Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

DECISION TREE RULE-BASED FEATURE SELECTION FOR IMBALANCED DATA

by
Haoyue Liu

A class imbalance problem appears in many real world applications, e.g., fault diagnosis, text categorization and fraud detection. When dealing with an imbalanced dataset, feature selection becomes an important issue. To address it, this work proposes a feature selection method that is based on a decision tree rule and weighted Gini index. The effectiveness of the proposed methods is verified by classifying a dataset from Santander Bank and two datasets from UCI machine learning repository. The results show that our methods can achieve higher Area Under the Curve (AUC) and F-measure. We also compare them with filter-based feature selection approaches, i.e., Chi-Square and F-statistic. The results show that they outperform them but need slightly more computational efforts.

DECISION TREE RULE-BASED FEATURE SELECTION FOR IMBALANCED DATA

by
Haoyue Liu

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Engineering

Helen and John C. Hartmann
Department of Electrical and Computer Engineering

May 2017

APPROVAL PAGE

DECISION TREE RULE-BASED FEATURE SELECTION FOR IMBALANCED DATA

Haoyue Liu

Dr. Mengchu Zhou, Thesis Advisor — Date
Distinguished Professor of Electrical and Computer Engineering, NJIT

Dr. Osvaldo Simeone, Committee Member — Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Yunqing Shi, Committee Member — Date
Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Haoyue Liu

Degree: Master of Science

Date: May, 2017

Undergraduate and Graduate Education:

- Master of Science in Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2017

- Bachelor of Science in Automation,
  Kunming University of Science and Technology, Kunming, P. R. China, 2014

Major: Computer Engineering

Presentations and Publications:

Liu, H.Y., and Zhou, M.C., "Decision tree rule-based feature selection for large-scale imbalanced data," *The 27th Wireless and Optical Communication Conference (WOCC)*, New Jersey, USA, April 2017.

Dedicated to my family, all inclusive, known and unknown –for giving birth to me at the first place and supporting me spiritually throughout my life.

ACKNOWLEDGMENT

Foremost, I would like to express my deepest gratitude to my advisor, Dr.Mengchu Zhou, for excellent guidance and patience. Professor Zhou served as my research advisor, but he was very influential in the academic path choice I have made and gave me many excellent suggestions.

I would also like to thank Dr. Osvaldo Simeone for sharing with me his deep knowledge of machine leaning and Dr. Yunqing Shi for his advice on my research.

My sincere thanks also go to my group members, Xiaoyu Lu and Liang Qi, and my good friends, Yanfei Gao and Keyuan Wu, for giving me many excellent suggestions and assisting me in completing this thesis work.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 High-dimensional Data

As the rapid development of technology and science in the recent decades, the size of data grows explosively, which has brought some unprecedented challenges to machine learning researches. One of challenges is relevant to high-dimensional data, which makes a learning task more complex and time-consuming. The characteristic of high-dimensional data is the number of features is very high, e.g., Microarray dataset has 26,000 features.

The problem of high-dimensional data has influenced a broad range of areas such as imaging processing, DNA microarray data analysis, cancer detection, bioinformatics and text data mining.

In order to deal with high-dimensional data, machine learning methods are confronted with the so-called "curse of dimensionality" problem [Donoho, 2000]. This problem can easily arise if the number of features is higher than the number of instances. In these cases, when the instances are scarce, it makes even more difficult for machine learning algorithms to obtain acceptable performance in their tasks. Besides the well-known problems as caused by the curse of dimensionality, another common issue is the computational cost. Most of existing machine learning algorithms require prohibitively many computing resources and/or huge data memory, thereby making their use infeasible.

Except previous two frequent issues that happen, when high-dimensional data is processed, another problematic issue is the presence of irrelevant or redundant features. If all the irrelevant or redundant features are used in machine learning algorithms, harmful consequences in terms of performance and computational cost can result. Moreover, huge memory or storage space is needed to handle a high-dimensional dataset.

1.2 Class Imbalance

Another challenge for recent machine learning researchers is a class imbalance problem. It arises when the numbers of observations among classes are significantly different. Real-world applications, such as fraud detection, software prediction, diagnosis of cancers and oil spill detection, can inherently result in class imbalancing issues [Wang *et al*., 2017]

The major issue about the imbalanced data learning problem is how to choose a significantly compromised method to handle the bias-to-majority problem from which most traditional machine learning algorithms suffer, when dealing with imbalanced data. Because traditional leaning algorithms commonly assume that class distributions are balanced and/or misclassification costs are equal among different classes. Such assumptions cause the classification ability of learning algorithms towards the majority classes, because algorithms try to optimize overall accuracy, which is overwhelmed by majority classes while ignoring minority classes. The difficulty of dealing with imbalanced problems comes from its imbalance rate, i.e., the imbalance ratio that is equal to the number of instances in the majority class divided by the number of instances in the minority class.

The drawback of using a conventional learning algorithm to classify an imbalanced dataset has met many thorny issues in many existing applications. For instances, in the cancer diagnosis field, this issue is particularly crucial. The minority class presents patients who have cancer and the majority class presents patients who do not. In the reality of this filed, the size of minority data is very much smaller than the size of majority data. However, the purpose of this application is to discover useful knowledge to make important decisions about which patients have cancer. At that time, if learning algorithms bias toward the majority class, it can be extremely costly to misclassify a minority instance.

1.3 Methods for Class Imbalance Problem

The methods proposed for dealing with imbalanced data include two aspects: data preprocessing and learning algorithms.

1.3.1 Sampling Methods

For the imbalanced data, one easy solution is to resample the data, i.e., by changing original class frequencies at a preprocessing step to balance the class distribution of training data [Chen & Wasikowski, 2008]. It can use under-sampling, over-sampling, or both.

Under-sampling is an efficient method for balancing data. Its main idea is to remove a subset of the majority observations from the training dataset to achieve the balance between imbalanced classes. One simplest way in the under-sampling is to randomly eliminate samples from the majority class until the data set gets balanced. In order to avoid any useful data eliminated during random under-sampling preprocessing

progress, some algorithms can guide one in selecting a subset of training data. They include clustering [Yen & Lee, 2009], and EasyEnsemble [Liu, 2009][Kang, *et al*, 2016]. When the volume of a dataset is extremely large, under-sampling methods can reduce computational time. However, in another aspect, if the volume of dataset is extremely small, each of instances is precious for learning algorithms. At that time, even eliminating one instance is highly risky.

Over-sampling is another sampling method to balance data by replicating or synthesizing the instances of minority class. Over-sampling is divided into two types: random over-sampling and algorithm over-sampling. The former is randomly replicating the minority class. The latter is based on some algorithms, such as Borderline-SMOTE [Han, *et al.*, 2005]. The advantage of over-sampling is that it avoids losing any useful information in training data. The disadvantage of this method is it may lead to over-fitting and cause additional computational cost.

1.3.2 Feature Selection

A dataset is characterized by the numbers of instances and features. Sampling methods change the data distribution in terms of instance counts. Feature selection can be considered as another method to solve an imbalanced class problem. When machine learning algorithms facing an imbalanced dataset, most of time they bias toward into the majority class. However, if a subset of features that are more related to minority class information is selected for training data, classification result may bias toward the minority class [Chawla, *et al*., 2004]. For instance, when a learning algorithm is used to solve an oil spill detection problem, it is more important to select a subset of features that can more accurately detect which oil is spilled. One advantage of using feature selection

is, when an imbalanced data has high dimensionality, reducing the size of features can drastically help reduce the computational time.

1.3.3 Cost-sensitive Learning

In the algorithmic aspect, cost-sensitive learning assigns different weights to different classes based on their importantance. One popular example for cost-sensitive learning is support vector machine (SVM) [Fumera & Roli, 2002]. Assume that a dataset is represented by a set of data points $J = \{(x_i, y_i)\}_{i=1}^{l}$ where $(x_i, y_i) \in R^{n+1}$, l and n represent the numbers of instances and features, respectively. $y_i \in \{+1, -1\}$ represents the positive and negative classes. In the cost-sensitive SVM, the costs for two classes are represented with C^+ minority and C^- majority. The formulation of separating a hyper-plane for the weighted SVM is given as follow:

$$\min \frac{1}{2} ||w||^2 + C^+ \sum_{\{i|y_i=+1\}}^{n^+} \xi_i + C^- \sum_{\{i|y_j=-1\}}^{n^-} \xi_j \tag{1.1}$$

$$s.t. y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i \qquad i = 1, \dots, l \tag{1.2}$$

$$\xi_i \geq 0 \qquad i = 1, \dots, l \tag{1.3}$$

where w is a vector, n^+ is the number of majority data, n^- is the number of minority data, ξ_i is the lack variable, y_i means the target label and $\phi(x_i)$ is the kernel function. Clearly $C^+ > C^-$ since correct identification of an instance in a minority class is more important than that in a majority class. C^+ and C^- are selected based on imbalanced ratio.

CHAPTER 2

REVIEW OF LITERATURES

2.1 Feature Selection

As the real-world data turns to be more complicated and diverse, data pre-processing plays a crucial role in machine learning and data mining. Feature selection is one of important topics in data pre-processing. It especially aims at improving the quality of data by riding noisy, irrelevant and redundant information for its further use. Kohavi and John [1997] classified features into three disjoint categories, strongly relevant, weakly relevant and irrelevant features. In their definition, relevance should be defined in terms of an optimal Bayes classifier. A strongly relevant feature X means that its removed results in performance deterioration of an optimal Bayes classifier. A weakly relevant feature X means that it is not strongly relevant and thus may or may not impact the classifier's performance in any significant way. A feature is irrelevant if it is neither strongly nor weakly relevant. Its use intends to worsen a classifier's performance. Feature redundancy is evaluated by calculating features' correlation. When two features' values are highly correlated, they are redundant to each other. However, it is hardly to determine if a feature is redundant if is correlated with a set of other features.

A common idea of reducing the dimensionality of data to be analyzed is to reduce the number of features to a more manageable number while not reducing the effectiveness of its study. Feature selection consists of reducing the features to an optimal or sub-optimal subset of them, which can be used to produce equal or better results than the original set. The main reasons to use feature selection are as follows:

1) Reduced computational time

As the data size dramatically increases in a big data era, many popular machine learning algorithms become time-consuming in the presence of a huge number of features. Reducing the feature set scales down the dimensionality of the data, which in turn reduces the training time required in many algorithms and thus computational cost. Especially for the situation of a learning algorithm combined with a genetic algorithm, using a "small" number of features, which are selected by using a feature selection method, reduces its computational load without degrading its performance [Cuadra, *et al*., 2008].

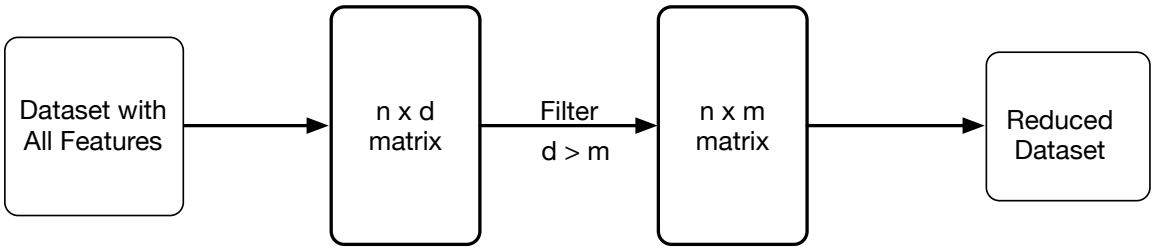
2) Improved accuracy

In many applications, feature selection can enhance prediction accuracy by reducing noise features or selecting a subset of relevant features. Even some state-of-art and sophisticated learning algorithms cannot achieve high prediction without getting rid of the irrelevant or weakly related features. However, once a good subset of features is selected, some relatively simple learning algorithms may produce desired prediction performance. Also, reducing the irrelevant features makes the data mining results easier to understand and more applicable [Guyon & Elisseeff, 2003]. While reducing the feature set may improve the performance of most classification algorithms, it may lower the accuracy of classifier based on decision trees [Li, *et al*., 2004]. Since decision trees have the capability of reducing the original feature set in a tree building process, beginning the process with fewer features may affect the final algorithm performance.

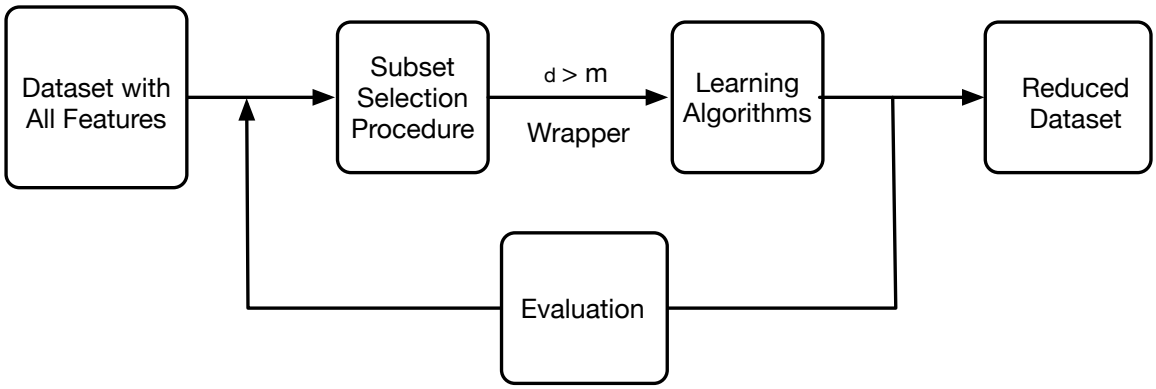
Feature selection is different from feature extraction (or feature transformation), which creates new features by combining the original features: Principal component

analysis (PCA) [Jolliffe, 1986], linear discriminant analysis (LDA) [Martinez & Kak, 2001], and locally linear embedding (LLE) [Roweis & Saul, 2000] are the examples of feature transformation techniques. On the other hand, feature selection maintains the original meanings of the selected features, which is highly desirable in many domains.

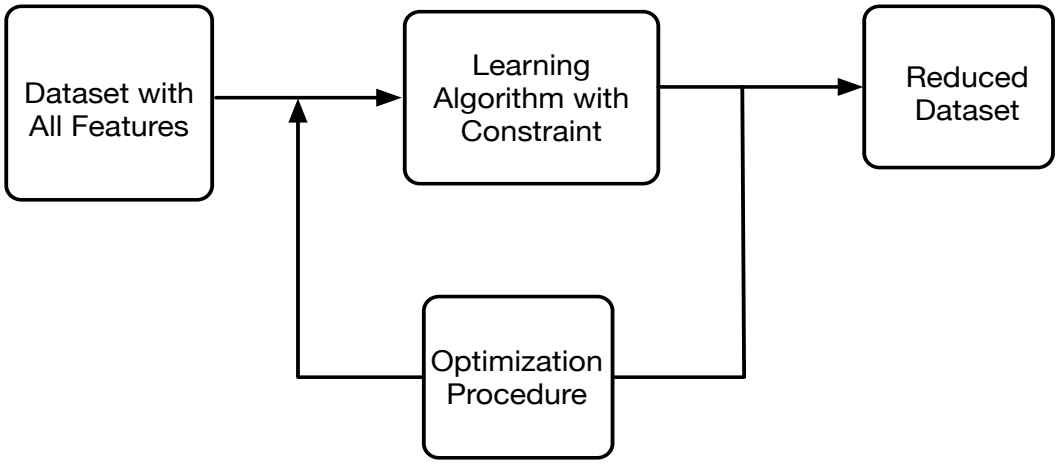
The approaches to feature selection fall into three categories: filter, wrapper and embedded approaches. Figure 2.1 shows graphically how these methods work on the a dataset to select a desired subset of features.



(a) Filter Methods



(b) Wrapper Methods



(c) Embedded Methods

Figure 2.1 A graphical view of how filter, wrapper and embedded methods work on a dataset.

2.1.1 Filter Methods

The filter methods for feature selection reduce the number of features by using the performance evaluation metric directly from the data, without feedback from predictors [John, *et al.*, 1994]. A suitable ranking criterion, such as the Pearson correlation coefficient and Chi Square, and a proper threshold of removing features are both important in this method. Zhang and Chen [2008] propose a filter method based on pairwise constraints, which does not have to access the whole training data and can save the computational time for large-size data sets. A correlation based feature selection method is proposed by [Hall, 2000]. The Laplacian score is used to reflect each feature's locality preserving power [He, *et al*., 2005].

The advantage for using filter methods in a data preprocessing step not only helps improve the performance of classification algorithms, but also reduces the amount of the computer processing time. The second advantage is that filter methods do not incorporate the final learning algorithm in their process [Ladha & Deepa, 2011]. The last but not lest benefit of a filter method is that the same features may be used in different learning algorithms for comparative analysis. According to [Hall & Smith, 1998], some filter methods, e.g., Correlation-based Feature Selection (CFS), may produce results that are similar to or better than wrapper methods in several domains. Yu and Liu [2003] propose a new correlation-based feature selection method. Their study shows the efficiency and effectiveness of such methods when dealing with high-dimension data sets. However, Saeys *et al*. [2007] note that filter methods have the disadvantage of not interacting with the classifier algorithm eventually used. Another disadvantage is that most filter methods are univariate in nature, meaning that they do not take into consideration the values of

other attributes. Their study is conducted on a high-dimension bioinformatics data set [Saeys *et al*., 2007].

2.1.2 Wrapper Methods

Unlike filter methods, wrapper methods use a preselected learning algorithm as a part of the feature selection process. They can be categorized into Sequential Selection and Heuristic Search Algorithms. The former only evaluates one feature each step, such as, Forward Sequential Selection (FSS) and Backward Sequential Selection (BSS). The latter evaluates different subsets to obtain the optimal evaluation results. As features are added or subtracted from a feature set, the final results are ranked in terms of effectiveness of the selection. Since the learning algorithm itself is used in the evaluation phase of the selection process, wrapper methods tend to score better results than filter methods. Kohavi and John [1997] compare the wrapper methods for feature subset selection against filter methods. They conclude that the relevancy of features contributes greatly to the performance of the learning algorithms when using a wrapper method as their feature selection method. However, wrapper methods have some limitations. The computational time of evaluating features in wrapper methods is far greater than in that of filter methods. Another disadvantage of wrapper methods is the likelihood of over-fitting the data. Instead of using a single wrapper method such as sequential forward selection, Gheyas and Smith [2010] propose a new method, called simulated annealing generic algorithm (SAGA), which incorporate the existing wrapper methods into a single solution. The results show that the combined methods can reduce the weaknesses each has inherently when used individually. Maldonado and Weber, [2009] propose a wrapper method based on the Support Vector Machine (SVM) classification. Their study concludes that using

such method can avoid over-fitting the data because of its capability of splitting the data. It also allows the use of different Kernel functions to provide better results. One drawback is that their proposed algorithm has used the backward elimination feature, which is computationally expensive when working with high-dimension data sets.

2.1.3 Embedded Methods

The main idea of an embedded method is to incorporate the feature selection as a part of a training process, which means that it relies on the classification. Therefore, the search for an optimal subset of features is built into the classifier construction and can be seen as a search in the combined space of feature subsets and hypotheses. For instance, Guyon, *et al*. [2002] present a recursive feature elimination method based on SVM. They achieve feature selection by iteratively training an SVM classifier with the current set of features and removing the least important feature indicated by the weights in the SVM solution. [Oh, *et al.*, 2002] introduce a Particle Swarm Optimization (PSO) and KNN embedded method for diagnosis of mammographic where PSO stands. According to [Yang, *et al*., 2013] the basic architecture of an embedded method consists of: (1) a search component, (2) a feature evaluation criterion, and (3) a learning algorithm. The advantage of embedded methods is their lower computational cost than wrapper methods. The disadvantage is that their feature selection performance highly depend on learning algorithms.

2.2 Decision Tree

A decision tree is one of the most commonly used learning algorithms in classification and regression. It has been widely used in the text classification [Apté, *et al*.,

1994][Forman, 2003], spam detection [Castillo, *et al*., 2007] and categorical loan data [Koh, 2004]. In a classification problem, a decision tree represents a classification's process that is based on features. Also, it can be viewed as a combination of a set of if-then rules.

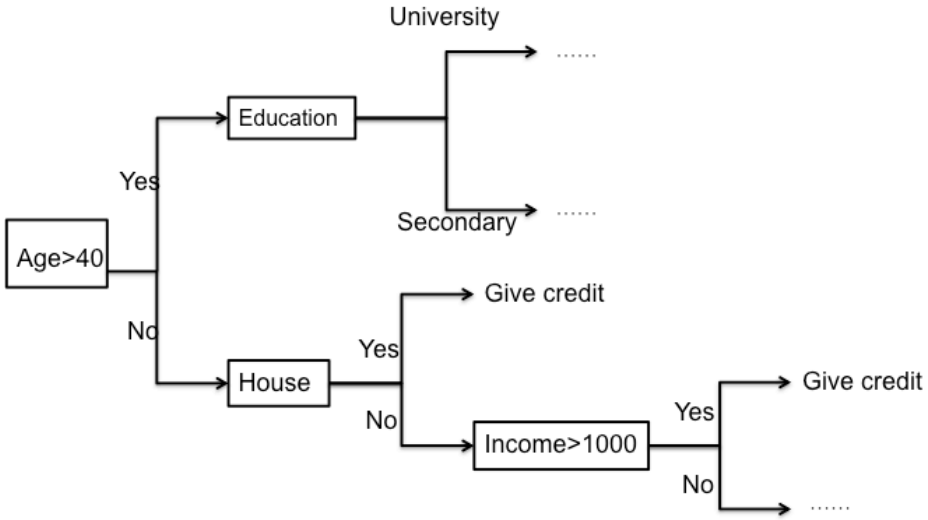


Figure 2.2 An example of a decision tree.

A decision tree consists of nodes and directed edges. A node can be an: internal or leaf node. Each internal node represents a feature and a leaf node represents a class. A tree begins at the root node. An example of a decision tree is shown in Figure 2.2. The root is "Age>40". "Education", "House" and "Income>1000" are internal nodes.

A decision tree is contrasted in a top-down fashion, by choosing a feature that best splits a dataset at each step. Different algorithms are based on different splitting criteria for measuring the "best" feature. [Quinlan, 1986] proposes a decision tree algorithm whose split is made by impurity measures called Iterative Dichotomiser 3 (ID3). This algorithm, howevr, does not support continuous attributes. Only categorical values are supported. Then [Quinlan, 1993] introduce C4.5 which can handle continuous

attributes. The splitting criterion in C4.5 uses the concept of information entropy. Chen, *et al*. [2011] use two commonly splitting criteria: information gain and Gini index during tree construction. Their method can test a node of both discrete and continuous data types. By default the tree tries to cover all possible outcomes in its structure. This feature leads the tree to over-fit the data into its solution.

To avoid over-fitting, optional pruning is required, where a further distinction can be made between pre-pruning and post-pruning [Esposito, *et al*., 1993]. Pre-pruning is a process that allows the construction of a tree to stop before all training data are correctly classified. It can easily reduce the complexity of the tree. Post-pruning is a process allowing the full construction of a tree, and then removing branches that are not considered to represent general properties of the learning.

There are several advantages to use decision trees. The algorithms are computationally fast and easy to understand. They can handle both continuous and discrete data types. Also, the missing values are well handled in a classification and regression tree (CART). Observing the internal nodes can allow onte to easily identify the important features. However, there are some disadvantages as well. Variations in data may produce different looking trees [Rokach & Maimon, 2005][Otero & Johnson, 2012], which are not good at predicting continuous attributes, because irrelevant attributes and noisy data may affect the tree structure. When using decision trees in imbalanced datasets, the minority class can lead to several problems. Because minority and majority classes are not evenly distribute in an imbalanced dataset. As a consequence, to detect minority instances from majority ones, the tree needs grow to be very complex. After applying the

pruning step, such specific branches that increase the accuracy of minority class may be removed. Then the resulting tree is probably biased to the majority class.

2.3 Evaluation Methods

In this section, we describe the measures that are used to evaluate the performance of the proposed methods in our experimental section. Evaluation methods are used to determine the effectiveness of classification algorithms. Commonly used measurements include accuracy, receiver operating characteristic (ROC) curves and area under the curve (AUC) and F-Measure.

2.3.1 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model. This table consists of two rows and two columns that introduce the numbers of true positive, false negative, false positive and true negative instances, as show in Table 2.1.

Table 2.1 Confusion Matrixes for Binary Classification

Confusion Matrix		**Predicted condition**	
		Positive	Negative
Actual condition	Positive	True Positive (Tp)	False Negative (Fn)
	Negative	False Positive (Fp)	True Negative (Tn)

In binary classification, a confusion matrix has four outcomes only:

True positive (Tp): positive instances correctly classified as positive.

False negative (Fn): positive instances classified as negative.

False positive (Fp): negative instances classified as positive.

True negative (Tn): negative instances correctly classified as negative.

Based on the values contained in a confusion matrix, several evaluation measures can be defined as follows:

$$\text{True positive rate (TPR)} = \frac{Tp}{Tp + Fn} \tag{2.1}$$

$$\text{True negative rate (TNR)} = \frac{Tn}{Tn + Fp} \tag{2.2}$$

$$\text{False positive rate (FPR)} = \frac{Fp}{Tn + Fp} \tag{2.3}$$

$$\text{False negative rate (FPR)} = \frac{Fn}{Tp + Fn} \tag{2.4}$$

2.3.2 Accuracy

The classification accuracy represents the accuracy of a model as the number of correct predictions from all predictions made. Accuracy is one of the most common performance measures.

For binary classification, we have:

$$Accuracy = \frac{Tp + Tn}{Tp + Fp + Tn + Fn} \tag{2.5}$$

However, this performance measure can be deceiving when facing an imbalanced dataset [He & Garcia, 2009]. In the following part, we give a simple example in Table 2.2 to illustrate the drawback of using accuracy measure for imbalanced classification.

Table 2.2 Confusion Matrixes for Two Models

Model 1		**Predicted Class**	
		Positive	Negative
Actual Class	Positive	3	7
	Negative	10	9990

Model 2		**Predicted Class**	
		Positive	Negative
Actual Class	Positive	8	2
	Negative	100	9900

The accuracy for Model 1 is (3+9990)/(3+7+10+9990) = 0.998. The accuracy for Model 2 is (8+9900)/(8+2+10+9900) = 0.990. If we select a model based on the accuracy only, Model 1 has better performance. However, the most important part for learning an imbalanced dataset is correctly identify the minority instances correctly. The true positive rate for Model 1 is 3/(3+7)=0.3. The true positive rate for Model 2 is 8/(8+2)=0.8. Model 2 clearly has much better prediction performance on the minority class.

2.3.3 F-Measure

F-Measure [Lewis & Gale, 1994] is one of the popular performance metrics. It is computed based on precision and recall.

$$Precision = \frac{Tp}{Tp + Fp} \tag{2.6}$$

$$Recall = \frac{Tp}{Tp + Fn} \tag{2.7}$$

Unlike the accuracy measure, these two measures avoid using true positive, which can be extremely high when dealing with the classification for an imbalanced dataset.

The meaning of the precision is to compare the correctly classified positive instances to the total number of instances that have been classified as positive. A larger value of precision corresponds to a higher number of correct positive predictions. The recall presents the percentage of correctly classified positive instances. Another name for recall is true positive rate.

Ideally, a classifier has both high recall and high precision, meaning the superb performance of correctly classifying a minority class. High recall represents that positive instances are mostly classified as positive and high precision illustrates the instances classified as positive mostly belong to the positive class. F-Measure contains the trade-off between precision and recall, which is defined as:

$$F = (1 + \beta^2) \cdot \frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{2.8}$$

where $\beta \in (0, +\infty)$ corresponds to the relative importance of recall over precision. When β=1, both are of the same importance [Rijsbergen, 1979].

2.3.4 ROC AUC

ROC analysis has received increasing attention in the recent data mining and machine learning literature [Fawcett, 2006][Chawla, 2005]. This curve models the trade-off between TPR and FPR. It is constructed in a two-dimensional space, plotting TPR and FPR on the x-axis and y-axis, respectively, as shown in Figure 2.3.

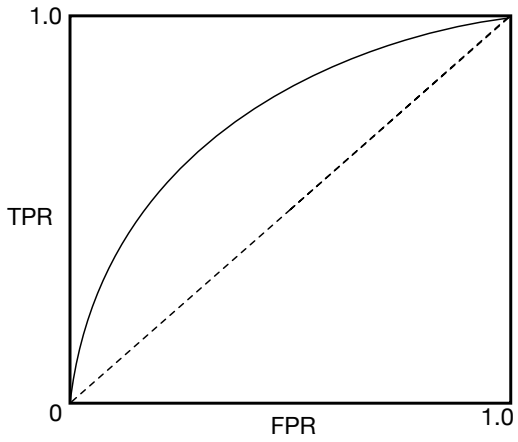


Figure 2.3 ROC curve.

The ROC AUC means that the area under the curve of receiver operation characteristic. The larger ROC AUC, the better prediction performance. The diagonal dashed line is the ROC curve of a random predicton. It has an ROC AUC of 0.5. The random predictor is commonly used as a baseline to see whether a model is useful.

CHAPTER 3

METHDOLOGY

3.1 Filter-based Feature Selection

Several feature selection methods have been introduced in the machine learning field [Liu & Motoda, 2012][Chandrashekar & Sahin, 2014]. Their main purpose is to remove either irrelevant or redundant features from the dataset. A filter-based feature selection method uses different statistical tests to determine the subset of features with the highest predictive power. For this method, different statistical metrics are chosen to calculate a score for each feature. Then, those features are ranked by scores and the feature columns with the highest scores are used to build the model, while others are kept in the dataset but not used for analysis. This method is independent of any particular classifier, and motivated by the properties of the data distribution itself. Two filter-based feature selection methods, i.e., Chi-square and F-statistic, are introduced for the comparison purpose in this thesis.

3.1.1 Chi-square

Chi-square [Rachburee & Punlumjeak, 2015], χ^2, test is used to test whether the given feature is related with the distribution of the class or not. The two-way Chi-squared test is a statistical method that measures the distance between expected values and actual results. The method assumes that variables are random and drawn from an adequate

sample of independent variables. For a given dataset about one of the features and the class, the observed instance count from the class is *O* and the expected instance count from the feature is *E*. Chi-square measures how much *E* and *O* derivate from each other. When the feature and the class are dependent, it means that this feature has relationship with the class and can be used to predict it. Thus, this method aims to select the feature that is highly dependent on the class.

The null hypothesis is that there is no relationship between the given feature and the class. The high value of χ^2 score indicates that the null hypothesis is incorrect. In other words, the higher value of χ^2 score, the greater relationship between the feature and the class is. Consequently, this given feature should be selected for model training.

The following example in Table 3.1 is used to present how to calculate χ^2 score between feature *X* and the class. In this example, $X=\{1, 2, \ldots, N\}$ is a categorical feature. X_i is the ith feature, and $X_i = \{x_{ij}\}$.

Table 3.1 Chi-square Computation Example

	Positive class	**Negative class**	**Total**
Value x_{ij} occurs	*A*	*B*	*A*+*B*
Value x_{ij} not occurs	*C*	*D*	*C*+*D*
Total	*A*+*C*	*B*+*D*	*N*

A: The number of positive instances that contain x_{ij};

B: The number of negative instances that contain x_{ij};

C: The number of positive instances that not contain x_{ij};

D: The number of negative instances that not contain x_{ij};

N: The total number of instances;

A+*C*: The total number of positive instances; and

A+*B*: The number of instances that contain x_{ij}.

The equation of expected value E_A is calculated based on the null hypothesis given as follows:

$$E_A = (A + B)\frac{A + B}{N} \tag{3.1}$$

Similarly, we can calculate E_B, E_C and E_D.

The χ^2 score between a feature and its target is computed as follows:

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{3.2}$$

where *r* is the number of different values of feature X_i, *c* is the number of classes, O_{ij} is the number of instances with value *i* in class *j*, and E_{ij} represents the expected number of instances with values *i* and *j*.

Based on the value of E_A, E_B, E_C and E_D, Equation (3.2) is extended to:

$$\chi^2 = \frac{(A - E_A)^2}{E_A} + \frac{(B - E_B)^2}{E_B} + \frac{(C - E_C)^2}{E_C} + \frac{(D - E_D)^2}{E_D} \tag{3.3}$$

Based on (3.1), χ^2 is updated as:

$$\chi^2 = \frac{N(AD - BC)^2}{(A + C)(B + D)(A + B)(C + D)} \tag{3.4}$$

3.1.2 F-statistic

F-statistic [Ding & Peng, 2005] is also called one-way analysis of variance (ANOVA) F-test, which can determine the importance of a feature in discriminant analysis [Hosseini & Mahdavi, 2015]. The F-test value of a given feature is computed as follows:

$$F = \frac{\sum_{j=1}^{J} N_j \left(\overline{x_J} - \bar{x}\right)^2 / (J - 1)}{\delta^2} \tag{3.5}$$

where J is the number of classes, N_j is the number of instances in the jth class, $\overline{x_j}$ is the mean of instances X in class j, $\bar{x}$ shows the mean value for all the instances, and δ^2 is the pooled variance computed as follows:

$$\delta^2 = \sum_{j=1}^{J} (N_j - 1) {s_j}^2 / (N - 1) \tag{3.6}$$

3.2 Decision Tree Rule-based Feature Selection Method

The decision tree method is one of popular algorithms used in machine learning. A standard tree consists of a number of branches, one root, and a number of nodes and leaves. One branch is a chain of nodes from the root to a leaf; and each node involves one feature. The occurrence of a feature in a tree provides the information about the importance of the associated feature. The high occurrence of a feature means that this feature is highly relevant with a target. A decision tree can be built by using Algorithm 3.1:

Algorithm 3.1 Decision tree

Input: a dataset *D*
Tree = {}
if *D* is "pure" OR other stopping criteria are met **then**
    terminate
end if
for all feature *f* **do**
    Compute splitting criteria if we split on *f*
end for
f_{best} = Best feature according to the above computed criteria
Tree = Create a decision node that tests f_{best} in the root
D_V = Induced sub-datasets from *D* based on f_{best}
for all D_V **do**
    Tree_V = Decision tree (D_V)
    Attach Tree_V to the corresponding branch of Tree
end for
return Tree

A decision tree rule-based feature selection method discovers the frequency of each node. One node corresponds to one feature. Choosing which feature can be constructed as a new node is answered by using different splitting criteria in different decision tree algorithms. Our proposed feature selection method does not calculate the frequency of features after the tree is built. Instead we calculate it as along as we evaluate features based on a splitting criterion. Because in some cases, several features have the same highest splitting value, then only one of features is selected as a node. At that time, these features that are not selected are not computed, even though it is as important as the selected feature. It means that we these features' information. However, if we calculate the frequency of features as along as the splitting criterion is used, this kind of loss is reduced.

3.2.1 Splitting Criteria

There are many metrics that can be used to determine the best way to split the dataset. In this section, a binary classification is used as an example. Let $p(i|j)$ denotes the relative frequency of class i at node j. The developed metrics for selecting the best splitting are often based on the degree of impurity of child nodes. The node contains a higher impurity means that it has a higher probability to be chosen. Examples of impurity metrics include:

$$Entropy(j) = -\sum_{i} p(i|j)\log\left(p(i|j)\right) \tag{3.7}$$

$$Error(j) = 1 - \max_i P(i|j) \tag{3.8}$$

$$Gini(j) = 1 - \sum_i [P(i|j)]^2 \tag{3.9}$$

We next provide one example of calculating different splitting criteria, as illustrated in Table 3.2.

Table 3.2 Count for Nodes *A* and *B*

	Node *A* count	**Node *B* count**
Positive class	1	2
Negative class	5	4

$$Entropy(A) = -\frac{1}{6}\log_2\frac{1}{6} - \frac{5}{6}\log_2\frac{5}{6} = 0.65$$

$$Error(A) = 1 - \max\left(\frac{1}{6}, \frac{5}{6}\right) = 0.167$$

$$Gini(A) = 1 - \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 = 0.278$$

Similarly, we can compute B as follows:

$$Entropy(B) = -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} = 0.92$$

$$Error(B) = 1 - \max\left(\frac{2}{6}, \frac{4}{6}\right) = 0.333$$

$$Gini(B) = 1 - \left(\frac{2}{6}\right)^2 - \left(\frac{4}{6}\right)^2 = 0.444$$

Clearly we choose A since its impurity metric is better performance than B's.

3.2.2 CART

Different splitting criteria lead to different decision trees. ID3 and C4.5 algorithms select nodes based on an information gain that relies on the concept of entropy. The CART algorithm selects a Gini index as a metric to test the node impurity. In our proposed feature selection method, we select the CART algorithm.

CART is a binary decision tree that is constructed by splitting data into two parts with maximum homogeneity. Breiman *et al*. (1984) presented this algorithm in 1980s. The advantages for using this algorithm are as follows:

(1) CART results are invariant to monotone transformations of its independent variables.

Changing one or several variables to its logarithm or square root does not change the structure of the tree. Only the splitting values are different.

(2) CART can easily handle outliers.

Outliers can negatively affect the results of some learning algorithms. However, the splitting criterion for CART can easily handle them. It isolates them in a separate

node.

(3) Features can repeatedly be used in CART.

When using a decision tree algorithm for feature selection, we need to calculate how often one feature is selected as a node in a tree. In the CART tree, it is easy to observe this frequency. Higher frequency of occurrence of a feature means that it has a higher chance to be selected in the feature selection method.

(4) CART can easily handle both continuous and categorical features

CART is a binary tree. For a continuous feature, the Gini index splitting criterion can traverse all possible values, and then calculate each node impurity corresponding to each value. Finally select the value that contains highest node impurity as this feature splitting value.

Building a CART tree consists of the following three steps:

(a) Constructing a maximum size of a tree, which is built by using recursive splitting of nodes. The best splitting nodes are chosen by searching all possible variables and all possible values. The splitting criterion for the CART algorithm is called Gini index. The Gini index is formally described as follows:

Let I be a set of classes, $j \in \{1,2,\ldots,M\}$ be a node, and $P(i|j)$ be the relative frequency of class i at node j. The Gini index at node j is defined as:

$$Gini(j) = 1 - \sum_{i} [P(i|j)]^2 \tag{3.10}$$

The Gini index used for different splitting nodes can be computed as:

$$Gini_{split} = \sum_{j=1}^{M} \frac{n_j}{n} Gini(j) \tag{3.11}$$

where n is the number of instances at node j. Therefore, the objective of choosing a splitting node is shown as:

$$\arg\min Gini_{split}(j) \tag{3.12}$$

(b) Performing the "pruning" step of a tree. The CART algorithm uses a "cost-complexity" pruning method in this step. It relies on complexity parameter α, which represents the amount of additional accuracy for one splitting that must add to the entire tree to reduce complexity.

(c) Optimizing the tree to avoid overfitting. A cross-validation procedure is used in this step to find a proper α to balance the size of the tree and reduce the misclassification error. A cost-complexity function is described as:

$$R_\alpha(T) = R(T) + \alpha(\tilde{T}) \tag{3.13}$$

where $R(T)$ is the misclassification error of a tree T. $\alpha(\tilde{T})$ is the complexity of the tree $\tilde{T}$, which is the number of nodes in the tree.

3.2.3 Weighted Gini Index for CART

Table 3.3 shows the parent node splitting result after chosing a feature as a child node.

Table 3.3 Matrix for One Splitting Node

		Child nodes	
		Positive (D_1)	Negative (D_2)
Parent Node	Positive	True Positive (Tp)	False Negative (Fn)
	Negative	False Positive (Fp)	True Negative (Tn)

The Gini index based on Table 3.3 is described as follows:

$$Gini(D) = 1 - \left(\frac{Tp + Fn}{N}\right)^2 - \left(\frac{Fp + Tn}{N}\right)^2 \tag{3.14}$$

$$Gini(D_1) = 1 - \left(\frac{Tp}{Tp + Fp}\right)^2 - \left(\frac{Fp}{Tp + Fp}\right)^2 \tag{3.15}$$

$$Gini(D_2) = 1 - \left(\frac{Fp}{Fn + Tn}\right)^2 - \left(\frac{Tn}{Fn + Tn}\right)^2 \tag{3.16}$$

$$Gini_R(D) = \frac{|D_1|}{D} Gini(D_1) + \frac{|D_2|}{D} Gini(D_2)$$
$$= \frac{Tp + Fp}{N} Gini(D_1) + \frac{Fn + Tn}{N} Gini(D_2) \quad (3.17)$$

$$\Delta Gini(R) = Gini(D) - Gini_R(D) \quad (3.18)$$

where D is a parent node, D_1 and D_2 are two child nodes, and N is the total number of parent nodes, which is equal to the sum of Tp, Fp, Fn and Tn. R is the splitting value for node D. $Gini(D)$ is the impurity value for node D. If all instances belong to a same class, this value equals 0. $\Delta Gini(R)$ denotes the decrement in impurity. The node is chosen if it is maximized or the Gini index is minimized.

However, when the dataset is highly imbalanced, the Gini index splitting criterion is biased to the majority class. An example is given to illustrate this problem as shown in Table 3.4-3.6.

Table 3.4 Splitting Results for Node 1

Node 1		**Child nodes**	
		Positive (D_1)	Negative (D_2)
Parent Node	Positive	3	7
	Negative	10	9990

$$Gini(D) = 1 - \left(\frac{3+7}{10010}\right)^2 - \left(\frac{10+9990}{10010}\right)^2 = 1.996 * 10^{-3}$$

$$Gini(D_1) = 1 - \left(\frac{3}{3+10}\right)^2 - \left(\frac{10}{3+10}\right)^2 = 0.355$$

$$Gini(D_2) = 1 - \left(\frac{7}{7+9990}\right)^2 - \left(\frac{9990}{7+9990}\right)^2 = 1.399 * 10^{-3}$$

$$Gini_R(D) = \frac{3+10}{10010} Gini(D_1) + \frac{7+9990}{10010} Gini(D_2) = 1.858 * 10^{-3}$$

$$\Delta Gini(R) = 1.996 * 10^{-3} - 1.858 * 10^{-3} = 0.138 * 10^{-3}$$

Table 3.5 Splitting Results for Node 2

Node 2		**Child nodes**	
		Positive (D_1)	Negative (D_2)
Parent Node	Positive	8	2
	Negative	100	9900

$$Gini(D) = 1 - \left(\frac{8+2}{10010}\right)^2 - \left(\frac{100+9900}{10010}\right)^2 = 1.996 * 10^{-3}$$

$$Gini(D_1) = 1 - \left(\frac{8}{8+100}\right)^2 - \left(\frac{100}{8+100}\right)^2 = 0.137$$

$$Gini(D_2) = 1 - \left(\frac{2}{2+9900}\right)^2 - \left(\frac{9900}{2+9900}\right)^2 = 0.404 * 10^{-3}$$

$$Gini_R(D) = \frac{8+100}{10010} Gini(D_1) + \frac{2+9900}{10010} Gini(D_2) = 1.878 * 10^{-3}$$

$$\Delta Gini(R) = 1.996 * 10^{-3} - 1.878 * 10^{-3} = 0.118 * 10^{-3}$$

Table 3.6 Splitting Results for Node 3

Node 3		**Child nodes**	
		Positive (D_1)	Negative (D_2)
Parent Node	Positive	8	2
	Negative	10	9990

$$Gini(D) = 1 - \left(\frac{8+2}{10010}\right)^2 - \left(\frac{10+9990}{10010}\right)^2 = 1.996 * 10^{-3}$$

$$Gini(D_1) = 1 - \left(\frac{8}{8+10}\right)^2 - \left(\frac{10}{8+10}\right)^2 = 0.494$$

$$Gini(D_2) = 1 - \left(\frac{2}{2+9990}\right)^2 - \left(\frac{9990}{2+9990}\right)^2 = 0.400 * 10^{-3}$$

$$Gini_R(D) = \frac{8+10}{10010} Gini(D_1) + \frac{2+9990}{10010} Gini(D_2) = 1.287 * 10^{-3}$$

$$\Delta Gini(R) = 1.996 * 10^{-3} - 1.287 * 10^{-3} = 0.709 * 10^{-3}$$

Based on the value of the decrement in impurity for three nodes, we conclude that Node 3 contains the highest decrement value and is thus chosen. Also, comparing Node 3

with other two nodes, it has higher result on TPR and TNR than the other two. So, it is reasonable to choose this one first. However, when we compare Node 1 with Node 2, Node 1 has its priority to be selected since it has higher value on $\Delta Gini(R)$. For the imbalanced data, the minority class is the one to which we pay more attention than the majority one. That means TPR is more important than TNR. At this time, node 2 clearly has higher result on TPR than Node 1.

For this imbalanced class problem, we propose a weighted Gini index to increase the chance for us to choose the features more bias to minority class.

$$Gini(j) = 1 - \sum_{i} [P(i|j)]^2 = 1 - \left(\frac{N^+ * w}{N^+ * w + N^-}\right)^2 - \left(\frac{N^-}{N^+ * w + N^-}\right)^2 \quad (3.19)$$

$$Gini(D) = 1 - \left(\frac{(Tp + Fn) * w}{N^*}\right)^2 - \left(\frac{Fp + Tn}{N^*}\right)^2 \quad (3.20)$$

$$Gini(D_1) = 1 - \left(\frac{Tp * w}{Tp * w + Fp}\right)^2 - \left(\frac{Fp}{Tp * w + Fp}\right)^2 \quad (3.21)$$

$$Gini(D_2) = 1 - \left(\frac{Fn * w}{Fn * w + Tn}\right)^2 - \left(\frac{Tn}{Fn * w + Tn}\right)^2 \quad (3.22)$$

$$Gini_R(D) = \frac{|D_1|}{D} Gini(D_1) + \frac{|D_2|}{D} Gini(D_2)$$
$$= \frac{(Tp + Fp) * w}{N^*} Gini(D_1) + \frac{Fn + Tn}{N^*} Gini(D_2) \quad (3.23)$$

$$N^* = (Tp + Fn) * w + (Fp + Tn) \quad (3.24)$$

Where *w* is the newly introduced weight.

According to Equations (3.19)-(3.23), the value of decrement in impurity of previous three nodes is calculated as follows where in this example, we select $w = \frac{N^-}{N^+} = \frac{10000}{10} = 1000$ is selected:

Node 1:

$$Gini(D) = 1 - \left(\frac{(3+7) * 1000}{10 * 1000 + 10000}\right)^2 - \left(\frac{10 + 9990}{10 * 1000 + 10000}\right)^2 = 0.5$$

$$Gini(D_1) = 1 - \left(\frac{3 * 1000}{3 * 1000 + 10}\right)^2 - \left(\frac{10}{3 * 1000 + 10}\right)^2 = 6.622 * 10^{-3}$$

$$Gini(D_2) = 1 - \left(\frac{7 * 1000}{7 * 1000 + 9990}\right)^2 - \left(\frac{9990}{7 * 1000 + 9990}\right)^2 = 0.484$$

$$Gini_R(D) = \frac{3 * 1000 + 10}{10 * 1000 + 10000} Gini(D_1) + \frac{7 * 1000 + 9990}{10 * 1000 + 10000} Gini(D_2) = 0.412$$

$$\Delta Gini(R) = 0.5 - 0.412 = 0.088$$

Node 2:

$$Gini(D) = 1 - \left(\frac{(8+2)*1000}{10*1000+10000}\right)^2 - \left(\frac{100+9900}{10*1000+10000}\right)^2 = 0.5$$

$$Gini(D_1) = 1 - \left(\frac{8*1000}{8*1000+100}\right)^2 - \left(\frac{100}{8*1000+100}\right)^2 = 0.024$$

$$Gini(D_2) = 1 - \left(\frac{2*1000}{2*1000+9900}\right)^2 - \left(\frac{9900}{2*1000+9900}\right)^2 = 0.280$$

$$Gini_R(D) = \frac{8*1000+100}{10*1000+10000} Gini(D_1) + \frac{2*1000+9900}{10*1000+10000} Gini(D_2) = 0.176$$

$$\Delta Gini(R) = 0.5 - 0.176 = 0.324$$

Node 3:

$$Gini(D) = 1 - \left(\frac{(8+2)*1000}{10*1000+10000}\right)^2 - \left(\frac{10+9990}{10*1000+10000}\right)^2 = 0.5$$

$$Gini(D_1) = 1 - \left(\frac{8*1000}{8*1000+10}\right)^2 - \left(\frac{10}{8*1000+10}\right)^2 = 2.494*10^{-3}$$

$$Gini(D_2) = 1 - \left(\frac{2*1000}{2*1000+9990}\right)^2 - \left(\frac{9990}{2*1000+9990}\right)^2 = 0.278$$

$$Gini_R(D) = \frac{8*1000+10}{10*1000+10000} Gini(D_1) + \frac{2*1000+9990}{10*1000+10000} Gini(D_2) = 0.167$$

$$\Delta Gini(R) = 0.5 - 0.167 = 0.333$$

Comparing $\Delta Gini(R)$ for these three notes, node 3 has the highest score, then node 2, and the last one is node 1. It means that the weighted Gini index splitting criterion contains better performance on TPR.

3.3 Classification

In order to verify the performance of the proposed feature selection approach, we test our new subset of features on eXtreme Gradient Boosting (Xgboost) classifier [Chen & He, 2015]. It is an efficient and faster implementation of gradient boosting machines [Chen & Guestrin, 2016]. Xgboost is an ensemble algorithm where new models are added to correct the errors made by the existing models. Models are added until no further performance improvement can be gained. This algorithm can support both regression and classification work.

CHAPTER 4

Experimental Results

4.1 Datasets

In order to test the performance of our proposed method, three datasets are collected and used. One of them is from Santander Bank and posted on the Kaggle competition [https://www.kaggle.com/c/santander-customer-satisfaction]. The other two datasets, i.e., the letter recognition dataset and statlog (ladsat satellite) dataset, are from UCI machine learning repository. Table 4.1 summarizes the number of instances and features and the class ratios of the datasets.

Table 4.1 Summary of Benchmark Datasets

Dataset	Instances	Features	Ratio (Majority/Minority)
Bank	76,020	371	24
Letter	20,000	16	24.3
Statlog	6,435	36	9.3

Originally, the Santander Bank dataset is used to predict whether a customer is satisfied or not with their banking experience. The number of instances is 76,020, and the number of features is 371. The imbalanced ratio is 24, which means that the number of negative instances is 24 times larger than the number of positive ones. Then, we randomly split the dataset into training and testing ones with a ratio of 6:4, i.e., 60% data for training and the remaining for test.

The letter recognition dataset consists of 20,000 instances and 16 features. All the features are of integer values. The objective is to identify the 26 capital letters in the English alphabet based on the 16 features from images. The character images are based on 20 different fonts and each letter within these 20 fonts is randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus is converted into 16 primitive numerical features which are then scaled to fit into a range of integer values from 0 to 15. For this dataset, we use 16,000 instances for training and the remaining 4,000 instances for testing. The original dataset contains 26 classes and correspond to 26 English letters from A to Z. For this dataset, we assume the class A as the minority class and the rest as majority class. Our target is to distinguish class A from the rest classes. In other words, we convert the dataset to exhibit a binary classification problem.

The statlog (ladsat satellite) dataset consists of the multi-spectral values of pixels in 3×3 neighborhoods in a satellite image, and the classification is associated with the central pixel in each neighborhood. The aim is to predict this classification by using the given multi-spectral values. This dataset contains 6,435 instances with 36 features. The features are numerical and their values range from 0 to 255. This is 6-class classification problem. For this multi-class dataset, we select the least frequent class label as the minority and the rest as the majority. Consequently, class 4 is treated as the minority class and the rest of classes are combined into one class as the majority class.

4.2 Experimental Design

In our experiments, two filter based feature selection methods, Chi-square and F-statistic, are used and compared with our proposed methods, i.e., decision tree rule-based feature

selection (DT-FS) and weighted Gini index feature selection (WGI-FS). In order to evaluate the performance among these feature selection methods, we adopt xgboost as our classification model. Note that 5-fold cross validation is used to avoid an overfitting issue. Finally, the metric ROC AUC and F-measure are computed to evaluate the performance among three feature selection methods.

4.3 Case Study 1: Santander Bank Dataset

In this section, we show the results of comparisons among DT-FS, Chi-square and *F-statistic.*

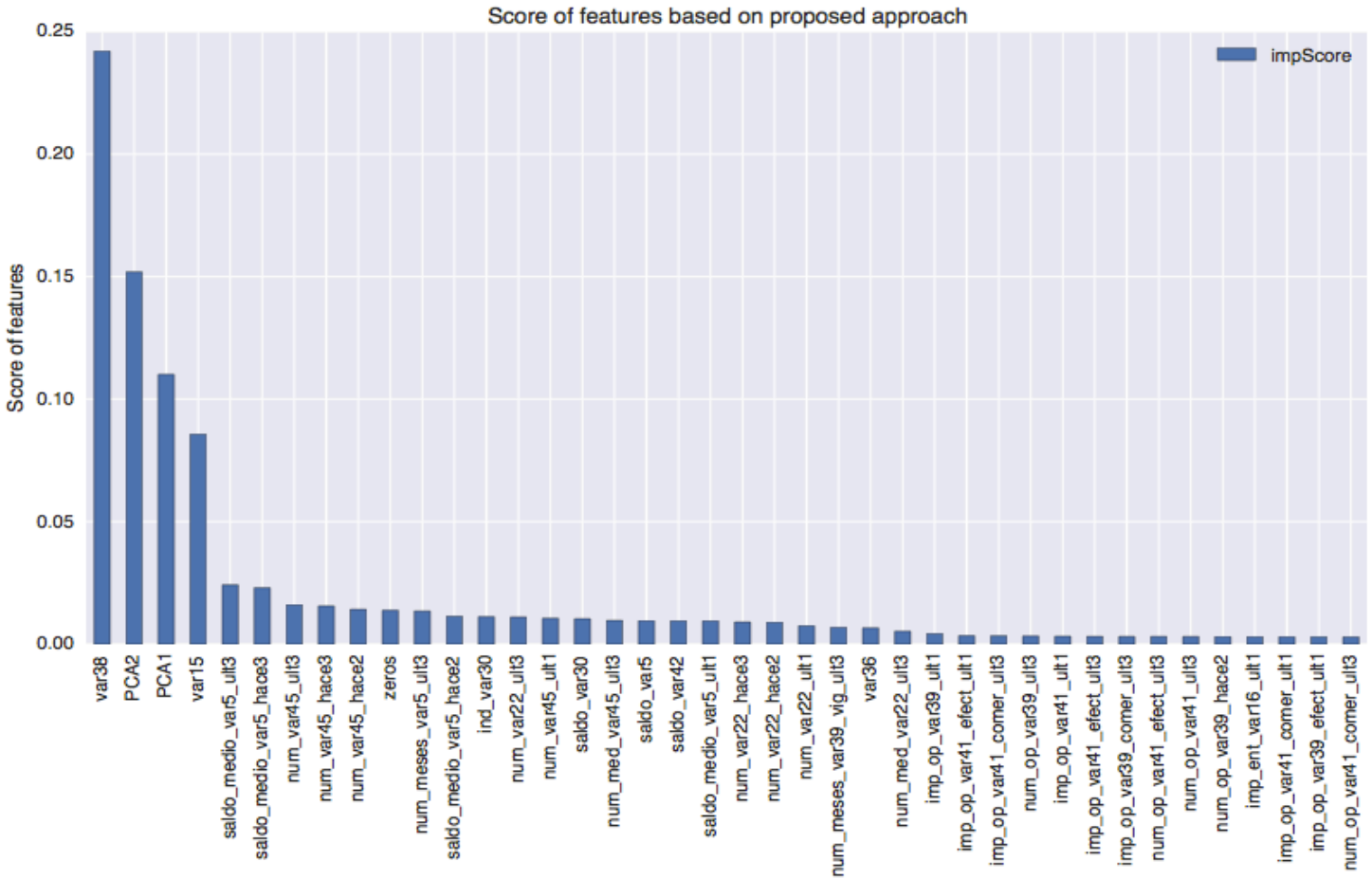


Figure 4.1 Score of features based on DT-FS

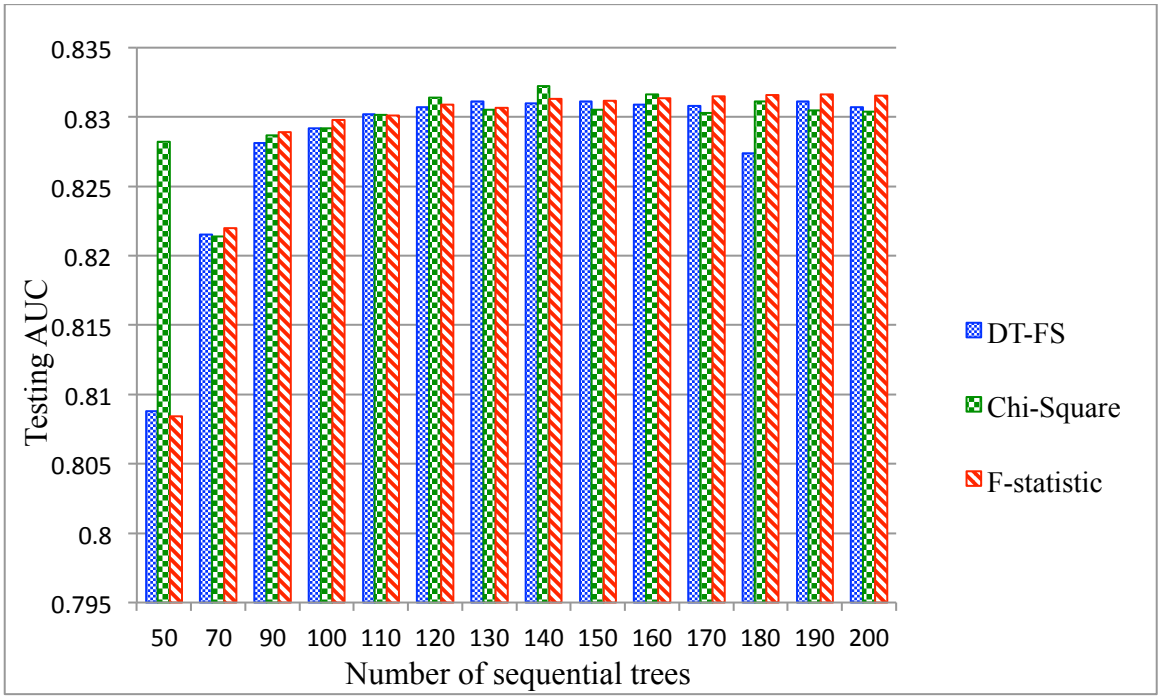


Figure 4.2 Performance of three feature selection methods in terms of ROC AUC.

Figure 4.1 shows the features' scores based on our proposed approach in descending order. If a feature receives a higher score, it represents that it is more relevant with the prediction. Figure 4.2 demonstrates the comparisons of highest testing ROC AUC results among the three feature selection methods with a different number of sequential trees for xgboost. Note that the horizontal axis represents the number of sequential trees which is a parameter for xgboost. This parameter means that the number of trees is used for boosting on xgboost. According to the result, we conclude that the ROC AUC results from three feature selection methods are slightly different, except when the number of sequential trees equals 50.

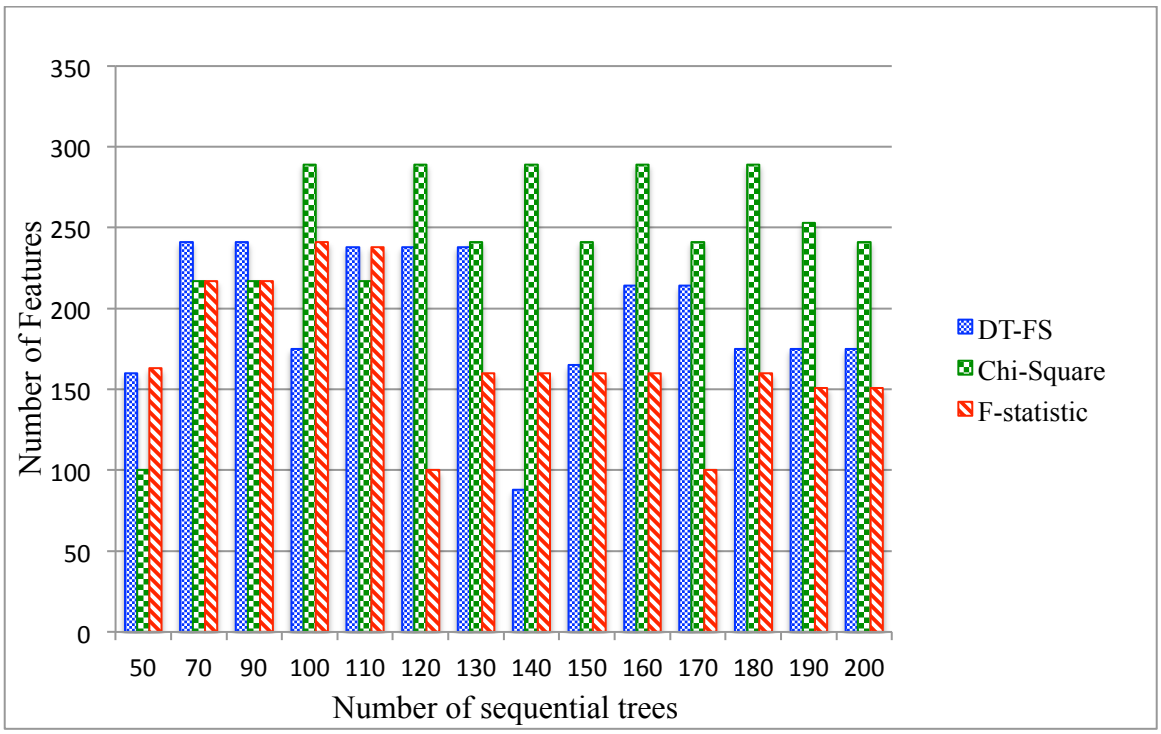


Figure 4.3 Number of needed features among three feature selection methods.

Figure 4.3 gives the numbers of needed features for different feature selection methods when they reach the highest testing ROC AUC under the same number of sequential trees. For example, when 100 sequential trees are chosen, both DT-FS and Chi-square achieve the same 0.828 ROC AUC result, with F-statistic being 0.829. We compare the number of needed features that are selected based on DT-FS and Chi-square. DT-FS needs fewer features than Chi-square to obtain same ROC AUC result. F-statistic needs more features than DT-FS, but fewer than Chi-square to obtain the slightly higher ROC AUC result.

It is observed that our proposed algorithm needs more features in order to get the same testing ROC AUC, when a small number of sequential trees is chosen. As the number of sequential trees increases, the Chi-square feature selection approach needs the

largest number of features, and F-statistic needs the smallest number of features. However, the highest testing ROC AUC is around 0.83, and only floats in a small range.

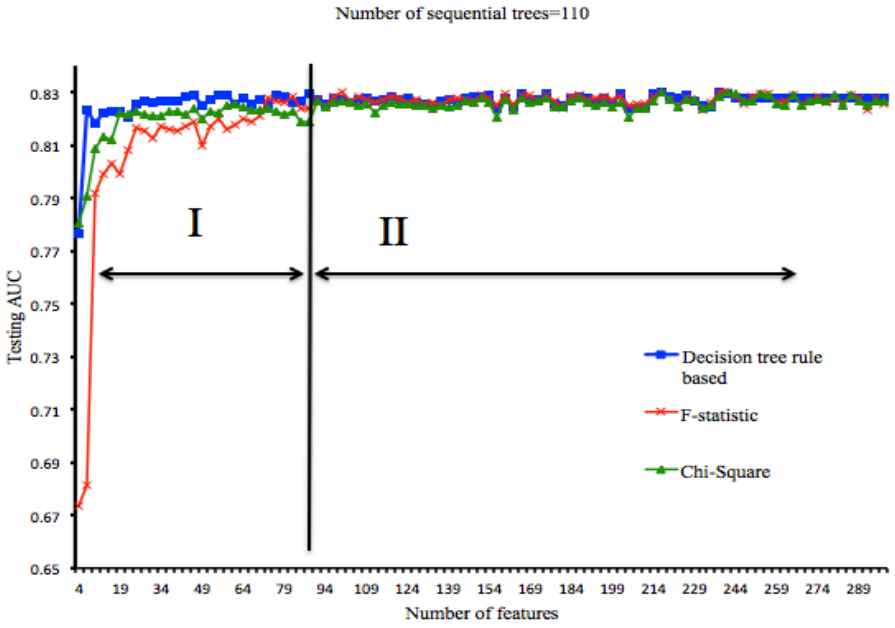


Figure 4.4 Performance of three feature selection methods in terms of ROC AUC (number of sequential trees = 110).

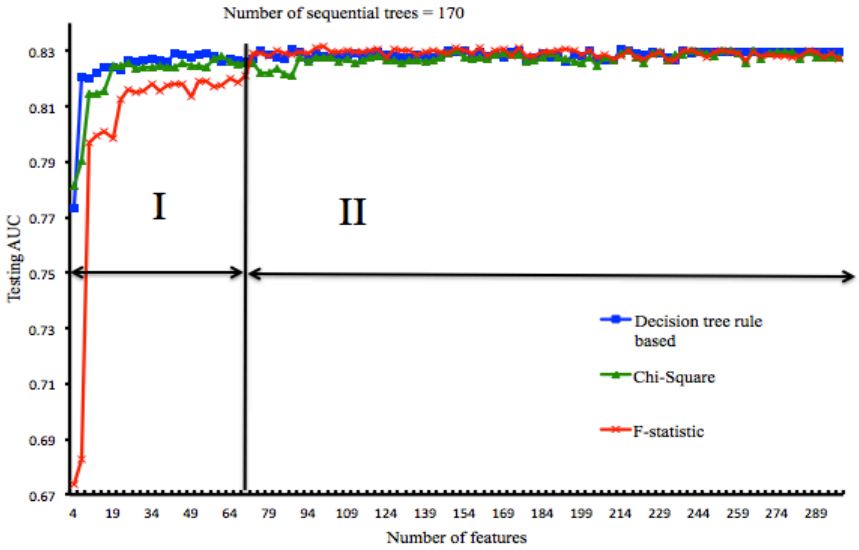


Figure 4.5 Performance of three feature selection methods in terms of ROC AUC (number of sequential trees = 170).

As shown in Figures 4.4-4.5, we compare the tendency of ROC AUC among the three different feature selection methods based on the number of sequential trees, i.e., 110 and 170. We select the number of features from 4 to 371. Note that 371 is the dimension of our dataset. We observe that the subset of features, which our proposed algorithm selects, has the better ROC AUC than F-statistic and Chi-Square in Part I of Figures 4.4-4.5. It shows that the proposed method can find the most relevant features quickly. In Part II of these two figures, the tendency of ROC AUC of these three different approaches tend to coincide. Thus, the proposed method obtains the best performance among the three methods while dealing with fewer features are the similar results after the number of features is over a certain number, e.g., 94 in Figure 4.4.

Table 4.2 Full features Versus Top Ranked Features Chosen Based on The DT-FS

	Feature size	**Testing ROC AUC**	**Training ROC AUC**	**Computational time**
Full features	371	0.822	**0.865**	95.90
Top rank (DT-FS)	31	**0.824**	0.858	39.26
	16	**0.824**	0.854	**30.96**

This part of experiment intends to perform the comparisons using the selected from the proposed method features and all features for classification. In Table 4.2, we choose the number of features as 31 and 16 to show results. The baseline of testing AUC, training AUC and computational time for the whole 371 features is 0.8221 and 0.8647, 95.90 seconds, respectively. We compare these results with the ones when our proposed feature selection method is used. We conclude that our proposed feature selection method exhibits a good performance, even though over 90% percent of features are reduced. Also,

its use drastically reduce the classification time as expected. It shows that our proposed feature selection approach can successfully work for features selection in the large-scale imbalanced dataset.

Table 4.3 Random Feature Selection Versus Top Ranked Features Chosen Based on The DT-FS

	Feature size	**Testing AUC**	**Training AUC**	**Computational time**
Random feature chosen	31	0.7495	0.7900	41.04
	16	0.6883	0.7003	35.87
Top rank (DT-FS)	31	**0.8239**	**0.8581**	**39.26**
	16	**0.8236**	**0.8543**	**30.96**

May a randomly selected subset of features perform well? We compare our proposed method's results with the random features selection method's results in Table 4.3. We randomly choose the number of features as 31 and 16 from the 371 features, and then present testing AUC, training AUC and computational time for these features, which are 0.7495, 0.7900 and 41.04, respectively. Based on the results that are produced with the random feature selection, we conclude that many features are irrelevant with the prediction. Our proposed feature selection method achieves higher testing AUC and training AUC with less computational time.

Table 4.4 Comparison Among Chi-Square, F-statistic and DT-FS

	Feature size	**Testing AUC**	**Training AUC**	**Computational time**
Chi-square	31	0.8230	0.8439	32.43
	16	0.8156	0.8307	**26.96**
F-statistic	31	0.8167	0.8368	**30.89**
	16	0.8036	0.8213	28.30
DT-FS	31	**0.8239**	**0.8581**	39.26
	16	**0.8236**	**0.8543**	30.96

From Table 4.4, we conclude that our proposed method is better than its two peers, Chi-square and F-statistic. The best testing AUC, training AUC and computational time are given in bold. In this table, in order to compare the results in detail, only the numbers of features being 16 and 31 are taken into consideration. For other numbers of features, part I of Figure 4.4 shows the same tendency as the use of 16 and 31 features has. The highest testing and training AUCs are achieved by using the subset of features that are selected by our proposed method. However, in both cases, 16 and 31 features, the proposed method is slightly slower than its two peers.

4.4 Case Study 2: Letter Recognition Dataset

In this section, the letter recognition dataset is used and we show the results of comparisons among DT-FS, chi-square, *F-statistic* and WGI-FS. For the WGI-FS method, the two different weights are selected which are equal to imbalanced ratio (ρ) and 1.5ρ, respectively.

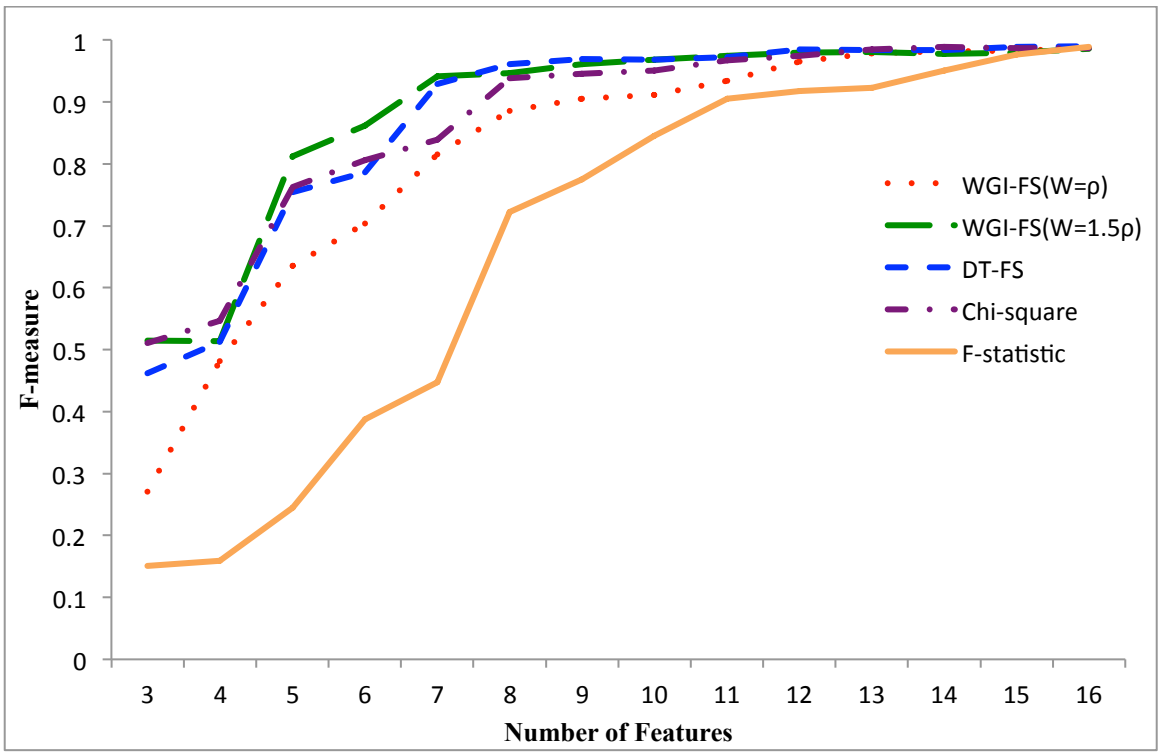


Figure 4.6 Performance of five feature selection methods in terms of F-measure (Letter dataset).

In Figure 4.6, we show the tendency of F-measure among the five different feature selection methods based on the number of sequential trees that equals 110. The results show the mean of 6-fold results. The number of features is selected from a range from 3 to 16. It is observed that when its weight w=1.5ρ, WGI-FS gets the highest F-measure among all the tested five methods. As the number of selected features increases, the performance of DT-FS also achieves a better result than the other two methods.

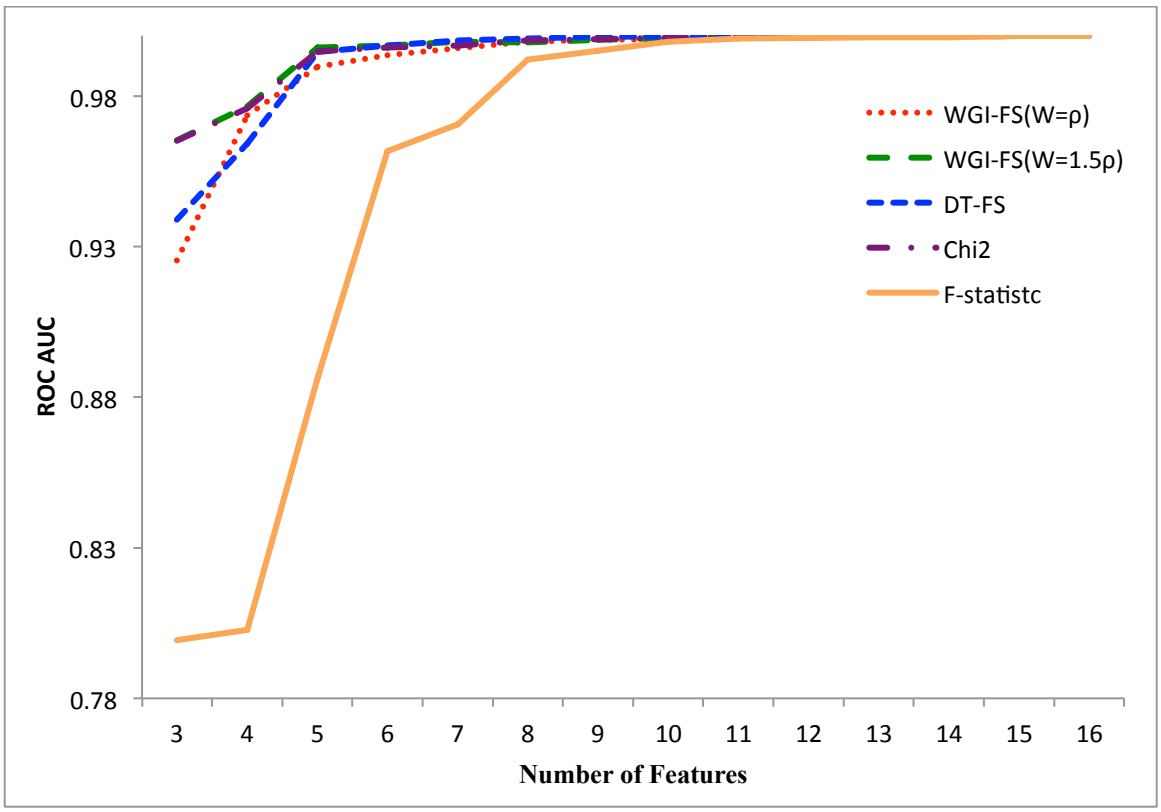


Figure 4.7 Performance of five feature selection methods in terms of ROC AUC (Letter dataset).

In Figure 4.7, the performance of ROC AUC is shown. According to the result, we can observe that WGI-FS(W=1.5ρ) and Chi-square obtain the highest score no matter how many features are chosen. As the size of selected features increases, DT-FS and WGI-FS(W=ρ) reach the same performance as WGI-FS(W=1.5ρ) and Chi-square. However, F-statistic gives the worst performance.

Table 4.5 Performance of Five Feature Selection Methods in Terms of F-measure (selecting 20% features)

F-measure	WGI-FS (W=ρ)	WGI-FS (W=1.5ρ)	DT-FS	Chi-square	F-statistic
1	0.642	**0.811**	0.768	0.753	0.263
2	0.262	**0.804**	0.751	0.757	0.262
3	0.240	**0.821**	0.757	0.787	0.240
4	0.235	**0.823**	0.766	0.756	0.235
5	0.237	**0.809**	0.747	0.761	0.237
6	0.233	**0.804**	0.736	0.763	0.233
Mean	0.308	**0.812**	0.754	0.763	0.245

Table 4.6 Performance of Five Feature Selection Methods in Terms of ROC AUC (selecting 20% features)

ROC AUC	WGI-FS (W=ρ)	WGI-FS (W=1.5ρ)	DT-FS	Chi-square	F-statistic
1	0.984	0.997	**0.998**	0.996	0.895
2	0.893	**0.997**	0.996	0.998	0.893
3	0.887	**0.999**	0.998	0.998	0.887
4	0.887	**0.999**	0.998	0.997	0.887
5	0.883	**0.995**	0.990	0.992	0.883
6	0.874	**0.988**	**0.988**	0.987	0.874
Mean	0.901	**0.996**	0.995	0.995	0.886

The performance of F-measure and ROC AUC are shown in Tables 4.5-4.6. In these two tables, 20% of features are selected based on the five feature selection methods. According to the results with the 6-fold cross validation, we can observe that WGI-FS(W=1.5ρ) has the highest performance in these two evaluation measurements, especially it achieves much better performance than F-statistic. In other words, WGI-FS(W=1.5ρ) has better performance when identifying the minority class.

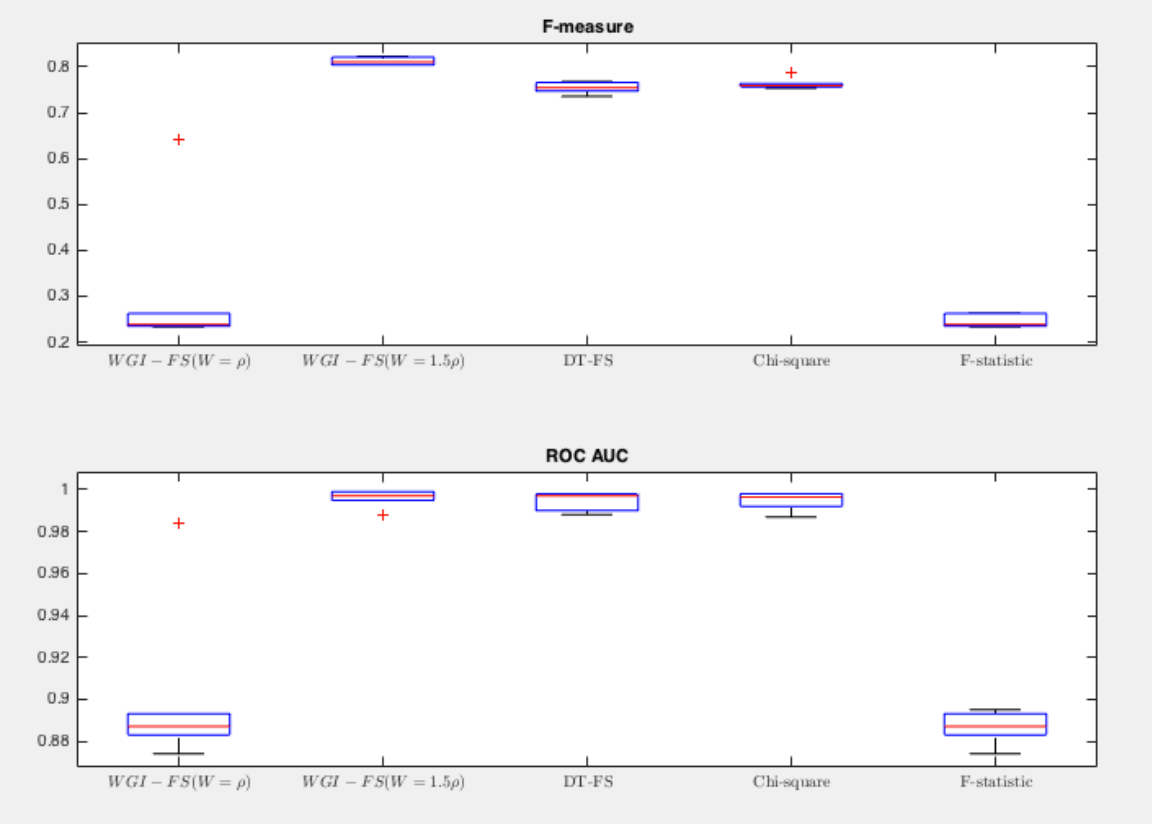


Figure 4.8 Boxplot for F-measure and ROC AUC (Letter dataset).

Based on Tables 4.5-4.6, we draw Figure 4.8 to show the boxplot for F-measure and ROC AUC. According to the figure, we observe that the subset of features selected from WGI-FS(W=1.5ρ) obtains the highest performance in both F-measure and ROC

AUC. DT-FS and Chi-square methods also achieve a comparable performance for this dataset.

From Figures 4.9 and 4.10, we can observe that WGI-FS(W=1.5ρ), DT-FS and Chi-square methods, when their selected features are larger than 20% of features, can achieve excellent performance on both F-measure and ROC AUC. If 20% of features is adopted, the proposed WGI-FS(W=1.5ρ) obtains the superb result on F-measure. It means that this method realizes a better result on recognizing the minority class at anytime than other four methods.

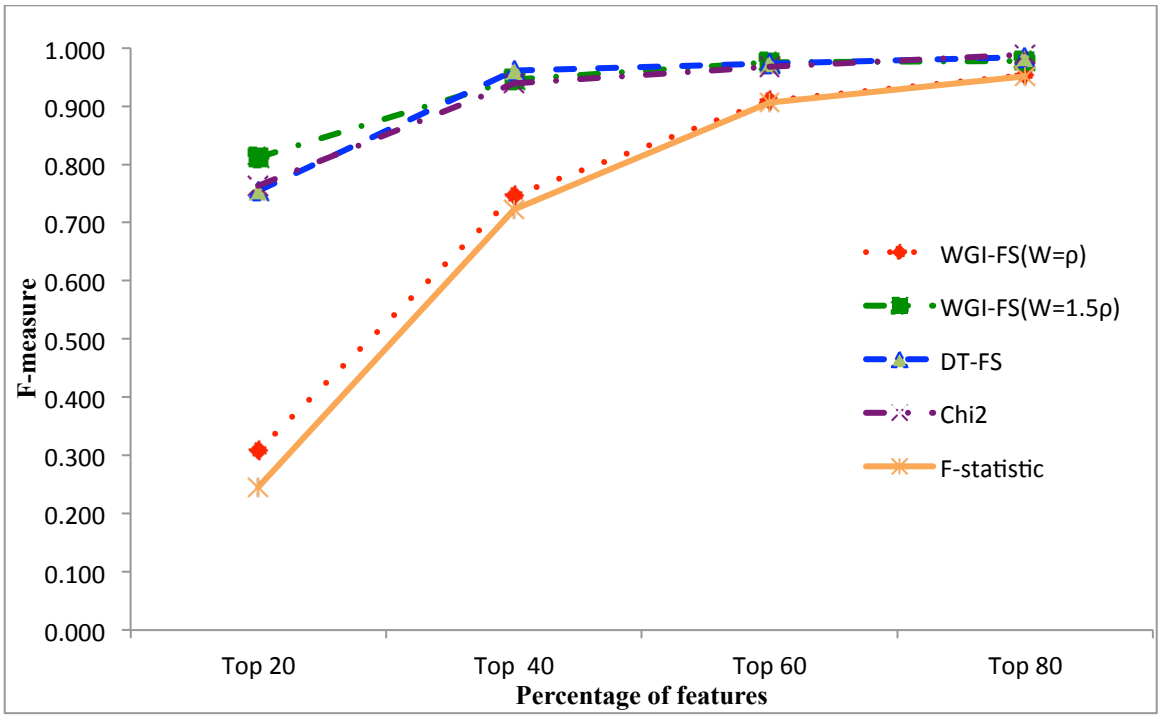


Figure 4.9 Performance of F-measure vs. feature ranking (Letter dataset).

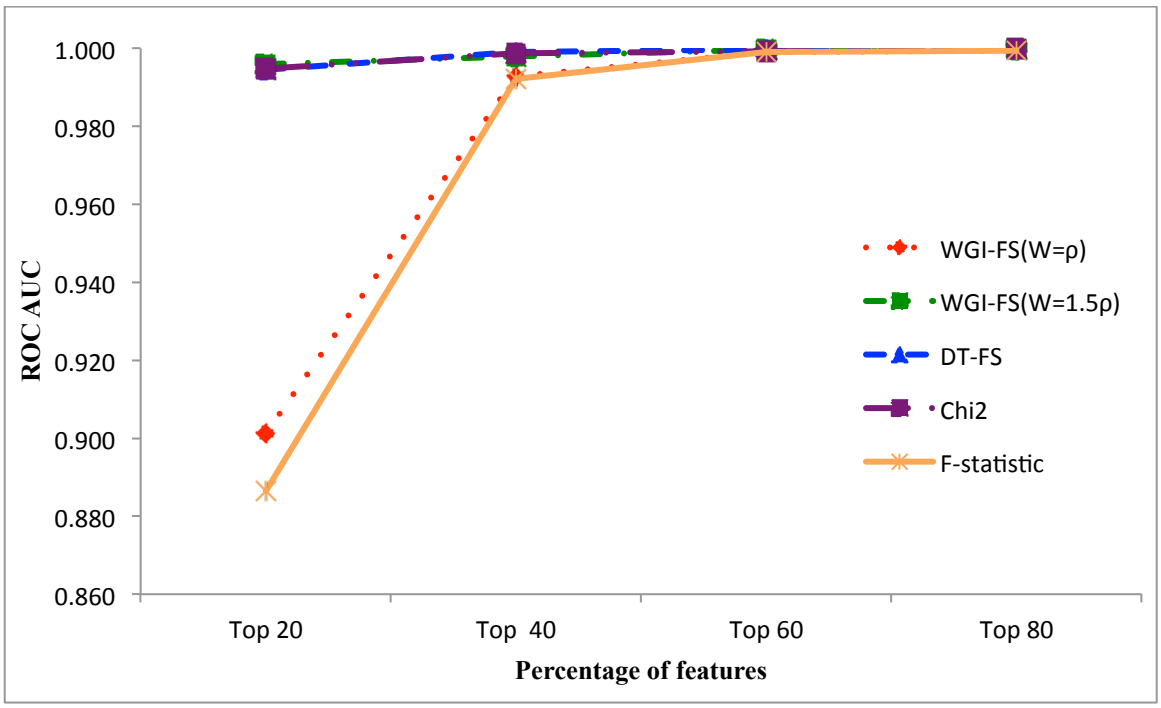


Figure 4.10 Performance of ROC AUC vs. feature ranking (Letter dataset).

4.5 Case Study 3: Statlog Dataset

In this section, the result of comparing WGI-FS(W=ρ), WGI-FS(W=1.5ρ), DT-FS, Chi-square and F-statistic feature selection methods are given and discussed. This dataset contains 6,435 instances with 36 features. 6-fold cross validation is applied in the experiment as well.

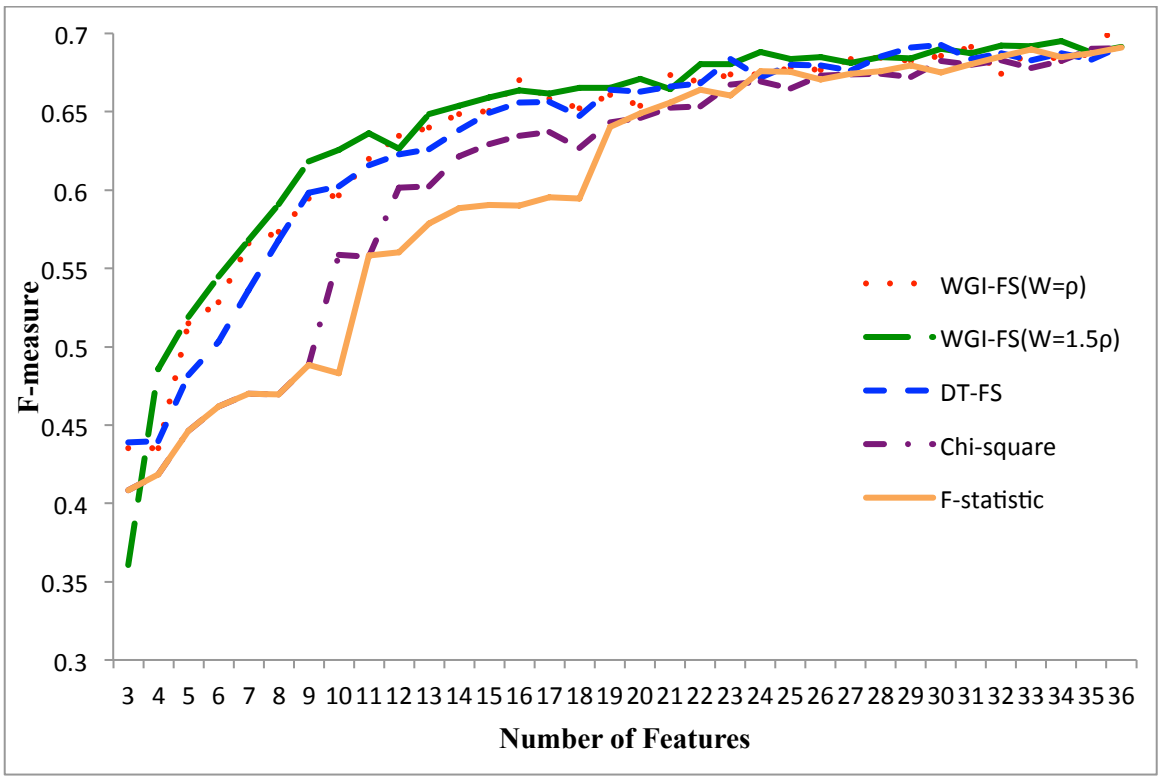


Figure 4.11 Performance of five feature selection methods in terms of F-measure (Statlog dataset).

As shown in Figure 4.11, we compare the tendency of F-measure among the five different feature selection approaches based on the number of sequential trees that equals 170. The number of features is selected from 3 to 36. 36 is the dimension of the Statlog dataset. We conclude that our proposed methods (WGI-FS and DT-FS) have the better F-measure than Chi-square and F-statistic. Then we compare the performance of F-measure among our proposed methods. The results show that WGI-FS(W=1.5ρ) achieves the better result than the others.

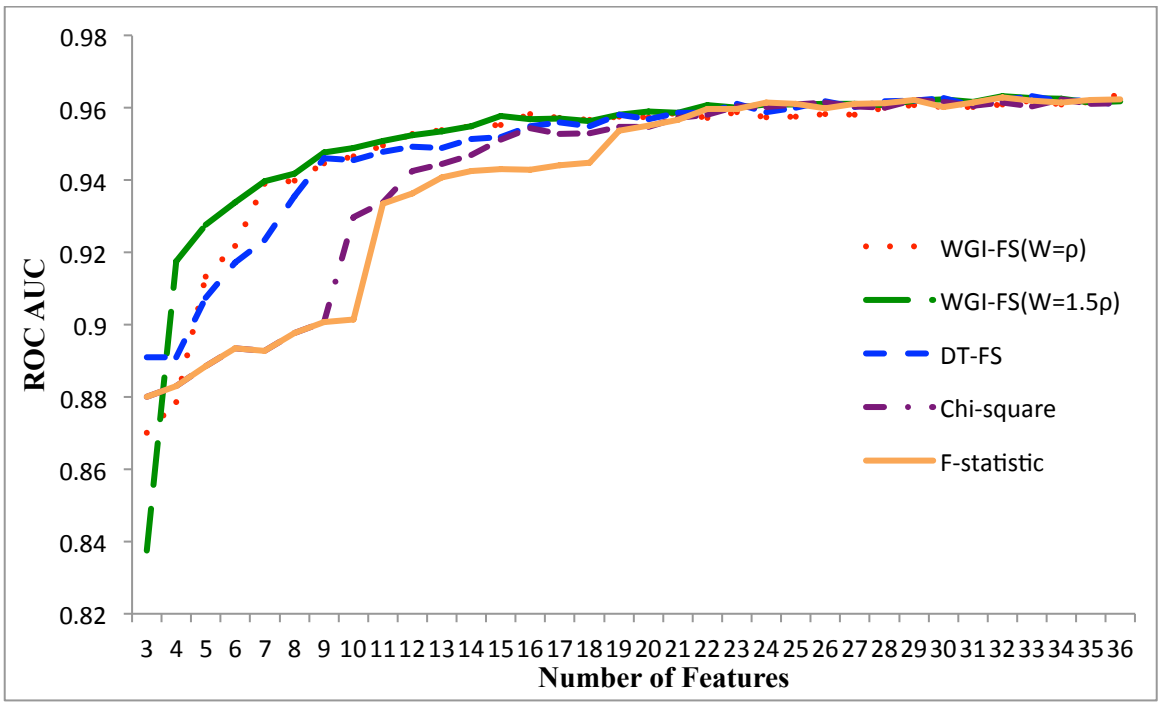


Figure 4.12 Performance of five feature selection methods in terms of ROC AUC (Statlog dataset).

In Figure 4.12, we compare the tendency of ROC AUC among the five different feature selection methods. It shows that WGI-FS(W=1.5ρ) achieves the best performance among all five methods. Also, WGI-FS, DT-FS and Chi-square need almost same number of features for reaching the maximum ROC AUC. It means that when the number of selected features is larger than a certain number, there is no difference to use any feature selection method among WGI-FS, DT-FS and Chi-square based on ROC AUC measurement. If a small subset of feature is selected, WGI-FS is a better method than others.

In Tables 4.7 and 4.8, we choose 20 percent of features as an example, to show the F-statistic and ROC AUC result for 6-fold cross validation. According to the results from the tables, we conclude that WGI-FS(W=1.5ρ) achieves the best performance.

However, some results exhibit only slight or no difference. So, we use boxplot to test whether these results contain significant difference.

Table 4.7 Performance of Five Feature Selection Methods in Terms of F-measure (selecting 20% features)

F-measure	WGI-FS (W=ρ)	WGI-FS (W=1.5ρ)	DT-FS	Chi-square	F-statistc
1	0.567	**0.624**	0.585	0.475	0.475
2	0.619	**0.637**	0.605	0.509	0.509
3	0.572	**0.631**	0.627	0.492	0.492
4	0.607	**0.634**	0.599	0.500	0.500
5	0.583	0.576	**0.592**	0.488	0.488
6	**0.618**	0.608	0.582	0.466	0.466
Mean	0.594	**0.618**	0.598	0.488	0.488

Table 4.8 Performance of Five Feature Selection Methods in terms of ROC AUC (selecting 20% features)

ROC AUC	WGI-FS (W=ρ)	WGI-FS (W=1.5ρ)	DT-FS	Chi-square	F-statistc
1	0.930	**0.933**	0.930	0.876	0.876
2	0.953	**0.960**	0.954	0.908	0.908
3	0.947	**0.955**	0.953	0.913	0.913
4	0.950	**0.952**	0.949	0.904	0.904
5	0.939	0.940	**0.943**	0.906	0.906
6	**0.951**	0.947	0.949	0.898	0.898
Mean	0.945	**0.948**	0.946	0.901	0.901

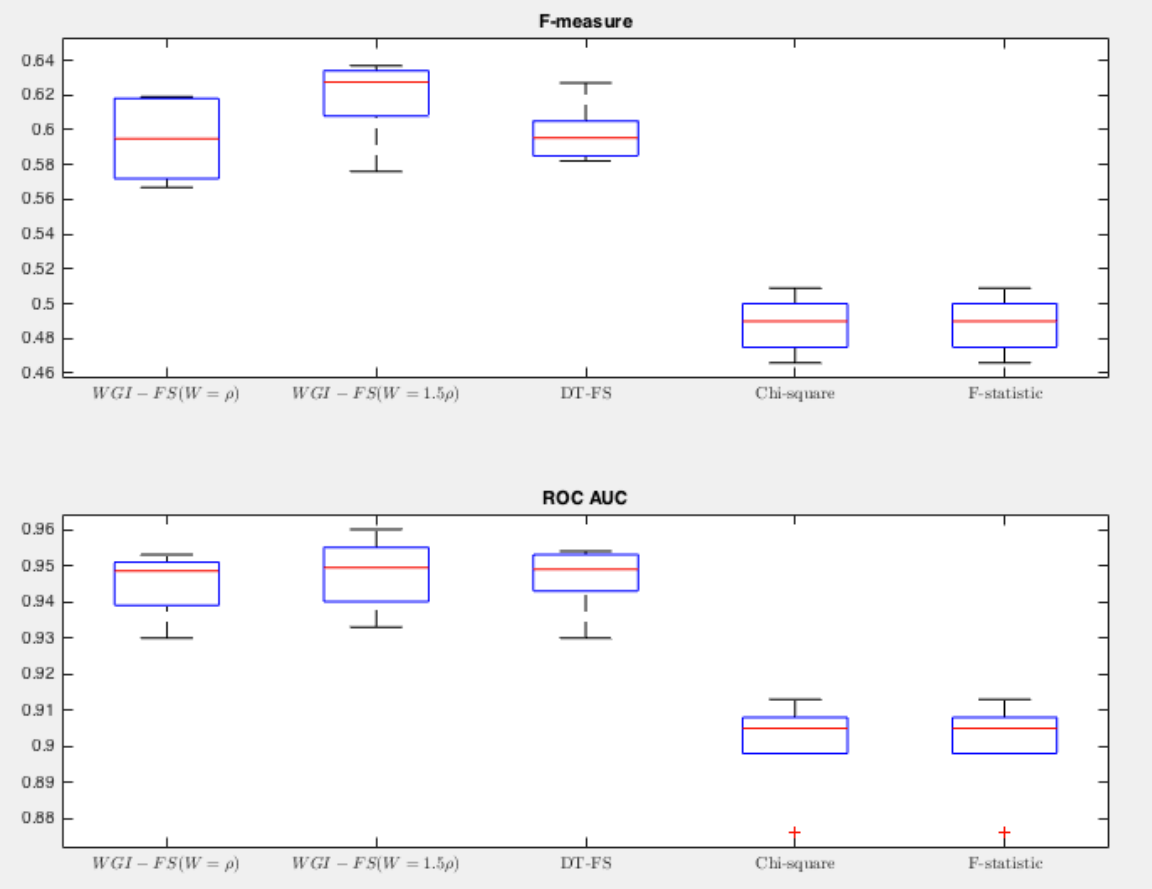


Figure 4.13 Boxplot for F-measure and ROC AUC (Statlog dataset).

Figure 4.13 demonstrate that the WGI-FS(W=1.5ρ) method performs better than other four methods on F-measure. For ROC-AUC, WGI-FS(W=1.5ρ) and DT-FS obtain the same performance, but significantly better than the other three methods. WGI-FS(W=ρ) only gets slightly better result than Chi-square and F-statistic.

In Figures 4.14-4.15, when the top 20% of features are selected, WGI-FS(W=1.5ρ) and DT-FS achieve a better performance on both F-measure and ROC AUC than the others. When we select the top 40% features, the result of F-measure is increased, but the performance of ROC AUC is decreased. The same superb performance of ROC AUC is realized, when top 60% features are selected based on these five methods. The only

difference among these five methods is F-measure. Our purposed method WGI-FS(W=1.5ρ) and DT-FS obtain the highest F-measure among all the five ones.

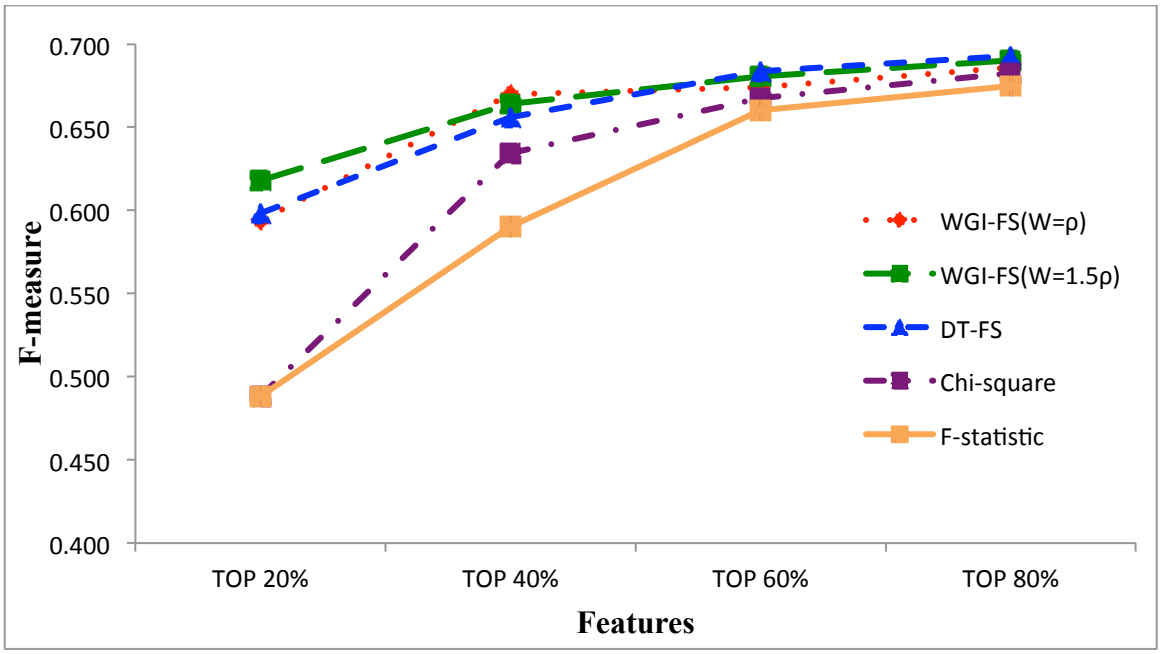


Figure 4.14 Performance of F-measure vs. feature ranking (Statlog dataset).

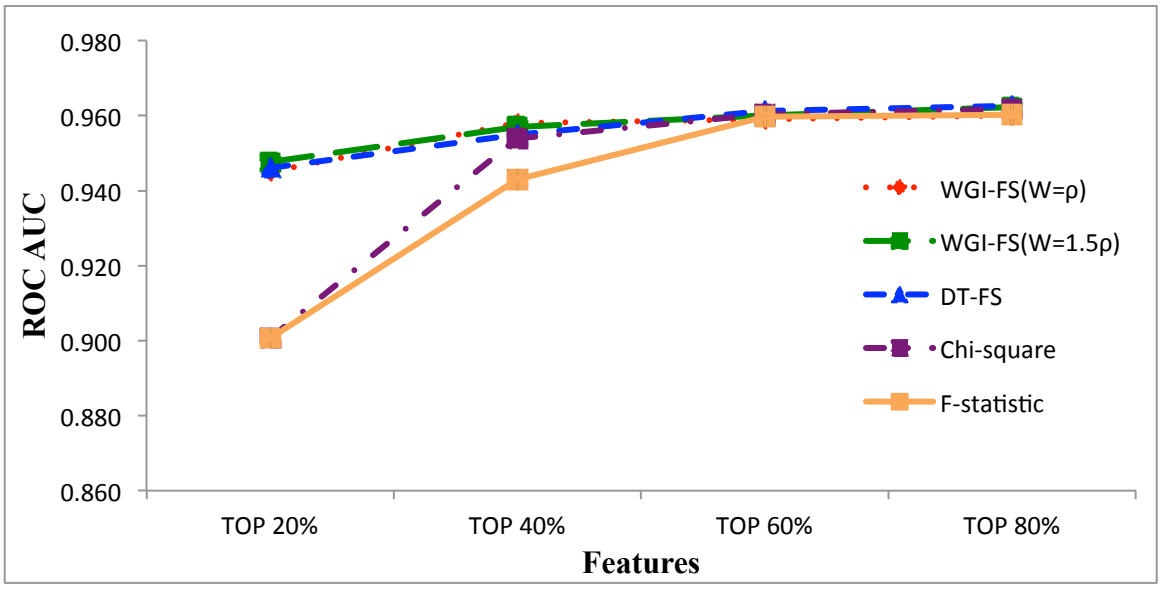


Figure 4.15 Performance of ROC AUC vs. feature ranking (Statlog dataset).

4.6 Summary

In summary, our proposed methods WGI-FS(W=1.5ρ) and DT-FS, compared to Chi-square and F-statistic feature selection methods, perform very well in terms of both ROC AUC and F-measure. They outperform the other methods, when a small subset of features is selected and used. As the number of selected features increases, Chi-square can achieve the similarly good result. In addition, we can observe that the F-statistic achieves the worst performance among all the five methods.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Summary of Contribution of This Thesis

This thesis proposes two feature selection methods that are based on a decision tree rule feature selection (DT-FS) method and based on weighted Gini index feature slection (WGI-FS) method. They can be used to deal with imblanced classification problems, which are commonly encountered in the real world datasets are imbalanced. This thesis makes the following contributions.

(1) Making literature review about imbalanced data, decision tree and feature selection.

Imbalanced data is one type of datasets that are frequently found in real world applications, i.e., fraud detection, cancer diagnosis and DNA microarray. For this type of datasets, improving the accuracy to identify the minority class is a critically important issue. Feature selection is one method to address this issue. An effective feature selection method can choose a subset of features that favor in the accurate determination of the minority class. A decision tree is a classifier that can be built up by using a different splitting criterion. Its advantage is the ease of detecting which feature is used as a splitting node. Thus, it is possible to use a decision tree splitting criterion as a feature selection method.

(2) Proposing two feature selection method: DT-FS and WGI-FS

For DT-FS method, we contain well performance on ROC AUC as using full features, when reducing 90% of features. For the WGI-FS method, we select a weight to

revise the original Gini index, which is set to be the imbalanced ratio and 1.5 times of that, respectively. As a result, we can effectively increase the minority classification accuracy.

(3) Conducting comparisons between the proposed methods and two filter-based methods, i.e., Chi-square and F-statistic feature selection methods and analyzing experimental results.

DT-FS, Chi-square and F-statistic methods are tested through the Santander bank dataset. According to the experimental results, DT-FS can select those highly relevant features for a large-scale imbalanced dataset. The better ROC AUC can be achieved by using those features chosen from DT-FS. DT-FS outperforms Chi-Square and F-statistic at the expense of slightly more computation time.

DT-FS, WGI-FS(W=ρ), WGI-FS(W=1.5ρ), Chi-square and F-statistic methods are tested by using two UCI datasets. Based on the experimental results, WGI-FS(W=1.5ρ) and DT-FS can well outperform the others in terms of ROC AUC and F-measure especially when a small subset of features, e.g., 20% of all features, is selected and utilized.

5.2 Limitations and Future Work

There are some limitations in our methods. A decision tree splitting criterion is a sensitive method. It means that a splitting node is easily changed with some instances' value. It makes the results of our feature selection methods change as we alter the value of instances. Also, it is hard to distinguish which feature should be first handled or selected, when multiple features achieve the same score. This work has just explored the

cases when the weight of WGI-FS equals the imbalanced ratio or 1.5 times of that. Will the other weights perform better? This question remains open.

One of interesting future studies is to find a way to remove the redundant features. Another direction is to detect the noise and remove them before the next stage of classifier training. Determining the optimal weight in WGI-FS remains unsolved.

REFERENCES

Apté, C., Damerau, F., and Weiss M. S.. "Automated learning of decision rules for text categorization." *ACM Transactions on Information Systems (TOIS)* 12, no. 3 (1994): 233-251.

Breiman, L., Friedman J., Charles J. S., and Richard A. O.. "Classification and regression trees". CRC press, 1984.

Chen, X.W. and Wasikowski, M.. "Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems." In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 124-132. ACM, 2008.

Chawla, N.V., Japkowicz, N., and Kotcz, A., "Editorial: special issue on learning from imbalanced data sets." *ACM Sigkdd Explorations Newsletter* 6, no. 1 (2004): 1-6.

Cuadra, L., Alexandre, E., Alvarez, L., and Rosa-Zurera, M.. "Reducing the computational cost for sound classification in hearing aids by selecting features via genetic algorithms with restricted search." In *Processing of International Conference on Audio, Language and Image(ICALIP)*, 2008, pp. 1320-1327. *IEEE*, 2008.

Chen, X., Ba, Y., Ma L., Cai, X., Yin Y., Wang, K., Guo, J. et al. "Characterization of microRNAs in serum: a novel class of biomarkers for diagnosis of cancer and other diseases." *Cell Research* 18, no. 10 (2008): 997-1006.

Castillo, C., Donato, D., Gionis, A., Murdock, V., Silvestri, F. Know your neighbors: Web spam detection using the web topology. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and Development in Information Retrieval* 2007 Jul 23 (pp. 423-430). *ACM*.

Chen, X., Wang, M., and Zhang, H.. "The use of classification trees for bioinformatics." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1, no. 1 (2011): 55-63.

Chandrashekar, G., and Sahin F.. "A survey on feature selection methods." *Computers & Electrical Engineering* 40, no. 1 (2014): 16-28.

Chen, T.Q., and He T.. "Xgboost: extreme gradient boosting." *R Package Version 0.4-2* (2015).

Chen, T.Q., and Guestrin, C.. "Xgboost: A scalable tree boosting system." In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785-794. ACM, 2016.

Donoho, D.L.. "High-dimensional data analysis: The curses and blessings of dimensionality." *AMS Math Challenges Lecture* 1, 32, 2000

Esposito, F., Malerba D., and Semeraro G.. "Decision tree pruning as a search in the state space." In *Machine Learning*: *ECML-93*, pp. 165-184. *Springer Berlin/Heidelberg*, 1993.

Fumera, G. and Roli, F.. "Support vector machines with embedded reject option." Pattern recognition with support vector machines. *Springer Berlin Heidelberg*, 2002. 68-82.

Fawcett, T.. "An introduction to ROC analysis." *Pattern Recognition Letters* 27, no. 8 (2006): 861-874.

Forman, G.. "An extensive empirical study of feature selection metrics for text classification." *Journal of Machine Learning Research*, no. 3, Mar (2003): 1289-1305.

Guyon, I., and Elisseeff, A.. "An introduction to variable and feature selection." *Journal of Machine Learning Research*, no. 3, Mar (2003): 1157-1182.

Gheyas, I. A., and Smith L. S.. "Feature subset selection in large dimensionality domains." *Pattern Recognition* 43, no. 1 (2010): 5-13.

Guyon, I., and Elisseeff, A.. "An introduction to variable and feature selection." Journal of Machine Learning Research , no. 3, Mar (2003): 1157-1182.

Han, H., Wang , W.Y., and Mao, B.H., "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning." *Advances in Intelligent Computing* (2005): 878-887.

He, H.B., and Edwardo A. G.. "Learning from imbalanced data." *IEEE Transactions on Knowledge and Data Engineering* 21, no. 9 (2009): 1263-1284.

He, X.F., Cai, D., and Niyogi, P.. "Laplacian score for feature selection." In *Neural Information Processing Systems*, vol. 186, p. 189. 2005.

Jolliffe, I. T.. "Principal component analysis. 1986." *Spring-verlag*, New York (1986).

Kang, Q., Liu, S.Y., Zhou, M.C., and Li, S.S.. "A weight-incorporated similarity-based clustering ensemble method based on swarm intelligence." *Knowledge-Based Systems* 104 (2016): 156-164.

Kohavi, R. and George H.J.. "Wrappers for feature subset selection."*Artificial intelligence* 97, no. 1-2 (1997): 273-324.

Koh, H. C.e, Tan, W. C., and Peng, G. C.. "Credit scoring using data mining techniques." *Singapore Management Review* 26, no. 2 (2004): 25.

Kohavi, R., and George H. J.. "Wrappers for feature subset selection."*Artificial Intelligence* 97, no. 1-2 (1997): 273-324.

Liu, X.Y., Wu, J., and Zhou, Z.H.. "Exploratory undersampling for class-imbalance learning."*IEEE Transactions on Systems, Man, and Cybernetics*, *Part B (Cybernetics)* 39, no. 2 (2009): 539-550.

Li, R., and Y. Hu. "A density–based method for reducing the amount of training data in kNN text classification." *Computer Research and Development* 45, no. 4 (2004): 539-544.

Ladha, L., and Deepa, T.. "Feature selection methods and algorithms." *International journal on Computer Science and Engineering* 3, no. 5 (2011): 1787-1797.

Lewis, D.D., and William A. G.. "A sequential algorithm for training text classifiers." In *Proceedings of the 17th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 3-12. Springer-Verlag New York, Inc., 1994.

Liu, H., and Hiroshi M.. "Feature selection for knowledge discovery and data mining". Vol. 454. *Springer Science & Business Media*, 2012.

Maldonado, S., and Weber R.. "A wrapper method for feature selection using support vector machines." *Information Sciences* 179, no. 13 (2009): 2208-2217.

Montgomery, J. C., and Bodznick, D.. "An adaptive filter that cancels self-induced noise in the electrosensory and lateral line mechanosensory systems of fish." *Neuroscience letters* 174, no. 2 (1994): 145-148.

Martínez, A. M., and Kak, A. C.. "Pca versus lda." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23, no. 2 (2001): 228-233.

Oh, I. S., Lee, J.S., and Moon, B.R.. "Local search-embedded genetic algorithms for feature selection." In *Proceedings of 16th International Conference on Pattern Recognition*, 2002. vol. 2, pp. 148-151. *IEEE*, 2002.

Otero, F.E., Alex, A. F., and Colin, G. J.. "Inducing decision trees with an ant colony optimization algorithm." *Applied Soft Computing* 12, no. 11 (2012): 3615-3626.

Punlumjeak, W., and Rachburee, N.. "A comparative study of feature selection techniques for classify student performance." In *Information Technology and Electrical Engineering (ICITEE), 2015 7th International Conference on*, pp. 425-429. IEEE, 2015.

Peng, H.C., Long F.H., and Ding C.. "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, no. 8 (2005): 1226-1238.

Quinlan, J. R.. "Induction of decision trees." *Machine Learning* 1, no. 1 (1986): 81-106.

Quinlan, J. R.. "Constructing decision tree." *C4.5* (1993): 17-26.

Rokach, L., and Maimon, O.. "Top-down induction of decision trees classifiers-a survey."*IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35, no. 4 (2005): 476-487.

Roweis, S. T., and Saul, L.K.. "Nonlinear dimensionality reduction by locally linear embedding." *Science* 290, no. 5500 (2000): 2323-2326.

Saeys, Y., Inza, I., and Larrañaga, P.. "A review of feature selection techniques in bioinformatics." *Bioinformatics* 23, no. 19 (2007): 2507-2517.

Wilson, J.,Chawla, J., and Fisher M.. "Sensitivity and specificity of electrodiagnostic criteria for CIDP using ROC curves: comparison to patients with diabetic and MGUS associated neuropathies." *Journal of the Neurological Sciences* 231, no. 1 (2005): 19-28.

Wang, F., Xu, T., Tang, T., Zhou, M.C., and Wang, H.. “Belevel feature extraction-based text mining for fault diagnosis of railway systems,” *IEEE Trans. on Intelligent Transportation Systems*, 18(1), pp. 49-58, Jan. 2017.

Yang, Z., Hu, B., Zhang, Y., Luo, Q., and Gong, H.. "Development of a plastic embedding method for large-volume and fluorescent-protein-expressing tissues." *Public Library of Science one* 8, no. 4 (2013): e60877.

Yen, S.J., and Lee, Y.S.. "Cluster-based under-sampling approaches for imbalanced data distributions." *Expert Systems with Applications* 36, no. 3 (2009): 5718-5727.

Yu, L., and Liu, H.. "Feature selection for high-dimensional data: A fast correlation-based filter solution." In *International Conference on Machine Learning*, vol. 3, pp. 856-863. 2003.