On Feature Selection Stability: A Data Perspective

by

Salem Alelyani

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2013 by the
Graduate Supervisory Committee:

Huan Liu, Chair
Guoliang Xue
Jieping Ye
Zheng Zhao

ARIZONA STATE UNIVERSITY

May 2013

ABSTRACT

The rapid growth in the high-throughput technologies last few decades makes the manual processing of the generated data to be impracticable. Even worse, the machine learning and data mining techniques seemed to be paralyzed against these massive datasets. High-dimensionality is one of the most common challenges for machine learning and data mining tasks. Feature selection aims to reduce dimensionality by selecting a small subset of the features that perform at least as good as the full feature set. Generally, the learning performance, e.g. classification accuracy, and algorithm complexity are used to measure the quality of the algorithm. Recently, the stability of feature selection algorithms has gained an increasing attention as a new indicator due to the necessity to select similar subsets of features each time when the algorithm is run on the same dataset even in the presence of a small amount of perturbation.

In order to cure the selection stability issue, we should understand the cause of instability first. In this dissertation, we will investigate the causes of instability in high-dimensional datasets using well-known feature selection algorithms. As a result, we found that the stability mostly data-dependent. According to these findings, we propose a framework to improve selection stability by solving these main causes. In particular, we found that data noise greatly impacts the stability and the learning performance as well. So, we proposed to reduce it in order to improve both selection stability and learning performance. However, current noise reduction approaches are not able to distinguish between data noise and variation in samples from different classes. For this reason, we overcome this limitation by using Supervised noise reduction via Low Rank Matrix Approximation, SLRMA for short. The proposed framework has proved to be successful on different types of datasets

with high-dimensionality, such as microarrays and images datasets. However, this framework cannot handle unlabeled, hence, we propose Local SVD to overcome this limitation.

DEDICATION

I would like to dedicate this thesis to my parents, my wife (*Fatimah*), my son (*Tamim*), my daughter (*Aleen*), my brothers and my sisters for their love, encouragement and support in all my endeavours. Without them, this thesis would not have been made possible.

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my advisor Prof. Huan Liu for the continuous support during my Ph.D journey , for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study. He did an unlimited effort to prepare me and my labmates to the real-world life, not only in research and school-related issues but, also, in different life-walks.

I would like, also, to thank the rest of my thesis committee: Prof. Guoliang Xue, Dr. Jieping Ye and Dr.Zheng Zhao, for their encouragement and insightful comments.

Beside those, I thank Data Mining and Machine Learning Lab members (DMMLers): Xufei Wang, Reza Zafarani, Mohammad-Ali Abbasi, Jiliang Tang, Shamanth Kumar, Fred Morstatter, Pritam Gundecha, Huiji Gao, Xia Hu, and Isaac Jones. Those whose daily presence in the lab made the difference. Although it was a nightmare the time when I present any work during the lab meeting, their critical comments and tough discussions sharpened my presentation skills and gave me the ability to survive outside the lab. Dr. Liu says: "*the one who survives here can survive outside*"; he was right.

My acknowledgements go to my sponsor: *King Khalid University* and the Ministry of Higher Education in Saudi Arabia represented in the Saudi Cultural Mission and the Royal Embassy in the USA for the full scholarship was given to me.

Last but not the least, I would like to thank my family: my parents, my wife, my kids, and my siblings for their love and support. I owe them lots of time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

The growth of the high-throughput technologies nowadays has led to exponential growth in the harvested data with respect to dimensionality and sample size. As a sequence, storing and processing these data becomes more challenging. Figure (1.1) shows the trend of this growth for UCI machine learning repository. This augmentation made manual processing for these datasets to be impractical. Therefore, data mining and machine learning tools were proposed to automating pattern recognition and knowledge discovery process. However, using data mining techniques on an ore data is mostly useless due to the high level of noise associated with collected samples. Usually, data noise is either due to imperfection in the technologies that collected the data or the nature of the source of this data itself. For instance, in medical images domain, any deficiency in the imaging device will be reflected as noise in the dataset later on. This kind of noise is caused by the device itself. On the other hand, text datasets crawled from the internet, are noisy by nature because they are usually informally written and suffer from grammatical mistakes, misspelling, and improper punctuation. Undoubtedly, extracting useful knowledge from such huge and noisy datasets is a painful task.

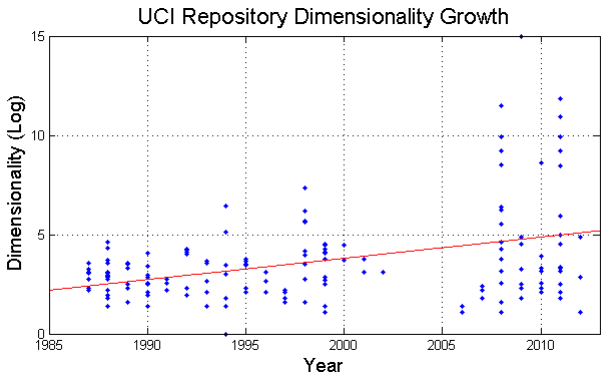
Dimensionality reduction is one popular technique to remove noisy (i.e. irrelevant) and redundant attributes (AKA features). Dimensionality reduction techniques can be categorized mainly into feature extraction and feature selection. In feature extraction approach, features are projected into a new space with lower dimensionality. Examples of feature extraction technique include Principle Component Analysis (PCA), Linear Discriminant Analysis (LDA), Singular Value Decomposition (SVD), to name a few. On the other hand, the feature selection approach aims to select a

small subset of features that minimize redundancy and maximize relevance to the target (i.e. class label). Popular feature selection techniques include: Information Gain, Relief, Chi Squares, Fisher Score, and Lasso, to name a few.

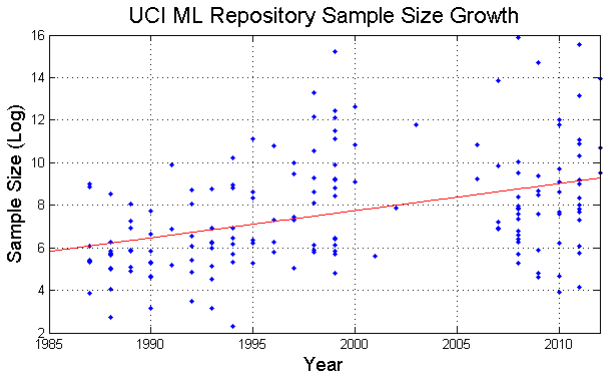
Both dimensionality reduction approaches are capable of improving learning performance, lowering computational complexity, building better generalizable models, and decreasing required storage. However, feature selection is superior in terms of better readability and interpretability since it maintains the original feature values in the reduced space while feature extraction transforms the data from the original space into a new space with lower dimension, which cannot be linked to the features in the original space. Therefore, further analysis of the new space is problematic since there is no physical meaning for the transformed features obtained from feature extraction technique.

Feature selection is broadly categorized into four models, namely: filter model, wrapper model, embedded model and hybrid model. As we mentioned above, feature selection selects subset of highly discriminant features. In other words, it selects features that are capable of discriminating samples that belong to different classes. Thus, we need to have labeled data as training samples in order to select these features. This kind of learning is called *supervised learning*, which means that the dataset is labeled. If the data were unlabled, this is called *supervised learning*. In supervised learning, it is easy to define what relevant feature means. It simply refers to the feature that is capable of distinguishing different classes. For example, a feature f_i is said to be relevant to a class label $\mathbf{y}$ if f_i and $\mathbf{y}$ are highly correlated.

In the last three decades, a large number of feature selection algorithms has been developed, and feature selection techniques have been successfully applied in various domains including pattern recognition [40, 77, 62, 52], text categorization



(a)



(b)

Figure 1.1: Plot (a) shows the dimensionality growth trend in UCI Machine Learning Repository from mid 80s to 2012 while (b) shows the growth in the sample size for the same period.

Table 1.1: Nomenclature

$\mathbf{X}$	Dataset
n	Sample size
m	Number of features
x_j	the j^{th} sample
$\mathbf{x}_j$	the sample vector
f_i	the i^{th} feature
$\mathbf{f}_i$	the feature vector
$\mathbf{y}$	the class label vector
c	number of classes
$\mathcal{F}$	the original feature set
$\mathcal{F}'$	Selected feature subset or result
k	Number of selected features

[89, 43, 65, 26], Image processing [40, 72], bioinformatics [63, 74] and so on. The high volume of existing feature selection approaches necessitates effective evaluation techniques to compare algorithms, so that proper ones can be chosen to serve the users' requirements.

One metric that is widely use to evaluate the quality of the selected feature is the classification accuracy. This metric demonstrates the ability of the selected features to distinguish the class label of the data. High classification accuracy means good selected features. Yet, it is noticed that there exists several subsets of features that preform equally good in terms of classification accuracy. This observation triggered an important questions especially for domain experts who are interested to probe further in data analysis using the selected set of features. The selection stability (i.e. inconsistency of the selected subsets) has drawn increasing attention lately.

1.1 Problem Statement

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ be the feature set where m is the number of features and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times m}$ be a given dataset with n data points where $n \leq m$. The supervised feature selection, illustrated in Figure 1.2, is formally stated as:

$$f(\mathcal{F}; \mathbf{X}, \mathbf{y}) \rightarrow \{\mathcal{F}'\}$$

where $f(\cdot)$ is the feature selection method, $\mathbf{y}$ is the class label and $\mathcal{F}' \subset \mathcal{F}$, where $|\mathcal{F}'| = k$ is the number of selected features, $k \ll m$.

The selected features in $\mathcal{F}'$ are assumed to be highly relevant to $\mathbf{y}$ and less redundant to each other. Assuming, we are using a correlation metric $\Gamma(\cdot)$ to evaluate the relevance and the redundancy of $\mathcal{F}'$, feature selection goal is to satisfy the

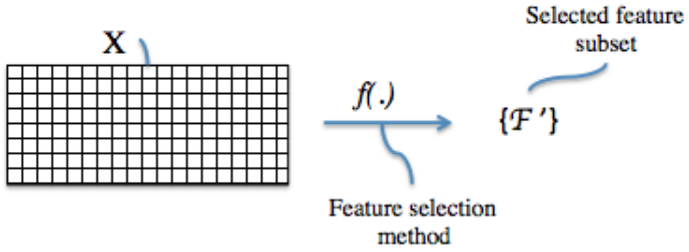


Figure 1.2: Feature Selection

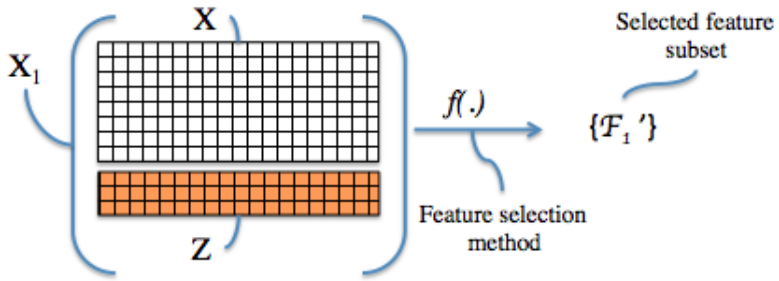


Figure 1.3: New samples may significantly impact the selection

following criteria:

$$|\Gamma(\mathbf{f}'_i, \mathbf{y})| \geq \hat{\gamma}, \ \textbf{and}$$

$$|\Gamma(\mathbf{f}'_i, \mathbf{f}'_j)| \leq \hat{\hat{\gamma}}, \quad \forall i, j = 1, \ldots, k \ \textit{and} \ i \neq \ j$$

where $\hat{\gamma}$ and $\hat{\hat{\gamma}}$ are two user-defined parameters or guided by the number of selected features k.

Lets now introduce a perturbation on $\mathbf{X}$ in the form of new set of samples $\mathbf{Z} \in \mathrm{R}^{n\mathbf{z} \times m}$ that are drawn from the same distribution as $\mathbf{X}$. Let $\mathbf{X}_1 = \begin{bmatrix} \mathbf{X} \\ \mathbf{Z} \end{bmatrix}$, see Figure 1.3. The set of features of $\mathbf{X}_1$ is exactly the same as $\mathcal{F}'$. Intuitively, since $\mathbf{X}$ and $\mathbf{X}_1$ belong to the same domain problem and have the same set of features, we expect $\{\mathcal{F}'\} \approx \{\mathcal{F}'_1\}$ to hold, where $f(\mathcal{F}; \mathbf{X}_1, \mathbf{y}) \rightarrow \{\mathcal{F}'_1\}$. To evaluate the similarity of $\mathcal{F}'$ and $\mathcal{F}'_1$, we utilize a similarity metric $\mathcal{S}(\cdot, \cdot)$ that returns a value representing

the similarity. $\mathcal{S}(\cdot,\cdot)$ may evaluate the amount of overlap between $\mathcal{F}'$ and $\mathcal{F}'_1$, the correlation or other kind of similarity measure.

If $\mathcal{S}(\mathcal{F}', \mathcal{F}'_1)$ is large, the algorithm $f(\cdot)$ is said to be stable. Otherwise, it is said to be unstable. The goal is this dissertation is to maximize $\mathcal{S}(\cdot,\cdot)$ while improving or at least maintaining the learning performance (i.e. classification accuracy).

1.2 Motivation

We will motivate our work using a real-world example. Given a dataset $\mathbf{X}$ that contains n microarrays corresponding to a certain disease, say Colon Cancer. $\mathbf{y}$ contains binary classes [0,1], where 1 means the sample is cancerous and 0 otherwise. If we apply a feature selection method $f(\cdot)$ on $\mathbf{X}$ to select the k-top relevant features, $\mathcal{F}'$, we assume these features are strongly relevant to $\mathbf{y}$. Therefore, adding a new sample $\mathbf{x}_i$ that was harvested for the same purpose to $\mathbf{X}$ should not significantly impact the selection. In other words, the selected results with or without $\mathbf{x}_i$ should not change much. However, It is observed that with an amount of perturbation on $\mathbf{X}$, $f(\cdot)$ may select significantly different subset of features. This degrades the confidence of domain experts in the selection algorithm and the selected subset. Also, it misleads any further domain analysis.

Motivated from these observations, we investigate the selection stability and attempt to improve it. One might say, an intuitive way to improve stability is to randomly select the same subset of features always. However, this is meaningless since the main goal behind the feature selection is select features that are relevant to the class label and able to provide reasonable learning performance. Thus, this approach is not desired. Our approach, instead, is to solve this problem by curing the main reasons behind selection instability.

1.3 Challenges

There are several challenges that we encountered throughout this dissertation:

1. ***The curse of dimensionality*** is the main reason that feature selection is indispensable step in any data mining or machine learning task. For example, one microarray that we use in this work has more than 20,000 features and only 85 samples. The number of relevant features in such dataset usually is very small, say around 100 features only, comparing to the huge m. The rest of the features are irrelevant to the problem. This is known to be harmful if we want to build the model on the dataset without selection. Furthermore, feature selection methods themselves face great challenges dealing with such huge dimensionality. These challenges include: scalability and efficiency. Yet, one challenge that is very related to selection stability is the existence of several subsets of features that preform equally good on terms of learning performance.

2. ***Small sample size*** is another challenge. As we mentioned above that the number of samples n in most of the datasets utilized throughout this work is mostly around 100 samples. That is normal to see in such domain since it is hard to obtain more samples. In fact, this make feature selection to be NP-hard problem. It is found that the number of samples is an important aspect that is strongly connected to stability as well accuracy. If the sample set does not cover the whole hypothesis area, the selected features may not generalize to unseen data.

3. ***Data noise*** exists in such dataset in different forms. Irrelevant features may be considered as noisy features. Misclassified samples can be considered as noise, as well, since they misguide the selection search in the unsupervised

feature selection. In addition, technologies that collect the data may introduce some data noise. This is found to degrade the learning and the stability as we will see later in this work.

4. ***The number of selected features*** imposes another challenge in this work. In other words, the number of relevant features is unknown, thus the number of selected features k is not known too. Similar to previous challenges, k is found to impact selection stability.

1.4 The Contribution

Our contribution in this dissertation consists of several folds:

1. How to reasonably evaluate selection stability.
2. Find underlying causes or factors that may impact selection stability.
3. Improving stability by reducing data noise.
4. Local SVD for stable feature selection.

1.5 Thesis Structure

The remaining of this dissertation is divided into several chapters. Each chapter is a natural flow of the previous one. Thus, it is strongly connected, yet, each one can standalone. The following chapters are organized as follows:

Chapter 2: Literature Review:
In this chapter, we review feature selection models. We, also, review some proposed methods and approaches that aim to stabilize selection results. In addition, the stability metrics are intensively reviewed in this chapter.

Chapter 3: Dilemma of Stability Evaluation:

After reviewing the approaches to evaluate selection stability, we investigate whether that is a reasonable way to estimate the stability of an algorithm. We found out that in order to estimate the stability of an algorithm and to say if it is stable or not, we need to take the variation between folds into consideration.

Chapter 4: The Cause of Selection Instability:

An important start to improve selection stability is to know the factors that affect the stability. We found out that these factors include data noise, sample size, dimensionality and others. Most of these factors found to be related to the dataset itself. Thus, we conclude that the stability is mostly data-dependent.

Chapter 5: SLRMA Framework for Stable Feature Selection:

Since the stability is mostly data-dependent, we proposed a Supervised Low Rank Matrix Approximation (SLRMA) framework in order to obtain lower ranked matrix which known to be less noisy in order to select more stable sunsets. This framework was fond to be effective in terms of stability and learning performance. Also, it was found to be superior in the precision of selecting relevant features than baseline approaches. However, one limitation of this framework that it cannot handle unlabeled data.

Chapter 6: LSVD for Stable Feature Selection for Clustering:

To overcome the shortcomings of the SLRMA, Local Singular Value Decomposition (LSVD) was proposed to handle unlabel data and provide stable selection for clustering.

Chapter 7: Conclusion and Future Directions:

Finally, we conclude the findings and the contribution and point out the limitations and the possible future directions.

Chapter 2

LITERATURE REVIEW

2.1 Feature Selection Algorithms

Dimensionality reduction can be either feature selection or feature extraction[29, 74, 84]. The latter, such as PCA, reduces data dimensionality by projecting the data into lower dimensional space. Although feature extraction has been successfully utilized to reduce dimensionality and improve learning performance, the new feature space does not represent the original one[81, 84]. In other words, the new features are not physically liked to the original features, hence, meaningless. Therefore, they neither can be used to justify the reduction nor for further domain analysis. Feature selection, on the other hand, does not suffer from this limitation. It selects a subset of the original features without any kind of transformation [6, 8, 4]. Therefore, the selected features keep their physical meaning, hence, justification and further domain analysis is possible.

A huge number of feature selection methods have been proposed to handle this problem differently. Feature selection methods can be broadly categorized with respect to utilizing the data labels into *supervised* and *unsupervised*. These two categories can be further categorized with respect to the utilization of learning method into: *filter*, *wrapper* and *hybrid* model. The following subsections briefly introduce these different categories.

Supervised Feature Selection

Some datasets are collected according to some given labels. For example, harvesting genome data for cancerous and non-cancerous or capturing photos of faces or objects for object and face recognition tasks. In some cases, data samples are being manually

labeled by domain experts. These labels, or *hypotheses*, are very useful in machine learning and data mining tasks. In fact, they become very useful in feature selection. Feature selection methods utilize the given labels to guide the feature search. In particular, with the existence of the class label, we can define the relevance measure. The feature is relevant to the class if it correlates with the class. Class labels can be used, also, to study the statistical relations and characteristics of samples that affiliate to the same class [57, 29, 53]. This kind of feature selection is called supervised, since the space search and feature evaluation is supervised by the given labels. There are enormous number of proposed supervised feature selection in the field so far. Some of them will be briefly mentioned when we discuss the feature selection model in the coming subsections.

Based on whether the feature selection process involves employing a learning algorithm to guide the search, supervised feature selection can be categorized into the following models:

Supervised Filter Model

Feature selection algorithms of filter model are independent of any classifier. The selection depends totally on the characteristics of the dataset itself with respect to the class label. For example, Fisher score evaluates each feature independently using fisher criterion [27]. Other methods uses different criteria to evaluate features' relevancy. Spectral feature selection SPEC [97] and Laplacian score [35] both select feature based on the analysis of the eigensystem. Another family is *lasso* [82]. It has attracted large number of researchers and shows significant success in feature selection lately [58, 60, 99]. Lasso penalizes the estimator with ℓ_1 norm. Therefore, a sparse weight will be produced where most of the features will be given zero weight. A variety of lasso versions where proposed to handle different data structures including:

Group Lasso[60], Overlapping Group Lasso[39, 94], Graph Lasso [39], and so on. A recent review regarding lasso and its variations may be found here [90].

Filter model is known to be very efficient and mostly scalable and generalizable since it is independent of any classifier. Due to these advantages most of the proposed methods belong to this model. However, it might not be as accurate as wrapper model especially if the classifier is known beforehand.

Well-known filter algorithms include: Information Gain [14], ReliefF [86], Chi Square [57, 86], Gini Index, t-test, FCBF [93], CFS [31], MRMR, and so forth.

Supervised Wrapper Model

Unlike filter model, wrapper model utilizes a classifier to evaluate the quality of the selected features [44, 50]. It start by selecting a subset of features, usually using greedy search strategy. Then, the given classifier evaluates the quality of the selected subset. If the quality is satisfactory, the selection stops. Otherwise, it searches for another, perhaps, better subset. This is very expensive and time consuming approach comparing to the filter model. Yet, the selected features using wrapper model are more accurate with respect to the given classifier than the filter model.

Different search strategies could be combined with any classifier and produce one possible wrapper feature selection method. For example, Recursive Feature Elimination Support Vector Machine(RFE-SVM) is widely utilized wrapper approach [30]. Also, ℓ_1 norm SVM could be considered as an embedded version of wrapper approach [11] although it has better complexity than other wrapper methods [74].

Supervised Hybrid Model

In order to overcome the limitations of the previous models, a hybrid model were proposed to bridge the gab and to provide reasonably efficient and accurate selection [16]. It follows filter model in the search step, where it selects small number of candidate subsets of features. Thus, unlike wrapper approach, hybrid evaluates the quality of small number of candidate subsets, which lead to less complex model. The selected subset is the one that produces the best classification accuracy. Accordingly, the hybrid model is more efficient than filter and less expensive than wrapper.

Similar to wrapper, different combinations of filter criteria and classifiers may produce new hybrid techniques. For example, Improved F-score and Sequential Forward Floating Search (IFSFFS) [87] combines F-score with Sequential Forward Floating Search and SVM to achieve high accuracy efficient selection. Similarly, Correlation-based Feature Selection with Taguchi-genetic algorithm (CFSTGA) achieved very high classification accuracy with kNN [13]. Other hybrid techniques may be found in [15, 67].

Unsupervised Feature Selection

In a vast majority of domains, collecting labeled data or manually labeling data is intractable. Usually, in this case, we do not have domain knowledge to guide the feature selection. Therefore, feature selection in the absence of class label is very challenging problem. Assume, we collect text documents from different newswires and our goal is to cluster them. Unlike labeled data, in this case each document may belong to more than one cluster. In other words, each document may have more than one underlying hypothesis. For example, a piece of news, say "a smart chip built by Intel to monitor athletes activities", may belongs to technology, economy,

health, and sport. If we do not have these topics at hand before trying to cluster the documents, optimal clustering is almost impossible. Each one of these topics has its own corresponding subset of features. In this example, the feature *athletes* belong the topic sport, while technology has the features: *chip* and *Intel* and so on. Therefore, if we do not have these topics or labels at hand when performing feature selection, we cannot measure the relevancy score.

Different methods have been proposed to handle feature selection problem in the absence of class label. One common approach is to automatically generate labels for the given samples before selecting features. These generated labels, then, will be utilized to guide the feature search similarly as the supervised feature selection. Some methods employ *k-means* clustering to generate the labels [10, 37, 64, 42]. While other methods use more complicated approaches such as harnessing spectral analysis to extract the underlying clusters[97, 12, 54]. Spectral Feature Selection (SPEC) [97] is an example of the latter, yet, it can handle supervised data as well as unsupervised. Thus, it is a unified feature selection method.

Entropy Weighting K-Means (EWKM) was proposed for subspace clustering. It simultaneously minimizes the within-cluster dispersion and maximizes the negative weight entropy in the clustering process [42]. It utilizes k-means to find the clusters before doing feature selection. This step is repeated several time until convergence.

Cai et al proposed Multi-Cluster Feature Selection (MCFS) [12] that used spectral analysis to measure the correlation between different features without label information needed. Using the top eigenvectors of graph Laplacian, spectral clustering can cluster data samples without utilizing label information.

Other methods evaluate feature's weight independently of any clustering techniques. Term Frequency (TF), Inverse Document Frequency (IDF) and TF-IDF are

among the most popular feature weighting, (aka term selection) techniques especially in text mining domain. Other methods cluster the features and select a representative one of each cluster to be the selected features [62, 38].

Similar to feature selection for supervised learning, methods of feature selection for clustering are categorized into filter [17] wrapper [71], and hybrid models [23]. A wrapper model evaluates the candidate feature subsets by the quality of clustering while filter model is independent of clustering algorithm. Thus, the filter model is still preferable in terms of computational time and unbiased toward any clustering method, while the wrapper model produces better clustering if we know the clustering method in advance. To alleviate the computational cost in the wrapper model, filtering criteria are utilized to select the candidate feature subsets in the hybrid model.

In the following subsections, we will briefly discuss feature selection for clustering methods that falls in the filter, wrapper and hybrid models. For more about conventional methods, we refer the reader to [23].

Unsupervised Filter Model

Similar to supervised filter model, Methods that belong to unsupervised filter model do not utilize any clustering algorithm to test the quality of the features [23]. They evaluate the score of each feature according to certain criteria. Then, it selects the features with the highest score. It is called the filter since it filters out the irrelevant features using given criteria. Furthermore, feature evaluation could be either *univariate* or *multivariate*. Univariate means each feature is evaluated independently of the feature space. This approach is much faster and more efficient than the univariate, which evaluates features with respect to the other features. Therefore, the

multivariate, unlike the univariate approach, is capable of handling redundant features. SPEC is an example of the univariate filter model, although it was extended to multivariate approach [98]. Other examples of filter model criteria used in feature selection for clustering include: feature dependency [80], entropy-based distance [17], and laplacian score [35, 97].

Unsupervised Wrapper Model

The wrapper model utilizes a clustering algorithm to evaluate the quality of selected features. It starts by (1) finding a subset of features. Then, (2) it evaluates the clustering quality using the selected subset. Finally, it repeats (1) and (2) until the desired quality is found. Evaluating all possible subsets of features is impossible in high-dimensional datasets. Therefore, heuristic search strategy is adopted to reduce the search space. The wrapper model is very computationally expensive compared to filter model. Yet, it produces better clustering since we aim to select features that maximize the quality. It is still biased toward the used clustering method. Different wrapper feature selection methods for clustering were proposed by changing the combination of search strategy and the utilized clustering algorithm. Feature Subset Selection wrapped around EM Clustering (FSSEM) was proposed in [24] to select a subset of features by first clustering using EM and then evaluate the resulting clusters and feature subset using the chosen feature selection criterion. In addition, the method proposed in [25] is another example of a wrapper that involves maximum likelihood criteria and feature selection and mixture of Gaussians as clustering method. Others use conventional clustering methods such as k-means and any search strategy as feature selector [48].

Unsupervised Hybrid Model

To overcome the drawback of filter and wrapper models a hybrid model is used to benefit from the efficient filtering criteria and better clustering quality from the wrapper model. A typical hybrid process goes through the following steps: (1) it utilizes filtering criteria to select different candidate subsets. Then, (2) it evaluates the quality of clustering of each candidate subsets. (3) The subset with highest clustering quality will be selected. Algorithms belonging to the hybrid model usually produce better clustering quality than those of filter model, yet, they are less efficient. Compared to the wrapper model, the hybrid model is much more efficient.

2.2 Selection Stability

Generally, the selection stability is a desired characteristic for feature selection algorithms. Since the target concept of a data is fixed, the relevant features should not change across different samples of the data. In real-world applications, such as genetic analysis, domain experts expect algorithms to select features that are always consistent even if there are new samples introduced to the data, as unstable feature selection results will confuse them and lower their confidence with the results [19, 28]. The topic of selection stability (this will be used interchangeably with stability of feature selection algorithms) has recently gained intensive attention in the research community. It is defined as *the sensitivity of a feature selection algorithm to perturbation in the training data* [47, 68, 91, 28, 36]. The perturbation of the data could occur in different format. For example, a new sample can be considered as a perturbation. Data noise or outlier samples are other perturbation format.

Stable Feature Selection Algorithms

Several methods have been proposed to improve stability of feature selection algorithms [88, 28, 36, 91, 92, 32, 1, 34, 33, 61]. These methods can be broadly categorized based on the used approach to handle selection stability to the following categories:

1. Ensemble approach
2. Data variance reduction approach
3. Density-based approach
4. Trade-off approach

In the following, I will review some feature selection algorithms that aim to improve stability and discuss their advantages and disadvantages.

Ensemble approach

Intuitive approach for feature selection to improve learning performance is the ensemble learning technique since more than one method are jointly supporting each other to choose the best features. It was, also, found that using different feature selection criteria that produce similar results to form an ensemble method will not help to improve learning performance [88]. In other words, the more diverse methods, the more likely they will complement each other. Using similar motivation, Saeys et al. in [73] proposed to use ensemble feature selection to improve rank and subset stability of four well-known methods, namely: Symmetrical Uncertainty, Relief, Support Vector Machine classifier with Recursive Feature Elimination (SVM-RFE), and Random Forest. All these methods showed significant improvement in the stability in both cases (i.e. rank and subset stability).

Similarly, [1] introduced an ensemble technique to improve selection stability. They utilized SVM-RFE to perform feature selection that, hopefully, leads to more stable selection since the performance of such technique was proven to be effective. T. Abeel et al. in [1] were able to improve selection stability of biomarker identification of microarray dataset using ensemble approach. They started by ranking all features using SVM classifier. Then, they eliminate features that correspond to the least scored features. These two consecutive steps are iteratively repeated until all features are removed. Finally, the final feature score is aggregated using a linear combination of all scores in all iterations.

RFE-SVM was able to improve the stability of biomarker identification in all used datasets. In addition, it is robust against both the number of selected features and the number of eliminated features in RFE step. Furthermore, it was able to eliminate irrelevant features.

There are several reasons that may have led to this improvement in the stability when using ensemble technique. First, usually, there may exist different subsets of features that perform equally good in terms of classification accuracy. Similarly, each selection criterion may prefer a subset of features over the other. However, with existence of perturbation, the selection criterion may give slightly different weight for each feature, which lead to change in the selected subset. These new selected features are most likely placed at the bottom of the list since the small amount of perturbation should not cause a huge change in the selected subset. Therefore, selecting the most relevant features based on different criteria leads to selecting features with the highest weight which are less effected by small amount of perturbation. In fact, that is the essence of the ensemble approach. Thus, missing the remainder of these potential result. Therefore, ensemble method reduces the risk of missing these

results and, also, reduces the risk of choosing wrong result. Second, aggregating the final result leads to diverse features that complement each other.

In contrast to these results, A.C. Haury et al. [33] have found that ensemble methods, surprisingly, have no significant impact on the selection stability. The empirical results shown in [33] were conducted using 9 different well-known selectors including from different models. They found that simple filter methods simply produced as good stability in average as the ensemble technique.

I believe that the contradiction in the conclusion between [1, 73] on one hand and [33] on the other hand is due to the sampling technique not due to the ensemble methods. It is observed that different sampling techniques may lead to different stability results. For example, when we use 10-fold cross-validation to generate samples, we end up with around 80% overlap between folds, while the overlap will be less when we use less than 10-folds, and so on. The amount of overlap between samples has a significant impact on the stability where the amount of perturbation will be less when larger overlap occurs. Another reason to inconsistent results could be the aggregation technique. Different techniques may lead to different stability results. Thus, we need further analysis and empirical experiments to conclude whether ensemble technique leads to better stability or not.

Some limitations of the ensemble technique include the choice of the feature selectors. If the selectors produces similar results, the stability will be higher. However, the diversity, which is the main goal for ensemble technique, will be lost. On the other hand, when the selectors generate diverse results, the stability will be lower. Therefore, we may need to have a trade-off parameter between stability and diversity in the ensemble techniques.

In addition, the aggregation method and the sampling techniques, as we discussed earlier, may have significant impact of the stability. For example, the score domination of one selector may has significant impact on the overall results. Hence, a normalization technique should be selected carefully. Also, the small sample size of the original dataset may be further reduced in the sampling process which leads to in accurate feature scoring. All things considered, further studies for these factors are required.

Sample weighting and variance reduction approach

Another way, yet, affective to improve stability is to train the model on samples from desired region in the space. Since the stability is mostly impacted by perturbation in the dataset, we may be able to improve stability by reducing this perturbation (i.e. variance between samples). Han and Yu [32] proposed a general framework for stable selection via variance reduction. Instead of refusing less desired samples, this approach assigns higher weights to preferred samples. Thus, the algorithm may benefit, also, from useful information gained from samples with less weights. The proposed framework in [32] assigns sample weight by, first, transforming the original sample x into a new sample x' in the margin space. Projecting each sample according to Eq(2.1) leads to capture the local feature relevance where the larger the value of x'_j the more the j^{th} feature contributes to the margin of sample x.

$$x'_j = |x_j - x_j^M| - |x_j - x_j^H| \tag{2.1}$$

Where x_j^M and x_j^H are the nearest miss and hit respectively. This transformation is very sensitive to outliers and noise in the dataset. Therefore, [32] suggests to use more than one nearest neighbor from each class.

$$x'_j = \sum_{l=1}^{n_1} |x_j - x_j^{M_l}| - \sum_{l=1}^{n_2} |x_j - x_j^{H_l}|$$

Where $n_1 + n_2$ equals the total number of instances in the training set excluding the given instance. The second step is to weight each sample **x'** based on the average distance between **x'** and x'_i, where $i = 1, \ldots, n-1$ and **x'** $\neq x'_i$. This weight will be used in any feature selection method that accept sample weighting.

One advantage of this algorithm is the computational efficiency, where the running time will be dominated by the distance computation in the transformation step, which is $O(n^2 * m)$, where n and m are the sample size and the dimensionality respectively. In addition, this approach tackles the problem of selection instability from the dataset prospective, which has the most significant impact of the stability. Since the stability is mostly data dependent, curing this issue should start from curing the dataset. One aspect that may cause instability is the variation in the dataset, hence, reducing the variation in a class-wise approach (i.e. hypothesis margin) is meaningful approach to solve selection instability, which is nicely done in this algorithm.

Yet, there is, always, room for improvement. For example, the weighting scheme in this algorithm is not strongly resistant to outliers. Distance-based weighting may not be the most appropriate method. For example, in the situation where high level of data noise exists, this weighting scheme may assign equally high weight to good and bad samples. This could be overcome via normalizing the summation of the sample entities (i.e. the weight $w(\mathbf{x}) = \sum_j^m x'_j$). Although this suggested approach of weighting seems very simple, it is effective since the vector **x'** will be better if all its values are in the far positive corner of the space. Therefore, $w(\cdot)$ will assign higher weights to samples in this desired region (i.e. positive corner) and the lowest weights to these in the far negative corners.

Density-based approach

In this approach, feature selection algorithm aims to group features into clusters of similar densities. Then, it selects the representative feature(s) from each cluster. Yu et al. in [91] proposed a Dense Group Finder (DGF) algorithm to find a number of unique density peaks in the data using kernel density estimator, namely: Parzen window. Then, DGF merges features to the closest peak if the distance is less than the window size h. The density is evaluated using the mean shift procedure according to Eq (2.2), which is proven to converge if the kernel K has a convex and monotonically decreasing profile.

$$x'_{j+1} = \frac{\sum_{i=1}^{n} x_i K(\frac{x'_j - x_i}{h})}{\sum_{i=1}^{n} K(\frac{x'_j - x_i}{h})} \tag{2.2}$$

where x'_j is always initialized with data vector $x_i \; \forall i = \{1, \ldots, n\}$ and computed until convergence. The dataset in this algorithm is transposed, so that the dimensionality m in the new space is the number of samples in the original dataset and the number of samples n became the dimensionality. This algorithm has a complexity of $O(\lambda n^2 m)$, where λ is the number of iterations.

Not all generated groups are relevant to the problem. Thus, after finding the dense feature groups, we need to evaluate the relevancy score for each group. Dense Relevant Attribute Group Selector (DRAGS) were proposed to evaluate how relevant each group is using *F-statistic* to identify differentially expressed genes. Finally, DRAGS will select the top k representative features from the top k groups according to the relevancy score.

One advantages of this algorithm is the simplicity, where it is very simple to find the clusters and to evaluate their relevance score. Also, It is able to handle

the redundancy among the features, which is desired property in feature selection literature. In addition, the density estimator was found superior to the k-means in finding robust clusters, which leads to better stability in feature selection step.

In addition to the previous advantages, DRAGS overcomes one important cause of instability, which is the small sample size, by ensuring the stability of clustering by evaluating density of features, which is very large number comparing to the sample size.

On the other hand, there are several disadvantages for this approach. First, this approach is built upon the claimed observation that the dense peak regions measured by the density estimation are stable with respect to the sampling of the dimension. This may be true, however, I cant see why this will lead to more stable selection. Since the representative feature is very sensitive to sampling. In other words, the representative feature is not necessarily to be the same across different folds, which degrade the selection stability.

In addition, choosing the window size h is an open problem which could be a limitation for this algorithm. If h is sufficiently small, each feature will form a group by itself. On the other hand, extremely large h leads to group all features in one cluster. Thus, carefully choosing h is necessary step toward successful clustering. In [91], h is estimated by calculating the average distance between samples in the space using k-Nearest Neighbors (kNN). This may capture the local density around each sample. In addition, the number of generated groups is not controlled. In other words, DGF may generate very small number of clusters that is way smaller than the number of selected features, k, determined by the user.

Another limitation in this approach is the relevance measure where it is a domain specific measure. In other words, *F-statistic* is used to identify differentially

expressed genes. Therefore, *F-statistic* may not generalize to different domains, e.g. images datasets. Thus, using a more general measure may overcome this limitation with non-microarray datasets.

Trade Off Parameter α

The stability of Minimum Redundancy Maximum Relevance MRMR feature selection algorithm was theoretically analyzed [28] with two different mutual information techniques that is used with MRMR. It was theoretically and empirically demonstrated that MID (Mutual Information Difference) is more stable than MIQ (Mutual Information Quotient). So, [28] proposed to trade off between stability and accuracy by controlling the relevancy value V and the redundancy value W which is obtained by adding a parameter α to MID as follows:

$$MID_\alpha = \alpha V - (1 - \alpha)W. \tag{2.3}$$

Based on the results shown in [28], it was not clear how this approach improves the stability. Also, different datasets prefer different α values. Thus, tuning α is another issue in this approach. However, choosing α to be 1 seems to give good accuracy and stability. This quite surprising since $\alpha = 1$ means to get rid of the redundancy value which will lead to more redundant features to be selected which also proven to be useless in terms of prediction accuracy.

2.3 Stability Measurements

With the increase attention on the stability of feature selection methods, the necessity of finding a good way to assess the stability increases too. Several methods have been proposed to assess the stability with different results of feature selection process. These measurements can be mainly categorized into three broad categories based on

the representation of the output of the selection method. These three categories introduced in [47]. The first category, *stability by index*, deals with indices of the selected features where the selected subset could be represented either as a subset of features' indices or as a full set of binary numbers where 1 means the corresponding feature is selected and 0 otherwise. In this category, the selected features will have no particular order or corresponding relevance weight. In contrast to stability by index,the second category, *stability by rank*, is a ranked list where the features order makes difference in the stability evaluation. In this category, each feature will be given a rank from 1 to m, where 1 is the most relevant feature and m is the least relevant one. Last category is *stability by weight*, where each feature is assigned a weight according to the degree of relevance.

In fact, all these three output representations are generated from the features' weights. However, different domain will be interested in different output and thus will be interested in the stability of that particular output. It is important to emphasize that same rank does not necessary mean same wight and same selected subset. In this work, we will follow this category when we investigate the stability measurements. In spite of these three categories, [51] proposed three requirements that each stability measurement should have:

1. **Monotonicity:** the larger the overlap between selected subsets, the larger the stability result should be.

2. **Limits:** each stability assessment method's result should be bounded between constants; for instant [0,1] or [-1,1]. Where these bounds are independent of any dataset factor such as the dimensionality of the dataset m or the number of selected features k. These limits should be minimum when the sets are completely unstable and maximum when they are identical or stable.

3. **Correction for chance:** the measurement should have a constant that correct the result in case of intersection by chance occur due to high dimensional selected subset where it is proven that the larger the cardinality of selected subsets the more chance for larger intersection between subsets.

We show in this work that these three desired properties were not taken in account during the design of each measurement. The only measure that consider all these requirements is the Kuncheva Index (KI) [51]. Most of the measurements will be discussed here obey the first requirement, the monotonicity, while it is rare to find a measurement that have correction for chance constant. Moreover, the limits varied from [0,1], [-1,1], to unbounded measurements. Late in this chapter, we will mention these requirements when we discuss the measurements.

In addition to these requirements, there are some important properties that, we believe, should be taken into consideration due to their impact on the stability result. These properties include: (1) the *dimensionality* of the dataset m is an important factor that may affect the stability of an algorithm. Also, (2) the *number of selected features* k. These two factors implicitly mentioned in the correction for chance requirement. However, they should be considered in other ways too. For example, in order to rank two algorithms in terms of the stability, we should take in mind these two factors, i.e. m and k. In addition, (3) the *sample size* n have a significant impact on the stability as we shown in Chapter 4. Thus, considering these three factors may help justifying the differences in the stability of one algorithm. Another important factor that should be considered is (4) the *data variance*. It was demonstrated in [32] that the data variance has a huge impact on the stability. Thus, it is not fair to judge or compare algorithms in terms of stability without taking the variance of the dataset and perhaps other important underlying characteristics of the dataset

into consideration. Furthermore, (5) the *symmetry* of the measurement is another desirable property so the stability value should not be sensitive to the order of the results.

According to the definition of the stability of the feature selection methods that defines the stability as the sensitivity of the selection to the variation (i.e. perturbation) of the dataset. We may assess the stability simply by pairwise comparison between the results. Therefore, the stability is higher if the similarity is greater. Since there are three different representations of the output of the feature selection methods, weighting, ranking, and indexing [47], different measures are used to fit different representations. To the best of our knowledge, [47] is the first work to propose a measurement for each kind of output. Here, we will go over these measurements and others categorized by the output scheme.

Stability By Index

In this category, the selected subset of features is represented as either a vector of indices that correspond to the selected features $\mathcal{F}' \subset \mathcal{F}$; or as a binary vector $\tilde{\mathbf{f}}$ with cardinality equals m, where $\tilde{\mathbf{f}}_i = 1$ means that the i^{th} feature is selected. The common property among these measurements is that they can handle number of selected features $k \leq m$ which is not the case in the rank or weight measurements. Otherwise, these measurements do not have any common result's limits where some in the interval [0,1] and others in [-1,1] while others are not bounded at all. However, most measurements in this category attempt to assess the amount of overlap between results in order to assess the stability. Follows, the stability by index measurements:

1. **Average Normal Hamming Distance (ANHD)**

   Average Normal Hamming Distance measure was used in an early work in [22],

to assess the stability of feature selection algorithm, which meant to be used for subset of selected features. ANHD measures the amount of overlap between two subsets. ANHD ($\hat{H}$) works with binary representation that represent the selected feature subset $\vec{\mathbf{f}}_{ik}$, 1 and 0 indicate whether the k^{th} feature was selected in the i^{th} run or not, respectively.

$$\hat{H}(\tilde{\mathbf{f}}_i, \tilde{\mathbf{f}}_j) = \frac{1}{m} \sum_{k=1}^{m} |\tilde{\mathbf{f}}_{ik} - \tilde{\mathbf{f}}_{jk}| \tag{2.4}$$

The larger m is, the smaller $\hat{H}$ will be which indicates more stable algorithm. In addition, when small number of features were selected, i.e. have values equal to 1, and the rest are set to zero, then $\hat{H}$ will be small as well. This is due to the fact that selected features across ℓ-folds will be treated as unselected ones. In other words, if a feature f_i is selected in all ℓ or not selected, will have the same impact on the stability result. This property of ANHD will lead in most cases to wrong conclusion about the stability especially when $k \ll m$ where the majority of the features are not selected. In terms of the results, ANHD is in the interval [0,1], where 0 is the most stable and 1 means not stable at all. In terms of capability, ANHD cannot deal with different sizes of selected features' sets. Also, there is no correction for chance constant in ANHD, so, the result will be misleading.

2. **Dice's Coefficient**

   Dice coefficient is a similarity measure related to the Jaccard index Eq(4.2). It was used in [91] to calculate the overlap between two sets.

$$Dice(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{2|\mathcal{F}'_1 \cap \mathcal{F}'_2|}{|\mathcal{F}'_1| + |\mathcal{F}'_2|} \tag{2.5}$$

Dice takes value between 0 and 1, where 0 means no overlap and 1 means the two sets are identical. Due to the similarity between Dice and Tanimoto and Jaccard, we will discuss it in more details when we discuss them.

3. **Tanimoto Distance and Jaccard's Index**

   Similarly, Tanimoto Eq(2.6) measures the amount of overlap between two data sets and produces value in the same range as Dice does.

$$Tanimoto(\mathcal{F}'_1, \mathcal{F}'_2) = 1 - \frac{|\mathcal{F}'_1| + |\mathcal{F}'_2| - 2|\mathcal{F}'_1 \cap \mathcal{F}'_2|}{|\mathcal{F}'_1| + |\mathcal{F}'_2| - |\mathcal{F}'_1 \cap \mathcal{F}'_2|} \tag{2.6}$$

   It is easy to proof that Tanimoto is equivalent to Jaccard's index Eq(4.2) [73]:

$$Jaccard(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{|\mathcal{F}'_1 \cap \mathcal{F}'_2|}{|\mathcal{F}'_1 \cup \mathcal{F}'_2|} \tag{2.7}$$

   In general, Dice, Tanimoto, and Jaccard behave similarly in all cases although it is noticeable that Dice sometimes give slightly higher and more meaningful stability results with respect to the intersection between the two subsets. For instance, assume we have two selected subsets with equal length, $k = 10$, and they intersect in 5 features, which is exactly 50% of total number of features for each set. Dice, in this case, is going to give a stability equals to this exact amount of overlap (namely: 0.5), yet, Tanimoto and Jaccard are going to be 0.33 for each of them due the fact that they divide by the length of union of the two selected sets. Another issue with these three measurements that they give higher values when the subsets cardinalities get closer to m, where the chance for more overlap by chance is higher. Thus, they don't have constant to correct in case of intersection by chance. An advantage of these measurements, unlike ANHD, they can deal with sets of different cardinalities. Beside that, they do not take the dimensionality m in account, yet, they comprise the number of selected features k in the measurement.

4. **Kuncheva Index** KI

   The drawback of most stability measurements is that the larger the cardinality of the selected features' lists, the more overlap between lists due to chance. Therefore, [51] proposed Kuncheva Index KI that contains correction term to avoid intersection by chance between the two subsets of the features which overcome the drawback of the previous measurements.

$$KI(\mathcal{F}_1', \mathcal{F}_2') = \frac{|\mathcal{F}_1' \cap \mathcal{F}_2'| \cdot m - k^2}{k(m-k)}. \tag{2.8}$$

   KI's results ranges [-1,1], where 1 means that $\mathcal{F}_1'$ and $\mathcal{F}_2'$ are identical which means the cardinality of the intersection set equals k. KI achieves -1 when there is no intersection between the lists and $k = \frac{m}{2}$. KI assumes values close to zero for independently drown lists. Furthermore, KI is the only measurements that obeys the requirements appeared in [51]. The correction for chance term that was introduced in [51] makes KI desirable. In other measurements, the larger the cardinality is, the higher the stability will be. However, this is not the case with KI where the larger the cardinality will not affect the stability value. Figure 2.1 shows the impact of the number of selected features k on the stability in *Jaccard Index* where it gives higher stability values when k gets larger and closer to m. However, KI does not suffer from the same drawback where the correction term gives negative weight to k.

5. **Percentage of Overlapping Gene** (POG)

   POG is used to measure the consistency of the feature subsets by counting the amount of overlap between them. Therefore, it is similar in a sense to the Tanimoto and Jaccard measures. However, POG is not symmetric and thus $POG(\mathcal{F}_1', \mathcal{F}_2')$ is not necessary equal to $POG(\mathcal{F}_2', \mathcal{F}_1')$, which is undesirable

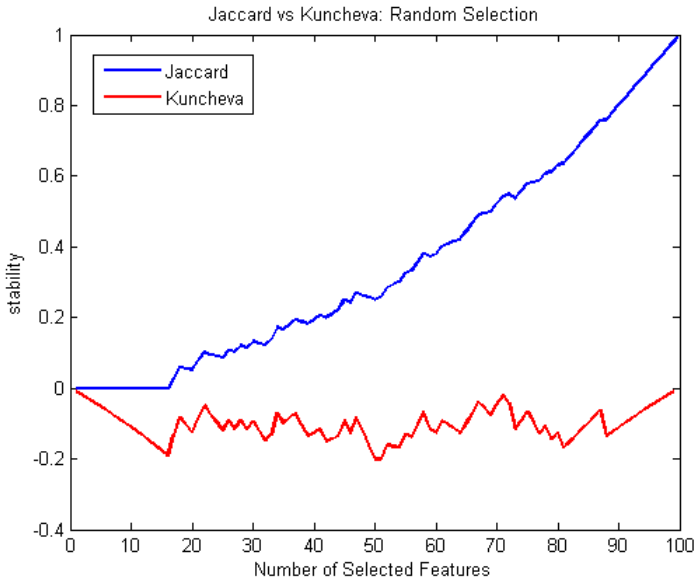


Figure 2.1: The effect of the number of selected feature on KI vs. Jaccard. This demonstrates the importance of correction for chance in the measurement.

property in general. However, it will be symmetric if $|\mathcal{F}'_1| = |\mathcal{F}'_2|$ [95] proposed a matrix that introduced a new variable z into POG that consider the correlated molecular changes in a biological data set. [95] defined $POGR$ percentage of overlapping genes, or features, related matrix to evaluate the consistence between two differentially expressed genes lists.

$$POG(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{|\mathcal{F}'_1 \cap \mathcal{F}'_2|}{|\mathcal{F}'_1|} \tag{2.9}$$

$$POGR(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{|\mathcal{F}'_1 \cap \mathcal{F}'_2| + z}{|\mathcal{F}'_1|} \tag{2.10}$$

Where z represents the number of genes in $\mathcal{F}'_1$ that are not in $\mathcal{F}'_2$ but they are significantly positively correlated to at least one gene in $\mathcal{F}'_2$. By having z we are overcoming one drawback of the previous measurements. All previous measurements ignore the redundancy or the correlation between the values of

the features. For illustration, assume $f_i \in \mathcal{F}'_1$ and $f_j \in \mathcal{F}'_2$ but $f_i, f_j \notin (R = \mathcal{F}'_1 \cap \mathcal{F}'_2)$. In pervious measures, including POG, these two features no way to be counted positively toward the stability. In other words, f_i and f_j won't be considered as one feature even if they are redundant or positively highly correlated. However, by introducing z, we are able to capture the correlation between the feature and, thus, consider such features as one single feature.

[95], also, introduced a new matrix normalized version for POG and $POGR$, or $nPOG$ and $nPOGR$ for short, to overcome the dependency between the result and the list length by introducing the expected of the shared features $\mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|)$. In addition, they introduced the expected number of z, $\mathbb{E}(z)$ onto the $POGR$, as follow:

$$nPOG(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{|\mathcal{F}'_1 \cap \mathcal{F}'_2| - \mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|)}{|\mathcal{F}'_1| - \mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|)} \tag{2.11}$$

$$nPOGR(\mathcal{F}'_1, \mathcal{F}'_2) = \frac{|\mathcal{F}'_1 \cap \mathcal{F}'_2| + z - \mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|) - \mathbb{E}(z)}{|\mathcal{F}'_1| - \mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|) - \mathbb{E}(z)} \tag{2.12}$$

Where $\mathbb{E}(|\mathcal{F}'_1 \cap \mathcal{F}'_2|)$ can be simply estimated by the average of the scores for arbitrary number of pairs of random lists of length $|\mathcal{F}'_1|$ and $|\mathcal{F}'_2|$ respectively. Similarly, $\mathbb{E}(z)$ can be estimated as the average number of features in the list $\mathcal{F}'_1$ which are not shared but significantly positively correlated with features in the other list $\mathcal{F}'_2$. These two parameters are the correction for chance terms in these two measurements. Finally, the limits of the results in POG measures family varies. POG and $POGR$ are bounded by 0 and 1 while $nPOG$ and $nPOGR$ are in the interval [-1,1] which make the latter obey $Kuncheva$ requirements.

6. **Consistency Measures**

The previous measures' main idea is to assess the overlap between the subsets

by comparing the subsets pairwise. So the complexity is usually equal to or greater than $O(\frac{k\cdot(\ell^2-\ell\ )}{2})$, where ℓ is the number of subsets of selected features. To overcome such shortcomings, [78] proposed three consistency measures that will be superior to these in complexity time. These measures take the frequency of each selected feature in mind when calculating the stability. So, each subset is processed only once to count the frequency of each selected feature which makes the complexity to be $O(k \cdot \ell)$. The following three consistency measures take $\mathbf{S}$ as an input where $\mathbf{S} = \tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2, ..., \tilde{\mathbf{f}}_\ell$. In addition, $\tilde{\mathbf{x}}$ is the union of all subsets in $\mathbf{S}$ and t is the total frequency in $\mathbf{S}$.

a) **Consistency Measure** C

$$C(\mathbf{S}) = \frac{1}{|\tilde{\mathbf{x}}|} \sum_{i}^{|\tilde{\mathbf{x}}|} \frac{r_i - 1}{\ell - 1} \tag{2.13}$$

b) **Weighted Consistency Measure** CW

$$CW(\mathbf{S}) = \sum_{i}^{|\tilde{\mathbf{x}}|} \frac{r_i}{t} \cdot \frac{r_i - 1}{\ell - 1} \tag{2.14}$$

c) **Relative Weighted Consistency Measure** CW_{rel}

$$CW_{rel}(\mathbf{S}, m) = \frac{m(t - z + \sum_i^m r_i(r_i - 1)) - t^2 + z^2}{m(h^2 + \ell(t - h) - z) - t^2 + z^2}, \tag{2.15}$$

where r_i is the rate of occurrence, i.e. frequency, of feature f_i and z and h are $t\ mod\ m$ and $t\ mod\ \ell$ respectively. CW_{rel}, unlike the others, neither evaluates the amount of overlap between the subsets nor the frequency of the features but it, in fact, shows the amount of randomness in the feature selection process. If CW_{rel} gives small number and the others give higher numbers that may indicate drawback in the process of selecting the features; such as: there are no preferable features or the methods overfit...etc. see [78] for more details.

On the other hand, the consistency measure C can be rewritten in a less complex way that show some hidden properties of this measure. Since, we know that $t = \sum_i^{|\tilde{\mathbf{x}}|} r_i$. Then, by subtracting $|\tilde{\mathbf{x}}|$ from both sides, we obtain:

$$t - |\tilde{\mathbf{x}}| = \sum_i^{|\tilde{\mathbf{x}}|} r_i - 1. \tag{2.16}$$

Also, we can say that:

$$\sum_i |\tilde{\mathbf{x}}|(k-1) = (k-1)|\tilde{\mathbf{x}}|. \tag{2.17}$$

Since W is a constant, we can rewrite $C(\mathbf{S})$ from 2.16 and 2.17 as follow:

$$C(\mathbf{S}) = \frac{t - |\tilde{\mathbf{x}}|}{(k-1)|\tilde{\mathbf{x}}|}. \tag{2.18}$$

Equation 2.18 seems less complicated than 2.13, yet, they have the same time complexity, which is $O(k \cdot \ell)$. However, 2.18 shows us clearly that this measure does not get use of the frequency of each feature in the system S. For instance, if we have two different systems $\tilde{\mathbf{f}}_1$ and $\tilde{\mathbf{f}}_2$ with the same characteristics of t, k, and $|\tilde{\mathbf{x}}|$, we will get the same consistency result for both systems regard less of the occurred features.

7. **Symmetrical Uncertainty** *SU*

L. Yu et al in [91] and G. Gulgenzen et al in [28] used an entropy based nonlinear correlation, the so-called Symmetrical Uncertainty SU. This similarity measure is quite different from the previous ones where it considers the similarity of the feature values instead of features indices. Therefore, it satisfies nice and desirable property when evaluating the stability which is the correlation between the values of the selected features across different selected subsets. For example, assume that f_i and f_j to be duplicated feature and they where selected in $\mathcal{F}'_1$ and $\mathcal{F}'_2$ respectively. Thus, they will be consider as two different features when

evaluating the stability although they are the same. However, SU will treat them as one single feature by considering the values of the features. SU is symmetric too since the information gain $IG(f_i|f_j) = IG(f_j|f_i)$. One undesirable property of SU is the the result is not bounded by any constants. Accordingly, we normalize the stability results by the number of selected features k.

Moreover, the differences between [91] and [28] in term of using this measure is that the first one used it to calculate the similarity between two sets of feature groups. [28], on the other hand, used it as similarity measure between two sets of individual features. We, in this paper, will follow the notion of the latter.

$$SU(f_i, fj) = 2[\frac{IG(f_i|f_j)}{H(f_i) + H(f_j)}]. \tag{2.19}$$

Where f_i and f_j are i^{th} and j^{th} selected features and IG and H are the information gain and the entropy, respectively, given by:

$$IG(f_i|f_j) = H(f_i - H(f_i|f_j)) \tag{2.20}$$

$$H(f_i) = \sum_{x \in\ f_i} p(x) \cdot\ \ \lg_2(p(x)) \tag{2.21}$$

$$H(f_i|f_j) = \sum_{y \in\ f_j} p(y) \sum_{x \in\ f_i} p(x|y) \cdot \lg_2(p(x|y)). \tag{2.22}$$

Finally, the similarity between two sets will be the average of SU for all unique pairs of i and j. The SU is the most expensive measure among all measurements in this paper. This complexity is due to the expensive computations of IG for each unique pairs of selected features. It depends on the number of selected features k, where the worst case is when $k = m$. Also, we had to

normalize, discretize, and center the datasets before computing the SU which make it even more expensive.

Stability By Weight

The second category is the measurements that deal with the weight of the feature set w. In this category there is only one measurement which is the Pearson's Correlation Coefficient PCC that takes two sets of weights w_i and w_j for the entire feature set in the dataset and return the correlation between them to be the stability. Unlike the stability by index, this category cannot deal with a subset of features or with different subsets size.

1. **Pearson's Correlation Coefficient** PCC

   The authors in [47] proposed three types of stability measures depend on the presentation, i.e. output, of the selected features, as we mentioned earlier. One way to represent the selected features is by assigning weight-scores to them. [47] uses Pearson's to measure the correlation between the weights of the features that returned from more than one run. Thus, the stability will be as following:

$$PCC(w, w') = \frac{\sum_i (w_i - \mu_w)(w'_i - \mu_{w'})}{\sqrt{\sum_i (w_i - \mu_w)^2 \sum_i (w'_i - \mu_{w'})^2}} \tag{2.23}$$

   Where μ is the mean. PCC takes values between -1 and 1, where 1 means the weight vector are perfectly correlated, -1 means they are anti-correlated while 0 means no correlation. It is noticeable that when the weight is equal to zero for big number of features, which is true some of the feature selection algorithms, the stability will be shown higher. Noteworthy, this will not be an issue in

situations where the algorithm assigns weight between 1 and -1. Finally, PCC is symmetrical measure.

To best of our knowledge, PCC is the only stability measure that handle feature weights.

Stability By Rank

Similar to the stability by weighting score, stability by rank evaluate the correlation between the ranking vectors. They, also, deal with full set of features. In other words, they cannot handle vectors with different cardinality or vector that correspond to different set of features.

1. **Spearman's Rank Correlation Coefficient** $SRCC$

   To evaluate the stability of two ranked features' sets r and r', A. Kalousis et al. in [47] adapted Spearman's Rank Correlation Coefficient.

   $$SRCC(r, r') = 1 - 6 \sum_{i} \frac{(r_i - r'_i)^2}{m(m^2 - 1)} \tag{2.24}$$

   Similar to Pearson's, the result of Spearman's will be in the range of [-1,1]. The maximum will be achieved when the two ranks are identical while the minimum is when they exactly in inverse order and 0 means no correlation at all between r and r'.

2. **Canberra Distance** CD

   Canberra Distance is the absolute difference between two rank sets. The generalized form is given by:

   $$CD(r, r') = \sum_{i}^{N} \frac{|r_i - r'_i|}{r_i + r'_i}. \tag{2.25}$$

   CD does not have an upper bound. The result depends on the number of features. The higher m is, the larger CD will be. Therefore, we normalized by

dividing by m in order to obtain results between 0 and 1. A weighted version of Eq(2.25) was proposed in [45] WCD:

$$WCD^{(k+1)}(r, r') = \sum_{i=1}^{N} \frac{|\min\{r_i, k+1\} - \min\{r'_i, k+1\}|}{\min\{r_i, k+1\} + \min\{r'_i, k+1\}} \tag{2.26}$$

This specialized version of CD is due to the fact that the most important features are located in the top-k positions of the ranked list. Thus, the variation in the lower position of the list should be less relevant than those in the top part [45]. Similar to CD,WCD is normalized by the number of features.

Measurements Categories

We categorized the stability measurements based on the 4 criteria we introduced early in this work.

Measures	Results			Capability				
	Index	Rank	Weight	Different Size	Complexity	Symmetrical	Bounds	Reference
$ANHD$	★			Yes	$O(\frac{m\cdot(\ell^2-\ell)}{2})$	Yes	[1,0]	[22]
$Spearman's$		★		No	$O(\frac{m\cdot(\ell^2-\ell)}{2})$	Yes	[-1,1]	[47]
$Pearson's$			★	No	$O(\frac{m\cdot(\ell^2-\ell)}{2})$	Yes	[-1,1]	[47]
CD		★		No	$O(\frac{m\cdot(\ell^2-\ell)}{2})$	Yes	[0,∞]	[45]
WCD		★		No	$O(\frac{m\cdot(\ell^2-\ell)}{2})$	Yes	[0,∞]	[45]
$Dice$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2})$	Yes	[0,1]	[91]
$Jaccard$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2})$	Yes	[0,1]	[73]
KI	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2})$	Yes	[-1,1]	[51]
$Tanimoto$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2})$	Yes	[0,1]	[73]
$Consistency$	★			Yes	$O(\frac{k\cdot\ell}{2})$	Yes	[0,1]	[78]
CW	★			Yes	$O(\frac{k\cdot\ell}{2})$	Yes	[0,1]	[78]
CW_{rel}	★			Yes	$O(\frac{k\cdot\ell}{2})$	Yes	[0,1]	[78]
POG	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2})$	No	[0,1]	[95]
$nPOG$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2}) + O(c)$†	No	[-1,1]	[95]
$POGR$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2}) + O(c)$†	No	[0,1]	[95]
$nPOGR$	★			Yes	$O(\frac{k\cdot(\ell^2-\ell)}{2}) + O(c)$†	No	[-1,1]	[95]
SU	★			Yes	$O(\frac{n\cdot(k^2-k)}{2})$	Yes	[0,∞]	[91]

Table 2.1: The categories of the current stability measurements.

†The complexity of evaluating z and/or $\mathbb{E}$

Table2.1 shows clearly that most existing stability assessment methods calculate the stability for results returned in an index format. This emphasis the necessity

to more measurements that handle stability of features' weights and ranks to overcome the limitations of the existing ones. Also, there is no existing measure that deals with two or more different output schemes. It is also shown that measures from the second and the third category, i.e. by rank and by weight, are not able to handle different subset sizes. However, this property is common among all measurements belonging to the first category, i.e. by index. In addition, the running time complexity of these methods is quite similar yet the evaluation of the $\mathbb{E}(z)$ in the *POG* family and discretizing and normalizing in features' value in *SU* make them more expensive than others.

2.4 Summary

In this chapter, we reviewed the literature of feature selection for clustering and classification and discussed the different feature selection models. In addition, this chapter includes the first comprehensive survey about stability measurements.

Chapter 3

DILEMMA OF STABILITY EVALUATION

3.1 Introduction

Stability of a feature selection algorithm refers to the insensitivity of an algorithm to various data perturbations, which are usually caused by noise. The existence of noise is ubiquitous; therefore, a good feature selection algorithm should be robust to noise, and can return stable results, obtaining only relevant features. Given two sample sets generated by perturbing the original data, the stability of a feature selection algorithm can be measured by evaluating the similarity of the feature lists obtained by applying the algorithm on the two sample sets [46]. Fig. 3.1 shows a representative process for feature selection stability assessment that contains four key steps. (1) Given a data set X, one first generates l sample sets, $\mathcal{X} = \{X_1, ..., X_l\}$, either by random sampling or l-fold cross-validation. (2) A feature selection algorithm is applied to each sample set and selects features that result in l feature list, $\mathbf{F}' = \{\mathcal{F}'_1, \ldots, \mathcal{F}'_l\}$. (3) Various similarity measures are applied to evaluate the pairwised similarity between the obtained features lists, which results in a similarity matrix $\mathbf{S}$. (4) The final stability estimation is computed by averaging overall obtained pairwised similarity. Among the four steps, step (3) is the pivot component of the process. And currently, most existing work on feature selection stability assessment is devoted to designing effective measurements to evaluate the similarity of two given feature lists [51, 9, 73, 78].

Unlike most existing work, in this chapter, we focus on step (2) of the process, and study how to generate sample sets in a sensible way, so that the thereafter steps of the process can produce meaningful results. The motivation of this work can be

shown by the following example. Assume the original data is X. And based on X, we generate two data sets X_1 and X_2. Among the two newly generated data sets, X_1 is very similar to X, while X_2 is very different to X. Given a feature selection algorithm $f(\cdot)$, whose stability is unknown, we can apply the algorithm on X, X_1 and X_2, generating three feature lists $\mathcal{F}'$, $\mathcal{F}'_1$ and $\mathcal{F}'_2$. Let $S(\cdot)$ be a measurement assessing the similarity of two feature lists. Using $S(\cdot)$, we can generate two results: $S(\mathcal{F}', \mathcal{F}'_1)$, and $S(\mathcal{F}', \mathcal{F}'_2)$. Let us further assume that $S(\cdot)$ returns a small value in both cases, which means, neither $\mathcal{F}'_1$, nor $\mathcal{F}'_2$ is similar to $\mathcal{F}'$. Our question is that, should we draw the same conclusion no matter if we are given $S(\mathcal{F}', \mathcal{F}'_1)$ or $S(\mathcal{F}', \mathcal{F}'_2)$? The answer is obviously "no".

When $S(\mathcal{F}', \mathcal{F}'_1)$ is small, we know that $f(\cdot)$ is unstable. Since the difference between X and X_1 is small, in this case it is reasonable for us to require a stable feature selection algorithm to generate similar results. However, if we are only given $S(\mathcal{F}', \mathcal{F}'_2)$, it is hard for us to draw any conclusion. Since X_2 is very different to X, in this case, even a stale feature selection algorithm may generate very different feature list, due to the fact that the target concept contained by the two data sets may be completely different.

Most existing work implicitly assumes the sample sets generated in step (2) are of little difference. However, this assumption may not hold in many real-world applications. As mentioned above, given a data X, the l-folds, $X_1, ..., X_l$, are usually generated via either random sampling or l-folds cross-validation. However, we noticed that neither of the two methods can guarantee the l sample sets are of small difference. And when the differences among $X_1, ..., X_l$ are actually big, any conclusion drawn based on the output of the process loses its foundation and may no longer correct.

In this paper we urge the importance of considering data variance in the process of stability assessment for feature selection algorithms and study how to effectively measure and control the variance of the generated sample sets. To the best of our knowledge, this paper forms the first attempt that jointly considers both sample sets' similarity and feature list similarity in stability assessment for feature selection algorithms. The remaining content of the paper is organized as follows. In Section 2, we review related work. In Section 3, we develop effective methods to measure and control the variance of the samples sets generated for stability assessment. We discuss our extensive experiments in Section 4. Finally, draw conclusion in Section 5.

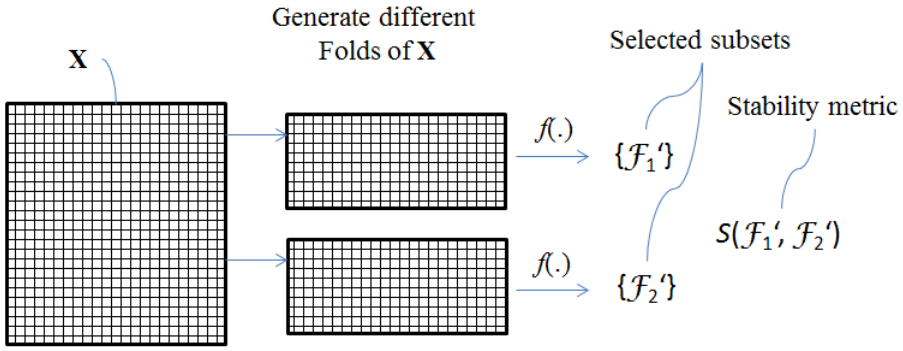


Figure 3.1: The process for assessing the stability of a feature selection algorithm.

3.2 Related Work

Most existing work for feature selection stability assessment focuses on designing effective measurements to evaluate the similarity of two given feature lists. In general, different similarity measurements fall into four categories: index based methods, weight based methods, rank based methods, and feature similarity based methods [47, 91]. In [47], three measurements are used to measure the similarity between two feature lists, which include Jaccard indx, Pearson's correlation, and Spearman's correlation. The formulations for the three measurements are defined as below:

$$S_J(\mathcal{F}'_i, \mathcal{F}'_j) = \frac{|\mathcal{F}'_i \cap \mathcal{F}'_j|}{|\mathcal{F}'_i \cup \mathcal{F}'_j|},$$

$$S_P(\mathcal{F}'_i, \mathcal{F}'_j) = \frac{\sum_k (w_{k,\mathcal{F}'_i} - \mu_{w,\mathcal{F}'_i})(w_{k,\mathcal{F}'_j} - \mu_{w,\mathcal{F}'_j})}{\sqrt{\sum_k (w_{k,\mathcal{F}'_i} - \mu_{w,\mathcal{F}'_i})^2 \sum_k (w_{k,\mathcal{F}'_j} - \mu_{w,\mathcal{F}'_j})^2}},$$

$$S_S(\mathcal{F}'_i, \mathcal{F}'_j) = 1 - 6 \sum k \frac{(r_{k,\mathcal{F}'_i} - r_{k,\mathcal{F}'_j})^2}{m(m^2 - 1)},$$

In the above equations, $\mathcal{F}'_i$ and $\mathcal{F}'_j$ are the two feature lists, S_J, S_P, and S_S are similarity measures derived from the Jaccard index, Pearson's correlation and Spearman correlation, respectively. $w_{k,\mathcal{F}'_i}$ denotes the feature weight of the k-th feature in feature list $\mathcal{F}'_i$, $\mu_{w,\mathcal{F}'_i}$ denotes the feature weight mean of the features in $\mathcal{F}'_i$, $r_{k,\mathcal{F}'_i}$ denotes the rank of the k-th feature in feature list $\mathcal{F}'_i$, and m is the total number of features. Among the three measurements, the Jaccard index deals with an index of selected features, and is an index based method. The other two deal with a feature's weight and rank, and are weight-based and rank-based methods, respectively. Jaccard index, aims to evaluate the amount of overlap between two set of feature indices, while Pearson's and Spearman's correlations aim to measure the consistency of weights or ranks of the features in the two lists. When l feature lists are given, the stability of a feature selection algorithm can be inferred from all feature list similarities via the following equation:

$$S_{(J,P,S)}(\mathbf{F}) = \frac{2}{l(l-1)} \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} S_{(J,P,S)}(\mathcal{F}'_i, \mathcal{F}'_j)$$

In [22], the authors propose average normal hamming distance (ANHD) to assess stability. Similar to Jaccard index, ANHD evaluates stability based on the

indices of the selected features. In [78], the authors proposed the consistency family measurements, which aim to evaluate the stability by considering the frequency of the features in the feature lists. Other index based similarity measurements, such as Dice-Sorensen's index and Tanimoto distance metric, have also been used to evaluate the similarity of two feature lists [100, 73]. Feature similarity based methods have also been developed by researchers. In [91, 28, 59], Symmetrical Uncertainty (SU) is used to evaluate the similarity of features in the two feature lists and has been reported to be effective for evaluating feature list similarity. Current approaches do not consider data variance when assessing feature selection stability, and this may cause serious problems, when data variance is big. In the next section, we develop effective methods to measure and control the variance of the sampled data. This ensures that sensible results can be generated from existing stability measurements.

3.3 The Dilemma of Stability Assessment

As we described in section 3.2, current stability measurements do not consider the influence of the variance on the results. In this paper, we first demonstrate the influence of the variation of the training datasets on the stability results by conducting an experiment using two extreme cases. The first scenario is the typical process of assessing stability that existing methods perform. We start by randomly sampling $l = 10$ different training datasets, $\mathcal{X}_1 = \{X_{11}, \ldots, X_{1l}\}$, from the original dataset X, where each subsample is 25% of the total number of samples m in X. We use 5 different datasets that vary in the number of samples m in dimensionality n; see Table 5.1. Next, we run the algorithm $f(\cdot)$ on $\mathcal{X}_1$ which will generate l different results, $\mathcal{F}'$. Finally, we assess stability using Jaccard index exactly as current methodology does. The second scenario is created by generating $l = 10$ training samples $\mathcal{X}_2$. In contrast to the first scenario, $X_{21}, \ldots, X_{2l}$ are exactly the same. In other words, we randomly

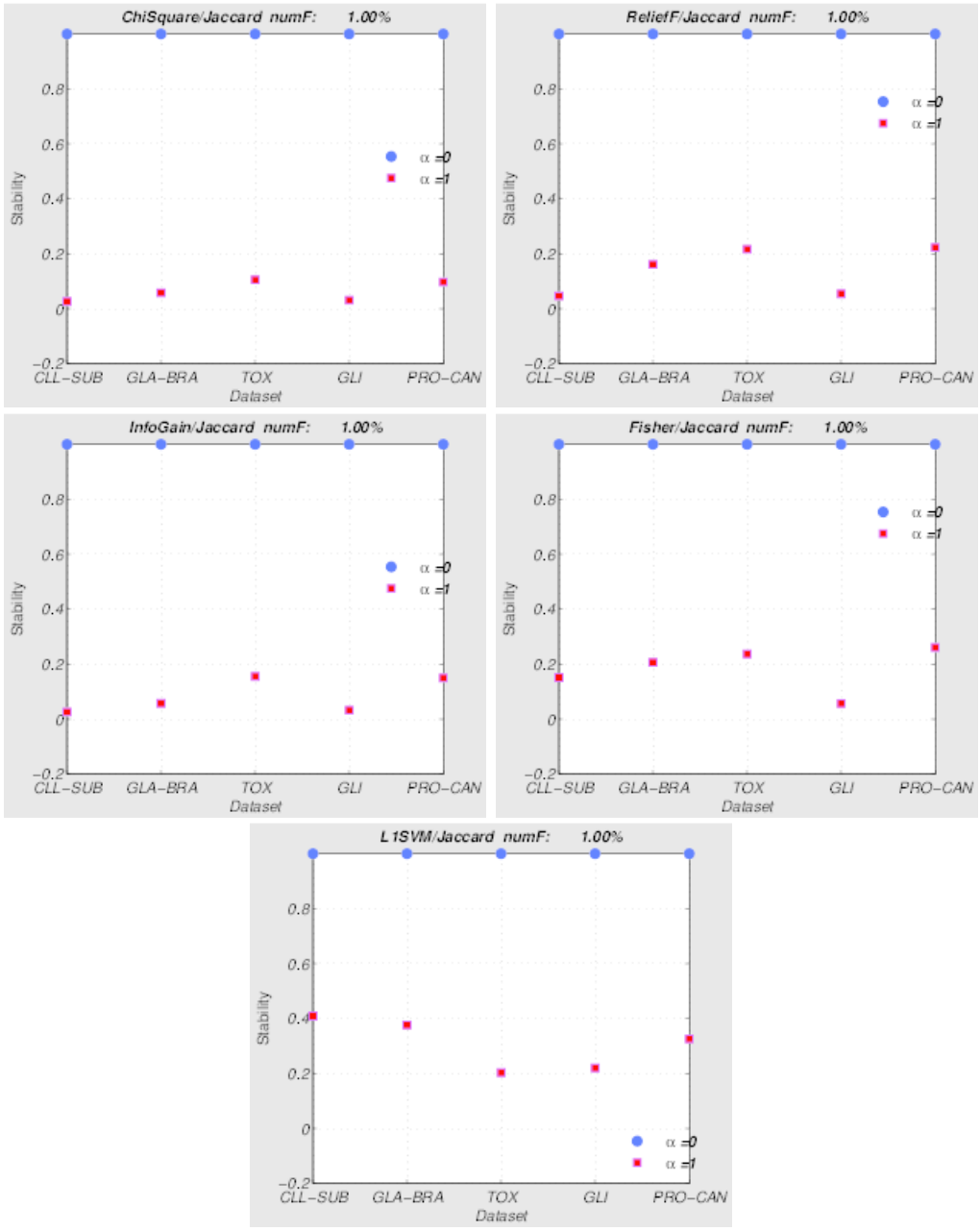


Figure 3.2: Stability $S_{(J)}(\mathcal{R})$ of the five methods across different datasets with two extreme cases in terms of the training samples's similarity where $\alpha = 1$ is the first scenario and $\alpha = 0$ is the second scenario.

sample 25% of m and duplicate this subsample l times. Then, we similarly run the same algorithm $f(\cdot)$ on $\mathcal{X}_2$ as we did in the first scenario. To summarize these two scenarios, we first assumed that the datasets suffer from very huge variation between the data samples, while in the second scenario, we assumed that there is no variance between the datasets at all. Five well-known feature selection algorithms are used in this experiment to demonstrate the consistency of the drawn conclusion. These algorithms are: ReliefF [86], ChiSquared [86], Information Gain [14], Fisher [21], and L1SVM [11]. An important question at this stage should be: which of these two scenarios' stability results should be the ground truth stability of the algorithm? In other words, there will be, for sure, two different stability results $S_{(J)}(\mathcal{R}_1)$ and $S_{(J)}(\mathcal{R}_2)$ corresponding to the first and second scenarios, respectively. Which should we consider to be the stability of the algorithm $f(\cdot)$?.

Figure 3.2 illustrates the stability of the above scenarios. $S_{(J)}(\mathcal{R}_1)$ is represented as $\alpha = 1$ and $S_{(J)}(\mathcal{R}_2)$ as $\alpha = 0$. As a result of the huge variance in the training samples in the first scenario, we obtained small stability in all algorithms and across different datasets with no exceptions. On the other hand, we got completely stable results in the second scenario, which means that all the generated results are always the same. Although this is an intuitive result, it provides strong evidence for the influence of the variance of the dataset on the stability results.

As a result, we can see that current measuring methods provide the assessment results that are heavily influenced by the sample variance. Now, we would like to establish the relationship between the stability measure and data variance.

The Impact of the Perturbation in $\mathcal{X}$ on the Stability

As we empirically prove in Section 3.3, the results of current stability assessment methods reflect the variance of the dataset, not the exact stability of the algorithm.

Table 3.1: Datasets statistics

Dataset Name	Number of Samples m	Dimensionality n
CLL-SUB	111	11340
GLA-BRA	180	49151
TOX	171	5748
GLI	85	22283
PRO-CAN	171	11302

Here, we go further in investigating the impact of the variance by controlling the difference between the training samples $\mathcal{X} = \{X_1, \ldots, X_l\}$. We apply different amounts of perturbation $\alpha = \{10, 20, 30, 40, 90\}$ % into $\mathcal{X}$. In order to do this, we randomly sample X_1 from the original dataset X. Then, we generate $X_2, \ldots, X_l$ by perturbing α% of the number of samples of X_1. So, there are at least $1 - \alpha$% out of the total number of data samples are the same in X_1 through X_l. Figure 3.3 shows the stability of each algorithm across the datasets, and it clearly tends to be higher with less amount of perturbation. In this experiment, we selected 1% of the original number of features in order to get a reasonable number of relevant features.

These empirical results suggest the dependency of the stability results on the variation of the training samples. In other words, it shows the relation between the repeatability, i.e. the stability $S_{(J)}(\mathcal{R})$, of the results $\mathcal{R}$ and the similarity between the training samples $\mathbf{S}_{\mathcal{X}}$. We found that $S_{(J)}(\mathcal{R})$ is higher when α is smaller. Hence, by taking the similarity of the data samples into account, we can mitigate the effect of the data samples in the assessment of stability. This proposed methodology helps to justify and to understand the stability results.

3.4 Ranking vs. Classification

After we demonstrate the influence of dataset variance on the stability results in Section 3.3, we find that existing stability measures cannot assess the exact stability of a given algorithm without considering factors that influence the result. In fact, current methods do not assess stability, but they can only rank the algorithms according to the repeatability of the results. For illustration, Figure 3.4(a) shows the stability using the Jaccard index for ChiSquare, ReliefF, Information Gain, Fisher, and L1SVM. These results can only rank the algorithms according to their stability. For example, we can infer from Figure 3.4(a) that L1SVM is the most stable, and Fisher is the second, and so on. However, we cannot tell whether they are, in fact, stable or not since we have no clue about the variation or the similarity between training samples. For instance, the training samples might be very similar to each other, thus, we cannot classify the algorithm as stable, owing to the fact that the cause of stability could be the the small variation in the dataset. Furthermore, the training samples might be very dissimilar. Thus, we cannot classify the algorithm to be instable as well. Similar to the first case, the instability might be caused by the huge variation in the dataset, and as a sequence, algorithms should not be expected to generate similar results. This example shows us the necessity of evaluating the similarity between training samples, in order to be able to classify the algorithms as stable or instable.

In order to infer whether a given algorithm is stable, we need to consider an important factor, which is the average pairwise similarity between the training samples. Thus, we need first to find an appropriate method to evaluate this similarity. Assume we are given $\mathcal{X}$ contains two folds X_1 and X_2, and let X_{1c_r} and X_{2c_k} to be the r^{th} and the k^{th} data point that belongs to the class c in X_1 and X_2 respectively. Then

the average distance $Sim(X_{1c}, X_{2c})$ between the samples in X_1 and X_2 that belong to class c is given by:

$$Sim(X_{1c}, X_{2c}) = \frac{1}{|X_{1c}||X_{2c}|} \sum_{r=1}^{|X_{1c}|} \sum_{k=1}^{|X_{2c}|} \|X_{1c_r} - X_{2c_k}\|,$$

and the average pairwise similarity, $S_{\mathcal{X}}$, between *l-folds* in $\mathcal{X}$ is given by:

$$S_{\mathcal{X}} = \frac{2}{l(l-1)} \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} \frac{1}{|C_{i,j}|} \sum_{c \in C_{i,j}} Sim(X_{ic}, X_{jc}),$$

where $C_{i,j}$ is the intersection between the classes in X_i and X_j and $|C_{i,j}|$ is the cardinality of $C_{i,j}$. By evaluating the similarity of the training samples and the similarity of the results, we can tell whether a given algorithm is stable or not. We propose a novel approach that could simply compare the similarity of the training samples $S_{\mathcal{X}}$ against the similarity of the results $S_{(J)}(\mathcal{R})$. Then we derive the result based on this comparison. For more illustration, let X_1 and X_2 be the training samples, where $l = 2$ in this case. And let $\mathcal{F}_1'$ and $\mathcal{F}_2'$ be the generated result by the algorithm $f(\cdot)$. $f(\cdot)$ is said to be stable if $S_{(J)}(\mathcal{R}) \geq S_{\mathcal{X}}$ and unstable otherwise. As a result for this approach, the similarity values of the training samples, $S_{\mathcal{X}}$, are connected by the red line in Figure 3.4(b). With these threshold values, we can determine if an algorithm is stable or not. In other words, if $S_{(J)}(\mathcal{R})$ exceeds or equals $S_{\mathcal{X}}$, then that particular algorithm is said to be stable.

As a result, Table 3.2 illustrates which algorithm is stable or not with which dataset. According to the values ($S_{\mathcal{X}}$) in the last row in Table 3.2, we check if $S_{(J)}(\mathcal{R}) \geq S_{\mathcal{X}}$ and determine whether an algorithm is stable or not. The stability values that are in **Bold** are the ones exceeding the threshold ($S_{\mathcal{X}}$) and thus are classified as stable. Therefore, L1SVM is always stable. Similarly, Fisher is almost

always stable except for GLI. In addition, ReliefF is considered stable with GLA-BRA, TOX, and PRO-CAN. While ChiSquare is stable with TOX and PRO-CAN only. Finally, Information Gain is always unstable except with TOX dataset. In short, we empirically show that existing methods can only rank algorithms, but using the average pairwise similarity of training samples enables us to distinguish stable and instable algorithms.

	CLL-SUB	GLA-BRA	TOX	GLI	PRO-CAN
ChiSquare	0.1145	0.0638	**0.0993**	0.0284	**0.0960**
ReliefF	0.0559	**0.1535**	**0.1240**	0.0853	**0.1362**
InfoGain	0.0186	0.0491	**0.1055**	0.0401	0.0636
Fisher	**0.1468**	**0.1745**	**0.1970**	0.0861	**0.2129**
L1SVM	**0.3915**	**0.3342**	**0.2141**	**0.2373**	**0.3435**
The threshold $S_{\mathcal{X}}$	*0.1325*	*0.0648*	*0.0380*	*0.1041*	*0.0902*

Table 3.2: The stability of each algorithm with each dataset compared against the threshold with the training sample similarities $S_{\mathcal{X}}$ in *Italic*. Algorithms' stability in **Boldface** are considered stable since they exceeded the threshold.

3.5 Discussion

The assessment of the stability of the feature selection algorithms happens to be influenced by the dataset variation. The literature also suggests that there are other factors that may impact the stability such as sample size [91] and the number of selected features k [47]. These factors can be investigated independently. For example, it was shown in [47] that the stability measures can increase proportionally with the number of selected features. However, this kind of influence can be mitigated by a good estimation of k relevant features. As Kalousis et al. found, the increase of the stability that associated with the increase of k is mainly due to a large number of selected features that are irrelevant to the learning problem. In other words, choosing features with weight $w = 0$ to evaluate the stability is going to give higher stability since the features are added to the selected list by their sequential order. As a result

of our experiment, we notice that the number of features that have $w \geq 0$ happened to be around 1% of the dimensionality. Thus, we selected 1% of features to alleviate the influence k has on the results.

Future work is to develop a feature selection method that improves selection robustness by reducing the data variance across different distributed datasets.

3.6 Summary

In this chapter, we demonstrated the dilemma of evaluating the stability of feature selection algorithms. We empirically prove that current stability assessment methodology can be heavily influenced by the variance of data samples. Therefore, the existing methods can only compare between the feature selection algorithms and rank them using stability values and cannot tell if an algorithm is stable or not, in presence of data variance. We proposed to take the training samples' similarity into account when assessing the stability. Thus, we could easily determine that the algorithm is stable or not by comparing the stability of the results with the similarity of the dataset. The stability assessment results given by our method show that some algorithms that were considered stable are actually not stable. To the best of our knowledge, this is the first work that considers the influence of the dataset variation in assessing the stability of feature selection algorithms and can provide an objective stability assessment in critical data mining and machine learning applications.

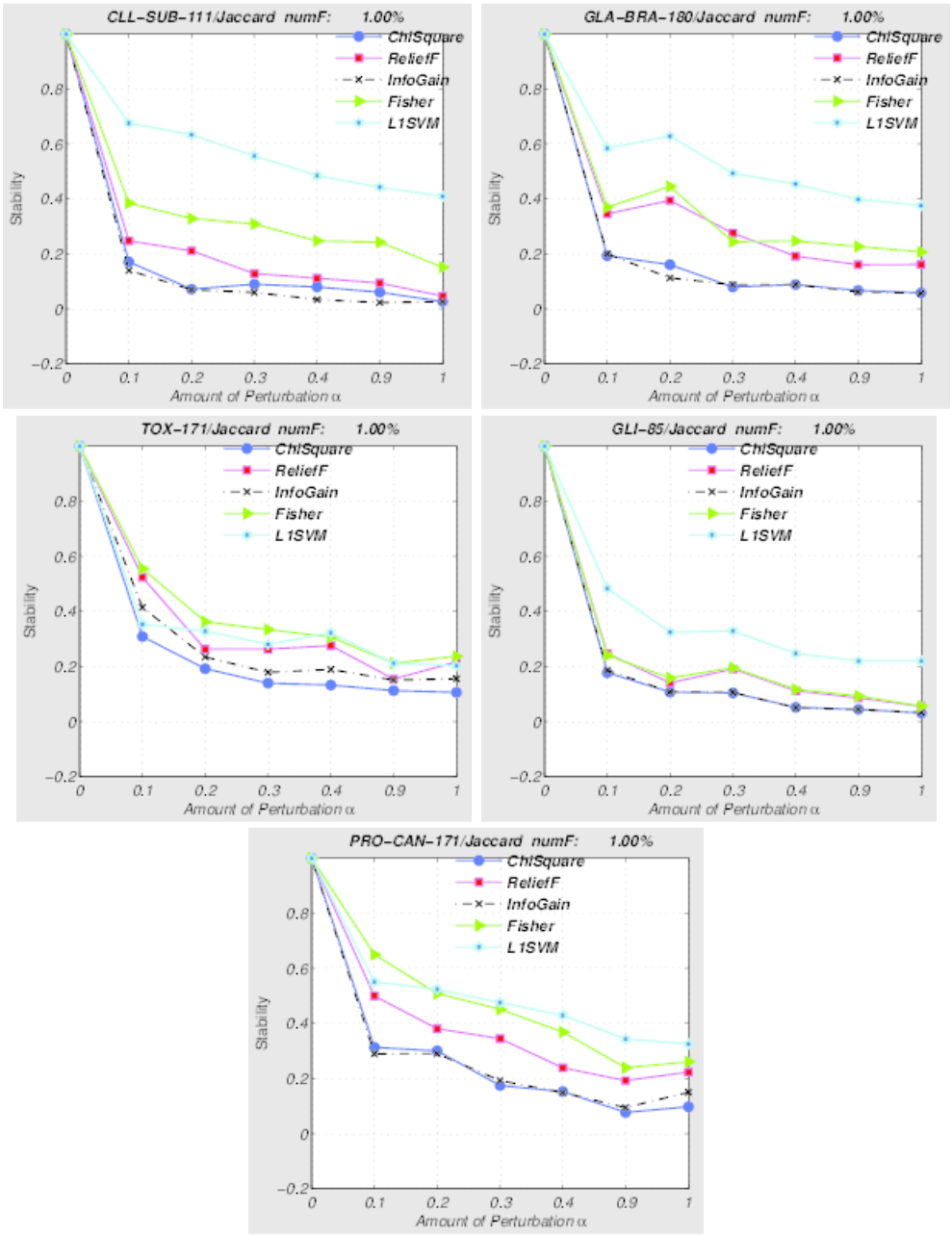
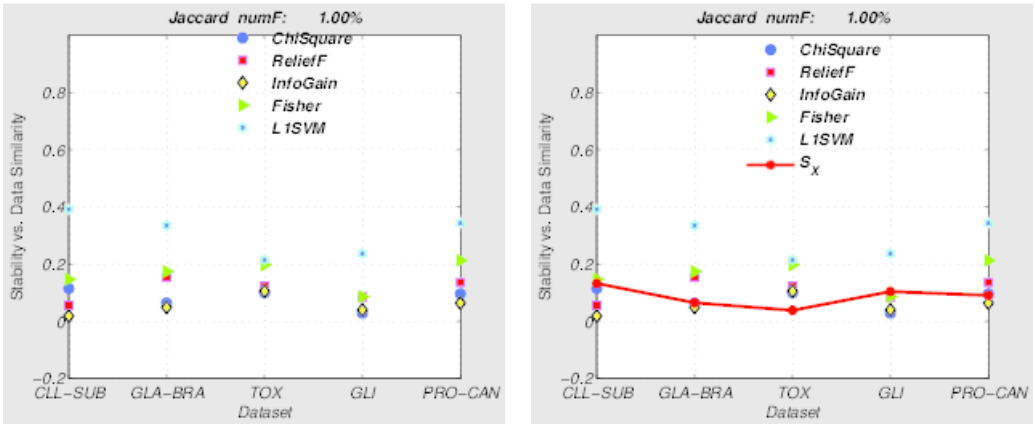


Figure 3.3: The stability using different amount of perturbation α. It shows the decreasing trend of the stability as α increases.



(a) (b)

Figure 3.4: (a) The stability $S_{(J)}(\mathcal{R})$ of the algorithms using existing approach. (b) The pairwise similarity between training samples $S_{\mathcal{X}}$, the red line, compared with $S_{(J)}(\mathcal{R})$, the marks, where $S_{\mathcal{X}}$ is the threshold that classifies the algorithm as either stable or not .

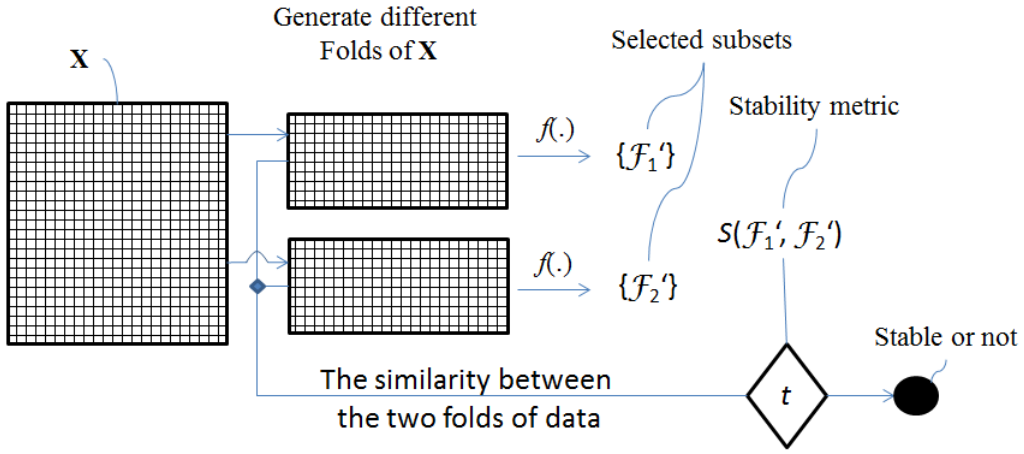


Figure 3.5: The proposed selection stability approach.

Chapter 4

THE CAUSE OF SELECTION INSTABILITY

Most of the existing work studies the stability from the algorithm prospective. They mostly propose new feature selection methods that claimed to be stable. We fundamentally disagree with this approach, as we believe that the underlying characteristics of the dataset have a significant impact on the stability. If this is true, we should, instead, propose a framework to handle these factors and has the ability to utilize any current feature selection algorithm.

In this work, we will discuss several factors that may effect selection stability. We are going to empirically demonstrate the effect of the dimensionality, the absolute sample size, and the variation of the underlying distribution of the dataset on stability. In addition, we study the effect of the number of selected features on the stability. Finally, we discuss the stability behavior of several well-known feature selection algorithms with a variety of datasets.

4.1 Literature Review

The stability of feature selection algorithms is the sensitivity of the selection to variation in the data set [47, 91]. The data variance is usually caused by noise. The existence of noise is ubiquitous; therefore, a good feature selection algorithm should be sufficiently robust to handle noise and can return stable results that contain only relevant features. Stability has gained increasing attention, becoming a hot topic in the feature selection. Furthermore, stability is an important criterion to evaluate the goodness of a feature selection method. One motivation behind this increasing attention to stability is the fact that in domains like bioinformatics, the domain experts would like to see the same or at least similar set of genes, i.e. features to

be selected, each time they obtain new samples in the presence of a small amount of perturbation. Otherwise, they will not trust the algorithm when they get different sets of features while the datasets are drown for the same problem.

Several stability measures have been proposed to evaluate the similarity among the selected feature subsets [47, 51, 91, 22]. These measures can be broadly categorized into three categories based on their inputs. The first category contains those methods that take the indices of the selected features as an input. A *Jaccard Index* is a representative stability measure in this category [47]. It assesses stability by evaluating the amount of overlap between the results that contains the selected features' indices. Besides *Jaccard Index, KI [51], Dice Index [91], and ANHD [22]* are other proposed measures. For the measures in the second category, the input is the rank of the selected features. These measurers assess stability by evaluating the similarity between two sets of features that are ranked based on their relevance. *Spearman's Rank Correlation Coefficient* [47] is a representative measure for stability by rank that evaluates the correlations between the ranked lists. The third category contains measures that take features' weight as an input. *Pearson's Correlation Coefficient* [47], which assesses the correlation between the weighted results, is a good example.

Given differen results $\mathcal{R} = \{R_1, R_2, \cdots, R_l\}$ corresponding to l runs of algorithm $f(\cdot)$ on l different folds of the data set $\mathbf{X}$, its stability can be assessed simply by assessing the amount of overlap between the sets in $\mathcal{R}$. The evaluation of the stability of an algorithm can be summarized as the following four key steps: (1) generating l different folds of $\mathbf{X}$ either by random sampling or cross-validation. Then, (2) a feature selection algorithm is applied to each fold which (3) produces the l different results shown in $\mathcal{R}$. Finally, (4) an average pairwise similarity is calculated

from the selection result to obtain the stability using a suitable stability measure. Most of the existing work has used *Jaccard Index* to evaluate stability [47]. In this work, we focus on this representative measure to conduct our experimental study.

The *Jaccard Index* was introduced in [47] to evaluate the stability for subsets of results that contain selected features' indices by evaluating the amount of overlap between the subsets. Equation(4.2) shows a *Jaccard Index* for two selected subsets.

$$S_J(R_i, R_j) = \frac{|R_i \cap R_j|}{|R_i \cup R_j|}. \tag{4.1}$$

Now, we can evaluate its stability by evaluating S_J in a pairwise manner:

$$\mathcal{S}_J(\mathcal{R}) = \frac{2}{l(l-1)} \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} S_J(R_i, R_j) \tag{4.2}$$

The *Jaccard Index* $\mathcal{S}_J$ returns a value in the interval of [0,1] where 0 means the feature selection results are not stable and 1 means the results are identical, hence very stable.

4.2 Problem Statement

Let $\mathbf{X} \in \mathbb{R}^{n \times m}$ be a dataset. Also, let $\mathbf{Z} \in \mathbb{R}^{n_\mathbf{Z} \times m_\mathbf{Z}}$ be another dataset with different characteristics and other hypotheses. Without lose of generality, assume the stability of algorithm $f(\cdot)$ on $\mathbf{X}$ and $\mathbf{Z}$ to be given by $\mathcal{S}_\mathbf{X}$ and $\mathcal{S}_\mathbf{Z}$ respectively.

Do $\mathcal{S}_\mathbf{X}$ and $\mathcal{S}_\mathbf{Z}$ have any correlation or any relationship? Most likely, no! It is found that given different datasets with different characteristics, the stability of each dataset is independent of the stability of another even though the algorithm is the same. In this chapter, we are going to investigate the underlying characteristics that influence the selection stability.

4.3 Motivation

In order to stabilize the selection process, we need to understand the causes of instability first. It is reasonable for one algorithm to have different stability results as we explained above. Therefore, we believe that stability is not totally algorithm-dependent. In contrast, it is mostly data-dependent. Thus, for an in-depth understanding of the stability issue, we should investigate the factors that affect it. Consequently, we will be able to cure by reducing the impact of these factors.

4.4 Our Contribution

As we mentioned above, the recent work in stability focuses on step (4) in the process: how to estimate the stability of an algorithm. Although this is an important step in order to give a reasonable estimation of how robust an algorithm is, the recent work does not answer some important questions. In this work, we are going to investigate the following questions: (1) what are the factors that may effect the stability of a feature selection algorithm? (2) Are the factors algorithm-related or dataset-related? In addition, we will investigate, given a dataset $\mathbf{X}$, (3) what is the most suitable feature selection algorithm to select robust, highly predictive and relevant features? To the best of our knowledge, these important questions have not been sufficiently addressed in the literature. In this paper, we are going to present an empirical study of these questions for better understanding of the stability.

The Effect of The Number of Selected Features k

In most real world datasets, e.g. microarrays, the number of truly relevant features k_{opt} is usually small compared with the dimensionality m. In general, it is often difficult to know what k_{opt} exactly is in advance, and the problem of identifying the

value of k_{opt} is not trivial. In practice, there are three possibilities when choosing k. (1) The first scenario is choosing k as k_{opt}. This case is rare in real world problems. Even if k_{opt} can be known beforehand, it cannot be guaranteed that the optimal feature subset can always be selected due to the noise and outliers in the data sets and due to the fact that finding the optimal subset is *NP-hard* problem. Thus, the selection is not guaranteed to be stable. The second scenario (2) is choosing $k < k_{opt}$. Even with an effective feature selection method, choosing a small k will make the selected features vary in the presence of small variations in the data set. As a result, the selection is not guaranteed to be robust. The third scenario (3) selects more features than the number of relevant features, that is, $k > k_{opt}$. Similar to the previous two, this scenario does not guarantee selection stability even in case of selecting all relevant features.

To illustrate the three scenarios, we assume that, for a given data set, features f_1 to f_{10} are relevant features while f_{11} to f_{100} are the irrelevant ones. First, we assume $k = 10$. When we run algorithm $f(\cdot)$ on l different folds, we may get different weights for the features each time due to variation across the folds, which may lead to slightly different subset of selected features. In the second scenario, we assume $k = 5$. Similarly, a small variation in the data set may result in different relevance weights for the relevant features, which in turn leads to selecting slightly different features at each fold. The last scenario, we assume $k = 15$. In this case, even if we select all the 10 relevant features, the other 5 features will be irrelevant and the order of these irrelevant features may vary significantly at each fold, which decreases stability. Evidently, the analysis is tightly related to the characteristics of the data set. We are going to recall this when we talk about the characteristics of the datasets later in this work.

Another observation is that the larger the value for k, the higher the stability will be. This is due to several reasons. First, with large k, the chance of selecting all relevant features becomes high. In addition, the probability that two selected feature subsects intersect with each other by chance will become higher too. Assume that R_i and R_j are the i^{th} and the j^{th} selected feature sets, respectively. The prior probability of selecting any feature f is given by:

$$p(f) = \frac{1}{m}$$

Thus, the probability of selecting k features in any R is:

$$p(R_i) = p(R_j) = \frac{1}{\begin{pmatrix} m \\ k \end{pmatrix}}$$

The probability of selecting at least one common feature in R_i and R_j is:

$$p(|R_i \cap R_j| \geqslant 1) = \frac{\begin{pmatrix} m \\ m-k \end{pmatrix}}{\begin{pmatrix} m \\ k \end{pmatrix}} \tag{4.3}$$

It is obvious that the larger k is, the larger $p(|R_i \cap R_j| \geqslant 1)$ will be.

The Effect of The Dataset Characteristics

Most existing works study the stability from an algorithm prospective while ignoring the effects the dataset exerts on it. Intuitively, following this approach only will not effectively solve the challenging questions on the feature selection stability such as: what are the factors that impact stability and how may we consider these factors in the selection process in order to improve selection stability. Also, observing the behavior of different algorithms on different datasets may help to answer the questions

regarding how to choose the most appropriate feature selection algorithm for a given dataset. We believe, as we will empirically demonstrate later, that the underlying characteristics of the dataset $\mathbf{X}$ can have a significant impact on the stability. Thus, studying the stability from the dataset perspective is necessary.

Let $\mathbf{X}_1$ and $\mathbf{X}_2$ denote two different datasets with the number of instances, n_1 and n_2, and the number of features, m_1 and m_2, respectively. Also, let k_1 and k_2 denote the number of selected features for the two datasets. We apply the algorithm, $f(\cdot)$ to each of the l-folds of $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively. Then, we assess the stability $\mathcal{S}_J(\mathcal{R}_1)$ and $\mathcal{S}_J(\mathcal{R}_2)$. Do we expect to have the same stability, although we use the same algorithm $f(\cdot)$? Intuitively, the answer is that the stability is not always the same. The important questions now is why $f(\cdot)$ does not behave similarly, in terms of stability, on these two datasets? There must be certain characteristics in $\mathbf{X}_1$ and $\mathbf{X}_2$ that may be affecting the stability. In the following, we are going to analyze some of these factors and discuss their potential influences on stability.

The Effect of Dimensionality m

The larger the dimensionality m, the lower the probability $p(R_i = R_j)$, where $p(R_i = R_j)$ is the prior probability of selecting the same set of features in R_i and R_j by chance. Furthermore, the number of combinations of k features chosen from m is known as m choose k, $\begin{pmatrix} m \\ k \end{pmatrix} = \frac{m!}{k!(m-k)!}$. Hence, the prior probability of choosing the same set of features twice is:

$$p(R_i = R_j) = \frac{1}{\begin{pmatrix} m \\ k \end{pmatrix}} = \frac{k!(m-k)!}{m!}, \tag{4.4}$$

According to Equation(4.4), when k is close to m, $p(R_i, R_j)$ approaches 1, and equals 1 when $k = m$. This, also, gives the same conclusion drawn from Equation(4.3).

The Effect of Sample Size n

The effect of sample size on the learning process has been well studied in the literature. It is proven that when a limited number of training samples is available, the potential for overtraining is high, and the learning performance may not generalize to a larger populations reducing learning quality [85, 41]. Unsurprisingly, this fact holds for selection stability too, as we will empirically prove in the experiment.

The Effect of the Underlying Data Distribution

In addition to the dimensionality and sample size, the underlying distribution of the data has a significant impact on stability. For example, when the l^{th} fold is significantly different from other folds, this may lead to selecting different subsets of features, which means lowering stability. This is related to the theory of important sampling, which suggests an increasing in the number of samples from regions that contribute more to better performance and decrease the number of samples from less attractive regions [32]. Although this approach will lead to reducing data variance, it may cause the loss of important information in the ignored samples. Therefore, [32] suggests, alternatively, to assign less weight to these undesired samples and higher weights to samples from attractive regions.

4.5 Experiment

In this experimental study, we aim to investigate several important questions regarding the impact of a dataset's characteristics and the number of selected features k to a feature selection algorithm. Also, we aim to highlight the importance of choosing

a feature selection method that can work well on a given dataset. In this section, we are going to empirically study these issues to gain better understanding of stability. In order to achieve this goal, we conduct the following experiments from 23 different datasets, five feature selection algorithms and a Jaccard Index to assess the stability.

Datasets

In this experiment, we use the 32 datasets listed in Table5.1, which are publicly available online. We divide these datasets to three different groups. The *first group* contains 5 microarrays, which can be downloaded from *ASU Feature Selection Repository*[1]. The datasets in this group share similar characteristics in terms of the number of features m and the number of samples n. They have a large number of features ranging from 5748 to 49151. In addition, the number of samples is also relatively large for a microarray, ranging from 85 to 180 samples. The *second group* contains 14 different datasets. Similar to the *first group*, the *second group* has a large number of features, yet, a small number of samples $n \leq 40$. Finally, the *third group* has generally a large number of samples and small dimensionality.

Feature Selection Algorithms

We chose five well-known feature selection algorithms to conduct this experiment. These algorithms are: ReliefF [86], ChiSquared [86], Information Gain [14], Fisher [21], and L1SVM [11]. All algorithms except L1SVM are publicly available at *ASU Feature Selection Repository*. All algorithms, except L1SVM, are filter-based, so they do not involve any classifier during the selection process. ReliefF and L1SVM both attempt to maximize the margin where the former is related to hypothesis margin

[1]http://featureselection.asu.edu/

maximization, and the latter is sample margin maximization [70]. Yet, Fisher score assigns higher scores to the features that are able to better discriminate the samples from different classes. ChiSquare assesses whether a particular feature is independent of the class label. Similar to ChiSquare, Information Gain assesses the independence between a feature and the class label by the difference between the entropy of the feature and the conditional entropy given the class label.

Experiment Methodology

In order to demonstrate the effect of different factors, mentioned above, we run the algorithms on the datasets using 10 cross-validation, that is, $l = 10$. At each run, each algorithm assigns weights to all the features, then, we select the features with the higher weights to be the feature selection results at the i^{th} fold, denoted by R_i. All Rs from the l runs will form the set $\mathcal{R}$. Finally, we calculate the average pairwise *Jaccard Index* based on Equation(4.2) for $\mathcal{R}$.

Results

We empirically evaluate the effect of the underlying characteristics of the dataset on the stability of the feature selection algorithm. We use 32 datasets that are grouped to three different sets. The datasets in each group share common characteristics in terms of dimensionality m and sample size n. For a clearer illustration, we will divide the results based on the observations.

The Effect of k

Figure 4.1 shows the stability of six datasets with different values of k. If the number of features in the datasets is less than 1000, we evaluate the stability for $k = \{1, \ldots, m\}$, otherwise, $k = \{1, \ldots, 1000\}$. Here, we chose only 6 representative

datasets because the rest of the datasets behave similarly almost all the time. Figure 4.1 shows that the larger the value of k is, the higher the stability will be. Furthermore, the stability on the microarrays, *CLL-SUB, GLI, and PRO-CAN*, will increase with the increase of k until reaching the maximum, where $k = m$. Nevertheless, this increase of stability does not necessarily reflect the real robustness of the selection due to the small number of the relevant feature k_{rel} compared with k.

Another interesting observation is noticed when a small number of features are selected, as we can obtain high stability during the selection of the first a few numbers of features, especially for the microarrays when m is very large. This might indicate that these features are significantly relevant to the problem. Accordingly, the stability here could be taken as a criterion to select a suitable k. Figure 4.3 shows *CLL-SUB, GLI, and PRO-CAN* with $k = \{1, \ldots, 100\}$ where the peaks of the stability, indicated by the black arrows mean that the features from 1 to the peak are more frequently selected in all folds than in others. These three plots show different behaviors. For example, *GLI* shows only one peak, which makes it easy to pick an appropriate k according to this criterion, while *CLL-SUB* has two and *PRO-CAN* has several peaks that make it more complicated to pick the best one. In this case, more constrains are needed. For example, k should be greater than a certain minimum value. The stability plot of *CLL-SUB* in Figure 4.3 shows the first peak at $k = 10$ with stability around 0.73. This means that the algorithms running at each fold agree with each other on a subset of features 73% of the time, which makes these features more frequent to occur at the top of the list. For features beyond the tenth, stability starts to degrade rapidly, which indicates the algorithms running on each fold become less agreeable.

The Effect of Sample Size n

First of all, it is important to mention that this study focuses on the absolute sample size. Figure 4.4 shows the stability of all 32 datasets. Figure 4.4(a) shows the stability of the *first group* where the number of samples is relatively large for microarrays. The red denotes average stability. Figure 4.4(b), on the other hand, corresponds to the *second group* of the datasets. Similarly, Figure 4.4(c) shows the stability of the *third group*, where the number of samples n is very large. From these figures, we can observe the significant difference between the average stability of each group. We can conclude from this observation that the sample size correlates positively with selection stability.

The Effect of the Dimensionality m

In contrast to the sample size, large dimensionality adversely affects stability. Figure 4.4(a)(b)(c) clearly shows the impact of the huge dimensionality of the microarrays in (a) and (b) compared with that of (c). However, Figure 4.4 does not show which factor has more impact on the stability: the sample size or the dimensionality. To answer this question, we run another experiment using a *TOX* dataset because it has large m and n. We first run each algorithm on the whole dataset to obtain the weight assigned to each feature. By sorting the weight values, we observe a long tail which corresponds to very low weights or simply zero weights, as shown in Figure4.6. We found that the first 100 features have the largest weights; hence, we consider them as the relevant features. Yet, the features from 1,465 to the end are assigned weights equal to zero, so they are considered as the irrelevant features here. The 1,365 remaining features in between that are moderately relevant are not used in the following experiment. Then, we partition the new version of *TOX* dataset

into different partitions, varying both the number of samples and features. The first partition contains 11 samples and 1,100 features. Where the relevant features are always included and 1,000 features are randomly selected from the irrelevant features. Then, we run each of the five feature selection algorithms on the first partition and assess the stability. Next, we increase the number of samples by 10 for the second partition and run the algorithms and evaluate the stability again. When the total number of samples is approached, we add 100 more irrelevant features and start again with 11 samples and so on until we reach the total number of features and samples. Figure 4.2 shows stability as a surface where each sub-figure represents the stability of one algorithm. It is observed that the stability becomes weakened, or lower, when the number of samples is smaller and the number of features is larger. This experiment also shows that the the number of samples impact is more significant than the impact of the number of features. Furthermore, the impact of the number of features vanishes when the sample size is significantly large. As the figure shows, the difference in stability between the full set of features and the partition with 1,100 features is at maximum with the smaller sample size, decreasing as larger sample size increases.

The Effect of the Underlying Distribution

To demonstrate the effect of the underlying distribution of the dataset, we perform a controlled experiment as follows. We sample 11 folds of each dataset. Among the 11 folds sampled from the original training set, ten are allowed to overlap each other, whereas the 11^{th} fold (called last fold or D_{11}) has no overlapping with any of the ten folds. In this case, we can expect that the difference among the underlying distribution represented by each of the ten folds (denoted as $D_1, D_2, , D_{10}$) is smaller, while the difference between the underlying distribution represented by the 11^{th} fold

(denoted as D_{11}) and any of $D_1, D_2, , D_{10}$is larger. By examining the results in Figure4.7, it is found that the feature selection results on the first ten folds agree more with each other, but the result on the 11^{th} fold is quite different. This shows the affect of variation of underlying distribution to feature selection. We show this only with CLL-SUB dataset due to page constraints. All other datasets behave similarly. Thus, we believe that the underlying distribution is an important factor that may effect the stability; consequently, sampling techniques may improve it.

Algorithms Behaviors

It is not easy to predict the performance of an algorithm on a given dataset. In this section we attempt to observe the behavior of the five feature selection methods on the datasets. Figure4.4 shows how the performance of the algorithms varies in terms of stability. Each algorithm behaves differently on different datasets, and with respect to the datasets' groups. In addition, the algorithms behave differently on a single dataset. For example, we find that with microarray datasets, i.e. *the first and the second groups*, where the dimensionality is huge, the stability difference between algorithms is much larger than *the third group* of datasets. We find that the algorithms give almost equally good stability results with datasets in *the third group*. However, the difference is higher with *the first and second groups*. In addition, we find that *L1SVM* selects stable results compared with other methods when dimensionality is high, *L1SVM* beats all other methods in having a high dimensionality and a small sample size, *in the second group* of datasets. However, *L1SVM* is a clear loser with *the third group*. It is obvious that *ChiSquare* and *Information Gain* are not the preferred methods when the dimensionality is high but they become more competitive in *the third group* which has no clear winner. Finally, *Fisher* and *ReliefF* are almost equally good with the three groups of the dataset. Based on these results, we prefer

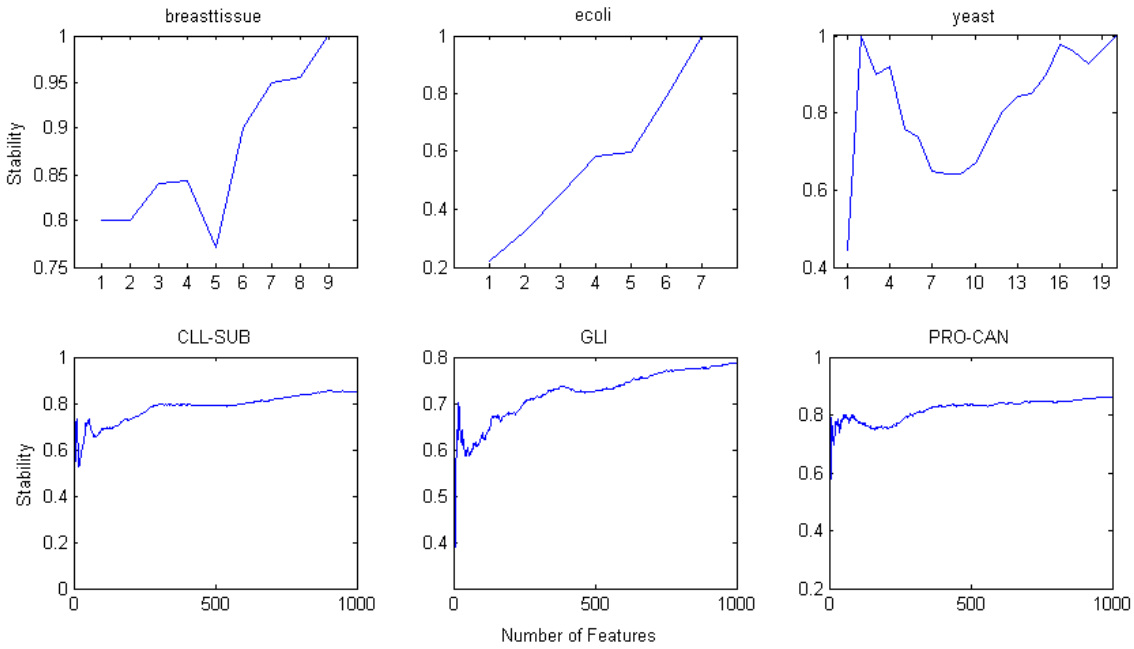


Figure 4.1: The effect of large k.

L1SVM in case of higher dimensionalities and smaller numbers of samples, although, it becomes less preferred in other cases. Still yet, we prefer either *Fisher* or *ReliefF* in other cases, or in case of a lack of information about the datasets.

4.6 Conclusion and Future Work

The researcher in the realm of feature selection should pay more attention to the underlying characteristics of the dataset for better understanding of selection stability. As the distribution of the dataset is not always known, we believe more attention to sampling techniques should be strengthened . In addition, choosing the appropriate method to perform selection on a given dataset is an interesting problem where algorithms do not perform equally well on different datasets. An algorithm that perform the best on a dataset may not perform good on another one.

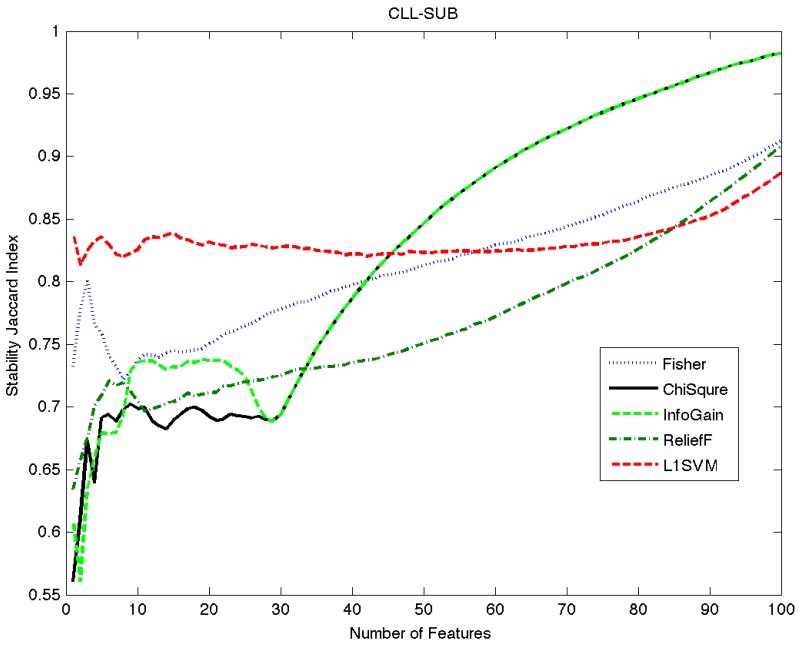


Figure 4.2: The effect of large k.

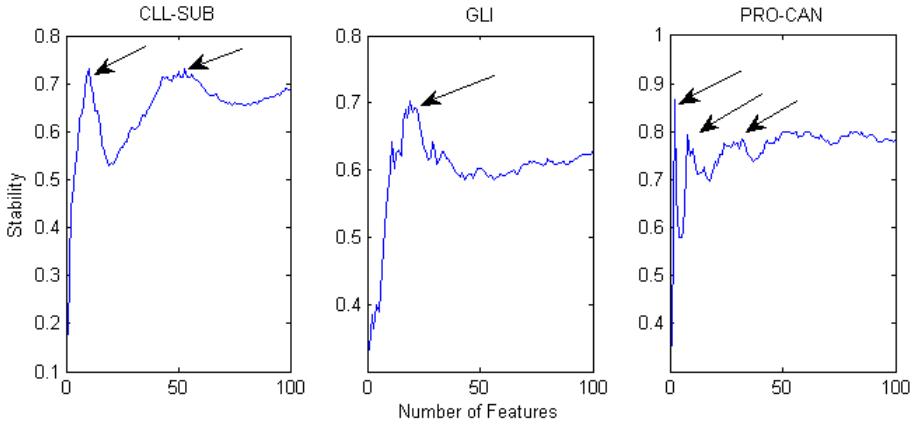


Figure 4.3: Demonstration of the stability as a potential criterion to choose the appropriate k.

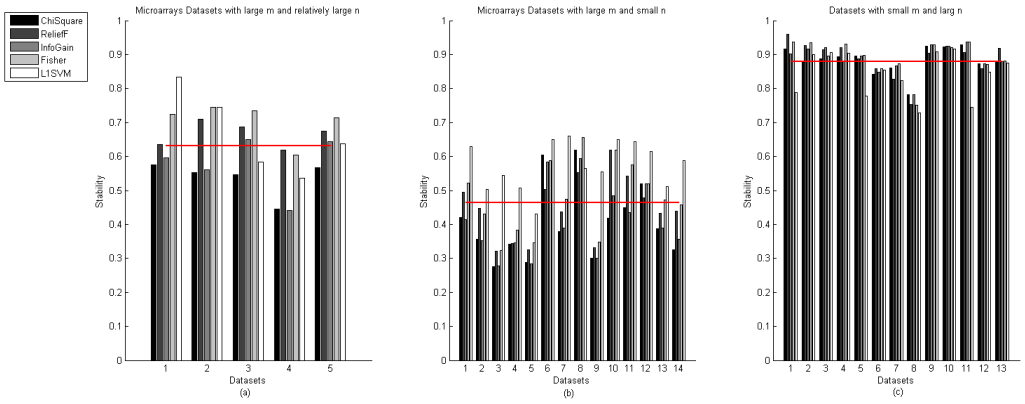


Figure 4.4: Jaccard Index stability for all algorithms and all datasets. (a) shows the stability on the *first group* of the datasets. (b) shows the stability on the *second group* of datasets. And (c) shows the stability on the datasets in the *third group*. For the detail of each dataset, see Table 5.1. *Note: the x-axis numbering corresponds to the numbering in Table 5.1.*

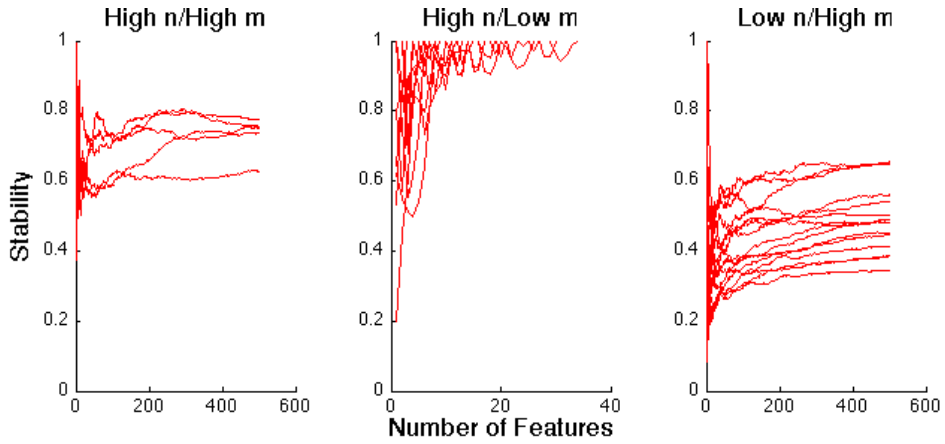


Figure 4.5: Jaccard Index stability for Fisher Score and all datasets. Similar to Figure except that this shows stability over different results cardinality ranges from 1 to min $(m, 500)$. We show results of Fisher Score only while the rest are omitted duo to similar behaivor.

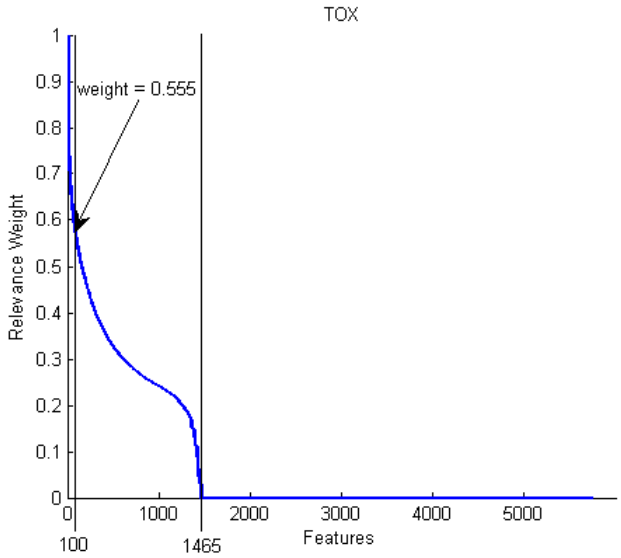


Figure 4.6: Features' relevance weight of TOX dataset using ChiSquare (sorted in an descending order).

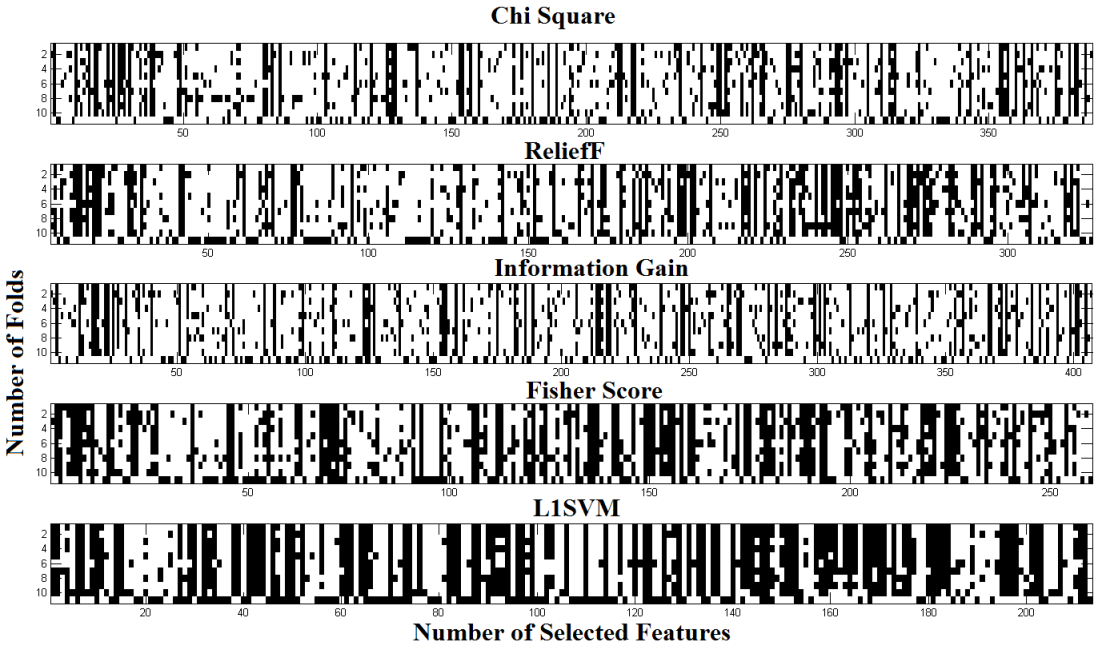


Figure 4.7: The frequency of selected features shows the impact of the sample variance on the selection, where the last fold has huge variation which leads to different sets of selected features. All these subplots show the same dataset, CLL-SUB.*Note: We show all features that were selected at least once in all folds.*

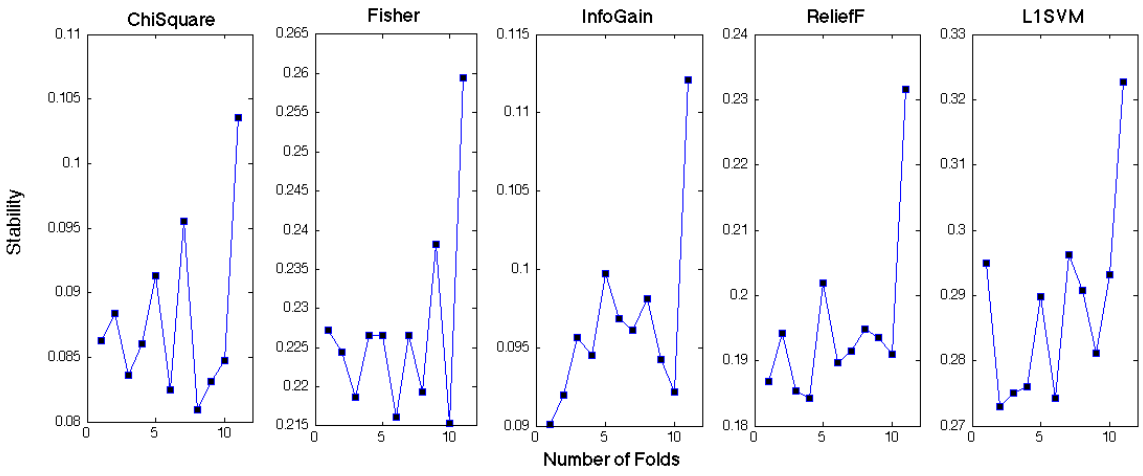


Figure 4.8: The stability after removing one fold each time. The stability is higher when we remove the last fold. The dataset is GLI.

In the future, we are going to investigate in more depth the above issues to propose a framework for a feature selection process that takes into account the factors mentioned above in effort to improve selection stability methods. We are going to study different forms of sample variation on the dataset, i.e. different distribution of the samples.

To conclude this paper, we studied several factors that affect the stability of selecting a subset of features. We empirically prove that the stability is dataset dependent, yet not completely algorithm independent. We found that the dimensionality and the sample size of the dataset have significant impact on the selection stability. The larger sample size has a positive impact while a larger dimensionality negatively impacts stability. We found that with a large enough sample size, the impact of dimensionality vanishes. Sample size and dimensionality, aside, the underlying distribution of the dataset plays an important role in stability. We found that the fold, with no overlap with other folds, tends to have different relevant features even when this fold is sampled from the same dataset. In addition, we studied

the impact of the number of selected features k on the stability. We found that determining k that is close to the optimal number is an important key to reflect the ultimate stability of the algorithm. Finally, we discuss the different behaviors of the feature selection algorithms on datasets with different characteristics. We showed that Fisher is less sensitive to the characteristics of the dataset when it provides good stability with the three groups of the datasets.

Table 4.1: Datasets statistics

		Dataset Name	#Samples n	Dimensionality m	#Classes
first group	1	CLL-SUB	111	11340	3
	2	GLA-BRA	180	49151	4
	3	TOX	171	5748	4
	4	GLI	85	22283	2
	5	PRO-CAN	171	11302	4
second group	1	ovarian-gilks	23	36534	2
	2	headneck-pyeon-2	23	54675	2
	3	oral-odonnell	27	22283	2
	4	leukemia-wei	27	21481	2
	5	renal-williams	29	17776	2
	6	colon-watanabe	30	54675	2
	7	colon-laiho	31	22283	2
	8	lung-bild	33	54675	2
	9	pancreas-ishikawa	36	22645	2
	10	breast-farmer	37	22215	3
	11	lung-barret	39	22283	2
	12	sarcoma-detwiller	40	22283	2
	13	prostate-true-2	40	12783	2
	14	lymphoma-booman	40	14362	2
third group	1	breasttissue	106	9	6
	2	dermatology	358	34	6
	3	ecoli	336	7	8
	4	glass	214	9	6
	5	heart	270	13	2
	6	iris	150	4	3
	7	liver-disorders	345	6	2
	8	post-operative	87	8	2
	9	soybean	47	35	4
	10	swissbank	200	6	2
	11	wdbc	569	30	2
	12	wine	178	13	3
	13	yeast	205	20	4

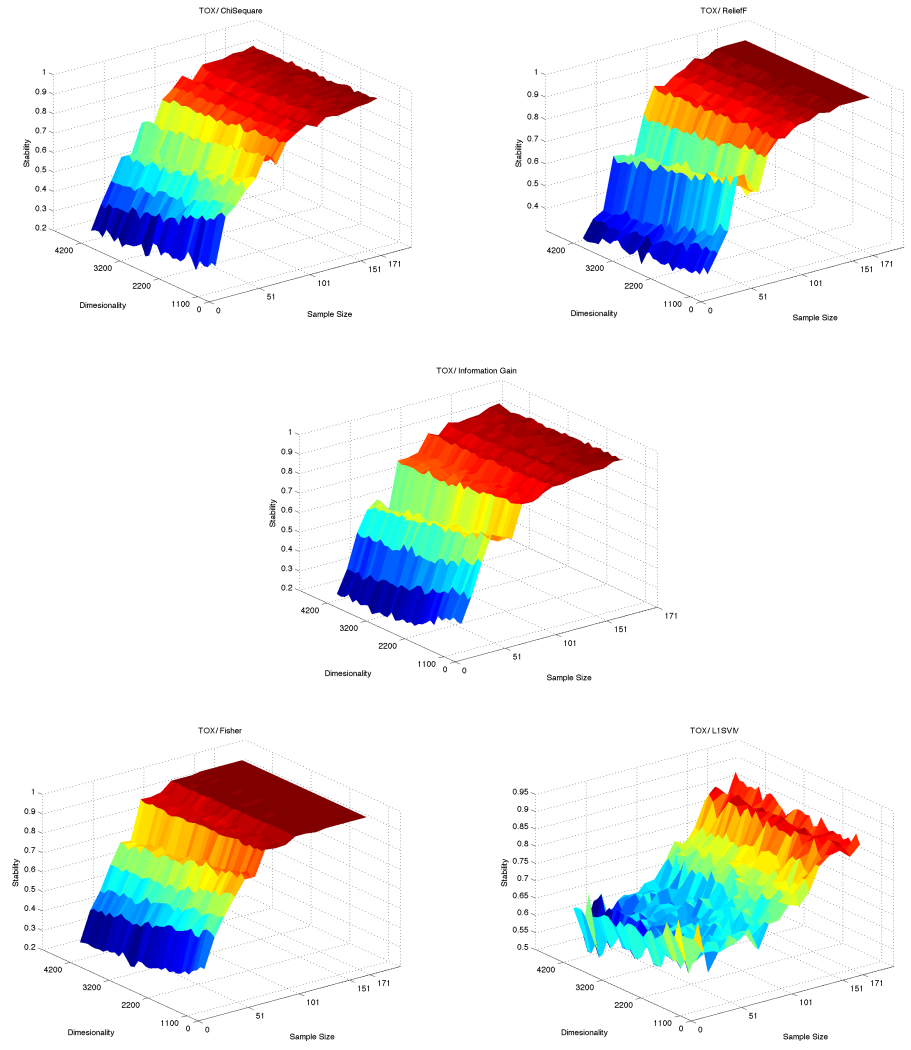


Table 4.2: The stability of the five feature selection algorithms vs. the sample size and the dimensionality

Chapter 5

SUPERVISED LOW RANK MATRIX APPROXIMATION FRAMEWORK FOR STABLE FEATURE SELECTION

As it was shown earlier, the stability of feature selection algorithms is mostly affected by certain dataset's characteristics. These characteristics include dimensionality m, sample size n and different data distribution across different folds. Thus the stability issue tends to be data dependent. This fact motivated this work to cure the instability of the algorithm by preprocessing the dataset rather than proposing a new selection technique.

Data noise is one of these undesired characteristics of the datasets. Unfortunately, most of high dimensional datasets, e.g. microarray, are extremely noisy [49] which degrades both the stability and the learning performance of the method [66, 49]. Therefore, reducing the level of noise is indispensable step toward more stable and accurate algorithms. However, noise reduction may lead to information lose if

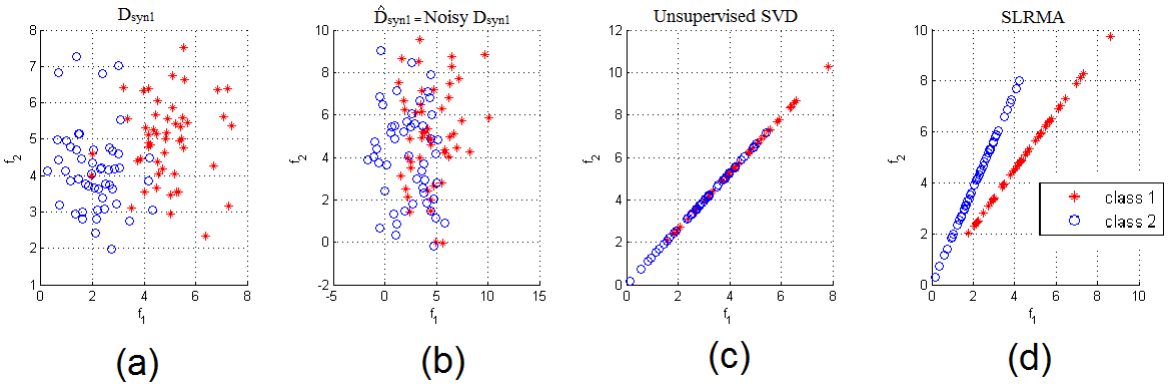


Figure 5.1: Supervised vs. unsupervised noise reduction.

Plot (a) is the original synthetic data $\mathbf{D_{syn1}}$, plot (b) shows $\hat{\mathbf{D}}_{\mathbf{syn1}}$ which is the dataset $\mathbf{D_{syn1}}$ after adding random noise, plots (c) shows the SVD low rank approximation for $\hat{\mathbf{D}}_{\mathbf{syn1}}$ without considering the class label, and plots (d) shows our contribution of supervised low rank matrix approximation of $\hat{\mathbf{D}}_{\mathbf{syn1}}$. Instances in plot (d) is linearly separable while (c) is not.

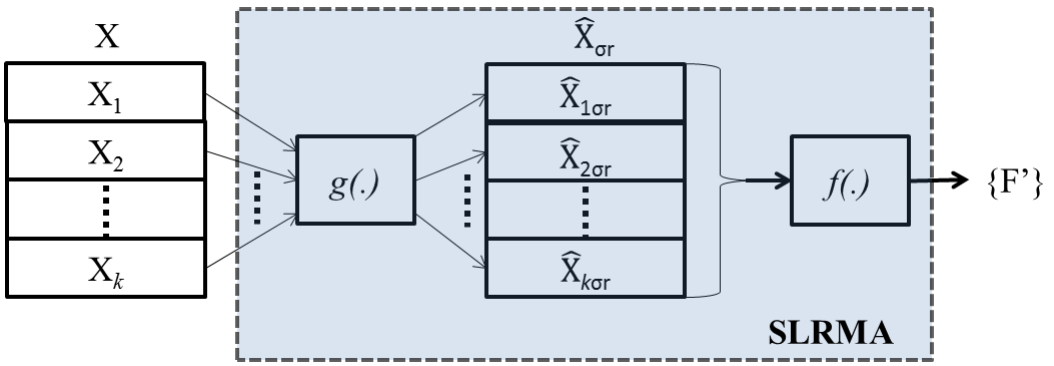


Figure 5.2: The proposed framework for supervised low rank matrix approximation SLRMA . $g(\cdot)$ is a low rank approximation method and $f(\cdot)$ is a feature selection algorithm that selects subset of features F'.

we ignore the class label in case of labelled data. In order to improve stability and learning performance, we should reduce the noise in a supervised manner. In this work, we propose a supervised noise reduction framework using low rank approximation techniques prior to feature selection step which proved to significantly improve stability and classification accuracy with different kind of datasets. To reduce data noise, we used two well-known low rank approximation techniques: Singular Value Decomposition SVD and Non-negative Matrix Factorization NMF.

5.1 Data Noise

Usually real-world datasets are corrupted by noise. Data noise is defined as the undesired data that interfere with the desired data[7]. The existence of noise is ubiquitous; therefore, reducing the noise level in the data is an essential preprocessing step in many domains including speech recognition, bioinformatics, image processing, signal processing. In addition, it reduces the space needed to store the data, reduces processing time, and improves learning performance etc. Thus, noise reduction not only maintains the current amount of information but also benefits from cleaning the data and discovers the hidden patterns.

Data noise differ from one case to another. Some noise is due to imperfection inherent in current technologies that collected the data. This kind of noise is known as *technical noise*. Technical noise was found to greatly degrade learning performance which makes eliminating noise is one important step toward improving performance [66, 49]. *Attribute noise* is another type of noise that alters the values of the component of the samples. It, also, could be due to irrelevant features within the data. According to the definition above, the irrelevant features can be interpreted as noise. However, in typical situations a feature can be partially relevant. In other words, a feature that is relevant to certain classes is not necessarily relevant to others. Figure 5.11 illustrates the notion of the relevance and irrelevance. In that figure, features f_1, f_2 and f_3 are relevant to Class 1 but irrelevant to Class 2 and 3. Similarly, $f_4, f_5$and f_6 are only relevant to Class 2. Likewise, f_7 and f_8 only are relevant to Class 3. While f_9 and f_{10} are irrelevant to all classes. Since these features are mostly irrelevant to most classes they could be considered as noise associated to that class. Thus, reducing the noise of a dataset while ignoring the class leads to information loss.

5.2 Low Rank Matrix Approximation

Low rank matrix approximation aims to find a low rank matrix $\hat{\mathbf{X}}_{\sigma_r}$ that approximate the matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$. Using the Forbenius norm, it minimizes the difference between these matrices as follows:

$$\min ||\mathbf{X} - \hat{\mathbf{X}}_{\sigma_r}||_F,$$

where $r \leq \{n, m\}$ is the rank of $\hat{\mathbf{X}}_{\sigma_r}$. The subject of matrix approximation is extensively studied [76, 2, 79]. There are different goals for those who apply matrix approximation to their domains. These goals include storage reduction, improving learning performance, improving computation efficiency, noise reduction, etc. For these purposes different methods were used including Singular Value Decomposition

(SVD), Non-negative Matrix Factorization (NMF), and so on. In this work, we will use these two well-known and widely used approaches as a noise reduction techniques in order to improve feature selection stability and maintain classification accuracy.

Singular Value Decomposition

SVD aims to decompose the original matrix $\mathbf{X}$ as following:

$$\mathbf{X} = U\Sigma V^T,$$

where U is an *n-by-n* matrix, Σ is *n-by-m* diagonal matrix with non-negative values in the diagonal, and V is an *m-by-m* matrix. The columns of U and V are called the left and right singular vectors respectively, while the diagonal entries of Σ, $\{\sigma_1, \cdots, \sigma_r\}$, is called the singular values of $\mathbf{X}$.

In the case of noise, the singular values of X are shifted uniformly [69]:

$$\sigma_i^2 = \hat{\sigma_i^2} + \xi^2.$$

This causes the singular values of the matrix to be non-zero. So, it is quite popular to reduce noise by eliminating lower singular values [5].

$$\hat{\mathbf{X}}_{\sigma_r} = U_r \Sigma_r V_r^T$$

Non-negative Matrix Factorization

NMF is unsupervised learning method in which non-negative matrix $\mathbf{X}$ is decomposed into two non-negative matrices W and H.

$$\mathbf{X} = WH$$

In order to find the optimal factorization for $\mathbf{X}$, the following cost function is used:

$$\min ||\mathbf{X} - WH||_F,$$

Although this cost function is widely used, it is not the only cost function. KL divergence is widely used too. NMF is successfully applied in different domains including text mining, natural language processing, image processing, information retrieval, speech recognition, molecular pattern discovery, etc. Similar to SVD, NMF reduces the noise and technological variation in the data [20].

5.3 Supervised Low Rank Matrix Approximation

As we discussed earlier, data noise can degrade learning performance and paralyze even highly powerful methods. In microarrys, for instance, the noise found in different gene arrays led biologists to false conclusions after much effort pursuing what they believed to be an "*Array of Hope*". This noise when discovered in 2003, the array became "*An Array of Problems*" instead [49]. Thus, *getting the noise out of gene arrays* [66] is a fundamental step toward more accurate and stable learning.

Different noise reduction techniques have been proposed as a preprocessing step to clean the dataset in order to improve learning performance and minimize the storage requirements without much loss of desired information. These reduction techniques aim to reduce variation in the dataset but usually they ignore the class label. A feature f_i may contain high variation in its values. This variation maybe considered noise if it is between instances from the same class. However, it will be a useful variation if it is between instances belonging to different classes. In Figure5.11, feature f_1 has a large variation between instances belonging to Class 1 and others that belong to Class 2 or 3. Existing noise reduction technique do not consider the class label leading to huge transformations in the feature and thus loses essential information embedded in the dataset. To overcome this shortcoming, we propose to use two popular low rank matrix approximation techniques for supervised noise reduction. In the experiment section we will show the benefit of using supervised

technique compared to approximating the whole matrix without considering the class information or approximating randomly partitioned matrix.

Supervised Approximation Framework - SLRMA

To formulate the problem, we assume $\mathbf{X} \in \mathbb{R}^{n \times m}$ and $\mathbf{Y}$ is a $k - mode$ matrix representation of the class label, where k is the number of classes. $\mathbf{Y}_{j,i} = 1$ if the instance $\mathbf{x}_j$ belongs to i^{th} class and 0 otherwise. Also, we assume $f(\cdot)$ to be a feature selection algorithm:

$$f(\mathbf{X}, \mathbf{Y}) \rightarrow \{\mathbf{F}'\},$$

where $\mathbf{F}'$ is a selected features subset. A typical low rank approximation method $g(\cdot)$ is usually defined:

$$g(\mathbf{X}, r) \rightarrow \{\hat{\mathbf{X}}_{\sigma_r}\}, \tag{5.1}$$

where $r \leq \min\{m, n\}$ is a predefined lower rank and $\hat{\mathbf{X}}_{\sigma_r}$ is a low rank matrix that approximate $\mathbf{X}$. Although this approach is effective in terms of noise reduction to some extent, it does not take class label into consideration which causes information loss. It can not distinguish between features' high variation due to noise or class affiliation. Thus, we propose to reduce matrix noise by utilizing the class information, see Figue5.2. This approach consists of two main steps. First, the dataset is partitioned into k partitions, $\{\mathbf{X}_1, \cdots, \mathbf{X}_k\}$. $\mathbf{X}_i$ contains instances, $\{x_1, \cdots, x_{n_i}\}$, that belongs to the i^{th} class and n_i is the sample size belong to that class. Then, each $\mathbf{X}_i$ is passed to $g(\cdot)$ to reduce the rank to a predefined rank r generating $\hat{\mathbf{X}}_{i,\sigma_r}$. Thus, the proposed method becomes:

$$SLRMA(\mathbf{X}, \mathbf{Y}, g(\cdot), r) \rightarrow \{\hat{\mathbf{X}}_{\sigma_r}\},$$

instead of Equation5.1, which preserves the class information. Second step, we pass the generated lower rank matrix to the feature selection method, $f(\hat{\mathbf{X}}_{\sigma_r})$. Algorithm1

shows how to generate the low rank matrix of $\mathbf{X}$ in a supervised manner. It iterates from 1 through k. For each single iteration i, it generates n_i-*by*-n matrix $\mathbf{Y}^i$, where $\mathbf{Y}^i_j$ equals 1 if x_j belongs to the i^{th} class and 0 otherwise. Next, it creates $\hat{\mathbf{X}}_i$, a n_i-*by*-m matrix that contains the samples that belong to the i^{th} class. Then, we use $g(\cdot)$ to approximate $\hat{\mathbf{X}}_i$. Finally, we combine all these sub-matrices approximations to create a matrix $\hat{\mathbf{X}}_{\sigma_r}$, where the operator $\otimes$ in Algorithm1 is a simple operator that put each data sample in its original position as it was in original dataset $\mathbf{X}$.

Algorithm 1: Supervised Low Rank Matrix Approximation SLRMA

input : $\mathbf{X}, \mathbf{Y}$, $g(\cdot)$, and *rank* r
output: $\hat{\mathbf{X}}_{\sigma_r}$: Low Rank Matrix

for $i \leftarrow 1$ **to** k **do**

`Generate`: $\mathbf{Y}^i \in \mathbb{R}^{n_i \times n}$, *where*

$$\mathbf{Y}^i_j = \begin{cases} 1, & x_j \text{ belongs to } i^{th} \text{ class.} \\ 0, & \text{otherwise.} \end{cases}$$

% *Low rank approximation using instances*
% *that belong to* i^{th} *class:*
$\hat{\mathbf{X}}_i = \mathbf{Y}^i \cdot \mathbf{X}$
$\hat{\mathbf{X}}_{i,\sigma_r} = g(\hat{\mathbf{X}}_i)$
% *Rejoin the whole matrix*
$\hat{\mathbf{X}}_{\sigma_r} = \otimes\, \hat{\mathbf{X}}_{i,\sigma_r}$

Table 5.1: Datasets statistics

	Dataset Name	Type	#Samples n	Dimensionality m	#Classes
1	BLOOD-89	Microarray	89	2759	2
2	SMK-CAN-192	Microarray	192	19993	3
3	warpAR10P	Image	130	2400	10
4	warpPIE10P	Image	210	2420	10

5.4 Experiments and Results

In this section, we conduct several experiments to demonstrate the effectiveness of the proposed method in improving selection stability, maintaining classification accuracy and selecting relevant features. The first experiment is conducted using synthetic datasets to compare conventional, i.e. global, matrix approximation approach to the proposed approach, i.e. supervised matrix approximation. The second experiment is conducted on four real-world datasets; the first two are microarrays and the others are face images datasets , see Table5.1 for datasets' characteristics.

In real-world datasets, the relevant features are not given. Therefore, we are not able to guarantee that this method is capable of selecting relevant features. So, we conducted another extensive experiment using synthetic datasets. We generated 500 synthetic datasets with given relevant set of features F_{rel} that is 25% of m. We used five well-known feature selection algorithms: ChiSquare, ReleifF, Information Gain, Fisher Score, and ℓ_1SVM. Also, we used Support Vector Machine (SVM) as a classifier and Jaccard Index to assess the stability of the selection methods.

In this experiment, we utilized Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF) to approximate the dataset in order to reduce data noise. In our proposed approach(SLRMA($\mathbf{X}, \mathbf{Y}, g(\cdot), r$)), we chose $g(\cdot) = \{SVD, NMF\}$. Although we chose $r = \{1, 2, \cdots, 6\}$, we show the results of $r = \{1, 2\}$ in this work because the higher the rank, the more similar the approximated matrix to the original matrix. Thus, the stability tends to be either lower or no significant improvement with higher ranks. It was found that the proposed approach significantly improves selection stability and maintains classification accuracy compared to the baseline method, the original matrix, and existing matrix approximation ap-

proach where the class label is not taken into consideration. Also, we used the synthetic datasets to elucidate the deficiency of the current approach with respect to selection stability.

Model Selection Selecting an optimal rank r is an open problem. A higher r makes the approximated matrix closer to the original matrix, while a smaller r results in a larger reduction in noise. Thus we tried variety of r ranging from $r = 1$ to $r = \min\{n_c\}$ where n_c is the smallest number of samples that belong to one class in that dataset. For example, the dataset BLOOD-89 has two classes the $\min\{n_c\}$ in this case is 42, which is the number of samples belonging to class 2. According to the stability results shown in Table 5.2, we obtain the highest stability with $r = 1$ and $r = 2$. The rest of the results are omitted because they expectedly get closer and closer to the original dataset due to higher r.

In terms of SVD, choosing r is usually based on the singular values σ_i. Where $\sigma_i = 0$ is associated to data noise, thus, they are removed. In contrast, largest singular values capture the desired information in the original dataset. Keeping the largest singular values leads to aggressively cleaning the data. We plotted the singular values and found that the difference between the first singular value σ_1 and σ_2 is great, while the differences between any other consecutive values, for example σ_2 and σ_3, are slim. Thus, theoretically, choosing $r = 2$ is enough to capture significant amount of desired information in $\mathbf{X}$.

On the other hand, NMF groups the samples into r clusters. Accordingly, choosing $r = 1$ will be optimal for supervised NMF since we assume that samples from the same class belong to one cluster only.

Supervised Approximation Framework - SLRMA

Using four real-world datasets shown in Table 5.1, we generated four approximated matrices out of the original dataset $\mathbf{X}$ using SVD and NMF each with rank $r = 1$ and $r = 2$, producing $\hat{\mathbf{X}}_{\sigma_1}$ and $\hat{\mathbf{X}}_{\sigma_2}$ for each method. Then, we run each of the subject algorithms, using 10-fold-Cross-Validation technique holding one fold each time as a validation set and the rest as the training set. Our proposed approach, as illustrated in Table 5.2, significantly improves the stability compared to using the original dataset in all cases except in one case, where the difference between our best result and the original is 0.01%. In addition, regardless of which reduction technique is used, supervised noise reduction was able to improve the stability by an average range from 46% to 61.54%. In addition, the proposed method maintains the the average accuracy as the baseline method which indicates that the selected features were informative as well which gives the superiority to the supervised approach because it select stable and accurate features, see Table5.3.

It is found that both SVD and NMF improve selection stability while maintaining classification accuracy, yet, SVD outperforms NMF on average. Thus, if a practitioner aims to improve stability and maintain accuracy, SVD can be the first option. On the other hand, aggressive rank reduction tends to give better results in terms of stability.

The Precision of Selecting Relevant Features

It is a known fact that there might be different subsets of features that are equally good in terms of classification accuracy. However, these subsets might not be as equally relevant. Therefore, we might select a particular subset over and the overs

but it is not the most relevant one. In this case, we improve stability and maintain reasonable accuracy, yet, we are not choosing the most relevant subset. This degrades the quality of further domain analysis. Thus, it is important to show that SLRMA framework selects not only a stable but also relevant subset of features.

In this experiment, we conducted another extensive experiment using synthetic datasets. We generated synthetic data with dimensionality $m = \{10, 20, 30, \ldots, 500\}$, fixed sample size $n = 100$ and binary class label. Of each dimensionality, we generated 10 datasets. Thus, the total number of datasets is 500. We chose $m_{rel} = 0.25 \cdot m$. In other words, the number of relevant features is 25% of the total number of features m. Also, to simulate the real-world situations, we introduced random noise to each data sample $X_i^{\alpha} = X_i + \alpha\mathcal{N}(\mu, \sigma)$, where $\mu = 0$ and $\sigma = |\max(X)|$. The level of noise $\alpha = \{0, 0.3, 0.7, 1\}$. $\alpha = 0$ means there is no introduced noise, hence, we use the original data. In addition, we introduced class noise $\eta = \{5, 10, 20, 30, 40, 50\}\%$. This means, we randomly misclassify $\eta\%$ of the class labels. Here, $\eta = 50\%$ is equivalent to random labeling. Since we are given the relevant feature set F_{rel}, we evaluate the precision of selecting relevant features using Eq(5.2), where F_{sel} is the set of selected features and $|F_{rel}| = |F_{sel}| = m_{rel}$.

$$P = \frac{|F_{rel} \cap F_{sel}|}{m_{rel}} \tag{5.2}$$

Since the results are very consistent across different dimensionalities, we show only results corresponding to $\eta = 5\%$ and $\eta = 30\%$ and dimensionality $m = \{10, 220, 430, 500\}$. Other results with different m and η values are omitted since the shown ones are representative. The figures from Figure 5.3 to Figure 5.10 illustrates the results for Fisher score, Chi Square, Information Gain and ReliefF respectively. ℓ_1SVM was omitted here since it failed miserably in selecting relevant

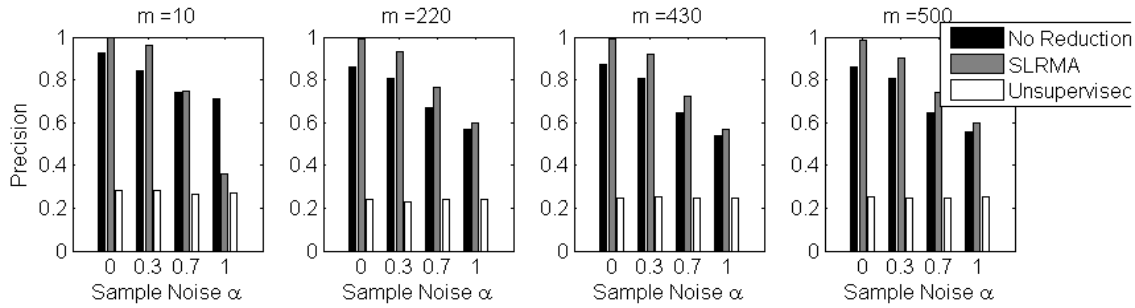


Figure 5.3: Fihsher Score, misclassification Rate $\eta = 5\%$

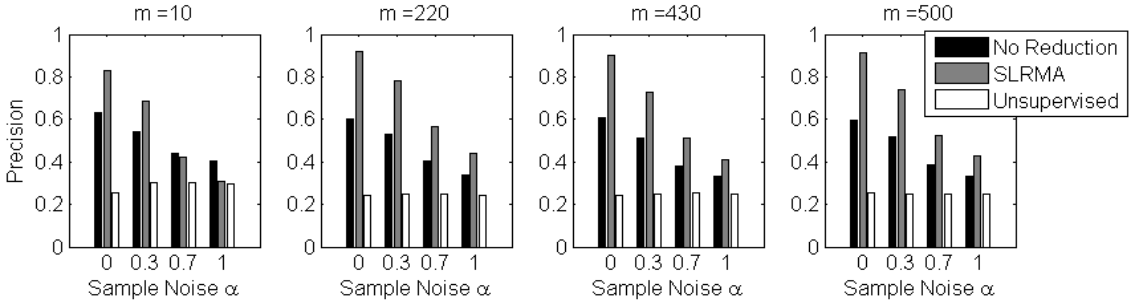


Figure 5.4: Fihsher Score, misclassification Rate $\eta = 30\%$

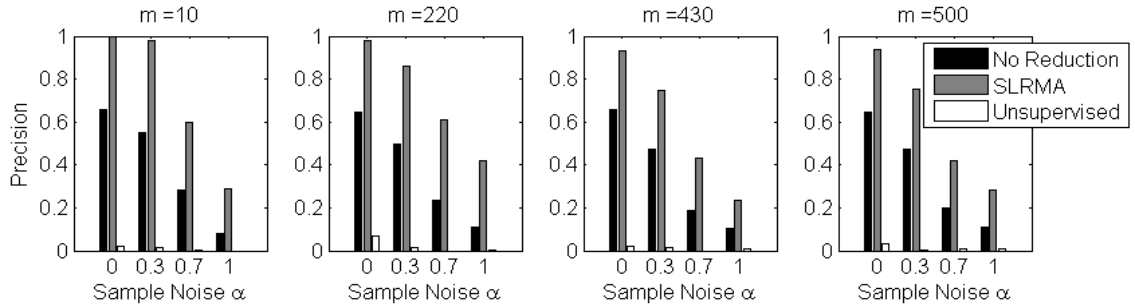


Figure 5.5: Chi Square, misclassification Rate $\eta = 5\%$

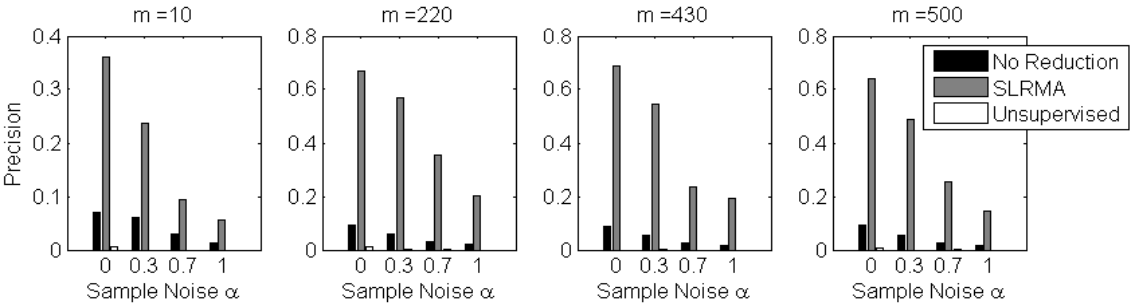


Figure 5.6: Chi Square, misclassification Rate $\eta = 30\%$

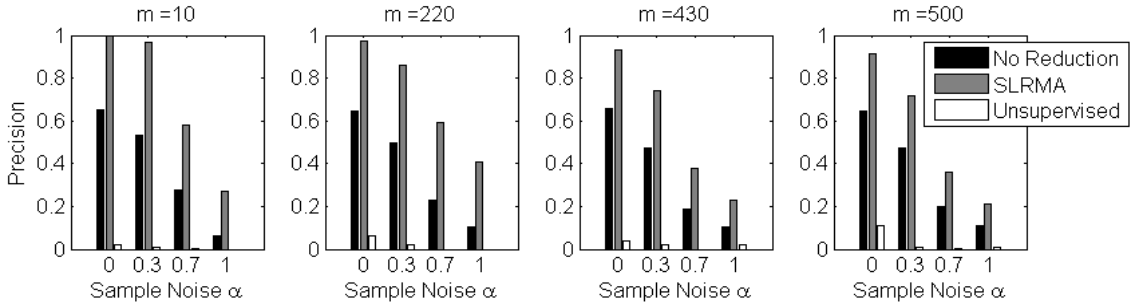


Figure 5.7: Information Gain, misclassification Rate $\eta = 5\%$

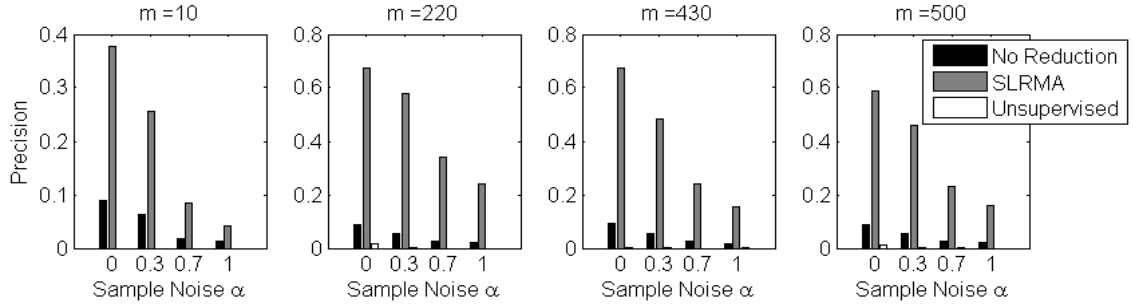


Figure 5.8: Information Gain, misclassification Rate $\eta = 30\%$

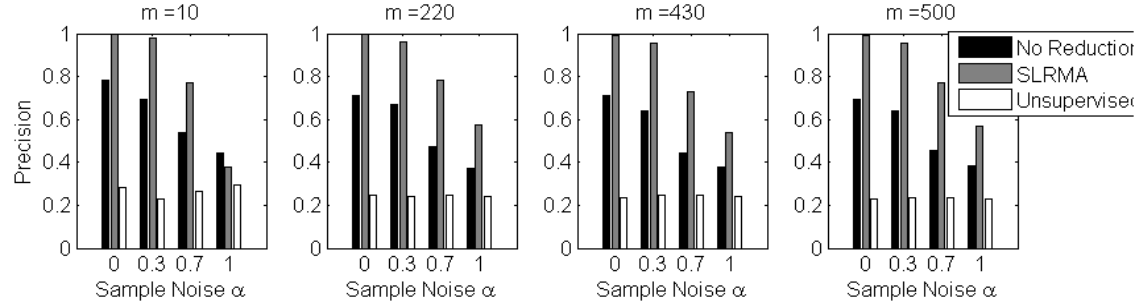


Figure 5.9: ReliefF, misclassification Rate $\eta = 5\%$

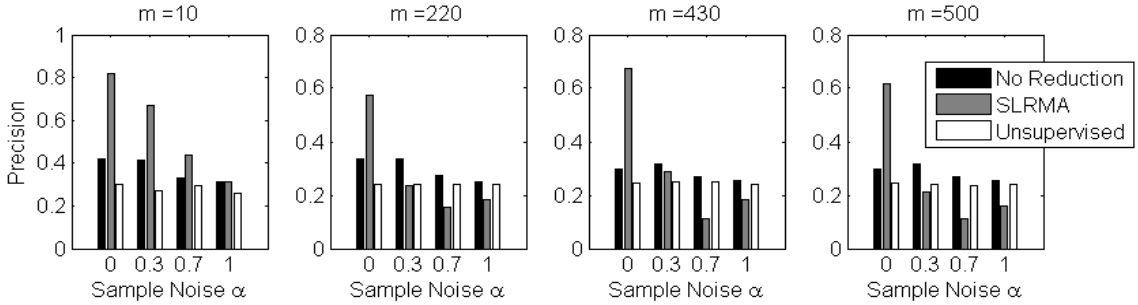


Figure 5.10: ReliefF, misclassification Rate $\eta = 30\%$

features in all cases. I show results correspond to misclassification rate, η equals 5% and 30% respectively.Each figure contains four plots that correspond to the dimensionality. The y-axes of each plot is the precision of selecting relevant features Eq(5.2). the x-axes, on the other hand, contains four groups of bars correspond to the sample noise α. The black bar represents the original dataset. The gray bar represents the our method, SLRMA. And the white bar represent the unsupervised noise reduction.

The results shown in the Figures (5.3-5.10) demonstrate the superiority of the proposed method over the baseline methods (i.e. without reduction or unsupervised reduction) in selecting relevant features. Unsupervised noise reduction was not better than random selection since it is always around 20% or less precision, which is similar to the ratio of m_{rel} to m. Chi Square and Information Gain behaved similarly, they could not perform selection in case of unsupervised noise reduction since the reduction significantly reduced variance between classes.

It is noticeable that when η equals 30%, the algorithms failed to select relevant features in the baseline approach. Unlike SLRMA where it was able to retrieve very large portion of the relevant features comparing to the baseline method even with the existence of high level of misclassification and data noise, Figures (5.6,5.8).

Each algorithm, on the other hand, was able to select more relevant features when $\eta = 5\%$ than $\eta = 30\%$. Expectedly, the precision was gradually reduced from $\eta = 5\%$ to $\eta = 40\%$, while $\eta = 50\%$ was almost random due to random labeling. Chi Square and Information Gain behaved similar to each other in precision. It was noticed that these algorithms with SLRMA framework was able to significantly improve precision when high level of class noise η exists comparing to both no or unsupervised noise reduction. From the results, we observe that Fisher Score is resis-

tant to both sample noise and class noise, Figures (5.3and 5.4). ReliefF, Figure 5.9 is similar to Fisher Score when η is low. However, when η is high and with $\alpha \geq 30\%$, ReliefF with SLRMA framework did not perform any better than no or unsupervised noise reduction, Figure 5.10. Yet, ReliefF still can distinguish relevant features if the sample noise is small even with existence of high level of misclassification.

From these results we can draw interesting conclusions. First, the accuracy of these selected subsets were comparable but why this is the case if the sets were not equally relevant? The answer could be due the argument that all we need to achieve high accuracy is few relevant features, not all or even large number. Although it is desired to perform good even with existence of irrelevant features, this will degrade the quality of further domain analysis. Also, we can say that Fisher score is trustworthy methods comparing to others since it shows strong resistance against sample and class noise.

Further Experiments and Discussion

We would like probe further what are the determining factors attributed to selection stability and performance improvement. Since supervised approximation divides the training data into partitions based on class labels, we naturally question (1) if the use of the label information would make any significant difference with respect to unsupervised approximation, and (2) if partitioning the training data into smaller sizes would make any difference.

Supervised vs. Unsupervised Approximation To answer the first question, we conducted controlled experiments using synthetic data $\mathbf{D_{syn1}}$, Figure 5.11, which has two features and two classes. Figure 5.11(a) shows that these two classes could be linearly separable with some outliers. f_1 is relevant while f_2 is not. Then, we

applied data noise on $\mathbf{D_{syn1}}$ to generate $\hat{\mathbf{D}}_{\mathbf{syn1}}$ as shown in Figure 5.11(b). Now, our goal is to reduce the introduced noise to $\hat{\mathbf{D}}_{\mathbf{syn1}}$ whilst maintaining the same level of class separability as in the original dataset $\mathbf{D_{syn1}}$. We approximate $\hat{\mathbf{D}}_{\mathbf{syn1}}$ using SVD, Figure 5.11(c). The approximated matrix is not linearly separable. Similarly, we apply our SLRMA to $\hat{\mathbf{D}}_{\mathbf{syn1}}$. Figure 5.11($d$) shows the approximated data that reveals separability inherited from $\mathbf{D_{syn1}}$. Therefore, utilizing the label information significantly helps clean the data, leading to better separable classes.

We generated another synthetic data $\mathbf{D_{syn2}}$, Figure 5.12, with similar characteristics to $\mathbf{D_{syn1}}$. The two classes in $\mathbf{D_{syn2}}$, Figure 5.12(a), are linearly separable. f_1 is relevant, i.e., it could linearly separate the two classes, while f_2 is not. Most feature selection algorithms can distinguish the relevant feature easily in such clean data. The question is whether a relevant feature is distinguishable after noise reduction. To verify this, we applied SVD on $\mathbf{D_{syn2}}$ with $r = 1$ in supervised and unsupervised manner. Figure 5.12(b) shows the approximated matrix using the unsupervised approach. The two classes are linearly separable, and the two features are equally good in terms of class separation. In other words, f_1 and f_2 became equally relevant after we applied unsupervised SVD, which is not the case in $\mathbf{D_{syn2}}$. Figure 5.12(c) shows the resulting data from applying supervised SVD on $\mathbf{D_{syn2}}$. The approximated matrix using the supervised approach has the same identity as the original matrix in terms of feature relevancy. f_1 is relevant, thus can linearly separate the classes, while f_2 remains irrelevant. This significantly degrades the selection stability.

We conducted another experiment with unsupervised noise reduction on real-world datasets in Table 5.1, and the five feature selection methods explained earlier. Because existing approximation approaches do not consider samples' affiliation, they do not preserve the class information and do not preserve the relevance score for

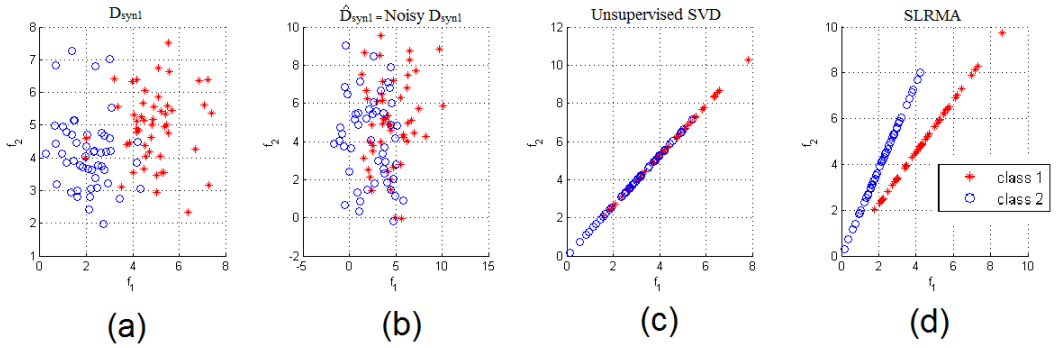


Figure 5.11: Plot (a) is the original synthetic data $\mathbf{D_{syn1}}$, plot (b) shows $\hat{\mathbf{D}}_{\mathbf{syn1}}$ which is the dataset $\mathbf{D_{syn1}}$ after adding random noise, plots (c) shows the SVD low rank approximation for $\hat{\mathbf{D}}_{\mathbf{syn1}}$ without considering the class label, and plots (d) shows our contribution of supervised low rank matrix approximation of $\hat{\mathbf{D}}_{\mathbf{syn1}}$. Instances in plot (d) is linearly separable while (c) is not.

each feature that can cause instability in the feature selection process. Hence, the unsupervised approach either significantly degrades the selection stability, or worse, it was not able to distinguish the relevant features and thus did not assign any score to the features. The results are not presented due to the page limit.

Partitioning One might argue that this improvement of selection stability is due to lower rank inherited in each fold of the dataset. Hence, we conducted another experiment to show that the improvement is from the supervised noise reduction. In this experiment, we randomly selected the folds regardless of their class affiliation. In a supervised manner, we partition the dataset using the class label $\mathbf{Y}$ but in this case we replace $\mathbf{Y}$ with a randomly generated target, $\mathbf{Y}'$. So, the partitioning is random, while keeping the original class label $\mathbf{Y}$ for feature selection and validation steps. We find that random partitioning performs worse than using the original dataset in all cases, and almost as poorly as the unsupervised matrix approximation. We thus conclude that the proposed approach is a promising way to preprocess the dataset to improve selection stability.

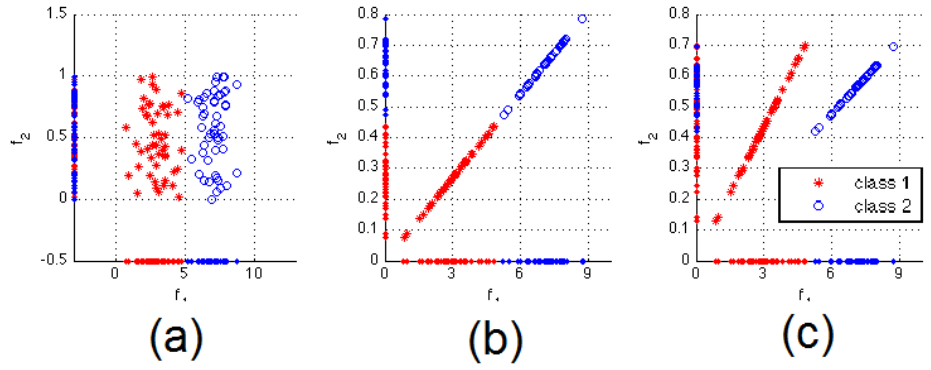


Figure 5.12: Plot (a) is the original synthetic data $\mathbf{D_{syn2}}$, plot (b) shows Supervised SVD for $\mathbf{D_{syn2}}$, and plots (c) shows supervised SVD of $\mathbf{D_{syn2}}$

Table 5.2: Jaccard index stability results

Algorithm	Datasets	X	SVD		NMF	
			$\hat{\mathbf{X}}_{\sigma_1}$	$\hat{\mathbf{X}}_{\sigma_2}$	$\hat{\mathbf{X}}_{\sigma_1}$	$\hat{\mathbf{X}}_{\sigma_2}$
ChiSquare	BLOOD-89	0.476	0.662	**0**.673	0.422	0.620
	SMK-CAN-192	0.221	0.604	**0**.741	0.619	0.577
	warpAR10P	0.230	**0**.435	0.238	0.382	0.351
	warpPIE10P	0.466	0.273	0.248	**0**.546	0.326
ReliefF	BLOOD-89	0.286	0.874	0.750	**0**.982	0.801
	SMK-CAN-192	0.439	0.893	0.843	**0**.978	0.841
	warpAR10P	0.426	0.690	**0**.831	0.549	0.800
	warpPIE10P	**0**.695	0.520	0.686	0.461	0.565
InfoGain	BLOOD-89	0.475	**0**.662	0.656	0.478	0.599
	SMK-CAN-192	0.256	**0**.896	0.719	0.610	0.676
	warpAR10P	0.237	0.336	0.261	0.331	**0**.383
	warpPIE10P	0.545	0.330	0.442	**0**.584	0.429
Fisher	BLOOD-89	0.300	0.903	0.799	**1**	0.721
	SMK-CAN-192	0.374	0.825	0.812	**0**.956	0.728
	warpAR10P	0.615	**0**.893	0.832	0.358	0.757
	warpPIE10P	0.686	0.794	0.815	0.471	**0**.831
ℓ_1SVM	BLOOD-89	0.311	**0**.997	0.987	0.815	0.996
	SMK-CAN-192	0.460	**0**.993	0.638	0.821	0.355
	warpAR10P	0.820	**1**	0.907	0.989	0.904
	warpPIE10P	0.703	**0**.993	0.928	0.800	0.911
Average		0.451	**0**.728	0.690	0.657	0.658
Average Improvement			**61.54%**	53.04%	45.79%	46%
Number of Max		1	**8**	3	6	2

Table 5.3: kNN accuracy results

Algorithm	Datasets	X	SVD		NMF	
			$\hat{\mathbf{X}}_{\sigma_1}$	$\hat{\mathbf{X}}_{\sigma_2}$	$\hat{\mathbf{X}}_{\sigma_1}$	$\hat{\mathbf{X}}_{\sigma_2}$
ChiSquare	BLOOD-89	0.42	0.51	0.59	0.55	**0.6**
	SMK-CAN-192	**0.65**	0.63	0.59	0.63	**0.65**
	warpAR10P	**0.79**	0.71	0.74	0.69	0.7
	warpPIE10P	0.89	**0.95**	0.87	0.88	0.86
ReliefF	BLOOD-89	0.49	0.54	**0.58**	0.51	0.52
	SMK-CAN-192	0.61	0.64	**0.65**	0.6	0.63
	warpAR10P	**0.82**	0.66	0.67	0.57	0.68
	warpPIE10P	**0.93**	**0.93**	0.92	0.9	0.92
InfoGain	BLOOD-89	0.42	0.51	**0.59**	0.51	**0.59**
	SMK-CAN-192	0.59	0.61	**0.67**	0.63	0.65
	warpAR10P	**0.8**	0.7	0.74	0.7	0.74
	warpPIE10P	0.84	0.89	0.9	0.88	**0.91**
Fisher	BLOOD-89	0.51	0.59	**0.65**	0.53	0.59
	SMK-CAN-192	0.57	0.64	0.6	0.62	**0.7**
	warpAR10P	**0.8**	0.64	0.63	0.6	0.69
	warpPIE10P	0.93	0.8	0.93	0.82	**0.94**
ℓ_1SVM	BLOOD-89	0.51	0.54	0.55	**0.57**	0.56
	SMK-CAN-192	0.64	0.65	0.66	0.66	**0.68**
	warpAR10P	**0.7**	0.67	0.69	0.68	0.69
	warpPIE10P	0.81	0.87	**0.89**	0.8	0.88
Average		0.69	0.69	**0.71**	0.67	**0.71**

Chapter 6

LOCAL SVD FOR STABLE FEATURE SELECTION FOR CLUSTERING

6.1 Introduction

Data clustering is a challenging problem especially with the presence of huge dimensionality. Usually, the number of relevant features to given clusters is very small while the rest of the features are irrelevant. This problem (a.k.a. the *curse-of-dimensionality*) not only degrades the clustering quality but also increases computational complexity. Therefore, feature selection for clustering is an indispensable step to select relevant and eliminate redundant and irrelevant features [17, 25, 10]. The goal of feature selection for clustering is to find the set of features that are relevant to the underlying clusters in the dataset. One common approach, demonstrated in Figure 6.1, is to (1) utilize a clustering method to generate clusters, then, (2) apply feature selection method that guided by the generated clusters to select the relevant features [64, 12].

To evaluate the quality of the selected features, most existing literature uses the learning performance, i.e., accuracy of clustering unseen data. If the selected subset is able to generalize on unseen data, the selection is considered to be good. However, it is noticed recently that there might be several candidate subsets of features, which might or might not overlap with each other, perform equally good on unseen data [47, 73, 91, 1, 3]. This raises the following question: which subset we should select? This problem is even more concerning in real-world applications due to the existence of data perturbation and noise by nature. The data perturbation might be in the form of new sample(s) introduced to the dataset. This leads to selecting significantly different features each time we apply the same feature selection algorithm on different sub-sampling of the dataset. In fact, selected features can

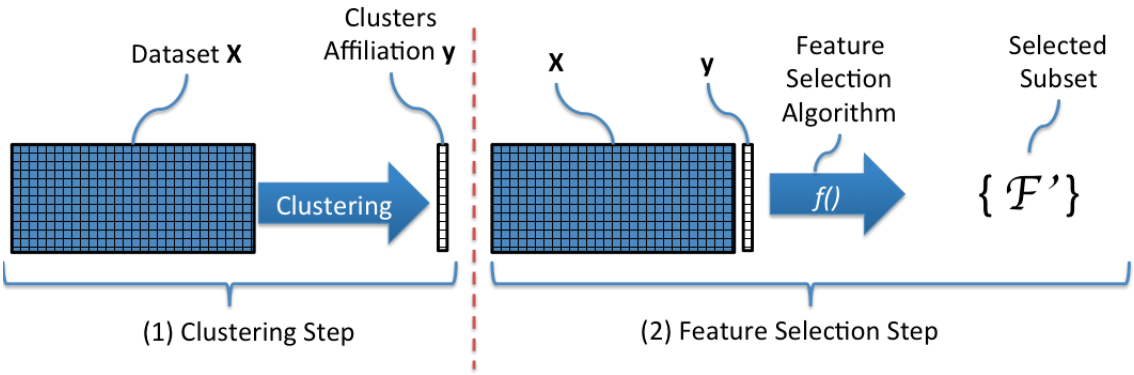


Figure 6.1: Conventional feature selection for clustering framework

define and interpret clusters[18, 97, 96]. If the selection is unstable, this infers different interpretations for the clusters. Each interpretation will correspond to one set of features. Consequently, domain expert confidence in the feature selection method degrades. Assume, for example, a microarray with high dimensionality m and n samples, usually, $n \ll m$. Domain experts, biologists in this case, believe that the number of genes (i.e. features) related to a certain disease is very small. Also, this set of relevant genes should not change dramatically from patient to another [92, 1]. What is noticed is that when we apply feature selection on samples randomly drawn from the same dataset, the selected feature subsets are very inconsistent even if the overlap of the sub-samples is as high as 80%. This is called selection stability or stability of feature selection algorithms. The selection stability has drawn an increasing attention lately [47, 73, 91, 1, 92, 3, 88]. It is defined as the *sensitivity of the selected subset to perturbation on the dataset*. It is important to note that improving stability without considering the learning performance is not desired. For example, we can just select the same set of features each time. This will be very stable selection, but the performance will be bad. Thus, the goal is to stabilize the selection while maintaining reasonable performance.

Challenges and Contributions

The existing work that tackles the selection stability problem mostly studies the stability of feature selection for classification. To the best of our knowledge, this is the first work that investigates the stability of feature selection for clustering. In the case of unsupervised learning [17, 25], it is even more challenging to decide whether the selection stability is desired property. In other words, we don't know how many underlying hypotheses in the data. Therefore, any set of features may produce good clustering that satisfies a hypothesis. The evaluation of the selected features, in this case, totally depends on the hypothesis of the clusters. For example, a microarray that was harvested for a Colon cancer may contain other human characteristics or diseases that are characterized by different sets of features. In case of supervised learning, we are given the hypothesis, say Colon Cancer. Thus, the selected set of features must satisfy Colon cancer hypothesis, although this hypothesis may not be the only or even the dominant one. In unsupervised learning, this is even more challenging due to (1) the potential existence of several hypotheses embedded in the huge dimensionality of the dataset and (2) the lack of domain knowledge regarding these hypotheses which might be helpful to guide the search of the relevant features to this exact hypothesis.

Motivated by the aforementioned facts, we investigate the selection stability for clustering. Noteworthy, we do not aim to improve clustering quality in this work. Instead, we assume we obtain good clusters and we want to stabilize the selection that maintains the quality of clusters. In this paper, we propose a framework that can help the a family of feature selection algorithms for clustering, demonstrated in Figure 6.1, to obtain stable and relevant features to a certain hypothesis. This framework involves low rank matrix approximation and approximates each cluster

separately. We call this framework Local Singular Value Decomposition (LSVD). The idea behind the local approximation is that each cluster should be independently dealt with in the preprocessing step. SVD is known to reduce data noise by reducing the variation of the samples. However, when we have more than one cluster in the data, we want to reduce the variation within the cluster while elevating the variation between clusters. The empirical results demonstrate the effectiveness of the proposed method. LSVD was able to significantly improve the stability while maintaining the clustering quality.

The remaining of this paper is organized as follows: we introduce the notion of stability for feature selection for clustering. Then, we define and formulate the problem. Next, the proposed LSVD framework is introduced. Finally, we empirically prove the effectiveness of LSVD in stable and accurate selection.

6.2 Selection Stability for Clustering

Feature selection consists of two steps: (1) feature space search and (2) feature evaluation [55, 56]. The goal is to find a small subset of features that is relevant to the defined hypothesis while removing irrelevant or redundant ones. The feature evaluation is quite straight forward when the class label is available. For example, the correlation matrix between the features and the class label could be one possible way to guide the feature selection. However, for the clustering problem, we do not have the class label, which making feature evaluation be a challenging problem.

A widely used way to do feature selection for clustering is to first extract clusters then apply feature selection that is guided by the extracted clusters [10, 64, 83]. Approaches in this family have several drawbacks. First, there might be more than one hypothesis embedded in the high dimensional dataset. The desired

or dominant hypothesis is not known, hence, it is not easy to be evaluated without domain knowledge. Second, for each hypothesis there could be several candidate sets of features that seem equally good in terms of clustering performance, arising the issue of selection stability. If the feature selection method is able to select similar sets each time with the existence of data perturbation, the method is called stable. Otherwise, it is unstable, which degrades confidence in the selected features because the method selects different set each time.

Accordingly, the selection stability of unsupervised learning has one more aspect than that of supervised learning. This aspect is the potential of existence of different unknown hypotheses while lack of knowledge about the desired one(s). For example, in subspace clustering, we aim to find all existing clusters in the dataset even if the clusters belong to different hypotheses and defined by different sets of features [4]. However, in this work we do not consider this case since our focus is the stability of feature selection. Therefore we aim to find only one hypothesis and select the set of features that form the clusters belong to this hypothesis. We extract k clusters then we need to select features that are able to accurately assign unseen samples. In addition, we aim to stabilize the selection of the features that form the extracted clusters.

6.3 Problem Statement

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ be the feature set where m is the number of features and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times m}$ be a given data set with n data points where $n \leq m$. One significant data mining task involving $\mathbf{X}$ is data clustering. The goal is to group similar samples in one cluster while dissimilar sample are in different clusters. Formally, we need to maximize the within-cluster similarity and maximize the between-clusters similarity.

Most of existing clustering algorithms cannot handle high-dimensional data effectively due to the curse of dimensionality and the presence of a large number of irrelevant and redundant features can further mislead the clustering algorithms. Therefore, feature selection is an indispensable step. For supervised problems, the class label guides the selection since we have a way to measure the relevancy score. Therefore supervised feature selection is formally stated as:

$$f(\mathcal{F}; \mathbf{X}, \mathbf{y}) \rightarrow \{\mathcal{F}'\} \tag{6.1}$$

where $f(\cdot)$ is a feature selection criteria and $\mathbf{y}$ is the class label. However, for the clustering problem, we don't have the class label $\mathbf{y}$. Thus, a clustering technique $h(\cdot)$ is utilized to generate $\mathbf{y}$ with a predetermined number of clusters k:

$$h(\mathbf{X}, k) \rightarrow \mathbf{y} \tag{6.2}$$

With the generated label, we can do unsupervised feature selection in a supervised manner as demonstrated in Figure 1 and we can substitute the generated clusters $\mathbf{y}$ from Eq(6.2) into Eq(6.1). Combining these two equations results in the following formal equation for feature selection for clustering:

$$f(\mathcal{F}; \mathbf{X}, h(\cdot)) \rightarrow \{\mathcal{F}'\} \tag{6.3}$$

Eq(6.3) is likely to find many sets of features that seem equally good and severely suffer from the selection instability problem. Therefore our goal is to develop a framework to make Eq(6.3) more stable while maintaining the clustering quality.

6.4 Framework for Stable Feature Selection for Clustering

The stability of supervised feature selection algorithms has gained an increasing attention in last few years. It is defined as the sensitivity of the selection to data perturbation. However, selection stability with the absence of the class label makes this problem even more challenging. We propose a stable feature selection framework that aims to provide more stable selected subsets.

A common feature selection for clustering approach first extracts class labels for the data samples by clustering techniques, mostly *k-means*. Then, these labels, i.e. clusters, would be used as class labels in the conventional supervised feature selection.

The proposed framework is demonstrated in Figure 6.2. We begin by extracting clusters using any desired clustering technique, *k-means* in this work. Then, local matrix approximation using SVD is performed on the dataset using the generated cluster affiliation. By local we mean that each set of samples that belong to one cluster will be approximated independently. The approximated clusters will be combined to obtain the new approximated dataset. To this end, we obtained an approximation of the original dataset, yet, the clustering we have was obtained using the original matrix which might do not hold accurately after the approximation. Therefore, one more clustering of the samples based on the approximated matrix is performed. Finally, we perform feature selection using any appropriate algorithm. An experiment we conducted indicates the importance of this step, otherwise, the clustering may not be accurate. We discuss this more in the discussion section. Therefore, the proposed framework can be formulated as follows:

$$f(\mathcal{F}; \mathbf{X}, h(\cdot), g(\cdot)) \rightarrow \{\mathcal{F}'\} \tag{6.4}$$

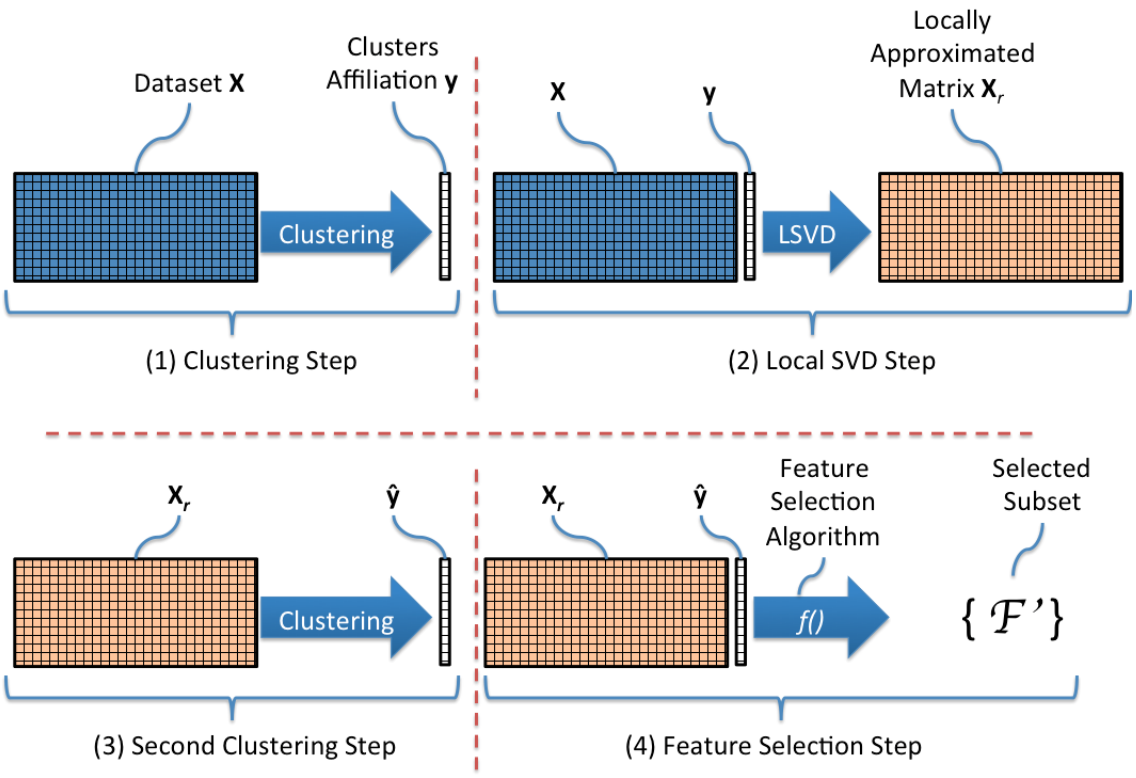


Figure 6.2: The proposed framework for stable feature selection for clustering.

where $g(\cdot)$ is the approximation method, which is SVD in this work and $h(\cdot)$ is *k-means*.

Assume that these n data points can be assigned to k clusters and $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$ is the cluster set. We use $\mathbf{y} \in \mathbb{R}^n$ to represent clustering affiliation of data points in $\mathbf{X}$ where $\mathbf{y}(i) = j$ if the i-th data point $\mathbf{x}_i$ belongs to the j-th cluster c_j. Let $\mathbf{X}_j \in \mathbb{R}^{n_j \times m} (1 \leq j \leq k)$ be the matrix including data points from the j-th cluster c_j where n_j is the number of data points in c_j. In this paper, we assume that each data point only belongs to one cluster thus $\sum_{j=1}^{k} n_j = n$.

The Singular Value Decomposition SVD of the matrix $\mathbf{X}$ is given by:

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^\top \tag{6.5}$$

where columns of $\mathbf{U}$ and $\mathbf{V}$ contain the left-singular vectors and right-singular vectors of $\mathbf{X}$ respectively, and Σ is a diagonal matrix and the diagonal elements are singular values of $\mathbf{X}$.

While the truncated SVD with rank $r \leq \min(n, m)$ is:

$$\hat{\mathbf{X}} \approx \mathbf{U}_r \Sigma_r \mathbf{V}_r^\top \tag{6.6}$$

where $rank(\hat{\mathbf{X}}) = r$. $\mathbf{U}_r$ and $\mathbf{V}_r$ contain the first r columns of $\mathbf{U}$ and $\mathbf{V}$ respectively, while Σ_r contains the first r columns and r rows of Σ.

The goal in data clustering is to group similar samples into one cluster while dissimilar samples in different clusters. In other words, the within-cluster similarity is maximized while minimizing the between-clusters similarity. It is very helpful to keep this notion in mind when we preprocess and prepare dataset $\mathbf{X}$ for further learning tasks, say feature selection. It is more meaningful to treat different clusters separately. In this work, the ultimate goal is to select stable features with the absence of class label, i.e. stable feature selection for clustering. Our proposed framework begins by generating the cluster affiliation $\mathbf{y}$ for each sample using *k-means*. Then, similar to [75], we approximate each cluster separately using SVD. We call this step Local Singular Value Decomposition (LSVD). Next, we cluster the new approximated matrix $\hat{\mathbf{X}}_r$ using *k-means* one more time to obtain a new clustering that will be used as the label in the feature selection step. Figure 6.2 illustrates the proposed framework. Using LSVD performs noise reduction on each cluster separately due to keeping the largest singular values while eliminating small ones [5, 69].

In LSVD we compute the low rank approximation for each cluster matrix, i.e., $\mathbf{X}_j (1 \leq j \leq m)$ and we further assume that the rank for the j-th cluster matrix $\mathbf{X}_j$ is r_j where r_j should be less than or equal to $\min(n_j, m)$. In this work, we will make $r_1 = r_2 = \ldots = r_k$ to avoid potential over-fitting. Then we denote the r_j low

rank approximate of the j-th cluster matrix $\mathbf{X}_j$ as,

$$\mathbf{X}_j \approx \mathbf{U}_j \Sigma_j \mathbf{V}_j^\top \tag{6.7}$$

where $\mathbf{U}_j$ and $\mathbf{V}_j$ include the first r_j left-singular vectors and right-singular vectors of $\mathbf{X}_j$ respectively, and Σ_j is a diagonal matrix that contains the first r_j singular values of $\mathbf{X}_j$.

After approximating each cluster, we can recombine the k approximated clusters to form the new approximated matrix $\mathbf{X}$. To distinguish between the original matrix $\mathbf{X}$ and its truncated approximation with rank $r = \sum_{j=1}^{k} r_j$, we call the latter $\hat{\mathbf{X}}_r$. The LSVD can be algebraically formulated as follows:

$$\hat{\mathbf{X}}_r = \hat{\mathbf{U}}_r \hat{\Sigma}_r \hat{\mathbf{V}}_r^\top, \tag{6.8}$$

where $\hat{\mathbf{U}}_r$, $\hat{\Sigma}_r$ and $\hat{\mathbf{V}}_r$ are defined as,

$$\hat{\mathbf{U}}_r = \begin{bmatrix} \mathbf{U}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{U}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{U}_k \end{bmatrix} \tag{6.9}$$

$$\hat{\Sigma}_r = \begin{bmatrix} \Sigma_1 & 0 & \cdots & 0 \\ 0 & \Sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_k \end{bmatrix} \tag{6.10}$$

$$\hat{\mathbf{V}}_r = [\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_k] \tag{6.11}$$

6.5 Experiment

In order to verify the effectiveness of the proposed framework, an extensive experiment was conducted using four microarray datasets and five well-known feature selection methods. The statistics of the datasets are shown in Table 6.1. Originally, these datasets have binary class, $\mathbf{y} = \{1, 2\}$. Although we don't consider these classes in our experiment since we are tackling a clustering problem, we consider this to be domain knowledge about the given datasets only for the evaluation purpose. Based on this knowledge, we know that there are at least two clusters in the dataset, which facilitates the quantification of the clusters. Therefore we fix k=*2* in this work.

Table 6.1: Datasets statistics

	Dataset Name	#Samples n	Dimensionality m
1	BLOOD	89	2759
2	SMK-CAN	187	19993
3	Colon	62	2000
4	Leukemia	72	12582

We choose five well-known feature selection methods: Fisher Score [21], Information Gain [14], Chi Square [86], ReliefF [86] and ℓ_1SVM [11]. The purpose of this evaluation is to demonstrate that our proposed framework can improve the stability of these chosen feature selection methods while maintain their performance.

The Evaluation: We apply the same feature selection algorithm several times, say l times, on $l-$sub-sampling of the data. We used Cross-Validation (CV) to generate the l-fold in this paper. The stability, then, will be the average pairwise similarity of the selected subsets. The more similar these subsets are, the more stable the algorithm is. There are different measurements to evaluate the stability. For more

about stability evaluation we refer the reader to [47]. In this work we use *Jaccard Index* ($J(\cdot)$) to evaluate the overlap between the selected subsets. We generate l-folds of $\mathbf{X}$ using cross-validation, where l=*10*. Then, l-1 folds are used as training set 1 fold is kept for validation. $f(\cdot)$ selects l subsets of features from each training set. We denote the l selected subsets of features $\mathbf{F} = \{\mathcal{F}'_1, \mathcal{F}'_2, \ldots, \mathcal{F}'_l\}$.

$$J(\mathcal{F}'_i, \mathcal{F}'_j) = \frac{|\mathcal{F}'_i \cap \mathcal{F}'_j|}{|\mathcal{F}'_i \cup \mathcal{F}'_j|}$$

$$S(\mathbf{F}) = \frac{2}{l(l-1)} \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} J(\mathcal{F}'_i, \mathcal{F}'_j)$$

$S(\cdot)$ is the average pairwise Jaccard Index which is the stability of the selection. In addition to the stability, the clustering quality is very important aspect. In fact, high stability without reasonable clustering ability is not desired. Therefore, we evaluate the ability of the selected feature subsets to cluster unseen samples. The overall goal of the proposed framework is to improve stability while maintaining the clustering quality. To ensure a fair comparison, $\mathbf{X}$ and $\mathbf{y}$ were only utilized in the test stage. In other words, the approximated matrix $\hat{\mathbf{X}}_r$ and the second clustering $\hat{\mathbf{y}}$ used only in the feature selection stage. This is due to the fact that the baseline methods has neither $\hat{\mathbf{X}}_r$ nor $\hat{\mathbf{y}}$.

The Baseline Methods: we compare our proposed framework against the existing feature selection for clustering approach. Note that there are an iterative approaches to do the generated clusters and feature selection in an EM manner. However we do not consider them in this work since the stability of these methods is not well-defined in the iteration process. Thus, we cannot fairly compare our method to these

approaches. Also, iterative *k-means* is mostly randomized approach [10] which leads to inconsistent clustering over iterations, hence, violating the essence of the stability.

Parameter Selection: There are several parameters that need to be selected in this experiment. First, the number of selected features is still an open problem.In the domain of genetic analysis (i.e. microarray), although the total number of features is relatively large, the relevant features to certain hypothesis is usually small. Based on preliminary experiment, we fixed the number of selected features to be 100 across all the datasets.

Similar to the number of selected features, the matrix rank $r = rank(\mathbf{X})$ is another open problem. Keeping the largest singular values preserves most of the information of the matrix while removing only a few of the very small singular values generates an approximated matrix that is very close to the original one. Therefore, the rank should be chosen carefully. To avoid of the rank selection problem, we select rank that ranges from 2 to 40 in all datasets except Colon Cancer dataset where the maximum rank is 32. We approximated each cluster separately. Hence, the rank r_1 and r_2 for each cluster $\mathbf{X}_1$ and $\mathbf{X}_2$ will be ranging from 1 to 20. Accordingly, $rank(\mathbf{X}) = r_1 + r_2$.

Finally, the cluster quantification, k, is selected based on the assumption of the domain knowledge since each dataset has originally two classes. Although we do not consider the original classes neither in the training nor the testing stages, we treat this as domain knowledge to choose the desired number of clusters. In this experiment, we fix *k=2*. This is owing to the fact that we know that there exists at least two clusters in each dataset.

Results

It is important to state that this experiment aims to show that the proposed method improves stability over the baseline methods whilst maintains the clustering performance. Figure 6.3 to Figure 6.7 shows the results of the experiment where each row of plots is an algorithm and each column is a dataset. Due to page constraints, we omitted the results of Chi Square and Information gain since they are very similar to the ones we show. The x-axis is the matrix rank r and the y-axis is the stability and the clustering accuracy. The blue asterisked-line and the red plain-line represent the stability of the proposed framework (LSVD) and the stability of the baseline method respectively. On the other hand, the black circled-line and the black plain-line are the accuracy of LSVD and baseline respectively.

In terms of clustering performance, we can see that proposed framework is able to maintain the high accuracy of the traditional method or in most cases can outperform it. The accuracy is not directly impacted by the matrix approximation because we test the accuracy using the original dataset. However, it is indirectly impacted due to the fact that the selection (i.e. the features were used in the test phase) was done over the approximated matrix. In a few cases, as in Figure6.6 ReliefF with Leukemia, for example, LSVD accuracy was less than the baseline accuracy. This is noticed a few times especially when the matrix rank is very small; namely $r \leq 4$. Yet, the proposed method outperforms the baseline method in most cases.

On the other hand, the improvement of the stability can be clearly observed in Figures (6.3 to 6.7). Particularly, we noticed the trend of the stability curve in almost all cases. The stability is higher when the rank is small. This finding

is reasonable owing to the fact that keeping only the largest singular values would capture most of the useful information in the original dataset and would reduce the data noise associated to the very small singular values. In most cases, the stability improves significantly when $r \leq 10$. The improvement rate is usually more than 50%. For example, Figure 6.3 with BLOOD dataset, the baseline stability is around 0.5 while stability of our method with r=*2* is 1. This huge gain is consistent across all the results.

In addition, we found the stability beyond $r = 20$ to settle around the stability of the baseline method, which understandable since the approximated matrix became closer to the full-ranked matrix $\mathbf{X}$. With respect to each algorithm, ℓ_1SVM surpasses the baseline method in all cases regardless of the value of r, yet, it is still generally true that the smaller the r, the better the stability, Figure 6.3. Comparing to other algorithms, ℓ_1SVM was mostly better in the stability gain even with very large rank such as: $r = 40$. The remaining algorithms were comparable. We believe that the reason behind the superior stability of ℓ_1SVM is the intrinsic of the ℓ_1SVM itself. It basically aims to select features that are able to maximize the decision boundary between the two clusters using only a few samples; namely: support vectors. Although these samples could differ with respect the folds, the sampling technique used in our experiment has the ability to capture the underling distribution of the data which is mainly characterized by the same or very similar subset of features.

To summarize the finding, our proposed method was very effective in terms of both clustering performance and stable feature selection, Figure 6.3 to Figure 6.7. These results indicate the significant contribution made to the feature selection for clustering. In the discussion, we will explain other potentially interesting approaches and their results.

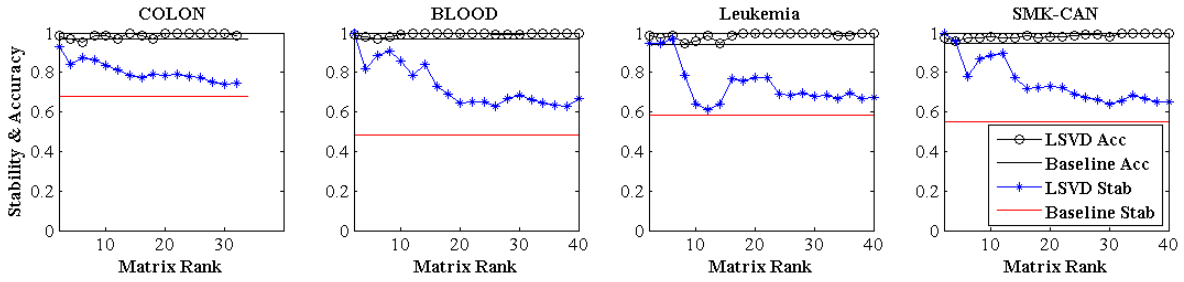


Figure 6.3: ℓ_1 SVM

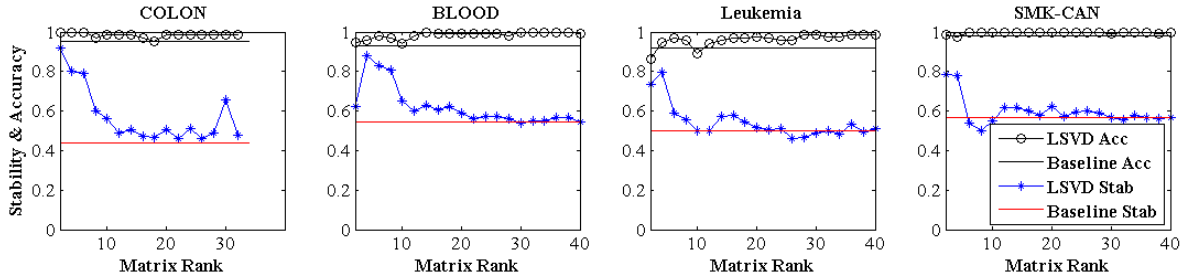


Figure 6.4: Chi Square

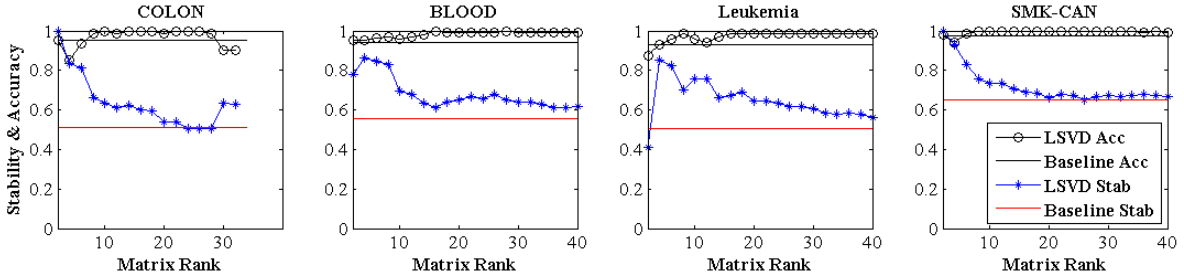


Figure 6.5: Fisher Score

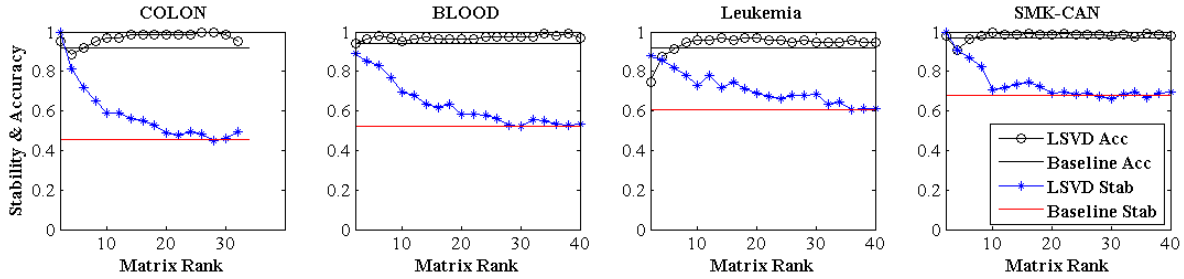


Figure 6.6: ReliefF

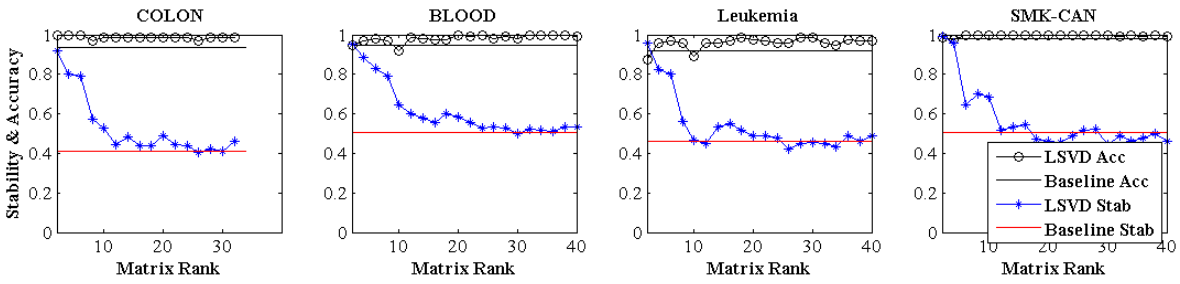


Figure 6.7: Information Gain

6.6 Discussion and Feature Directions

Anyone may argue that this framework could have been proposed differently. In fact, that is totally true. We have tried different frameworks, yet, this one worked the best. One possible framework could have been just ignoring the second clustering (i.e. step (3) shown in Figure 6.2). Another way is to consider the iterative approximation and clustering instead of just one iteration [1]. We evaluated both approaches but neither one worked properly. The second clustering, which was done using *k-means* over the approximated matrix, was necessary to guide the feature selection. It was meaningless to select features from approximated matrix while the selection guide is extracted from different matrix especially when we know that this was local approximation. We found that the stability in the iterative clustering, on the other hand, did not gain much after the second clustering (step (3)). This could be due to inconsistency of *k-means* from one iteration to another.

Future directions include investigating subspace clustering stability. In addition, It is important to study and customize measurements to evaluate selection stability for clustering. Stability measurements for clustering may consider the po-

[1] It is important to distinguish between this iterative approach and the other iterative approach we mentioned the baseline method.

tential existence of the unknown patterns in the data. This means, the evaluation is not independent of the dataset as it is the case in existing measures including *Jaccard Index*. In addition, it is important to define the stability with respect to the utilized approach such as the iterative one.

6.7 Summary

In this work, we investigated the stability of feature selection for clustering. We proposed a framework based on local low rank matrix approximation using SVD to improve selection stability while maintaining clustering performance. The empirical results demonstrated the effectiveness of the proposed method. We found that when the matrix rank is smaller, the stability tends to be better and the accuracy mostly maintained or improved over the traditional approach. The maximum stability gain reaches 100% mostly when $r=2$ where it improved in some cases from 0.5 to 1. Also, we found that when r is large enough, the stability does not gain much. Thus, to insure stability gain, r should be small.

Chapter 7

CONCLUSION

In this dissertation, I studied the stability of feature selection algorithms from data perspective. Existing work tackles the stability from algorithm viewpoint. In order to study the stability, we need first to evaluate the stability. We found that current evaluation approach does not take data variance into consideration, thus, we proposed a new approach that evaluates data similarity to perceive the relation between data variation and selection stability. This awareness of the relationship gives us the ability to fairly judge the stability of the algorithm.

In addition, we found the stability to be data-dependent. There are several factors that were found to influence the selection stability. For example, data dimensionality, sample size, data noise, and so forth. Accordingly, these factors are data dependent. Therefore, resolving the stability issue should begin by curing the dataset itself.

We proposed a framework to reduce data noise before feature selection step. It is known that reducing the matrix rank by decomposing the matrix while removing the smallest singular values helps reducing data noise level. This reduces the variation between samples. Though, current approaches does not consider class label which leads to reducing variation even between samples from different classes. This is against the notion of the feature selection where we need to preserve the between-classes variance at maximum possible level. We introduced SLRMA to reduce matrix rank for each class independently. The empirical results demonstrated the effectiveness this framework in three aspects. First, SLRMA was able to select very stable feature subsets comparing to baseline methods. Second, the selected subsets were

able to achieve even higher classification accuracy than baseline methods. Third, it demonstrated high precision in selecting relevant features. Therefore, we known concluded that the proposed framework was very effective in terms of preparing the data for the feature selection task.

SLRMA, however, cannot handle unlabeled data. To overcome this limitation, I proposed a framework that involves data clustering the applies local SVD on each cluster independently. This framework called LSVD. Similar to SLRMA, LSVD shows superiority in terms of stability and clustering accuracy. This is, to best of our knowledge, is the first work that tackles the stability of feature selection for clustering.

Discussion and Future Directions

With respect to the proposed methods, SLRMA and LSVD, theoretical justification is necessary to explain why they in fact work. Also, these frameworks cannot handle sparse datasets. They transform them into dense matrices. Thus, sparsefication step is needed to maintain the sparsity level in the low ranked matrices.

There are still several approaches still need to be investigated in depth. For example, the effect of different sampling strategies may vary in terms of stable results. Leave-One-Out (LOO) sampling technique most likely produces more stable selected subsets than Cross-Validation (CV), as the permutation level is larger in the latter while the sample size is smaller. Another approach that requires more attention is sample weighting which might be affective in terms of assigning samples from desired regions more weight than those in undesired regions. Theoretically, it leads to less variation in the training data.

As I mentioned earlier, I investigated the stability from data viewpoint. However, we can study the stability from slightly different perspective. For example,

trading off between stability and accuracy. Although we believe there is know correlation between them, it is possible to trade off after we acquire the selected subset by replacing less frequent features with high frequent ones while the accuracy is not greatly affected.

[1] 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*, 26(3):392–398, Feb 2010.

[2] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 611–618. ACM, 2001.

[3] S. Alelyani, H. Liu, and L. Wang. The effect of the characteristics of the dataset on the selection stability. In *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, pages 970–977. IEEE, 2011.

[4] S. Alelyani, J. Tang, and H. Liu. Feature selection for clustering: A review. 2013.

[5] O. Alter, P. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97(18):10101, 2000.

[6] L. Belanche and F. González. Review and evaluation of feature selection algorithms in synthetic problems. *arXiv preprint arXiv:1101.2320*, 2011.

[7] J. Benesty, J. Chen, Y. Huang, and I. Cohen. *Noise reduction in speech processing*, volume 2. Springer Verlag, 2009.

[8] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and information systems*, pages 1–37, 2012.

[9] A.-L. Boulesteix and M. Slawski. Stability and aggregation of ranked gene lists. *Brief Bioinform*, 10(5):556–568, 2009.

[10] C. Boutsidis, M. Mahoney, and P. Drineas. Unsupervised feature selection for the k-means clustering problem. *Advances in Neural Information Processing Systems*, 22:153–161, 2009.

[11] P. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. pages 82–90. Morgan Kaufmann, 1998.

[12] D. Cai, C. Zhang, and X. He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 333–342. ACM, 2010.

[13] L.-Y. Chuang, C.-H. Yang, K.-C. Wu, and C.-H. Yang. A hybrid feature selection method for dna microarray data. *Computers in biology and medicine*, 41(4):228–237, 2011.

[14] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, 1991.

[15] M. D'Alessandro, R. Esteller, G. Vachtsevanos, A. Hinson, J. Echauz, and B. Litt. Epileptic seizure prediction using hybrid feature selection over multiple intracranial eeg electrode contacts: a report of four patients. *Biomedical Engineering, IEEE Transactions on*, 50(5):603–615, 2003.

[16] S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 74–81, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[17] M. Dash, K. Choi, P. Scheuermann, and H. Liu. Feature selection for clustering - a filter solution. In *In Proceedings of the Second International Conference on Data Mining*, pages 115–122, 2002.

[18] M. Dash and H. Liu. Feature selection for clustering. pages 110–121. Springer-Verlag, 2000.

[19] C. A. Davis, F. Gerick, V. Hintermair, C. C. Friedel, K. Fundel, R. Kfner, and R. Zimmer. Reliable gene signatures for microarray classification: assessment of stability and performance. *Bioinformatics*, 22(19):2356–2363, Oct 2006.

[20] K. Devarajan. Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. *PLoS computational biology*, 4(7):e1000029, 2008.

[21] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2 edition, 2001.

[22] K. Dunne, P. Cunningham, and F. Azuaje. Solutions to instability problems with sequential wrapper-based approaches to feature selection. Technical Report TCD-CD-2002-28, Department of Computer Science, Trinity College, Dublin, Ireland, 2002.

[23] J. Dy. Unsupervised feature selection. *Computational Methods of Feature Selection*, pages 19–39, 2008.

[24] J. G. Dy and C. E. Brodley. Feature subset selection and order identification for unsupervised learning. In *In Proc. 17th International Conf. on Machine Learning*, pages 247–254. Morgan Kaufmann, 2000.

[25] J. G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *J. Mach. Learn. Res.*, 5:845–889, 2004.

[26] G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

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

[28] G. Gulgezen, Z. Cataltepe, and L. Yu. Stable and accurate feature selection. In *ECML/PKDD (1)*, pages 455–468, 2009.

[29] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

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

[31] M. A. Hall. Correlation-based feature selection for machine learning. Technical report, 1999.

[32] Y. Han and L. Yu. A Variance Reduction Framework for Stable Feature Selection. In *2010 IEEE International Conference on Data Mining*, pages 206–215. IEEE, 2010.

[33] A. Haury, P. Gestraud, and J. Vert. The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. *PloS one*, 6(12):e28210, 2011.

[34] A. Haury, L. Jacob, and J. Vert. Improving stability and interpretability of gene expression signatures. *Arxiv preprint arXiv:1001.3109*, 2010.

[35] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. *Advances in Neural Information Processing Systems*, 18:507, 2006.

[36] Z. He and W. Yu. Stable feature selection for biomarker discovery, 2010.

[37] J. Huang, M. Ng, H. Rong, and Z. Li. Automated variable weighting in k-means type clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(5):657–668, 2005.

[38] L. Huawei, W. Xindong, and Z. Shichao. Feature selection using hierarchical feature clustering. 2011.

[39] L. Jacob, G. Obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 433–440. ACM, 2009.

[40] A. Jain and D. Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:153–158, 1997.

[41] A. Jain and D. Zongker. Feature selection: Evaluation, application, and small sample performance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(2):153–158, 1997.

[42] L. Jing, M. Ng, and J. Huang. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *Knowledge and Data Engineering, IEEE Transactions on*, 19(8):1026–1041, 2007.

[43] T. Joachims, F. Informatik, F. Informatik, F. Informatik, F. Informatik, and L. Viii. Text categorization with support vector machines: Learning with many relevant features, 1997.

[44] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. pages 121–129. Morgan Kaufmann, 1994.

[45] G. Jurman, S. Merler, A. Barla, S. Paoli, A. Galea, and C. Furlanello. Algebraic stability indicators for ranked lists in molecular profiling. *Bioinformatics*, 24(2):258–264, Jan 2008.

[46] A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms. page 8 pp., nov. 2005.

[47] A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and Information Systems*, 12(1):95–116, May 2007.

[48] Y. Kim, W. Street, and F. Menczer. Evolutionary model selection in unsupervised learning. *Intelligent Data Analysis*, 6(6):531–556, 2002.

[49] L. Klebanov and A. Yakovlev. How high is the level of technical noise in microarray data. *Biol Direct*, 2(9), 2007.

[50] R. Kohavi and G. H. John. Wrappers for feature subset selection, 1996.

[51] L. I. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th conference on Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications*, pages 390–395, Anaheim, CA, USA, 2007. ACTA Press.

[52] J. Lampinen, J. Laaksonen, and E. Oja. Pattern recognition. In C. Leondes, editor, *Image Processing and Pattern Recognition*, volume 5 of *Neural Network Systems Techniques and Applications*, pages 1 – 59. Academic Press, 1998.

[53] Y. Y. Leung, C. Q. Chang, Y. S. Hung, and P. C. W. Fung. Gene selection for brain cancer classification. *Conf Proc IEEE Eng Med Biol Soc*, 1:5846–5849, 2006.

[54] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu. Unsupervised feature selection using nonnegative spectral analysis. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[55] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Boston: Kluwer Academic Publishers, 1998.

[56] H. Liu and H. Motoda, editors. *Computational Methods of Feature Selection*. Chapman and Hall/CRC Press, 2007.

[57] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In J. Vassilopoulos, editor, *Proceedings of the Seventh IEEE International Conference on Tools with Artificial Intelligence, November 5-8, 1995*, pages 388–391, Herndon, Virginia, 1995. IEEE Computer Society.

[58] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 339–348. AUAI Press, 2009.

[59] S. Loscalzo, L. Yu, and C. Ding. Consensus group stable feature selection. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 567–576, New York, NY, USA, 2009. ACM.

[60] L. Meier, S. Van De Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

[61] N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

[62] P. Mitra, S. Member, C. A. Murthy, and S. K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:301–312, 2002.

[63] F. Model, P. Adorjan, A. Olek, and C. Piepenbrock. Feature selection for dna methylation based cancer classification. *Bioinformatics*, 17 Suppl 1:S157–S164, 2001.

[64] D. Modha and W. Spangler. Feature weighting in k-means clustering. *Machine learning*, 52(3):217–237, 2003.

[65] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. In *Machine Learning*, pages 103–134, 1999.

[66] O. Nih. Getting the noise out of gene arrays. 2004.

[67] I.-S. Oh, J.-S. Lee, and B.-R. Moon. Hybrid genetic algorithms for feature selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(11):1424–1437, 2004.

[68] P. Pavel Krek, J. Kittler, and V. Hlav. *Computer Analysis of Images and Patterns*, chapter Improving Stability of Feature Selection Methods, pages 929–936. Springer Berlin / Heidelberg, 2007.

[69] B. Pilgram, W. Schappacher, and G. Pftirtscheller. A noise reduction method using singular value decomposition. In *Engineering in Medicine and Biology Society, 1992 14th Annual International Conference of the IEEE*, volume 6, pages 2756–2758. IEEE, 1992.

[70] R. G.-B. Ranb, A. Navot, and N. Tishby. Margin based feature selection - theory and algorithms. In *In International Conference on Machine Learning (ICML*, pages 43–50. ACM Press, 2004.

[71] V. Roth and T. Lange. Feature selection in clustering problems. *Advances in neural information processing systems*, 16, 2003.

[72] Y. Rui and T. S. Huang. Image retrieval: Current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10:39–62, 1999.

[73] Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques, 2008.

[74] Y. Saeys, I. Inza, and P. Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, Oct 2007.

[75] B. Savas, I. Dhillon, et al. Clustered low rank approximation of graphs in information science applications. In *Proceedings of the 2011 SIAM Conference on Data Mining*, 2011.

[76] K. Shin, J. Hammond, and P. White. Iterative svd method for noise reduction of low-dimensional chaotic time series. *Mechanical Systems and Signal Processing*, 13(1):115–124, 1999.

[77] W. Siedlecki and J. Sklansky. On automatic feature selection. pages 63–87, 1993.

[78] P. Somol and J. Novovivcova. Evaluating the stability of feature selectors that optimize feature subset cardinality, 2010.

[79] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *In 20th International Conference on Machine Learning*, volume 20, page 720. AAAI Press, 2003.

[80] L. Talavera. Feature selection as a preprocessing step for hierarchical clustering. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 389–397. MORGAN KAUFMANN PUBLISHERS, INC., 1999.

[81] K. Thangavel and A. Pethalakshmi. Dimensionality reduction based on rough set theory: A review. *Applied Soft Computing*, 9(1):1–12, 2009.

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

[83] C. Tsai and C. Chiu. Developing a feature weight self-adjustment mechanism for a k-means clustering algorithm. *Computational statistics & data analysis*, 52(10):4658–4672, 2008.

[84] L. Van der Maaten, E. Postma, and H. Van Den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:1–41, 2009.

[85] T. Way, B. Sahiner, L. Hadjiiski, and H. Chan. Effect of finite sample size on feature selection and classification: A simulation study. *Medical physics*, 37:907, 2010.

[86] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.

[87] J. Xie, W. Xie, C. Wang, and X. Gao. A novel hybrid feature selection method based on ifsffs and svm for the diagnosis of erythemato-squamous diseases. In *JMLR Workshop and Conference Proceedings. Workshop on Applications of Pattern Analysis*, volume 11, pages 142–151, 2010.

[88] F. Yang and K. Mao. Improving robustness of gene ranking by resampling and permutation based score correction and normalization. In *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, pages 444–449. IEEE.

[89] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. pages 412–420. Morgan Kaufmann Publishers, 1997.

[90] J. Ye and J. Liu. Sparse methods for biomedical data. *SIGKDD Explor. Newsl.*, 14(1):4–15, Dec. 2012.

[91] L. Yu, C. Ding, and S. Loscalzo. Stable feature selection via dense feature groups. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 803–811, New York, NY, USA, 2008. ACM.

[92] L. Yu, Y. Han, and M. E. Berens. Stable gene selection from microarray data via sample weighting. 9(1):262–272, 2012.

[93] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In T. Fawcett and N. Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning (ICML-03),*, pages 856–863, Washington, D.C., August 21-24, 2003 2003. Morgan Kaufmann.

[94] L. Yuan, J. Liu, and J. Ye. Efficient methods for overlapping group lasso. *Adv. Neural Inform. Process. Syst*, 2011.

[95] M. Zhang, L. Zhang, J. Zou, C. Yao, H. Xiao, Q. Liu, J. Wang, D. Wang, C. Wang, and Z. Guo. Evaluating reproducibility of differential expression discoveries in microarray studies by considering correlated molecular changes. *Bioinformatics*, 25(13):1662–1668, Jul 2009.

[96] Z. Zhao and H. Liu. Semi-supervised feature selection via spectral analysis. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2007.

[97] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1151–1157, New York, NY, USA, 2007. ACM.

[98] Z. Zhao and H. Liu. *Spectral Feature Selection for Data Mining*. Chapman & Hall/Crc Data Mining and Knowledge Discovery. Taylor & Francis, 2011.

[99] J. Zhou, J. Liu, V. A. Narayan, and J. Ye. Modeling disease progression via fused sparse group lasso. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1095–1103. ACM, 2012.

[100] M. Zucknick, S. Richardson, and E. A. Stronach. Comparing the characteristics of gene expression profiles derived by univariate and multivariate classification methods. *Stat Appl Genet Mol Biol*, 8:7, Biol 2008.