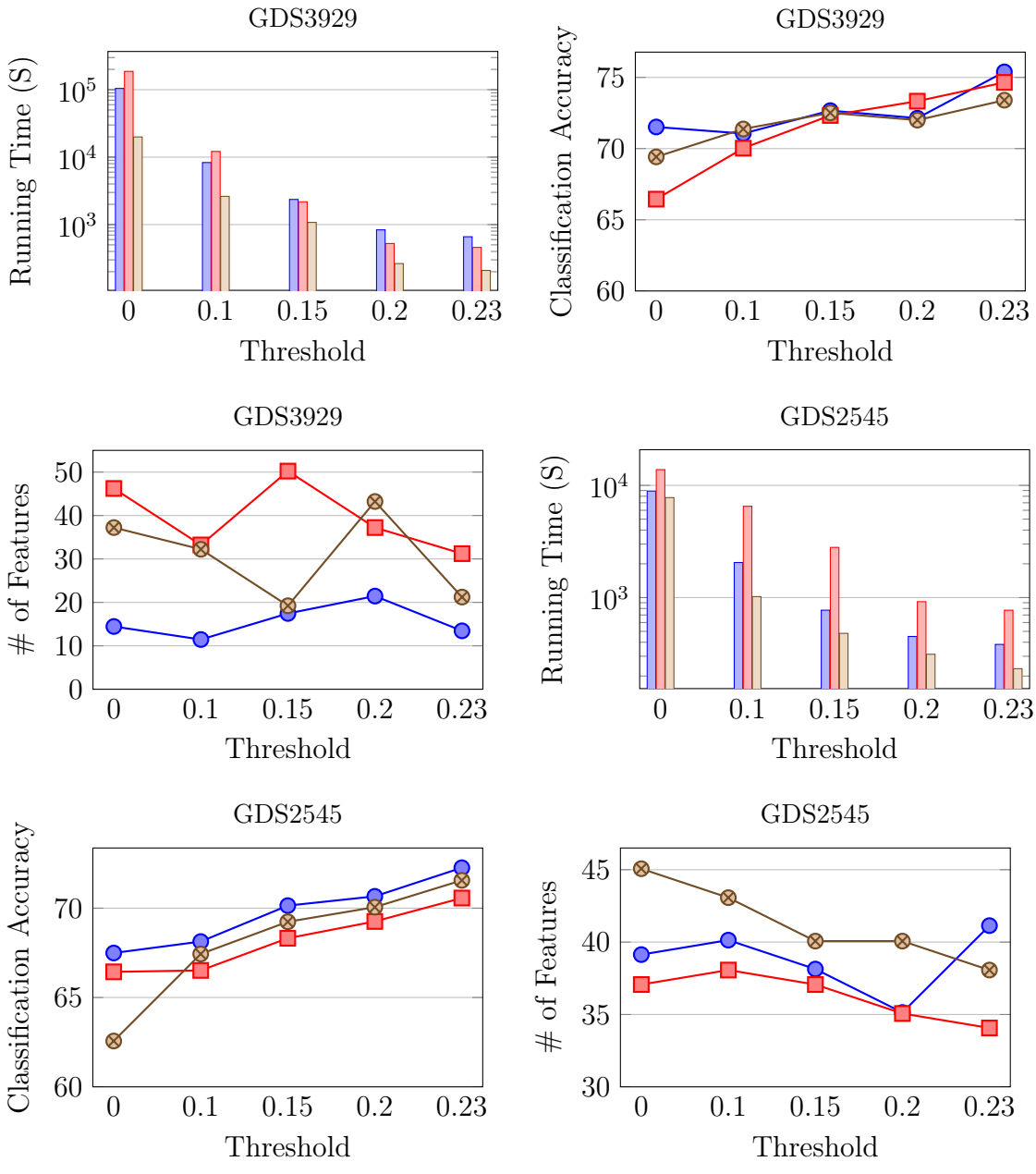
Figure 2.1: Running time and classification accuracy of feature selection by SVMRFE, mRMR and ReliefF, over 15 runs considering 5 different thresholds on text datasets

process.

2.4.5 Experiments on genomic datasets

Fig. 2.2 represents the results of running time in seconds, CA, and the number of selected features for six genomic datasets described in Table 2.1. For all datasets excluding GDS1615, SLS decreases the running time up to 400 times for all the three FS methods where threshold zero refers to the original dataset and increasing threshold makes datasets smaller. For GDS1615, although we get marginal running time reduction particularly for mRMR and SVMRFE, SLS clearly improves the CA of all FS models. Moreover, the superiority of our proposed model is evident by looking at the upward trends in CA across all datasets and FS methods. While we experience improvement in CA over the increasing threshold, the number of selected features decreases or remains relatively the same compare to the number of selected features at threshold 0 (original dataset). Experiments over genomic datasets clearly expose SLS removes irrelevant features by producing less noisy and smaller datasets and feeding FS models more informative data.


SVMRFE
mRMR
ReliefF
GDS3268
Running Time (S)
10^5
10^4
10^3
0
0.1
0.15
0.2
0.24
Threshold
SVMRFE
mRMR
ReliefF
GDS3268
Classification Accuracy
80
75
70
65
60
0
0.1
0.15
0.2
0.24
Threshold
GDS3268
# of Features
50
40
30
20
10
0
0
0.1
0.15
0.2
0.24
Threshold
GDS1615
Running Time (S)
10^6
10^4
10^2
0
0.1
0.15
0.2
0.23
Threshold
GDS1615
Classification Accuracy
85
80
75
70
65
60
0
0.1
0.15
0.2
0.23
Threshold
GDS1615
# of Features
50
40
30
20
10
0
0
0.1
0.15
0.2
0.23
Threshold


GDS3929
Running Time (S)
10^5
10^4
10^3
0
0.1
0.15
0.2
0.23
Threshold
GDS3929
Classification Accuracy
75
70
65
60
0
0.1
0.15
0.2
0.23
Threshold
GDS3929
# of Features
50
40
30
20
10
0
0
0.1
0.15
0.2
0.23
Threshold
GDS2545
Running Time (S)
10^4
10^3
0
0.1
0.15
0.2
0.23
Threshold
GDS2545
Classification Accuracy
70
65
60
0
0.1
0.15
0.2
0.23
Threshold
GDS2545
# of Features
45
40
35
30
0
0.1
0.15
0.2
0.23
Threshold

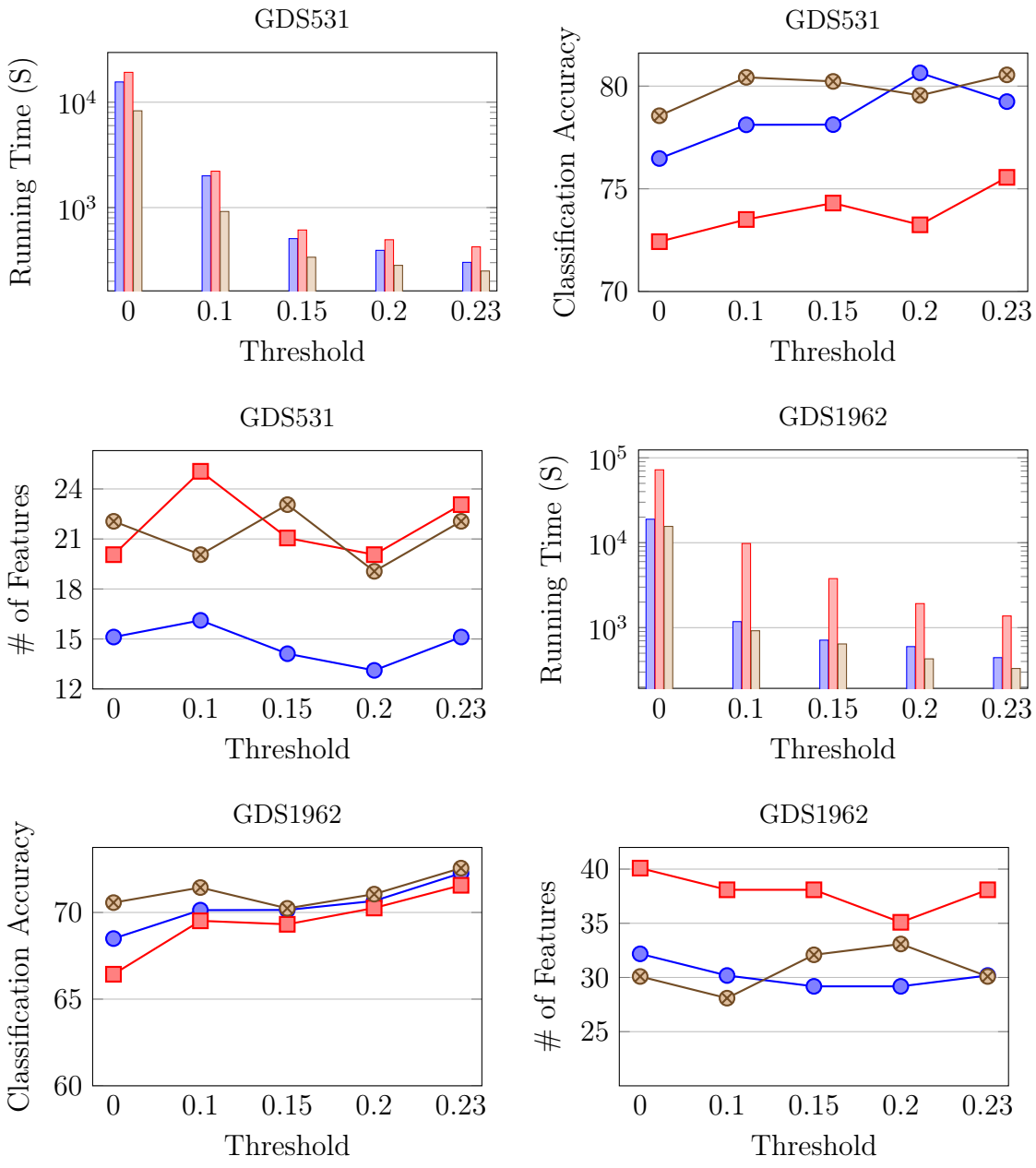
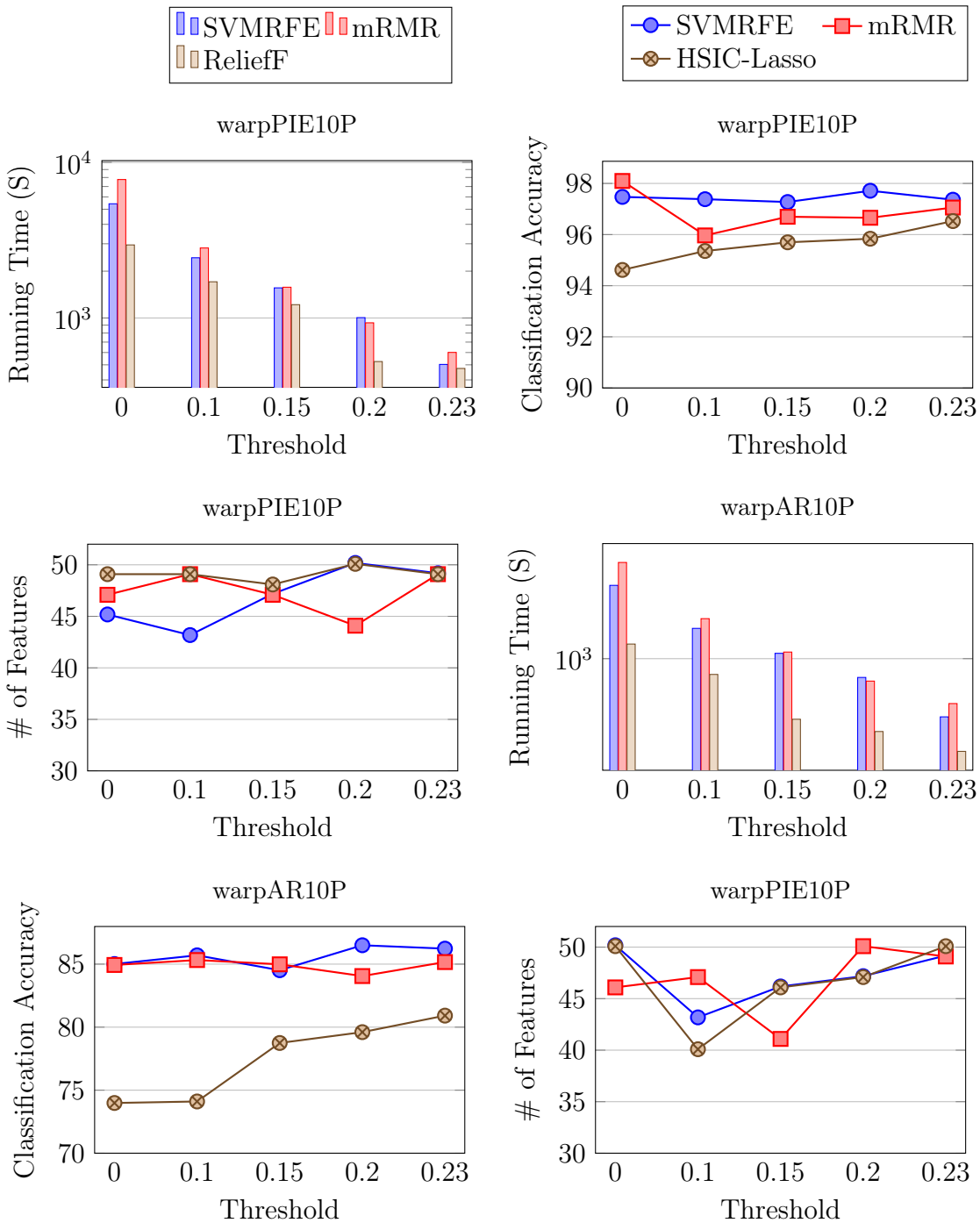


Figure 2.2: Running time and classification accuracy of feature selection by SVM-RFE, mRMR and ReliefF, over 15 runs considering 5 different thresholds on genome datasets

2.4.6 Experiments on image datasets

Fig. 2.3 illustrates a significant decrease in running time for all three FS methods over all image datasets while CA remains almost stable for SVMRF and mRMR models across different thresholds. The ReliefF has gained a moderate increase in CA on some of the datasets.


SVMRFE
mRMR
ReliefF
warpPIE10P
Running Time (S)
10^4
10^3
0
0.1
0.15
0.2
0.23
Threshold
SVMRFE
mRMR
HSIC-Lasso
warpPIE10P
Classification Accuracy
98
96
94
92
90
0
0.1
0.15
0.2
0.23
Threshold
warpPIE10P
# of Features
50
45
40
35
30
0
0.1
0.15
0.2
0.23
Threshold
warpAR10P
Running Time (S)
10^3
0
0.1
0.15
0.2
0.23
Threshold
warpAR10P
Classification Accuracy
85
80
75
70
0
0.1
0.15
0.2
0.23
Threshold
warpPIE10P
# of Features
50
45
40
35
30
0
0.1
0.15
0.2
0.23
Threshold

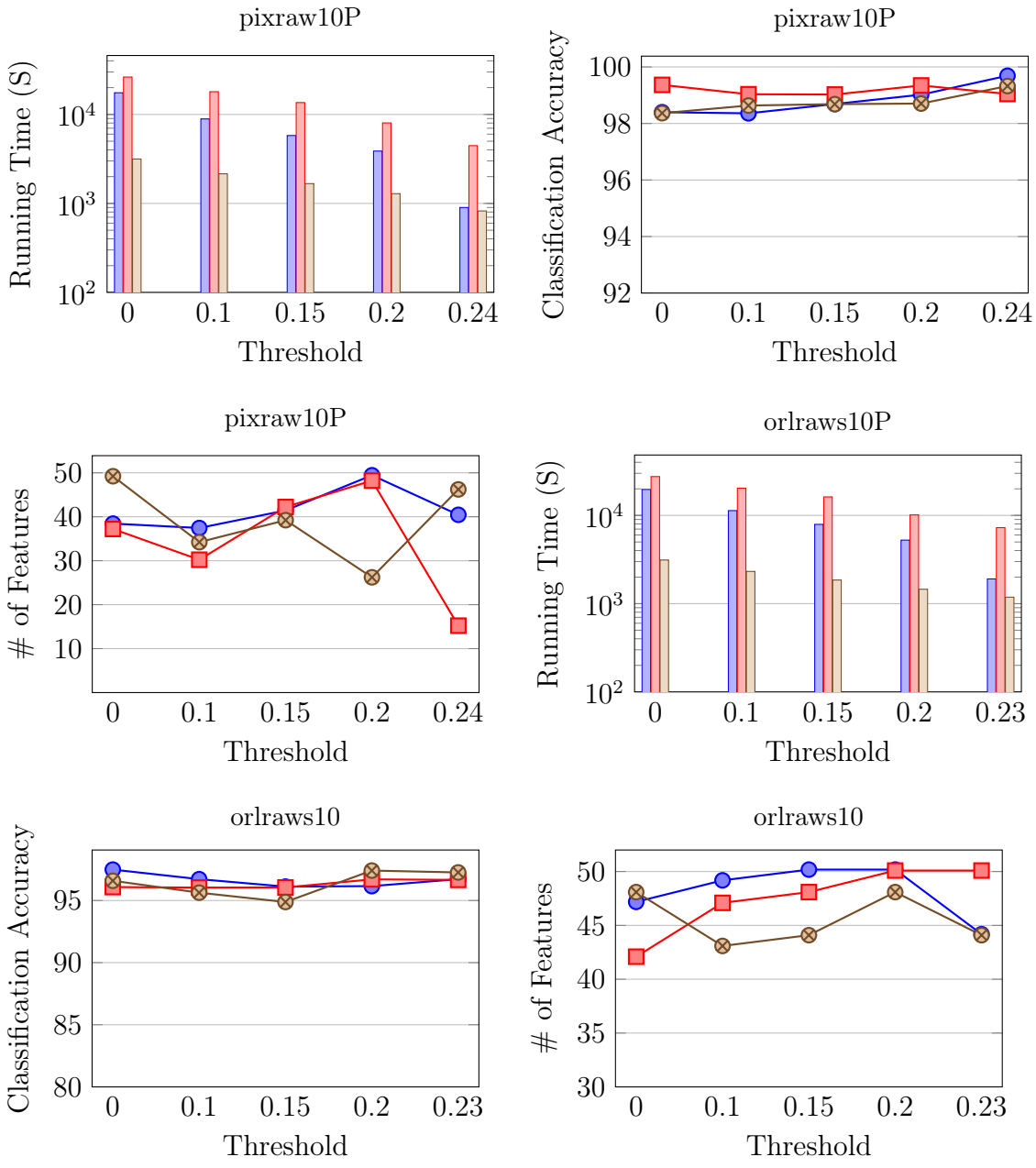


Figure 2.3: Running time and classification accuracy of feature selection by SVMRFE, mRMR and ReliefF, over 15 runs considering 5 different thresholds on image datasets

It is worthwhile to note that the CA on pixraw10P and orlraws10P are basically

above 95% at zero thresholds (original dataset) and this high accuracy is preserved when we filter out some features using SLS. This is a good indication that SLS efficiently removes the irrelevant features and preserves important features.

Overall experiments on genomic, text, and image datasets exhibit that SLS efficiently removes irrelevant features and turns the datasets into a lower dimension while preserving more informative features. Over some thresholds, we did not achieve a significant improvement in AC, but computation cost decreased in the magnitude of order. This comprehensive experiment also reflects the term of relevancy is not quantitative in real and benchmark datasets, and we omit features with low relevance to the class label based on a soft threshold.

2.5 Conclusions

In this chapter, we proposed a method (SLS) based on least squares to remove irrelevant features. We can think of the class label $\mathbf{b}$ along with the features $\mathbf{F}_1, \cdots, \mathbf{F}_n$ as a many body problem so that $\mathbf{b}$ may interact (related) with some of the $\mathbf{F}_j$s and the features $\mathbf{F}_1, \cdots, \mathbf{F}_n$ might interact with each other due to their correlations with each other. We proved that perturbing irrelevant features will not affect the equilibrium state of the whole system. Since in real datasets the notion of relevancy is not quantitative, we have to decide on the relevancy based on a threshold. We showed by experiments that SLS can optimize the performance of feature selection algorithms both in terms of running time and classification accuracy by choosing a soft threshold.

Bibliography

[1] Amrhein, V., Greenland, S., and McShane, B. Scientists rise up against statistical significance, 2019.

[2] Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillippy, K. H., Sherman, P. M., Holko, M., et al. NCBI GEO: archive for functional genomics data sets—update. *Nucleic acids research 41*, D1 (2012), D991–D995.

[3] Braga-Neto, U. M., and Dougherty, E. R. Is cross-validation valid for small-sample microarray classification? *Bioinformatics 20*, 3 (2004), 374–380.

[4] Campbell, S. L., and Meyer, C. D. *Generalized inverses of linear transformations*. SIAM, 2009.

[5] Cawley, G. C., and Talbot, N. L. On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research 11* (2010), 2079–2107.

[6] Fleuret, F. Fast binary feature selection with conditional mutual information. *Journal of Machine learning research 5*, Nov (2004), 1531–1555.

[7] Golub, G. H., and Van Loan, C. F. *Matrix Computations*. Johns Hopkins Univ Pr, 2012.

[8] Gutkin, M., Shamir, R., and Dror, G. Slimpls: a method for feature selection in gene expression-based disease classification. *PloS one 4*, 7 (2009), e6416.

[9] Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *Journal of machine learning research 3*, Mar (2003), 1157–1182.

[10] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine learning 46*, 1-3 (2002), 389–422.

[11] Hancer, E., Xue, B., and Zhang, M. Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-Based Systems 140* (2018), 103–119.

[12] Jabbar, H., and Khan, R. Z. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices* (2015), 163–172.

[13] Kira, K., and Rendell, L. A. A practical approach to feature selection. In *Machine Learning Proceedings 1992*. Elsevier, 1992, pp. 249–256.

[14] Kira, K., Rendell, L. A., et al. The feature selection problem: Traditional methods and a new algorithm. In *AAAI* (1992), vol. 2, pp. 129–134.

[15] Kohavi, R., and John, G. H. Wrappers for feature subset selection. *Artificial intelligence 97*, 1-2 (1997), 273–324.

[16] Kononenko, I. Estimating attributes: analysis and extensions of relief. In *European conference on machine learning* (1994), Springer, pp. 171–182.

[17] Krstajic, D., Buturovic, L. J., Leahy, D. E., and Thomas, S. Cross-validation pitfalls when selecting and assessing regression and classification models. *Journal of cheminformatics 6*, 1 (2014), 1–15.

[18] Liu, S., Xu, C., Zhang, Y., Liu, J., Yu, B., Liu, X., and Dehmer, M. Feature selection of gene expression data for cancer classification using double RBF-kernels. *BMC bioinformatics 19*, 1 (2018), 1–14.

[19] Pal, M., and Foody, G. M. Feature selection for classification of hyperspectral data by svm. *IEEE Transactions on Geoscience and Remote Sensing 48*, 5 (2010), 2297–2307.

[20] Peng, H., Long, F., and Ding, C. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 8 (2005), 1226–1238.

[21] Sebban, M., and Nock, R. A hybrid filter/wrapper approach of feature selection using information theory. *Pattern recognition 35*, 4 (2002), 835–846.

[22] Tang, Y., Zhang, Y. Q., and Huang, Z. Development of two-stage svmrfe gene selection strategy for microarray expression data analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics 4*, 3 (2007), 365–381.

[23] Urbanowicz, R. J., Olson, R. S., Schmitt, P., Meeker, M., and Moore, J. H. Benchmarking relief-based feature selection methods for bioinformatics data mining. *Journal of biomedical informatics 85* (2018), 168–188.

[24] Wasserstein, R. L., Schirm, A. L., and Lazar, N. A. Moving to a world beyond "$p < 0.05$", 2019.

[25] Wei, Z., Wang, W., Bradfield, J., Li, J., Cardinale, C., Frackelton, E., Kim, C., Mentch, F., Van Steen, K., Visscher, P. M., et al. Large sample size, wide variant spectrum, and advanced machine-learning technique boost risk prediction for inflammatory bowel disease. *The American Journal of Human Genetics 92*, 6 (2013), 1008–1012.

[26] Zhou, X., and Mao, K. LS bound based gene selection for dna microarray data. *Bioinformatics 21*, 8 (2005), 1559–1564.

Chapter 3

High-Dimensional Feature Selection for Genomic Datasets

(This chapter is based on a paper published in Knowledge-Based Systems, 2020 [2])

3.1 Introduction

Supervised learning is a central problem in machine learning and data mining [9]. In this process, a mathematical/statistical model is trained and generated based on a pre-defined number of instances (train data) and is tested against the remaining (test data). A subcategory of supervised learning is classification, where the model is trained to predict class labels [28]. For instance, in tumor datasets, class labels can be malignant or benign, the former being cancerous and the latter being non-cancerous tumors [36]. For each instance in a classification problem, there exists a set of features

that contribute to the output [22].

In high-dimensional datasets, there are a large number of irrelevant features that have no correlation with the class labels. Irrelevant features act as noise in the data that not only increase the computational costs but, in some cases, divert the learning process toward weak model generation [23, 15]. The other important issue is the presence of correlation between good features, which makes some features redundant. Redundancy is known as multicollinearity in a broader context and it is known to create overfitting and bias in regression when a model is trained on data from one region and predicted on another region [16, 38].

The goal of feature selection (FS) methods is to select the most important and effective features [21]. As such, FS can decrease the model complexity in the training phase while retaining or improving the classification accuracy. Recent FS methods [19, 45, 10] usually find the most important features through a complex model which introduce a more complicated framework when followed by a classifier.

In this chapter, we present a linear FS method called dimension reduction based on perturbation theory (DRPT). Let $D = [A \mid \mathbf{b}]$ be a dataset where $\mathbf{b}$ is the class label and A is an $m \times n$ matrix whose columns are features. We shall focus on datasets where $m << n$ and of particular interests to us are genomic datasets where gene expression level of samples (cases and controls) are measured. So, each feature is the expression levels of a gene measured across all samples. Biologically speaking, there is only a limited number of genes that are associated to a disease and, as such, only expression

levels of certain genes can differentiate between cases and controls [27, 18, 11]. So, a majority of genes are considered irrelevant. One of the most common methods to filter out irrelevant features in genomic datasets is using p-values. That is, one can look at the expression levels of a gene in normal and disease cases and calculate the p-values based on some statistical tests. It has been customary to conclude that genes whose p-values are not significant are irrelevant and can be filtered out. However, genes expressions are not independent events (variables) and researchers have been warned against the misuse of statistical significance and p-values, as it is recently pointed out in [4, 41].

We consider the system $A\mathbf{x} = \mathbf{b}$ where the rows of A are independent of each other and $A\mathbf{x} = \mathbf{b}$ is an underdetermined linear system. This is the case for genomic datasets because each sample has different gene expressions from the others. Since $A\mathbf{x} = \mathbf{b}$ may not have a unique solution, instead we use the least squares method and the pseudo-inverse of A to find the solution with the smallest 2-norm. One can view each component x_i of $\mathbf{x}$ as an assigned weight to the column (feature) $\mathbf{F}_i$ of A. Therefore, the bigger the $|x_i|$ the more important $\mathbf{F}_i$ is in connection with $\mathbf{b}$.

It then makes sense to filter out those features whose weights are very small compared to the average of local maximums over $|x_i|$'s. After removing irrelevant features, we obtain a reduced dataset, which we still denote it by $[A \mid \mathbf{b}]$. In the next phase, we detect correlations between columns of A by perturbing A using a randomly generated matrix E of small norm. Let $\tilde{\mathbf{x}}$ be the solution to $(A+E)\tilde{\mathbf{x}} = \mathbf{b}$.

It follows from Theorem 3.3.3 that features $\mathbf{F}_i$ and $\mathbf{F}_j$ correlate if and only if $|x_i - \tilde{x}_i|$ and $|x_j - \tilde{x}_j|$ are almost the same. Next, we cluster $\Delta\mathbf{x} = |\mathbf{x} - \tilde{\mathbf{x}}|$ using a simplified least-squares method called Savitsky-Golay smoothing filter [33]. This process yields a step-wise function where each step is a cluster. We note that features in the same cluster do not necessarily correlate and so we further break up each cluster of $\Delta\mathbf{x}$ into sub-clusters using entropy of features. Finally, from each sub-cluster, we pick a feature and rank all the selected features using entropy.

The "stability" of a feature selection algorithm is recently discussed in [29]. An algorithm is 'unstable' if a small change in data leads to large changes in the chosen feature subset. In real datasets, it is possible that there are small noise or error involved in the data. Also, the order of samples (rows) in a dataset should not matter; the same applies to the order of features (columns). In Theorem 3.3.4 we prove that DRPT is noise-robust and in Theorem 3.3.5 we prove that DRPT is stable with respect to permuting rows or columns.

We compare our method with seven state-of-the-art FS methods, namely mRMR [31], LARS [17], HSIC-Lasso [46], Fast-OSFS [42], group-SAOLA [52], CCM [12] and BCOA [14] over ten genomic datasets ranging up from 9,117 to 267,604 features. We use Support Vector Machines (SVM) and Random Forest (RF) to classify the datasets based on the selected features by FS algorithms. The results show that, over all, the classification accuracy of DRPT is favorable compared with each individual FS algorithm. Also, we report the running time, CPU time, and memory usage and

demonstrate that DRPT does well compared to other FS methods.

The rest of this chapter is organized as follows. In Section 3.2, we review related work. We present our approach and the algorithm in Section 3.3. Experimental results and performance comparisons are shown in Section 3.4 and we conclude the chapter in Section 3.5.

3.2 Related Work

FS methods are categorized as filter, wrapper and embedded methods [25]. Filter methods evaluate each feature regardless of the learning model. Wrapper-based methods select features by assessing the prediction power of each feature provided by a classifier. The quality of the selected subset using these methods is very high, but wrapper methods are computationally inefficient. The last group consolidates the advantages of both methods, where a given classifier selects the most important features simultaneously with the training phase. These methods are powerful, but the feature selection process cannot be defused from the classification process.

Providing the most informative and important features to a classifier would result in a better prediction power and higher accuracy [49]. Selecting an optimal subset of features is an NP-hard problem that has attracted many researchers to apply meta-heuristic and stochastic algorithms [44, 40, 30].

Several methods exist that aim to enhance classification accuracy by assigning a common discriminative feature set to local behavior of data in different regions of the

feature space [37, 5]. For example, localized feature selection (LFS) is introduced by Armanfard et al. [5], in which a set of features is selected to accommodate a subset of samples. For an arbitrary query of the unseen sample, the similarity of the sample to the representative sample of each region is calculated, and the class label of the most similar region is assigned to the new sample.

On the other hand, some approaches use aggregated sample data to select and rank the features [31, 17, 46, 39, 13]. The least absolute shrinkage and selection operator (LASSO) is an estimation method in linear models which simultaneously applies variable selection by setting some coefficients to zero [39].

Least angle regression (LARS) proposed by Efron et al. [17] is based on LASSO and is a linear regression method which computes all least absolute shrinkage and selection operator [39] estimates and selects those features which are highly correlated to the already selected ones.

Chen et al. [13] introduced a semi-supervised FS called Rescaled Linear Square Regression (RLSR), in which rescaling factors are incorporated to exploit the least square regression model and rank features. They solve the minimization problem shown in equation 3.1 to learn $\mathbf{\Theta}$ and $\mathbf{Y}_U$, which are a matrix of rescaling factors and unknown labels, respectively.

$$\begin{aligned} &\min \left(||\mathbf{X}^T \mathbf{\Theta} \mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Y}||_F^2 + \gamma ||\mathbf{W}||_F^2 \right) \\ &\text{st.} \mathbf{W}, \mathbf{b}, \theta > 0, \mathbf{1}^T \theta = 1, \mathbf{Y}_U \geq 0, \mathbf{Y}_U \mathbf{1} = \mathbf{1}, \end{aligned} \tag{3.1}$$

where $\mathbf{W}$ is a sparse matrix that represents the importance of features, $\mathbf{X}$ is the

dataset, $\mathbf{Y}$ is class labels, and $\mathbf{b} = \frac{1}{n}(\mathbf{Y}^T\mathbf{1} - \mathbf{W}^T\mathbf{X}\mathbf{1})$, where n is number of samples in a dataset. Their proposed algorithm continuously updates $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{Y}_U$ until convergence.

Yamada et al. [46] proposed a non-linear FS method for high-dimensional datasets called Hilbert-Schmidt independence criterion least absolute shrinkage and selection operator (HSIC-Lasso), in which the most informative non-redundant features are selected using a set of kernel functions, where the solutions are found by solving a LASSO problem. The complexity of the original Hilbert-Schmidt FS (HSFS) is $O(n^4)$. In a recent work [47] called Least Angle Nonlinear Distributed (LAND), the authors have improved the computational power of the HSIC-Lasso. They have demonstrated via some experiments that LAND and HSIC-Lasso attain similar classification accuracies and dimension reduction. However, LAND has the advantage that it can be deployed on parallel distributed computing. Another kernel-based feature selection method is introduced in [12] using measures of independence and minimizing the trace of the conditional covariance operator. It is motivated by selecting the features that maximally account for the dependence on the covariates' response.

In some recent real-world applications, we need to deal with sequentially added dimensions in a feature space while the number of data instances is fixed. Yu et al. [50] developed an open source Library of Online FS (LOFS) using state-of-the-art algorithms. The learning module of LOFS consists of two submodules, Learning Features added Individually (LFI) and Learning Grouped Features added sequentially

(LGF). The LFI module includes various FS methods including Alpha-investing [53], OSFS [43], Fast-OSFS [42], and SAOLA [51] to learn features added individually over time, while the LGF module provides the group-SAOLA algorithm [52] to mine grouped features added sequentially.

In [32], the authors proposed a bio-inspired optimization algorithm called Coyote Optimization Algorithm (COA) simulating the behavior of coyotes. Using their new strategy, COA makes a balance between exploration and exploitation processes to solve continuous optimization problems. Very recently, the authors [14] upgraded their method by proposing a binary version of COA called Binary Coyote Optimization Algorithm (BCOA), which is a wrapper feature selection method.

There is a great interest in the applications of FS methods in disease diagnoses and prognoses. For example, Parkinson's disease (PD) is a critical neurological disorder and its diagnoses in its initial stages is extremely complex and time consuming. Recently, voice recordings and handwritten drawings of PD patients are used to extract a subset of important features using FS algorithms to diagnose PD [3, 6, 34, 24, 48] with a good success rate.

3.3 Proposed approach

Consider a dataset D consisting of m samples where each sample contains $n + 1$ features. Let us denote by A the first n columns of D and by $\mathbf{b}$ the last column. We also denote by $\mathbf{F}_i$ the i-th feature (column) of A. We shall first consider eliminating

the irrelevant features. Throughout this chapter, by the norm of a vector, we always mean its 2-norm. Recall that

$$||A||= \text{Sup}_{\mathbf{x}\neq 0}\frac{||A\mathbf{x}||}{||\mathbf{x}||} = \text{Sup}_{||\mathbf{x}||=1}||A\mathbf{x}||$$

Denote by $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ the singular values of A, where $r = \min(m, n)$. The smallest non-zero singular value of A is denoted by $\sigma_{\min}$ and the greatest of the σ_i is also denoted by $\sigma_{\max}$. It is well-known that $||A||_2= \sigma_{\max}$. Recall that A admits a singular value decomposition (SVD) in the form $A = U\Sigma V^T$, where U and V are orthogonal matrices and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$ is an $m \times n$ diagonal matrix. Here, V^T denotes the transpose of V.

We first normalize the columns of A so that each $\mathbf{F}_i$ has norm 1. Then, we solve the linear system $A\mathbf{x} = \mathbf{b}$ by using the method of least squares (see theorem 3.3.1). Here, $\mathbf{x} = [x_1 \cdots x_i \cdots x_n]^T$. The idea is to select a small number of columns of A that can be used to approximate $\mathbf{b}$. Since $A\mathbf{x} = \mathbf{b}$ may not have a unique solution, instead we consider a broader picture by solving the least squares problem $\min_{\mathbf{x}}||A\mathbf{x} - \mathbf{b}||_2$ whose solution is given in terms of pseudo-inverse and SVD of A. The following result is well-known, see [8].

Theorem 3.3.1 (All Least Squares Solutions)**.** *Let A be an $m \times n$ matrix and $\mathbf{b} \in \mathbb{R}^m$. Then all the solutions of $min_{\mathbf{x}}||A\mathbf{x} - \mathbf{b}||_2$ are of the form $\mathbf{y} = A^+\mathbf{b} + \mathbf{z}$, where $\mathbf{z} \in \ker(A)$. Furthermore, the unique solution whose 2-norm is the smallest is given by $\mathbf{x} = A^+\mathbf{b}$.*

In other words, we can approximate the label column $\mathbf{b}$ as a linear combination $x_1\mathbf{F}_1 + \cdots + x_n\mathbf{F}_n$. So each $|x_i|$ can be viewed as an assigned weight to $\mathbf{F}_i$. Given that each $\mathbf{F}_i$ has norm 1, if $|x_i|$ is small compared to others, then the vector $x_i\mathbf{F}_i$ will have a negligible effect on $\mathbf{b}$. It then makes sense to filter out those features whose weights are very small. In other words, we shrink the weights of irrelevant features to zero.

In this chapter, we shall mostly focus on datasets where $m << n$. Of special interest to us are genomic datasets where there are usually tens or hundreds of samples compared to tens of thousands of genes. The matrix A in these datasets has full row-rank because gene expressions of different samples are independent of each other. Since $\mathbf{b} \in \mathbb{R}^m$, it is enough to identify only m independent columns of A. Intuitively, it makes sense to eliminate the columns that are less important.

We prove below how weight of each feature $\mathbf{F}_i$ is directly affected by the relevance of $\mathbf{F}_i$ to $\mathbf{b}$. Suppose that A is an $m \times n$ of full row-rank and consider the SVD of A as $A = U\Sigma V^T$. Let $U = \left[\begin{array}{c|c|c} \mathbf{u}_1 & \cdots & \mathbf{u}_m \end{array}\right]$. Note that $\mathbf{u}_1, \ldots, \mathbf{u}_m$ form an orthonormal basis of $\mathbb{R}^m$.

Theorem 3.3.2. *Let A be a full row-rank matrix and denote by $\mathbf{x} = [x_1 \ldots x_n]^T$ the least squares solutions to $A\mathbf{x} = \mathbf{b}$. Then, each component x_i of $\mathbf{x}$ is given by $x_i = \langle \mathbf{F}_i, \mathbf{z} \rangle$, where $\mathbf{z} = U[\langle \mathbf{u}_1, \mathbf{b} \rangle / \sigma_1^2 \cdots \langle \mathbf{u}_m, \mathbf{b} \rangle / \sigma_m^2]^T$.*

Proof. Since A is full row-rank, the right inverse of A is $A^+ = A^T(AA^T)^{-1}$. Consider the SVD of A as $A = U\Sigma V^T$. Then $AA^T = U\Sigma\Sigma^T U^T = \sum_{i=1}^m \sigma_i^2 u_i u_i^T$. Note that

the solution of $A\mathbf{x} = \mathbf{b}$ with the smallest norm is $\mathbf{x} = A^{+}\mathbf{b} = A^{T}(AA^{T})^{-1}\mathbf{b}$. Let $\mathbf{z} = (AA^{T})^{-1}\mathbf{b}$. So, $\mathbf{b} = AA^{T}\mathbf{z} = \sum_{i=1}^{m} \sigma_i^2 u_i u_i^T \mathbf{z}$. Since the $\mathbf{u}_i$s are orthonormal, we get

$$\langle \mathbf{u}_k, \mathbf{b} \rangle = \sigma_k^2 \langle \mathbf{u}_k, \mathbf{z} \rangle, \quad k = 1, \ldots, m. \tag{3.2}$$

Since A has full row-rank, we have $\sigma_k > 0$, for all k. Let $\bar{b}_1, \ldots, \bar{b}_m$ be the coordinates of $\mathbf{b}$ with respect to the basis $\mathbf{u}_1, \ldots, \mathbf{u}_m$ of $\mathbb{R}^m$. Similarly, let $\bar{z}_1, \ldots, \bar{z}_m$ be the coordinates of $\mathbf{z}$ with respect to this basis. So, $\mathbf{b} = U[\bar{b}_1 \ \ldots \ \bar{b}_m]^T$ and $\mathbf{z} = U[\bar{z}_1 \ \ldots \ \bar{z}_m]^T$. Equation (3.2) can be written as $\bar{b}_k = \sigma_k^2 \bar{u}_k$, for each $1 \leq k \leq m$. On the other hand, $\mathbf{x} = A^{+}\mathbf{b} = A^{T}(AA^{T})^{-1}\mathbf{b} = A^{T}\mathbf{z}$. Since $\mathbf{z} = U[\bar{z}_1 \ \ldots \ \bar{z}_m]^T$, we deduce that $x_i = \langle \mathbf{F}_i, \mathbf{z} \rangle$, for each i. □

We note that the extent to which a feature is relevant to $\mathbf{b}$ also depends on how important the other features are in determining $\mathbf{b}$. This fact is reflected in Theorem 3.3.2 by taking into account the singular values of A that encode part of the information about A. Also, the definition of relevancy is not quantitative and one has to set a threshold for the degree of relevancy. We set a dynamic threshold by calculating the average of all local maxima in $\mathbf{x}$ and remove those features that their corresponding value $|x_i|$ is smaller than the threshold. In a sense, the threshold is set so that rank of the reduced matrix is still the same as of the original A. So, in the reduced matrix, we only keep features that have a higher impact on $\mathbf{b}$ and yet the reduced matrix retains the same prediction power as A in approximating $\mathbf{b}$.

If A is full row-rank then, it follows from Theorem 3.3.1 that the solution $\mathbf{x}^R$ of

smallest 2-norm to the system $A\mathbf{x} = \mathbf{b}$ is in the row space of A. So, there is a vector $\mathbf{y}$ such that $\mathbf{x}^R = -A^T\mathbf{y}$. Hence, $\mathbf{x}^R$ satisfies $\mathbf{x}^R + A^T\mathbf{y} = 0, A\mathbf{x}^R = \mathbf{b}$. In other words, $\mathbf{x}^R$ is part of the solution to the non-singular linear system

$$\begin{pmatrix} I & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x^R \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}$$

Next, we show how we can detect correlations between features. Recall that a perturbation of A is of the form $A + E$, where E is a random matrix with the normal distribution. We choose E to be a random matrix such that $||E||_2 \approx 10^{-s}\sigma_{\min}(A)$, for some $s \geq 0$. We set $s = 3$ where our estimates are correct up to a magnitude of 10^{-3}.

Theorem 3.3.3. *Let* $\mathbf{x}$ *and* $\tilde{\mathbf{x}}$ *be solutions of* $A\mathbf{x} = \mathbf{b}$ *and* $(A + E)\tilde{\mathbf{x}} = \mathbf{b}$*, where* E *is a perturbation such that such that* $||E||_2 = 10^{-s}\sigma_{min}(A)$*. If a feature* $\mathbf{F}_i$ *is independent of the rest of the features, then* $|x_i - \tilde{x}_i| \approx 0$*. Furthermore, suppose that features* $\mathbf{F}_j$ *and* $\mathbf{F}_k$ *correlate, say* $\mathbf{F}_j = c\mathbf{F}_k$ *for some scalar* c*. If* $\mathbf{F}_j$ *and* $\mathbf{F}_k$ *are independent from the rest of the features, then* $c = \frac{x_k - \tilde{x}_k}{x_j - \tilde{x}_j}$.

Proof. From $A\mathbf{x} = \mathbf{b}$ and $(A + E)\tilde{\mathbf{x}} = \mathbf{b}$, we get $A(\mathbf{x} - \tilde{\mathbf{x}}) = E\tilde{\mathbf{x}}$. Consider the SVD of $A + E$ which is of the form $A + E = U\Sigma V^T$. So, $\tilde{\mathbf{x}} = V\Sigma^{-1}U^T\mathbf{b}$. Since U and V are orthogonal and for orthogonal matrices we have $||U\mathbf{v}||_2 = ||\mathbf{v}||_2$, we get

$$\begin{aligned} ||\tilde{\mathbf{x}}||_2 = ||V\Sigma^{-1}U^T\mathbf{b}||_2 &= ||\Sigma^{-1}\mathbf{b}||_2 \\ &\leq ||\Sigma^{-1}||_2||\mathbf{b}||_2 = \frac{1}{\sigma_{\min}(A + E)} \\ &\leq \frac{1}{-||E||_2 + \sigma_{\min}(A)}. \end{aligned}$$

Hence, $||E\tilde{\mathbf{x}}||_2 \leq ||E||||\tilde{\mathbf{x}}||_2 \leq 10^{-s}$ and we deduce that

$$(x_1 - \tilde{x}_1)\mathbf{F}_1 + \cdots + (x_t - \tilde{x}_t)\mathbf{F}_t + \cdots + (x_n - \tilde{x}_n)\mathbf{F}_n \approx 0.$$

Now, if a feature, say $\mathbf{F}_i$, is independent of the rest of features, then it follows that $|x_i - \tilde{x}_i| \approx 0$. Furthermore, since $\mathbf{F}_j$ and $\mathbf{F}_k$ are independent from the rest of the features, we must have

$$(x_j - \tilde{x}_j)\mathbf{F}_j + (x_k - \tilde{x}_k)\mathbf{F}_k \approx 0.$$

So, $\mathbf{F}_j = \frac{x_k - \tilde{x}_k}{x_j - \tilde{x}_j}\mathbf{F}_k$. Hence, $c = \frac{x_k - \tilde{x}_k}{x_j - \tilde{x}_j}$, as required. □

Theorem 3.3.3 shows how we can filter out irrelevant features by looking at the components of $\mathbf{x} - \tilde{\mathbf{x}}$ that are close to zero. Also, as we mentioned before, we normalize the columns of A so that each $\mathbf{F}_i$ has norm 1. So, if in the raw dataset $\mathbf{F}_j$ and $\mathbf{F}_k$ correlate then after normalization we must have $\mathbf{F}'_j = \pm\mathbf{F}'_k$. Here, $\mathbf{F}'_j = \frac{\mathbf{F}_j}{||\mathbf{F}_j||}$. So, by Theorem 3.3.3, if $\mathbf{F}_j$ and $\mathbf{F}_k$ are independent from other features, we must have $|x_k - \tilde{x}_k| = |x_j - \tilde{x}_j|$. We explain these notions further in a synthetic dataset. Consider a synthetic dataset with 22 features and 100 samples and the label column $\mathbf{b}$ which we set as $\mathbf{b} = 3\mathbf{F}_{19} + 5\mathbf{F}_{17} + 2\mathbf{F}_{20}$. The first 20 features of this dataset are generated randomly in the interval of -1 and 1. The correlations between remaining features are set as follows: $\mathbf{F}_{21} = 2\mathbf{F}_{18} + 4\mathbf{F}_{19}$ and $\mathbf{F}_{22} = 3\mathbf{F}_{20}$. First, we normalize A. Then solve $A\mathbf{x} = \mathbf{b}$ and $(A + E)\tilde{\mathbf{x}} = \mathbf{b}$ and calculate $\Delta\mathbf{x}$ as shown in Table 3.1.

Table 3.1: Perturbation of the synthetic Dataset

	$\mathbf{x}$	$\tilde{\mathbf{x}}$	$\Delta\mathbf{x} = \lvert\mathbf{x} - \tilde{\mathbf{x}}\rvert$
$\mathbf{F}_1 \ldots \mathbf{F}_{16}$	$\leq$3.0987e-14	$\leq$2.6907e-05	$\leq$4.7316e-05
$\mathbf{F}_{17}$	29.1715	29.1715	2.7239e-05
$\mathbf{F}_{18}$	-3.4494	-10.2466	6.7972
$\mathbf{F}_{19}$	9.9339	-3.1806	13.1145
$\mathbf{F}_{20}$	-5.3307	-6.0073	0.6766
$\mathbf{F}_{21}$	7.3630	21.8723	14.5093
$\mathbf{F}_{22}$	-5.3307	-4.6541	0.6766

Let $\Delta\mathbf{x} = |\mathbf{x}-\tilde{\mathbf{x}}|$ and denote its i-th component with Δx_i. Since each of $\mathbf{F}_1, \ldots, \mathbf{F}_{17}$ are independent from the other features, as we expected, we have $\Delta x_i \approx 0$, for all $1 \leq i \leq 17$. However, $\mathbf{F}_{17}$ is relevant because it correlates with $\mathbf{b}$. Indeed, $\mathbf{F}_{17}$ is a very important feature because we cannot make up for its loss using other features. Hence, we should be able to distinguish and preserve $\mathbf{F}_{17}$ from other $\mathbf{F}_i$ for which $\Delta x_i = 0$. This can be accomplished by noting that irrelevant features have smaller $|x_i|$ compared to other features as can be seen from Table 3.1.

Since $\mathbf{F}_{22} = 3\mathbf{F}_{20}$, these features correlate and are independent from other features. By normalization, we get $\mathbf{F}'_{22} = \mathbf{F}'_{20}$. So, we expect to have $\Delta x_{20} \approx \Delta x_{22}$ as shown in Table 3.1. We deduce that $\mathbf{F}_{22}$ and $\mathbf{F}_{20}$ are dependent and so we should only choose one of them.

Finally, after normalization and rewriting the relation $\mathbf{F}_{21} = 2\mathbf{F}_{18} + 4\mathbf{F}_{19}$, we obtain $\mathbf{F}'_{21} \approx \mathbf{F}'_{19}$ modulo $\mathbf{F}'_{18}$. The reason for this is because norms of $\mathbf{F}'_{21}$ and $\mathbf{F}'_{19}$ outweigh norm of $\mathbf{F}'_{18}$. This is confirmed in Tabel 3.1 as Δx_{19} and Δx_{21} are closest to each other compared to the others. So, $\mathbf{F}_{19}$ and $\mathbf{F}_{21}$ fall into the same cluster of $\Delta\mathbf{x}$. This means that one of $\mathbf{F}_{19}$ or $\mathbf{F}_{21}$ must be removed as a redundant feature. We shall now explain the clustering process of $\Delta\mathbf{x}$.

By Theorem 3.3.3, if features $\mathbf{F}_i$ and $\mathbf{F}_j$ correlate, then the differences Δx_i and Δx_j are almost the same. That is, the correlations between features are encoded in $\Delta\mathbf{x}$. Now, we sort $\Delta\mathbf{x}$ and obtain a stepwise function where each step can be viewed as a cluster consisting of features that possibly correlate with each other. To find an optimal number of steps, it makes sense to smooth $\Delta\mathbf{x}$ where we view $\Delta\mathbf{x}$ as a signal and use a simplified least-squares method called Savitsky-Golay smoothing filter [33]. Figure 3.1 exhibits how the smoothening process on $\Delta\mathbf{x}$ preserves its whole structure without changing the trend.

We note that the converse of Theorem 3.3.3 may not be true in general. That is Δx_i and Δx_j being the same does not necessarily imply that $\mathbf{F}_i$ and $\mathbf{F}_j$ correlate. Hence, in the next step, we want to further break up each cluster of $\Delta\mathbf{x}$ into sub-clusters. There are several ways to accomplish this step and one of the most natural ways is to use entropy of features.

Generally, entropy is a key measure for information gain and it is capable of quantifying the disorder or uncertainty of random variables. Also, entropy effectively

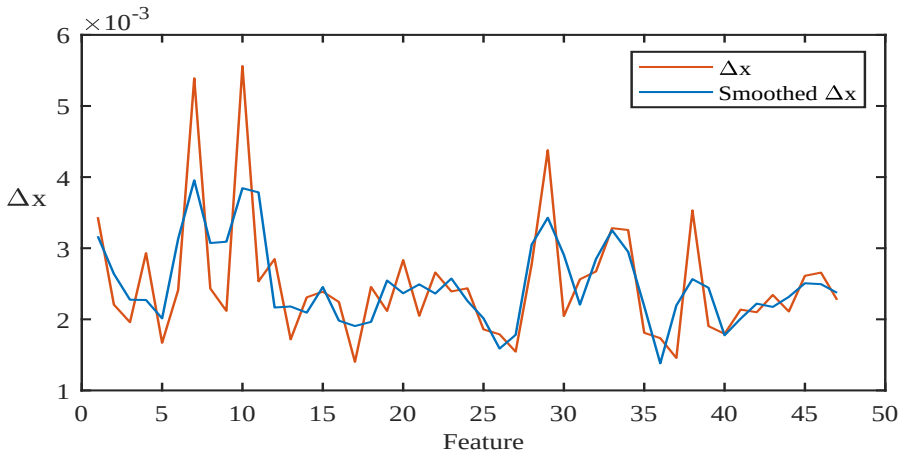


Figure 3.1: $\Delta\mathbf{x}$ vs. smoothed $\Delta\mathbf{x}$

scales the amount of information that is carried by random variables. Entropy of a feature $\mathbf{F}$ is defined as follows:

$$H(\mathbf{F}) = -\sum_{k=1}^{m} f_k \log f_k \tag{3.3}$$

where m is the number of samples and f_k is the frequency with which $\mathbf{F}$ assumes the k-th value in the observations.

Figure 3.2(a) shows clustering the set of all features based on $\Delta\mathbf{x}$, and then a typical cluster splits into sub-clusters using entropy as shown in Figure 3.2(b). To do so, we sort the features of a cluster based on their entropy which yields another step-wise function. At this stage, we pick one candidate feature from each sub-cluster based on the corresponding values $|x_i|$. Finally, the selected features are ranked based on both their entropies and the corresponding $|x_i|$'s. The final sorting of the features is an amalgamation of these two rankings.

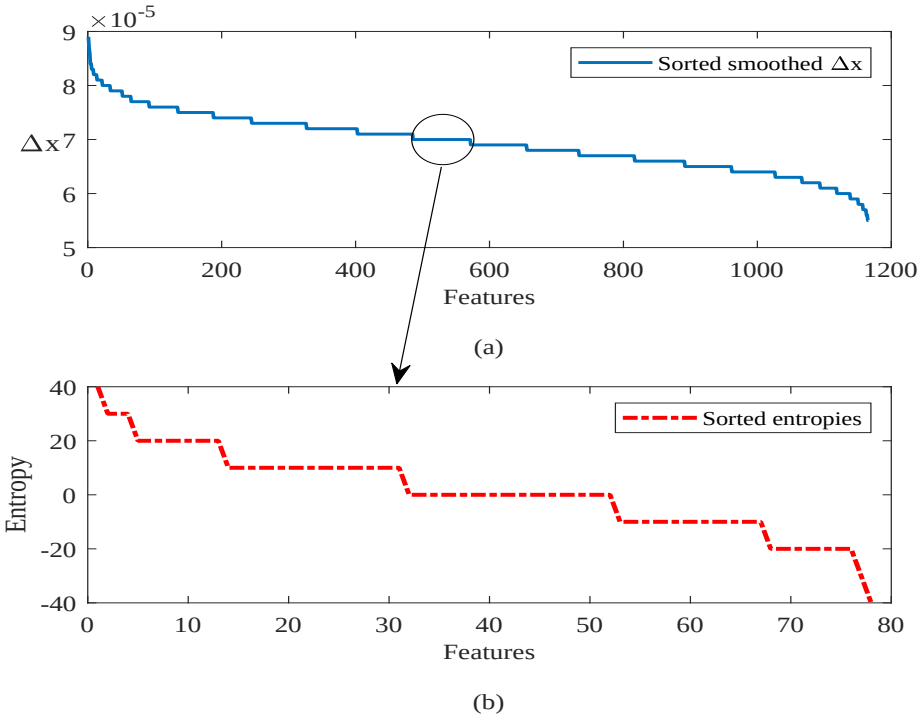


Figure 3.2: (a) Sorted smoothed $\mathbf{\Delta x}$ (b) Sorted entropies of the magnified cluster

3.3.1 Noise-robustness and stability

In real datasets, it is likely that D involves some noise. For example, in genomics, it is conceivable that through the process of preparing a genomic dataset, some error/noise is included and as such the dataset D is noisy. We note that the label column $\mathbf{b}$ is already known to us (and without noise). So instead of $D = [A \mid \mathbf{b}]$ we deal with $D = [A_1 \mid \mathbf{b}]$, where $A_1 = A + E_1$ and $||E_1||_2$ is small ($||E_1||_2 = 10^{-s}\sigma_{\min}(A)$). A perturbation of A_1 is of the form $\tilde{A}_1 = A_1 + E_2$, where $||E_2||_2 = 10^{-s}\sigma_{\min}(A)$. Our aim is to show that if certain columns of A correlate, then so do the same columns of A_1 and vice versa.

Theorem 3.3.4. *Let $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ be solutions of $A_1\tilde{\mathbf{x}} = \mathbf{b}$ and $\tilde{A}_1\tilde{\mathbf{y}} = \mathbf{b}$, respectively. Suppose that $S' = \{\mathbf{F}_1, \ldots, \mathbf{F}_t\}$ is set of columns of A such that $\sum_{i=1}^{t} c_i\mathbf{F}_i = 0$, for some non-zero c_i. If*

1. *any subset of S' is linearly independent,*

2. *$\mathbf{F}_1, \ldots, \mathbf{F}_t$ are linearly independent from the remaining columns of A.*

Then the vectors $\begin{bmatrix} c_1 & \cdots & c_t \end{bmatrix}$ and $\begin{bmatrix} \tilde{x}_1 - \tilde{y}_1 & \cdots & \tilde{x}_t - \tilde{y}_t \end{bmatrix}$ are proportional.

Proof. From $(A + E_1)\tilde{\mathbf{x}} = \mathbf{b}$ and $(A + E_1 + E_2)\tilde{\mathbf{y}} = \mathbf{b}$, we get $A(\tilde{\mathbf{x}} - \tilde{\mathbf{y}}) = -E_1\tilde{\mathbf{x}}_1 + (E_1 + E_2)\tilde{\mathbf{y}}$. Similar arguments as in the proof of Theorem 3.3.3 can be used to show

that

$$\begin{aligned}||-E_1\tilde{\mathbf{x}} + (E_1 + E_2)\tilde{\mathbf{y}}|| &\leq ||-E_1\tilde{\mathbf{x}}_1|| + ||(E_1 + E_2)\tilde{\mathbf{y}}|| \\ &\leq \frac{1}{10^s - 1} + \frac{2 \cdot 10^{-s}}{-2 \cdot 10^{-s} + 1} \\ &\leq \frac{1}{10^s - 1} + \frac{2}{10^s - 2} \\ &\approx 3 \cdot 10^{-s}\end{aligned}$$

We deduce that

$$(\tilde{x}_1 - \tilde{y}_1)\mathbf{F}_1 + \cdots + (\tilde{x}_t - \tilde{y}_t)\mathbf{F}_t + \cdots + (\tilde{x}_n - \tilde{y}_n)\mathbf{F}_n \approx 0. \tag{3.4}$$

Since $\mathbf{F}_1, \ldots, \mathbf{F}_t$ are linearly independent from the rest of features in S, we get

$$(\tilde{x}_1 - \tilde{y}_1)\mathbf{F}_1 + \cdots + (\tilde{x}_t - \tilde{y}_t)\mathbf{F}_t \approx 0. \tag{3.5}$$

Now, if $\begin{bmatrix} c_1 & \ldots & c_t \end{bmatrix}$ and $\begin{bmatrix} \tilde{x}_1 - \tilde{y}_1 & \ldots & \tilde{x}_t - \tilde{y}_t \end{bmatrix}$ are not proportional, we can use Equation (3.5) and our first hypothesis to get a dependence relation of a shorter length between the elements of S', which would contradict our assumption that any proper subset of S' is linearly independent. The proof is complete. □

We also remark that our method is insensitive to shuffling of the dataset D. That is, if we exchange rows (or columns), there is an insignificant change in $\Delta\mathbf{x}$. We have demonstrated this fact through experiments in Tables 3.4; we offer a proof as follows.

Theorem 3.3.5. *DRPT is insensitive to permuting rows or columns.*

Proof. We show this for permutation of rows and a similar argument can be made for permuting columns. Suppose that $D_1 = [A_1 \mid \mathbf{b}_1]$ is obtained from D by shuffling rows.

Note, matrix T is an elementary matrix where the identity matrix (ones on the main diagonal and 0s for all other entries) is reached from one elementary row operation like interchanging two rows. Hence, assume that only two rows are exchanged. Then there exists an elementary matrix T such that $TA = A_1$ and $T\mathbf{b} = \mathbf{b}_1$. Since T is invertible, it follows that $A\mathbf{x} = \mathbf{b}$ if and only if $A_1\mathbf{x} = \mathbf{b}_1$. For the general case, we note that every shuffling is a composition of elementary matrices.

3.3.2 Algorithm

The Flowchart and algorithm of DRPT are as follows. The MATLAB® implementation of DRPT is publicly available in GitHub [1].

[1] http://github.com/majid1292/DRPT

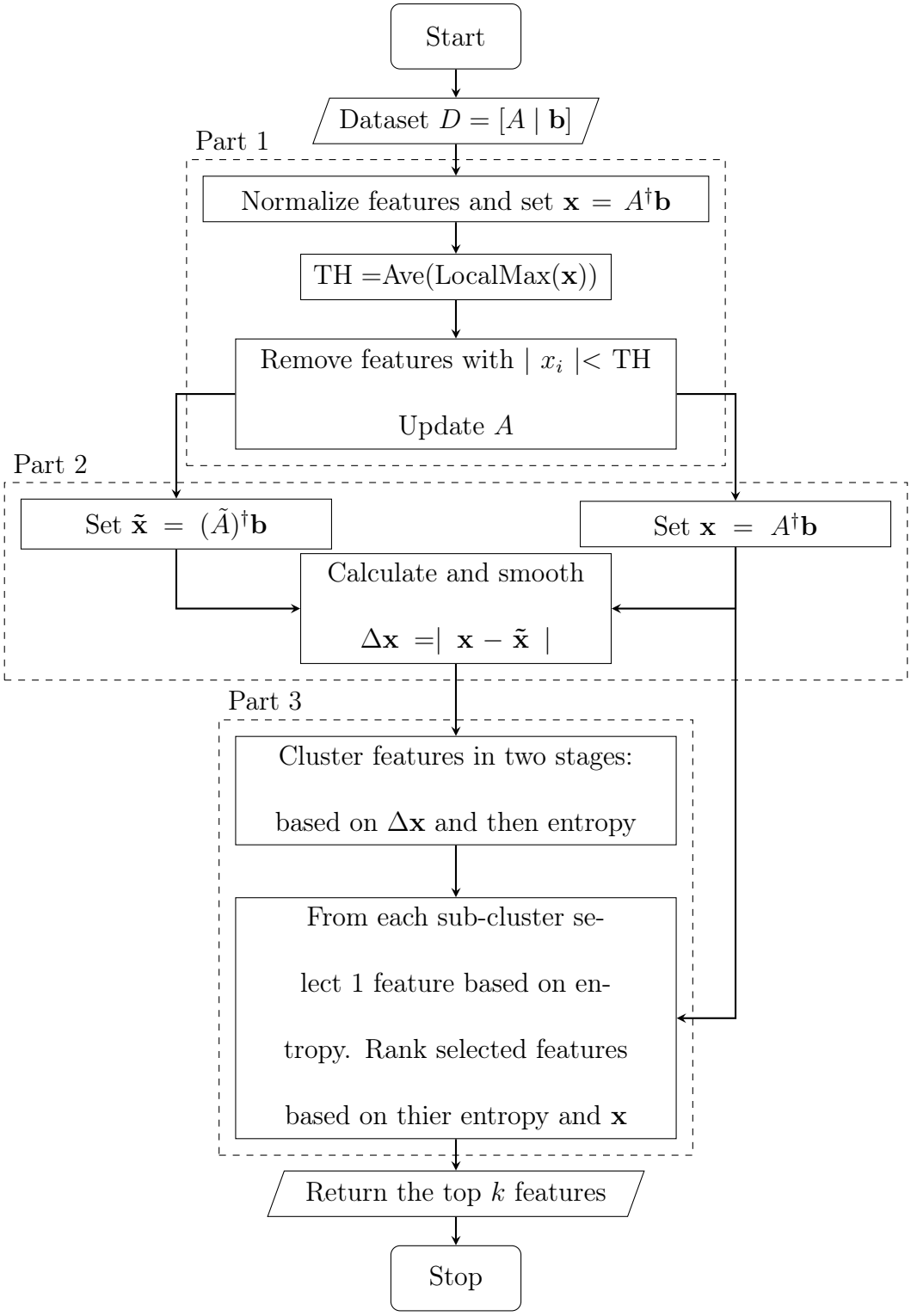


Figure 3.3: Flowchart of Dimension Reduction based on Perturbation Theory (DRPT)

3.3.3 Complexity

The complexity of our proposed method is dominated by the complexity of the SVD which is $O(mn^2, m^2n)$, since the inverse of perturbed $\tilde{A}$ is calculated using SVD.

3.4 Experimental Results

We compared our method with seven state-of-the-art FS methods, namely minimal-redundancy-maximal-relevance criterion (mRMR) [31], least angle regression (LARS) [17], Hilbert-Schmidt Independence Criterion Lasso (HSIC-Lasso) [46], Fast Online Streaming FS (Fast-OSFS) and Scalable, Accurate Online FS (group-SAOLA) [50], Conditional Covariance Minimization (CCM) [12] and Binary Coyote Optimization Algorithm (BCOA) [14] . We used MATLAB® implementations of LARS and LASSO by Sjöstrand [35], HSIC-Lasso and BCOA by their authors, Fast-OSFS and group-SAOLA given in the open source library [50]. The CCM method is implemented in Python by their authors and its code available at GitHub[2].

To have a fair comparison among different FS methods, we read the datasets by the same function and use a stratified partitioning of the dataset so that 70% of each class is selected for FS. Then we use SVM and RF classifiers implemented in MATLAB®, to evaluate the selected subsets of features on the remaining 30% of the dataset. We have used linear kernel in SVM (default setting of SVM in MATLAB®) and as for RF, we set 30 as the number of trees and the other parameters have default

[2]https://github.com/Jianbo-Lab/CCM

Algorithm 2: Dimension reduction based on perturbation theory (DRPT)

Data: $D = [A \mid \mathbf{b}], k$

Result: A subset of features of size k

`//  ***Part1:  Irrelevant Feature Removal***`

(1) Normalize columns of A within $[0, 1]$;

(2) $\mathbf{x} = A^{+}\mathbf{b}$;

(3) TH =Average(LocalMaxima of $\mathbf{x}$);

(4) $I = \emptyset$;

(5) **for** *each* $x_i \in \boldsymbol{x}$ **do**

(6) | **if** $x_i \geq TH$ **then**

(7) | | $I = I \cup i$

(8) $A \leftarrow A[I]$;

`//  ***Part2:  Detecting Correlations***`

(9) $s = 3$;

(10) $(m, n) = \text{Size}(A)$;

(11) $minSVD = Min$(singular value of A);

(12) $\mathbf{x} = A^{+}\mathbf{b}$;

(13) $t = 10^{-s} \cdot minSVD$;

(14) Set E be a random $m \times n$ matrix with uniform dist. in the interval (0,1);

(15) $E = t. * E$;

(16) $\tilde{A} = A + E$;

(17) $\Delta \mathbf{x} = |(\tilde{A})^{+}\mathbf{b} - \mathbf{x}|$;

(18) $\Delta \mathbf{x} = \text{Smooth}(\Delta \mathbf{x})$;

// ***Part3: Ranking Features***

(19) **while** $z \in unique(\Delta \boldsymbol{x})$ **do**

(20) $\quad \text{CL} = \{\mathbf{F}_i \mid |x_i - \tilde{x}_i| = z\}$

(21) $\quad$ **for** $h \in unique(H(CL))$ **do**

(22) $\quad\quad \text{subCL} = \{\mathbf{F}_i \in \text{CL} \mid H(\mathbf{F}_i) = h\}$;

(23) $\quad\quad$ Pick $\mathbf{F}_i$ in subCL with $|x_i| = \max \mathbf{x}_{\text{subCL}}$;

(24) $\quad\quad \text{Output} \leftarrow \text{Output} \cup \mathbf{F}_i$;

(25) *Ranked Output* $\leftarrow$ Rank (*Output*, $\{H(F)$ & $\mathbf{x}\}$);

(26) **Return:** the top k features;

values.

3.4.1 Datasets

We select a variety of dataset from Gene Expression Omnibus (GEO) [3] and dbGaP [4] to perform FS and classification. The specifications of all datasets are given in Table 3.2.

Table 3.2: Dataset Specifications

Dataset	**Samples**	**# Original F**	**# Cleaned F**	**# Labels**	**Proportion of labels**			
					1	**2**	**3**	**4**
GDS1615	127	22,282	13,649	3	33%	20.5%	46.5%	–
GDS3268	202	44,290	29,916	2	36.1%	63.9%	–	–
GDS968	171	12,625	9,117	4	26.3%	26.3%	22.8%	24.6%
GDS531	173	12,625	9,392	2	20.8%	79.2%	–	–
GDS2545	171	12,625	9,391	4	10.6%	36.8%	38%	14.6%
GDS1962	180	54,675	29,185	4	12.8%	14.4%	45	27.8%
GDS3929	183	24,526	19,334	2	69.9%	30.1%	–	–
GDS2546	167	12,620	9,583	4	10.2%	35.3%	39.5%	15%
GDS2547	164	12,646	9,370	4	10.4%	35.4%	39%	15.2%
NeuroX	11,402	535,202	267,601	2	48.6%	51.4%	–	–

All the GEO datasets are publicly available. To pre-process the data, we develop

[3] https://www.ncbi.nlm.nih.gov/geo/

[4] https://www.ncbi.nlm.nih.gov/gap/

an R code to clean and convert any NCBI dataset to CSV format [5]. We use GEO2R [7] to retrieve the mappings between prob IDs and gene samples. Probe IDs without a gene mapping were removed. Expression values of each gene are the average of expression values of all mapped prob IDs to that gene. We also handle missing values with k-Nearest Neighbors (kNN) imputation method.

The dataset NeuroX holds SNP information about subjects' Parkinson disease status and sociodemographic (e.g., onset age/gender) data. Parkinson's disease status coded as 0 (control) and 1 (case), from clinic visit using modified UK Brain Bank Criteria for diagnosis. The original NeuroX has 11402 samples, and it is only accessible by authorized access via dbGaP. It has 535202 features that each two sequence features are considered as a SNP. So after cleaning and merging features of NeuroX, we use two subsets of 100 and 200 samples with 267601 SNPs (NX100 and NX200) for this chapter.

3.4.2 Parameters

A FS method that selects most relevant and non-redundant features (Minimum Redundancy and Maximum Relevancy) is favorable in the sense that the top k selected features retain most of the information about the dataset. On the other hand, the top selected genes in a genomic dataset must be further analyzed in wet labs to confirm the biological relevance of the genes to the disease. For example, authors in [54] first

[5] http://github.com/majid1292/NCBIdataPrep

identified 50 top genes of a Colon cancer dataset using their FS method. Then, they selected the first 15 genes, because adding more genes would not result in significant changes to the prediction accuracy. Similar studies [20, 26] suggest considering the top 50 features. So, in Table 3.3, we set $k = 50$ to select a subset of 50 features using FS algorithms. Then, for $t = 1$ to k, we feed the first t features to the classifier to find an optimal t so that the subset of first t features yields the highest accuracy. This set up is applied across all FS methods. In Figure 3.4, we expand this idea by considering up to 90 features using each FS method. We can see that there is small, incremental changes in classification accuracies when we increase the number of features from 50 to 90.

We report the average classification accuracies and average number of selected features over 10 independent runs where the dataset is row shuffled in each run. We note the top k selected features using a FS algorithm might differ over different runs because the dataset is row shuffled and so the training set changes on every run. Also, optimal subset size for SVM and RF might be different, in other words SVM might attain the maximum accuracy using the first 20 features while RF might attain its maximum using the first 25 features.

Both Fast-OSFS and group-SAOLA have a parameter α, which is a threshold on the significance level. The authors of the LOFS [50] in their Matlab user manual [6] recommend setting $\alpha = 0.05$ or $\alpha = 0.01$. However, our experiments based on

[6] https://github.com/kuiy/LOFS/tree/master/LOFS_Matlab/manual

these parameters showed inferior classification accuracies compared to other methods across all datasets given in Table 3.2. We also note that the running times of both Fast-OSFS and group-SAOLA increased as we increased α.

Experimenting with various values of α, we realized that increasing α from 0.01 to 0.5 exhibited clear improvement in classification accuracies on all datasets except NeuroX. So, we set $\alpha = 0.5$ for all datasets except NeuroX.

We also experienced that both Fast-OSFS and group-SAOLA on NeuroX may not execute all the time when $\alpha > 0.0005$; often errors were generated as part of a statistics test function. So, for NeuroX, we set $\alpha = 10^{-5}$ for both Fast-OSFS and group-SAOLA.

The group-SAOLA model has an extra parameter for setting the number of selected groups, *selectGroups*. As there was no default or recommended value for this parameter, we obtained results by varying *selectGroups* from 2 to 10 for all the datasets, and we chose the highest accuracy for each dataset.

3.4.3 Hardware and Software

Our proposed method DRPT and other FS methods in section 3.4 have been run on an IBM®LSF 10.1.0.6 machine (Suite Edition: IBM Spectrum LSF Suite for HPC 10.2.0) with requested 8 nodes, 24 GB of RAM, and 10 GB swap memory using MATLAB®R2017a (9.2.0.556344) 64-bit(glnxa64). Since CCM is implemented in Python and uses TensorFlow[1], we requested 8 nodes, 120 GB of RAM, and 40 GB

swap memory on the LFS machine using Python 3.6.

3.4.4 Results.

The average number of selected features and average classification accuracies over 10 independent runs using SVM and RF on the datasets described in Section 3.4.1 are shown in Table 3.3.

The empty spaces in Table 3.3 under LARS, HSIC-Lasso, CCM and BCOA's columns simply mean that these methods do not run on those datasets; this is a major shortfall of these methods and it would be interesting to find out why and to what extent LARS, HSIC-Lasso and BCOA fail to run on a dataset. Since the NeuroX datasets have 267,601 features, CCM method requires 1.5 TB of RAM to execute.

In terms of accuracy using either of SVM or RF, we can see from Table 3.3 that DRPT is at least as good as any of the other seven methods. We can further infer that, in general, SVM has a better performance than RF on these datasets and SVM requires more features than RF to attain the maximum possible accuracy.

In Table 3.4, we report the standard deviation (SD) of the number of selected features and SD of the classification accuracies over 10 independent runs. Lower SDs are clearly desirable, which is also an indication of the method's stability with respect to permutation of rows.

Figure 3.4 shows the average classification accuracy results of our DRPT compared

Table 3.3: Superscript is average number of selected features and subscript is resulting classification accuracies (CA) based on SVM and RF using mRMR, LARS, HSIC-Lasso, Fast-OSFS, group-SAOLA, CCM, BCOA and DRPT over 10 runs.

Classifier	Dataset	(# of selected features)$_{\text{classification accuracy}}$							
		mRMR	**LARS**	**HSIC-Lasso**	**Fast-OSFS**	**group-SAOLA**	**CCM**	**BCOA**	**DRPT**
SVM	GDS1615	$(40.20)_{87.37}$	$(26.60)_{91.67}$	$(18.70)_{91.35}$	$(17.20)_{84.31}$	$(12.40)_{83.13}$	$(29.20)_{80.82}$	$(33.90)_{84.90}$	$(37.00)_{\mathbf{91.89}}$
	GDS3268	$(38.50)_{85.69}$	$(43.50)_{89.62}$	–	$(35.90)_{87.89}$	$(16.60)_{84.13}$	$(38.65)_{85.82}$	$(34.50)_{73.55}$	$(33.90)_{\mathbf{90.45}}$
	GDS968	$(39.80)_{80.87}$	$(38.70)_{\mathbf{83.73}}$	–	$(19.80)_{72.41}$	$(14.10)_{70.53}$	$(34.14)_{78.82}$	$(32.86)_{76.19}$	$(38.30)_{81.06}$
	GDS531	$(30.90)_{69.78}$	$(27.60)_{79.96}$	$(4.00)_{67.93}$	$(26.50)_{77.43}$	$(11.60)_{77.70}$	$(30.45)_{\mathbf{80.82}}$	$(32.67)_{74.17}$	$(25.20)_{77.16}$
	GDS2545	$(34.00)_{75.90}$	$(33.70)_{79.02}$	$(33.8)_{76.40}$	$(18.80)_{74.95}$	$(12.30)_{75.55}$	$(30.11)_{70.82}$	$(29.85)_{75.40}$	$(31.30)_{\mathbf{83.23}}$
	GDS1962	$(39.50)_{65.12}$	$(32.50)_{76.56}$	$(31.5)_{\mathbf{76.81}}$	$(24.60)_{65.15}$	$(10.50)_{66.593}$	$(40.12)_{66.82}$	$(35.45)_{66.89}$	$(37.60)_{72.87}$
	GDS3929	$(41.10)_{73.57}$	$(41.10)_{\mathbf{83.78}}$	–	$(40.20)_{83.11}$	$(21.60)_{76.97}$	$(39.90)_{75.82}$	$(41.20)_{72.12}$	$(37.90)_{78.76}$
	GDS2546	$(33.10)_{74.13}$	$(32.70)_{\mathbf{83.51}}$	$(27.00)_{77.69}$	$(26.40)_{81.25}$	$(17.70)_{80.88}$	$(35.30)_{73.82}$	$(32.50)_{72.98}$	$(32.70)_{81.48}$
	GDS2547	$(39.40)_{67.31}$	$(32.50)_{73.88}$	$(12.3)_{71.16}$	$(23.60)_{73.13}$	$(24.30)_{76.85}$	$(28.40)_{66.82}$	$(26.60)_{67.35}$	$(33.70)_{\mathbf{80.53}}$
	NX100	$(2.00)_{\mathbf{100.00}}$	–	–	$(2.00)_{\mathbf{100.00}}$	$(11.00)_{\mathbf{100.00}}$	–	–	$(21.00)_{\mathbf{100.00}}$
	NX200	$(2.40)_{\mathbf{100.00}}$	–	–	$(2.00)_{\mathbf{100.00}}$	$(2.00)_{\mathbf{100.00}}$	–	–	$(12.00)_{\mathbf{100.00}}$
RF	GDS1615	$(32.70)_{81.96}$	$(20.20)_{88.24}$	$(22.70)_{\mathbf{92.88}}$	$(15.20)_{82.34}$	$(13.00)_{82.26}$	$(31.20)_{79.55}$	$(30.80)_{81.08}$	$(31.20)_{85.46}$
	GDS3268	$(26.50)_{\mathbf{87.26}}$	$(41.70)_{86.52}$	–	$(30.20)_{86.40}$	$(13.60)_{81.19}$	$(34.55)_{82.82}$	$(33.50)_{78.87}$	$(32.60)_{86.15}$
	GDS968	$(44.20)_{79.44}$	$(42.70)_{79.77}$	–	$(19.50)_{72.84}$	$(18.20)_{71.28}$	$(41.30)_{77.53}$	$(40.73)_{76.42}$	$(38.50)_{\mathbf{81.55}}$
	GDS531	$(23.90)_{63.69}$	$(20.70)_{71.44}$	$(4.70)_{67.82}$	$(14.80)_{75.48}$	$(16.40)_{74.67}$	$(23.60)_{\mathbf{77.36}}$	$(21.50)_{73.92}$	$(14.30)_{75.88}$
	GDS2545	$(31.40)_{79.31}$	$(33.10)_{75.81}$	$(33.10)_{80.64}$	$(14.80)_{74.16}$	$(12.00)_{76.05}$	$(34.20)_{74.82}$	$(33.57)_{75.63}$	$(32.60)_{\mathbf{86.78}}$
	GDS1962	$(29.40)_{72.37}$	$(30.80)_{72.41}$	$(42.1)_{\mathbf{78.45}}$	$(21.90)_{69.88}$	$(13.30)_{63.28}$	$(32.20)_{69.17}$	$(30.62)_{67.95}$	$(29.30)_{74.32}$
	GDS3929	$(29.10)_{71.94}$	$(28.60)_{73.44}$	–	$(28.10)_{70.49}$	$(15.90)_{71.56}$	$(28.50)_{67.50}$	$(24.10)_{65.13}$	$(18.90)_{66.60}$
	GDS2546	$(36.30)_{70.53}$	$(34.30)_{75.86}$	$(45.90)_{\mathbf{83.09}}$	$(25.80)_{77.04}$	$(18.20)_{78.46}$	$(36.30)_{72.90}$	$(31.20)_{75.28}$	$(33.30)_{80.31}$
	GDS2547	$(22.40)_{68.44}$	$(24.80)_{71.68}$	$(32.6)_{\mathbf{81.67}}$	$(30.00)_{75.85}$	$(20.50)_{77.10}$	$(25.40)_{69.70}$	$(24.20)_{71.28}$	$(23.20)_{78.95}$
	NX100	$(2.00)_{\mathbf{100.00}}$	–	–	$(2.00)_{\mathbf{100.00}}$	$(2.00)_{\mathbf{100.00}}$	–	–	$(22.00)_{\mathbf{100.00}}$
	NX200	$(2.40)_{\mathbf{100.00}}$	–	–	$(2.00)_{\mathbf{100.00}}$	$(2.00)_{\mathbf{100.00}}$	–	–	$(11.00)_{\mathbf{100.00}}$

to other FS methods using k features and the SVM classifier, where k is between 10 and 90. When a FS method returns a subset of k features, we use SVM to find an optimal $t \leq k$ so that the first t features yield the best accuracy. Note that we do not

Table 3.4: Superscript is SD of # selected features and subscript is the SD of resulting classification accuracies (CA) based on SVM and RF using mRMR, LARS, HSIC-Lasso, Fast-OSFS, group-SAOLA, CCM, BCOA and DRPT over 10 runs.

Classifier	Dataset	(SD of selected features)$_{\text{SD of CA}}$							
		mRMR	**LARS**	**HSIC-Lasso**	**Fast-OSFS**	**group-SAOLA**	**CCM**	**BCOA**	**DRPT**
SVM	GDS1615	$(7.87)_{4.29}$	$(16.54)_{4.95}$	$(8.38)_{4.20}$	$(6.18)_{4.79}$	$(6.50)_{6.60}$	$(9.35)_{5.84}$	$(13.60)_{7.43}$	$(7.05)_{\mathbf{3.61}}$
	GDS3268	$(9.35)_{3.58}$	$(5.28)_{3.31}$	–	$(8.99)_{4.34}$	$(3.66)_{3.47}$	$(6.31)_{4.43}$	$(4.72)_{5.64}$	$(9.49)_{\mathbf{1.83}}$
	GDS968	$(6.20)_{5.12}$	$(8.12)_{4.96}$	–	$(4.15)_{4.44}$	$(3.78)_{6.53}$	$(6.60)_{6.76}$	$(6.50)_{4.12}$	$(7.07)_{\mathbf{4.79}}$
	GDS531	$(17.46)_{4.39}$	$(12.89)_{4.79}$	$(1.45)_{5.74}$	$(9.91)_{6.30}$	$(4.77)_{5.71}$	$(14.05)_{4.79}$	$(14.46)_{3.61}$	$(10.89)_{\mathbf{3.02}}$
	GDS2545	$(13.14)_{3.37}$	$(13.71)_{3.04}$	$(12.13)_{\mathbf{2.66}}$	$(9.33)_{6.44}$	$(6.00)_{6.93}$	$(12.87)_{5.27}$	$(11.08)_{4.97}$	$(8.97)_{2.79}$
	GDS1962	$(11.03)_{2.91}$	$(11.55)_{3.68}$	$(15.54)_{4.03}$	$(14.04)_{7.01}$	$(4.40)_{6.00}$	$(14.15)_{3.84}$	$(11.04)_{3.57}$	$(10.50)_{\mathbf{2.89}}$
	GDS3929	$(10.51)_{4.64}$	$(11.12)_{3.32}$	–	$(8.87)_{3.69}$	$(5.93)_{5.22}$	$(13.84)_{4.19}$	$(15.53)_{\mathbf{3.13}}$	$(9.44)_{3.65}$
	GDS2546	$(7.96)_{\mathbf{2.12}}$	$(10.89)_{4.65}$	$(13.41)_{3.10}$	$(2.72)_{4.57}$	$(5.81)_{4.02}$	$(7.76)_{4.87}$	$(5.25)_{6.82}$	$(10.67)_{4.24}$
	GDS2547	$(9.36)_{4.42}$	$(9.98)_{3.77}$	$(7.86)_{4.61}$	$(13.57)_{4.95}$	$(8.60)_{4.64}$	$(8.38)_{5.29}$	$(10.23)_{7.53}$	$(9.48)_{\mathbf{3.18}}$
	NX100	$(00.00)_{\mathbf{00.00}}$	–	–	$(00.00)_{\mathbf{00.00}}$	$(00.00)_{\mathbf{00.00}}$	–	–	$(3.10)_{\mathbf{00.00}}$
	NX200	$(00.71)_{\mathbf{00.00}}$	–	–	$(00.00)_{\mathbf{00.00}}$	$(00.00)_{\mathbf{00.00}}$	–	–	$(2.00)_{\mathbf{00.00}}$
RF	GDS1615	$(9.64)_{4.63}$	$(9.27)_{3.44}$	$(7.23)_{\mathbf{2.25}}$	$(5.33)_{3.95}$	$(6.00)_{4.08}$	$(10.60)_{5.19}$	$(9.60)_{5.29}$	$(9.26)_{3.12}$
	GDS3268	$(10.24)_{\mathbf{2.54}}$	$(6.33)_{2.46}$	–	$(8.27)_{5.40}$	$(5.54)_{2.86}$	$(7.95)_{5.81}$	$(9.04)_{4.13}$	$(8.37)_{2.95}$
	GDS968	$(4.69)_{4.01}$	$(6.85)_{4.10}$	–	$(5.23)_{5.13}$	$(5.34)_{4.89}$	$(5.64)_{4.43}$	$(4.22)_{4.03}$	$(5.26)_{\mathbf{3.90}}$
	GDS531	$(10.34)_{3.98}$	$(17.76)_{5.36}$	$(2.11)_{6.57}$	$(7.32)_{9.07}$	$(6.98)_{6.90}$	$(9.60)_{6.72}$	$(11.37)_{3.63}$	$(7.38)_{\mathbf{2.86}}$
	GDS2545	$(14.60)_{2.87}$	$(12.08)_{3.28}$	$(15.30)_{\mathbf{2.21}}$	$(6.86)_{6.63}$	$(5.27)_{5.61}$	$(10.60)_{3.94}$	$(9.98)_{3.71}$	$(9.37)_{2.74}$
	GDS1962	$(12.00)_{3.20}$	$(15.46)_{\mathbf{1.82}}$	$(7.11)_{2.88}$	$(7.29)_{3.54}$	$(4.87)_{5.48}$	$(13.52)_{4.82}$	$(10.06)_{5.11}$	$(11.33)_{4.11}$
	GDS3929	$(11.86)_{2.94}$	$(16.34)_{3.37}$	–	$(10.84)_{5.31}$	$(8.94)_{4.14}$	$(12.60)_{3.96}$	$(9.60)_{4.49}$	$(8.83)_{\mathbf{2.48}}$
	GDS2546	$(11.14)_{3.59}$	$(12.64)_{3.80}$	$(2.73)_{3.33}$	$(9.39)_{2.36}$	$(4.02)_{5.25}$	$(8.73)_{4.17}$	$(10.23)_{4.62}$	$(8.11)_{\mathbf{3.15}}$
	GDS2547	$(15.31)_{4.42}$	$(10.06)_{3.83}$	$(11.19)_{4.16}$	$(7.63)_{4.58}$	$(6.15)_{6.03}$	$(10.28)_{5.19}$	$(11.93)_{4.86}$	$(9.33)_{\mathbf{3.25}}$
	NX100	$(00.00)_{\mathbf{00.00}}$	–	–	$(00.00)_{\mathbf{00.00}}$	$(00.00)_{\mathbf{00.00}}$	–	–	$(2.20)_{\mathbf{00.00}}$
	NX200	$(00.00)_{\mathbf{00.00}}$	–	–	$(00.00)_{\mathbf{00.00}}$	$(00.00)_{\mathbf{00.00}}$	–	–	$(2.30)_{\mathbf{00.00}}$

look for the best subset and rather add the features sequentially to find the optimal t. Then we calculate the accuracy using the first t features and take the average of these

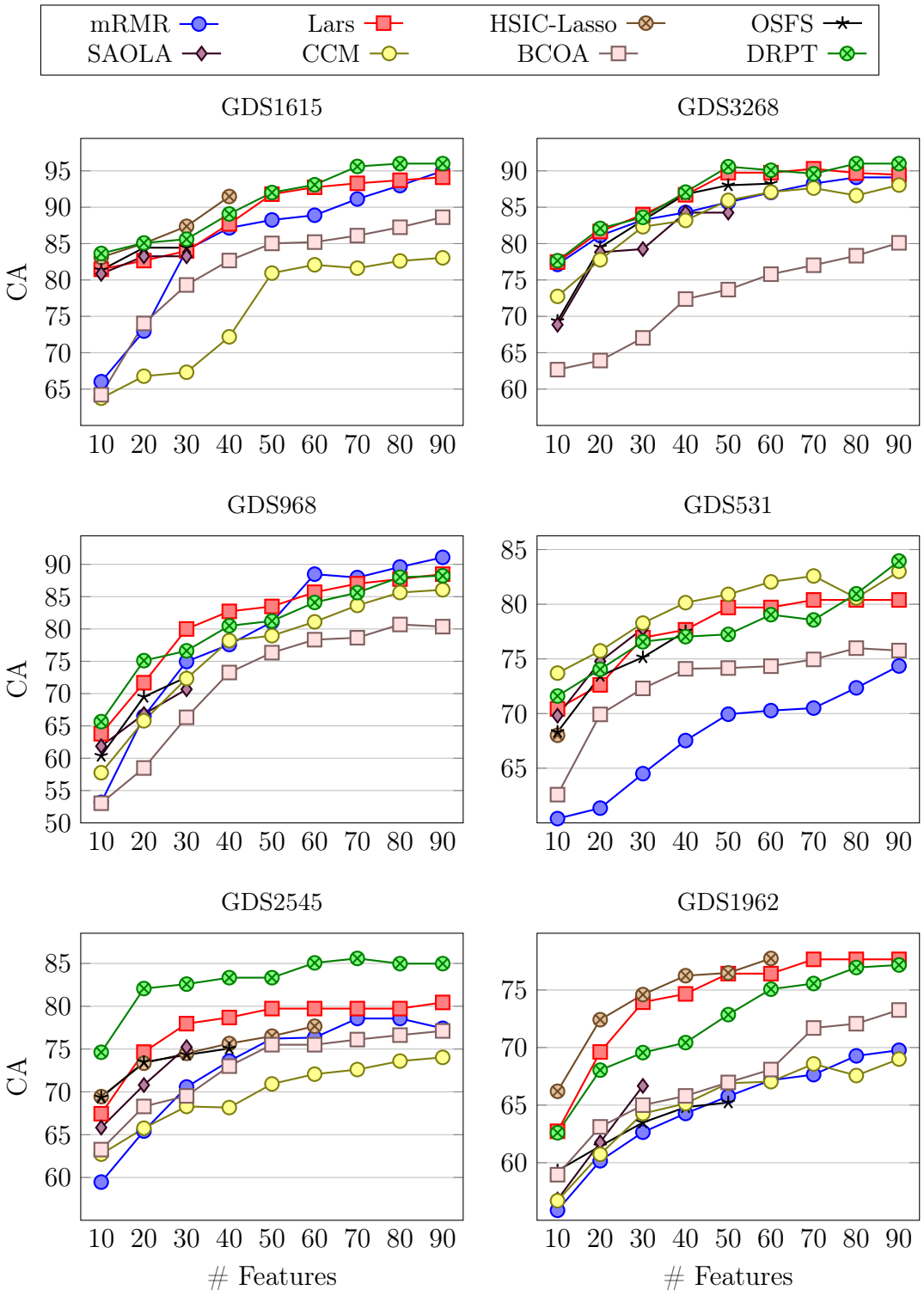
accuracies over 10 runs. Our evaluation metric is consistent across all FS methods.

Figure 3.4 shows the general superiority of the classification accuracy of our proposed model over the other models for the 9 genomic datasets used in our study. We can see a steady increase in classification accuracies of different FS methods as we increase k from 10 to 50, however the curves usually flat out when k is between 50 and 90. Note that HSIC-Lasso, Fast-OSFS, and group-SAOLA models output a subset of fewer than 60 features.

We note that the default number of selected features by LARS is almost the number of samples in a dataset. In Table 3.5, we perform a further comparison between LARS and DRPT where we set k to be the default number of features suggested by LARS and we use the classifier to find an optimal subset of size at most k. For example, the dataset GDS1615, has 127 samples in total. Since we take approximately 70% of samples for FS, the suggested number of features by LARS is $k = 87$.

If we look at the performance of LARS just based on its default number of features, we note that CA of LARS significantly drops. This, in particular, suggests that LARS does not select an optimal subset of features.

We also take advantage of IBM®LSF to report running time, CPU time and memory usage of each FS method. We just remark that through parallelization, an algorithm might achieve a better running time at the cost of having greater CPU time.


mRMR
Lars
HSIC-Lasso
OSFS
SAOLA
CCM
BCOA
DRPT
GDS1615
GDS3268
GDS968
GDS531
GDS2545
GDS1962
CA
Features

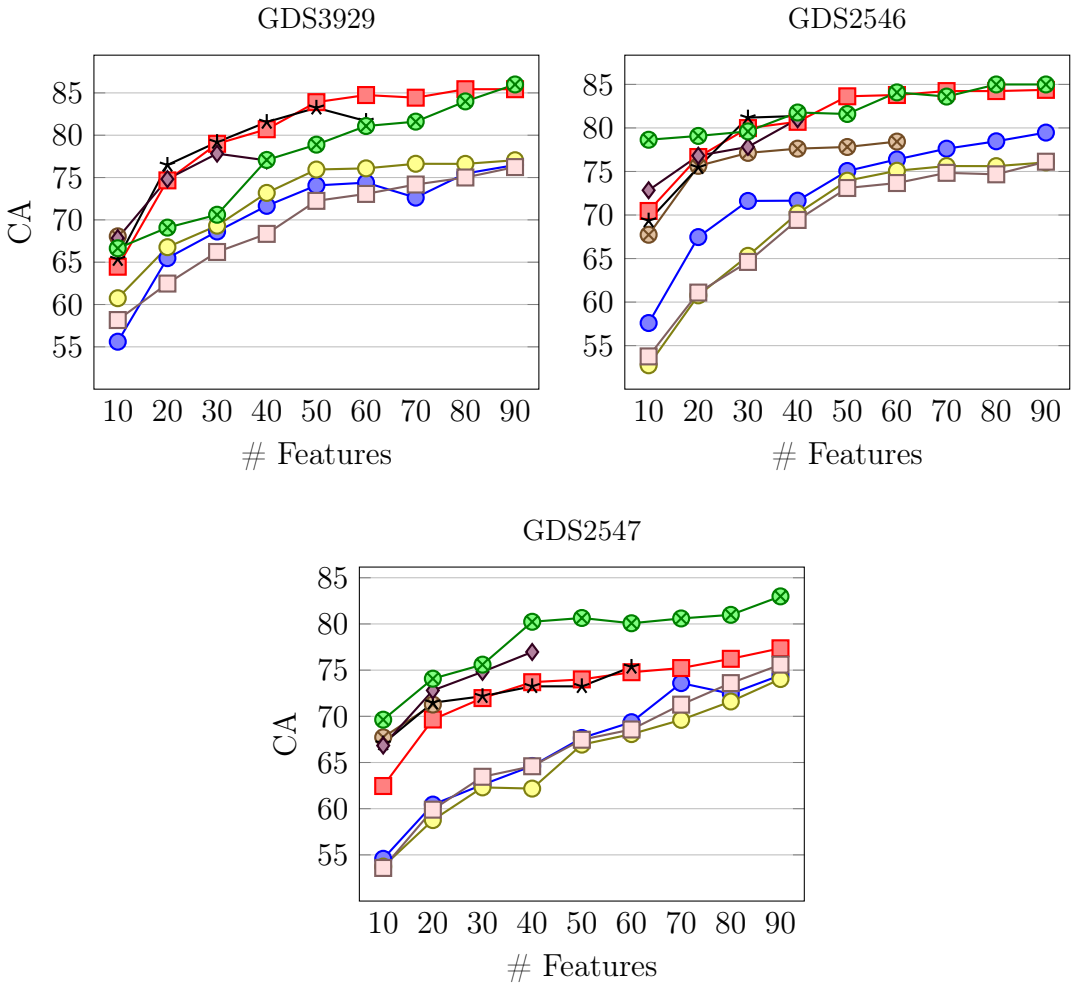


Figure 3.4: Classification accuracies (CA) based on SVM using mRMR, LARS, HSIC-Lasso, Fast-OSFS, group-SAOLA, CCM, BCOA and DRPT over 10 runs for different number of features

Table 3.5: Superscript is average number of selected features and subscript is resulting classification accuracies (CA) based on SVM and RF using LARS Suggestion (LS) for 10 independent runs of DRPT and LARS.

Classifier	Dataset	(# of selected features)$_{\text{classification accuracy}}$		
		# of LS	**DRPT**	**LARS**
SVM	GDS1615	87	$(69.90)_{95.89}$	$(62.2)_{93.99}$
	GDS3268	140	$(94.50)_{95.13}$	$(103.30)_{95.15}$
	GDS968	118	$(90.50)_{86.93}$	$(68.30)_{87.20}$
	GDS531	120	$(92.80)_{83.79}$	$(51.15)_{81.07}$
	GDS2545	118	$(63.00)_{84.33}$	$(58.70)_{80.28}$
	GDS1962	124	$(75.47)_{75.19}$	$(44.3)_{77.43}$
	GDS3929	127	$(93.57)_{89.75}$	$(78.70)_{86.77}$
	GDS2546	115	$(89.12)_{85.92}$	$(61.90)_{84.31}$
	GDS2547	113	$(83.94)_{85.14}$	$(67.30)_{77.67}$
RF	GDS1615	87	$(57.76)_{89.83}$	$(54.00)_{90.23}$
	GDS3268	140	$(84.4)_{89.35}$	$(87.20)_{90.67}$
	GDS968	118	$(91.30)_{87.20}$	$(85.30)_{85.14}$
	GDS531	120	$(26.50)_{76.23}$	$(25.60)_{75.54}$
	GDS2545	118	$(73.94)_{89.72}$	$(75.50)_{78.65}$
	GDS1962	124	$(89.75)_{75.00}$	$(70.90)_{74.68}$
	GDS3929	127	$(52.10)_{70.51}$	$(22.90)_{74.32}$
	GDS2546	115	$(47.80)_{82.49}$	$(53.10)_{77.93}$
	GDS2547	113	$(53.80)_{78.86}$	$(28.00)_{73.87}$

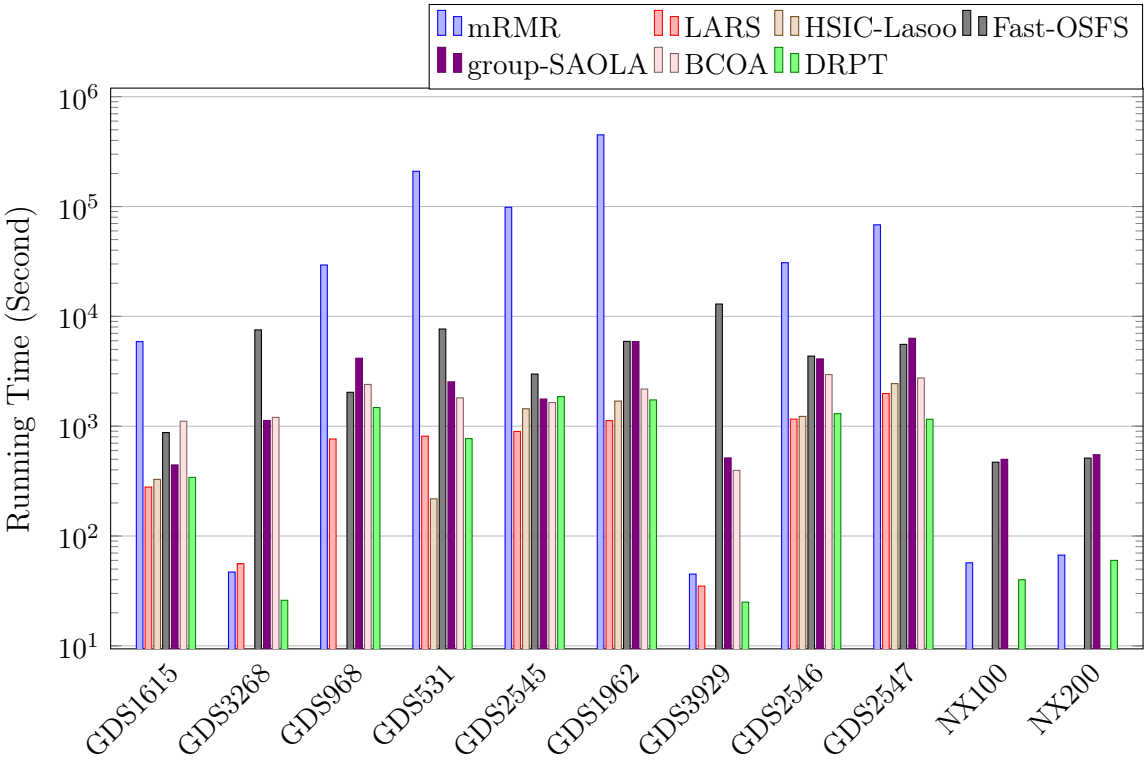
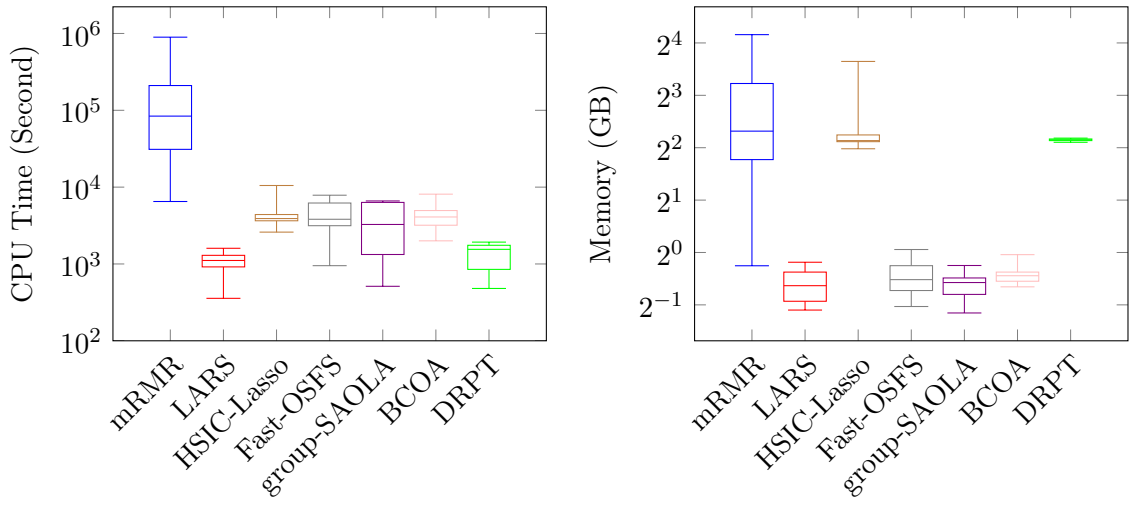


Figure 3.5: Running Time of feature selection by DRPT, HSIC-Lasso, LARS, Fast-OSFS, group-SAOLA, and mRMR over 10 runs using SVM

Figure 3.5 depicts running time of FS methods that includes the classification time using SVM as well. We can see that LARS, HSIC-Lasso, and DRPT have comparable running times. The running times of Fast-OSFS, group-SAOLA and BCOA are higher than DRPT while the running time of mRMR is the worst among all by order of magnitude.

Next, we compare CPU time. For a non-parallelized algorithm, the CPU time is almost the as same as the running time. However, a parallelized algorithm takes more CPU time as it hires multi-processes. Figure 3.6(a) shows the CPU time that is taken



(a) (b)

Figure 3.6: (a) CPU Time and (b) Memory taken by DRPT, HSIC-Lasso , LARS, Fast-OSFS, group-SAOLA, and mRMR

by FS methods on six common datasets. Clearly, mRMR takes the highest CPU time and it is also obvious that HSIC-Lasso uses more processes as it is implemented in parallel.

We also quantify the computational performance of all methods based on the peak memory usage over six common datasets (Figure 3.6(b)). We observe that mRMR and HSIC-Lasso require an order of magnitude higher memory than LARS. Although the peak memory usage by DRPT is significantly lower than mRMR and HSIC-Lasso, DRPT takes almost the same amount of memory across all datasets. In this regard, there is a potential for more efficient implementation of DRPT.

We had to leave the CCM method out of the comparison in Figures 3.5 and 3.6

Table 3.6: Running time, CPU time and memory taken by CCM model

Dataset	Running Time	CPU Time	Memory
GDS1615	850	26855	107
GDS531	3327	36193	150
GDS2545	1478	36009	74
GDS1962	3621	38730	148
GDS2546	1389	35331	73
GDS2547	985	30500	71

because it is implemented in Python and required a high volume of RAM while the other methods implemented in Matlab. Table 3.6 shows the CCM performance in terms of running time, CPU time, and memory usage, where running time and CPU time are measured by second and memory usage is scaled in GB.

3.5 Conclusions

In this chapter, we presented a linear feature selection method (DRPT) for high-dimensional genomic datasets. The novelty of our method is to remove irrelevant features outright and then detect correlations on the reduced dataset using perturbation theory. While we showed DRPT precisely detects irrelevant and redundant features on a synthetic dataset, the extent to which DRPT is effective on real dataset was tested on ten genomic datasets. We demonstrated that DRPT performs well on

these datasets compared to state-of-the-art feature selection algorithms. We proved that DRPT is robust against noise. Performance of DRPT is insensitive to permutation of rows or columns of the data. Even though the running time of DRPT is comparable to other FS methods, an efficient implementation of DRPT in Python or C++ can help improve both memory usage and the running time.

In this chapter, we focused only on genomic datasets because inherently they are similar. For example, they all have full-row rank. Besides, it is widely accepted that there is no dimension reduction algorithm that performs well on all datasets (compared to other methods). In a future work, we aim to revise our current algorithm to offer a new FS algorithm that performs well on face and text datasets.

Bibliography

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)* (2016), pp. 265–283.

[2] Afshar, M., and Usefi, H. High-dimensional feature selection for genomic datasets. *Knowledge-Based Systems 206* (2020), 106370.

[3] Ali, L., Zhu, C., Zhou, M., and Liu, Y. Early diagnosis of parkinson's disease from multiple voice recordings by simultaneous sample and feature selection.

Expert Systems with Applications 137 (2019), 22–28.

[4] Amrhein, V., Greenland, S., and McShane, B. Scientists rise up against statistical significance, 2019.

[5] Armanfard, N., Reilly, J. P., and Komeili, M. Local feature selection for data classification. *IEEE transactions on pattern analysis and machine intelligence 38*, 6 (2015), 1217–1227.

[6] Ashour, A. S., Nour, M. K. A., Polat, K., Guo, Y., Alsaggaf, W., and El-Attar, A. A novel framework of two successive feature selection levels using weight-based procedure for voice-loss detection in parkinson's disease. *IEEE Access 8* (2020), 76193–76203.

[7] Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillippy, K. H., Sherman, P. M., Holko, M., et al. Ncbi geo: archive for functional genomics data sets—update. *Nucleic acids research 41*, D1 (2012), D991–D995.

[8] Ben-Israel, A., and Greville, T. N. *Generalized inverses: theory and applications*, vol. 15. Springer Science & Business Media, 2003.

[9] Breiman, L., Friedman, J., Olshen, R., et al. Classification and regression trees routledge.

[10] Chaudhari, P., and Agarwal, H. Improving feature selection using elite breeding qpso on gene data set for cancer classification. In *Intelligent Engineering Informatics*. Springer, 2018, pp. 209–219.

[11] Chen, C., Qiao, R., Wei, R., Guo, Y., Ai, H., Ma, J., Ren, J., and Huang, L. A comprehensive survey of copy number variation in 18 diverse pig populations and identification of candidate copy number variable genes associated with complex traits. *BMC genomics 13*, 1 (2012), 1–10.

[12] Chen, J., Stern, M., Wainwright, M. J., and Jordan, M. I. Kernel feature selection via conditional covariance minimization. *arXiv preprint arXiv:1707.01164* (2017).

[13] Chen, X., Yuan, G., Nie, F., and Huang, J. Z. Semi-supervised feature selection via rescaled linear regression. In *IJCAI* (2017), vol. 2017, pp. 1525–1531.

[14] de Souza, R. C. T., de Macedo, C. A., dos Santos Coelho, L., Pierezan, J., and Mariani, V. C. Binary coyote optimization algorithm for feature selection. *Pattern Recognition 107* (2020), 107470.

[15] Ding, C., and Peng, H. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology 3*, 02 (2005), 185–205.

[16] Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., Marquéz, J. R. G., Gruber, B., Lafourcade, B., Leitao, P. J., et al. Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography 36*, 1 (2013), 27–46.

[17] Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. Least angle regression. *Annals of statistics 32*, 2 (2004), 407–499.

[18] Farazi, P. A., and DePinho, R. A. Hepatocellular carcinoma pathogenesis: from genes to environment. *Nature Reviews Cancer 6*, 9 (2006), 674–687.

[19] Gaudioso, M., Gorgone, E., Labbé, M., and Rodríguez-Chía, A. M. Lagrangian relaxation for svm feature selection. *Computers & Operations Research 87* (2017), 137–145.

[20] Gutkin, M., Shamir, R., and Dror, G. Slimpls: a method for feature selection in gene expression-based disease classification. *PloS one 4*, 7 (2009), e6416.

[21] Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *Journal of machine learning research 3*, Mar (2003), 1157–1182.

[22] John, G. H., Kohavi, R., and Pfleger, K. Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 121–129.

[23] Khalid, S., Khalil, T., and Nasreen, S. A survey of feature selection and feature extraction techniques in machine learning. In *2014 science and information conference* (2014), IEEE, pp. 372–378.

[24] Kim, M., Won, J. H., Hong, J., Kwon, J., Park, H., and Shen, L. Deep network-based feature selection for imaging genetics: Application to identifying biomarkers for parkinson's disease. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)* (2020), IEEE, pp. 1920–1923.

[25] Kohavi, R., and John, G. H. Wrappers for feature subset selection. *Artificial intelligence 97*, 1-2 (1997), 273–324.

[26] Liu, S., Xu, C., Zhang, Y., Liu, J., Yu, B., Liu, X., and Dehmer, M. Feature selection of gene expression data for cancer classification using double rbf-kernels. *BMC bioinformatics 19*, 1 (2018), 1–14.

[27] Lohmueller, K. E., Pearce, C. L., Pike, M., Lander, E. S., and Hirschhorn, J. N. Meta-analysis of genetic association studies supports a contribution of common variants to susceptibility to common disease. *Nature genetics 33*, 2 (2003), 177–182.

[28] Nasrabadi, N. M. Pattern recognition and machine learning. *Journal of electronic imaging 16*, 4 (2007), 049901.

[29] Nogueira, S., Sechidis, K., and Brown, G. On the stability of feature selection algorithms. *J. Mach. Learn. Res. 18*, 1 (2017), 6345–6398.

[30] Paul, S., and Das, S. Simultaneous feature selection and weighting–an evolutionary multi-objective optimization approach. *Pattern Recognition Letters 65* (2015), 51–59.

[31] Peng, H., Long, F., and Ding, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence 27*, 8 (2005), 1226–1238.

[32] Pierezan, J., and Coelho, L. D. S. Coyote optimization algorithm: a new metaheuristic for global optimization problems. In *2018 IEEE congress on evolutionary computation (CEC)* (2018), IEEE, pp. 1–8.

[33] Schafer, R. What is a savitzky-golay filter, ieee signal process. *Mag 28* (2011), 111–7.

[34] Sharma, P., Sundaram, S., Sharma, M., Sharma, A., and Gupta, D. Diagnosis of parkinson's disease using modified grey wolf optimization. *Cognitive Systems Research 54* (2019), 100–115.

[35] Sjöstrand, K. Matlab implementation of lasso, lars, the elastic net and spca.

[36] Sørlie, T., Perou, C. M., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., Hastie, T., Eisen, M. B., Van De Rijn, M., Jeffrey, S. S., et al. Gene expression patterns of breast carcinomas distinguish tumor

subclasses with clinical implications. *Proceedings of the National Academy of Sciences 98*, 19 (2001), 10869–10874.

[37] Sun, Y., Todorovic, S., and Goodison, S. Local-learning-based feature selection for high-dimensional data analysis. *IEEE transactions on pattern analysis and machine intelligence 32*, 9 (2009), 1610–1626.

[38] Tamura, R., Kobayashi, K., Takano, Y., Miyashiro, R., Nakata, K., and Matsui, T. Best subset selection for eliminating multicollinearity. *Journal of the Operations Research Society of Japan 60*, 3 (2017), 321–336.

[39] Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological) 58*, 1 (1996), 267–288.

[40] Wang, Z., Li, M., and Li, J. A multi-objective evolutionary algorithm for feature selection based on mutual information with a new redundancy measure. *Information Sciences 307* (2015), 73–88.

[41] Wasserstein, R. L., Schirm, A. L., and Lazar, N. A. Moving to a world beyond "p< 0.05", 2019.

[42] Wu, X., Yu, K., Ding, W., Wang, H., and Zhu, X. Online feature selection with streaming features. *IEEE transactions on pattern analysis and machine intelligence 35*, 5 (2012), 1178–1192.

[43] Wu, X., Yu, K., Wang, H., and Ding, W. Online streaming feature selection. In *ICML* (2010).

[44] Xue, B., Zhang, M., Browne, W. N., and Yao, X. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation 20*, 4 (2015), 606–626.

[45] Xue, Y., Zhang, L., Wang, B., Zhang, Z., and Li, F. Nonlinear feature selection using gaussian kernel svm-rfe for fault diagnosis. *Applied Intelligence 48*, 10 (2018), 3306–3331.

[46] Yamada, M., Jitkrittum, W., Sigal, L., Xing, E. P., and Sugiyama, M. High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation 26*, 1 (2014), 185–207.

[47] Yamada, M., Tang, J., Lugo-Martinez, J., Hodzic, E., Shrestha, R., Saha, A., Ouyang, H., Yin, D., Mamitsuka, H., Sahinalp, C., et al. Ultra high-dimensional nonlinear feature selection for big biological data. *IEEE Transactions on Knowledge and Data Engineering 30*, 7 (2018), 1352–1365.

[48] Yaman, O., Ertam, F., and Tuncer, T. Automated parkinson's disease recognition based on statistical pooling method using acoustic features. *Medical hypotheses 135* (2020), 109483.

[49] Yang, Y., and Pedersen, J. O. A comparative study on feature selection in text categorization. In *Icml* (1997), vol. 97, Nashville, TN, USA, p. 35.

[50] Yu, K., Ding, W., and Wu, X. Lofs: a library of online streaming feature selection. *Knowledge-Based Systems 113* (2016), 1–3.

[51] Yu, K., Wu, X., Ding, W., and Pei, J. Towards scalable and accurate online feature selection for big data. In *2014 IEEE International Conference on Data Mining* (2014), IEEE, pp. 660–669.

[52] Yu, K., Wu, X., Ding, W., and Pei, J. Scalable and accurate online feature selection for big data. *ACM Transactions on Knowledge Discovery from Data (TKDD) 11*, 2 (2016), 1–39.

[53] Zhou, J., Foster, D. P., Stine, R. A., and Ungar, L. H. Streamwise feature selection. *Journal of Machine Learning Research* (2006), 1861–1885.

[54] Zhou, X., and Mao, K. Ls bound based gene selection for dna microarray data. *Bioinformatics 21*, 8 (2005), 1559–1564.

Chapter 4

Dimensionality Reduction Using Singular Vectors

(This chapter is based on a paper published in Scientific Reports-Nature, 2021 [2])

4.1 Introduction

With the extraordinary advancements in high throughput gene expression profiling and DNA sequencing technologies, we are presented with the challenge of interpreting high-dimensional datasets. Nonetheless, this presents an opportunity for discovery of biological biomarkers that in turn can help for early detection of disease [17] and identification of predictive and prognostic factors in disease management [21]. Genome-wide association studies (GWAS) can be performed on single-nucleotide polymorphism (SNP) arrays to identifying associations between loci and traits. Even

though GWAS are proved to be useful [44], there are some drawbacks as well. GWAS identifies loci so that each locus is statically significant (on its own). However, complex diseases are extremely polygenic and it therefore important to identify a subset of SNPs or genes that cumulatively explain the disease. Furthermore, most GWA studies require thousands of samples which can pose as a significant challenge.

Feature selection (FS) is another alternative for biomarker discovery. FS involves filtering and determining the relevant features from numerous irrelevant and redundant features, so FS can decrease the learning costs and improve the classification performance in many applications such as genomic data and remote sensing by turning the high-dimensional data into a lower dimension [30]. Features can be embedded into a lower-dimensional subspace in which different patterns appear to be considerably distinct with lower cost [33]. The importance of using FS methods on genomic data to supplement and improve the process of disease diagnosis is gaining increasing attention [23, 11, 19, 12]. Hikichi et al. [22] applied a correlation-centered approach and proposed a set of 12 predictive genes to diagnose cancer metastasis; their selected genes showed higher performance compared to the 76 genes previously reported by Wang et al. [48]. Recently, Jiang et al. [24] applied a hybrid FS method for analyzing Endometrial Cancer data. In another study [39], the authors focused on colon cancer and applied a hybrid FS method to obtain the optimal subset of genes using two independent datasets. Among 17,814 genes in the original dataset, 6 top relevant genes were selected in two phases. An independent dataset of colon cancer was

used to validate the selected genes, resulting in 99.9% classification accuracy. Shukla et al. [41] present a gene expression analysis on lymphoma cancer using several FS methods. Their experimental results showed that the highest classification accuracy is achieved using the top 20 selected genes. In a recent study, Sun et al. [43] worked on high-dimensional microarray datasets and filtered data using the ReliefF method [26] to reduce the dimensionality of gene expression data and then applied a modified Ant Colony Optimization algorithm [53] to find the optimal subset of genes for colon, leukemia, lung, prostate, and brain cancers.

In this chapter, we propose a new FS method based on singular vectors (SVFS). Let $D = [A \mid \mathbf{b}]$ be a dataset, where A is an $m \times n$ matrix with m instances and n features, and $\mathbf{b}$ is the class label. We define the signature matrix S_A of A by setting $S_A = I - A^{\dagger}A$, where $A^{\dagger}$ is the pseudo-inverse of A. We introduce a two-step irrelevant features filtering that maps the given dataset into a lower-dimensional subspace that includes less noisy and more informative features. Using the signature matrix S_A, features that have correlations to each other are clustered. The most important features are then picked from each cluster. This process can be optimized using two thresholds to make our model capable of handling a wide range of high dimensional data types. We view the data and interactions between all features globally in the sense that we measure the relevancy of features to $\mathbf{b}$ all at once and then breakdown the original feature space into a collection of lower dimensional subspaces. In contrast, many FS methods apply one or two discriminative concepts locally and

at the individual feature level to obtain the most important features. Thus, they may perform well on some types of datasets and have inferior performances on other types of datasets. For example, as we shall see in Section 4.4, Fisher score [14] and Trace ratio criterion [36] have good performances on biological benchmark datasets while they produce weak results on the image benchmark datasets.

We show in Section 4.3, that S_A is the same as the orthogonal projection P onto the null space of A; hence S or P can be constructed using right singular vectors. We define a graph G where the nodes are columns of A and there is an edge between columns $\mathbf{F}_i$ and $\mathbf{F}_j$ if and only if $S_{i,j} \neq 0$. As we shall explain, each connected component of G corresponds to a subset of columns of A that are linearly dependent. In other words, the correlations between columns of A are encoded in the signature matrix S_A.

We view D as a matrix and form the signature matrix $S_D = I - D^\dagger D$. The cluster of D containing $\mathbf{b}$ consists of relevant features to $\mathbf{b}$ and all features in the other clusters are considered irrelevant. After removing irrelevant features, we update A and use the graph associated to S_A to find the clusters. There are many efficient algorithms to find the clusters of a graph. We use Breadth-First Search (BFS) [6] to find the features which are directly or indirectly connected to the other features. The novelty of our method is to use the signature matrix S_D of D to detect and remove irrelevant features and then use the signature matrix S_A of the reduced matrix A to partition the columns of A into clusters so that columns within a cluster correlate

only with columns within the same cluster. Finally, we rank the features in a cluster based on the entries on the main diagonal of S_A and select a small subset of top ranked features with the highest Mutual Information (MI) with respect to $\mathbf{b}$.

In order to evaluate the performance and efficiency of our method, we compare it with the state-of-the-art FS methods, namely Conditional Infomax Feature Extraction (CIFE) [29], Joint Mutual Information (JMI) [52], Fisher score [14], Trace ratio criterion [36], Least angle regression (LARS) [15], Hilbert-Schmidt independence criterion least absolute shrinkage and selection operator (HSIC-Lasso)[50], Conditional Covariance Minimization (CCM) [10], and Sparse Multinomial Naive Bayes (SMNB) [3] on a series of high dimensional benchmark as well as biological datasets.

The rest of this chapter is structured as follows. An overview of the existing FS approaches is given in section 4.2. Then, in Section 4.3, we give a theoretical background along with some examples on synthetic data to show how our method removes irrelevant features and finds correlations between the rest of the features using the signature matrix S. Section 4.4 gives an account on specifications of the datasets and reports our experiment results. Finally, we provide a summary in Section 4.5.

4.2 Related work

FS methods are categorized as filter, wrapper, and embedded methods [25]. The filter methods use some underlying and intrinsic properties of the features measured via

univariate statistics, while the wrapper methods measure the importance of features based on the classifier performances. While optimizing the classifier performance is the essential goal of FS, and the wrapper methods have their own efficient internal classifiers, these methods are computationally more expensive in comparison with the filter methods due to the iterated learning steps of the wrapper methods and their cross-validation to avoid the risk of overfitting the model. The embedded methods are similar to the wrapper methods; however, the former mainly uses an intrinsic model building metric during the learning process.

Many FS algorithms work based on information-theoretical approaches which utilize various criteria to measure and rank the importance of features. The basic idea behind many information-theoretic methods is to maximize feature relevance and minimize feature redundancy [14]. Since feature correlation with class labels normally measures the relevance of the feature, most algorithms in this group are applied in a supervised manner. A brief introduction to basic information-theoretic concepts is given here.

Shannon entropy, as the primary measurement in information-theoretical approaches, measures the uncertainty of a discrete random variable. The entropy of a discrete random variable X is described as below:

$$H(X) = -\sum_{x_i \in X} P(x_i) log(P(x_i)),$$

where x_i is a specific value of X and $P(x_i)$ refers to the probability of x_i over all values of X.

Second concept is the conditional entropy of X and Y, which is another discrete random variable, defined as follows:

$$H(X|Y) = -\sum_{y_i \in Y} P(y_i) \sum_{x_i \in X} P(x_i|y_i) log(P(x_i|y_i))$$

where $P(y_i)$ is the prior probability of y_i, $P(x_i|y_j)$ refers to the conditional probability of x_i and y_j.

To measure the amount of information shared between X and Y, MI or information gain is used, which is defined as follows:

$$I(X;Y) = H(X) - H(X|Y) = \sum_{x_i \in X} \sum_{y_i \in Y} P(x_i, y_i) log(\frac{P(x_i, y_i)}{P(x_i)P(y_i)})$$

where $P(x_i, y_j)$ is the joint probability of x_i and y_j. MI is symmetric such that $I(X;Y) = I(Y;X)$ and in case X and Y are independent, their MI would is zero. Since we applied the MI concept in our proposed method, two representative algorithms of information-theoretical based family are selected for comparison, including Conditional Infomax Feature Extraction (CIFE) [29], Joint Mutual Information (JMI) [52].

Several studies including CIFE[29] and [16, 20] are based on the idea that the conditional redundancy between unselected features and selected features given class labels should be maximized rather than minimizing the feature redundancy. Minimum Redundancy Maximum Relevance (MRMR) reduces feature redundancy in the feature selection process. In contrast, JMI [52, 35] is introduced to increase the MI that is distributed between selected features and unselected features. There have been some improvements of JMI, see [8].

Another category of FS methods is the similarity-based approaches that measure the feature relevances by their ability to preserve data similarities. The two superior similarity-based methods, i.e. the Fisher score [14] and Trace Ratio criterion [36] are selected to provide a basis for comparison with our proposed method.

Fisher score is a supervised feature selection method that explores features with high discriminant capacity. For sample points in different classes, Fisher score aims to maximize distances between samples; in contrast, it minimizes the distances between sample points in the same class. Trace Ratio criterion has the same idea of maximizing data similarity between-class of instances, while minimizing data similarity the within-class of instances. It computes a Trace Ratio norm by building two affinity matrices S_w and S_b to designate within-class and between-class data similarity.

Some approaches use aggregated sample data to select and rank the features [38, 45, 15, 50]. The least absolute shrinkage and selection operator (LASSO) is an estimation method in linear methods that performs two main tasks: regularization and feature selection. For the first task, it calculates the sum of the absolute values of the model parameters, and the sum must be less than a prefixed upper bound. Therefore, by applying a regularization (shrinking) process, it penalizes the coefficients of the regression variables shrinking, some of them are set to zero. For the second task, the features that still have a non-zero coefficient after the regularization process are chosen to be part of the model. The goal of this process is to lessen the prediction error.

Least angle regression (LARS) proposed by Efron et al. [15] works based on LASSO and is a linear regression method that computes all least absolute shrinkage and selection operator [45] estimates and selects those features which are highly correlated to the already selected ones. Yamada et al. in [50] proposed a nonlinear FS method for high-dimensional datasets called Hilbert-Schmidt independence criterion least absolute shrinkage and selection operator (HSIC-Lasso). By solving a Lasso problem and using a set of kernel functions, HSIC-Lasso selects informative non-redundant features. In another work [51] called Least Angle Nonlinear Distributed (LAND), the authors have improved the computational power of the HSIC-Lasso. They illustrated through comprehensive examinations that LAND and HSIC-Lasso achieve comparable classification accuracies and dimension reduction. However, LAND has the advantage that it can be developed on parallel distributed computing.

HSIC-Lasso and LAND are based on a convex optimization problem with a ℓ_1-norm penalty on the regression coefficients to improve sparsity while having a significantly high computational cost, especially on high dimensional data. Very recently, Askari et al. [3] proposed a sparse version of naive Bayes, leading to a combinatorial maximum likelihood capable of solving the binary data and providing explicit bounds on the duality gap for multinomial data, at a fraction of the computing cost.

We also remark that FS is applied and used in various domains including gene selection, face recognition, handwriting identification, and remote sensing [34, 32, 40,

28].

4.3 Proposed Approach

Let A be an $m \times n$ matrix of rank ρ and consider the singular value decomposition (SVD) of A as $A = U\Sigma V^T$, where $U_{m\times m}$ and $V_{n\times n}$ are orthogonal matrices and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_\rho, 0, \ldots, 0)$ is an $m \times n$ diagonal matrix. We denote column j of V by $\mathbf{v}_j$ and row j of V by $\mathbf{v}^j$. Furthermore, we partition $\mathbf{v}^j$ as $\mathbf{v}^j = \left[\begin{array}{c|c} \mathbf{v}^{j,1} & \mathbf{v}^{j,2} \end{array}\right]$, where $\mathbf{v}^{j,1}$ consists of the first ρ entries of $\mathbf{v}^j$ and $\mathbf{v}^{j,2}$ is the remaining $n-\rho$ entries. Note that $A\mathbf{v}_j = 0$, for all $\rho + 1 \leq j \leq n$, and moreover $\ker(A)$ is spanned by all $\mathbf{v}_{\rho+1}, \ldots, \mathbf{v}_n$. We denote by $\mathbf{F}_j$ the j-th column of A.

Let $\bar{V}$ be the matrix consisting of columns $\rho + 1, \ldots, n$ of V, that is $\bar{V} = \left[\begin{array}{c|c|c} \mathbf{v}_{\rho+1} & \cdots & \mathbf{v}_n \end{array}\right]$. Let $P = \bar{V}\bar{V}^T$. Note that $P\mathbf{w} = \mathbf{w}$, for every $\mathbf{w} \in \mathcal{N}(A)$, where $\mathcal{N}(A)$ is the null space of A. Recall that the null space of matrix A consists of all the vectors $\mathbf{b}$ such that $A\mathbf{b} = 0$ and $\mathbf{b}$ is not zero. Indeed, P is the orthogonal projection onto $\mathcal{N}(A)$, that is range of P is $\mathcal{N}(A)$, $P^2 = P$ and $P^T = P$. We also let $S = I - A^\dagger A$. By Lemma 2.1 in [47], we know that S and P are indeed the same. Nevertheless, the computational complexity of computing of S and P might be different. For to compute P we just need the right singular vectors of the symmetric matrix A^TA. On the other hand, if A is full row rank then we know $A^\dagger = A^T(AA^T)^{-1}$. So in case A has full row-rank, the complexity of computing S is the same as complexity of matrix inversion.

Let $D = [A \mid \mathbf{b}]$ be a dataset, say a binary Cancer dataset, where rows of A are samples (patients), columns of A are features (gene expressions) and $\mathbf{b}$ is the class label that each of its entries are either 0 (noncancerous) or 1 (cancerous). In large datasets that are a large number of features that are irrelevant. For example, in gene expression datasets, there are a large number of genes that are not expressed. So, identifying and removing features that have negligible correlations with the class labels is crucial. The aim of FS is to come up with a minimal subset of features that can be used to predict the class labels as accurate as possible. There might be redundancies (correlations) among relevant features that must be detected and removed.

As we explain below, we use the matrix S (or P) to divide the set of all features into clusters where features within a cluster correlate with each other and different clusters are linearly independent from each other. So, a set of linear dependencies defines the correlations within a cluster.

Without loss of generality, we assume that $\{\mathbf{F}_1, \ldots, \mathbf{F}_t\}$ is a cluster, that is $\mathbf{F}_1, \ldots, \mathbf{F}_t$ are linearly dependent and independent of the rest of the $\mathbf{F}_k$, where $k \geq t+1$. The following theorem from [47], is the first major step to identify clusters.

Theorem 4.3.1. *Suppose that* $\{\mathbf{F}_1, \ldots, \mathbf{F}_t\}$ *is a cluster. Then* $P_{i,j} = 0$*, for every* $1 \leq j \leq t$ *and every* $i \geq t+1$*.*

Example 4.3.2. *Consider a* 100×80 *synthetic matrix* A *with the only relations*

between columns of A as follows:

$$-\mathbf{F}_1 + 3\mathbf{F}_2 + 6\mathbf{F}_4 = 0, \quad -\mathbf{F}_6 - 2\mathbf{F}_{10} + 2\mathbf{F}_5 - 4\mathbf{F}_{11} = 0, \quad -\mathbf{F}_3 - 6\mathbf{F}_2 + 3\mathbf{F}_4 = 0,$$

$$-\mathbf{F}_7 - \mathbf{F}_{10} - 3\mathbf{F}_{11} = 0, \qquad -\mathbf{F}_5 + 3\mathbf{F}_{11} + \mathbf{F}_{10} = 0, \qquad -\mathbf{F}_8 + 3\mathbf{F}_{10} + 2\mathbf{F}_{11} = 0,$$

$$-\mathbf{F}_9 + 5\mathbf{F}_5 - \mathbf{F}_7 = 0.$$

The signature matrix S_A (rounded up to two decimals) is:

$$\begin{pmatrix}
0.02 & -0.07 & 0 & -0.13 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
-0.07 & 0.98 & 0.13 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 0.13 & 0.02 & -0.07 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
-0.13 & 0 & -0.07 & 0.98 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0.97 & 0 & 0.03 & 0 & -0.16 & 0.01 & -0.01 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0.37 & 0 & -0.44 & 0 & -0.19 & 0.06 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0.03 & 0 & 0.97 & 0 & 0.16 & -0.01 & 0.01 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & -0.44 & 0 & 0.69 & -0.03 & -0.13 & 0.04 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & -0.16 & 0 & 0.16 & -0.03 & 0.06 & 0.03 & -0.06 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0.01 & -0.19 & -0.01 & -0.13 & 0.03 & 0.94 & 0.02 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & -0.01 & 0.06 & 0.01 & 0.04 & -0.06 & 0.02 & 0.99 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0
\end{pmatrix}$$

We note that A is randomly generated and the only constrain on A is the set of dependent relations given above. We can see that S has a block diagonal form, where each block corresponds to a cluster. So, features $\mathbf{F}_1, \ldots, \mathbf{F}_4$ constitute a cluster. Similarly, $\{\mathbf{F}_5, \ldots, \mathbf{F}_{11}\}$ is another cluster. Note that $\{\mathbf{F}_i\}$ is a singleton cluster, for all $i \geq 12$. We provide some details about these facts in the next lemma.

Lemma 4.3.3. *Let A be the matrix in Example 4.3.2. Then, $P_{i,j} = 0$ for all $1 \leq i \leq 4$ and $5 \leq j \leq n$.*

Proof. We note that rank of A is $\rho = 73$. Hence, $A\mathbf{v}_k = 0$, for every $74 \leq k \leq 80$. Since $A\mathbf{v}_k = 0$ yields a dependence relation between columns of A and $\mathbf{F}_1, \ldots, \mathbf{F}_4$ are independent from the rest of the columns, we deduce that $A\bar{\mathbf{v}}_k = 0$, where $\bar{\mathbf{v}}_k$ consists of the first 4 entries of $\mathbf{v}_k$. Then we form the matrix $M = \left[\begin{array}{c|c|c} \bar{\mathbf{v}}_{74} & \cdots & \bar{\mathbf{v}}_{80} \end{array}\right]$. Since any linear combination of columns of M provides a dependence relation between $\mathbf{F}_1, \ldots, \mathbf{F}_4$, we can use elementary (column) operations to transform M into the matrix $\bar{C}_1$:

$$\bar{C}_1 = \begin{pmatrix} -1.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1.0 & 0 & 0 & 0 & 0 \\ -0.5 & -0.17 & 0 & 0 & 0 & 0 \\ 7.5 & 0.5 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Then $\left[\begin{array}{c|c|c} \mathbf{F}_1 & \cdots & \mathbf{F}_4 \end{array}\right] \bar{C}_1 = 0$; in other words columns of $\bar{C}_1$ give us the minimal relations between $\mathbf{F}_1, \ldots, \mathbf{F}_4$. Let k be in the range $74 \leq k \leq 80$. Since $A\mathbf{v}_k = 0$, we have $v_{1,k}\mathbf{F}_1 + v_{2,k}\mathbf{F}_2 + v_{3,k}\mathbf{F}_3 + v_{4,k}\mathbf{F}_4 = 0$. Substituting for $\mathbf{F}_1$ and $\mathbf{F}_2$ in terms of $\mathbf{F}_3$ and $\mathbf{F}_4$ using the matrix $\bar{C}_1$, we get

$$v_{1,k}(-0.5\mathbf{F}_3 + 7.5\mathbf{F}_4) + v_{2,k}(-\frac{1}{6}\mathbf{F}_3 + 0.5\mathbf{F}_4) + v_{3,k}\mathbf{F}_3 + v_{4,k}\mathbf{F}_4 = 0.$$

We deduce that

$$-0.5v_{1,k} - \frac{1}{6}v_{2,k} + v_{3,k} = 0, \quad 7.5v_{1,k} + 0.5v_{2,k} + v_{4,k} = 0.$$

Since the above equations hold for every k in the range $\rho+1 \leq k \leq n$, we deduce that

$$-0.5\mathbf{v}^{1,2} - \frac{1}{6}\mathbf{v}^{2,2} + \mathbf{v}^{3,2} = 0, \quad 7.5\mathbf{v}^{1,2} + 0.5\mathbf{v}^{2,2} + \mathbf{v}^{4,2} = 0.$$

Let j be in the range $5 \leq j \leq n$. Then taking the dot product with $\mathbf{v}^{j,2}$ yields

$$0.5P_{1,j} - \frac{1}{6}P_{2,j} + P_{3,j} = 0, \qquad 7.5P_{1,j} + 0.5P_{2,j} + P_{4,j} = 0. \tag{4.1}$$

Let $C = \left[\begin{array}{c|c} \bar{C}_1 & 0 \\ \hline 0 & 0 \end{array}\right]$ be an $n \times n$ matrix. Let $\mathbf{c}_1, \ldots, \mathbf{c}_n$ be the columns of C and denote by $\mathbf{p}^j$ the j-th row of P. Since $P\mathbf{c}_i = \mathbf{c}_i$, we deduce that $\mathbf{p}^j\mathbf{c}_i = \mathbf{c}_{i,j} = 0$, since $j \geq 5$. Hence,

$$-P_{1,j} - 0.5P_{3,j} + 7.5P_{4,j} = 0, \quad -P_{2,j} - \frac{1}{6}P_{3,j} + 0.5P_{4,j} = 0. \tag{4.2}$$

Putting together the Equations (4.1) and (4.2), we deduce that

$$B\begin{bmatrix} P_{1,j} & P_{2,j} & P_{3,j} & P_{4,j} \end{bmatrix}^T = 0,$$

where

$$B = \begin{bmatrix} -1 & 0 & -0.5 & 7.5 \\ 0 & -1 & -0.17 & 0.5 \\ -0.5 & -0.17 & 1 & 0 \\ 7.5 & 0.5 & 0 & 1 \end{bmatrix} = \left[\begin{array}{c|c} -I_2 & Z^T \\ \hline Z & I_2 \end{array}\right], \quad Z = \begin{bmatrix} -0.5 & -0.17 \\ 7.5 & 0.5 \end{bmatrix}.$$

Since, by Lemma 2.4 in [47], B is invertible, we deduce that $P_{1,j} = \cdots = P_{4,j} = 0$.

□

In general it follows from Theorem 4.3.1 that after re-ordering the columns of A, the matrix S has a block-diagonal form where each block corresponds to a cluster. Of course, a priori, columns within the same cluster are not next to each other in the matrix A. Furthermore, the converse of Theorem 4.3.1 is not true in general. In other words, $P_{i,j}$ could be zero even when $\mathbf{F}_i$ and F_j are in the same cluster as can be seen in Example 4.3.2 where $P_{1,3} = P_{5,6} = 0$.

To find the clusters, we define a graph G whose vertices consists of $\mathbf{F}_1, \ldots, \mathbf{F}_n$ and we define an edge between $\mathbf{F}_i$ and $\mathbf{F}_j$ if and only if $P_{i,j} \neq 0$. The graph associated to matrix A in Example 4.3.2 is depicted in Figure 4.1.

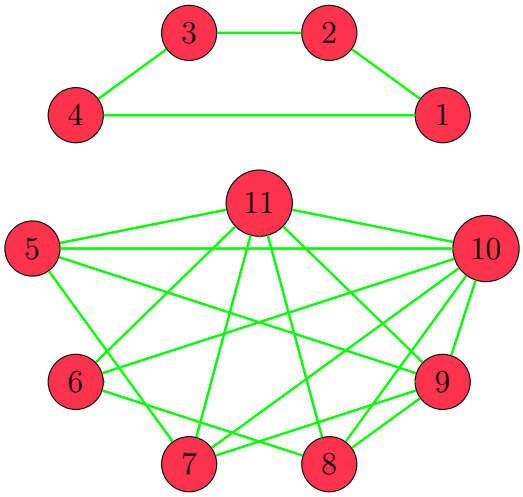


Figure 4.1: The graph associated to matrix A demonstrating the two clusters.

Even though, there may not be an edge between two nodes of the same cluster, it turns out there is always a path connecting every two nodes in the same cluster. This fact which is Theorem 2.10 in [47], can be summarized as follows.

Theorem 4.3.4. *The sub-graph of G consisting of nodes $\mathbf{F}_1, \ldots, \mathbf{F}_t$ and corresponding edges is connected.*

As we mentioned, in real datasets there are many irrelevant features. To identify the irrelevants, we construct the signature matrix S_D of D and identify the cluster that includes $\mathbf{b}$. The remaining clusters consist of features that have a negligible correlation with $\mathbf{b}$. So, we can remove all other clusters from A.

Example 4.3.5. *Let A be the synthetic matrix as in Example 4.3.2 and $\mathbf{b} = \mathbf{F}_1 - 3\mathbf{F}_3 + 2\mathbf{F}_9 - \mathbf{F}_{14}$. The last row of signature matrix S_D (rounded up to four decimals) is:*

$$\begin{pmatrix} -0.0364 & -0.0170 & 0.1093 & 0.0024 & -0.0234 & 0.0006 & 0.0234 & -0.0043 & -0.1403 & 0.0049 & -0.0094 & 0 & 0 & 0.0373 & 0 & \cdots & 0 & 0.0373 \end{pmatrix}$$

The cluster containing $\mathbf{b}$ consists of features $\mathbf{F}_i$ such that $S_{i,n+1} \neq 0$. So, we identify the columns $\mathbf{F}_j$ where $j = 12, 13$ or $15 \leq j \leq 100$ as irrelevant features and remove them from A.

Alternatively, we can also identify irrelevant features by looking at the least-squares solutions of the system $A\mathbf{x} = \mathbf{b}$. Note that $\mathbf{x} = A^{\dagger}\mathbf{b}$, where $A^{\dagger}$ is the pseudo-inverse of A. Each component x_i of $\mathbf{x}$ can be considered as an assigned weight to the feature $\mathbf{F}_i$ of A. Hence, the bigger the $|x_i|$, the more salient $\mathbf{F}_i$ is in correlation with $\mathbf{b}$.

Example 4.3.6. *Let A be the synthetic matrix as in Example 4.3.2 and $\mathbf{b} = \mathbf{F}_1 - 3\mathbf{F}_3 + 2\mathbf{F}_9 - \mathbf{F}_{14}$. We solve $A\mathbf{x} = \mathbf{b}$ using the least-squares method where the vector*

$\mathbf{x}$ *(rounded up to two decimals) is:*

$$\begin{pmatrix} 0.98 & 0.46 & -2.93 & -0.07 & 0.63 & -0.02 & -0.63 & 0.11 & 3.77 & -0.13 & 0.25 & 0 & 0 & -1 & 0 & \cdots & 0 \end{pmatrix}$$

Let $\mathbf{x} = [x_1, \ldots, x_n]$*, where each* x_i *is an assigned weight to* $\mathbf{F}_i$*. Hence, we can approximate* $\mathbf{b}$ *as a linear combination of the form* $x_1\mathbf{F}_1 + \cdots + x_n\mathbf{F}_n$*. Therefore,* $x_i = 0$ *implies* $\mathbf{F}_i$ *has no impact on* $\mathbf{b}$ *and that* $\mathbf{F}_i$ *is irrelevant. According to vector* $\mathbf{x}$*,* $x_i = 0$ *for* $i = 12, 13$ *and* $15 \leq i \leq n$ *and we remove the corresponding* $\mathbf{F}_i$ *from* A*.*

Since, the notion of relevancy is not quantitative and one has to be cautious in removing features, we set a soft threshold Th_{irr} and incorporate both the methods explained in Examples 4.3.5 and 4.3.6. In this chapter, we first filter out features with minimal weight, that is features with $|x_i|$ less than $\frac{1}{n}\sum_{i=1}^{n}|x_i| \times Th_{irr}$ where $\frac{1}{n}\sum_{i=1}^{n}|x_i|$ is the average of the $|x_i|$s. Then we set $|P_{i,n+1}| = 0$ whenever $|P_{i,n+1}| < Th_{irr}$. Note that the last row of S_D reflects the correlations with $\mathbf{b}$. We sort the last row of S_D as descending and remove the features outside the length of $\frac{1}{n}\sum_{i=1}^{n}|P_{i,n+1}| \times (Th_{irr} + 1)$. So, we apply a two-step process with a soft threshold at each step to remove the irrelevant features. Note that we still denote by A the reduced matrix obtained after removing the irrelevant features.

In the next step, we identify redundant features. To do so, we use the signature matrix S_A of A and consider the associated graph. There are many efficient algorithms to find the clusters or connected components of a graph. One such algorithm is Breadth-First Search (BFS) [6]. By applying the BFS starting from vertex $\mathbf{F}_i$, we

can determine its accessible vertices. In other words, different clusters can be specified using BFS on the unvisited vertex $\mathbf{F}_i$. For example, in Fig. 4.1, the first unvisited vertex (feature) is $\mathbf{F}_1$, and applying BFS on $\mathbf{F}_1$ would visit $\mathbf{F}_2, \mathbf{F}_4, \mathbf{F}_3$, respectively. Since there is no unvisited connected feature, the first cluster consists of $\mathbf{F}_1$ to $\mathbf{F}_4$. Then, BFS should be applied to the next unvisited $\mathbf{F}_i$, and add the consequently visited features to the next cluster until all the connected vertices in the current cluster are visited.

From each resulting cluster, a feature that carries the highest MI with $\mathbf{b}$ is selected as the output of the SVFS method. The selected feature from each cluster is, indeed, the one that best represents that cluster. In real datasets we might inherently encounter minor correlations between features, that is in the matrix S_A we might see very small entries that indicate weak correlations. We use a threshold Th_{red} to map the weak feature correlations to zero. Also, in case we encounter a few clusters with numerous vertices, we set a threshold α to split the clusters with more than α vertices into sub-clusters with the maximum of α vertices. The features in each sub-cluster are then sorted based on the last row of S_D, and the top β features are selected to find their highest MI with $\mathbf{b}$. The choice of β features in each sub-cluster is with the aim of reducing the computational cost of the MI calculations.

4.3.1 Algorithm

In this section, we present the algorithm and flowchart of SVFS in Figure 4.2.

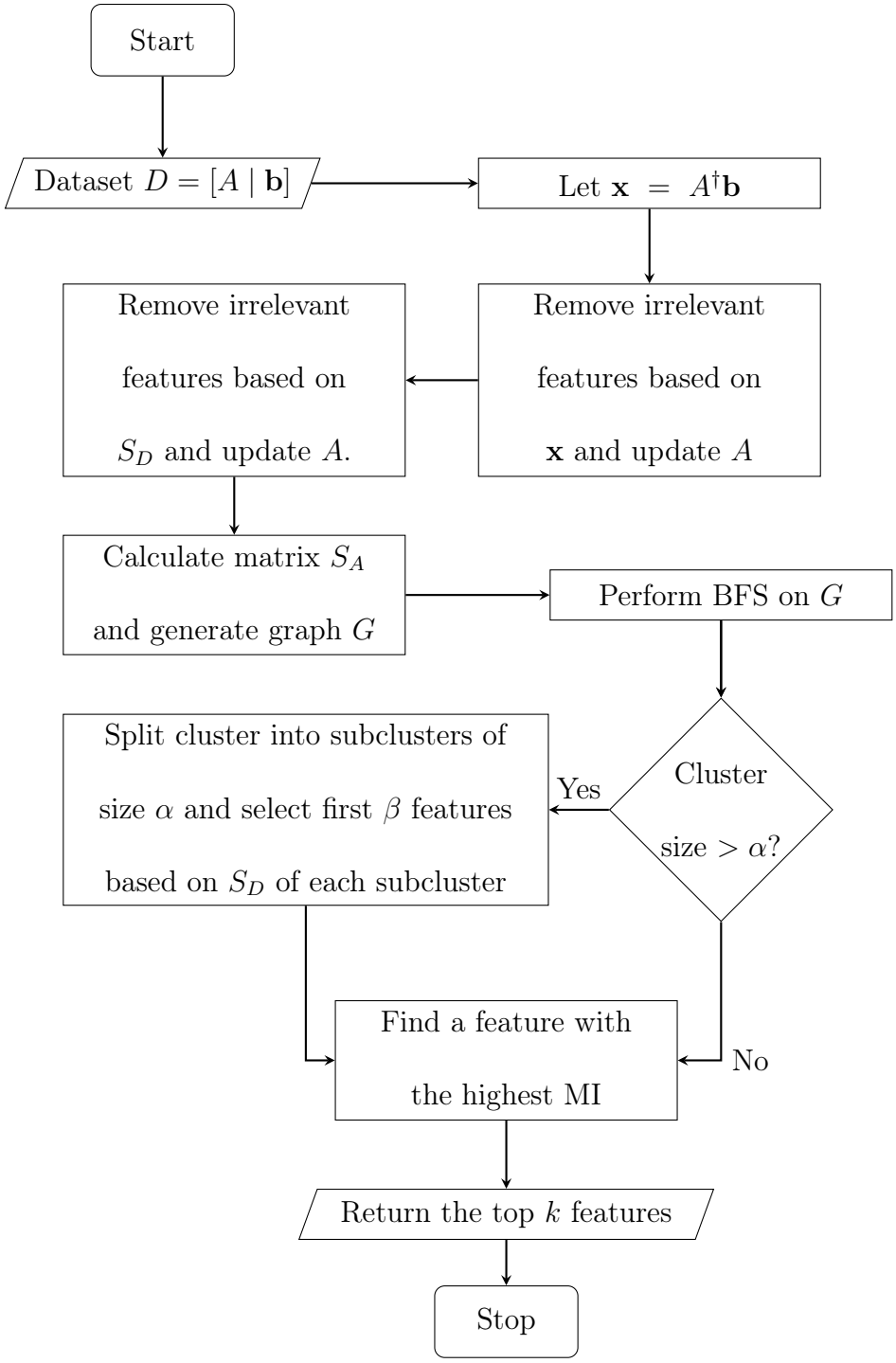


Figure 4.2: Flowchart of SVFS

Algorithm 3: Singular Vectors Feature Selection

Data: $D, k, Th_{irr}, Th_{red}, \alpha, \beta$

Result: Top k of features

(1) $D \leftarrow [A|\mathbf{b}]$;

(2) $\mathbf{x} \leftarrow A^{+}\mathbf{b}$;

(3) cuttoff$\leftarrow Th_{irr} \times Mean(\mathbf{x})$;

(4) $I \leftarrow \emptyset$;

(5) **for** *each* $x_i \in \mathbf{x}$ **do**

(6) | **if** $x_i \geq$ *cuttoff* **then**

(7) | | $I \leftarrow I \cup i$;

(8) $A \leftarrow A[I]$;

(9) $D \leftarrow [A|\mathbf{b}]$;

(10) $S_D \leftarrow \boldsymbol{I} - (D^{\dagger} \times D)$;

(11) $S \leftarrow$ Sort(Last row of S_D);

(12) $\mathbf{F}_{cleaned} \leftarrow S[1 : S_i > Mean(S) \times Th_{irr}]$;

(13) $A \leftarrow A[\mathbf{F}_{cleaned}]$;

(14) $P \leftarrow \boldsymbol{I} - A^{\dagger}A$;

(15) **for** *each column* $\mathbf{p}_i$ *of* P **do**

(16) **if** $P_{i,j} \geq Th_{red} \times Mean(\mathbf{p}_i)$ **then**

(17) $G \leftarrow Add_{node}(j)$;

(18) $G \leftarrow Add_{edge}(i, j)$;

(19) **while** *(node* $\in G$*) or (len*$(k_{Top}) < k$*)* **do**

(20) $Cluster \leftarrow$ BFS$(node)$;

(21) **if** $len(Cluster) > \alpha$ **then**

(22) $subClusters \leftarrow$ Split $(Cluster, \alpha)$;

(23) **for** *each* $subCluster_i \in subClusters$ **do**

(24) $c_i \leftarrow$ Sort $subCluster_i$ based on S;

(25) $k_{Top} \leftarrow k_{Top} \cup$ Max (MI$(c_i[1{:}\beta])$);

(26) **else**

(27) $k_{Top} \leftarrow k_{Top} \cup$ Max (MI $(node \in Cluster\ )$);

(28) $G \leftarrow Remove_{node}(G, Cluster)$;

(29) **return** k_{Top};

The while loop in the algorithm essentially demonstrates finding the connected components of the graph associated to P. The well-known BFS algorithm finds the connected components of a graph $G(V, E)$ with complexity $\mathcal{O}(|V|+|E|)$. In our case, $|E|$ is the number of non-zero entries in P. So, the worst case in the algorithm can happen when $|E|= \frac{n(n-1)}{2}$. Hence, the complexity of the while loop is $\mathcal{O}(n^2)$. We also mention that parallel algorithms for BFS have been of great interest, see for example [9].

The complexity of computing $S = I - A^{\dagger}A$ is more delicate. There is extensive research on finding efficient and reliable methods to find $A^{\dagger}$, see for example [42, 49, 46]. One of the most commonly used methods is the Singular Value Decomposition (SVD) which is very accurate but time and memory intensive especially in the case of large matrices. The complexity of computing SVD of $A_{m\times n}$ is $\mathcal{O}(\min(mn^2, m^2n))$.

Pseudo-inverses are used in neural learning algorithms to solve large least square systems. So, there is a great interest in finding the pseudo-inverse efficiently. Courrieu in [13] proposed an algorithm called Geninv based on Cholesky factorization and showed that the computation time is substantially shorter, particularly for large systems. It is noted in [13] that the complexity of Geninv on a single-threaded processor is $\mathcal{O}(\min(m^3, n^3))$ whereas in a multi-threaded processor, the time complexity is $\mathcal{O}(\min(m, n))$. The authors in [31] investigated the effective computation of the pseudo-inverse for neural networks and concluded that QR factorization with column pivoting along with Geninv works well. Since our implementation is single-threaded

and $m << n$, the complexity of pseudo-inverse is $\mathcal{O}(m^3)$. We can conclude that the complexity of our algorithm is at most $\mathcal{O}(\max(m^3, n^2))$.

4.4 Experimental Result

We compared our method with eight state-of-the-art FS methods including Conditional Infomax Feature Extraction (CIFE), Joint Mutual Information (JMI), Fisher score, Trace Ratio criterion, Least angle regression (LARS), Hilbert-Schmidt independence criterion least absolute shrinkage and selection operator (HSIC-Lasso), Conditional Covariance Minimization (CCM), and Sparse Multinomial Naive Bayes (SMNB). We used the scikit-feature library, which is an open-source feature selection repository in Python developed in the Arizona State University (ASU). It includes the implementation of CIFE, JMI, LARS, Fisher, and Trace Ratio methods. The reset of methods, namely, HSIC-Lasso, CCM, and SMNB are implemented in Python by their authors. To have a fair comparison among the different FS methods, we take advantage of 5-fold stratified cross-validation (CV) of the dataset so that 80% of each class is selected for FS. Then we use the Random Forest (RF) classifier with its default setting implemented in [37], to build a model based on the selected features and evaluate the model on the remaining 20% of the dataset. We report the average classification accuracy over 10 independent runs (twice 5-fold CV) using the RF classifier on each dataset.

4.4.1 Datasets

We selected a variety of publicly available datasets from two sources, i.e. Gene Expression Omnibus (GEO) which has various real genomic data, and the scikit-feature selection repository at Arizona State University which has benchmark biological and face image data to perform feature selection and classification. The specifications of these datasets are given in Tables 4.1 and 4.2.

The pre-processing of GEO datasets used in this research was carried out by cleaning and converting the NCBI datasets to the CSV format. The mapping between the gene samples and the probe IDs has been retrieved using GEO2R [5] and the probe IDs that did not have a gene mapping have been removed. For each gene, the expression values are obtained by averaging the expression values of all the probe IDs mapped to that specific gene. The k-Nearest Neighbors (kNN) imputation method was used to handle the missing values.

4.4.2 Hardware and Software

Our proposed method SVFS and other methods described in section 4.4 have been run on an IBM LSF 10.1.0.6 machine (Suite Edition: IBM Spectrum LSF Suite for HPC 10.2.0) with requested 8 nodes, 16 GB of RAM, and 8 GB swap memory using Python 3.6. Note that we only set 240 GB of RAM for the CCM model as it requires a high volume of memory.

Table 4.1: Benchmark Datasets Specifications

Dataset	**#Samples**	**#Features**	**Type**	**#Labels**	**Proportion of labels**									
					1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
TOX_171	171	5,748	Biological	4	26.3%	26.3%	22.8%	24.6%	–	–	–	–	–	–
SMK_CAN_187	187	19,993	Biological	2	48.1%	51.9%	–	–	–	–	–	–	–	–
Prostate_GE	102	5,966	Biological	2	49%	51%	–	–	–	–	–	–	–	–
lymphoma	96	4,026	Biological	9	47.9%	10.4%	9.4%	11.4%	6.3%	6.3%	4.1%	2.1%	2.1%	–
leukemia	72	7,070	Biological	2	65.3%	34.7%	–	–	–	–	–	–	–	–
lung	203	3,312	Biological	5	68.5%	8.4%	10.3%	9.8%	3%	–	–	–	–	–
GLIOMA	50	4,434	Biological	4	28%	14%	28%	30%	–	–	–	–	–	–
GLI_85	85	22,283	Biological	2	30.6%	69.4%	–	–	–	–	–	–	–	–
CLL_SUB_111	111	11,340	Biological	3	9.9%	44.1%	46%	–	–	–	–	–	–	–
ALLAML	72	7,129	Biological	2	65.3%	34.7%	–	–	–	–	–	–	–	–
colon	62	2,000	Biological	2	64.5%	35.5%	–	–	–	–	–	–	–	–
NCI9	60	9,712	Biological	9	15%	15%	13.3%	8.3%	11.7%	10%	13.3%	10%	3.33%	–
pixraw10P	100	10,000	Image	10	10%	10%	10%	10%	10%	10%	10%	10%	10%	10%
warpAR10P	130	2,400	Image	10	10%	10%	10%	10%	10%	10%	10%	10%	10%	10%
warpPIE10P	210	2,420	Image	10	10%	10%	10%	10%	10%	10%	10%	10%	10%	10%
orlraws10P	100	10,304	Image	10	10%	10%	10%	10%	10%	10%	10%	10%	10%	10%

Table 4.2: Genomic Datasets Specifications

Dataset	Samples	# Original F	# Cleaned F	# Labels	Proportion of labels			
					1	2	3	4
GDS1615	127	22,282	13,649	3	33%	20.5%	46.5%	–
GDS3268	202	44,290	29,916	2	36.1%	63.9%	–	–
GDS968	171	12,625	9,117	4	26.3%	26.3%	22.8%	24.6%
GDS531	173	12,625	9,392	2	20.8%	79.2%	–	–
GDS2545	171	12,625	9,391	4	10.6%	36.8%	38%	14.6%
GDS1962	180	54,675	29,185	4	12.8%	14.4%	45	27.8%
GDS3929	183	24,526	19,334	2	69.9%	30.1%	–	–
GDS2546	167	12,620	9,583	4	10.2%	35.3%	39.5%	15%
GDS2547	164	12,646	9,370	4	10.4%	35.4%	39%	15.2%

4.4.3 Parameters

The input parameters of our proposed SVFS method are $k, Th_{irr}, Th_{red}, \alpha, \beta$. The parameter k denotes the number of selected features and is a common parameter in all the methods evaluated in this study. There is no fixed procedure in the literature for determining the optimum value of k, but in many research works [27, 18, 7, 4], it is set to 50 which seems to be satisfactory in many cases. However, we take k in a wider range from 10 and 90 to ensure a fairground for comparison. When a subset of k features are returned as the output of a FS algorithm, we feed the first t features from the subset to the classifier to find an optimal t so that the subset of first t features yields the highest accuracy. This set up is applied across all FS methods.

Also, we report average classification accuracy of a model over 10 independent runs (we run stratified 5-fold CV twice).

The parameter Th_{irr} is the threshold set to filter out the irrelevant features. We set the value of Th_{irr} to 3. The parameter Th_{red} is another threshold defined to deal with the low level of sparsity of S. In real-world large datasets, the condition $S_{i,j} = 0$ might rarely be encountered. Indeed, the threshold Th_{red} maps the weak feature correlations to zero. Here, we have set the value of Th_{red} to 4 for the biological datasets and 7 for the face image datasets. The parameter α is used when facing big clusters to divide the clusters into subclusters with α members. The parameter β is the number of features selected from each of the subclusters with α members. In this work, we have set the values of α and β to 50 and 5, respectively.

4.4.4 Results

The average classification accuracies over 10 independent runs (twice 5-fold CV) using the RF classifier on the datasets described in Section 4.4.1 are presented in this section. In Figure 4.3, we present the classification accuracy of SVFS compared to the other FS methods on 4 benchmark face image datasets. As it can be seen, our method attains either the best or second best accuracy compared to other FS methods. It is interesting to note that SVFS attains 100% accuracy on all of pixraw10P, warpPIE10P, and orlraws10P with at most 90 features.

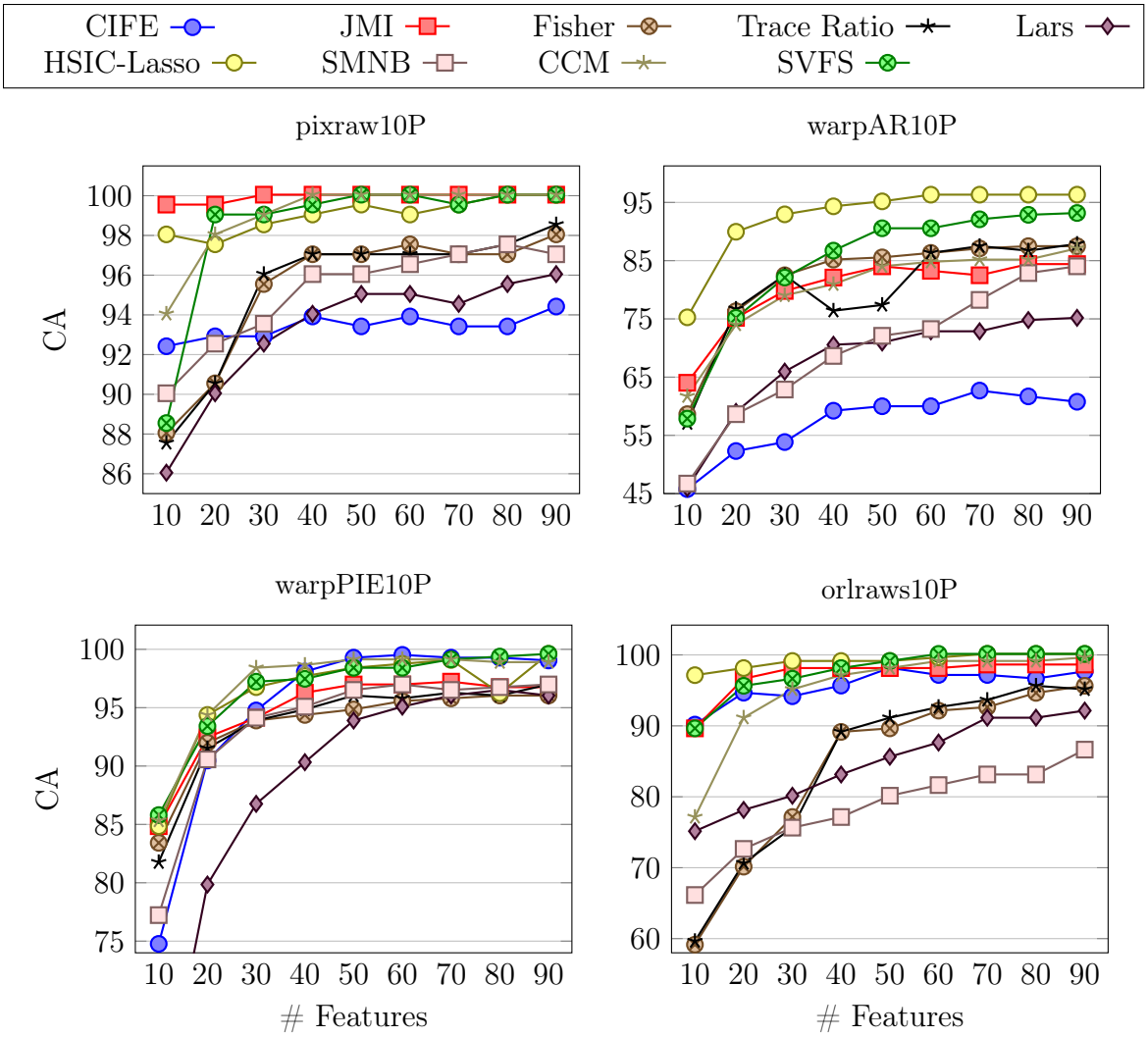


Figure 4.3: Average classification accuracy of feature selection by CIFE, JMI, Fisher, Trace Ratio, Lars, HSIC-Lasso, SMNB, CCM and SVFS over 10 runs on benchmark face image datasets

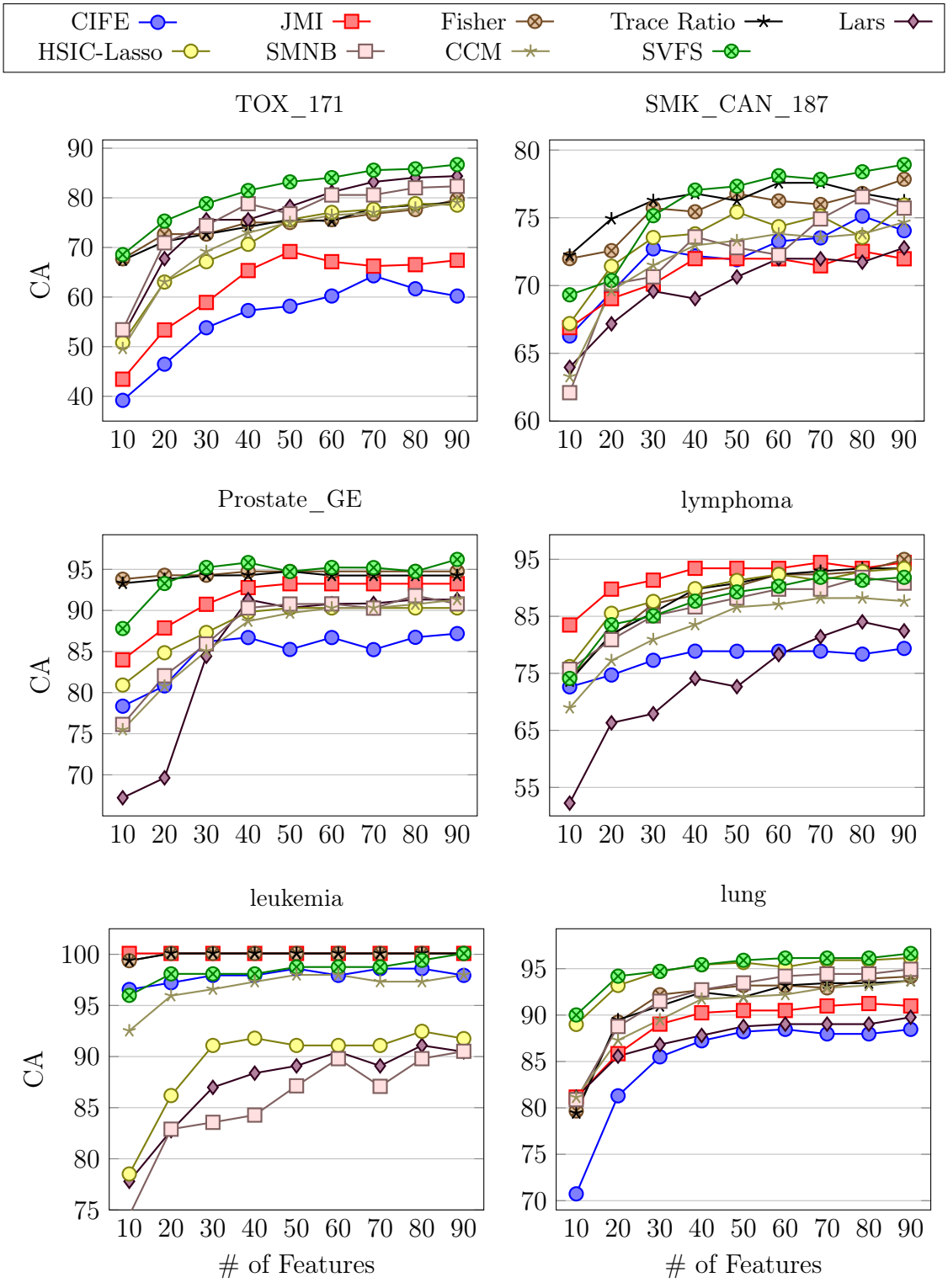
Figure 4.4 shows the classification accuracy performance of SVFS compared to the other methods on benchmark biological datasets. As we can see, SVFS has performed consistently well and achieved the highest accuracy in 7 out of the 12 cases, while

producing reasonably good accuracies in most of the other cases as well. JMI has produced the highest accuracy in 3 cases, where Fisher and HSIC-Lasso have shown their best performance in GLIOMA and ALLAML datasets, respectively. As we mentioned, the thresholds Th_{irr} and Th_{red} are set for 3 and 4, respectively for all biological datasets. However, it is possible to tune these thresholds and get better results. For example, if we set $Th_{irr} = 1.2$ and $Th_{red} = 2$, we get an average accuracy of 94.52 and 96.37 on ALLAML and Lymphoma datasets, respectively, and using at most 50 features ($\alpha = 50, \beta = 15$). Similarly, $Th_{irr} = 1.1$ and $Th_{red} = 2$, gives an average accuracy of 87 on GLIOMA dataset ($\alpha = 50, \beta = 15$), while $Th_{irr} = 1.2$ and $Th_{red} = 4$, gives an average accuracy of 74.14 on NCI9 dataset ($\alpha = 50, \beta = 10$).

The general superiority of SVFS can be further witnessed on genomics datasets with large number of features as shown in Figure 4.5. Note again that $Th_{irr} = 3$ and $Th_{red=4}$ for all these datasets. However, it is possible to tune the parameters Th_{irr} and Th_{red} to obtain better results per dataset. This can be particularly useful when we focus on specific datasets for disease diagnosis and biomarker discovery.

We conclude from Figures 4.3, 4.4, and 4.5 that our proposed SVFS has achieved the highest accuracy on 12 datasets out of the total 25 datasets, while noting that no other method has achieved the highest accuracy for more than 4 datasets. In cases where SVFS has not produced the highest accuracy, its performance is nonetheless among the most accurate ones.

Since IBM LSF is capable of reporting running time, CPU time, and memory


CIFE
JMI
Fisher
Trace Ratio
Lars
HSIC-Lasso
SMNB
CCM
SVFS
TOX_171
SMK_CAN_187
Prostate_GE
lymphoma
leukemia
lung
CA
# of Features

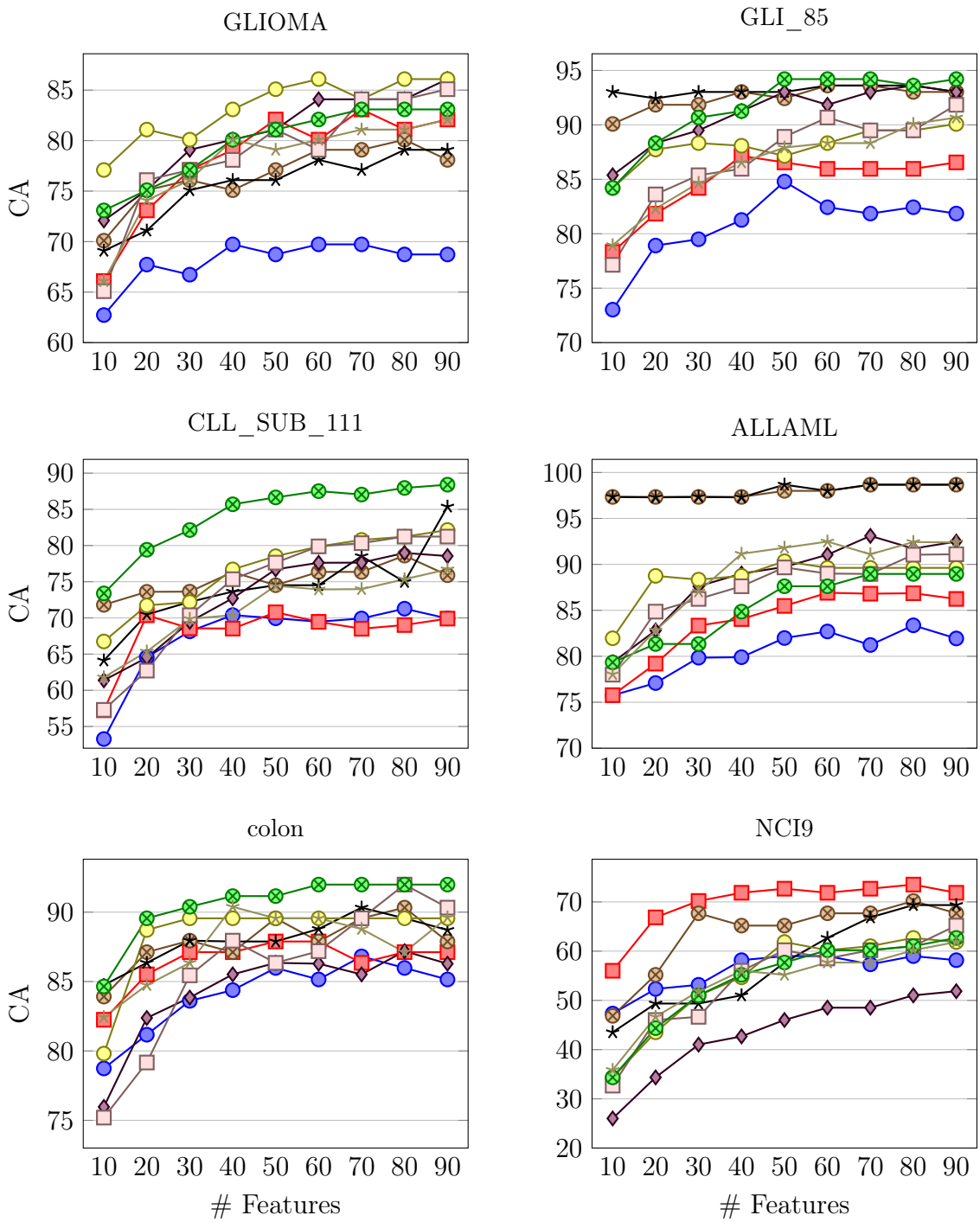
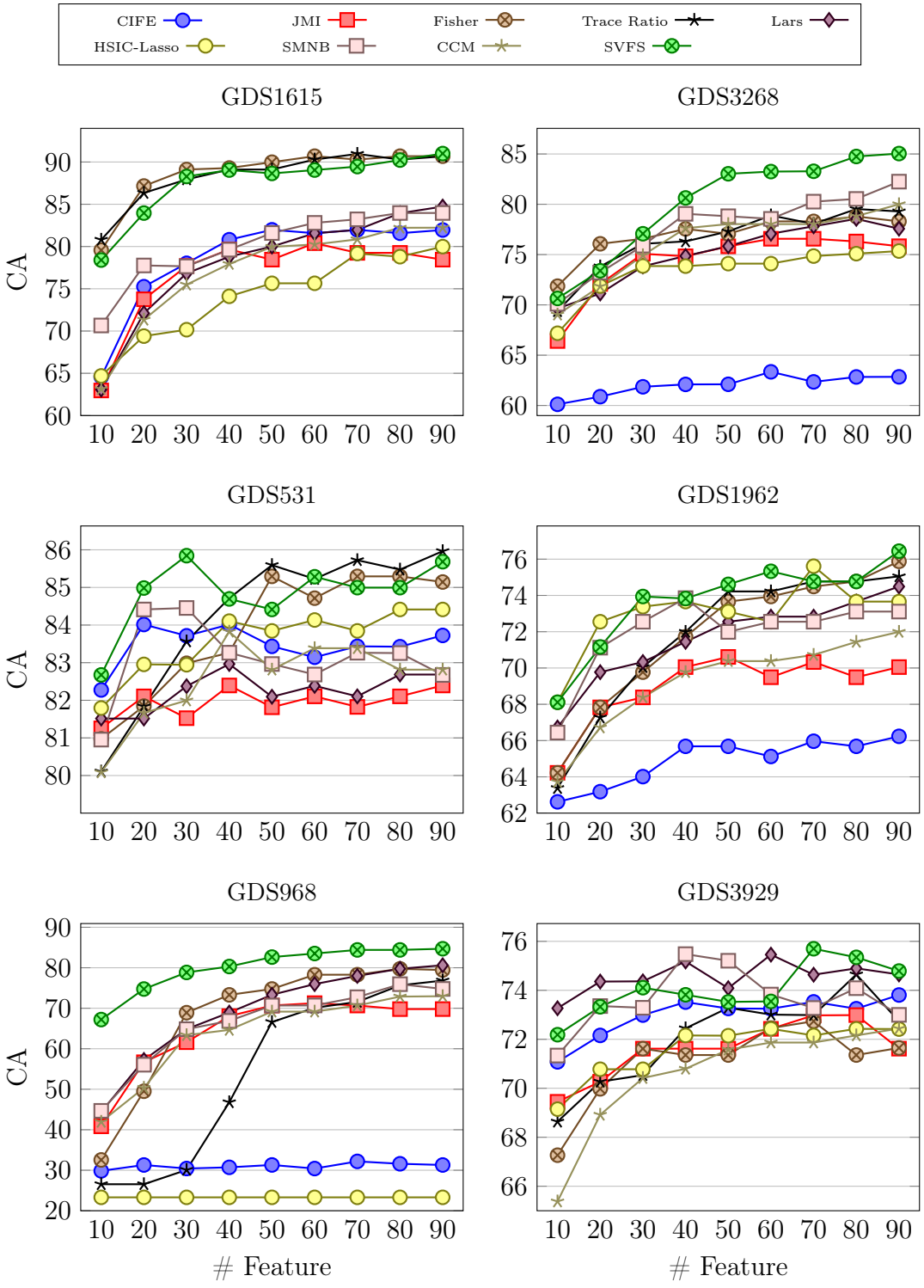


Figure 4.4: Average classification accuracy of feature selection by CIFE, JMI, Fisher, Trace Ratio, Lars, HSIC-Lasso, SMNB, CCM and SVFS over 10 independent runs on benchmark biological datasets


CIFE
JMI
Fisher
Trace Ratio
Lars
HSIC-Lasso
SMNB
CCM
SVFS
GDS1615
GDS3268
GDS531
GDS1962
GDS968
GDS3929
CA
Feature

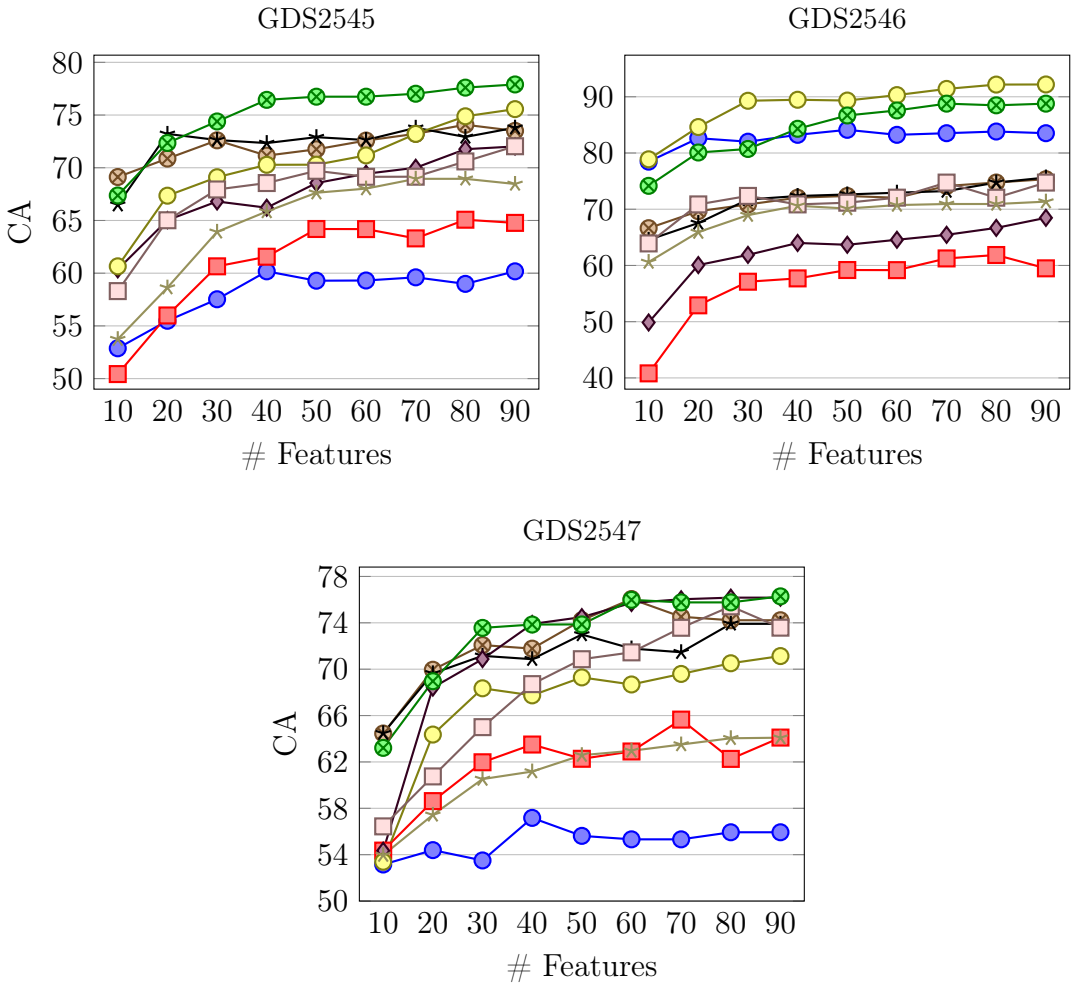


Figure 4.5: Average classification accuracy of feature selection by CIFE, JMI, Fisher, Trace Ratio, Lars, HSIC-Lasso, SMNB, CCM and SVFS over 10 independent runs on genomic datasets

usage by each feature selection model, we depict the running time in seconds for all feature selection methods in Figure 4.6. As there are 25 datasets for the evaluation process, Figure 4.6(a) includes the running time on the benchmark biological and benchmark image datasets and Figure 4.6(b) covers the running time on the genomic datasets. Note that the reported running times include the RF classification time. It can be seen that the running times of CIFE and JMI are worse than other methods while the running time of CCM method on GEO datasets is high and roughly the same as CIFE and JMI. The other methods including SVFS have comparable and very reasonable running times in the sense that these methods can be comfortably run on regular PCs.

Some methods because of their immense cost of computing are implemented in parallel to perform in reasonable running time. Since HSIC-Lasso hired all available core of CPUs, its CPU time is comparable with CIFE and JMI methods, as shown in Figure 4.6(c). Moreover, the CCM model takes advantage of TensorFlow [1] with an optimized CPU implementation in a parallel way, leading to a high CPU time on most of the datasets. The rest of the methods are implemented in a non-parallelized manner; therefore, their CPU times are relatively similar to their running times.

In terms of performance in memory usage, Figure 4.6(d) shows that CIFE, JMI, Fisher, SMNB, and SVFS are efficient and required comparatively low memory. In contrast, CCM, HSIC-Lasso, and Trace Ratio required a high volume of memory in the magnitude of thousands.

CIFE
JMI
Fisher
Trace Ratio
Lars
HSIC-Lasso
CCM
SMNB
SVFS
Running Time (Second)
10^4
10^3
10^2
TOX_171
SMK_CAN_187
Prostate_GE
lymphoma
leukemia
lung
GLIOMA
GLI_85
CLL_SUB_111
ALLAML
colon
NCI9
Pixraw10P
warpAR10P
warpPIE10P
orlraws10P

(a)

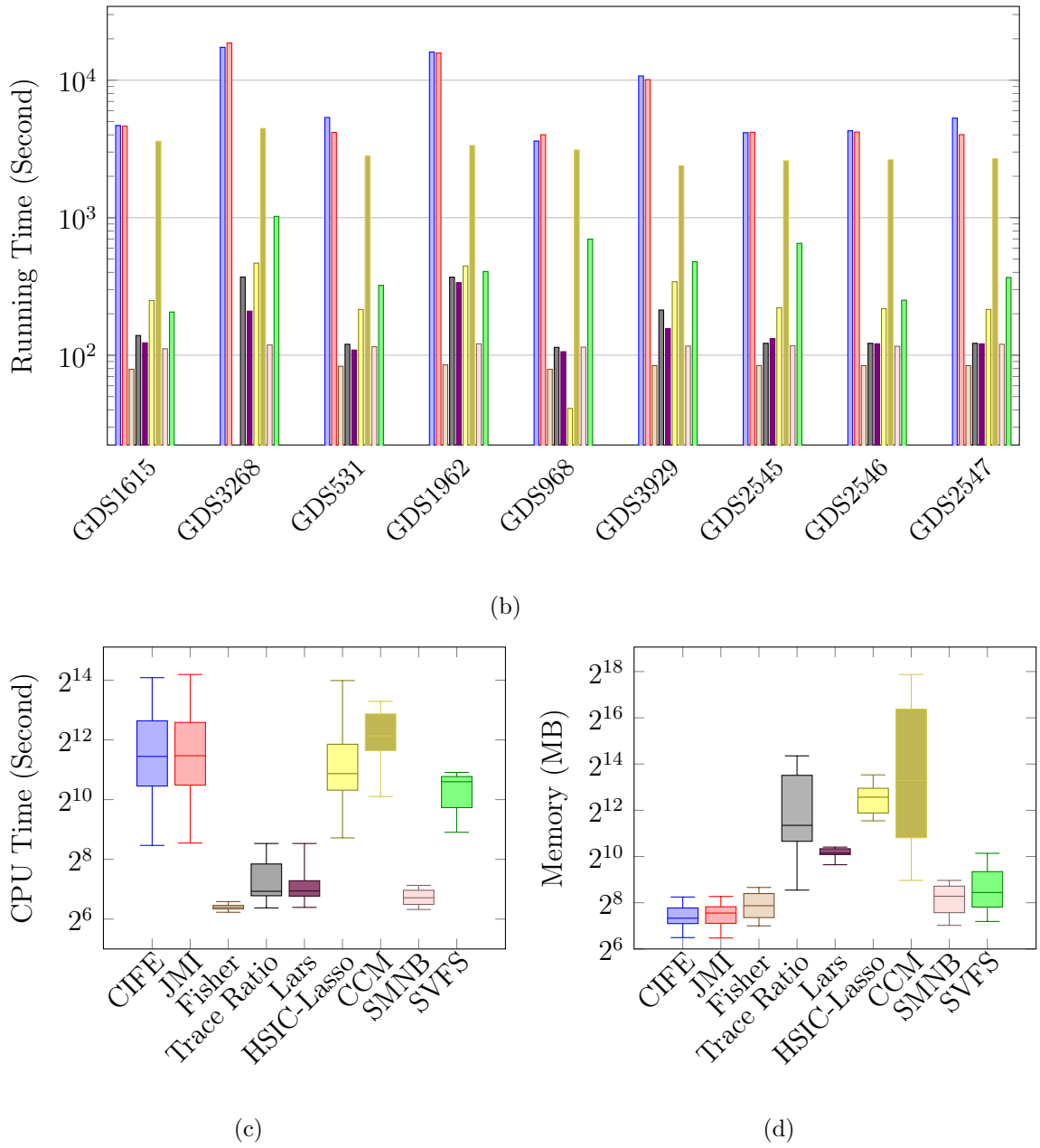


Figure 4.6: (a), (b) Running Time, (c) CPU Time and (d) Memory taken by CIFE, JMI, Fisher, Trace Ratio, Lars, HSIC-Lasso, CCM, SMNB and SVFS over 10 runs using RF classifier

4.5 Conclusion

In this chapter, we have proposed a feature selection method (SVFS) based on singular vectors of a matrix. Given a matrix A with its pseudo-inverse $A^{\dagger}$, we showed that the signature matrix $S_A = I - A^{\dagger}A$ can be used to determine correlations between columns of A. To do this, we associate a graph where the vertices are the columns of A and columns $\mathbf{F}_i$ and $\mathbf{F}_j$ are connected if $S_{i,j} \neq 0$. We show that connected components of this graph are the clusters of columns of A so that columns in a cluster correlate only with columns in the same cluster. We consider a dataset $D = [A \mid \mathbf{b}]$, where rows of A are samples, columns of A are features, and $\mathbf{b}$ is the class label. Then we use the signature matrix S_D and its associated graph to find the cluster of columns of D that correlate with $\mathbf{b}$. This allows us to reduce the size of A by filtering out the columns in the other clusters as irrelevant features. In the next step, we use the signature matrix S_A of A to partition columns of A into clusters and then pick the most important features from each cluster.

A comprehensive assessment on benchmark and genomic datasets shows that the proposed SVFS method outperforms the state-of-the-art feature selection methods. Our algorithm includes two thresholds Th_{irr} and Th_{red} that are used to filter out irrelevant and remove redundant features, respectively. The thresholds have been set identical for the same type of datasets. However, it is possible to further tune the parameters Th_{irr} and Th_{red} to obtain better results. This can be particularly useful when we focus on specific datasets for disease diagnosis and biomarker discovery.

Bibliography

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation OSDI* (2016), pp. 265–283.

[2] Afshar, M., and Usefi, H. Dimensionality reduction using singular vectors. *Scientific Reports-Nature 11*, 1 (2021), 1–13.

[3] Askari, A., d'Aspremont, A., and El Ghaoui, L. Naive feature selection: sparsity in naive bayes. In *International Conference on Artificial Intelligence and Statistics* (2020), pp. 1813–1822.

[4] Balin, M. F., Abid, A., and Zou, J. Concrete autoencoders: Differentiable feature selection and reconstruction. In *International Conference on Machine Learning* (2019), pp. 444–453.

[5] Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillippy, K. H., Sherman, P. M., Holko, M., et al. NCBI GEO: archive for functional genomics data sets—update. *Nucleic acids research 41*, D1 (2012), D991–D995.

[6] Beamer, S., Asanovic, K., and Patterson, D. Direction-optimizing breadth-first search. In *Proceedings of the International Conference on High Per-*

formance Computing, Networking, Storage and Analysis (2012), IEEE, pp. 1–10.

[7] Behzadian, B., Gharatappeh, S., and Petrik, M. Fast feature selection for linear value function approximation. In *Proceedings of the International Conference on Automated Planning and Scheduling* (2019), vol. 29, pp. 601–609.

[8] Brown, G., Pocock, A., Zhao, M.-J., and Luján, M. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *Journal of machine learning research 13*, Jan (2012), 27–66.

[9] Buluç, A., and Madduri, K. Parallel breadth-first search on distributed memory systems. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis* (2011), pp. 1–12.

[10] Chen, J., Stern, M., Wainwright, M. J., and Jordan, M. I. Kernel feature selection via conditional covariance minimization. In *Advances in Neural Information Processing Systems* (2017), pp. 6946–6955.

[11] Chen, Y., Zhang, Z., Zheng, J., Ma, Y., and Xue, Y. Gene selection for tumor classification using neighborhood rough sets and entropy measures. *Journal of biomedical informatics 67* (2017), 59–68.

[12] Coleto-Alcudia, V., and Vega-Rodríguez, M. A. Artificial bee colony algorithm based on dominance (abcd) for a hybrid gene selection method. *Knowledge-Based Systems* (2020), 106323.

[13] Courrieu, P. Fast computation of moore-penrose inverse matrices. *Neural Information Processing-Letters and Reviews 8*, 2 (2005).

[14] Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern classification*. John Wiley & Sons, 2012.

[15] Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. Least angle regression. *The Annals of Statistics 32*, 2 (2004), 407–499.

[16] El Akadi, A., El Ouardighi, A., and Aboutajdine, D. A powerful feature selection approach based on mutual information. *International Journal of Computer Science and Network Security 8*, 4 (2008), 116.

[17] Etzioni, R., Urban, N., Ramsey, S., McIntosh, M., Schwartz, S., Reid, B., Radich, J., Anderson, G., and Hartwell, L. The case for early detection. *Nature reviews cancer 3*, 4 (2003), 243–252.

[18] Gao, X., Liu, S., Song, H., Feng, X., Duan, M., Huang, L., and Zhou, F. Ageguess, a methylomic prediction model for human ages. *Frontiers in Bioengineering and Biotechnology 8* (2020), 80.

[19] Ghosh, M., Begum, S., Sarkar, R., Chakraborty, D., and Maulik, U. Recursive memetic algorithm for gene selection in microarray data. *Expert Systems with Applications 116* (2019), 172–185.

[20] Guo, B., and Nixon, M. S. Gait feature subset selection by mutual information. *IEEE Transactions on Systems, MAN, and Cybernetics-part a: Systems and Humans 39*, 1 (2008), 36–46.

[21] Hayes, D. F. Prognostic and predictive factors revisited. *The breast 14*, 6 (2005), 493–499.

[22] Hikichi, S., Sugimoto, M., and Tomita, M. correlation-centred variable selection of a gene expression signature to predict breast cancer metastasis. *Scientific Reports 10*, 1 (2020), 1–8.

[23] Jain, I., Jain, V. K., and Jain, R. Correlation feature selection based improved-binary particle swarm optimization for gene selection and cancer classification. *Applied Soft Computing 62* (2018), 203–215.

[24] Jiang, L., Greenwood, C. M., Yao, W., and Li, L. Bayesian hyper-lasso classification for feature selection with application to endometrial cancer RNA-seq data. *Scientific Reports 10*, 1 (2020), 1–16.

[25] Kohavi, R., and John, G. H. Wrappers for feature subset selection. *Artificial intelligence 97*, 1-2 (1997), 273–324.

[26] Kononenko, I. Estimating attributes: analysis and extensions of relief. In *European conference on machine learning* (1994), Springer, pp. 171–182.

[27] Leordeanu, M. Feature selection meets unsupervised learning. In *Unsupervised Learning in Space and Time*. Springer, 2020, pp. 125–155.

[28] Li, B., Lei, L., and Zhang, X.-P. Constrained discriminant neighborhood embedding for high dimensional data feature extraction. *Neurocomputing 173* (2016), 137–144.

[29] Lin, D., and Tang, X. Conditional infomax learning: an integrated framework for feature extraction and fusion. In *European conference on computer vision* (2006), Springer, pp. 68–82.

[30] Lorenzo, P. R., Tulczyjew, L., Marcinkiewicz, M., and Nalepa, J. Hyperspectral band selection using attention-based convolutional neural networks. *IEEE Access 8* (2020), 42384–42403.

[31] Lu, S., Wang, X., Zhang, G., and Zhou, X. Effective algorithms of the moore-penrose inverse matrices for extreme learning machine. *Intelligent Data Analysis 19*, 4 (2015), 743–760.

[32] Luo, F., Huang, H., Duan, Y., Liu, J., and Liao, Y. Local geometric structure feature for dimensionality reduction of hyperspectral imagery. *Remote Sensing 9*, 8 (2017), 790.

[33] Luo, F., Zhang, L., Du, B., and Zhang, L. Dimensionality reduction with enhanced hybrid-graph discriminant learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing* (2020).

[34] Lv, M., Hou, Q., Deng, N., and Jing, L. Collaborative discriminative manifold embedding for hyperspectral imagery. *IEEE Geoscience and Remote Sensing Letters 14*, 4 (2017), 569–573.

[35] Meyer, P. E., Schretter, C., and Bontempi, G. Information-theoretic feature selection in microarray data using variable complementarity. *IEEE Journal of Selected Topics in Signal Processing 2*, 3 (2008), 261–274.

[36] Nie, F., Xiang, S., Jia, Y., Zhang, C., and Yan, S. Trace ratio criterion for feature selection. In *AAAI* (2008), vol. 2, pp. 671–676.

[37] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research 12* (2011), 2825–2830.

[38] Peng, H., Long, F., and Ding, C. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 8 (2005), 1226–1238.

[39] Sayed, S., Nassef, M., Badr, A., and Farag, I. A nested genetic algorithm for feature selection in high-dimensional cancer microarray datasets. *Expert Systems with Applications 121* (2019), 233–243.

[40] Shi, G., Huang, H., and Wang, L. Unsupervised dimensionality reduction for hyperspectral imagery via local geometric structure feature learning. *IEEE Geoscience and Remote Sensing Letters* (2019).

[41] Shukla, A. K., Singh, P., and Vardhan, M. DNA gene expression analysis on diffuse large b-cell lymphoma (DLBCL) based on filter selection method with supervised classification method. In *Computational Intelligence in Data Mining*. Springer, 2019, pp. 783–792.

[42] Stanimirović, I. *Computation of generalized matrix inverses and applications*. CRC Press, 2017.

[43] Sun, L., Kong, X., Xu, J., Zhai, R., Zhang, S., et al. A hybrid gene selection method based on relieff and ant colony optimization algorithm for tumor classification. *Scientific Reports 9*, 1 (2019), 1–14.

[44] Tam, V., Patel, N., Turcotte, M., Bossé, Y., Paré, G., and Meyre, D. Benefits and limitations of genome-wide association studies. *Nature Reviews Genetics 20*, 8 (2019), 467–484.

[45] Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

[46] Toutounian, F., and Ataei, A. A new method for computing moore–penrose inverse matrices. *Journal of Computational and applied Mathematics 228*, 1 (2009), 412–417.

[47] Usefi, H. Clustering, multicollinearity, and singular vectors. *arXiv preprint arXiv:2008.03368* (2020).

[48] Wang, Y., Klijn, J. G., Zhang, Y., Sieuwerts, A. M., Look, M. P., Yang, F., Talantov, D., Timmermans, M., Meijer-van Gelder, M. E., Yu, J., et al. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *The Lancet 365*, 9460 (2005), 671–679.

[49] Wei, Y., Stanimirovic, P., and Petkovic, M. *Numerical and symbolic computations of generalized inverses*. World Scientific, 2018.

[50] Yamada, M., Jitkrittum, W., Sigal, L., Xing, E. P., and Sugiyama, M. High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation 26*, 1 (2014), 185–207.

[51] Yamada, M., Tang, J., Lugo-Martinez, J., Hodzic, E., Shrestha, R., Saha, A., Ouyang, H., Yin, D., Mamitsuka, H., Sahinalp, C., et al. Ultra high-dimensional nonlinear feature selection for big biological data. *IEEE Transactions on Knowledge and Data Engineering 30*, 7 (2018), 1352–1365.

[52] Yang, H. H., and Moody, J. Data visualization and feature selection: New algorithms for nongaussian data. In *Advances in Neural Information Processing Systems* (2000), pp. 687–693.

[53] Yu, H., Gu, G., Liu, H., Shen, J., and Zhao, J. A modified ant colony optimization algorithm for tumor marker gene selection. *Genomics, proteomics & bioinformatics* 7, 4 (2009), 200–208.

Chapter 5

Conclusions

The amount of high-dimensional data has been significantly growing in the past few years. Hence, machine learning methods face more complicated challenges in handling and interpreting the large number of input features. To apply machine learning methods effectively and efficiently, feature selection is needed as an essential data pre-processing technique. This process makes data mining algorithms faster and more reliable, increases predictive accuracy, and enhances comprehensibility. Moreover, the proper application of feature selection methods improves the learning process, in terms of generalization capacity, learning speed, and reducing the complexity of the induced model, and computation costs.

Feature selection involves detecting relevant (important) features and removing irrelevant, redundant, or noisy data. Irrelevant features provide no useful information, and redundant features produce no more information than the selected important

features. In microarray high-dimensional data, the quantity of samples is considerably less than the number of features; therefore, in such a case, machine learning falls in hassle states as the search space gets sparsely populated. Accordingly, the model will not be able to distinguish between noise and relevant data accurately. Selecting the most important features turns this type of data into a balanced structure; this process is also known as dimensionality reduction.

To filter out the irrelevant features, we have proposed a method (SLS) based on the least-square solutions. This methods was presented in Chapter 2. We only focused on detecting irrelevant features, and examined the state-of-the-art feature selection methods with both original features and filtered features. With less noisy data, all examined feature selection methods achieved competitive prediction accuracy and performed much faster. In other words, we turned the benchmark datasets into a lower dimension by merely removing irrelevant features. We determined the relevance of features to the class label based on a threshold because the notion of relevancy is not quantitative, particularly in real-world datasets. We showed that SLS could be added to any pre-processing technique as our experiments clearly reflect that SLS optimizes the performance of feature selection algorithms over different dataset types in terms of running time and classification accuracy by fitting a soft threshold (see Sections 2.4.4, 2.4.5, and 2.4.6).

After successfully detecting and removing irrelevant features, the main goal of this thesis is finding important features among redundant features. Since redundant

features have some correlations with some other features and the class label, we can view them as different clusters, in which each cluster includes only a few important features that have a considerable impact on the class label compared to the rest of the cluster members. Therefore, in Chapter 3, we proposed a noise-robust feature selection method (DRPT) for high-dimensional genomic datasets. The novelty of DRPT is removing the irrelevant features outright using the SLS optimizer, and then clustering the resulting reduced dataset to detect the features correlated to the class label in each cluster using the perturbation theory. We also took advantage of the entropy of each feature during the clustering process and the final ranking. We showed that DRPT perfectly distinguishes irrelevant and redundant features on a synthetic dataset, and for proof of concept, we extended our examination over ten genomic datasets. Moreover, DRPT is insensitive to the permutation of rows or columns of the data, and our evaluation showed it outperformed the state-of-the-art feature selection algorithms in classification accuracy and running time (see Section 3.4.4).

To further extend our investigations in the clustering process of the reduced dataset for feature selection, we proposed a new feature selection method (SVFS) based on singular vectors of a matrix in Chapter 4. We introduced a signature matrix in which the correlations between features are encoded. We considered each feature as a vertex and transfered the signature matrix to a graph representation where a correlation between two features is exposed as an edge. Then we partitioned the set of all features into different clusters where the features within a cluster cor-

relate with each other, and different clusters are linearly independent of each other. Finally, using mutual information, the most important features from each cluster were selected. We introduced two thresholds in the process of filtering irrelevant and redundant features. These thresholds can be adjusted to improve results for different types of datasets. The general superiority of SVFS was further observed through a comprehensive assessment of the genomic and face datasets (see Section 4.4.4).

This thesis aims to provide a new filter-based feature selection that transforms the data into a lower dimension outright by removing irrelevant features and then detecting the important features by clustering. For future work, we plan to turn our filter-based method into a wrapper-based one. Using a solid grid search module, we will be able to obtain the most desirable value for the adjustable parameters, particularly for Th_{irr} and Th_{red} (see Section 4.4.3), with the help of a solid classifier feedback. Incorporating a classifier will certainly increase the running time, and the parallel implementation can resolve this problem because, the powerful FS methods follow this path and utilize the immense potential of parallel computations. Moreover, we can upgrade our model to an unsupervised feature selection as the signature matrix can also represent the correlations between features of a dataset without the class label.