QATAR UNIVERSITY

COLLEGE OF ENGINEERING

FEATURES RANKING TECHNIQUES FOR SINGLE NUCLEOTIDE POLYMORPHISM DATA

BY

MOHANAD FEISAL M H ABOUNADA

A Thesis Submitted to the Faculty of

the College of Engineering

in Partial Fulfillment

of the Requirements

for the Degree of

Masters of Science in Computing

June 2017

COMMITTEE PAGE

The members of the Committee approve the Thesis of Mohanad Feisal M H Abounada defended on 16/05/2017.

_____

Abbes Amira
Thesis/Dissertation Supervisor

_____

Ali Jaoua
Committee Member

_____

Tamer Elsayed
Committee Member

Approved:

_____

Khalifa Al-Khalifa, Dean, College of Engineering

ABSTRACT

ABOUNADA, MOHANAD, F., Masters: June : [2017], Masters of Science in Computing

Title: Features Ranking Techniques for Single Nucleotide Polymorphism Data

Supervisor of Thesis: Abbes Amira.

Identifying biomarkers like single nucleotide polymorphisms (SNPs) is an important topic in biomedical applications. Such SNPs can be associated with an individual's metabolism of drugs, which make these SNPs targets for drug therapy, and useful in personalized medicine applications. Yet another important application is that SNPs can be associated with an individual's genetic predisposition to develop a disease. Identifying these associations allow proactive steps to be taken to hinder, delay or eliminate the disease. However, the problem is challenging; data are high dimensional and incomplete, and features (SNPs) are correlated. The goal of this thesis is to propose features ranking methods to reduce the number of selected features and the computational cost required to select these features in a binary classification task.

The main idea of the hypothesis is that specific values within a feature might be useful in predicting specific classes, while other values are not. In this context, three heuristic methods are applied to select the best features. The methods are applied to the Wellcome Trust Case Control Consortium (WTCCC1) dataset, and evaluated on Texas A&M University Qatar's High Performance Computing platform.

The results show that the classification accuracy achieved by the proposed methods is comparable to the baseline. However, one of the proposed methods reduced the execution time of the feature selection and the number of features required to achieve

similar accuracy in the baseline by 40% and 47% respectively.

DEDICATION

For my father and my mother.

ACKNOWLEDGMENTS

I would like to thank my thesis supervisor Prof. Abbes Amira for his efforts in guiding and helping me through my thesis journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ARM	Association Rule Mining
AUC/AUROC	Area Under the ROC Curve
BD	Bipolar Disorder
CAD	Coronary Artery Disease
CE	Conditional Entropy
CV	Cross Validation
DNA	Deoxyribonucleic acid
FCBF	Fast Correlation Based Filter
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GE	Grammatical Evolution
GWAS	Genome Wide Association Study
HGP	Human Genome Project
HT	Hypertension
IBD	Inflammatory Bowel Disease
IG	Information Gain
KI	Kuncheva Index
KNN	K Nearest Neighbors
LOOCV	Leave One Out Cross Validation
MCC	Matthews Correlation Coefficient

NB	Naïve Bayes
NCBI	National Center for Biotechnology Information
PPM	Paths Towards Personalized Medicine
QNRF	Qatar National Research Fund
QNRS	Qatar National Research Strategy
RA	Rheumatoid Arthritis
RF	Random Forests
SNP	Single Nucleotide Polymorphism
SU	Symmetrical Uncertainty
SVM	Support Vector Machine
T1D	Type 1 Diabetes
T2D	Type 2 Diabetes
TN	True Negative
TP	True Positive
TPR	True Positive Rate

CHAPTER 1: INTRODUCTION

Human Genome

The Deoxyribonucleic acid (DNA) is a molecule that takes the shape of a double helix. Nucleotides are the main building block of the DNA. Each nucleotide is composed of a sugar group, a phosphate group and a nitrogen base. There are four different nitrogen bases; Adenine (A), Thymine (T), Guanine (G) and Cytosine (C). To form a base pair, two nucleotides are bound together. In nature, A pairs with T and G with C. As shown in Figure 1, the base pairs are chained together to give the DNA its double helix shape. The complete set of base pairs is called Genome [1].

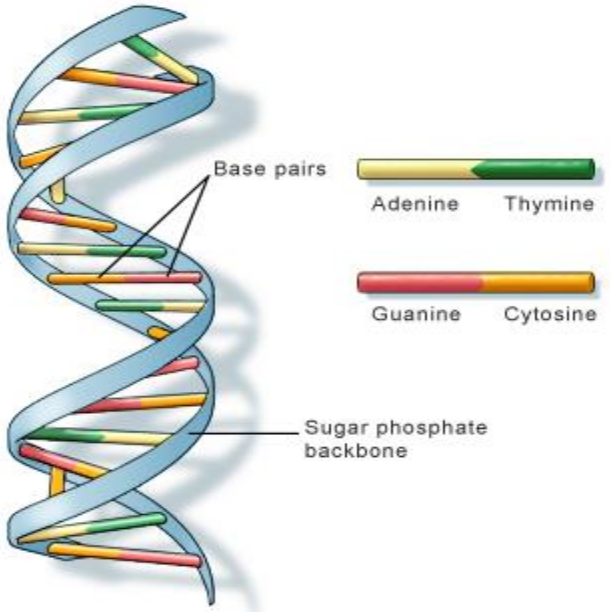


Figure 1. An illustration of DNA double helix [1]

The genome can be divided into two types of regions; coding (genes) and non-coding. While the former regions occupy around 1% of the genome, they are responsible for protein production. The latter are responsible for regulating the protein production [2]. In homo sapiens (humans), the genome consists of around 3.2 billion base pairs. The order of these base pairs is responsible for making each human unique [3].

Bioinformatics in Qatar

In the past few years, there was an increased interest in genomics and bioinformatics in Qatar. This section, spotlights these areas of interests in Qatar.

The Qatar National Research Strategy (QNRS) identifies the grand challenges of Qatar and serves as guidelines for investments in research. Since its launch in 2012, the strategy clearly identified bioinformatics as one of its goals. One of the goals of the Computer Since and Information Technology pillar aims to "Develop a demand-driven bioinformatics research program serving both genomics-driven investigations and emerging research effort in energy and environment" [4].

Qatar Biobank, is a collaborative project between Hamad Medical Corporation and the Ministry of Health. The project serves as repository of biological samples of Qatari nationals and long term residences. In addition, the project hosts the Qatar Genome Program, which aims to develop personalized healthcare through the integration of innovative genomics technologies into medical and research practice [5].

The Paths Towards Personalized Medicine (PPM) is a research program funded by Qatar National Research Fund (QNRF). The program addresses three main challenges. All of which implicate the development of bioinformatics technologies [6].

Motivation

Studies show that there are differences among humans' genomes in the order of 0.1% [7]. These differences are called variations and truly responsible of defining the uniqueness of each individual. There are many types of variations which can be distinguished based on length, location, being passed to offspring or not, and if it is a deletion, insertion, duplication, or rearrangement [8]. However, the focus of this thesis is the Single Nucleotide Polymorphism (SNP), as it accounts for 90% of variations [9]. As the name implicates, a SNP is an alternation in a single nucleotide in a specific location within the genome in at least 1% of a population [10]. Figure 2 shows an illustration of a SNP.

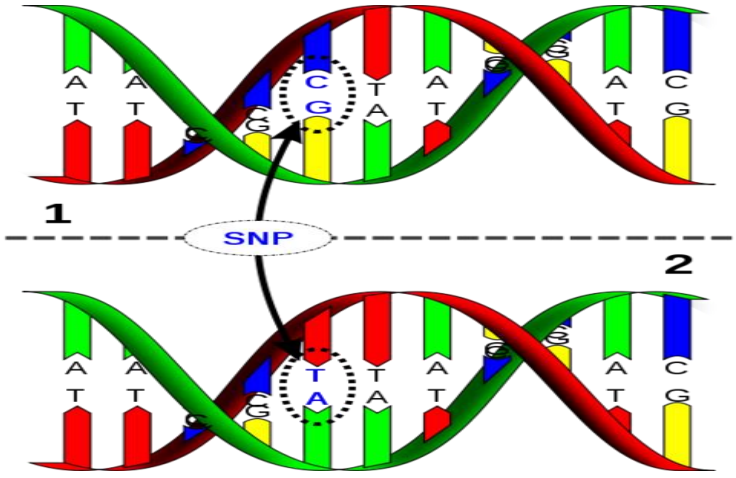


Figure 2. An illustration of a SNP [11]

Since the release of the first human reference genome in 2003 by the Human Genome Project (HGP), the cost of genome sequencing has dropped significantly. The main reason for this decrease in the cost is the revolutionary sequencing technologies developed in the following decade. As a consequence of this, a torrent of raw sequencing data became available, making the human genome a fertile field for studies [12].

There are wide range of studies and analysis that can be applied to the genome. However, the focus of this thesis is the Genome Wide Association Study (GWAS). In a GWAS, many variants, most likely SNPs, are investigated to find which of these variants are associated with a trait of interest [13]. Because the number of traits that can be examined in GWAS is vast, there are several applications for the GWAS.

If a SNP or SNPs are successfully associated with a disease, these SNPs can be used as predictive markers for an individual's susceptibility to develop the disease. This allows proactive actions to be taken in order to prevent, delay, or reduce the effect of that disease [14]. Yet another important application is that SNPs can be associated with metabolism of drugs, which make these SNPs target for drug therapy [15], and useful in personalized medicine applications [16]. Finally, SNPs can be associated with visually observed characteristics like height, eye, and hair color and ethnicity, which opens the door for applications in DNA forensic [17].

The aforementioned useful applications come with challenges, these challenges are discussed latter in this chapter.

It is worth mentioning that dbSNP, the National Center for Biotechnology Information's (NCBI) SNPs database, as of its build 147, lists around 150 million SNPs in the human genome [18]. With this large number of SNPs, testing every individual's genome for all SNPs would be expensive. For that reason, a SNPs microarray is used to test the genome for a predefined set of SNPs, and rather than sequencing the whole genome, only the values of these predefined SNPs are reported. The selection of these predefined SNPs is based on the possibility of these SNPs to carry the most information about patterns of genetics variations [19]. For example, the Affymetrix SNP 6.0 lists around 900k SNPs [20].

There are multiple criteria for SNPs classification, a common criterion is the number of alleles. Each allele, is an observed variant at the SNP location. In a bi-allelic SNP, there are two different variants alternating among the population. In contrary, in a multi-allelic SNP there are three or more variants alternating among the population [21]. Bi-allelic SNPs are the most common SNPs observed in human genome [22]. Figure 3 shows a toy example of bi-allelic and multi-allelic SNPs.

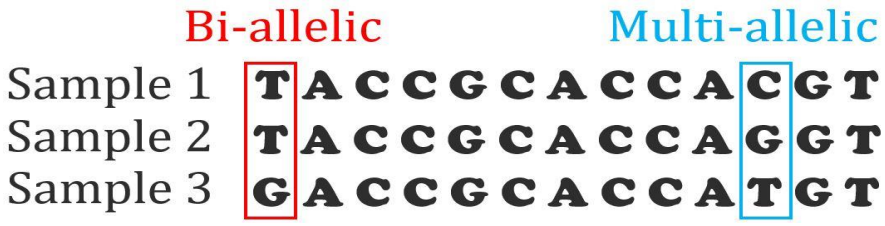


Figure 3. Toy example of bi-allelic and multi-allelic SNPs

Problem Statement

There is a set of individuals' SNPs samples. These samples can be divided into two groups. The first group has samples that are known to have a specific trait or disease and called "Cases". While the second group has samples that known to be free of that specific trait or disease, and called "Controls". The goal is to design a machine learning technique that can classify an unknown sample to either group.

Such machine learning technique faces several challenges. Firstly, due to the large number of features (SNPs), a dimensionality reduction phase is required. The goal of the dimensionality reduction phase is to select or extract the most informative features. Secondly, the number of samples is small when compared to the number of features. Thirdly, features are correlated. This is due to the fact that multiple SNPs are working in coordination to manifest complex diseases [23], this challenge must be addressed either in the dimensionality reduction phase or classification phase. Finally, samples are incomplete; this requires either to adapt techniques that can tolerate such missing data or to introduce an additional phase to fill the missing data properly.

Research Objectives

The overall objective of this thesis is to design machine learning techniques suitable for the SNPs data that can classify an unknown sample to one of the classes in a binary classification problem. However, the focus of this thesis is the dimensionality reduction of the SNPs data. The objectives of this thesis are shown in the following.

- Conduct a literature review on the existing machine learning techniques for the SNPs data, while focusing on the dimensionality reduction methods presented in the literature. The literature review will be used to identify gaps, current challenges, and serves as guidance for the proposed solution.
- Investigate the existing dimensionality reduction techniques for SNPs data, and evaluate unused techniques.
- Propose dimensionality reduction techniques that aim to reduce the number of selected features and the time required to select these features.
- Design machine learning techniques for SNPs data. The designed techniques should include codifying, imputation, dimensionality reduction, and classifiers training and testing.
- Evaluate the existing and proposed dimensionality reduction techniques, using two different classifiers and multiple datasets.

Research Contributions

The main contributions of this thesis are shown in the following.

- A literature review is conducted. For each of the selected studies, the addressed challenges, main idea, used dataset, and the achieved performance are presented. The changes in the addressed challenges over time were discussed, and a comparison between the reviewed studies is presented.
- Symmetrical Uncertainty (SU) is used in the baseline as it was performing well in the literature, and Conditional Entropy (CE) is evaluated as it was never used.
- The main contribution of this thesis is a feature scoring scheme that is suitable for categorical features. The proposed scheme is based on that specific value within a feature might be useful in predicting specific class labels, while other values are not. In this context, each group of similar values within a feature are considered a region. For each region, a score that is ranging from zero to one is computed and a class label is assigned. A positive score means that the region is more informative for one of the class labels in a binary classification problem.
- Complete machine learning techniques are designed, and three features ranking techniques are proposed based on the proposed scoring scheme.
- The proposed solutions and the existing methods are evaluated with Support Vector Machine (SVM) and K-Nearest Neighbors (KNN)

classifiers on seven different datasets. The proposed solutions reduced the computational time for feature selection and the number of selected features when compared to the baseline.

Thesis Structure

Chapter 2 presents a formal definition of the problem. In Chapter 3, a literature review of existing machine learning techniques for SNPs data is provided. In addition, a comparison between the reviewed methods is presented. In Chapter 4, the dataset preprocessing, baseline and the proposed methods are described in details. While Chapter 5 presents the experimental results, the conclusion and future work are provided in Chapter 6.

CHAPTER 2: FORMAL PROBLEM DEFINITION

Problem Description

m SNPs, $\{s_1, s_2, s_3, \ldots, s_m\}$ for n individuals are given. The existential status of a specific trait for these individuals is known. Individuals with the trait are known as cases, while individuals without the trait are known as controls. The goal is to learn from the given SNPs to be able to identify to which of these two groups, cases and controls, an unseen individual belongs to. However, the number of SNPs is vast and some values of some SNPs for some individuals are not provided. In additions, it is known that multiple SNPs may play a role in the existence of the trait. Moreover, multiple SNPs may play the same role in the existence of the trait.

From the machine learning point of view, each SNP is a feature, and each individual's given values of the all SNPs is a sample. The learner that should learn from the given samples is the classifier. The existential status of the trait in the given samples are the targets or classes. The complete set of samples and targets is the dataset. Because there are two classes, and each sample can only be assigned one class label, the problem is a binary classification problem.

To design a proper machine learning solution for the given problem, several questions must be addressed first, like how to measure the effectiveness of the solution? And how to deal with the missing data? These questions and more are investigated in the following sections.

Preprocessing

In most cases, the obtained raw datasets are not suitable for classification tasks. Thus, some preprocessing steps might be required to transform the obtained dataset to a suitable format. These steps are described in the following.

Quality Control

Quality control, is the process of cleaning the dataset from noisy data. One method is to remove samples that fails to satisfy some criteria. Another method, is to remove samples that show unusual behavior compared to the samples within the same class, such samples are called outliers. Some common examples of the used criteria are percentage of the missing values in the sample or the feature.

Imputation

Imputation, is the process of filling the missing values in the dataset. One method is to replace the missing value with the most common value in the feature within the same class. Another method, is to replace the missing value with the most common value in the samples that share similar factors with sample where the value is missing. It is worth mentioning that some classifiers can tolerate missing values in the dataset.

Codifying

Most classifiers require their input to be numerical. Codifying is the process of converting the values of a feature or class labels from text format to numerical format. Some common examples of the used codifying strategies are binary codifying and one-hot encoding. Some codifying strategies might increase the dimensionality of the data.

Classification Performance Metrics

Several solutions can be applied to the given problem, and it is important to find which of these solutions is the best. One way is the measure how good the used classifier is performing. Several metrics can be used for that purpose; however, most of these metrics are derived from the confusion matrix [25] which shown in Figure 4. The terminology used in the confusion matrix is described in the following:

- Condition Positives (P): The number samples have the trait.
- Condition Negatives (N): The number of samples does not have the trait.
- True Positives (TP): The number of samples predicted to have the trait, and in fact they have the trait.
- True Negatives (TN): The number of samples predicted to not have the trait, and in fact they do not have the trait.
- False Positives (FP): The number of samples predicted to have the trait, and in fact they do not have the trait.
- False Negatives (FN): The number of samples predicted to not have the trait, and in fact they have the trait.

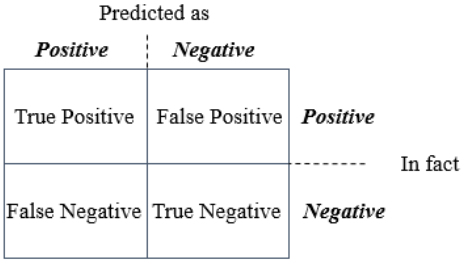


Figure 4. Confusion Matrix

Accuracy

The classification accuracy is used to find the percentage of samples that were correctly classified by the classifier. The following equation shows how the accuracy can be derived from the confusion matrix:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

The accuracy; however, can be misleading, specifically when the dataset is unbalanced. For example, consider a dataset with 100 samples where 95 samples do not have the trait and 5 samples have the trait. A classifier that classifies any given sample as not having the trait, will achieve a 95 percent of accuracy. For that reason, additional metrics are required to assess the performance of the classifier.

Sensitivity and Specificity

The sensitivity is used to measure the ability of the classifier to correctly classify samples with the trait. While specificity, is used to measure the classifier ability to correctly classify samples without the trait. Both measures must be considered together, as considering one of them alone can also be misleading. The following equations show how the sensitivity and specificity can be derived from the confusion matrix:

$$Sensitivity = \frac{TP}{TP + FN} \quad (2)$$

$$Specificity = \frac{TN}{TN + FP} \quad (3)$$

Area Under the ROC Curve

Some classifiers are capable of producing probability scores for the classified samples. The user can set a threshold *T*, and the sample is classified into one of the classes if the probability score is above the threshold, and to another, if the probability score is below the threshold. All previous metrics assess the classifier for a single threshold. In the ROC curve [26], the True Positive Rate (TPR) versus the False Positive Rate (FPR) are plotted while varying the threshold. The ROC curve allows the user to visualize how the classifier is performing under multiple thresholds, which also allow the user to select the most appropriate threshold. The area under the ROC curve (AUROC / AUC) is a one number measure that can summarize the overall performance of the classifier, and make it easier to compare with other classifiers. The following equations show how TPR and FPR can be derived from the confusion matrix:

$$TPR = Sensitivity = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = Fallout = \frac{FP}{FP + TN} \tag{5}$$

Dimensionality Reduction

The number of SNPs that the classifier should learn from is large. In addition, it is known that most of these SNPs are nonrelevant to the trait under investigation. Thus, for the given *m* SNPs, $\{s_1, s_2, s_3, \ldots, s_m\}$, the goal is to find the subset *S* of the most informative SNPs. This problem is called the feature selection problem, which is the focus of this thesis.

Reducing the number of the selected features, brings four main benefits to the classification task [27]:

- By selecting fewer number of features, the data required for the classification task will be less.
- Fewer number of features, make the interpretation of the classifier outcomes easier.
- Less computational cost to train the classifier.
- The classification accuracy is increased.

The simplest way to find the subset *S*, is to try every possible subset. However, due to the large number of SNPs, this method is intractable. The authors in [28], classified the problem as NP-hard optimization problem, which means that there is no known solution that can find the optimal subset in a reasonable amount of time. Therefore, the feature selection methods that found in the literature can only find near-optimal solutions.

To precisely define the optimal solution, three criteria are defined, which will be used later to differentiate among the feature selection methods. For any feature selection method *M* that finds a subset *S'*, the three criteria are:

- Accuracy (A): the classification accuracy achieved by using the subset *S'*, as shown in the following equation:

$$Select\ S' \in \{s_1, s_2, \ldots, s_m\},\ to\ maximize\ (A) \quad (6)$$

- Time (T): the time required to find the subset *S'*, as shown in the following equation:

$$Select\ S' \in \{s_1, s_2, \ldots, s_m\},\ to\ minimize\ (T) \quad (7)$$

- The number of SNPs (*K*): the number of SNPs in the subset *S'*, as shown in the following equation:

$$Select\ S' \in \{s_1, s_2, \ldots, s_m\},\ to\ minimize\ (K) \quad (8)$$

In [28], the feature selection methods are categorized into three categories; filter methods, wrapper methods and embedded methods; each of these categories are described in the following sections.

Filter Methods

The filter methods work independently from the classifier. Relevancy scores are computed between each feature and the class labels. Based on these scores, the features are ranked and higher-scoring features are selected. These methods are fast, because the feature selection is performed only one time. However, in most of these methods, each feature is considered alone, which can miss the possible interactions among features. In addition, the selected subset may not be suitable for the used classifier.

Wrapper Methods

In the wrapper methods, multiple subsets are found by a predefined search function. Each of these subsets, are evaluated, and the best subset is then selected. The evaluation of each subset is done by training and testing a given classifier. Thus, feature selection is said to be wrapped around the classifier. The main advantage of such methods is that the selected subset is tailored for the used classifier, which is also a disadvantage in some sense. In addition, these methods can capture the interactions among features. However, the time complexity of these methods is very high, and there is a high risk of overfitting.

Embedded Methods

Like the wrapper methods; however, the internal parameters of the classifier are embedded in the predefined search function. These methods can capture the interactions among features and less computationally intensive than the wrapper methods.

Cross-Validation

The designed machine learning solution must be evaluated with unseen samples. The simplest way to do this, is to split the dataset into train and test splits. If parameters tuning is required, a development split is used. This is possible if the obtained dataset is large enough. If the dataset is not large enough, cross-validation (CV) [53] can be used, which allow the model to be tested on the whole dataset. In *N*-fold CV, the dataset is split into *N* equal size splits, and the evaluation of the solution is done *N* times (folds). In each time, *N-1* splits are used for training and one split for testing. The results from all folds are then averaged to obtain the overall result. If the dataset is very small, leave-one-out CV (LOOCV) can be used where the number of splits is equal to the number of samples in the dataset.

CHAPTER 3: LITERATURE REVIEW

Overview

The goal of the literature review is to investigate the existing methods in the literature that address the dimensionality reduction problem for the SNPs data. The dimensionality reduction problem is one part of a larger problem. In fact, the literature was searched for studies that harness machine learning to predict the existence status of a given disease based on SNPs data. From these studies, the dimensionality reduction part is extracted and reviewed.

As shown in Figure 5, The selected studies are grouped based on the type of the used dimensionality reduction technique, as defined in Chapter 2. For each study, the addressed challenge, main idea, used dataset, and the achieved performance, are presented. It is worth noting that the performance can be expressed in terms of classification performance metrics or by the computational cost and time.

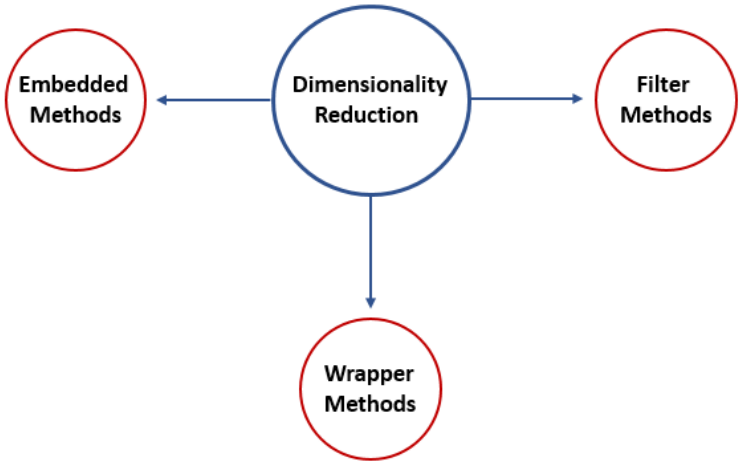


Figure 5. Literature Review outline

Filter Methods

Before the work presented in [29], there were some efforts to investigate the problem; however, these efforts were relying on selecting one SNP to address the problem. The work in [29], was the oldest observed study that incorporated multiple SNPs, following the fact that multiple SNPs might play a role in the manifestation of complex diseases. The authors, thus, ranked the features based on information gain (IG) and the top features are selected and fed to a classifier, one feature at a time. The idea is to reduce the uncertainty about the class labels, by accumulating features with high information gain. The method is applied to 332 samples (158 controls and 174 breast cancer cases) each genotyped for 245 SNPs. Multiple classifiers are used with the top selected features. The used classifiers are: Naïve Bayes (NB), decision tree, and linear SVM. The best achieved accuracy was 69% using the top 3 features with linear SVM. It is important to note that no imputation strategy was implemented. For the NB classifier, the data is used as it is, while for SVM and decision tree, only complete samples (74 controls and 63 breast cancer cases) were used. The paper proved that incorporating multiple SNPs is better than relying on a single SNP.

Similarly, the authors in [30, 31, 32] applied IG to two different datasets and evaluated multiple classifiers. On one dataset, decision tree was the best when compared to SVM, decision rules and KNN. While on the other dataset, decision rules were the best. This shows that the performance of the same filter method might vary across datasets even when used with the same classifier.

In [34], the authors suggested that ranking features using one metric might introduce some bias. Thus, they ranked the features using two metrics and the overall rank

of the feature, is the average of its ranks in both measures. The idea is that each ranking method will reduce the bias introduced by the other method. The idea was evaluated by ranking the features using IG and chi-squared. The method was tested with multiple classifiers; NB, linear and non-linear SVM, and decision tree, and compared to wrapper based methods of the same classifiers. The experiments showed that the proposed methods achieved comparable AUCs to the wrapper based methods. The best achieved AUC was using NB. However, in the wrapper based methods, 8 features only were required, while in the proposed method 12 features were required. The proposed method reduced the computational cost when compared to the wrapper based methods; however, with a slight increase in the number of required features.

With the increased number of SNPs, applying wrapper and embedded methods became impractical. In [37], relieff filter was used with a dataset of 300k features. The motivation was that relieff was never used with such large dataset and it can tolerate missing values. In relieff, a random sample is selected and the nearest neighbor from each class is found. The values of each feature in the three samples are investigated. If the feature can distinguish between the two classes, its weight is increased, and decreased otherwise. The top features produced by relieff are fed to multiple classifiers, one feature at a time. The subset achieved the best accuracy is selected. The highest achieved AUC was 0.73 with logistic regression using 10 features, and outperformed NB and SVM. Additionally, the time required to rank the features was 4400 seconds.

To address the problem of selecting redundant features in the filter methods, the authors in [41] applied the fast correlation based filter (FCBF) to 17000 samples genotyped for 500k SNPs. The algorithm of FCBF is performed in two phases. In the first, the SU is

computed between each feature and the class labels, the top *N* features are selected. In the second, the SU among all selected features is computed, the features that have approximate markov blanket are removed. The average AUC achieved by the model was 0.86, while the average number of features was 253.

In [43], the authors combined grammatical evolution (GE) and association rule mining (ARM) to select the least number of features that can predict all other features. In an iterative approach, random association rules are generated. The generated rules are assessed using the Apriori algorithm. The best features are selected and used to generate new generation. The algorithm stops when a specific number of generations is reached. The selected features are fed to a two-layer feed forward neural network. The algorithm achieved 90% of accuracy on a dataset with 111 samples.

Wrapper Methods

The authors in [30, 31, 32] presented two wrapper methods; forward selection with backtracking (FS-BT) and backward elimination with backtracking (BE-BT). The main goal is to overcome the limitations of some previous works that use only one SNP. In the FS-BT, each feature is used alone to train a specific classifier. The feature achieved the highest accuracy is selected as the first feature. Each unselected feature, is combined with the first feature separately, and used to train the classifier. The combination achieved the best accuracy is selected, if the backtracking is not enabled. If enabled, all possible combinations are considered, which increase the complexity of the algorithm. The algorithm stops when the accuracy starts to drop. In BE-BT, the algorithm starts with a classifier that is trained with all features. Each feature is removed separately, and the rest

features are used to train a classifier. The remaining subset that achieved the highest accuracy is selected, if the backtracking is not enabled. If enabled, all possible subsets are considered. The algorithm stops when there is no increase in the accuracy is achieved. Both methods were tested with multiple classifiers on two different datasets. In general, the BE-BT performed better when combined with decision tree and decision rules. Interestingly, similar performance observed if the backtracking is enabled or not.

In [33], the authors argued that the current methods in the literature can only find relevant features, and multiple redundant features might be selected. To address the problem, they proposed the supervised recursive feature addition (SRFA) method which tries to select relevant but independent features. In SRFA, to select a feature, the features are ranked based on their classification accuracy, using a specific classifier, and the top feature is selected. To select the subsequent features, each unselected feature is combined with the selected features and used to train the same classifier. The features in the subsets that achieved the best accuracy are elected as candidates features. Based on the spearman correlation coefficient, the feature that is the least statistically similar to the already selected features, is selected. The algorithm stops when there is no improvement in the accuracy. The method was tested with two datasets; the first with 31 SNPs and the second with 2300 SNPs. In addition, the performance of the method was investigated when different classifiers are used; one for feature selection and another for classification. While the best accuracy achieved by the method was on the second dataset using the same classifier, it was not the case in the first dataset.

In [33], the problem of selecting redundant features was addressed, while in [35] the robustness of the selected features was addressed. In [39], a solution that aims to

address both problems is presented. In the bag of NB (BoNB) method, *N* bootstrapped datasets are generated from the training dataset. For each generated dataset, a NB classifier is trained using each feature separately, and the Matthews correlation coefficient (MCC) is recorded for each SNP. For each NB classifier, the features are ranked based on the MCC and added one feature at a time. Each time a feature is added, the features that are correlated with the selected feature are removed from the ranked list. To classify a sample, all NB classifiers are used and the resulting predictions are averaged, to produce a weighed prediction. The method is applied to 5000 samples genotyped for 500k SNPs. The experiments showed that the method is performing better than a single NB classifier and can be compared to other methods in the literature in terms of AUC.

The work presented in [40] is aiming to reduce the computational cost of wrapper methods. To that end, two techniques were used. The first is to use regularized least squares (RLS) classifier. RLS is similar to SVM it terms of its outcomes. However, it can reduce the computational cost through some matrix algebra optimization. The second is to parallelize the code to work on 4 CPU cores. The method is applied to 3300 samples genotyped for 500k SNPs. The highest achieved AUC was 0.9 using 21 features, more importantly, the algorithm required only 5 minutes to run.

Embedded Methods

In [30, 31, 32], the authors observed a difference in the performance between a decision tree classifier and its equivalent decision rules. They also observed that not all features are used in the decision rules. Thus, they proposed an iterative rule based feature selection (RFS). In each iteration, a decision tree classifier is trained and converted to

decision rules, the features are not used by the rules are eliminated. The remaining features are used again to train new decision tree classifier. The algorithm stops when all features used to train the decision tree classifier are used by the rules.

To avoid selecting redundant features, the work presented in [33] embedded the weights of the SVM in the learning process and used a statistical measure to select non-redundant features. In the proposed algorithm; the support vector based recursive feature addition (SVRFA), the first selected feature, is the feature that has the minimum squared weights vector when used alone to train a SVM classifier. Each unselected feature is combined with the already selected feature and used to train a SVM classifier. The unselected features within the subsets achieved the lowest squared weights vector is elected as candidates features. The feature that is the least statistically similar to the already selected features is selected. The method was tested with two datasets; the first with 31 SNPs and the second with 2300 SNPs, and was the best on the first dataset.

In [35], the authors are concerned with stability and robustness of the feature selection methods. They suggested that in the older methods, a small change in the dataset might cause a big difference in the selected features, which affects the reproducibility and the validation of the selected features. Thus, to ensure the robustness of the selected features, multiple bootstraps are generated from the original dataset. Each bootstrap is 10% different from the original dataset. In addition, a SVM recursive feature elimination (RFE) is applied to each bootstrap. RFE starts with the full features set, on each iteration the least important feature is eliminated. The importance of a feature is measured by its absolute weight in the separating hyperplane of SVM. Each bootstrap produces different ranking for the features. To aggregate the produced ranking, two methods are used. The first is to

average the ranking of all bootstraps. The second is to average the ranking of all bootstraps; however, each ranking is given a weight based on the AUC achieved by the bootstrap. The method was applied to four datasets, and the proposed method achieved a 30 % increase in robustness based on the Kuncheva index (KI) and 15% increase in the AUC, when compared to a SVM RFE without bootstrapping.

In [36], the authors argued that the current methods in the literature can be used with datasets where the number of features is in the thousands; however, the effectiveness of these methods for datasets where the number of features is in the100 thousands, is not validated. On the other hand, random forests (RF) are showing excellent performance in other tasks like pattern recognition. Using RF; however, with large number of features is impractical. Thus, to reduce the computational cost of RF, the authors generated one million trees and the gini importance of each feature in these trees is averaged. The features are then ranked based on the gini importance and fed to a RF classifier by adding one feature at a time. The subset achieved the highest classification accuracy is selected. The method was tested with a dataset with over 100k features and achieved minimum classification error of 8.5% using 84 features.

The nearest shrunken centroid is a classification method that is useful for large-scale datasets where the features are continuous. SNPs values; however, are categorical. In [38], a modified version of the method that is suitable for categorical features is presented. The presented algorithm is performed in two phases; feature selection and classification. In the first phase, three distribution vectors are computed for each SNP; overall and one for each class. The Euclidian distance between the distribution vectors of each class and the overall vector is computed. If the both computed distances are lower than a given

threshold, the feature is dropped. In the second phase, for each class, a centroid is computed, which consists of the most frequent value of each SNP within the class. To classify an unknown sample, the distance between the sample SNPs and the centroid of each feature is computed. The nearest class is predicted as the sample class. The method was applied to a dataset with 3500 samples genotyped for 500k SNPs. The experiment showed that the method achieved 87% of classification accuracy using 221 features.

To address the same problem in [36], the authors in [42] applied a feature selection phase before feeding the features to the RF. In the feature selection phase, *N* random forests are generated. In each tree, a shadow SNP that has no predictive power is inserted. The importance of each SNP in all tress is measured by averaging the gini importance. The Wilcoxon signed rank test was used to compare the importance of the real SNPs and the shadow SNP. All real SNPs with p-value below a threshold are selected. Chi-squared test is then applied to the selected SNPs, and the SNPs are divided into two groups; high informative and weak informative. When building the RF for classification, SNPs from both groups are selected with specific percentages. The proposed method is applied to two datasets and achieved an average of 90% of classification accuracy. The method outperformed other RF methods in the literature.

Comparison

In this Section, the reviewed studies are compared. The comparison is presented in Table 1, while the comparison criteria are shown in the following:

1. Type of the methods presented in the study: filter (F), wrapper (W), embedded (E)
2. The Number of real datasets used in the study.
3. Number of samples in the real dataset, if multiple datasets are used, then the average number is reported.
4. The number of SNPs in the dataset, if multiple datasets are used, then the average number of SNPs is reported.
5. Classifiers: a list of the used classifiers: SVM (S), NB (N), RF (R), decision tree (D), decision rules (DR), KNN (K), RLS, Others (O)
6. Is cross-validation used: Yes or No?
7. Is an imputation method reported: Yes or No?
8. Is the classification accuracy reported: Yes or No?
9. Is sensitivity and specificity reported: Yes or No?
10. Is the AUC reported: Yes or No?
11. Is the paper concerned about the computational cost: Yes or No?

Table 1

Comparison of the Reviewed Papers

Criterion	Reference												
	29	30,31,32	33	34	35	36	37	38	39	40	41	42	43
1	F	F, W, E	W, E	F	E	W	F	E	W	W	F	E	F
2	1	2	2	1	4	1	1	1	1	1	7	2	1
3	332	155	1074	109	204	146	1411	3492	4901	3382	2428	452	111
4	245	28	1165	42	4797	116K	500k	500k	500k	500k	500k	390k	42
5	S, N, D	S, D, DR, K	S, K, O	S, N, D	S	R	S, N, O	O	N	RLS	O	R, O	O
6	Y	Y	Y	Y	N	N	Y	Y	N	N	Y	Y	N
7	N	N	N	Y	N	N	N	N	N	N	Y	N	N
8	Y	Y	Y	N	N	Y	N	Y	N	N	N	Y	Y
9	Y	Y	N	Y	Y	N	Y	N	N	N	Y	N	N
10	N	N	N	Y	Y	N	Y	N	Y	Y	Y	Y	N
11	N	N	N	N	N	Y	Y	N	N	Y	Y	Y	N

Discussion

While conducting the literature review, it was observed that the addressed challenges, main concerns, and used classifiers in the reviewed studies are varying over time. These observations guided some of the choices in the proposed solutions in this thesis.

In the early reviewed studies [29-32], the main motivation was the fact that multiple SNPs are working in coordination to manifest complex diseases. The research studies pursuit this fact by incorporating machine learning methods because of its ability to deal with uncertainty. However, predicting the existential status of a given disease was not enough to quench the researchers' thirst; it was also important to identify which of these SNPs is related to the given disease. This challenge was merely addressed by ranking the features using statistical measures or searching for the subset that gives the best classification accuracy in a hill climbing fashion. While multiple classifiers were tested, SVM and decision trees were the best, and KNN was the least successful. Given that the number of samples and SNPs is in the hundreds or less, the computational cost of the methods was not an issue.

When the number of SNPs and samples became in the thousands [33-35], two main challenges faced the previous methods. The first challenge is that many redundant features might be selected, which was addressed by considering the statistical similarity between the selected SNPs, or ranking the features using multiple metrics. The second challenge is that small changes in the dataset lead to selecting different SNPs, which was addressed by using bootstrapping techniques and assessing the robustness of the selected features. In

addition to SVM and decision tree, NB was showing a comparable performance, and the doubts about KNN were confirmed.

SNPs microarrays were introduced and the number of SNPs became in the hundred thousand [36-42]. Computational cost became an issue, and some studies [36,37,40,41,42] were mainly concerned about it. The issue was addressed by advanced multi-stage filters, preceding RF with a feature selection stage, and code parallelism. Other studies [38,39], were concerned about selecting robust independent features, and tailoring well-known algorithms for SNPs data. SVM and NB were still performing well. Decision tree; however, was replaced by RF, and KNN was never appeared.

In the context of the literature review, the following choices were made:

- The proposed solution should be in filter form to overcome the computational cost issue. However; redundant features should be taken into account.
- SU will be used as the baseline because it performed well in the reviewed studies.
- CE will be investigated because it was never used for the same purpose.
- SVM is selected because it was the most used classifier and performed well in multiple studies.
- KNN is selected because it was not used in the more recent studies.

CHAPTER 4: RESEARCH METHODOLOGY

In this Chapter, the methodology of the proposed solution is presented. This includes a description of the used dataset in terms of the number of samples and SNPs for the cases and controls, and the transformations that were applied to it. In addition, the quality control criteria, performed imputation, and codifying strategy were discussed. Moreover, the theories behind the dimensionality reductions in the baseline and the used classifiers, are explained. Finally, the proposed scoring scheme and features ranking methods are explained in details. For each, an algorithm is given, and an example whenever applicable. Figure 6 shows an overview of the methodology.

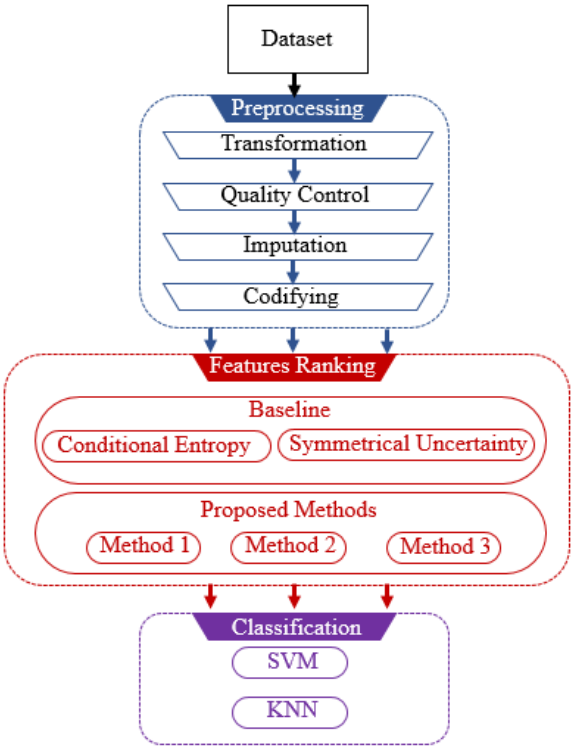


Figure 6. Methodology Overview

Dataset

The dataset used in this thesis is the Wellcome Trust Case Control Consortium (WTCCC1) dataset [24]. The dataset contains around 14000 cases for seven diseases and around 1500 controls genotyped for 500k SNPs using the Affymetrix 500k microarray [44]. Table 2 shows the details of the dataset.

Table 2

Dataset Details

Dataset	Number of samples	Number of features	Number of classes
1958 British Birth Cohort samples (Controls)	1504	500,568	1
Bipolar Disorder (BD) samples	1998	500,568	1
Coronary Artery Disease (CAD) samples	1998	500,568	1
Inflammatory Bowel Disease (IBD) samples	2005	500,568	1
Hypertension (HT) samples	2001	500,568	1
Rheumatoid arthritis (RA) samples	1999	500,568	1
Type 1 Diabetes (T1D) samples	2000	500,568	1
Type 2 Diabetes (T2D) samples	1999	500,568	1

Dataset Preprocessing

This Section describes how the dataset was transformed to be suitable for the Scikit-Learn [45] machine learning library.

Transformation

The obtained dataset consists of eight directories. Each directory corresponds to one class in Table 2, and contains 23 files. Each file, contains the genotyping of SNPs in a specific chromosome for all samples in the class. The size of the dataset was 253 GB. Figure 7 shows a snapshot of one of the files. The files were in TAB delimited format and the order of the columns was: *SNP ID, SAMPLE ID, SNP VALUE, CONFIDENCE*. It is important to note that the *NN* SNP value represents a missing value.

After exploring the files, a lot of redundancy was observed. The *ID* of each sample was repeated a number of times equal to the number of SNPs. In addition, the *ID* of each SNP was repeated a number of times equal to the number of samples. The following transformations were applied to each class files.

- Samples *IDs* and SNPs *IDs* are extracted and kept in separate files.
- The SNPs values of each sample are appended in one row. The order of the rows is the same as in the samples *IDs* file, while the order of the SNPs is the same as the order in the SNPs *IDs* file.

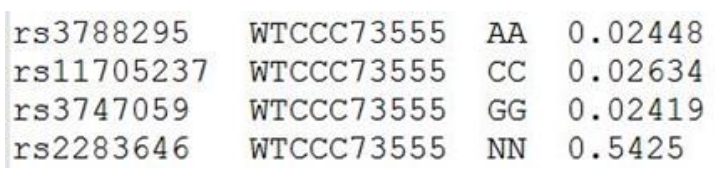


Figure 7. Snapshot of one of the files

After applying the transformations, SNPs values of each class were combined in one TAB delimited file. Each row contains the values of all SNPs of one sample, while each column contains the values of one SNP for all samples in the class. It is worth noting that the confidence value was not maintained, because it will not be used in the classification task. The size of the dataset was reduced to 17.8 GB.

Quality Control

The original paper of the dataset [24] excluded some samples from the study, these samples were excluded. In addition, samples and SNPs with more than 20% of missing data were also excluded. Specifically, 804 samples and 31,216 SNPs were excluded.

Imputation

After applying quality control, the observed percentage of missing values in the dataset was 0.81. These missing values were replaced with the most frequent value in the same SNP. In other words, if sample *X* from class *Y* has a missing value of the SNP *Z*. The missing value is replaced with the most frequent value of SNP *Z* in all samples in class *Y*.

Codifying

As shown in Figure 7, each SNP value is represented in two letters. There are four possible letters *A, C, T*, and *G*. Thus, there are 16 possible combinations. Each of these combinations and the corresponded numerical value are shown in Table 3.

Table 3

Codifying Strategy

SNP Value	Numerical Value	SNP Value	Numerical Value
AA	1	CA	9
AG	2	CG	10
AC	3	CC	11
AT	4	CT	12
GA	5	TA	13
GG	6	TG	14
GC	7	TC	15
GT	8	TT	16

Merging and Class Labels Generation

The problem is formulated as a binary classification problem. To generate datasets suitable for the binary classification problem and compatible with Scikit-Learn library, the following steps were applied.

- The control samples were merged with each of diseases samples in Table 2. The resulting of this is seven files; one for each disease.
- For each file, a separate file is generated for the class labels. In each file, the controls samples assigned the class -1 while the disease samples assigned the class 1.

Baseline

This section describes the dimensionality reduction techniques and the classifiers used in the baseline. It is worth noting that the both used dimensionality reduction techniques are filter methods and based on information theory measures.

Dimensionality Reduction

Symmetrical Uncertainty

The first selected dimensionality reduction technique for the baseline is ranking features based on SU [46]. SU was selected because it was part of a method presented in [41] that achieved high classification accuracy on the WTCCC1 dataset. The SU between two variables *X* and *Y*, is a value in the range [0,1], where 0 indicates that the two variables are completely independent, and 1 indicates that one variable can be completely predicated by the other variable. In fact, the SU is built on top of the Mutual Information (MI), which can be described as the reduction in entropy of a variable, achieved by knowing another variable [46]. The SU of two variables *X* and *Y* is given by the following equation, where MI is the mutual information and *H* is the entropy.

$$SU(X,Y) = 2\left[\frac{MI(X,Y)}{H(X)+H(Y)}\right] \tag{9}$$

Conditional Entropy

The second selected dimensionality reduction technique for the baseline is ranking features based on CE [47]. The reason behind the selection of CE is that, up to the author's

knowledge, it was never used as feature ranking method with SNPs data in classification problems. The CE of variable *Y* given the variable *X*, can be described as the amount of information required to describe the variable *Y* knowing the variable *X*.

Classifiers

Two classifiers were used in the baseline, which are described in the following.

K-Nearest Neighbors

In the KNN classifier [48], training samples are represented as points in the space. To classify a test sample *Z*, the Euclidean distances between the test sample and all training samples are computed. The most frequent class in the *K* nearest training samples is predicted as the class of the sample *Z*. In Figure 8, the unknown sample is assigned the class *A* based on *K* equal to four. Several *K* values were tested on a small subset and eleven was the best. Thus, in this thesis, *K* is equal to eleven because

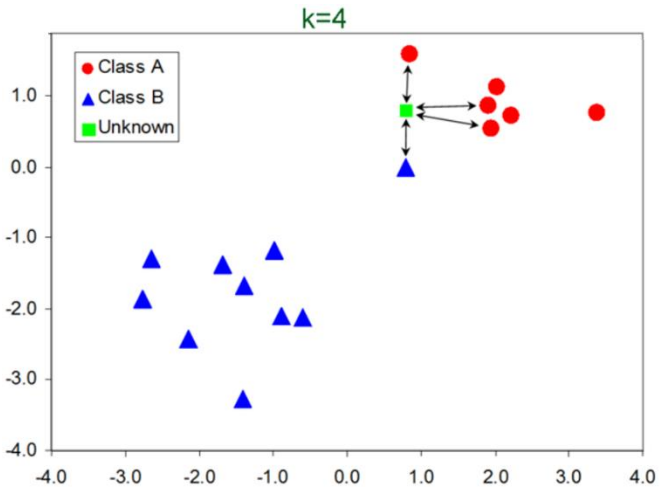


Figure 8. Toy example of KNN [49]

Support Vector Machine

In SVM [50], the goal is to find an optimal hyperplane the separates the two classes of the training samples. A test sample is assigned a class based on its relative position to the separating hyperplane. An optimal hyperplane, as shown in Figure 9, is the hyperplane that correctly separates all training samples to its either sides, and correctly categorize all the training examples. In addition, the optimal hyperplane, should maximize the margins with nearest training samples of both classes. The nearest training samples to the hyperplane are called the support vectors, while the shape of the hyperplane is called the kernel, which can be linear, polynomial or other basic functions. Figure 9 shows a toy example of linear SVM with three support vectors. In this thesis, a linear kernel is used and the default parameters values of the *LinearSVC* classifier in the Scikit-Learn library are used. The values of these parameters are; penalty parameter *C=1*, loss function *loss=squared_hinge*, penalization norm *penalty=l2*, solve the dual optimization problem *dual=True*, tolerance for stopping criteria *tol=1e-4*, no class weights are used *class_weight=None*, and the algorithm is set to maximum 1000 iterations *max_iter=1000*.

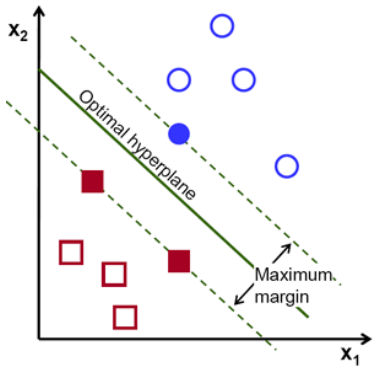


Figure 9. Toy example of linear SVM [51]

Proposed Scoring Scheme

The hypothesis behind the proposed scoring scheme is that specific values within a feature might be useful in predicting specific class labels, while other values are not. In this context, each group of similar values within a feature are considered a region as shown in Figure 10. It is clear from the toy example shown in Figure 10 that whenever the value of feature 1 is *A*, the corresponded class label is almost 1 in all samples. While whenever the value of the feature is *C*, the corresponded class label is always zero. However, samples with the value *A*, are more frequent than samples with value *C*. Thus, the proposed scoring scheme must consider the usefulness of the region, and the region's significance within the feature. It is important to note that the scoring scheme only computes scores and assign class labels for the regions, and further steps are still required to select the best features. The following describes the proposed scoring scheme for one feature, while a pseudo code of the scheme is presented in Algorithm 1.

For each region, a score that is ranging from zero to one is computed and a class label is assigned. The assigned class, is the class that the region is useful for. A positive score means that the region is more informative for one of the class labels in a binary classification problem. The higher the score the more informative the region for a specific class label. Each region's score is normalized to the length of the region within the feature.

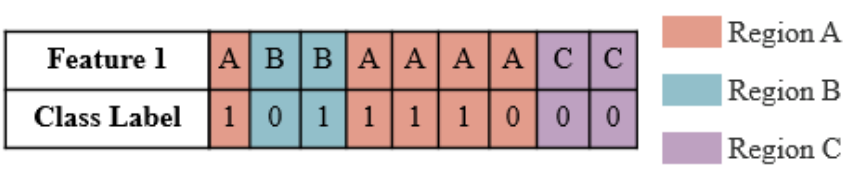


Figure 10. Toy example of regions

Algorithm 1 Pseudo code of the scoring scheme

Input **Vector *V*** of the values of one feature, categorical values
Vector *T* of the class label of each sample, class label can be 0 or 1

Output **Vector *O*** of tuples, each tuple in the form (n, s, c), where n is the region name, s the score of the region, and c is the assigned class label.

```
1:  UniqueValues ← FINDUNIQUEVALUES (V)
2:  Regions ← new list, O ← new list
3:  for each UniqueValue u ∈ UniqueValues do
4:      Temp ← FINDCLASSLABELS (u, T)
5:      add (Regions, (u, Temp))
6:  end for
7:  for each Region r (value, labels) ∈ Regions do
8:      ClassZeroCount ← COUNT (0, r (labels))
9:      ClassOneCount ← COUNT (1, r (labels))
10:     MaxCount ← max (ClassZeroCount, ClassOneCount)
11:     MinCount ← min (ClassZeroCount, ClassOneCount)
12:     s ← (MaxCount – MinCount) / length (V)
13:     c ← FINDMOSTFREQCLASS (r (labels))
14:     add (O, (r (value), s, c))
15: end for
```

In Algorithm 1, in line 1, the unique values within the feature are found. In lines 3-6, the correspondent class labels for each unique value are found and stored in the array *Regions*. Lines 7-15 are repeated for each region in *Regions* array. In lines 8-9, the frequency of each class label in the region is computed. The class label with maximum frequency, and the class label with the minimum frequency are identified in lines 10-11. In line 12, the score of the region is computed by subtracting the frequency of the class with minimum occurrences from the frequency of the class with maximum occurrences. The result of the subtraction is divided by the length of the feature vector. In line 13, the class of the max frequency is assigned as the class label of the region.

The perfect region based on the scoring scheme, is the region that contains only one of the class labels in its correspondent class labels. In other words, whenever the region value is observed within the feature, the class label of the sample will always be the same. This indicates that the region is useful for identifying samples from a specific class label. Figure 11 shows how the toy example in Figure 10 is scored.

Length (feature 1) = 9
Region A:
Region A= (1, 1, 1, 1, 0)
Freq (1) = 4, Freq (0) = 1 => assigned class label = 1
Score (A) = ((4 -1)/9) = 0.33
Region B:
Region B= (0,1)
Freq (1) = 1, Freq (0) = 1 => assigned class label = any
Score (B) = ((1 -1)/9) = 0
Region C:
Region C= (0,0)
Freq (1) = 0, Freq (0) = 2 => assigned class label = 0
Score (C) = ((2-0)/9) = 0.22

Figure 11. Example of the scoring scheme

Feature Ranking Method 1

After the scoring scheme is applied to all features, the first proposed heuristic to rank the features, is to sum the scores of all regions within each feature, and rank the features in descending order based on the summed score. The top *N* features are then selected and fed to a classifier by adding one feature at a time and the accuracy of the classifier is recorded.

The rationale behind this method is that the perfect feature will achieve a summed score of 1, while the worst feature will achieve a summed score of zero. When the summed score of a feature is 1, the regions within this feature can perfectly distinguish between the class labels. In other words, within each region, there is only one class label. Figure 12 shows how the total score for the example in Figure 10 is computed, while the pseudo code of the method is given in Algorithm 2.

Length (feature 1) = 9
Region A:
Region A= (1, 1, 1, 1, 0)
Freq (1) = 4, Freq (0) = 1 => assigned class label = 1
Score (A) = ((4 -1)/9) = 0.33
Region B:
Region B= (0,1)
Freq (1) = 1, Freq (0) = 1 => assigned class label = any
Score (B) = ((1 -1)/9) = 0
Region C:
Region C= (0,0)
Freq (1) = 0, Freq (0) = 2 => assigned class label = 0
Score (C) = ((2-0)/9) = 0.22

Score (feature 1) = **(0.33 + 0 + 0.22) = 0.55**

Figure 12. Example of method 1

Algorithm 2 Pseudo code of method 1

Input **Vector *F*** of tuples, each tuple in the form (*fid*, (s_1, s_2, … s_i)), where *fid* is the feature id and s_1 to s_i are the scores of the regions in the feature.

N the number of the top features to be retrieved.

Output **Vector *O*** of *N* ids

1: TotalScores ← new list, O ← new list

3: **for** each Feature (fid, (s_1, s_2, … s_i)) f ∈ F **do**

4: **add** (TotalScores, (fid, **sum** (s_1, s_2, … s_i)))

5: **end for**

6: SortedTotalScores ← SORTDESCENDING (TotalScores)

7: **for** each SortedTotalScore (fid, TotalScore) s ∈ SortedTotalScores **do**

8: **add** (O, fid)

9: **if** length **(**O) == N

10: **Break**

11: **end for**

Feature Ranking Method 2

There is a predefined set of values that a feature can take on the problem under investigation of this thesis. Therefore, the second proposed heuristic, is to select the features that contain the highest scoring region for each possible region. The selected features are then sorted in descending order based on the score of the highest scoring region within each feature. The selected features are then fed to classifier one feature at time and the accuracy is recorded. It is worth mentioning that this method selects a number of features that is less than or equal to the number of possible values of the features.

In method 1, the values of the features that achieved the highest total scores are not investigated. In the top *N* selected features, some of the possible values might never appear. Thus, the rationale of method 2 is to consider the significance of each possible value.

There are two reasons behind why method 2 may select features less than the number of possible values. The first, some possible values may never appear in the dataset. The second, the same feature may contain more than one highest scoring region.

The pseudo code of method 2 is given in Algorithm 3. The presented algorithm scans the entire scored features only once. A naïve implementation may scan the entire scored features more than that. In line 3, the dictionary is initialized with empty tuples. The first entry of each tuple will hold a score of region, while the second entry of each tuple will hold a feature *id*.

Algorithm 3 Pseudo code of method 2

Input

Vector *F* of tuples, each tuple in the form (*fid*, (s_1, s_2, … s_i), (r_1, r_2, … r_i)), where *fid* is the feature id and s_1 to s_i are the scores of the regions in the feature, and r_1 to r_i are the names of the regions

Vector *P* of all possible values of the features.

Output **Vector *O*** of *N* ids where *N* <= length of (*P*)

```
1:  HighestScoringRegions ← new dictionary, O ← new list
2:  for each PossibleValue p ∈ P do
3:      HighestScoringRegions [p] ← (ϕ, ϕ)
4:  end for
5:  for each Feature (fid, (s1, s2, … si), (r1, r2, … ri)) f ∈ F do
6:      for i repetitions do
7:          if HighestScoringRegions [ri] [0] < si
8:              HighestScoringRegions [ri] [0] = si
9:              HighestScoringRegions [ri] [1] = fid
10:     end for
11: end for
12: SortedRegionsScores ← SORTDESCENDING (HighestScoringRegions)
13: for each SortedRegionsScore (s, fid) sr ∈ SortedRegionsScores do
14:     add (O, sr [1])
15: end for
```

Feature Ranking Method 3

The problem is formulated as a binary classification problem. This means that the class label assigned to any region can only be one of two classes. However, within the same feature, multiple regions might be assigned the same class label. Thus, the third proposed heuristic is to sum the scores of all regions within the same feature that assigned the same class label. The result of this, is that two scores are computed for each feature; one for each class label. After that, the features are ranked in descending order based on the computed score of each class label within the feature in two separate lists. From each list, the top N features are selected and fed to a classifier one feature from each list at a time, and the classification accuracy is recorded.

In method 1 and method 2, the class assigned to the regions were not taken into account. This may result in an unbalanced selection of features. In other words, it could be possible that most of the selected features contain high scoring regions for one class only. Thus, the rationale behind method 3 is to ensure balanced selection of features for each class label. Figure 13 shows how the score for each class label is computed for the example in Figure 10, while the pseudo code of the method is given in Algorithm 4.

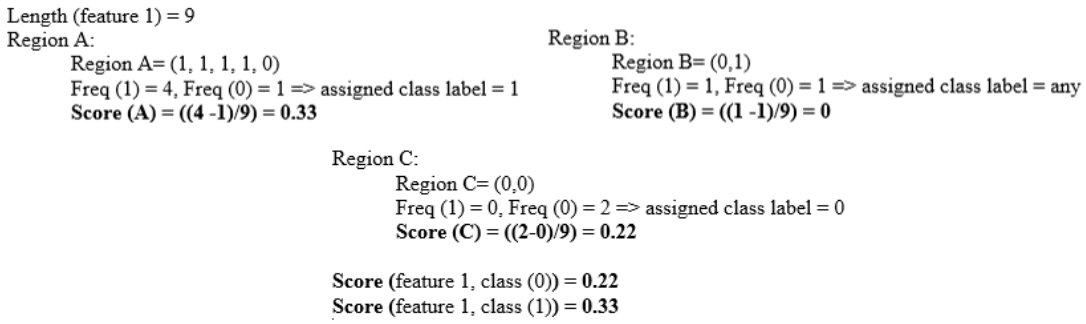


Figure 13. Example of method 3

Algorithm 4 Pseudo code of method 3

Input

Vector *F* of tuples, each tuple in the form (*fid*, (s_1, s_2, … s_i), (c_1, c_2, … c_i)), where *fid* is the feature id and s_1 to s_i are the scores of the regions in the feature, and c_1 to c_i are the class labels assigned to the regions.

N the number of top the features to be retrieved.

Output **Vector *O*** of *N* ids

1: ClassOneScores, ClassZeroScores, O ← new dictionary

2: **for** each Feature (fid, (s_1, s_2, … s_i), (c_1, c_2, … c_i)) f ∈ F **do**

3: ClassOneScores [fid] ← SUMSCORES (1, (s_1, … s_i), (c_1, … c_i))

4: ClassZeroScores [fid] ← SUMSCORES (0, (s_1, … s_i), (c_1, … c_i))

5: **end for**

6: SortedClassOneScores ← SORTDESCENDING (SortedClassOneScores)

7: SortedClassZeroScores ← SORTDESCENDING (SortedClassZeroScores)

8: Counter ← 1

9: **while** length (O) < N **do**

10: **add** (O, SortedClassZeroScores [Counter])

11: if length (O) == N

12: **Break**

13: **add** (O, SortedClassOneScores [Counter])

14: Counter++

15: **end while**

CHAPTER 5: EXPERIMENTAL RESULTS

Experiments Setup

As mentioned in Chapter 4, there are seven sub-datasets generated from the obtained WTCCC1 dataset. Each of these sub-datasets is suitable for binary classification. It is important to note that all experiments were conducted on RAAD [52] high performance computing platform. For each of the seven sub-datasets, 10 experiments were conducted. For each experiment, a job was submitted to RAAD. Each job used 16 CPUs and 24 GB of memory. Each sub-dataset was split with 75 percent for train and 25 percent for testing. Table 4 shows the details of each of the 10 experiments.

In each experiment, the highest achieved classification accuracy, the number of features required to achieve this accuracy, and the time required to rank the features are recorded. For the baseline and method 1 experiments, the top 600 features are selected and fed to the classifier one feature at a time. For method 2 experiments, the features selected by the method are fed to the classifier one feature at a time based on the ranking produced by the method. For method 3 experiments, the top 300 features are selected from each list and fed to a classifier two features at a time. In all experiments, the accuracy achieved at each step is plotted against the number of used features.

Table 4

Experiments Details

	Experiment #	Dimensionality Reduction	Classifier
Baseline	1	SU	KNN
	2	SU	SVM
	3	CE	KNN
	4	CE	SVM
Method 1	5	Method 1	KNN
	6	Method 1	SVM
Method 2	7	Method 2	KNN
	8	Method 2	SVM
Method 3	9	Method 3	KNN
	10	Method 3	SVM

Results

In Table 5, the average of the results of each experiment on the seven sub-datasets are shown. It is important to note that the computed averages are for the maximum achieved accuracy, the number of features required to achieve that accuracy, and the time required to rank the features. While Tables 6 – 10 show the detailed results of each method, Figures 14 – 33 show the plots of the experiments

Table 5

Average of Maximum Accuracy, Number of Features and Running Time of the Baseline and the Proposed Methods

Dimensionality Reduction	Classifier	Accuracy	# Features	Time (Sec)
SU	KNN	81.1	128	4042
SU	SVM	87.7	553	
CE	KNN	81.0	54	**2497**
CE	**SVM**	**87.8**	**414**	
Method 1	KNN	80.2	57	1464
Method 1	SVM	86.7	229	
Method 2	KNN	77.9	11	1514
Method 2	SVM	78.6	10	
Method 3	KNN	80.3	30	**1494**
Method 3	**SVM**	**86.8**	**218**	

Table 6

The Maximum Accuracy, Number of Features and Running Time of Symmetrical Uncertainty

	Symmetrical Uncertainty				
	KNN		**SVM**		
	Max Accuracy	Number of Features	Max Accuracy	Number of Features	**Time(s)**
BD	79.7	76	88.8	573	4009
CAD	84.3	117	88.9	454	4002
IBD	79.5	87	86.1	599	4008
HT	72.6	20	81.1	568	4048
RA	90.6	48	95.2	503	4022
T1D	84.1	393	90.8	594	4088
T2D	77.5	155	83.5	583	4123

Table 7

The Maximum Accuracy, Number of Features and Running Time of Conditional Entropy

	Conditional Entropy				
	KNN		**SVM**		
	Max Accuracy	Number of Features	Max Accuracy	Number of Features	**Time(s)**
BD	79.3	32	89.9	547	2549
CAD	82.4	58	88.4	267	2458
IBD	79.3	8	84.9	449	2461
HT	75.6	135	83.1	494	2485
RA	89.6	41	95.6	336	2490
T1D	84.1	63	90.1	494	2515
T2D	76.8	42	82.9	315	2522

Table 8

The Maximum Accuracy, Number of Features and Running Time of Feature Ranking Method 1

	Method 1				
	KNN		**SVM**		
	Max Accuracy	Number of Features	Max Accuracy	Number of Features	**Time(s)**
BD	81	74	87	359	1472
CAD	84.2	19	85.6	26	1445
IBD	73.5	113	84.5	325	1447
HT	73.6	54	83.1	221	1473
RA	89.3	11	93.5	107	1458
T1D	81.8	109	89.8	308	1466
T2D	78.3	22	83.6	261	1488

Table 9

The Maximum Accuracy, Number of Features and Running Time of Feature Ranking Method 2

	Method 2				
	KNN		**SVM**		
	Max Accuracy	Number of Features	Max Accuracy	Number of Features	**Time(s)**
BD	80.4	12	86.6	10	1501
CAD	83.1	12	82.6	13	1509
IBD	70.7	7	69.7	5	1491
HT	71.3	14	71.3	11	1544
RA	87.9	15	87.3	11	1526
T1D	78.4	12	77	13	1500
T2D	73.5	6	76.2	13	1530

Table 10

The Maximum Accuracy, Number of Features and Running Time of Feature Ranking Method 3

	Method 3				
	KNN		**SVM**		
	Max Accuracy	Number of Features	Max Accuracy	Number of Features	**Time(s)**
BD	81.3	16	88.3	118	1483
CAD	82.7	20	85.7	173	1465
IBD	74.2	23	85.5	320	1471
HT	74.1	24	82.4	245	1523
RA	89.6	11	93.5	107	1514
T1D	81.8	109	89.8	308	1494
T2D	78.6	7	82.9	257	1512

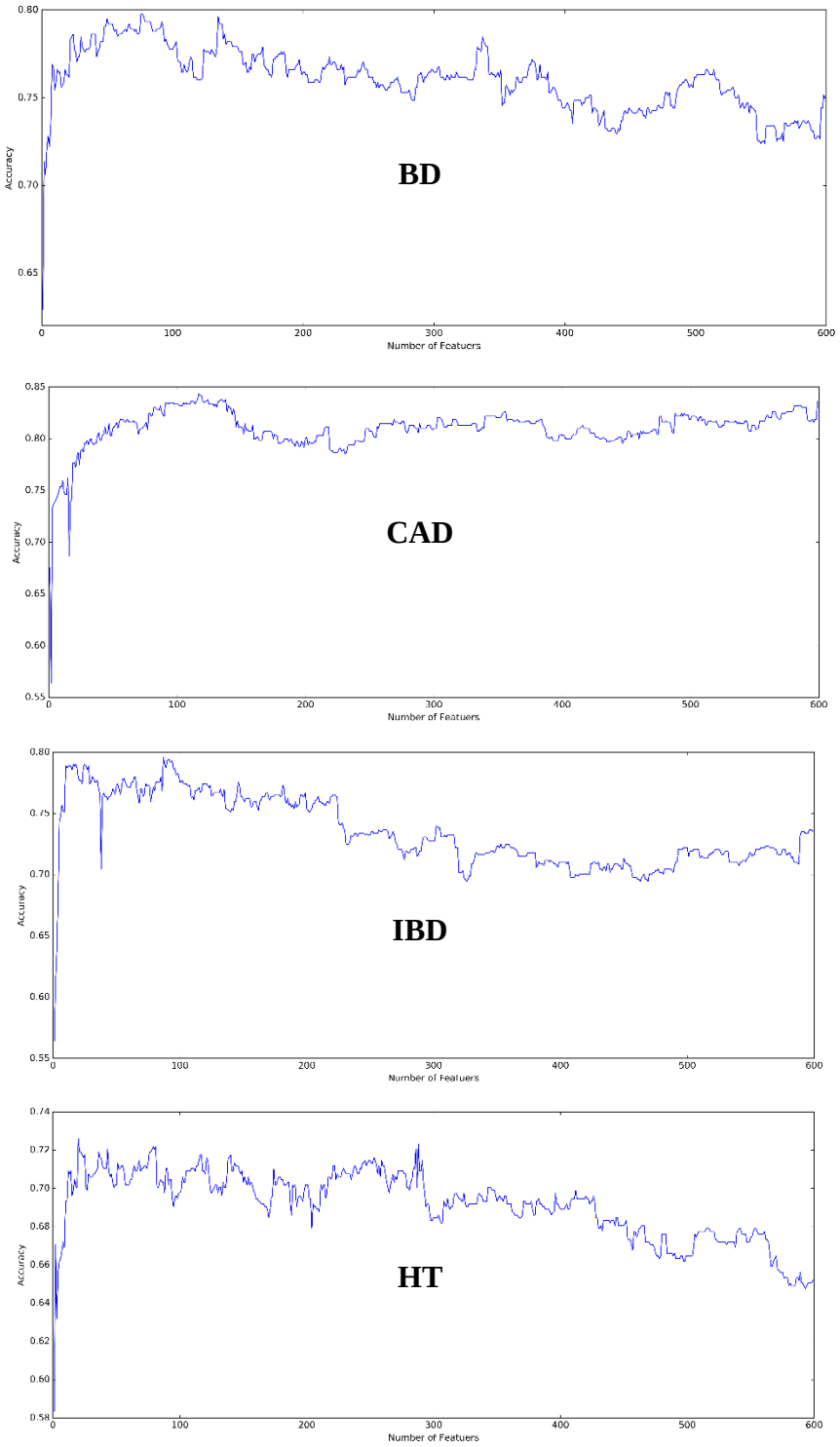
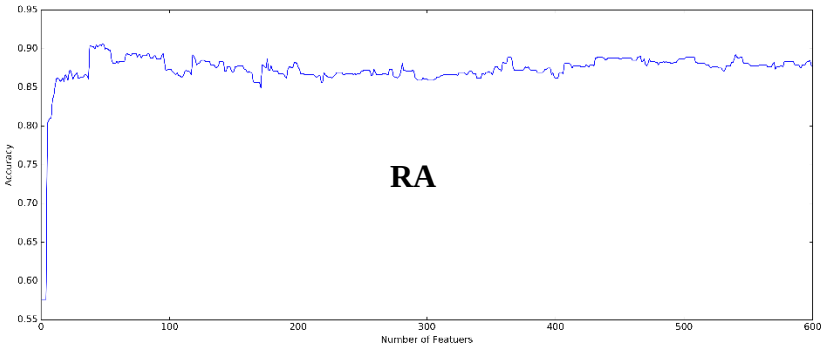
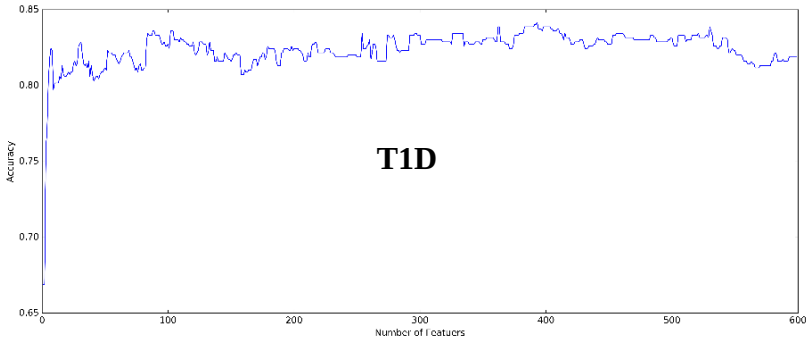


Figure 14. Plots of SU with KNN





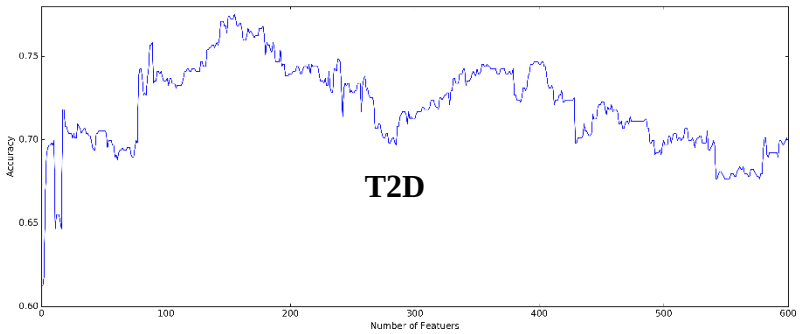


Figure 15. Plots of SU with KNN (cont)

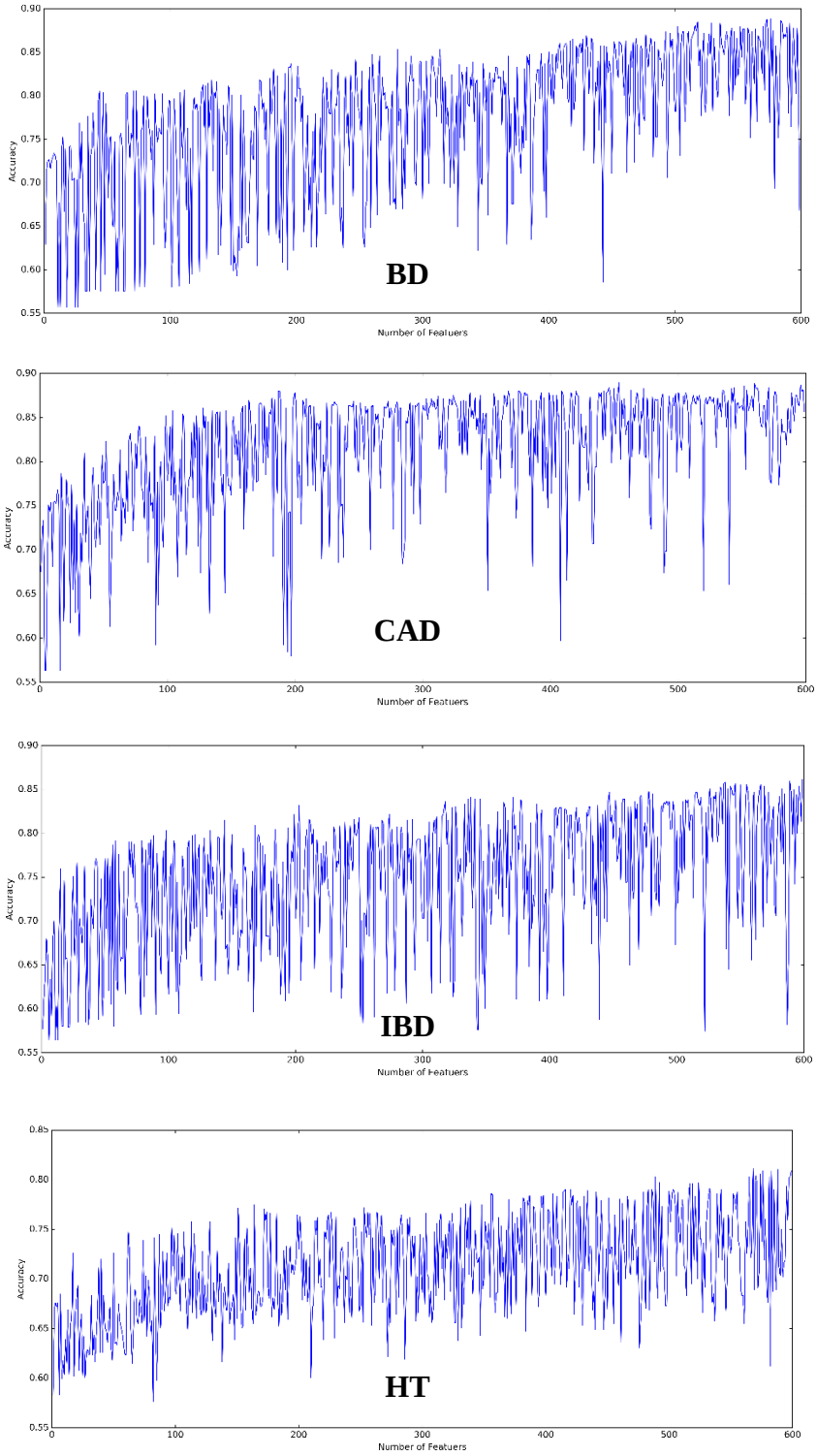


Figure 16. Plots of SU with SVM

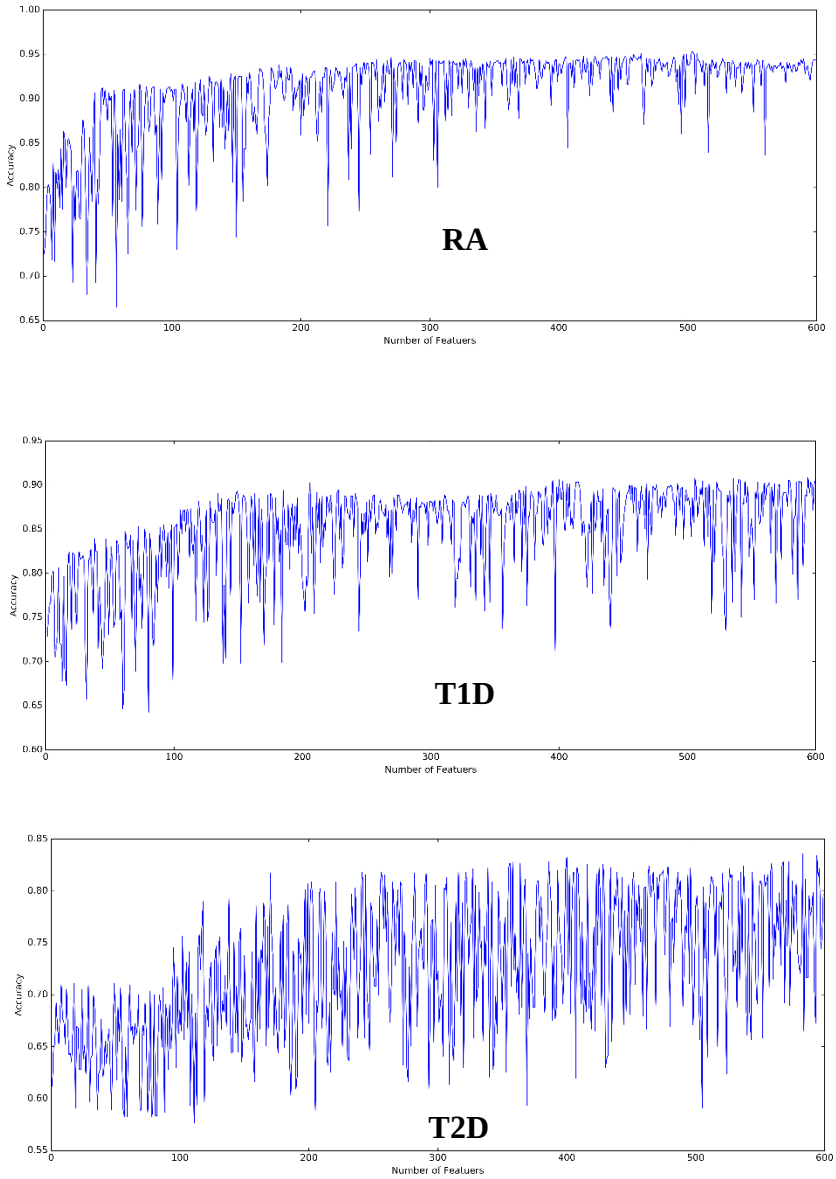


Figure 17. Plots of SU with SVM (cont)

The results show that SU is always performing better in terms of accuracy when used with SVM, the differences are between 5% and 10%. However, the average number of features used by KNN is less.

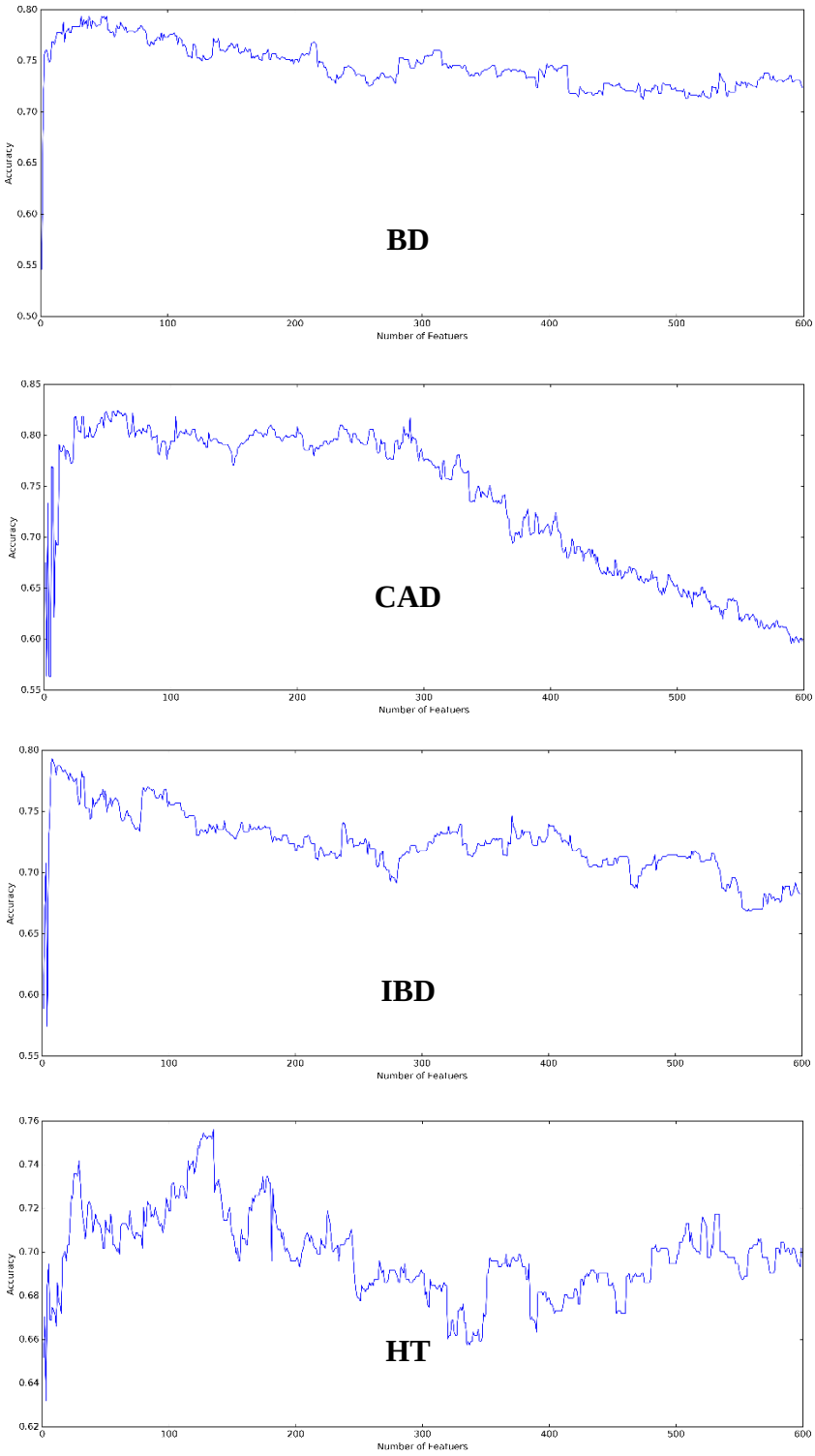


Figure 18. Plots of CE with KNN

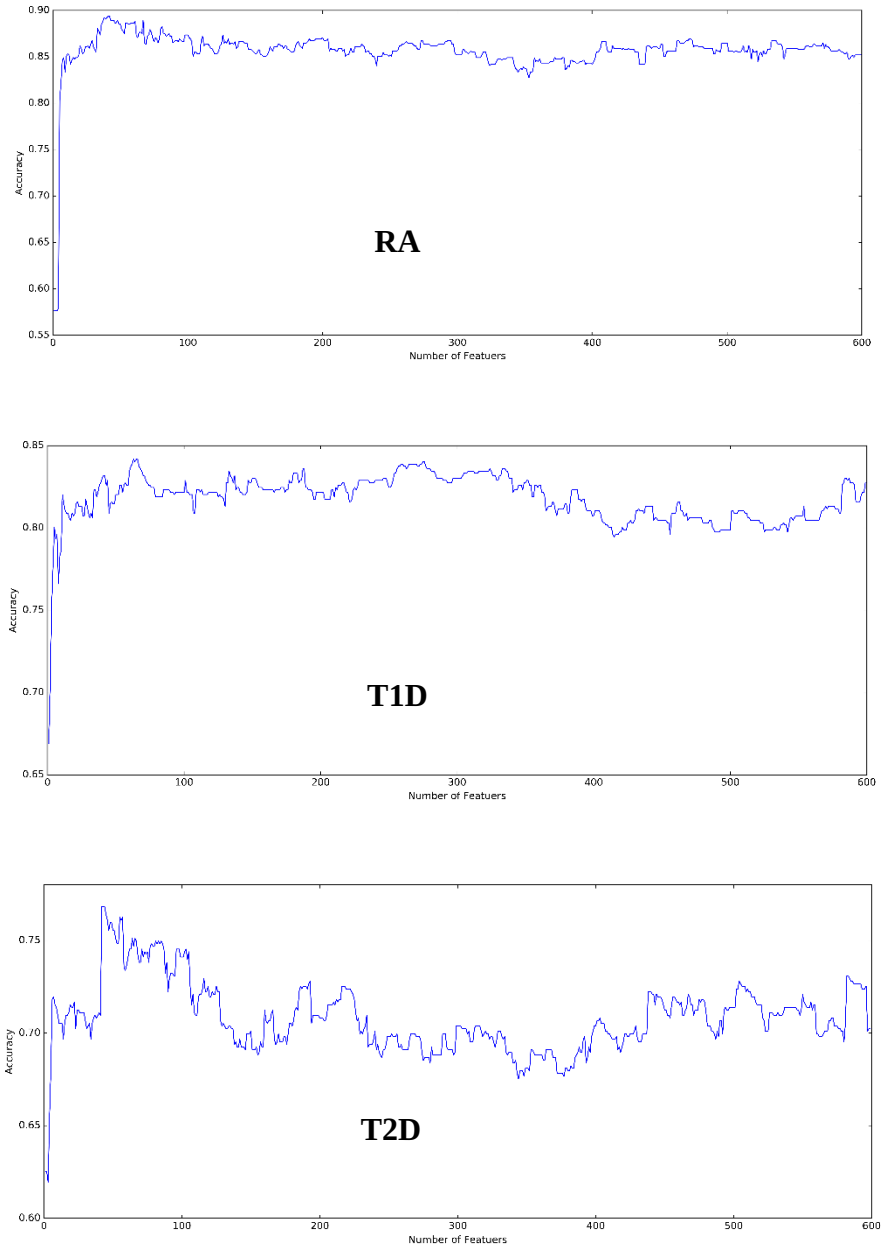


Figure 19. Plots of CE with KNN (cont)

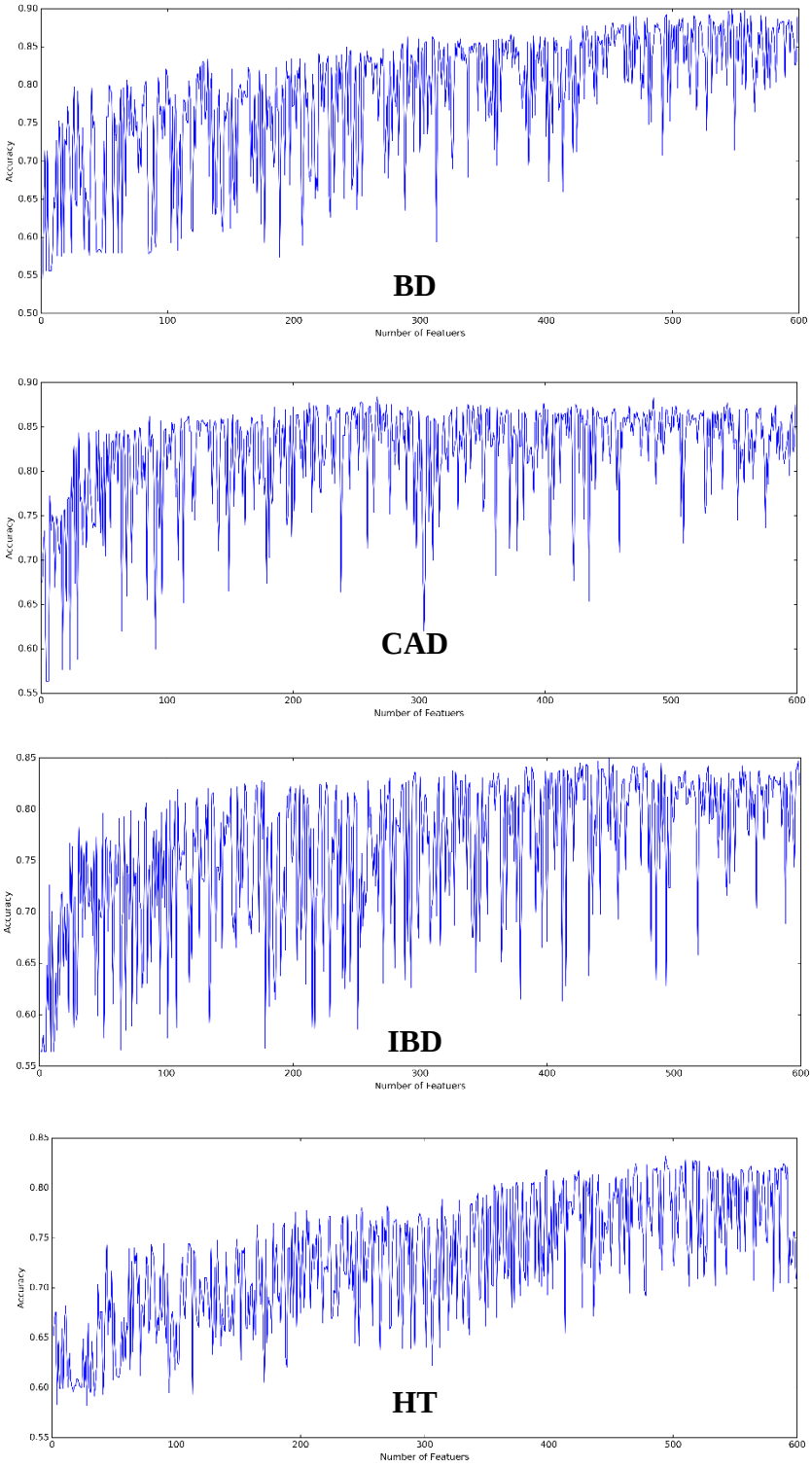


Figure 20. Plots of CE with SVM

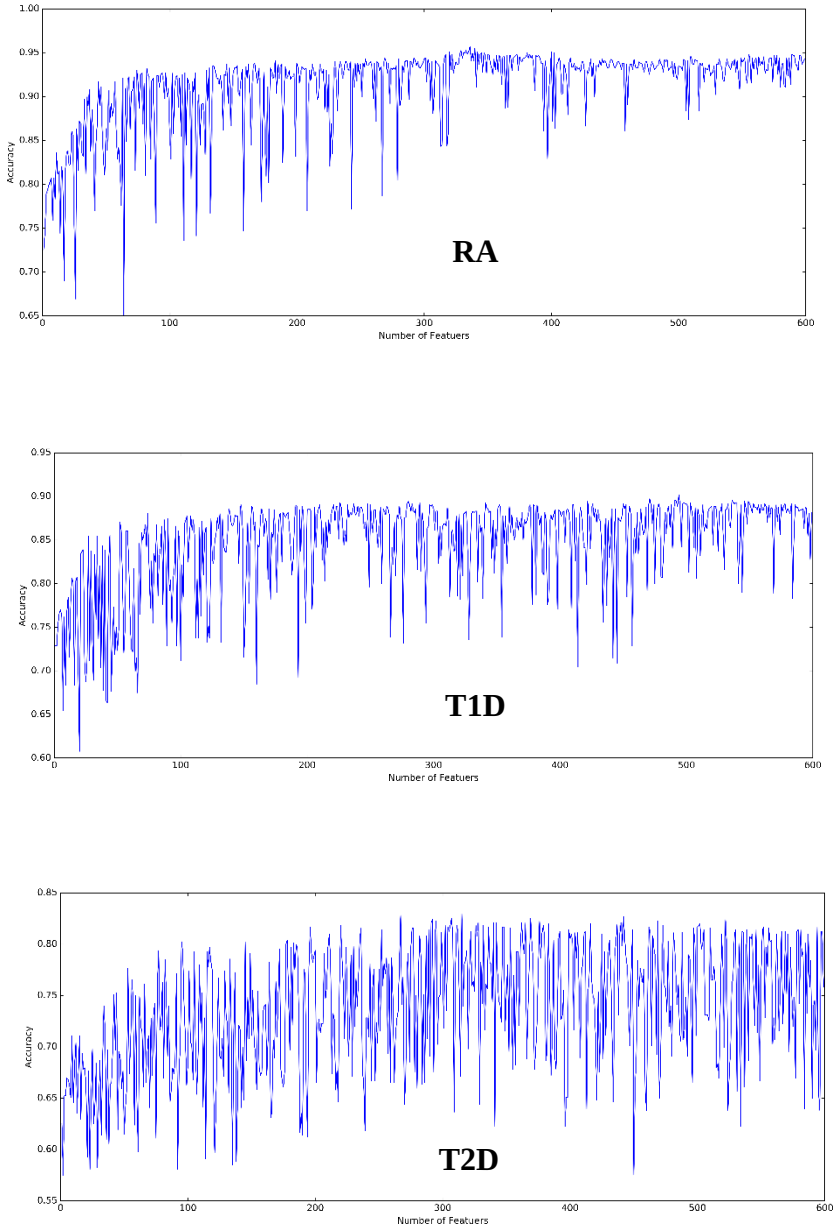


Figure 21. Plots of CE with SVM (cont)

Similar to SU, the results show that CE is performing better with SVM. However, CE requires less average number of features and less average running time when compared to SU.

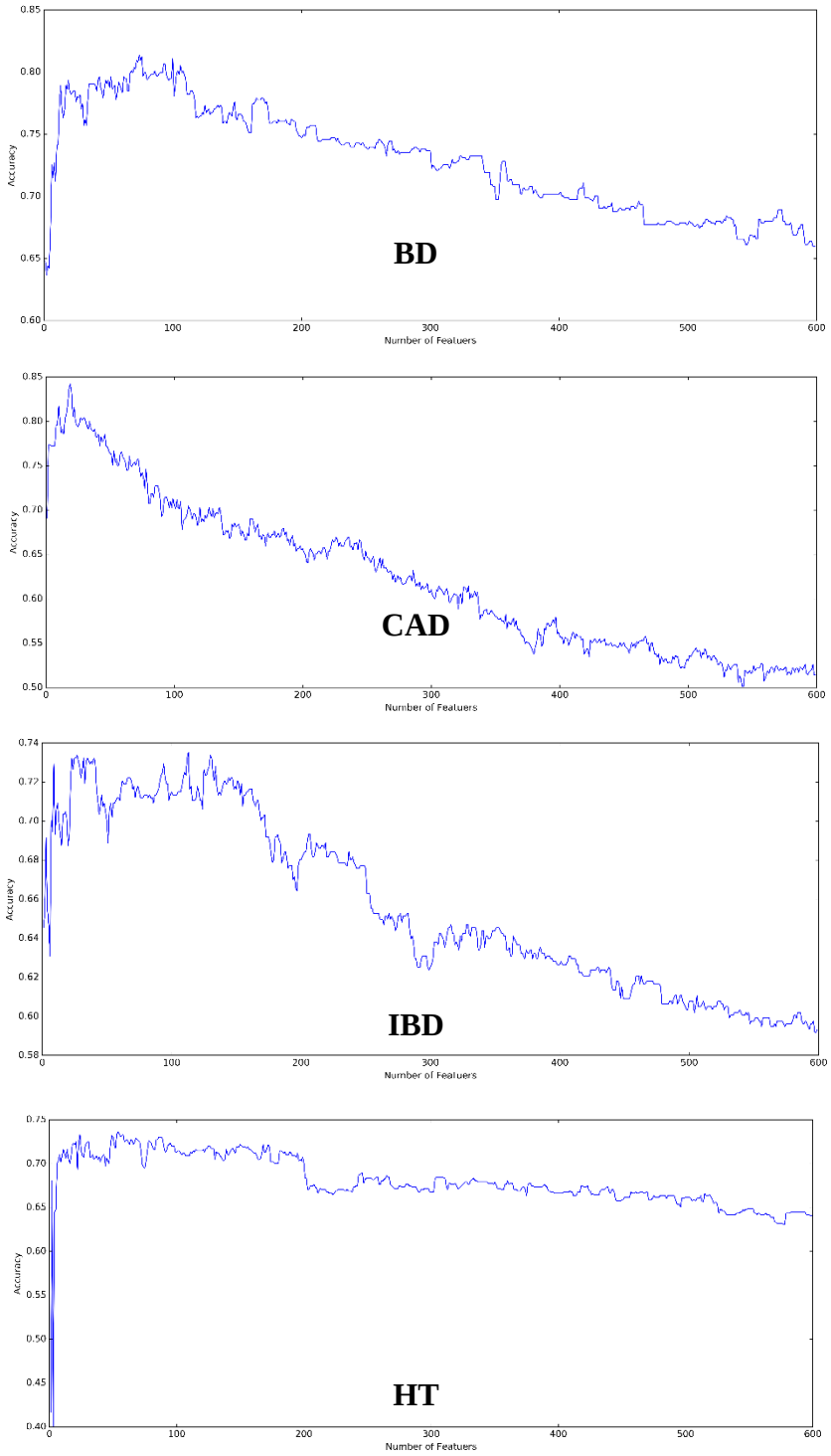


Figure 22. Plots of Method 1 with KNN

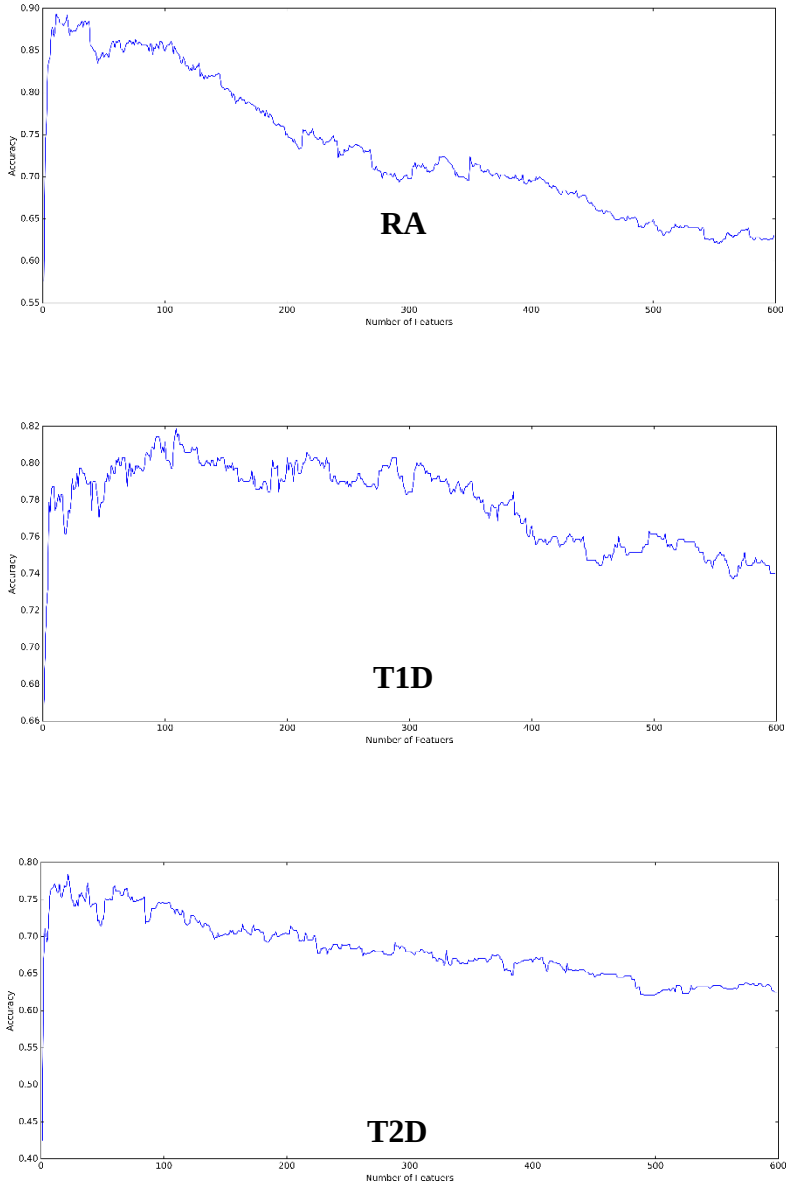


Figure 23. Plots of Method 1 with KNN (cont)

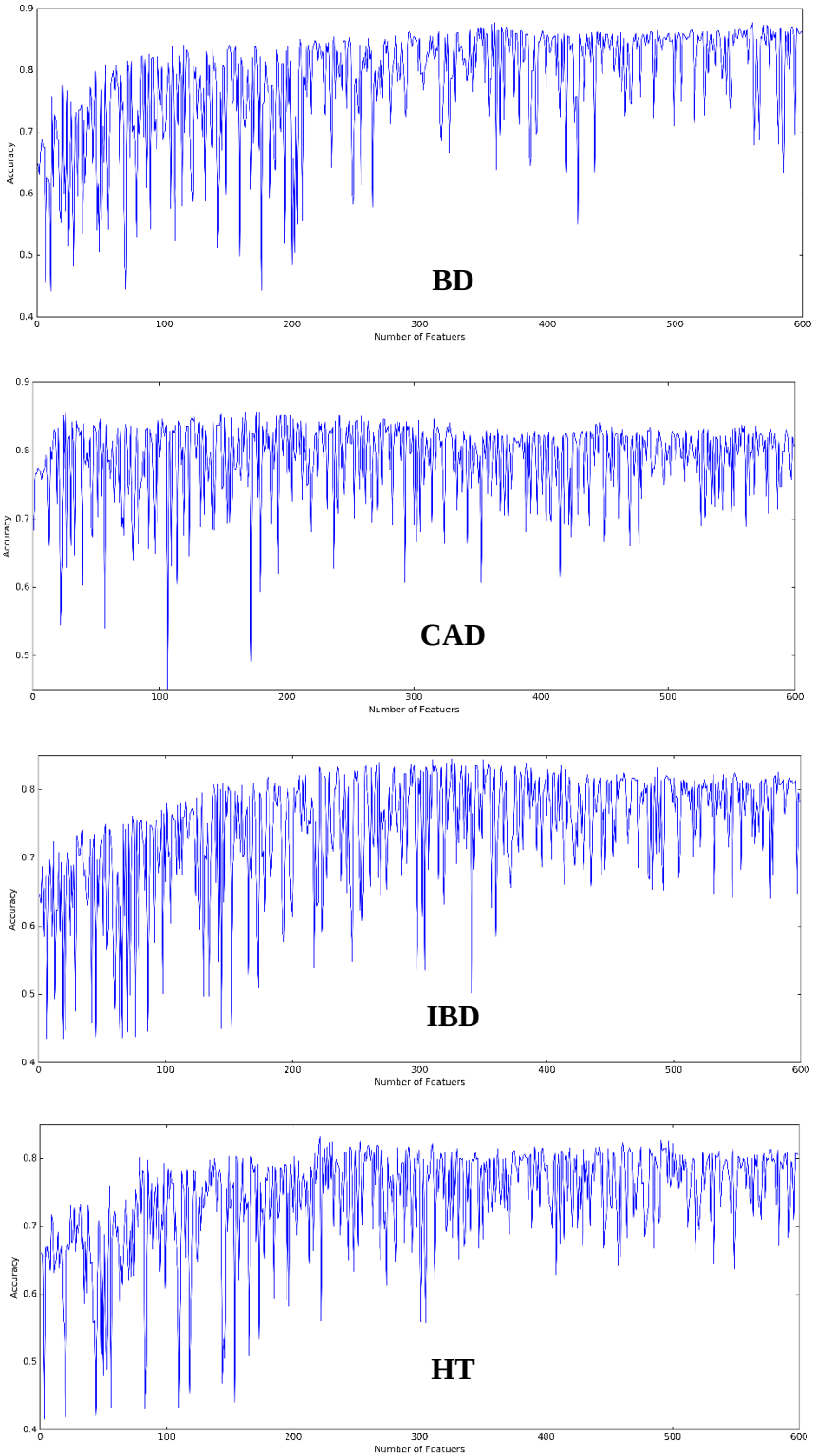


Figure 24. Plots of Method 1 with SVM

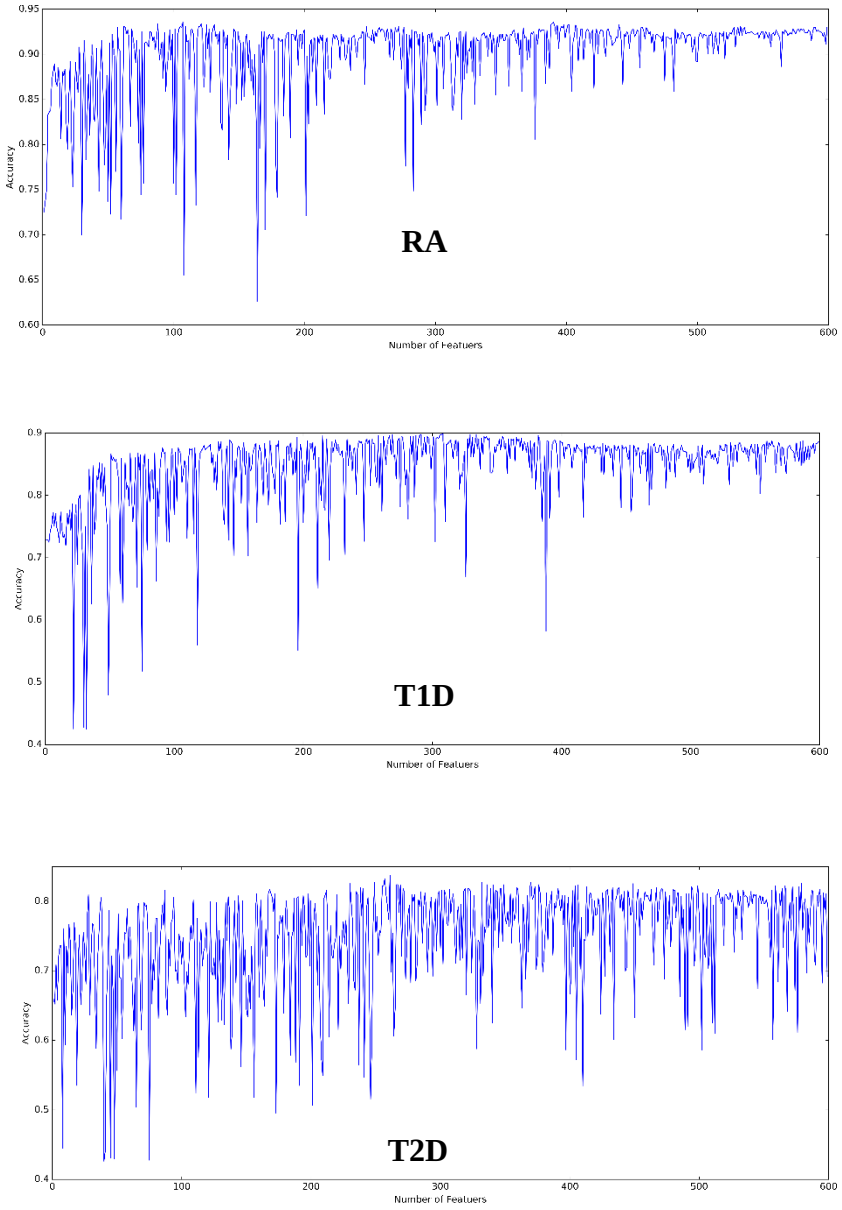


Figure 25. Plots of Method 1 with SVM (cont)

The results show that Method 1 is achieving comparable average accuracy to CE and SU when used with SVM. However, method 1 requires less average number of features and less average running time when compared to SU and CE.

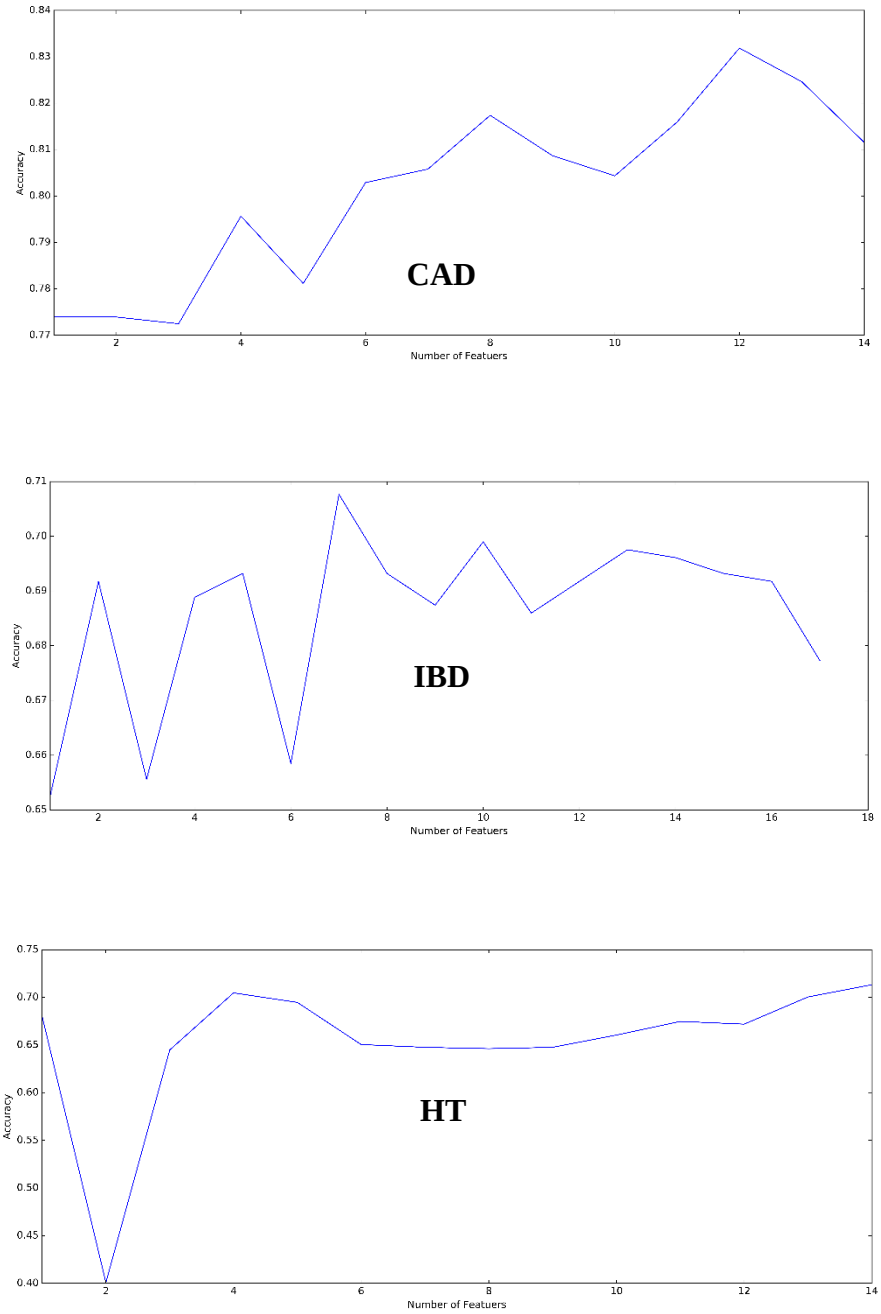


Figure 26. Plots of Method 2 with KNN

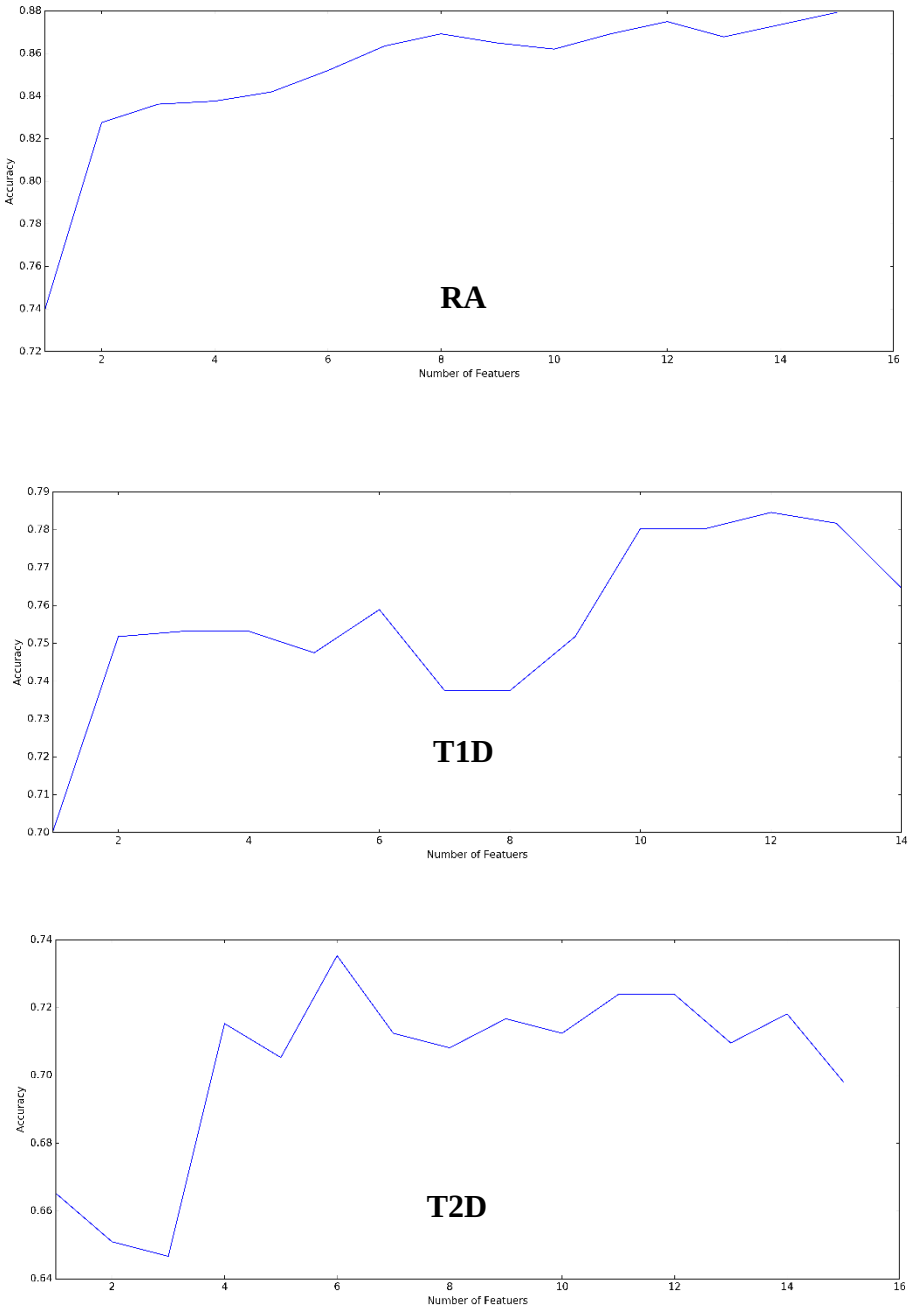


Figure 27. Plots of Method 2 with KNN (cont)

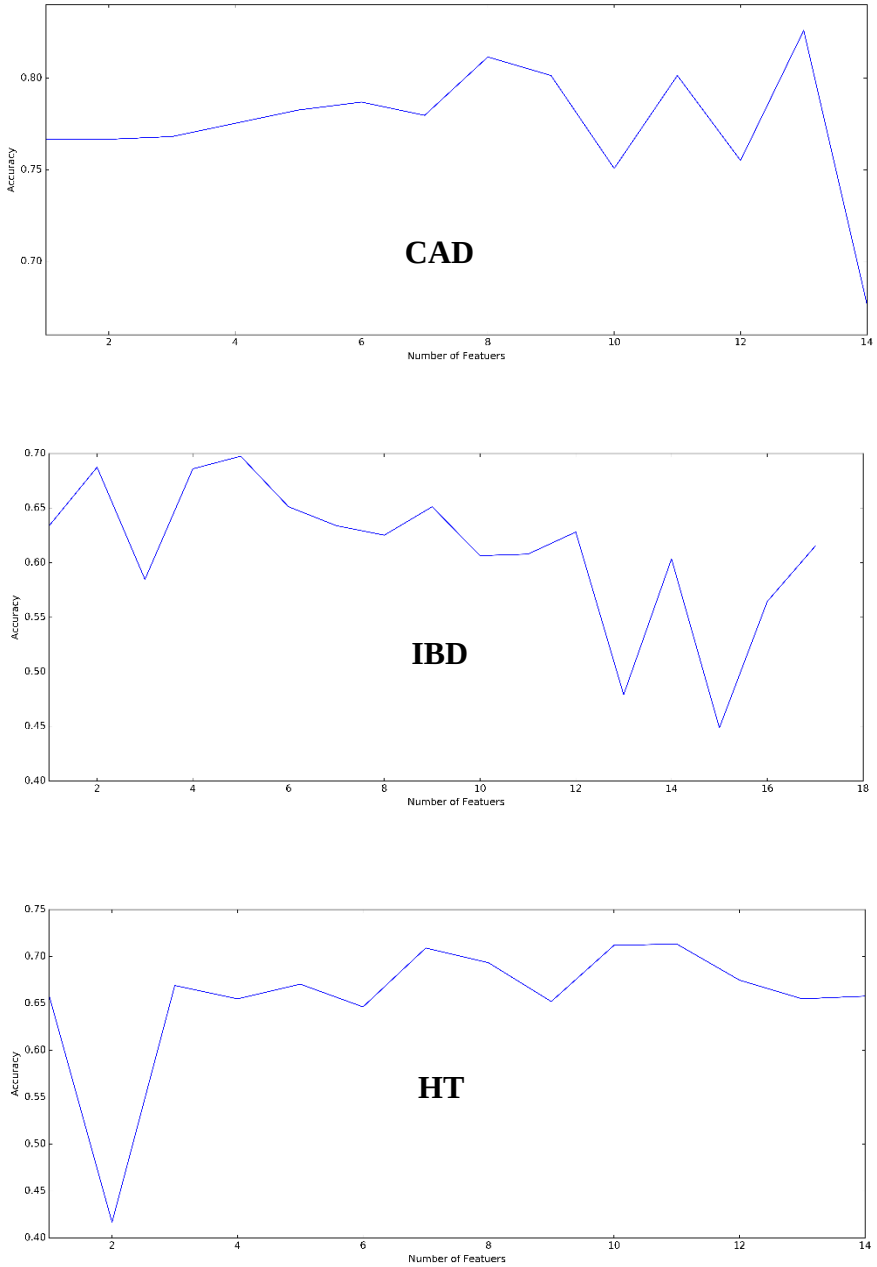


Figure 28. Plots of Method 2 with SVM

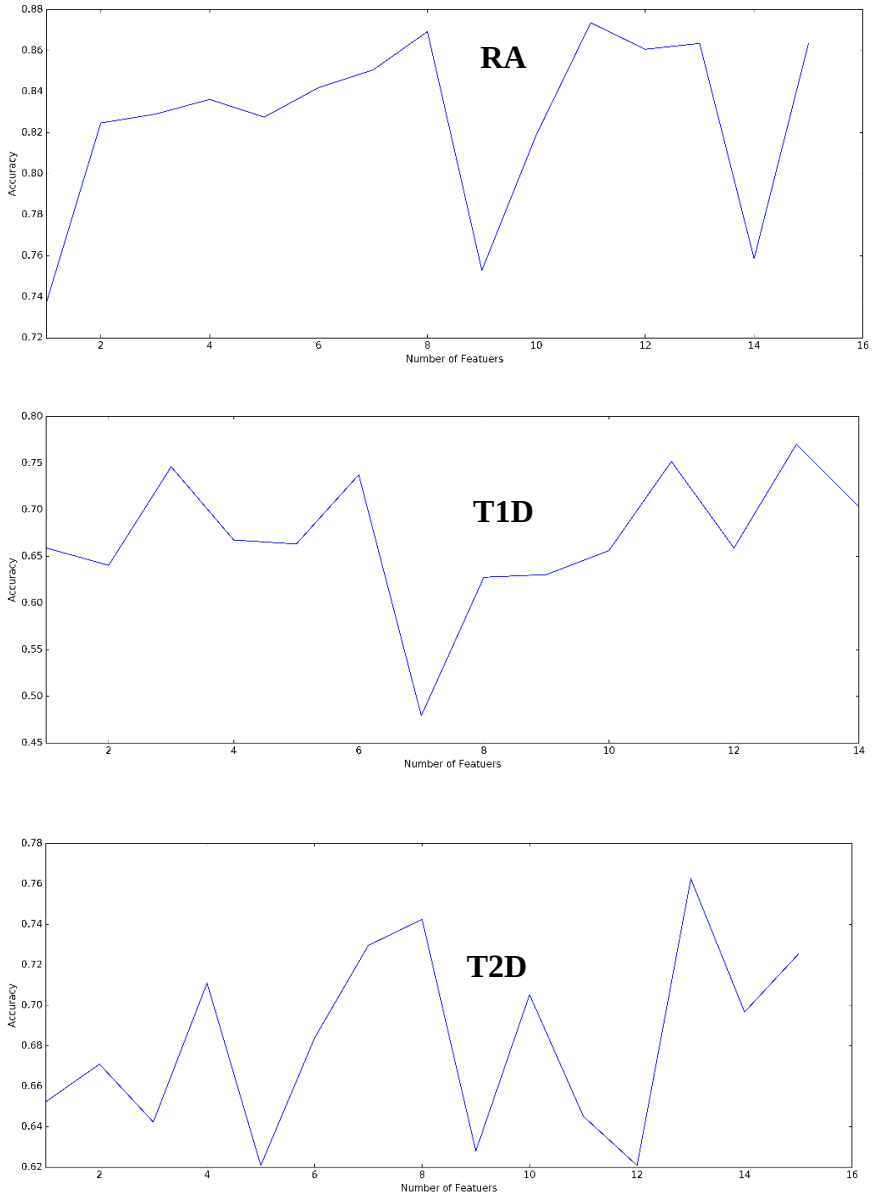


Figure 29. Plots of Method 2 with SVM (cont)

The results show that Method 2 is achieving lower average accuracy when compared to CE, SU and method 1. However, when method 2 is used, SVM and KNN are achieving comparable average accuracy.

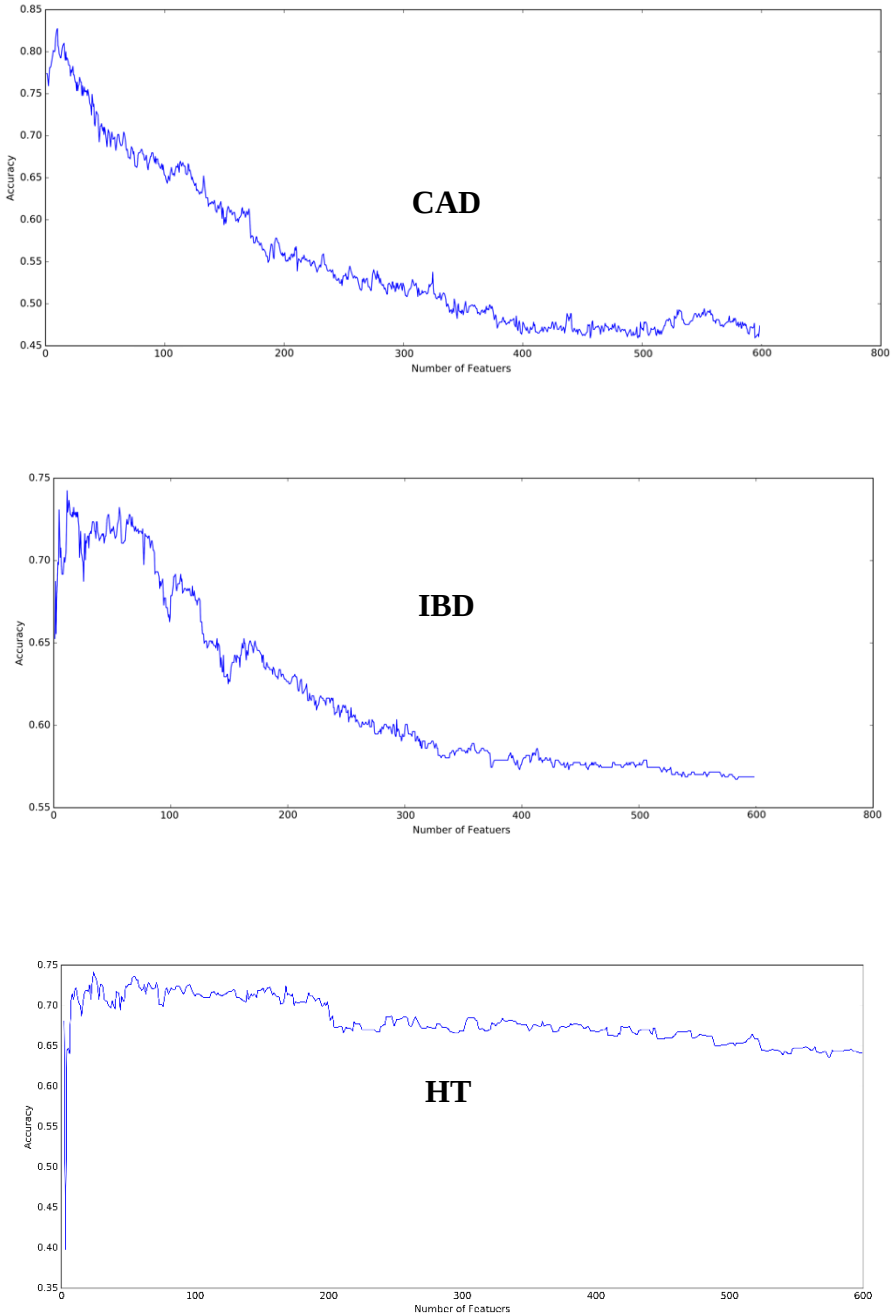
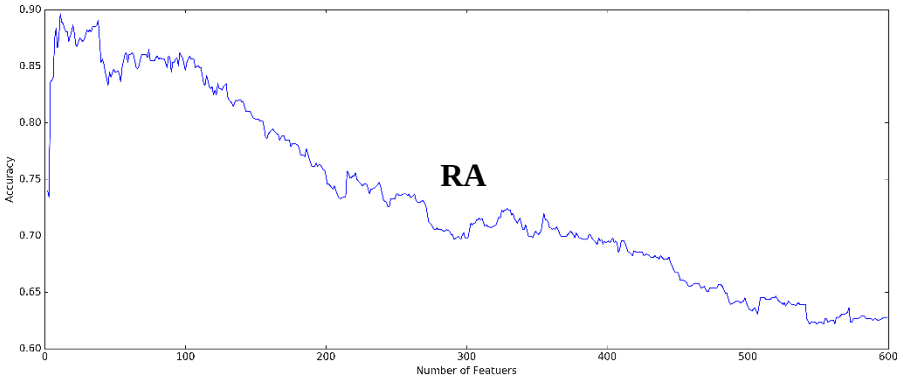
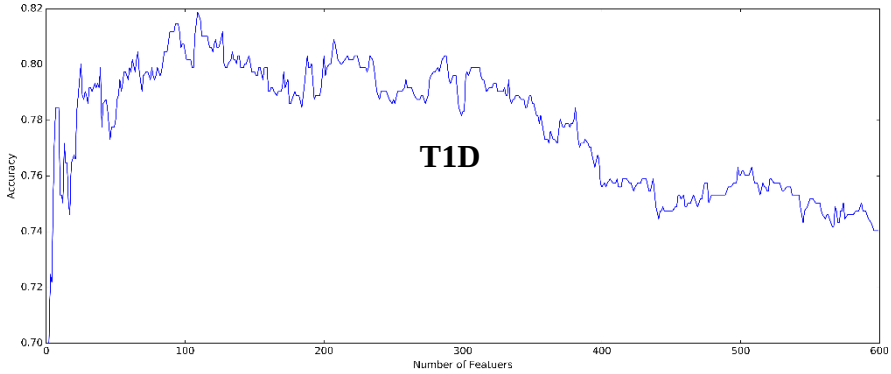


Figure 30. Plots of Method 3 with KNN





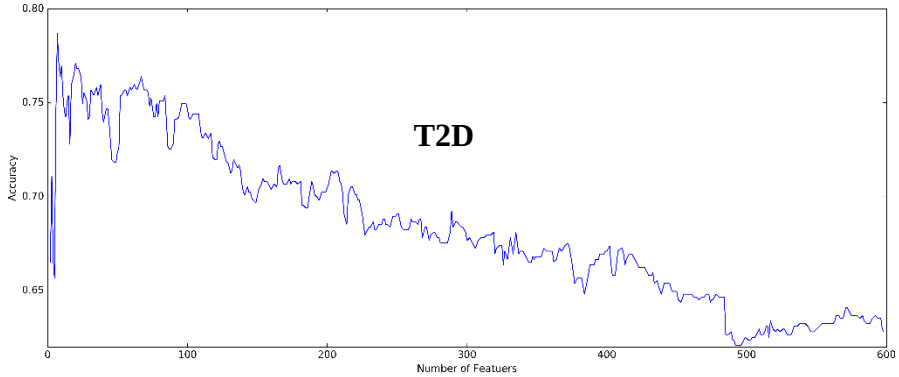
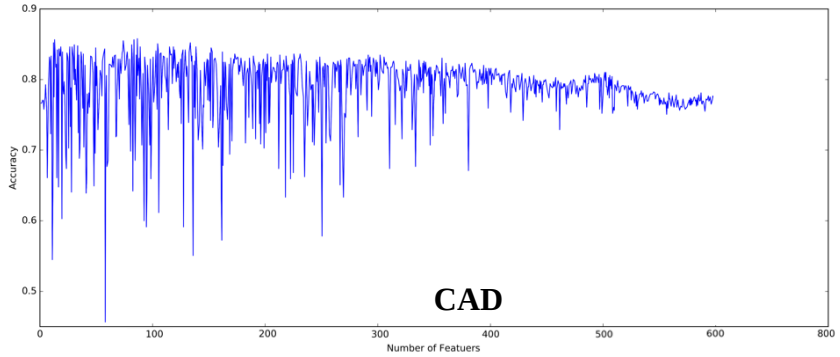
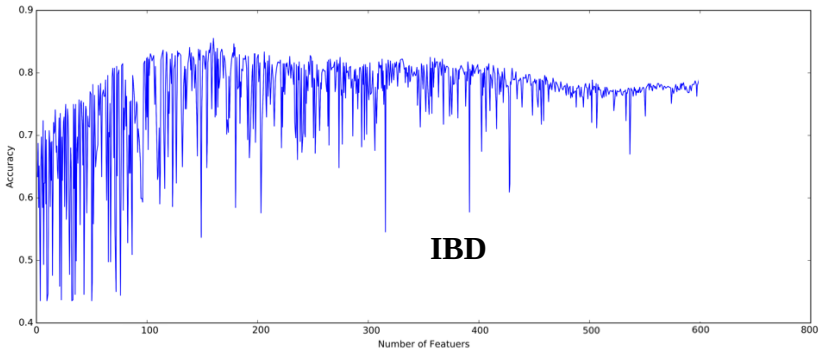


Figure 31. Plots of Method 3 with KNN (cont)





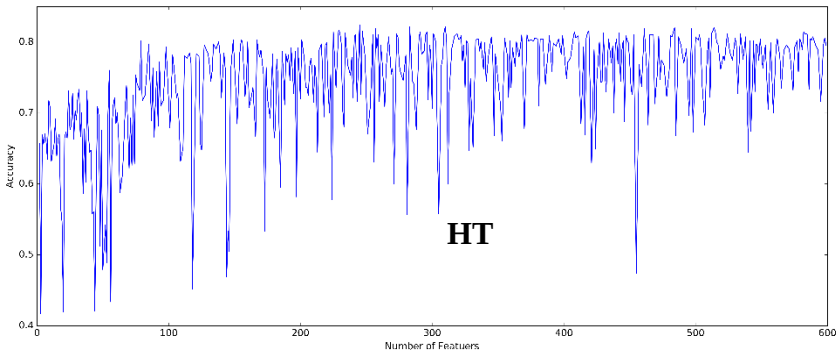


Figure 32. Plots of Method 3 with SVM

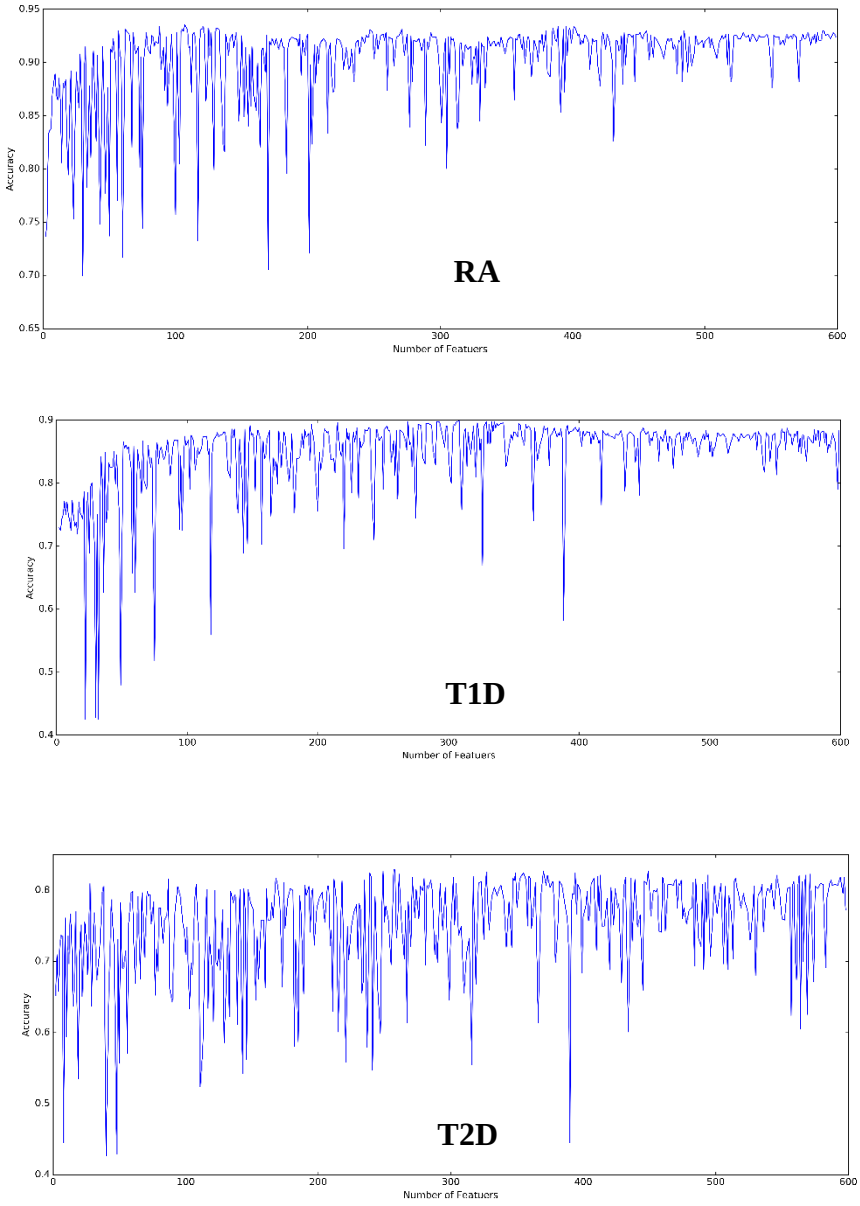


Figure 33. Plots of Method 3 with SVM (cont)

The results show that Method 3 is achieving comparable average accuracy and average running time when compared to method 1. However, method 3 requires less average number of features.

Discussion

The results show that the highest classification accuracy of 87.8 is achieved when CE is used with SVM. The number of features required to reach this accuracy is 414, and the time required to rank the features is 2497 seconds. A comparable classification accuracy is achieved when SU is used with SVM; however, with a larger number of features and almost double of the time is required to rank the features. This clearly shows that CE is better than the SU when the used classifier is SVM for SNPs data.

When comparing the proposed ranking methods, methods 1 and 3 achieved comparable result, with method 3 being slightly better. Method 2; however, achieved degraded results in terms of classification accuracy. This degraded performance can be justified by the limited number of features the method can select.

The results also show that method 3 and CE are achieving comparable classification accuracy. However, method 3 reduced the number of required features and the time required to rank the features by 47 percent and 40 percent respectively. The reduction in the number of required features, can be justified by that when the features are ranked based on the CE, the ranking is only based on the relevancy of the features. This means that some of the selected relevant features might be redundant. In method 3; however, the features are ranked in two lists, the features in each list are specifically useful for one of the classes. When the features are selected, one feature from each list is added at a time, which reduce the chances of selecting redundant features.

In general, the results show that SVM is performing better than KNN regardless of the used dimensionality reduction technique. This can be justified by that in KNN, samples with more similar feature values are assumed to have similar classes, and the model lacks

the sense of discriminating between classes. In addition, this conforms to the results of some of the older reviewed studies [30-33], and justifies the absence of KNN in the more recent reviewed studies [34-43].

CHAPTER 6: CONCLUSION AND FUTURE WORK

Overview

In this thesis, the problem of predicting the existence status of a trait in individuals from their genome-wide genotyping of SNPs, is investigated. The problem is formally defined and the various aspects of the problem are described. The methods exist in the literature to solve the problem are reviewed, and a comparison between these methods is presented. In addition, CE is used to rank features, which is up to the author's knowledge, was never used in similar context. Moreover, three feature ranking methods are proposed. The proposed methods applied to WTCCC1 dataset and one of the methods outperformed the strong baseline in terms of the number of features required and the time required to rank the features.

During the literature review, it was observed that most of the creative work was focused on the features selection part, and the classification part was limited to the use of existing models. In addition, with the increase of the number of SNPs, the complexity of the proposed methods was also increased.

Achievements

The existing feature ranking methods are either requiring one stage or multiple stages to rank features. The former methods only concerned about finding the relevant features. The latter methods; however, are higher in complexity and can find relevant features and remove redundant features. This raises the need for new low-complexity methods that can detect both relevant and redundant features. In this thesis, three feature ranking methods were presented based on the proposed scoring scheme. Method 3 reduced

the execution time of feature selection and the number of features required to achieve similar accuracy in the baseline by 40% and 47% respectively. In addition, method 3 showed how such low-complexity methods can be developed by simply embedding a mechanism that can discriminate to which class label the feature is useful for.

Limitations

While the proposed methods reduced the number of selected features and the computational time compared to the baseline, the classification accuracy was not improved. In addition, method 2 achieved low classification accuracy, which can be due to the limited number of features that the method can select.

Future Work

The literature review presented in this thesis was mainly concerned about the solutions that use machine learning techniques to address the problem. A potential future work, is to investigate how the problem is addressed using statistical methods. The benefit of this is that solutions that combine techniques form both areas can be proposed.

The solution presented in [41] is based on SU and achieved high classification accuracy on the WTCCC1 dataset. In this thesis, the experiments show that the CE and methods 1 and 3 are achieving better performance than SU. A potential future work is to reproduce the work in [41]; however, using CE and methods 1 and 3.

The proposed feature ranking method 2 has limitation by design. The method can only select the highest scoring feature per region. Thus, a potential future work is to produce an enhanced version of method 2. The enhancement shall increase the number of selected features per region.

Finally, SVM and KNN are the only classifiers that used in this thesis. A potential future work is to evaluate the proposed methods with other classifiers. In addition, convolutional neural networks (CNNs) have revolutionized many learning tasks, specifically in the image classification area. One of the main advantages of CNN, is that it can be accelerated using GPUs and reconfigurable hardware. A solution that harness CNN to address the problem is under investigation by the author of this thesis.

REFERENCES

[1] "What is DNA? - Genetics Home Reference." [Online]. Available: https://ghr.nlm.nih.gov/primer/basics/dna. [Accessed: 08-Sep-2016].

[2] "Deoxyribonucleic Acid (DNA) Fact Sheet." [Online]. Available: https://www.genome.gov/25520880/. [Accessed: 08-Sep-2016].

[3] I. Human Genome Sequencing Consortium, "Finishing the euchromatic sequence of the human genome," Nature, vol. 431, no. 7011, pp. 931–945, Oct. 2004.

[4] "Qatar National Research Strategy (QNRS)." [Online]. Available: http://www.qnrf.org/en-us/About-Us/QNRS. [Accessed: 08-Sep-2016].

[5] "Qatar Biobank for Medical Research." [Online]. Available: http://www.qatarbiobank.org.qa/home. [Accessed: 08-Sep-2016].

[6] "Path Towards Personalized Medicine (PPM)." [Online]. Available: http://www.qnrf.org/en-us/Funding/Research-Programs/Thematic-Program/PPM. [Accessed: 08-Sep-2016].

[7] T. 1000 G. P. Consortium, "An integrated map of genetic variation from 1,092 human genomes," Nature, vol. 491, no. 7422, pp. 56–65, Nov. 2012.

[8] "Understanding Human Genetic Variation - NIH Curriculum Supplement Series - NCBI Bookshelf." [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK20363/. [Accessed: 09-Sep-2016].

[9] F. S. Collins, L. D. Brooks, and A. Chakravarti, "A DNA polymorphism discovery resource for research on human genetic variation," Genome Res., vol. 8, no. 12, pp. 1229–1231, Dec. 1998.

[10] G. T. Marth et al., “A general approach to single-nucleotide polymorphism discovery,” Nat Genet, vol. 23, no. 4, pp. 452–456, Dec. 1999.

[11] “Single-nucleotide polymorphism - ISOGG Wiki.” [Online]. Available: http://isogg.org/wiki/Single-nucleotide_polymorphism. [Accessed: 10-Sep-2016].

[12] “The Cost of Sequencing a Human Genome.” [Online]. Available: https://www.genome.gov/27565109/the-cost-of-sequencing-a-human-genome/. [Accessed: 10-Sep-2016].

[13] J. N. Hirschhorn and M. J. Daly, “Genome-wide association studies for common diseases and complex traits,” Nat Rev Genet, vol. 6, no. 2, pp. 95–108, Feb. 2005.

[14] E. Lai, “Application of SNP technologies in medicine: lessons learned and future challenges,” Genome Res., vol. 11, no. 6, pp. 927–929, Jun. 2001.

[15] J. A. Goldstein, “Clinical relevance of genetic polymorphisms in the human CYP2C subfamily,” Br J Clin Pharmacol, vol. 52, no. 4, pp. 349–355, Oct. 2001.

[16] R. E. Laing, P. Hess, Y. Shen, J. Wang, and S. X. Hu, “The role and impact of SNPs in pharmacogenomics and personalized medicine,” Curr. Drug Metab., vol. 12, no. 5, pp. 460–486, Jun. 2011.

[17] K. K. Kidd et al., “Developing a SNP panel for forensic identification of individuals,” Forensic Sci. Int., vol. 164, no. 1, pp. 20–32, Dec. 2006.

[18] “dbSNP Summary.” [Online]. Available: http://www.ncbi.nlm.nih.gov/projects/SNP/snp_summary.cgi?view+summary=view+summary&build_id=147. [Accessed: 13-Sep-2016].

[19] T. LaFramboise, “Single nucleotide polymorphism arrays: a decade of biological, computational and technological advances,” Nucleic Acids Res, vol. 37, no. 13, pp. 4181–4193, Jul. 2009.

[20] N. Nishida et al., “Evaluating the performance of Affymetrix SNP Array 6.0 platform with 400 Japanese individuals,” BMC Genomics, vol. 9, p. 431, 2008.

[21] A. Vignal, D. Milan, M. SanCristobal, and A. Eggen, “A review on SNP and other types of molecular markers and their use in animal genetics,” Genet Sel Evol, vol. 34, no. 3, pp. 275–305, May 2002.

[22] T. Casci, “Population genetics: SNPs that come in threes,” Nat Rev Genet, vol. 11, no. 1, pp. 8–8, Jan. 2010.

[23] M. Singh, P. Singh, P. K. Juneja, S. Singh, and T. Kaur, “SNP-SNP interactions within APOE gene influence plasma lipids in postmenopausal osteoporosis,” Rheumatol. Int., vol. 31, no. 3, pp. 421–423, Mar. 2011.

[24] P. R. Burton et al., “Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls,” Nature, vol. 447, no. 7145, pp. 661–678, Jun. 2007.

[25] S. V. Stehman, “Selecting and interpreting measures of thematic classification accuracy,” Remote Sensing of Environment, vol. 62, no. 1, pp. 77–89, Oct. 1997.

[26] M. H. Zweig and G. Campbell, “Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine,” Clin. Chem., vol. 39, no. 4, pp. 561–577, Apr. 1993.

[27] P. Larrañaga et al., “Machine learning in bioinformatics,” Brief Bioinform, vol. 7, no. 1, pp. 86–112, Mar. 2006.

[28] C. H. Papadimitriou, “Computational Complexity,” in Encyclopedia of Computer Science, Chichester, UK: John Wiley and Sons Ltd., pp. 260–265.

[29] J. Listgarten et al., “Predictive Models for Breast Cancer Susceptibility from Multiple Single Nucleotide Polymorphisms,” Clin Cancer Res, vol. 10, no. 8, pp. 2725–2737, Apr. 2004.

[30] S. Uhmn, D. H. Kim, S. W. Cho, J. Y. Cheong, and J. Kim, “Chronic Hepatitis Classification Using SNP Data and Data Mining Techniques,” in Frontiers in the Convergence of Bioscience and Information Technologies, 2007. FBIT 2007, pp. 81–86, 2007.

[31] D.-H. Kim, S. Uhmn, Y.-W. Ko, S. W. Cho, J. Y. Cheong, and J. Kim, “Chronic Hepatitis and Cirrhosis Classification Using SNP Data, Decision Tree and Decision Rule,” in Computational Science and Its Applications – ICCSA 2007, O. Gervasi and M. L. Gavrilova, Eds. Springer Berlin Heidelberg, pp. 585–596, 2007.

[32] S. Uhmn, D.-H. Kim, Y.-W. Ko, S. Cho, J. Cheong, and J. Kim, “A study on application of single nucleotide polymorphism and machine learning techniques to diagnosis of chronic hepatitis,” Expert Systems, vol. 26, no. 1, pp. 60–69, Feb. 2009.

[33] Q. Liu, J. Yang, Z. Chen, M. Q. Yang, A. H. Sung, and X. Huang, “Supervised learning-based tagSNP selection for genome-wide disease classifications,” BMC Genomics, vol. 9, no. 1, pp. 1–9, 2008.

[34] L.-C. Huang, S.-Y. Hsu, and E. Lin, “A comparison of classification methods for predicting Chronic Fatigue Syndrome based on genetic data,” Journal of Translational Medicine, vol. 7, p. 81, 2009.

[35] T. Abeel, T. Helleputte, Y. V. de Peer, P. Dupont, and Y. Saeys, “Robust biomarker identification for cancer diagnosis with ensemble feature selection methods,” Bioinformatics, vol. 26, no. 3, pp. 392–398, Feb. 2010.

[36] R. Jiang, W. Tang, X. Wu, and W. Fu, “A random forest approach to the detection of epistatic interactions in case-control studies,” BMC Bioinformatics, vol. 10, no. 1, pp. 1–12, 2009.

[37] G. F. Cooper, P. Hennings-Yeomans, S. Visweswaran, and M. Barmada, “An Efficient Bayesian Method for Predicting Clinical Outcomes from Genome-Wide Data,” AMIA Annu Symp Proc, vol. 2010, pp. 127–131, 2010.

[38] Y. Liu, M. K. Ng, and J. Zhou, “SNP Specific Extraction and Analysis Using Shrunken Dissimilarity Measure,” in Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology, New York, NY, USA, pp. 378–381, 2010.

[39] F. Sambo, E. Trifoglio, B. Di Camillo, G. M. Toffolo, and C. Cobelli, “Bag of Naïve Bayes: biomarker selection and classification from genome-wide SNP data,” BMC Bioinformatics, vol. 13, no. 14, pp. 1–10, 2011.

[40] S. Okser, T. Pahikkala, A. Airola, T. Aittokallio, and T. Salakoski, “Fast and parallelized greedy forward selection of genetic variants in Genome-wide association studies,” in 2011 IEEE International Workshop on Genomic Signal Processing and Statistics (GENSIPS), pp. 214–217, 2011.

[41] J. R. Quevedo, A. Bahamonde, M. Perez-Enciso, and O. Luaces, “Disease Liability Prediction from Large Scale Genotyping Data Using Classifiers with a Reject

Option," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 9, no. 1, pp. 88–97, Jan. 2012.

[42] T.-T. Nguyen, J. Huang, Q. Wu, T. Nguyen, and M. Li, "Genome-wide association data classification and SNPs selection using two-stage quality-based Random Forests," BMC Genomics, vol. 16 Suppl 2, p. S5, 2015.

[43] A. Boutorh and A. Guessoum, "Classification of SNPs for breast cancer diagnosis using neural-network-based association rules," in 2015 12th International Symposium on Programming and Systems (ISPS), pp. 1–9, 2015.

[44] "Human Mapping 500K Array Set | Affymetrix." [Online]. Available: http://www.affymetrix.com/estore/catalog/131459/AFFY/Mapping-500K-Array-Set#1_1. [Accessed: 06-Apr-2017].

[45] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, p. 2825−2830, Oct. 2011.

[46] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing. New York, NY, USA: Cambridge University Press, 1992.

[47] F. Fleuret, "Fast binary feature selection with conditional mutual information," Journal of Machine Learning Research, vol. 5, no. Nov, pp. 1531–1555, 2004.

[48] N. S. Altman, "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression," The American Statistician, vol. 46, no. 3, pp. 175–185, Aug. 1992.

[49] L. Peterson, "K-nearest neighbor," Scholarpedia, vol. 4, no. 2, p. 1883, 2009.

[50] C. Cortes and V. Vapnik, "Support-vector networks," Mach Learn, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[51] “Introduction to Support Vector Machines — OpenCV 2.4.13.2 documentation.” [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html. [Accessed: 08-Apr-2017].

[52] “Research Computing.” [Online]. Available: https://rc.qatar.tamu.edu/Pages/hpc/raad/overview.aspx. [Accessed: 12-Apr-2017].

[53] R. Kohavi, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection,”, pp. 1137–1143, 1995.