Université de Tunis

Institut Supérieur de Gestion

Ecole Doctorale Sciences de Gestion

LARODEC

On Feature Selection Methods for Credit Scoring

THESE

En vue de l'obtention du Doctorat en

Informatique de Gestion

Présentée et soutenue publiquement par

Mlle Bouaguel Waad

Le 22 Janvier 2015

Membres du Jury

Mr. AZAIEZ Mohamed Nacer	Professeur	Université de Tunis	Président
Mr. LIMAM Mohamed	Professeur	Université de Tunis	Directeur de Thèse
Mr. AYADI Mohamed	Professeur	Université de Tunis	Rapporteur
Mme. CHAOUACHI Jouhaina	Maître de Conférence	Université de Carthage	Rapporteur
Mr. BEN MAATOUG Abderrazak	Maître de Conférence	Université de Tunis	Membre

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Professor Mohamed Limam, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research as well as on my career have been priceless. This thesis would not have been possible without your help, support and patience.

Furthermore I would also like to thank with much appreciation Dr. Ghazi Bel Mufti for the useful comments, remarks and engagement through the learning process of this thesis.

I take this opportunity to express my deepest gratitude to the thesis committee members. It is my honor that they accepted to be members of this committee and spent their precious time helping me to improve this thesis.

I would especially like to acknowledge the financial, academic and technical support of the ISG, university of Tunis and I also thank all LARODEC members for their support and assistance since the start of my masters' work.

A special thanks to my family. Words cannot express how grateful I am to my mother, father, brother and sister for all the sacrifices that you've made on my behalf. Your prayer for me was what sustained me thus far.

I would like to dedicate my thesis and to express my deepest appreciation to my beloved fiancé who always was my support in all times.

Abstract

Credits' granting is a fundamental question for which every credit institution is confronted and one of the most complex tasks that it has to deal with. This task is based on analyzing and judging a large amount of receipts credits' requests. Typically, credit scoring databases are often large and characterized by redundant and irrelevant features. With so many features, classification methods become more computational demanding. This difficulty can be solved by using feature selection methods. Many such methods are proposed in literature such as filter and wrapper methods. Filter methods select the best features by evaluating the fundamental properties of data, making them fast and simple to implement. However, they are sensitive to redundancy and there are so many filtering methods proposed in previous work leading to the selection trouble. Wrapper methods select the best features according to the classifier's accuracy, making results well-matched to the predetermined classification algorithm. However, they typically lack generality since the resulting subset of features is tied to the bias of the used classifier. The purpose of this thesis is to build simple and robust credit scoring models based on selecting the most relevant features. Three feature selection methods are proposed. First we propose a new filter rank aggregation based on an optimization method using genetic algorithms and similarity. Second, we introduce an ensemble wrapper feature selection method based on an improved exhaustive search. Combining both methods seems a natural choice to benefit from their advantages and avoid their shortcomings. Thus, a three stage feature selection using quadratic programming is considered. Based on different performance criteria and on four real credit datasets our three methods are evaluated. Results show that feature subsets selected by the proposed methods are either superior or at least as adequate as those selected by their competitor methods.

Keywords: Feature selection, filter, wrapper, hybrid, rank aggregation.

Résumé

L'octroi de crédit est une question fondamentale à laquelle chaque établissement de crédit est confronté. Il s'agit de l'une des tâches les plus complexes qu'il doit traiter. Cette tâche est basée sur l'analyse et le jugement d'une grande quantité de demandes de crédit reçues. Généralement, les bases de données utilisées en credit scoring sont très grandes et se caractérisent par la présence de variables redondantes et non significatives. Avec tant de variables, les méthodes de classification deviennent plus complexes. Cette difficulté peut être résolue en utilisant des méthodes de sélection de variables. De nombreuses méthodes de sélection de variables ont été proposées en littérature dont les méthodes filtre et wrapper. D'une part les méthodes filtre choisissent les meilleures variables en évaluant les propriétés fondamentales des données, ce qui les rend rapides et faciles à mettre en œuvre. Cependant, ils ne tiennent pas en compte de la redondance entre les variables. De plus la multitude des méthodes filtre proposées dans les travaux antérieurs pose le problème de choix de la méthode la plus appropriée. D'autre part les méthodes wrapper choisissent les meilleures variables selon le taux de classification généré par un classifieur, de ce fait le résultat est bien adopté à l'algorithme de classification utilisé. Cependant, ces méthodes manquent de généralité puisque le sous-ensemble résultant de variables est biaisé par le classifieur utilisé. Le but de cette thèse est de construire des modèles de credit scoring simples et robustes tout en sélectionnant les variables les plus pertinentes. D'abord nous proposons une nouvelle méthode filtre d'agrégation de rangs basée sur l'optimisation, les algorithmes génétiques et la similarité. Dans un second temps, nous présentons une méthode d'ensemble wrapper de sélection de variables basée sur une recherche exhaustive améliorée. La combinaison des deux méthodes semble un choix naturel pour profiter de leurs avantages et éviter leurs défauts. Ainsi, nous proposons une troisième méthode de sélection à trois niveaux utilisant la programmation quadratique. En se basant sur différents critères de performance et sur quatre bases de données réelles de crédit, nous avons évalué nos trois méthodes. Les résultats obtenus par les sous-ensembles de variables choisis par les méthodes proposées sont meilleurs ou au moins aussi pertinents que ceux donnés par les méthodes concurrentes.

Mots clés: Sélection de variables, filter, wrapper, hybride, agrégation des rangs.

List of Abbreviations and Symbols Used

CS	Credit Scoring.
DA	Discriminant Analysis.
DT	Decision Tree.
LR	Logistic Regression.
SVM	Support Vector Machines.
ANN	Artificial Neural Networks.
KNN	K-Nearest-Neighbor.
LinR	Linear Regression.
$\mathbf{\Omega}$	The population of credit applicants.
χ	The space of observations.
n	Number of instances.
x	Matrix of observations.
d	Number of features.
Y	Vector of class labels.
X	Original features set.
w	Weight vector associated with the features.
W	Weight vector associated with the ranked lists.
γ	Stopping criterion.
F	Features subset.
m	Number of filters to be aggregated.
L	List of ranked features.
r	Feature rank.
σ	Optimal rank list.
D	Distance function.
k	Cardinality of a ranked list.
GA	Genetic Algorithms.
MaxRel	Maximal Relevance.

mRMR	minimal-Redundancy-Maximum-Relevance.
Q	Matrix describing the coefficients of the quadratic terms.
Z	d-dimensional row vector describing the coefficients of the linear terms.
α	Tradeoff between relevance and redundancy.
PCC	Pearson correlation coefficient.
MI	Mutual information.
χ^2	Chi-squared test.
$\mathbf{O_i}$	Observed frequency.
$\mathbf{E_i}$	Expected theoretical frequency.
ANOVA	Analysis of variance.
RSFS	Fough set feature selection.
CFS	Correlation-based feature selection.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Introduction

Motivations for feature selection in credit scoring

Failures of financial institutions are generally related to their inability of controlling an ensemble of financial risks. Different kinds of risks exist but the most important one is credit risk. Credit granting decision is an important and widely studied topic in the lending industry. The set of decision models and their underlying methods that serve lenders in granting consumer credits are called credit scoring (CS) (Zhang et al. 2010).

The general scheme in CS is to use the credit history of previous customers to compute the new applicant's defaulting risk (Tsai and Wu 2008; Thomas 2009). The collected portfolio, i.e. collection of booked loans, is used to build a CS model that would be used to identify the association between the applicant's characteristics and how good or bad is the credit worthiness of the applicant. Generally, portfolios used for the scoring task are voluminous and they are in the range of several thousands. These portfolios are characterized by noise, missing values by redundant or irrelevant features and complexity of distributions (Piramuthu 2006). The number of considered features is called data dimension and high dimensionality in the feature space has advantages but also some serious shortcomings. In fact, as the number of features increases more computation is required and model accuracy and scoring interpretation are reduced (Liu and Schumann 2005; Howley et al. 2006). One solution is to perform a feature selection on the original data.

Feature selection is a term commonly used in machine learning to describe existing

set of methods to reduce a dataset to a convenient size for processing and investigation. This process involves not only a pre-defined cutoff on the number of features that can be considered when building a credit model but also the choice of appropriate features based on their relevance to the study (Fernandez 2010). Further, it is often the case that finding the correct subset of predictive features is an important problem in its own right.

Research questions in feature selection

Three main classes of feature selection are identified in the literature (Rodriguez et al. 2010): filter, wrapper and hybrid feature selection methods.

Usually, filter methods choose the best features by using some informative measure. Various filtering methods and their modifications are proposed in the literature leading to the selection trouble of how to choose the best criterion for a specific feature selection task (Wu et al. 2009). This question is still an open research field. In order to handle this issue, ensemble methods, i.e rank aggregation, could be an interesting solution (Dietterich 2000; Dittman et al. 2013). Aggregation methods provide robust results where the issue of selecting the appropriate filter is alleviated to some level (Saeys et al. 2008). However, in many cases, where rankings are incomplete or highly similar features are given divergent rankings, effective rank aggregation becomes a difficult task (Sculley 2007). These difficulties may be addressed by considering similarity between features in various ranked lists in addition to their rankings. The intuition is that similar features should receive similar rankings given an appropriate measure of similarity.

Filter feature selection does not take into account the properties of the classifier, as it performs usually statistical tests on variables. Therefore, results obtained from a wrapper are different from that of a filter because the former actually takes into consideration the classifier proprieties. In fact, using a single classifier in the wrapper evaluation process may influence the final selection result because each particular classifier has its own specificity and nature (Chrysostomou 2008). When the classifier is changed, due of its bias, the result may differ in terms of the amount of time, the

accuracy and the number of selected features. As such, a possible remedy for this drawback is to use an ensemble of classifiers and combine their outcomes. Nevertheless, we have a reduced knowledge about the effects of using multiple classifiers on feature selection applied to CS tasks, specially the effects of using different number of classifiers with similar or different nature (Chrysostomou et al. 2008). Then, we focus on how the number and the nature of the used classifiers affect the number of selected features and the accuracy of the credit model.

In order to find the best subset of features to be evaluated, the ideal approach is to perform a complete search in the whole search space (Chan et al. 2010). However, searching all possibilities is sometime unrealistic (Liu and Yu 2005). Hence, in order to minimize the number of evaluations done by the classifier, and at the same time maintain the accuracy, we look for a combined search algorithm that reduces the number of possible candidates using a mixture between complete and heuristic search methods.

Usually, hybrid feature selection methods, combining the two discussed approaches, are needed to serve more complicated purposes, (Wu et al. 2009). In fact, constructing a hybrid feature selection process benefitting from advantages of filters and wrappers is a very interesting research question. The challenge here is how to make these two methods work together in order to hid the shortcomings of each one.

Thesis structure

This thesis is organized as follows: in Chapter 2 we review the necessary background for this work and the relevant literature. In Chapter 3 we propose a new rank aggregation approach based on optimization, genetic algorithm (GA) and similarity for CS. In Chapter 4 we introduce an ensemble wrapper feature selection based on an improved exhaustive search for CS. Chapter 5 presents a hybrid three-stage feature selection approach using quadratic programming for CS. Finally, Chapter 6 summarizes the key findings along with their limitations and underlines some possible future research topics.

Chapter 1

Overview of Feature Selection in Credit Scoring

Contents

1.1 Introduction

Feature selection is a fundamental topic in CS. As such, this chapter will provide an overview of CS in Section (1.2). Then, in order to provide a foundation to the most commonly used feature selection methods in CS, a brief introduction of the basics of feature selection is given in Section (1.3) namely search direction, search strategy, evaluation function and stopping criterion. Subsequently, Section (1.4) explains how feature selection is performed using filter, wrapper and hybrid feature selection with some examples and a brief comparison between theses three approaches. Then, Section (1.5) introduces the used datasets throughout this thesis. In Section (1.6) the performance measures are given.

1.2 Credit scoring: state of the art

Credit risk is one of the major issues in financial research (Matjaz 2012; Jiang 2009). Over the past few years, many companies fell apart and were forced into bankruptcy or to a significantly constrained business activity because of the deteriorated financial and economic situation (Haizhou and Jianwu 2011). When banks are unprepared to a variation in the economic activity they will probably suffer from huge credit losses. In fact it is very obvious that credit risk increases in economic depression. However, this effect could increase when bank experts under or over estimate the creditworthiness of credit applicants. Expressing why some companies or individuals do default while others don't and what are the main factors that drive credit risk and how to build robust credit model is very important for financial stability.

1.2.1 Background of credit scoring

CS is basically a way of recognizing the different groups in a population when one cannot see the characteristics that separate the groups but only related ones (Thomas 2000). This idea of differentiating between groups in the same population was first introduced in statistics by Fisher (1936). He wanted to distinguish between three varieties of iris by measurements of the physical size of the plants. Then Durand (1941) was the first to recognize that one could use the same techniques to discriminate between good and bad loans. His research was done in the context of a research project for the US National Bureau of Economic Research. Since then, CS was a true success and banks started using it for their predictive activities (Thomas et al. 2002).

CS consists of the evaluation of the risk related to lending money to an organization or a person. In the past few years, the business of credit products increased enormously. Approximately every day, individual's and company's records of past lending and repaying transactions are collected and evaluated (Hand and Henley 1997). This information is used by lenders such as banks to evaluate an individual's or company's means and willingness to repay a loan. According to Yang (2001) the set of collected information makes the deciders task simple because it helps determine: whether to extend credit duration or to modify a previously approved credit limit and to quantify the probability of default, bankruptcy or fraud associated to a company or a person. When assessing the risk related to credit products, different problems arise depending on the context and the different types of credit applicants. Sadatrasoul et al. (2013) summarize different kinds of scoring as follows: application scoring, behavioral scoring, collection scoring and fraud detection.

Application scoring

Application scoring refers to the assessment of the credit worthiness for new applicants. It quantifies the risks associated with credit requests by evaluating the social, demo-graphic, financial and other data collected at the time of the application.

Behavioral scoring

Behavioral scoring involves principles that are similar to application scoring with the difference that it refers to existing customers. As a consequence, the analyst already has evidence of the borrower's behavior with the lender. Behavioral scoring models analyze the consumer's behavioral patterns to support dynamic portfolio management processes.

Collection scoring

Collection scoring is used to divide customers with different levels of insolvency into groups, separating those who require more decisive actions from those who don't need to be attended to immediately. These models are distinguished according to the degree of delinquency (early, middle, late recovery) and allow a better management of delinquent customers, from the first signs of delinquency (30-60 days) to subsequent phases and debt write-off.

Fraud detection

Fraud scoring models rank the applicants according to the relative likelihood that an application may be fraudulent.

We will address the application scoring problem also known as consumer CS. In this context the term *credit* will be used to refer to an amount of money that is borrowed to a credit applicant by a financial institution and which must be repaid with interest in a regular interval of time. The probability that an applicant will default must be estimated from information about the applicant provided at the time of the credit application and the estimate will serve as the basis for an accept or a reject decision. According to Sadatrasoul et al. (2013), accurate classification is of benefit both to the creditor in terms of increased profit or reduced loss and to the applicant in terms of avoiding over commitment. Deciding whether or not to grant a credit is generally carried out by banks and various other organizations. It is an economic activity which has seen rapid growth over the last 30 years.

Traditional methods of deciding whether to grant credit to a particular individual use human judgment of the risk of default based on experience of previous decisions (Thomas et al. 2002). Nevertheless, economic demands resulting from the arising number of credit requests, joined with the emergence of new machine learning methods, have led to the development of sophisticated models to help the credit granting decision.

Statistical CS models, called scorecards or classifiers, use predictors from application forms and other sources to estimate the probabilities of defaulting. A credit granting decision is taken by comparing the estimated probability of defaulting with a suitable threshold (Bardos 2001). Standard statistical methods used in the industry for developing scorecards are discriminant analysis (DA), linear regression (LinR) and logistic regression (LR). Despite their simplicity, Tufféry (2007) and Thomas (2009) show that both DA and LR prediction need strong assumptions on data. Hence, other models based on data mining methods are proposed. These models do not lead to scorecards but they indicate directly the class of the credit applicant (Jiang 2009). Artificial intelligence methods such as decision trees (DT), artificial neural networks (ANN), K-nearest-neighbor (KNN) and support vector machines (SVM) can be used as alternative methods for CS (Bellotti and Crook 2009). These methods extract knowledge from training datasets without any assumption on the data distributions. The classification methods are described in Appendix (A). A brief summary about the used classification methods in this thesis is included in Chapter 3.

1.2.2 Basic notations in credit scoring

In what follows, we present the main notations which will be used in this CS context. Let Ω the population of credit applicants. We denote by χ the space of observations in $\mathbb{R}^d$ defined by the random variable X given by

$$\begin{aligned} X: \quad & \Omega \to \chi \subset \mathbb{R}^d \\ & i \rightsquigarrow x_i = (x_i^1, x_i^2, ..., x_i^d). \end{aligned} \tag{1.1}$$

We have n individuals described by d variables as shown by the matrix $\mathbf{x}$ given below:

$$\mathbf{x} = \begin{bmatrix} x_1^1 & \dots & x_1^d \\ \dots & \dots & \dots \\ x_n^1 & \dots & x_n^d \end{bmatrix}$$

Let $\mathbf{X}$ denote the set of features such that $\mathbf{X} = (X^1, X^2, ..., X^d)$. The n observations are divided into two groups, where the group label of an applicant i is represented through the modalities $\{0, 1\}$ of a binary target variable Y, where the label 0 denotes a bad applicant and 1 a good one. We denote by $Y = (y_1, \dots, y_n)$ the vector of class labels for the n instances. Figure (1.1) summarizes the process of CS and its basic notions.

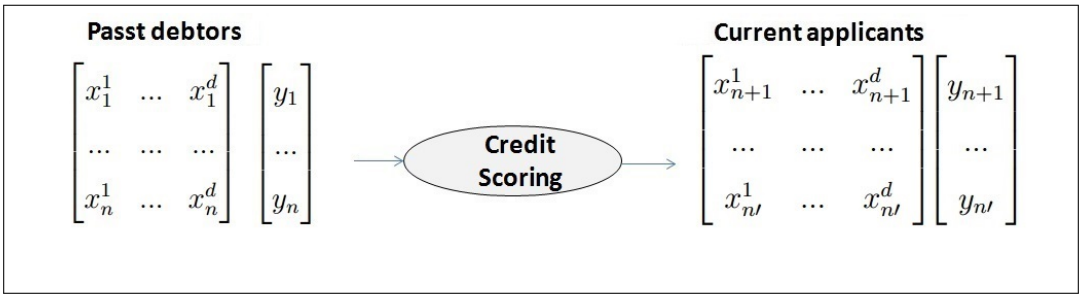


Figure 1.1: The process of credit scoring.

1.2.3 Proprieties of financial data

According to Hand and Henley (1997) CS portfolios are frequently voluminous and they are in the range of several thousands, over 100000 applicants measured by more than 100 variables are quite common. These portfolios are characterized by missing values, complexity of distributions and by redundant or irrelevant features (Piramuthu 2006). Clearly, applicants characteristics will vary from one situation to another. An applicant looking for a small loan will be asked for information which is different from another who is asking for a big loan. Furthermore, the data which may be used in a credit model is always subject to changing legislation (Hand and Henley 1997).

Based on initial application forms filled by credit applicants some are rejected based on some obvious characteristics. Further information is then collected on the

remaining credit applicants using further forms. This process of collection of the borrower's information allows banks to avoid losing time on obvious non worthy applicants as well as allowing quick decisions.

As any classification problem, choosing the number of appropriate features to be included in the credit model is an important task. One might try to use as many features as possible. However, the more the number of features grows the more computation is required and model accuracy and scoring interpretation are reduced (Liu and Schumann 2005; Howley et al. 2006). There are also other practical issues, in fact with too many questions or a lengthy vetting procedure, applicants will deter and will go elsewhere. Based on Hand and Henley (1997), standard statistical and pattern recognition strategy is to explore a large number of features and to identify an effective subsetof those features to be considered for building the credit model.

1.3 Basics of feature selection

There are two famous special methods of dimensionality reduction. The first one is feature extraction where the input data is transformed into a reduced representation set of features, so new attributes are generated from the initial ones. The second category is feature selection. In this category a subset of existing features is selected without a transformation. Generally, feature selection is preferred over feature extraction since it keeps all information about the importance of each single feature while in feature extraction obtained variables are usually not interpretable (Giudici 2003).

Conserving the information of each feature provides much simplicity and interpretability to financial data processing. Hence, feature selection is more appropriate to our study. Feature selection is an important framework in knowledge discovery and specially in financial applications, not only for the insight gained from determining predictive modeling features but also for the improved performance, understandability and accuracy of the credit models.

The idea behind feature selection is to reduce the effect of tricky features in the

dataset, where tricky and unneeded features include:

- irrelevant features are those that can never contribute to improve the predictive accuracy of credit model, where the accuracy is how close a measured value is to the actual or true value. However, the algorithm may mistakenly include them in the model. Removing such features reduces the dimension of the search space and speeds up the learning algorithm.

- redundant features are those that can replace others in a feature subset. They basically bring similar information as other features. For example, a dataset may include two features that provide similar information as date of birth and age. Typically feature redundancy is defined in terms of feature correlation, where two features are redundant to each other if they are correlated.

According to Rodriguez et al. (2010) a successful feature selection: a) reduces the dimensionality of the feature space, b) speeds up and reduces the cost of a learning algorithm and c) obtains the feature subset which is the most relevant to classification. Feature selection algorithms are typically composed of the following four components: search direction, search strategy, evaluation function and the stopping point. Figure (1.2) gives a flowchart presenting the general process of feature selection based on these four components.

1.3.1 Search direction

Choosing the starting point in the process of searching for the most important features is the first issue to be considered when performing a feature selection on the original features set. Once the starting point is defined, the search direction is determined (Liu and Motoda 1998; Yun et al. 2007). The search for the most relevant feature subset may start with an empty set and successively add the most relevant features. In this case, the search direction is called forward direction. On the other hand, the search may begin with the full set and successively removes less relevant features. In this case, the direction is named backward. Other ways of starting points can be

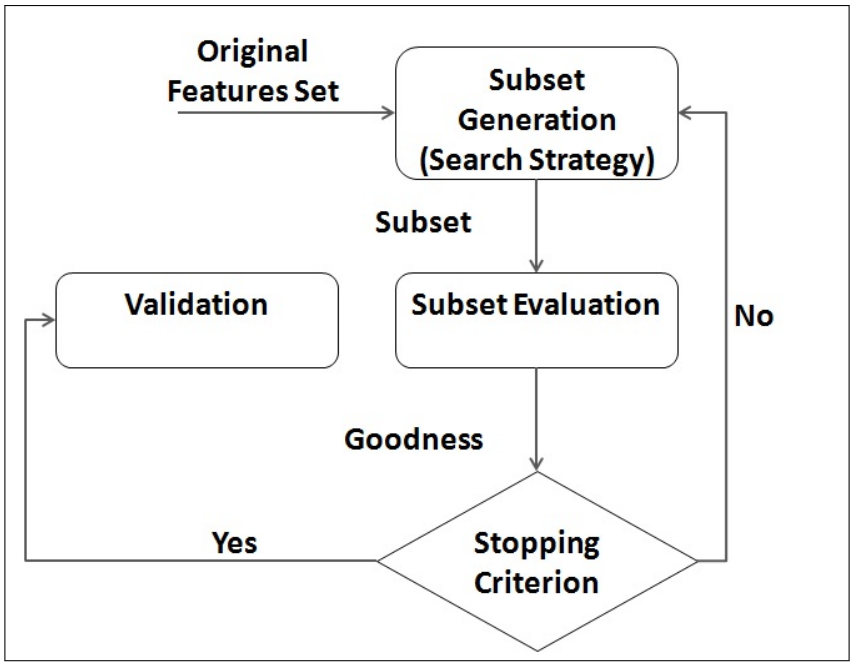


Figure 1.2: Key steps of feature selection process.

used, we may start with both ends and add and remove features at the same time i.e. bidirectional. The search may also begins with a random subset of features in order to avoid being trapped into local optima (Liu and Yu 2005).

1.3.2 Search strategy

Once the starting point and the search direction are decided, the search strategy must be chosen. The search strategy is a fundamental part in the process of subset generation. Typically, for a dataset of d features, 2^d possible subsets are candidates for further examinations (Yun et al. 2007). Even for a moderate d, the search space may be too large for a complete search (Kwang 2002). Consequently, two strategies have been explored in the literature as discussed by Liu and Yu (2005) and also by Legrand and Nicoloyannis (2005): exhaustive and heuristic.

Exhaustive search

An exhaustive search performs a complete search to find all possible features' subsets and picks the optimal subset of features by examining all possible candidate subsets. Since exhaustive search examines all possible subsets, it always guarantees to find the

optimal result. However, as the number of features grows, exhaustive search becomes rapidly impractical because the search space is in the order of $O(2^d)$.

Heuristic search

Naturally a search does not have to be exhaustive in order to guarantee good or acceptable results (Legrand and Nicoloyannis 2005). Heuristic methods are a set of realistic and practical approaches that are easier to put into practice. Still, such search strategy does not always guarantee to produce an optimal solution, but nonetheless a greedy heuristic may yield locally optimal solutions that approximate a global optimal solution. Many research works discussed heuristic search in CS. Wang et al. (2012) proposed a novel approach to feature selection based on rough set (RSFS),and scatter search. In RSFS, conditional entropy is regarded as the heuristic to search for optimal solutions. Falangis and Glen (2010) proposed a variety of heuristic feature selection methods for CS problems with large numbers of observations. These heuristic procedures, which are based on the mixed integer programming model for maximizing classification accuracy, were applied to three CS datasets and proved to be efficient. The two most popularly used categories of heuristic search strategies are sequential search and random search.

- **Sequential search**

  Sequential search includes: forward selection, backward elimination and bidirectional search (Chan et al. 2010). These approaches consider local changes to the feature subsets during the search for the appropriate feature subset, where a local change is basically adding or removing a single feature from the subset. These approaches are known for their efficiency in generating fast results as the order of the search space is typically in the order of $O(d^2)$.

- **Random search**

  Generally, random search starts with selecting a random features' subset and may proceed in two different ways. Either it follows a classical sequential search and adds randomness into it, or it generates the next subset in a completely

random manner (Liu and Yu 2005).

1.3.3 Evaluation function

Feature selection methods search for the best subset that optimally describes the target variable. Once all candidate subsets are generated, each one is evaluated and compared with the other subsets according to an evaluation criterion. As established by Dash and Liu (2003), a subset is optimal always relative to the chosen evaluation criteria, which means that the chosen best subset using one evaluation criterion may not be the same using another one. Many criteria have been proposed in previous works as discussed by Kumar and Kumar (2011). Dash and Liu (2003) grouped the evaluation functions into five categories: distance, information, dependence, consistency, and classifier error rate. Liu and Yu (2005) on the other hand, has divided evaluation criteria into two classes based on their dependency on the classification algorithms that will finally be applied on the selected feature subset. Considering these groupings, we divide the evaluation functions as given below.

Independent criteria

Independent criteria, by definition, are independent of the used classification algorithm and are generally used in filter methods. They evaluate the relevance of a feature or feature subset by exploiting the intrinsic characteristics of the data without involving any classification algorithm. In the following we discuss the most well known independent criteria.

- **Distance measures**

  As discussed by Dash and Liu (2003) and by Liu and Yu (2005) in a binary context, distance measures, also known as separability, divergence, or discrimination measure, study the difference between the two-class conditional probabilities. In other words, a feature X^j is chosen over another feature $X^{j'}$ if it induces a greater difference between the two-class conditional probabilities than $X^{j'}$. In the case where the difference is zero then the two features are identical.

Relief is one of the most famous features selection method based on distance measures. This method uses the Euclidean distance to select a sample composed of a random instance x_i and their two nearest instances in $\mathbf{x}$ of the same class, i.e. $nearmiss(\mathbf{x})$, and opposite class i.e. $nearhit(\mathbf{x})$. Then, a routine is used to update the feature weight vector for every sample triplet and determines the average feature weight vector relevance. Then, features with average weights over a given threshold are selected. Algorithm (1.1) gives a more detailed picture of the process of Relief method, where $\mathbf{w} = (w_1, \ldots, w_d)$ is a weight vector associated to $\mathbf{X}$ and T is the number of iterations.

Algorithm 1.1 Relief algorithm

Require: $\mathbf{x}$ matrix of observations.
T number of iterations.
Ensure: selected features subset.
1: initiate the weight vector to zero: $\mathbf{w} = 0$.
2: **for** cpt=1 . . . T **do**
3: pick randomly an instance x_i from $\mathbf{x}$
4: **for** j =1 . . . d **do**
5: $w_j = (x_i^j - nearmiss(\mathbf{x})^j)^2 - (x_i^j - nearhit(\mathbf{x})^j)^2$
6: **end for**
7: **end for**
8: the chosen feature set is $\{X^j \mid w_j > threshold\}$

- **Information measures**

  The information theory approach has proved to be effective in solving many problems as discussed by Kumar and Kumar (2011). One of these problems is feature selection where information theory basics can be exploited as metrics or as optimization criteria. These measures are typically used with filter feature selection methods including mutual Information (MI). They provide a solid theoretical framework for measuring the relation between the classes and a feature or more than one feature (Bonev 2010).

  Formally, the MI of two continuous random variables X^j and $X^{j'}$ is defined as follows:

$$MI(X^j, X^{j\prime}) = \int\int p(x^j, x^{j\prime}) log \frac{p(x^j, x^{j\prime})}{p(x^j)p(x^{j\prime})} dx^j dx^{j\prime}, \tag{1.2}$$

where $p(x^j, x^{j\prime})$ is the joint probability density function and $p(x^j)$ and $p(x^{j\prime})$ are the marginal probability density functions. In the case of discrete random variables, the double integral becomes a summation, where $p(x^j, x^{j\prime})$ is the joint probability mass function, and $p(x^j)$ and $p(x^{j\prime})$ are the marginal probability mass functions. MI is an information metric used to measure the relevance of features taking into account the amount of information shared by two features (Kumar and Kumar 2011). Large values of MI indicate high correlation between the two features and zero indicates that two features are uncorrelated. Many authors proposed feature selection methods based on MI in different evaluation functions such as Kumar and Kumar (2011) and Al-Ani and Deriche (2001).

- **Dependency measures**

  As discussed by Dash and Liu (2003) and Yu and Liu (2003), dependency measures or correlation measures quantify the ability to predict the value of one variable based on the value of the other. If the correlation between two features is adopted as an evaluation function, the above definition becomes that a feature is relevant if it is strongly associated with the class. In other words, if the correlation of a feature X^j with the class variable is superior to the correlation of feature $X^{j\prime}$ with the class variable, then feature X^j is considered more predictive.

  The Pearson's correlation coefficient (PCC) for continuous features is a simple measure but effective in a wide variety of feature selection methods (Rodriguez et al. 2010). Formally PCC is defined by

$$PCC = \frac{cov(X^j, X^{j\prime})}{\sqrt{var(X^j)var(X^{j\prime})}}, \tag{1.3}$$

  where *cov* is the covariance and *var* is the variance. Another popular feature

selection method is Pearson's chi-squared test (χ^2). This test is usually used with nominal or categorical variables. The χ^2 test can also be used with numerical variables by converting them into nominal or categorical types. The first step in performing the χ^2 test for independence is to convert the raw data into a contingency table. Then, the independence between each variable and the target variable is measured using the contingency table. χ^2 is defined by :

$$\chi^2 = \sum_{i=1}^{c} \frac{(O_i - E_i)^2}{E_i}, \tag{1.4}$$

where O_i is the observed frequency; E_i is the expected theoretical frequency, asserted by the hypothesis of independency and c the number of cells in the contingency table.

- **Consistency measures**

  A consistency measure evaluates the distance of a feature subset from the consistent class label. Consistency is established when a data set with the selected features alone is consistent. That is, no two instances may have the same feature values if they have a different class label (Arauzo-Azofra et al. 2008).

  According to Arauzo-Azofra et al. (2008) having consistency in a dataset is usually accompanied while looking for a small feature set. Because, as the number of features increases the more the consistent hypothesis can be rejected. In any case, the search for small feature sets is the common goal of feature selection methods, so this is not a particularity of consistency methods. The most basic of these measures is the one that simply guesses if the training data set is consistent or not with the selected features. Its output is just a boolean value.

Dependent criteria

Dependent critera are generally used with wrapper feature selection methods when the performance of a specific scoring algorithm is used to determine which features are selected. When using dependant criteria we generally obtain superior results as the found features are well-matched to the predetermined mining algorithm. However, it

also tends to be more computationally expensive and may not be suitable for other scoring algorithms (Chrysostomou 2008).

In general, classification accuracy is widely used as the primary measure of dependent criteria. Features are selected by the classification algorithm and later used in predicting the class labels of unseen instances. Usually, accuracy is high but it is computationally costly to estimate accuracy for every feature subset (Yu and Liu 2004).

1.3.4 Stopping criterion

The final step in the process of feature selection is to choose a stopping criterion for the search of feature subsets. The stopping criteria depends on the level of dependency of the used evaluation function. As discussed by Chrysostomou (2008), in case independent criteria are used, a commonly used stopping criterion is the ordering of the features according to some relevance score. When dealing with dependent evaluation function one might stop adding or removing features when there is no more improvement in the accuracy of the current feature subset. Some frequently used stopping criteria are:

- The search is completed when all feature subsets are evaluated.

- A suitable high-quality subset is selected when the smallest feature subset with the highest discriminant power is found and as a result the search algorithm stops the searching process.

- A specific bound is achieved where a bound can be a particular number of features or number of iterations is reached.

- A subsequent addition, or deletion, of any feature does not produce a better subset.

1.4 Feature selection algorithms

1.4.1 Filter methods

A feature selection algorithm is considered a filter if it filters out all unwanted features (Molina et al. 2002; Blum and Langley 1997). According to Forman (2008) a filter technique is a pre-selection process which is independent of the applied classification algorithm. The process of filter methods is illustrated in Algorithm (1.2) (Yu and Liu 2004).

Algorithm 1.2 Generalized filter feature selection algorithm

Require: $\mathbf{X}$: all features.
F_0 : a subset of features from which to start the search $F_0 \subset \mathbf{X}$.
γ : a stopping criterion.
Ensure: F_{best}: selected features subset.
initialize: $F_{best} = F_0$.
$\gamma_{best} = eval(F_0, \mathbf{X}, M)$; evaluate F_0 by an independent criteria M.
while $\gamma == \gamma_{best}$ **do**
  $F = generate(\mathbf{X})$; generate a subset for evaluation.
  $\gamma = eval(F, \mathbf{X}, M)$; evaluate the current subset F by M.
  if γ is better than γ_{best} **then**
    $\gamma_{best} = \gamma$.
    $F_{best} = F$.
  end if
end while
return F_{best}.

Filter methods typically evaluate the importance of features by looking at the intrinsic properties of the data (Saeys et al. 2007). Basically, in filter approach, a relevance score is assigned to each feature in the dataset. Then, they are ordered according to their relevance score. In general, features with high scores are then selected and low scoring features are eliminated (Chrysostomou 2008). Once all features are ranked, the selected features are introduced as inputs to the classifier. Figure (1.3) illustrates the filter feature selection process.

Filters can be exceptionally effective since they easily scale down high dimensional data. They are computationally fast and simple since the selection criterion

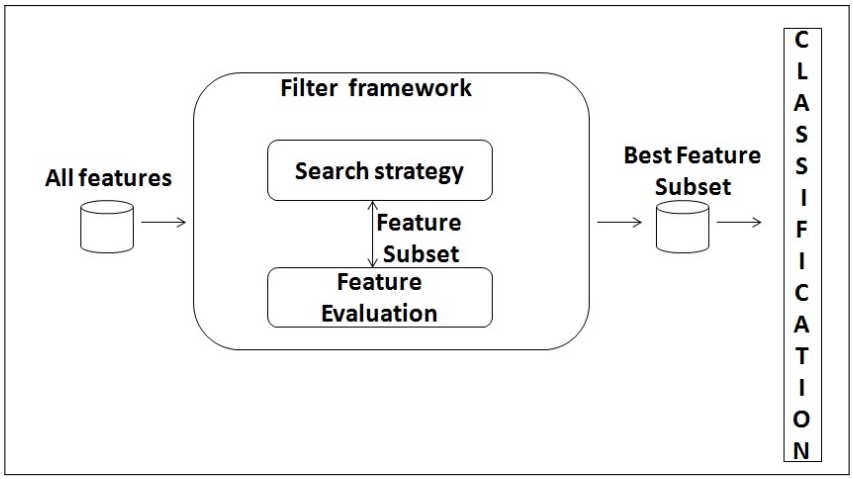


Figure 1.3: The process of filter feature selection.

is completely independent of the classifier (Guyon and Elisseeff 2003). Several ranking criteria for filter methods have been proposed in the literature. Examples of the commonly used filter ranking criteria are summarized in Table (1.1).

Table 1.1: Taxonomy of filter feature selection methods.

Model search	**Advantages**	**Disadvantages**	**Examples**
Univariate	Fast. Scalable. Independent of the classifier.	Ignores feature dependencies. Ignores interaction with the classifier.	PCC χ^2 Entropy
Multivariate	Models feature dependencies. Better computational complexity than wrapper methods. Independent of the classifier.	Slower than univariate techniques. Less scalable than univariate techniques. Ignores interaction with the classifier.	Correlation-based feature selection(CFS)

1.4.2 Wrapper methods

Wrapper methods use specific classifiers and use resulting classification performance to select features. While filter methods treat the problem of finding the best feature

subset independently of the learning step, wrapper methods use the model accuracy within the feature subset search. They use search methods to pick subsets of variables and evaluate their importance based on the estimated classification accuracy (Rodriguez et al. 2010). Details of the wrapper process are described in Algorithm (1.3) (Yu and Liu 2004).

Algorithm 1.3 Generalized wrapper feature selection algorithm

Require: $\mathbf{X}$: all features.
F_0 : a subset of features from which to start the search $F_0 \subset \mathbf{X}$.
γ : a stopping criterion.
Ensure: F_{best}: selected features subset.
1: initialize: $F_{best} = F_0$.
2: $\gamma_{best} = eval(F_0, \mathbf{X}, A)$; evaluate F_0 by a classification algorithm A.
3: **while** $\gamma = \gamma_{best}$ **do**
4: $F = generate(\mathbf{X})$; generate a subset for evaluation.
5: $\gamma = eval(F, \mathbf{X}, A)$; evaluate the current subset F by A.
6: **if** γ is better than γ_{best} **then**
7: $\gamma_{best} = \gamma$.
8: $F_{best} = F$.
9: **end if**
10: **end while**
11: return F_{best}.

According to Kohavi and John (1997), a wrapper model incorporates the classification algorithm into the feature selection process and considers it as a perfect "black box". In other words it is not necessary to know the classification algorithm or how it works, only its ability to test the solution on the validation set.

Wrappers use a search procedure in the space of possible features, and then generate and evaluate various subsets in order to find the best one. The evaluation of a specific subset of features is obtained by training and testing a specific classification model repetitively, rendering this approach tailored to a specific classification algorithm. To search the space of all feature subsets, a search algorithm is then 'wrapped' around the classification model. Figure (1.4) illustrates the process of wrapper feature selection.

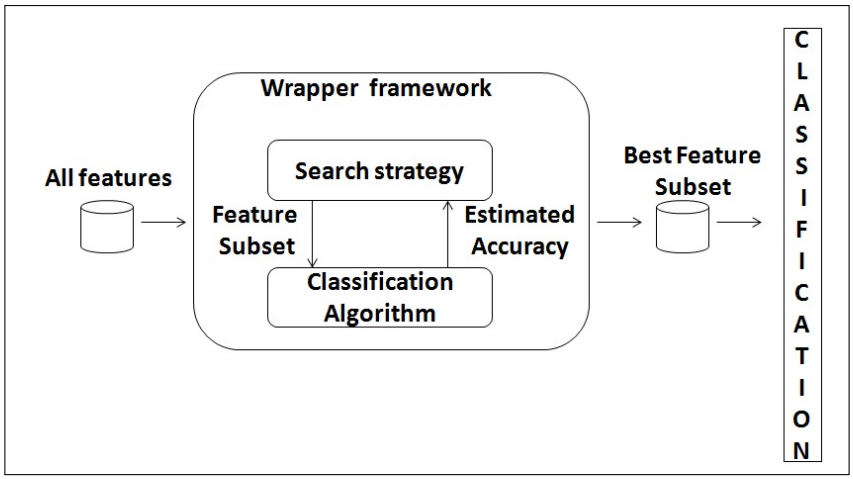


Figure 1.4: The process of wrapper feature selection.

1.4.3 Embedded methods

In embedded feature selection methods the search for an optimal subset of features is built into the classifier construction; i.e. feature selection occurs naturally as a part of the learner. Typically, these methods use all features as input to generate a model. Then, they evaluate the model to infer the relevance of the features. As a result, they directly link features relevance to the learner used to model the relationship (Tuv et al. 2009).

Just like wrappers, embedded methods are specific to a given learning algorithm. In fact, the classifier has its own feature selection algorithm and both interact together. So, implicitly, features' dependencies are taken into account. Also embedded methods are far less computationally intensive than wrapper methods.

As discussed earlier, the similarly to wrapper methods is linked to the classification stage. This same link is much stronger when the feature selection of the embedded methods is included into the classifier construction. Embedded methods offer the same advantages as wrapper methods concerning the interaction between the feature selection and the classification. However, since the embedded-based approaches are algorithm-specific they are not adequate for our requirement.

1.4.4 Hybrid methods

The hybrid model attempts to take advantage of other feature selection approaches by using their different evaluation criteria in different search stages. In case the chosen feature selection technique proves to be too slow to allow complex search schemes for a large number of candidate features, it may be more practical to introduce another fast but less accurate feature selection method to pre-filter some of unwanted features. So, only more promising features are eventually presented to the primary slow feature selection technique.

Many hybrid feature selection methods were proposed in the past few years to construct accurate CS model. An interesting hybrid filter-wrapper approach is introduced by Huang et al. (2007) where a genetic algorithm based approach is used to optimize the parameters of SVM classifier and feature subset simultaneously, without reducing the SVM classification accuracy. Cho et al. (2010) proposes a hybrid method for effective bankruptcy prediction, based on the combination of variable selection using decision trees and case-based reasoning using the Mahalanobis distance with variable weight.

In general, hybrid algorithms focus on combining filter and wrapper algorithms to achieve the best possible performance with similar accuracy of wrapper and time complexity of filter algorithms.

1.4.5 Comparison of feature selection algorithms

Numerous feature selection techniques are available. In order to better understand the inner instrument of each technique and the commonalities and differences among them, we present a categorizing framework in Table (1.2) based on the previous discussions.

Comparing feature selection methods is not an easy task, since it depends on numerous factors. Feature selection methods could be compared according to different purposes, for general purpose of irrelevancy removal, filters are good choices as they are unbiased and fast. On the other hand, to improve the classification performance,

wrappers should be preferred over filters since they are more appropriate to the classification tasks. Sometimes, hybrid feature selection methods are needed to serve more complicated purposes.

In terms of the amount of time, a feature selection method that is considered to be theoretically complex may take longer to select relevant features than a feature selection method which is regarded as theoretically simple. The time concern is also about whether the feature selection process is time critical or not.

When time is not an important issue, based-complete search methods are recommended to achieve optimality, otherwise heuristic-based search methods should be selected for fast results. Time constraints can also affect the choice of feature selection models as different models have different computational complexities. The filter model is preferred in applications where applying a particular classifier is too costly.

Table 1.2: Summary and comparison of feature selection methods.

	Filter	**Wrapper**	**Hybrid**
Evaluation Criterion	distance, information, dependency and consistency	predictive accuracy	independent criteria, dependent criteria
Search	feature weighting, subset search	exhaustive, heuristic	mixture
Characteristics	unbiased and fast, robust against overfitting, reasonable computation cost, reasonable statistical scalability	achieve higher optimality, interact with the classifier, consider dependencies	take advantage of other feature selection approaches

1.5 Datasets description and pre-processing

1.5.1 Datasets description

The adopted herein datasets used for evaluation are four real-world datasets: two datasets from the UCI repository of machine learning databases: Australian and

German credit datasets (http://archive.ics.uci.edu/ml/datasets.html), a dataset from a Tunisian bank and the HMEQ dataset. Table (1.3) displays the characteristics of these datasets.

Table 1.3: Summary of datasets used for evaluating the feature selection methods.

Names	Australian	German	HMEQ	Tunisian
Total instances	690	1000	5960	2970
Nominal features	6	13	2	11
Numeric features	8	7	10	11
Total features	14	20	12	22
Number of classes	2	2	2	2

Australian credit dataset:

Australian dataset presents an interesting mixture of attributes: continuous, nominal with small numbers of values, and nominal with larger numbers of values, with few missing values. Appendix (A) contains the complete list of variables used in this data set. It is composed of 690 instances where 307 are creditworthy while 383 are not. All attribute names and values have been changed to meaningless symbols for confidentiality. This dataset was used in the European StatLog project, which involves comparing the performances of machine learning, statistical and neural network algorithms on data sets from real-world industrial areas including medicine, finance, image analysis and engineering design.

German credit dataset:

The German credit dataset is often used by credit specialists for classification purposes. This dataset covers a sample of 1000 credit consumers where 700 are creditworthy and 300 are not. For each applicant 21 numeric input variables are available, .i.e. 7 metric, 13 categorical and a target attribute, with information pertaining to past and current customers who borrowed from a German bank (http://www.stat.uni-muenchen.de/service/datenarchiv/kredit/kredite.html).

Among the 20 input variables assumed to affect the target variable we mention: duration of credits in months, behavior repayment of other loans, value of savings

or stocks, stability in the employment and further running credits. Appendix (A) contains the complete list of variables used in this data set.

HMEQ credit dataset:

The HMEQ dataset is composed of 5960 instances describing recent home equity loans where 4771 instances are creditworthy and 1189 are not. The target is a binary variable that indicates if an applicant eventually defaulted. For each applicant, 12 input variables were recorded where 10 are continuous features, 1 is binary and 1 is nominal, more details are provided in Appendix (A). .

Tunisian credit dataset:

Tunisian dataset covers a sample of 2970 instances of credit consumers where 2523 instances are creditworthy while 446 are not. Each credit applicant is described by a binary target variable and a set of 22 input variables where 11 features are numerical and 11 are categorical (see Appendix (A)).

1.5.2 Data pre-processing

In this section, we describe the adopted data pre-processing steps. Each dataset is cleaned from missing values, then it is discretized and split into training and testing samples as shown in Figure (1.5).

Missing value replacement

Most financial datasets contain missing values that should be properly handled. Many methods dealing with missing values are available. The simplest one is to remove all instances with missing values. This method is suitable when missing data are not important. Another simple way is to substitute missing values with the corresponding mean or median values over all instances. In this context, we estimate missing values with the average or mode of features depending on their nature, meaning either numerical or categorical.

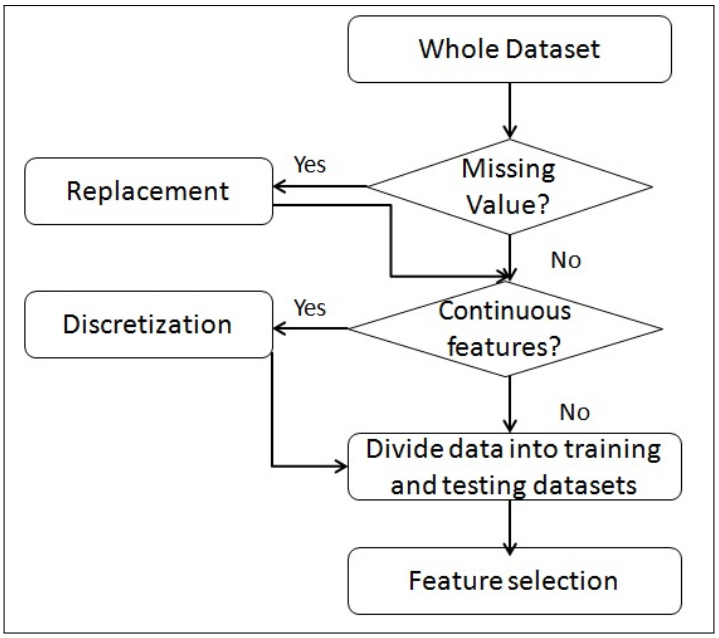


Figure 1.5: Data pre-processing flowchart

Features discretization

For simplicity, each variable is discretized, knowing that discretization of continuous features depends of the context. In this study, we are in the supervised learning context. The discretization step should be performed prior to the learning process. Several tools can be used for that, and we selected Weka 3.7.0 machine learning package (Bouckaert et al. 2009) for its simplicity.

Splitting datasets

Datasets for the scoring task are usually extremely large. In order to reduce classification tools complexity and to increase scoring models accuracy sampling becomes necessary as stated by Fernandez (2010). In order to obtain a calibrated model, the credit database should be split. Sampled subsets are expected to be balanced and cover the complete database. Subsequently, we split the datasets into a training sample and a test sample, where the first deals with the new feature selection approach and diverse classification models and the second one checks the reliability of the constructed models in the learning step.

1.6 Performance metrics for feature selection

The performance of our proposed methods is evaluated using the standard information retrieval performance measures: precision, recall and F-measure metrics. In a classification context, the precision is calculated as the ratio of the number of credit applicants correctly identified by the model as positives $Y = 1$, i.e. true positive (TP), to the total number of credit applicants. The total number of credit applicants is the number of applicants correctly identified as positives plus the number of incorrectly classified applicants, i.e. false positive (FP).

The recall, also known as TP rate or sensitivity, measures how often a classification model correctly finds the right class to a credit applicant. It is defined as the proportion of TP against the total number of applicants that actually belong to the positive class. The total number of potential correct applicants is the number of TP plus the count of false negatives (FN) which are the applicants that were not labeled as belonging to the positive class but should have been.

The precision rate of 1 for a class C means that every applicant affected to this class does indeed belong to it, but this rate does not inform about the number of applicants from this class that were not correctly classified. A recall rate of 1 means that every applicant from class C is labeled as belonging to class C but does not inform about the number of applicants that were incorrectly labeled as belonging to class C. In general, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. The F-measure combines recall and precision into a global measure.

In general, the terms TP, TN, FP, and FN evaluate the results of the classifier. The terms positive (P) and negative (N) refer to the classifier's prediction, and the terms true (T) and false (F) refer to whether that prediction corresponds to the external observation.

The four outcomes can be formulated in confusion matrix, as follows:

Table 1.4: Confusion matrix

		Predicted		
		Creditworthy $(Y = 1)$	**Not creditworthy** $(Y = 0)$	**Total**
Observed	$(Y = 1)$	TP	FP	$TP + FP$
	$(Y = 0)$	FN	TN	$FN + TN$
	Total	$TP + FN$	$FP + TN$	n

Precision, recall and F-measure are then given by :

$$Precision = \frac{|TP|}{|TP| + |FP|}. \tag{1.5}$$

$$Recall = \frac{|TP|}{|TP| + |FN|}. \tag{1.6}$$

$$F\text{-}measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}, \tag{1.7}$$

The cited performance measures are obtained when the cut-off is 0.5. However, changing this threshold might modify previous results and allows to catch a greater number of good or bad applicants. Graphical tools can also be used as an evaluation criterion instead of a scalar criterion, such as the area under the receiver operating characteristic (ROC) curve used to evaluate the effect of selected features on classification models. ROC curve shows how errors change when the threshold varies. This curve situates positive instances against negative ones to allow finding the middle ground between specificity and sensitivity. An area of 1 represents a perfect test; an area equal or below 0.5 represents a worthless test. So the combination of features that gives the highest area under the ROC curve will be considered as the most suitable for the classification task.

1.7 Conclusion

This chapter gives an overview of CS and feature selection methods. A brief state of the art of the commonly used feature selection methods, namely filter, wrapper and

hybrid feature selection methods, is given. Filter methods do not use classifiers but instead they use independent criteria and the characteristics of the dataset to select relevant features. Wrapper methods, on the other hand, are classifier dependant. Moreover, hybrid methods present a mixture between filters and wrappers. In the following chapters three feature selection methods will be proposed. Details about the proposed methods and their related results are presented in Chapters 2, 3 and 4.

Chapter 2

A Filter Rank Aggregation Approach Based on Optimization, Genetic Algorithm and Similarity for Credit Scoring

Contents

2.1 Introduction

Filters are commonly used feature selection methods. This chapter discusses the major issues of this approach and presents a new approach based on rank aggregation, GA and similarity. As such, in Section (2.2) we give a brief reminder of the filter framework and two major issues when dealing with filtering methods: the selection trouble and the issue of disjoint ranking for similar features. Then, we present our new approach in Section (2.3) and the experimental study in Section (2.4).

2.2 Filter framework

According to Yu and Liu (2003) filter methods can be grouped into two categories: feature weighting methods and subset search methods. This categorization is based on whether they evaluate the relevance of features separately or through feature subsets. In what follows, we present the advantages and shortcomings of some well known feature selection methods in each category.

2.2.1 Feature weighting methods

In feature weighting methods, weights are assigned to each feature independently and features are ranked based on their relevance to the target variable. Relief is a famous algorithm that studies features relevance (Kira and Rendell 1992). Algorithm (1.1) in Chapter (1) presents the basic concepts of this method. Notice that the fundamental idea of Relief is to estimate the relevance of features according to how well their values separate the instances of the same and different classes that are near each other (Yu and Liu 2003).

For a dataset with n instances and d features the complexity of relief is in order of $O(nd)$, which makes it very practical to data sets with large number of instances and features, such as CS datasets. Although simple, Relief doesn't remove redundant features. If feature weights are superior to a particular threshold, these features will be selected even though many of them are highly correlated to each other (Kira and

Rendell 1992).

In general, feature weighting methods have similar shortcomings as Relief. They are good in capturing the relevance of features to the target variable but fail to capture redundancy among features.

2.2.2 Subset search methods

Subset search methods use a particular evaluation measure which captures the relevance of each subset. In this way not only relevance is considered but also redundant features are identified within the selected subset. In this context Hall (2000) used a correlation measure to evaluate the relevance of the feature subsets. He based his work on the hypothesis that a good feature subset contains highly correlated features to the target variables, yet uncorrelated to each other. His proposed approach, named CFS, also uses heuristic search to find a candidate subsets to be evaluated.

According to Arauzo-Azofra et al. (2008) correlation measures efficiently decrease irrelevance and redundancy. Yu and Liu (2003) recommended two main approaches to measure correlation, one is based on classical linear correlation between random variables and the other is based on information theory.

Many correlation coefficients can be used under to first approach but the most common are PCC and χ^2 (see Section (1.3)). According to Yu and Liu (2003) PCC is not able to capture correlations that are not linear. Another limitation is that the calculation requires all features to have numerical values. On the other hand,χ^2 is used to investigate whether two distributions of categorical variables differ. To overcome these shortcomings, the correlation measure based on the information theory could be used.

The second approach based on information theory measures how much knowledge two variables carry about each other. MI is a well known information theory measure that captures nonlinear dependencies between variables (for more details see Section (1.3)).

In general, Subset search methods need to evaluate all possible subsets. Consequently, a search is performed to find the candidate subsets. Therefore, these methods suffer from time complexity issues which make them not practical to deal with high dimensional data.

Feature ranking makes use of a scoring function computed from the values (x_i^j, y_i) using one of the criteria discussed above such as weighting, consistency and correlation. It is assumed that a high score is indicative of a valuable variable and that variables are sorted in decreasing order of the scoring function. Even when feature ranking is not optimal, it could be preferable than any other feature subset selection method because of its computational and statistical scalability. It is computationally efficient since it requires only the computation of d scores and sorting them. It is statistically robust against overfitting because it introduces bias, however it may have considerably less variance (Hastie et al. 2001).

2.2.3 Issue I: Selection trouble and rank aggregation

Given the variety of filter based methods, it is difficult to identify which of the filter criteria would provide the best output for the experiments. The question is then how to choose the best criterion for a specific feature selection task? Wu et al. (2009) call this problem a selection trouble. There exists no universal solution for this problem unless to evaluate all existing methods and then establish a general conclusion, which is an impossible task. The best approach is to independently apply a mixture of the available methods and evaluate the results.

Combining preference lists from those individual rankers into a single better ranking is known as rank aggregation. Rank aggregation methods have emerged as an important tool for combining information in CS. Ensemble feature selection methods, i.e. rank aggregation, use an idea similar to ensemble learning for classification (Dietterich 2000). In a first step, a number of different feature selectors, i.e. rankers, are used and then the output of these separate selectors is aggregated and returned as the final ensemble result.

Ensemble methods have been widely applied to bring together a set of classifiers for building robust predictive models. It has been shown that these ensemble classifiers are competitive with other individual classifiers and in some cases are superior. Recently, there have been studies applying the ensemble concept to the process of feature selection (Dittman et al. 2013). Rank aggregation could be used to improve the robustness of the individual feature selection methods. Different rankers may yield different ranking lists that can be considered as local optima in the space of feature subsets and ensemble feature selection might give a better approximation to the optimal ranking of features. Also, the representational power of a particular feature selector might constrain its search space such that optimal subsets cannot be reached. Ensemble feature selection could help in alleviating this problem by aggregating the outputs of several feature selectors (Saeys et al. 2008).

As discussed earlier, rank aggregation has many merits. However, with ensemble feature selection the question is how to aggregate results of individual rankers. A number of different rank aggregation methods have been proposed in the literature. Some of them are easy to set up like the mean, median, highest rank or lowest rank aggregation and some are more difficult (Dittman et al. 2013).

All rank aggregation methods assume that the ranked lists being combined assign a value to each feature, from 1 to d, where the rank 1 is assigned to the most relevant feature, the second best feature is 2, and so on until the least relevant feature is assigned d. Simple rank aggregation method use straightforward way to find the final aggregated list, in all cases, once each feature has been given a single value based on the mean, median, highest, or lowest value, all features are ranked based on these new values. For example, mean aggregation simply finds the mean value of the feature's rank across all the lists and uses this as that feature's value. Likewise, median finds the median rank value across all the lists being combined, using the mean of the middle two values if there is an even number of lists. Highest rank and lowest rank use related strategies: either the highest (best, smallest) or the lowest (worst, largest) rank value across all the lists is assigned as the value for the feature

in question. Figure (2.1) shows the general rank aggregation process to obtain a consensus rank list from m individual filters.

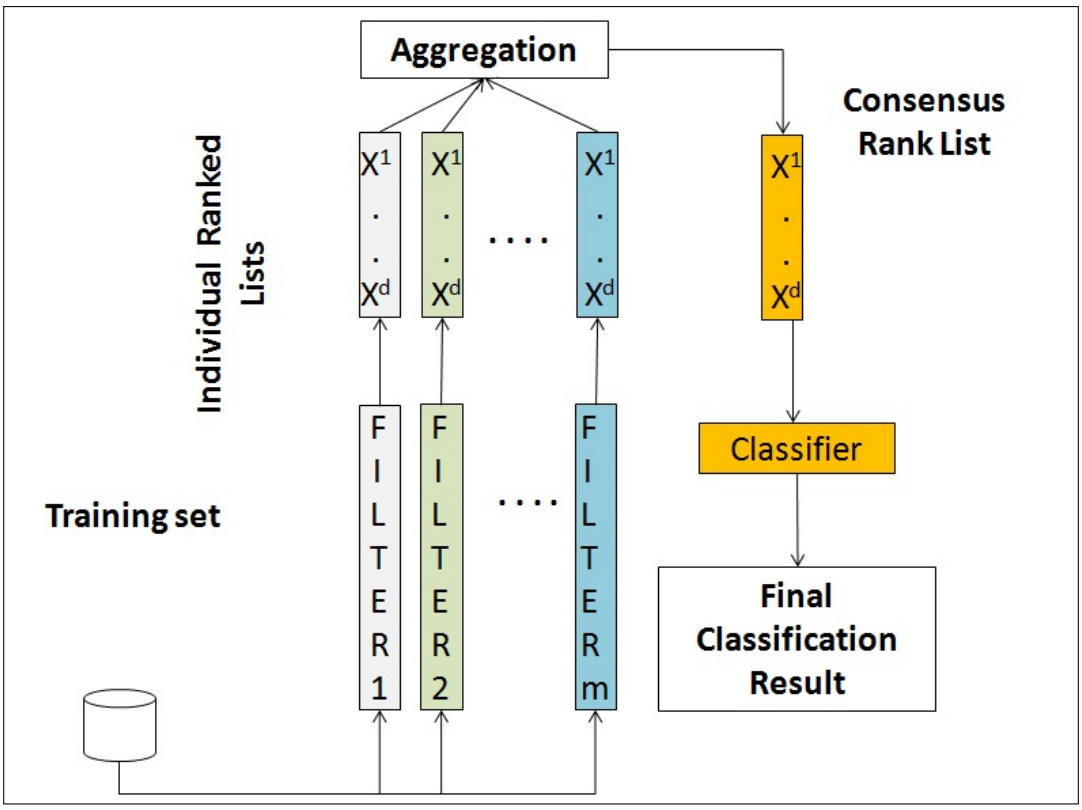


Figure 2.1: General scheme of filter rank aggregation.

Simple ranking methods are easy to set up. However, in many cases it is possible for two features to end up tied, even if this was not the case in any of the lists being combined and even when these features do not have any tie of similarity (Dittman et al. 2013). Recent works in the area of rank aggregation methods have developed unique and innovative approaches. These new methods can focus on different aspects of the ranking process including comparing results to randomly generated results. Kolde et al. (2012) proposed an approach that detects features that are ranked consistently better than expected under null hypothesis of uncorrelated inputs and assigns a significance score for each feature. The underlying probabilistic model makes the algorithm parameter free and robust to outliers, noise and errors. Other research focused on giving more weight to top ranking features or combining well known aggregation methods. In this work we use rank aggregation from another

perspective. In fact we aim to find the best list which would be the closest as possible to all individual ordered lists all together and this can be seen as an optimization problem. More details will be given in Section (2.3).

2.2.4 Issue II: Incomplete ranking and disjoint ranking for similar features

The rankings provided by different filters may be in many cases incomplete or even disjoint. In fact incomplete rankings may come in two forms.

- In the first form, different filters or some of them may each provide rankings for only the k best features and ignores the remaining features provided in the beginning (Sculley 2007). Assume we have 7 features $\{X^1, X^2, X^3, X^4, X^5, X^6, X^7\}$, where $\{X^1, X^3\}$ are not the most relevant features. In this case one of the filter may provide a ranking just over the set $\{X^2, X^4, X^5, X^6, X^7\}$ and ignores X^1 and X^3.

- In the second form, used filters may provide complete rankings over a limited subset of available features due to incomplete knowledge (Sculley 2007). Having the same example where we have 7 features $\{X^1, X^2, X^3, X^4, X^5, X^6, X^7\}$ and only information about features $\{X^3, X^5, X^6\}$ is available. In this case one of the filter may provide a ranking just over the set $\{X^3, X^5, X^6\}$ and ignore the set $\{X^1, X^2, X^4, X^7\}$.

Incomplete rankings are common in many financial applications but still it is not the only problem with rank aggregation. In fact the majority of rankings involve a set of similar features, but despite the similarity between these features they are not ranked similarly which additionally to the problem of incomplete rankings may lead to noisy rankings.

Let us give an illustrative example. Assume we have 7 features $\{X^1, X^2, X^3, X^4, X^5, X^6, X^7\}$, where X^2 and X^5 are highly similar but not identical. We consider the two following rank lists from two different filters:
list one is

$$\{X^3, X^2, X^7, X^5\}$$

and list two is

$$\{X^2, X^7, X^3, X^4, X^1\}.$$

If we have no preference to either one then standard methods of rank aggregation may produce the rankings in the following way:

$$\text{Aggregation 1: } \{X^2, X^3, X^7, X^5, X^4, X^1\}.$$

$$\text{Aggregation 2: } \{X^3, X^2, X^7, X^5, X^4, X^1\}.$$

And if we want to take advantage of similarity in rank aggregation, we need a new aggregation method. The latter should use the additional information provided by a defined similarity measure. Therefore, a more acceptable ranking that agrees with our point of view is:

$$\{X^3, X^2, X^5, X^7, X^4, X^1\}.$$

To avoid disjoint ranking for similar features, we present in the Section (2.3.3) a simple approach that extends any standard aggregation method in order to take similarity into account.

2.3 New approach for filter feature selection

In this section we propose a novel approach for filter feature selection. We consider building a two-stage filter feature selection model. In the first step, an optimization function and GA are used to solve the selection trouble and the rank aggregation problem and to sort the features according to their relevance. In the second step, a standard algorithm is proposed in order to solve the problem of disjoint rankings for similar features and to eliminate redundant ones.

2.3.1 Optimization problem

The aim of rank aggregation when dealing with feature selection is to find the best list which would be the closest as possible to all individual ordered lists all together.

This can be seen as an optimization problem when we look at $argmin(D, \sigma)$, where $argmin$ gives a list σ at which the distance D with a randomly selected ordered list is minimized. In this optimization framework the objective function is given by :

$$f(\sigma) = \sum_{i=1}^{m} W_i \times D(\sigma, L_i), \tag{2.1}$$

where W_i denotes the weight associated with the lists L_i, D is a distance function measuring the distance between a pair of ordered lists, m is the number of lists and L_i is the i^{th} ordered list of cardinality k. The best solution is then to look for σ^*, which would minimize the total distance between σ^* and L_i, given by

$$\sigma^* = argmin \sum_{i=1}^{m} W_i \times D(\sigma, L_i). \tag{2.2}$$

Measuring the distance between two ranking lists is classical and several well-studied metrics are known (Carterette 2009; Kumar and Vassilvitskii 2010), including the Kendall's tau distance and the Spearman footrule distance. Before defining this two distance measures and their corresponding weighted distances some necessary notations are needed.

For each feature $X^j \in L_i$, $r(X^j)$, $j = 1, \ldots, d$ shows the ranking of this feature, where $r(X^j) = 1$ is associated with the feature on top of L_i, that is the most important one and $r(X^j) = d$ is associated with those feature which is at the bottom, or the least important one with regard to the target concept. All other ranks correspond to the features that would be in-between. Note that rankings are always positive, and higher rank shows lower preference in the list.

Spearman footrule distance

Spearman footrule distance between two given rankings lists L and σ is defined as the sum overall the absolute differences between the ranks of all unique elements from both ordered lists combined. Formally, the Spearman footrule distance between L and σ is given by

$$Spearman(L, \sigma) = \sum_{X \in (L \cup \sigma)} |r_L(X) - r_\sigma(X)|. \tag{2.3}$$

Spearman footrule distance is a very simple way to compare two ordered lists. The smaller the value of this distance the more similar the lists are. When the two lists to be compared have no elements in common, the metric is $k(k+1)$.

Kendall's tau distance

Kendall's tau distance between two ordered rank lists L and σ is given by the number of pairwise adjacent transpositions needed to transform one list into another (Dinu and Manea 2006). This distance can be seen as the number of pairwise disagreements between the two rankings. Hence, the formal definition of the Kendall's tau distance is:

$$Kendall(L, \sigma) = \sum_{X^j, X^{j\prime} \in (L \cup \sigma)} K, \tag{2.4}$$

where

$$K = \begin{cases} 0 \quad \textit{if } r_L(X^j) < r_L(X^{j\prime}),\ r_\sigma(X^j) < r_\sigma(X^{j\prime}) \\ \text{or } r_L(X^j) > r_L(X^{j\prime}),\ r_\sigma(X^j) > r_\sigma(X^{j\prime}) \\ \\ 1 \quad \textit{if } r_L(X^j) > r_L(X^{j\prime}),\ r_\sigma(X^j) < r_\sigma(X^{j\prime}) \\ \text{or } r_L(X^j) < r_L(X^{j\prime}),\ r_\sigma(X^j) > r_\sigma(X^{j\prime}) \\ \\ p \quad \textit{if } r_L(X^j) = r_L(X^{j\prime}) = k+1, \\ \text{or } r_\sigma(X^j) = r_\sigma(X^{j\prime}) = k+1 \end{cases} \tag{2.5}$$

That is, if we have no knowledge of the relative position of X^j and $X^{j\prime}$ in one of the lists we have several choices: impose no penalty (0), full penalty (1), or a partial penalty p such that $0 < p < 1$.

Weighted distance

In case, the only information available about the individual list is the rank order, the Spearman footrule distance and the Kendall's tau distance are adequate measures. However, the presence of any additional information about the individual list may improve the final aggregation. Typically with filter methods, weights are assigned to each feature independently and then the features are ranked based on their relevance to the target variable. It would be beneficial to integrate these weights w into our aggregation scheme. Hence, the weight associated with each feature consists of taking the average score across all the ranked feature lists. We find the average for each feature by adding all the normalized scores associated to each lists, and dividing the sum by the number of lists. According to Pihur et al. (2009) the weighted Spearman's footrule distance between the two lists L and σ is given by

$$\mathbf{w}.Spearman(L,\sigma) = \sum_{X\in(L\cup\sigma)} |w(r_L(X)) - w(r_\sigma(X))| \times |r_L(X) - r_\sigma(X)|, \qquad (2.6)$$

were $w(r_L(X))$ and $w(r_\sigma(X))$ denote the weights associated with the feature X with rank r in the lists L and σ. Analogously to the weighted Spearman's footrule distance, the weighted Kendall's tau distance (Pihur et al. 2009) is given by:

$$\mathbf{w}.Kendall(L,\sigma) = \sum_{X^j,X^{j\prime}\in(L\cup\sigma)} |w(r_L(X^j)) - w(r_\sigma(X^{j\prime}))|K. \qquad (2.7)$$

2.3.2 Solution to optimization problem using genetic algorithm

The introduced optimization problem in Section (2.3.1) is a typical integer programming problem. As far as we know, there is no efficient solution to such kind of

problem. One possible approach would be to perform complete search. However, it is too time demanding to be useful in real applications, and more practical solutions are needed.

The introduced method uses GA for rank aggregation. GAs were developed by Holland (1992) to imitate the mechanism of genetic models of natural evolution and selection. GAs are powerful tools for solving complex combinatorial problems, where a combinatorial problem involves choosing the best subset of components from a pool of possible components such that the mixture has some desired quality (Clegg et al. 2009). GAs are computational models of evolution. They work on the basis of a set of candidate solutions. Each candidate solution is called a "chromosome", and the whole set of solutions is called a "population". The algorithm allows movement from one population of chromosomes to a new population in an iterative fashion. Each iteration is called a "generation". In our case, GA proceeds in the following way: initialization, selection, cross-over and mutation.

Initialization

Once a set of aggregation rank lists are generated by several filtering methods, it is necessary to create an initial population of features to be used as starting point for the genetic algorithm, where each feature in the population represents a possible solution. This starting population is then obtained by randomly selecting a set of ordered rank lists.

Despite the success of GA on a wide collection of problems, the choice of the population size is still an issue. Gotshall and Rylander (2000) proved that the larger the population size is the better chance of it containing the optimal solution. However, increasing population size increases the number of generations. In order to have great results, the population size should depend on the length of the ordered lists and on the number of unique elements in these lists. From empirical studies, over a wide range of problems, a population size between 30 and 100 is usually recommended (Pihur et al. 2009).

Selection

Once the initial population is fixed, we need to select new members for the next generation. In fact, each element in the current population is evaluated on the basis of its overall fitness given by Equation (2.1). Depending on which distance is used, new members, i.e. rank lists, are produced by selecting high performing elements.

Cross-over

The selected members are then crossed-over with the cross-over probability. Crossover randomly selects a point in two selected lists and exchanges the remaining segments of these lists to create new ones. Therefore, crossover combines the features of two lists to create two similar ranked lists.

Mutation

In case only the crossover operator is used to produce the new generation, one possible problem that may arise is when all the ranked lists in the initial population have the same value at a particular rank. Then, all future lists will have the same value at this particular rank. To overcome this unwanted situation a mutation operator is used. Mutation operates by randomly changing one or more elements of any list. It acts as a population perturbation operator. Typically mutation does not occur frequently so mutation is of the order of 0.001 (Pihur et al. 2009).

Figure (2.2) presents a flowchart summarizing the fundamental steps of the proposed rank aggregation method using GAs.

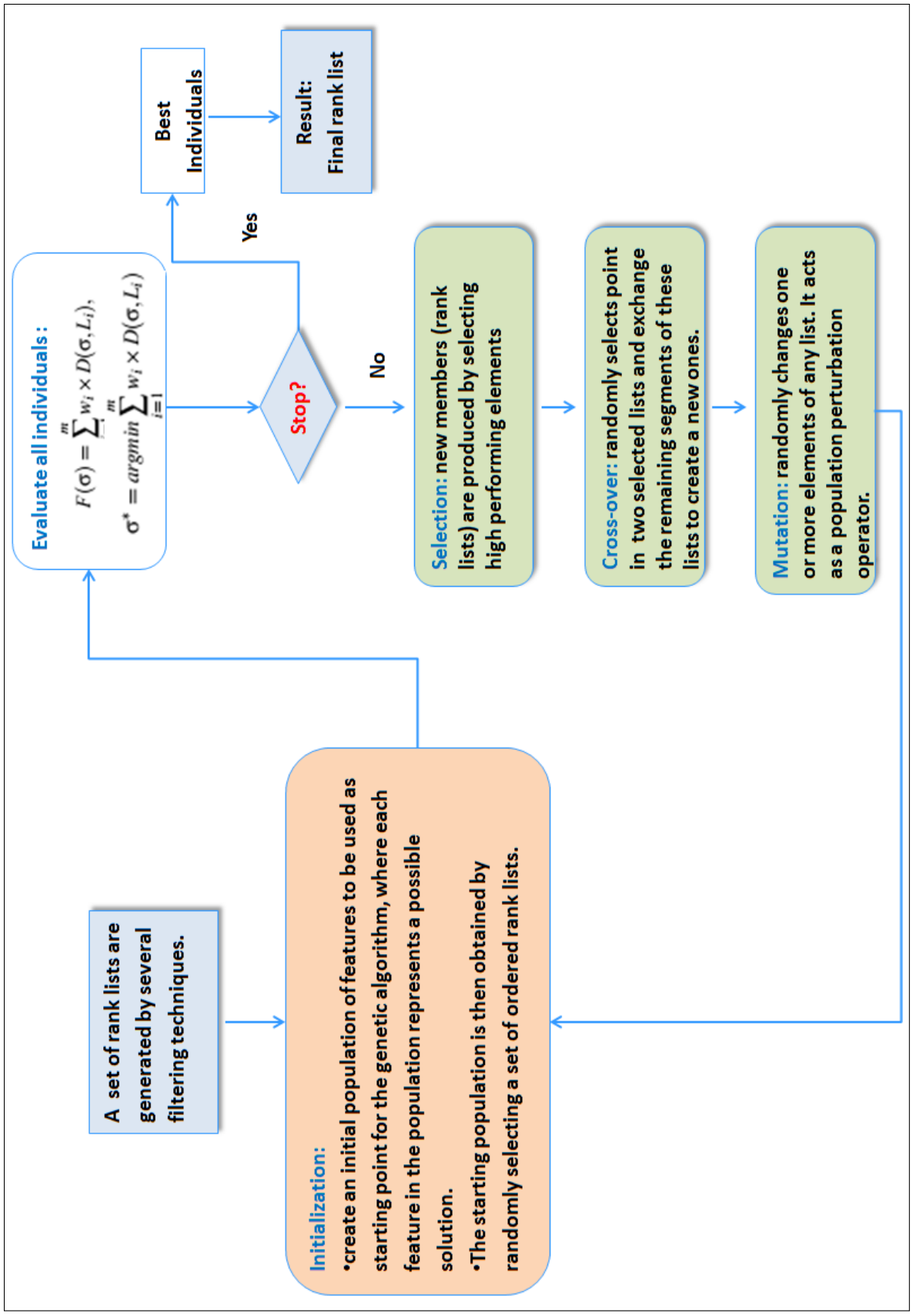


Figure 2.2: A summary flowchart of the proposed genetic algorithm rank aggregation

2.3.3 A rank aggregation based on similarity

In this section we solve the problem of disjoint ranking for similar features and eliminating redundant ones. First, we perform a simple algorithm that incorporates similarity knowledge in the final ranking in order to handle disjoint ranking of similar features. Then, redundant features are eliminated by comparing the relevance of each pair of redundant features to the target class. Figure (2.3) gives a summary of performing rank aggregation based on similarity.

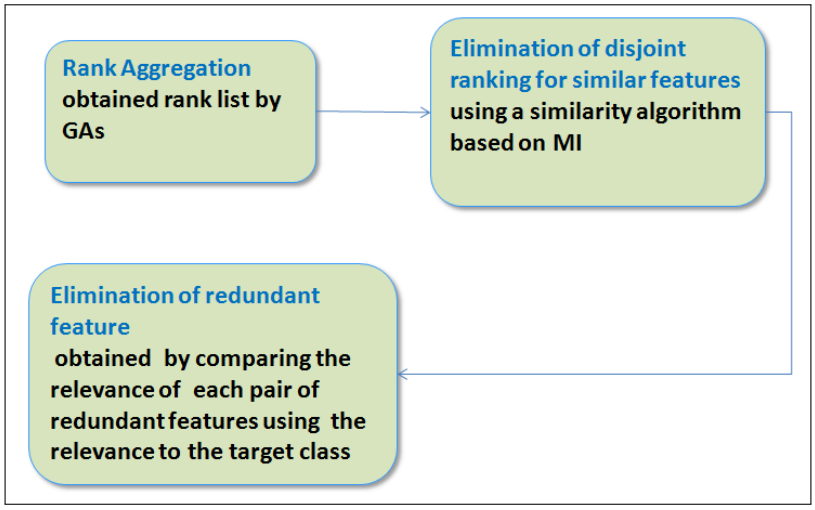


Figure 2.3: Flowchart summarizing the rank aggregation approach based on similarity

Solution to disjoint ranking for similar features

First we perform a feature selection using three different feature selection methods namely: Relief, χ^2 and MI and three different rankings are obtained. Second an aggregation is performed on the obtained three ranking lists using the proposed genetic rank aggregation algorithm in Section (2.3.2) yielding a combined list $Initial_R$.

In each iteration we study the similarity between the first feature in $Initial_R$, i.e. the feature with $r(X^j) = 1$ that we denote Var, and the remaining features in this list using a function denoted SIM. Before using the SIM function, the possible similarities between features in $\mathbf{X}$ are summarized using the full MI matrix. Where each element in this matrix represents the pairwise similarity between two features X^j and $X^{j'}$.

This matrix is in general a $(d \times d)$ symmetric positive semi definite matrix and takes on values in $[0, \ldots, 1]$, with diagonal values equal to 0. A large value indicates a close relationship between variables X^j and $X^{j'}$.

SIM function compares the similarity between the features using this matrix. If Var has 80% of similarity with any of the features in the list $Initial_R$, the function SIM return 'TRUE' elsewhere it returns 'FALSE'.

In accordance with the function SIM result, two possible scenarios arise depending on whether Var has a strong link of similarity with the remaining features or not.

First scenario: if the function SIM returns 'FALSE' then Var doesn't have any strong connection with any other features in the list $Initial_R$. In this case there is no disagreement among rankings and Var is removed from the list $Initial_R$ and added to $Final_R$ which is the final aggregation list that will be used for classification.

Lets take the previous illustrative example of Section (2.2.4) where we aggregate two ranking lists $\{X^3, X^2, X^7, X^5\}$ and $\{X^2, X^7, X^3, X^4, X^1\}$. If the obtained aggregated list is $\{X^3, X^2, X^7, X^5, X^4, X^1\}$ then, the function SIM studies the similarity between feature X^3 and $\{X^2, X^7, X^5, X^4, X^1\}$ and returns false. Then, X^3 will be added to $Final_R$ and $\{X^2, X^7, X^5, X^4, X^1\}$ is investigated in the next iteration. Figure (2.4) illustrates the first scenario.

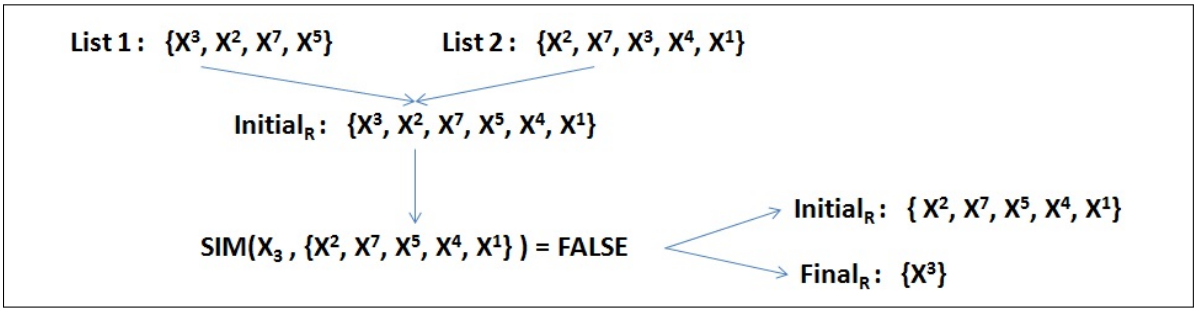


Figure 2.4: Illustrative example of the first scenario

Second scenario: If the SIM function returns the value ' TRUE ' then the feature Var has a strong link of similarity with one of the other features in $Initial_R$. Then, we check if these two features have divergent rankings in spite of their strong similarity.

First we use the function PLUS-SIM, which returns the most similar feature to Var in the list $Initial_R$. Then, we examine if the result of PLUS-SIM is equal to the feature with the next ranking of Var in $Initial_R$. In case the feature with the next ranking to the feature Var is the most similar, then Var and its neighbor are removed from $Initial_R$ and added to $Final_R$. Else we use the functions DIST-POS and PERMUT in order to move closer the similar features. More details are given in Algorithm (2.4) with a detailed description of the different functions used in this approach.

Algorithm 2.4 Rank aggregation based on similarity

Require: $Initial_R$: Initial rank aggregation.
Ensure: $Final_R$: Final rank list.

while $Initial_R == \varnothing$ **do**
  $Var = Initial_R[1]$.
  $Var_{list} = SUBLIST(Initial_R, 2)$.
  if SIM(Var, Var_{list})==FALSE **then**
    $Final_R$= CONCAT ($Final_R$, Var).
    $Initial_R$ =Var_{list}.
  else
    Var_{next}= $Var_{list}[1]$.
    if Var_{next} ==PLUS-SIM (Var, Var_{list}) **then**
      $Final_R$= CONCAT ($Final_R$, Var).
      $Final_R$= CONCAT ($Final_R$, Var_{next}).
      REMOVE(Var_{next}, Var_{list}).
      $Initial_R$ =Var_{list}.
    else
      while Var_{next}==PLUS-SIM (Var, Var_{list}) **do**
        if DIST-POS(Var-next ,PLUS-SIM (Var, Var_{list}), Var_{list}) > 1 **then**
          PERMUTE(Var-next, Var,$Initial_R$).
        else
          PERMUTE(PLUS-SIM (Var, $Initial_R$),Var-next,$Initial_R$).
        end if
      end while
    end if
  end if
end while
Return $Final_R$.

- **SIM(E, L)** return : false, true

  Takes a parameter list L and a feature E and check if feature E has a similarity with one of the elements of list L. If the similarity with one of the elements of the list is superior to 80 %, the function returns true elsewhere false.

- **CONCAT (L, E)** return : list

  Takes a parameter list L to be concatenated and appends the second argument E into the end of list L.

- **POS(E,L)** return: number

  Searches for feature E in List L, and returns its position in list L, or zero if feature E was not found in L.

- **PLUS-SIM(E, L)** return : feature

  Searches for a feature in list L with the biggest similarity to feature E.

- **SUBLIST(L, P)** return : list

  Returns a list of the elements in list L, starting at the specified position P in this list.

- **REMOVE(E,L)**

  Removes element E given as argument from list L.

- **DIST-POS(E1,E2,L)** return : number

  Counts the number of positions between two given elements E1 and E2 in list L.

- **PERMUT(E1,E2,L)**

  Swaps the position of two features E1 and E2 in list L.

Removing unwanted features

Once the selection trouble is solved and a consensus list of mutual features is obtained, we come across the issue of choosing the appropriate number of features to retain. In

fact a list of sorted features doesn't provide us with the optimal features subset. In general a predefined small number of features is retained from the consensus list in order to build the final model. If the number of used features is relatively small or big, then the final classification results may be degraded.

Despite the fact that most of the features that had a disjoint ranking in Section (2.3.3) are relevant, the underlying concepts can be concisely captured using only a few features, while keeping all of them has substantially detrimental effect on the credit model accuracy. So while we solve the problem of disjoint ranking, we use a marker to mark each pair of treated feature as similar items. A matrix MAT_S is then created in order to stock each pair of similar features, where each row of MAT_S contains a feature and their similar items. Then, we study each row of MAT_S by looking into the computed MI in order to identify the feature that supplies the most information about the target class. As a result the feature with the highest MI is kept and other similar features are removed from the aggregated list. Let's take the illustrative example used in Section (2.2.4). We suppose that after dealing with the problem of disjoint ranking we obtain list $\{X^3, X^2, X^5, X^7, X^4, X^1\}$, introduced before, features X^2 and X^5 are highly similar an while looking into the results of MI we notice that X^5 has the highest MI, consequently X^2 is removed from the list.

2.4 Experimental investigations

Our feature selection ensemble is composed by three different filter selection algorithms namely: Relief , χ^2 and MI (see Section (2.2)). These algorithms are available in Weka 3.7.0 machine learning package (Bouckaert et al. 2009).

The aggregation of these filters is first performed by our GA approach with Kendall (GA-K) and Spearman distances (GA-S) and then compared to the mean, median, highest rank or lowest rank aggregation (see Section (2.2.3)). These aggregation methods were tested using a Matlab implementation of the R package "RobustRankAggreg" written by Kolde et al. (2012), and compared to the results given by the individual feature selection methods. We use in this chapter four different classifiers,

namely DT, SVM, ANN and KNN. These four classifiers are available in Weka 3.7.0 machine learning package (Bouckaert et al. 2009).

The parameters setting for GA are given in Table (2.1). These parameters were chosen based on the result of several preliminary runs of the proposed approach. A 10-fold cross validation is used to compare the classifier's performance against others.

Table 2.1: Parameters of experimental environment for genetic algorithm.

Parameter	**Value**
Size of population	100
Mutation rate	0.001
Crossover rate	0.7
Number of generation	10

2.4.1 Results and discussion

First, the three feature selection methods: Relief, χ^2 and MI are applied to the datasets and three rankings of features are obtained. Next, the obtained rankings are aggregated using the available aggregation methods. Then, we pick a number of top-ranked features to get a few feature subsets. Then, DT, SVM, ANN and KNN classify the datasets using these feature subsets. Results are presented in Tables (2.3)-(2.6). The best results are shown in bold.

Table 2.2: Summary of the best performance results archived by the set of feature selection methods for the four datastes within the filter framework.

	DT	**SVM**	**ANN**	**KNN**
Relief				
MI				
χ^2			⊗	
Mean	⊗⊗	⊗	⊙⊗	
Median		⊗		
Highest rank			⊕	⊖
Lowest rank		⊗	⊗	
GA-K	⊖ ⊗ ⊕⊙ ⊕⊙ ⊖ ⊕ ⊙	⊖⊙ ⊖ ⊗ ⊕⊙ ⊖⊕ ⊖ ⊕ ⊙	⊖⊕ ⊖⊕ ⊖⊙ ⊖	⊖ ⊖ ⊗ ⊕⊙⊗ ⊕ ⊙
GA-S	⊖ ⊗ ⊕⊙⊖	⊕ ⊙	⊗ ⊕ ⊙ ⊕⊙	⊗ ⊕ ⊙ ⊖⊗ ⊕ ⊙

⊖ Precision, ⊗ Recall, ⊕ F-measure, ⊙ ROC Area.
Color: - Australian, - German , - HEMQ, - Tunisian.

From Table (2.2) we notice that for the Australian dataset, GA-K provides the highest precision in comparison with other feature selection methods except the case where DT is used as classifier, GA-S achieves the best precision. With the German dataset GA-K achieves the highest precision with all classifiers. For the HMEQ dataset GA-K achieves the highest precision with SVM and ANN classifiers while the highest precision of the other two classifiers are achieved by GA-S. Finally for the Tunisian dataset the highest precision rate is also achieved by GA-K aggregation for the DT, SVM and ANN classifiers while the highest precision rate for KNN classifier is achieved by highest rank aggregation.

We further investigate the recall results for the set of feature selection methods. For the Australian dataset the best recalls are achieved by GA-S for the DT and KNN while for the SVM and ANN classifiers the best rates are achieved by the lowest rank aggregation. Looking for German dataset results in Table (2.4) we see that the highest recall is achieved in three times by GA-K expect for ANN where the best recall is given by GA-S. Table (2.5) shows that the highest recall for the HMEQ dataset is obtained with GA-K for KNN and with the mean aggregation for the DT and SVM and χ^2 for ANN classifier. For the Tunisian dataset Table (2.6) shows that mean aggregation achieves twice the highest recall with ANN and DT followed by GA-S for the KNN classifier and median aggregation for SVM.

Table (2.3) shows that GA-S achieves the best F-measure three times except for KNN where the highest F-measure for the Australian dataset is achieved with GA-K. From Tables (2.4) and (2.5) we see that the highest recall is achieved by GA-K except for ANN where the highest rank aggregation gives the best performance with ANN and HMEQ dataset. Finally, Table (2.6) shows that GA-K achieves twice the highest recall with SVM and DT and GA-S did the same with ANN and KNN.

Table 2.3: Performance comparison of the new filter method and the other feature selection methods for the Australian dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
Relief	0.786	0.917	0.846	0.655
MI	0.930	0.870	0.900	0.642
χ^2	0.932	0.860	0.905	0.680
Mean	0.931	0.890	0.910	0.700
Median	0.931	0.888	0.909	0.713
Highest rank	0.920	0.943	0.931	0.689
Lowest rank	0.900	0.902	0.901	0.681
GA-K	0.946	0.923	0.934	0.727
GA-S	**0.952**	**0.950**	**0.951**	**0.762**
		Support Vector Machine		
Relief	0.795	0.898	0.843	0.702
MI	0.931	0.870	0.900	0.711
χ^2	0.918	0.935	0.927	0.690
Mean	0.923	0.943	0.928	0.720
Median	0.921	0.945	0.932	0.721
Highest rank	0.933	0.940	0.936	0.707
Lowest rank	0.894	**0.980**	0.935	0.705
GA-K	**0.945**	0.921	0.933	**0.898**
GA-S	0.943	0.942	**0.943**	0.890
		Artificial Neural Network		
Relief	0.885	0.926	0.905	0.653
MI	0.929	0.873	0.902	0.700
χ^2	0.926	0.924	0.926	0.683
Mean	0.927	0.934	0.931	0.752
Median	0.925	0.937	0.931	**0.755**
Highest rank	0.929	0.940	0.934	0.732
Lowest rank	0.896	**0.975**	0.933	0.726
GA-K	**0.931**	0.953	**0.941**	0.728
GA-S	0.929	0.883	0.905	0.742
		K-Nearest Neighbor		
Relief	0.784	0.881	0.829	0.801
MI	0.890	0.892	0.891	0.789
χ^2	0.912	0.799	0.851	0.788
Mean	0.920	0.886	0.902	0.825
Median	0.926	0.906	0.915	0.859
Highest rank	0.940	0.932	0.936	0.832
Lowest rank	0.944	0.930	0.937	0.834
GA-K	**0.950**	0.920	0.934	0.860
GA-S	0.942	**0.942**	**0.942**	**0.866**

For ROC area results we notice from Tables (2.4) and (2.5) that GA-K achieves the highest values except with German dataset and SVM where GA-S gives the best ROC area and respectively with the HMEQ dataset and ANN the GA-S gives the

Table 2.4: Performance comparison of the new filter method and the other feature selection methods for the German dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
Relief	0.682	0.555	0.669	0.631
MI	0.516	0.534	0.525	0.621
χ^2	0.737	0.477	0.579	0.600
Mean	0.750	0.542	0.612	0.682
Median	0.750	0.545	0.613	0.727
Highest rank	0.788	0.605	0.684	0.689
Lowest rank	0.700	0.642	0.669	0.760
GA-K	**0.792**	**0.701**	**0.743**	**0.795**
GA-S	0.756	0.697	0.725	0.789
		Support Vector Machine		
Relief	0.517	0.511	0.514	0.692
MI	0.603	0.534	0.566	0.701
χ^2	0.705	0.489	0.577	0.622
Mean	0.766	0.552	0.627	0.780
Median	0.756	0.560	0.643	0.781
Highest rank	0.762	0.623	0.685	0.766
Lowest rank	0.708	0.602	0.650	0.802
GA-K	**0.823**	**0.812**	**0.817**	**0.812**
GA-S	0.812	0.799	0.805	0.809
		Artificial Neural Network		
Relief	0.556	0.511	0.533	0.605
MI	0.612	0.534	0.572	0.589
χ^2	0.721	0.500	0.591	0.602
Mean	0.781	0.586	0.656	0.689
Median	0.778	0.591	0.671	0.677
Highest rank	0.770	0.600	0.674	0.678
Lowest rank	0.765	0.602	0.673	0.700
GA-K	**0.821**	0.706	**0.759**	0781
GA-S	0.819	**0.708**	**0.759**	**0.786**
		K-Nearest Neighbor		
Relief	0.703	0.688	0.695	0.702
MI	0.740	0.700	0.719	0.751
χ^2	0.697	0.700	0.698	0.743
Mean	0.751	0.723	0.736	0.750
Median	0.749	0.730	0.739	0.800
Highest rank	0.720	0.763	0.740	0.791
Lowest rank	0.719	0.758	0.738	0.802
GA-K	**0.820**	**0.811**	**0.815**	**0.813**
GA-S	0.817	0.800	0.808	0.810

best performance. For the Australian dataset the highest results are achieved twice with GA-S and once with the mean aggregation for ANN and with GA-K for SVM classifier. Finally, for the Tunisian dataset Table (2.6) shows that GA-K achieves the

Table 2.5: Performance comparison of the new filter method and the other feature selection methods for the HMEQ dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
Relief	0.747	0.800	0.736	0.782
MI	0.814	0.831	0.801	0.791
χ^2	0.818	0.832	0.798	0.760
Mean	0.821	**0.981**	0.887	0.786
Median	0.808	0.926	0.863	0.788
Highest rank	0.906	0.921	0.913	0.806
Lowest rank	0.842	0.922	0.880	0.805
GA-K	0.920	0.921	**0.921**	**0.822**
GA-S	**0.923**	0.912	0.917	0.815
		Support Vector Machine		
Relief	0.845	0.807	0.728	0.722
MI	0.822	0.828	0.784	0.755
χ^2	0.822	0.828	0.784	0.690
Mean	0.830	**0.987**	0.902	0.702
Median	0.823	0.906	0.862	0.689
Highest rank	0.905	0.945	0.924	0.744
Lowest rank	0.900	0.891	0.895	0.742
GA-K	**0.966**	0.933	**0.949**	0.810
GA-S	0.942	0.940	0.941	**0.812**
		Artificial Neural Network		
Relief	0.663	0.715	0.688	0.689
MI	0.681	0.788	0.730	0.781
χ^2	0.838	**0.974**	0.901	0.763
Mean	0.850	0.966	0.904	0.745
Median	0.848	0.971	0.905	0.723
Highest rank	0.897	0.842	**0.980**	0.788
Lowest rank	0.870	0.880	0.875	0.801
GA-K	**0.902**	0.972	0.935	**0.825**
GA-S	0.896	0.955	0.924	0.822
		K-Nearest Neighbor		
Relief	0.734	0.817	0.773	0.691
MI	0.805	0.820	0.812	0.799
χ^2	0.688	0.801	0.740	0.760
Mean	0.822	0.822	0.822	0.801
Median	0.842	0.820	0.830	0.810
Highest rank	0.830	0.826	0.828	0.808
Lowest rank	0.828	0.821	0.824	0.806
GA-K	0.843	**0.900**	**0.870**	**0.850**
GA-S	**0.850**	0.867	0.858	0.823

best results with DT and SVM while GA-S achieves the highest results with ANN and KNN.

The computed values or scores of recall, precision, and the F-measures are used to

Table 2.6: Performance comparison of the new filter method and the other feature selection methods for the Tunisian dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
Relief	0.876	0.888	0.882	0.681
MI	0.885	0.883	0.884	0.680
χ^2	0.876	0.880	0.879	0.623
Mean	0.860	**0.962**	0.913	0.796
Median	0.871	0.899	0.884	0.791
Highest rank	0.901	0.907	0.904	0.793
Lowest rank	0.889	0.902	0.895	0.799
GA-K	**0.922**	0.912	**0.917**	**0.813**
GA-S	0.917	0.908	0.912	0.811
		Support Vector Machine		
Relief	0.845	0.807	0.728	0.682
MI	0.822	0.828	0.784	0.651
χ^2	0.822	0.828	0.784	0.645
Mean	0.830	0.987	0.902	0.762
Median	0.889	**0.975**	0.930	0.755
Highest rank	0.922	0.907	0.914	0.800
Lowest rank	0.881	0.880	0.880	0.815
GA-K	**0.967**	0.952	**0.959**	**0.831**
GA-S	0.966	0.923	0.944	0.823
		Artificial Neural Network		
Relief	0.827	0.847	0.830	0.703
MI	0.822	0.852	0.826	0.700
χ^2	0.833	0.850	0.832	0.623
Mean	0.875	**0.964**	0.917	0.802
Median	0.881	0.951	0.914	0.801
Highest rank	0.905	0.901	0.894	0.729
Lowest rank	0.878	0.888	0.887	0.725
GA-K	**0.924**	0.902	0.912	0.822
GA-S	0.916	0.943	**0.929**	**0.826**
		K-Nearest Neighbor		
Relief	0.788	0.800	0.794	0.810
MI	0.821	0.688	0.748	0.799
χ^2	0.753	0.677	0.713	0.780
Mean	0.809	0.801	0.805	0.812
Median	0.811	0.799	0.805	0.821
Highest rank	**0.950**	0.688	0.700	0.798
Lowest rank	0.940	0.691	0.796	0.792
GA-K	0.889	0.888	0.888	0.900
GA-S	0.887	**0.901**	**0.893**	**0.905**

measure the performance of the feature selection methods. The differences between any two features selection methods may be due to chance or there is a significant difference between them. To rule out the possibility that the difference is due to

chance and to confirm our conclusions, statistical hypothesis testing is used.

Analysis of variance (ANOVA) is a particular form of statistical hypothesis testing mainly used in the analysis of experimental data to test the equality of three or more population means. Here, we are interested in determining whether the mean values of a given performance measure significantly differ accordingly with the used feature selection method and classification method. A two-way ANOVA is performed to test the difference between different features selection methods and classification methods. The first factor represent the different feature selection methods and the second represent the different classification methods. Then, ANOVA tests the null hypotheses that the means of all groups of factor 1 are equal, that the means of all groups of factor 2 are equal and the relationship between one factor and the dependent variable , i.e. level of F-measure, changes for different levels of the other factor. Then, H_0 and alternative hypothesis H_1 for factor 1 would be

$$\begin{cases} H_0 : \mu^1_{Relief} = \mu^1_{MI} = \mu^1_{\chi^2} = \mu^1_{Median} = \mu^1_{Mean} = \mu^1_{Highest} = \mu^1_{Lowest} \\ = \mu^1_{GA-S} = \mu^1_{GA-K} \text{ Performances of selection methods are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the feature selection methods mean performance is different} \\ \text{from the others} \end{cases}$$

H_0 and H_1 for factor 2 would be

$$\begin{cases} H_0 : \mu^2_{DT} = \mu^2_{SVM} = \mu^2_{ANN} = \mu^2_{KNN} \text{ Performances of classifiers are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the classifer mean performance is different from the others} \end{cases}$$

Interaction between the two factors is given by the following hypotheses:

$$\begin{cases} H_0 : \text{There is no interaction between the two factors,} \\ \text{versus} \\ H_1 : \text{There is an interaction between the two factors} \end{cases}$$

A significant F statistic suggests that we reject H_0. We use the data in Table (2.7) and ANOVA results are summarized in Table (2.8)

Table 2.7: Summary of F-measures for all feature selection methods with the four classification methods in filter framework.

	Relief	**MI**	χ^2	**Mean**	**Median**	**Highest**	**Lowest**	**GA-K**	**GA-S**
DT	0.856	0.900	0.905	0.910	0.909	0.931	0.901	0.934	0.951
	0.669	0.525	0.579	0.612	0.613	0.684	0.669	0.743	0.725
	0.736	0.801	0.798	0.887	0.863	0.913	0.88	0.921	0.917
	0.882	0.884	0.879	0.913	0.884	0.904	0.895	0.917	0.912
SVM	0.843	0.900	0.927	0.928	0.932	0.936	0.935	0.933	0.943
	0.514	0.566	0.577	0.627	0.643	0.685	0.650	0.817	0.805
	0.728	0.784	0.784	0.902	0.862	0.924	0.895	0.949	0.941
	0.728	0.784	0.784	0.902	0.930	0.914	0.880	0.959	0.944
ANN	0.905	0.902	0.926	0.931	0.931	0.934	0.933	0.941	0.905
	0.533	0.572	0.591	0.656	0.671	0.674	0.673	0.759	0.759
	0.688	0.730	0.901	0.904	0.905	0.98	0.875	0.935	0.924
	0.830	0.826	0.832	0.917	0.914	0.894	0.887	0.912	0.929
KNN	0.829	0.891	0.851	0.902	0.915	0.936	0.937	0.934	0.942
	0.695	0.719	0.698	0.736	0.739	0.74	0.738	0.815	0.808
	0.773	0.812	0.740	0.822	0.830	0.828	0.824	0.870	0.858
	0.794	0.748	0.713	0.805	0.805	0.700	0.796	0.888	0.893

Table 2.8: Tests of between-subjects effects in filter framework.

Source	**Type III Sum of Squares**	**DF**	**Mean Square**	**F**	**Sig. (p-value)**
Corrected Model	0.359^a	35	0.010	0.766	0.815
Intercept	98.109	1	98.109	7337.855	0.000
Selection Method	0.304	**8**	0.038	2.840	**0.007**
Classifier	0.006	**3**	0.002	0.160	**0.923**
Selection Method * Classifier	0.048	**24**	0.002	0.151	**1.000**
Error	1.444	108	0.013		
Total	99.912	144			
Corrected Total	1.802	143			

Dependant Variable : F-measure

The particular rows we are interested in are the " Selection Method", " Classifier " and " Selection Method * Classifier " rows, and these are highlighted above in red. These rows inform us whether our independent variables (the " Selection Method " and "Classifier" rows) and their interaction (the " Selection Method * Classifier" row) have a statistically significant effect on the dependent variable, "F-measure". It is important to first look at the " Selection Method * Classifier " interaction as this will determine how we can interpret our results.

We notice that we don't have a significant interaction between the two factors which means that the effect on the outcome of any specific level of F-measure change for one factor is the same for every fixed setting of the other factor.

We also report the results of "Selection Method" and "Classifier", but again, these needs to be interpreted in the context of the interaction result. We can see from the above table that there was no statistically significant difference in mean interest in F-measure between the different classifiers (p-value= 0.923), but there are statistically significant differences between the different feature selection methods (p-value =0.007).

ANOVA only tells us if there is any difference between the groups. If we want to know where the differences are, then we need to do some additional analysis. Then we use Tukey post hoc test in order to perform multiple comparisons for the different feature selection methods and we obtain a multiple comparisons table, as shown in Table (2.9).

From Table (2.9) we notice that there is some repetition of the results, but regardless of which row we choose to read from, we are interested in the differences between (1) GA-K and the individual feature selection methods, i.e. relief, MI and χ^2, (2) GA-S and the individual feature selection methods. From these results, we can see that there is a statistically significant difference between the obtained results from GA-K and GA-S and individual results (p-value < 0.05).

Table 2.9: Multiple comparisons table for feature selection methods in filter framework.

Selection method (I)	**Selection method (J)**	**Mean difference (I-J)**	**Sig.**	**Selection method (I)**	**Selection method (J)**	**Mean difference (I-J)**	**Sig.**
χ^2	GA-K	-.108875*	0.009	Mean	χ^2	0.054313	0.187
	GA-S	-.104438*	0.012		GA-K	-0.054562	0.185
	Highest	-0.06825	0.098		GA-S	-0.050125	0.223
	Lowest	-0.055188	0.18		Highest	-0.013937	0.734
	Mean	-0.054313	0.187		Lowest	-0.000875	0.983
	Median	-0.053813	0.191		Median	0.0005	0.99
	MI	0.008813	0.83		MI	0.063125	0.125
	Relief	0.030125	0.463		Relief	.084438*	0.041
GA-K	χ^2	0.108875*	0.009	Median	χ^2	0.053813	0.191
	GA-S	0.004437	0.914		GA-K	-0.055062	0.181
	Highest	0.040625	0.323		GA-S	-0.050625	0.218
	Lowest	0.053688	0.192		Highest	-0.014437	0.725
	Mean	0.054562	0.185		Lowest	-0.001375	0.973
	Median	0.055062	0.181		Mean	-0.0005	0.99
	MI	0.117688*	0.005		MI	0.062625	0.128
	Relief	0.139000*	0.001		Relief	0.083938*	0.042
GA-S	χ^2	0.104438*	0.012	MI	χ^2	-0.008813	0.83
	GA-K	-0.004437	0.914		GA-K	-0.117688*	0.005
	Highest	0.036188	0.378		GA-S	-0.113250*	0.007
	Lowest	0.04925	0.231		Highest	-0.077063	0.062
	Mean	0.050125	0.223		Lowest	-0.064	0.12
	Median	0.050625	0.218		Mean	-0.063125	0.125
	MI	0.113250*	0.007		Median	-0.062625	0.128
	Relief	0.134563*	0.001		Relief	0.021313	0.603
Highest	χ^2	0.06825	0.098	Relief	χ^2	-0.030125	0.463
	GA-K	-0.040625	0.323		GA-K	-0.139000*	0.001
	GA-S	-0.036188	0.378		GA-S	-0.134563*	0.001
	Lowest	0.013062	0.75		Highest	-0.098375*	0.018
	Mean	0.013937	0.734		Lowest	-0.085313*	0.039
	Median	0.014437	0.725		Mean	-0.084438*	0.041
	MI	0.077063	0.062		Median	-0.083938*	0.042
	Relief	0.098375*	0.018		MI	-0.021313	0.603
Lowest	χ^2	0.055188	0.18				
	GA-K	-0.053688	0.192				
	GA-S	-0.04925	0.231				
	Highest	-0.013062	0.75				
	Mean	0.000875	0.983				
	Median	0.001375	0.973				
	MI	0.064	0.12				
	Relief	0.085313*	0.039				

2.5 Conclusion

In this chapter, we investigated the effect of the fusion of a set of ranking methods. First, we conducted a preliminary study in which the issue of rank aggregation is presented as an optimization problem solved using GA and distance measures. Second we focused on solving the problem of disjoint ranking for similar features and choosing the right number of features from the final ranked list, for that we relate the similarity of the feature to their rankings. We evaluated the proposed approach on four credit datasets. Results show that there is generally a beneficial effect of aggregating feature rankings as compared to those produced by single methods. We also compared the proposed approach with four well known aggregation methods. Results are either superior or at least as adequate as those selected by the other aggregation methods. The second method for selecting the most important features is to use wrapper feature selection. Details about this method are presented in the next chapter.

Chapter 3

An Ensemble Wrapper Feature Selection Based on an Improved Exhaustive Search for Credit Scoring

Contents

3.1 Introduction

Wrappers feature selection usually selects a feature subset of the most relevant features with respect to the classification performance given by a particular classifier.

Although efficient, wrapper feature selection, as pointed out in the introduction, has some limitations due to the fact that their result depends on the search strategy and on using a single classifier in the evaluation process. Thus, we introduce a new approach based on ensemble methods that deals with the major issues of wrapper approach. As such, we give in Section (3.2) some details on wrapper framework and we discuss subset generation, search strategies and the use of multiple classifiers as an evaluation function. Then, we give the principal points of our new approach in Section (3.3). In Section (3.4) we present the results of recall, precision and F-measure obtained on four credit datasets.

3.2 Wrapper Framework

The main idea of wrapper feature selection is to remove unwanted features from the data by using the predictive accuracy of a particular classifier. It has been showen that wrappers generally outperform filters (Liu and Schumann 2005) in terms of accuracy since they are tuned to the specific interactions between the classifier and the dataset. However, wrapper methods have practical and theoretical limitations (Chrysostomou 2008). Wrappers typically lack generality since the resulting subset of features is tied to the bias of the classifier used in the evaluation function. The optimal feature subset will be specific to the classifier under consideration. Also, finding the optimal feature subset has a high computational cost. This cost depends on the number of times the classifier is trained on the evaluation process, on the number of subsets to be investigated and on the size of these feature subsets. In fact the number of subsets and their size depend on the used search strategy.

If a complete search is used the number of subsets will increase along with the time and if an heuristic is used not all subsets will be investigated and we may not have some interesting combination of features. In the following, we focus on two of the discussed wrapper's shortcomings: the bias of the classifier and subsets generation process.

3.2.1 Issue I: Evaluation using a single classifier

Using a single classifier in the wrapper process, may favor one candidate subset over the others. In fact the difference in biases and assumptions of each classifier may affect the final result in term of accuracy and execution time (Chrysostomou et al. 2008). According to Chrysostomou (2008), when a classifier used for evaluation is changed the set of selected features will change and as a result different levels of accuracy are obtained, inducing a lack of generality in the produced model. The level of complexity of the classifier is also a fundamental factor to be investigated. In theory when a complex classifier is used, it may take much longer to choose the best subset of features than a classifier considered to be simple. For example, when SVM is used as an evaluation function in the process of finding the best features subset, it may take more time to identify the most relevant features than using LR or KNN.

The number of classifiers used in the combination framework may also affect the evaluation process. If a small number of classifiers is used for feature selection, then it is likely that the level of agreement among them will be high. High agreement among classifiers may subsequently result in more relevant features being selected and differences in accuracy levels. However, if a high number of classifiers is used we may end up getting fewer relevant features. Indeed the level of agreement between classifiers will probably be low since more classifiers are required to agree on the relevance of a feature.

Based on the important limitation of using a single classifier, we consider using more than one classifier within wrapper feature selection framework to improve the general accuracy. In fact we look for mutually approved sets of significant features. Such sets will possibly give higher classification accuracies and reduce the biases of individual classifiers.

3.2.2 Issue II: Subset generation and search strategies

The ideal feature selection approach is the exhaustive search of the full set of features to find the optimal subset. However, as the number of features increases the

exhaustive search becomes rapidly impractical even for a moderate number of features (Chan et al. 2010), denoted by d. If we look at different ways in which features subsets are generated among many variations, three basic schemes are available in the literature namely forward selection, backward elimination and random scheme (Liu and Yu 2005).

Forward selection and backward elimination are considered as heuristics. Generally, sequential generation can help in getting a valid subset within a reasonable time but still it cannot find an optimal subset. This is due to the fact that the generation scheme uses an heuristic to obtain an optimal subset by selecting sequentially the best, as in the forward case, or removing the worst as in the backward case. Using such kind of generator will without doubt speed up the selection process. However, if the search falls in a local optima it cannot turn back. In fact the generator has no way to get out of the local optima because what has been removed cannot be added and what has been added cannot be removed. This is a big shortcoming of sequential schemes.

To overcome this problem we may use the random generation scheme, to add randomness to the fixed rule of sequential generation and avoid getting stuck at some local optima. Although random generation scheme could improve sequential results it still does not guarantee finding an optimal subset. This can be further elaborated in terms of search strategies (Yun et al. 2007) .

Hence, in order to minimize the search space, we propose to reduce the number of features by forward selection and backward elimination so that the exhaustive search method can handle the generation process within a realistic time. In this way, the selected feature set is much better in terms of accuracy than those from forward selection and backward elimination and the feature subsets are obtained much faster than the exhaustive method.

3.3 New approach for wrapper feature selection

In this section we propose a novel approach for wrapper feature selection. We consider building a three-stage wrapper feature selection model.

- At first, a based on similarity study with the prior knowledge primary dimensionality reduction step is conducted on the original feature space. This step is used to reduce the search space.
- Second, the subset generation step is performed using a mixture of heuristic and exhaustive search methods.
- The final step is the evaluation of wrapper feature selection and the effect of using multiple classifiers with different and similar nature.

3.3.1 Primary dimensionality reduction step: similarity study

The first step of our proposed approach is designed specifically to select less redundant features without sacrificing quality. Redundancy is measured by a similarity measure between a preselected set of features and the remaining features in the dataset. In this step we enhance an existing set of preselected features by adding additional features as a complement.

In CS we may already have a set of features preselected with prior information. In fact, experts in banks have years of experience on some particular category of credits and knowledge about which features are more important. This knowledge is generally obtained by years of use of classical feature selection methods. Thus, a possible improvement of the exhaustive search is to use the prior knowledge and to eliminate redundant features before generating the candidate subsets. Since our goal is to take advantage of any additional information about the feature, we may want to select a set of features complementary to those preselected by bank experts. Hence, we need to study the effect of using prior information on relevant feature complexity.

1. First, we split the features set in two sets. The first one regroups a set of features

that were assumed to be more relevant according to some prior knowledge. The second set contains the remaining ones.

2. Once the two sets are obtained we conduct a similarity study and a similarity matrix is constructed. In this step the MI is chosen as a similarity measure given its efficiency (see Chapter 1, Section 1.3 for more information about MI).

3. Then, we investigate level of similarity of each feature from the remaining set with the features of the first set. If the similarity is over 80%, the evaluated feature is eliminated else it is retained for further examination.

The first part of Figure (3.2) shows a simplified flow chart of the dimensionality reduction before the exhaustive search.

3.3.2 Subset generation step: speeding up exhaustive search by heuristics

Once the redundant feature elimination step is performed the search space is reduced. However, even though the search space is reduced by the previous step the search method still poses the problem of being computationally prohibitive. In fact an exhaustive search method is an enumerative search method that works by considering all possible features' combinations. According to Chan et al. (2010) this method is practical when the number of features is less than 10. Using more than 10 features would be costly in terms of computational time. In this case, specific heuristics can be used to reduce the set of candidate solutions to a manageable size. We think that using the first step combined with an heuristic will reduce the search space to less than 10 features, making the exhaustive search a realistic task. This, considering the fact that all datasets in this research have less than 40 features.

In theory each search strategy has its particular effects on the selected feature subset and on the performance of the induction algorithm. When the heuristic is changed the result may differ in terms of the number of selected features, As such, we extend the idea of ensemble method to search strategies. In the following we propose to perform several heuristics in order to get diverse results. We use both sequential

forward feature selection and backward feature elimination as a part of a combined feature selection. Figure (3.1) illustrates the proposed combination process for an example of 10 features.

In the first step, the forward selection and backward elimination methods are simultaneously applied to the reduced feature set resulting in two different intermediate feature lists. Each list includes a set of complementary variables.

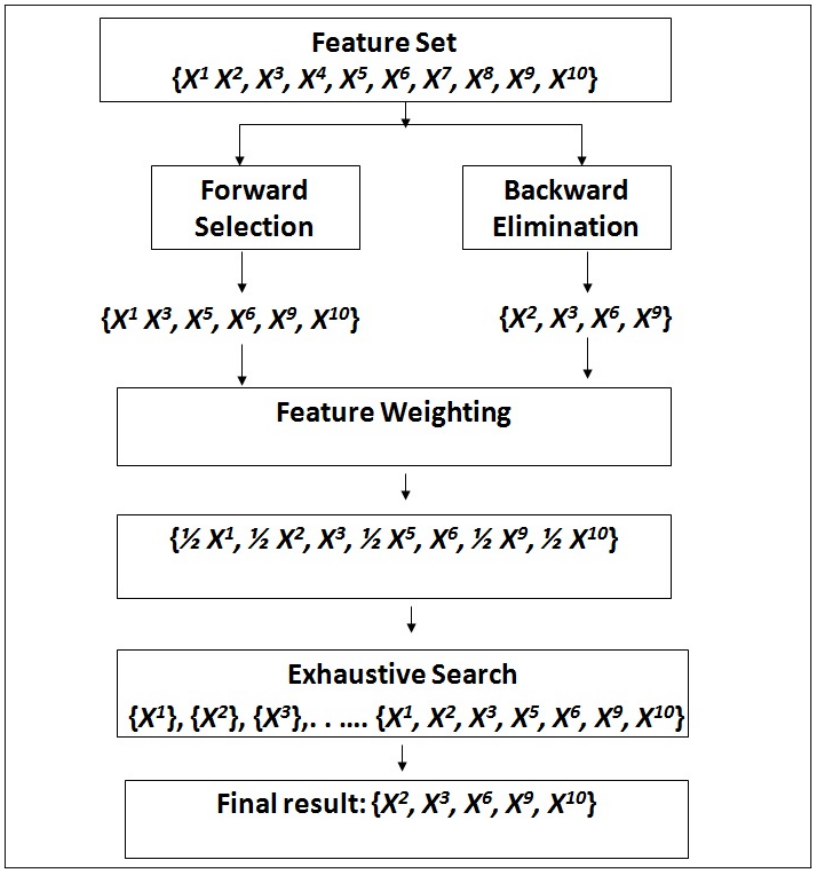


Figure 3.1: A flowchart combining heuristic and exhaustive search

In the second step, the two lists are merged into one single list of the most relevant features while the non selected features are eliminated. Since some selected features may appear in one of the intermediate feature lists and not in the other, these features must be re-weighted in order to take into consideration their relevance degree. A feature selected by both forward and backward selection is considered as more relevant than another feature selected only once. Consequently, the resulting features are then re-weighted according to their number of appearances in the intermediate lists.

Actually , the weight is equal to 1 if it appears in the two intermediate feature subsets, otherwise it is 0.5. In the third step, a complete search is used on the weighted features.

3.3.3 Evaluation step: effects of using multiple classifiers

Many classification methods were proposed to deal with the credit worthiness problem on the basis of information from past applicants. The most common statistical methods to evaluate applicants' solvability are LR and DA (Paleologo et al. 2010). Unfortunately, these two methods need some fundamental assumptions on data (Šušteršič et al. 2009). In addition to traditional methods different machine learning and artificial intelligence methods have been used such as: DT, ANN, SVM and many others. Although the majority of these methods are simple and do not need assumptions on data, they need a good mechanism to search for optimal model parameters and feature subsets.

Each of these individual methods produces a single discrimination rule and has some qualities and restrictions which may influence the feature evaluation process. No one can prove for sure the superiority of one classifier on another. Rather than to try to optimize the accuracy of one classifier, it is better to integrate multiple classifiers. This approach has been recognized to be successful, achieves better performance and has a higher precision of predictability in the learning process (Hsieh and Hung 2010; Chen and Li 2010). Here, the same ensemble concept is adopted in the feature evaluation process as part of the pre-processing course. Figure (3.2) shows how the results of a set of classifiers are merged to form a new evaluation function.

Classifiers

Many statistical classifiers are based on many assumptions as normality distribution and absence of multicollinearity. However, if the data are not normal the statistical family is not appropriate.

The chosen algorithms in this study are representative of the most popular family

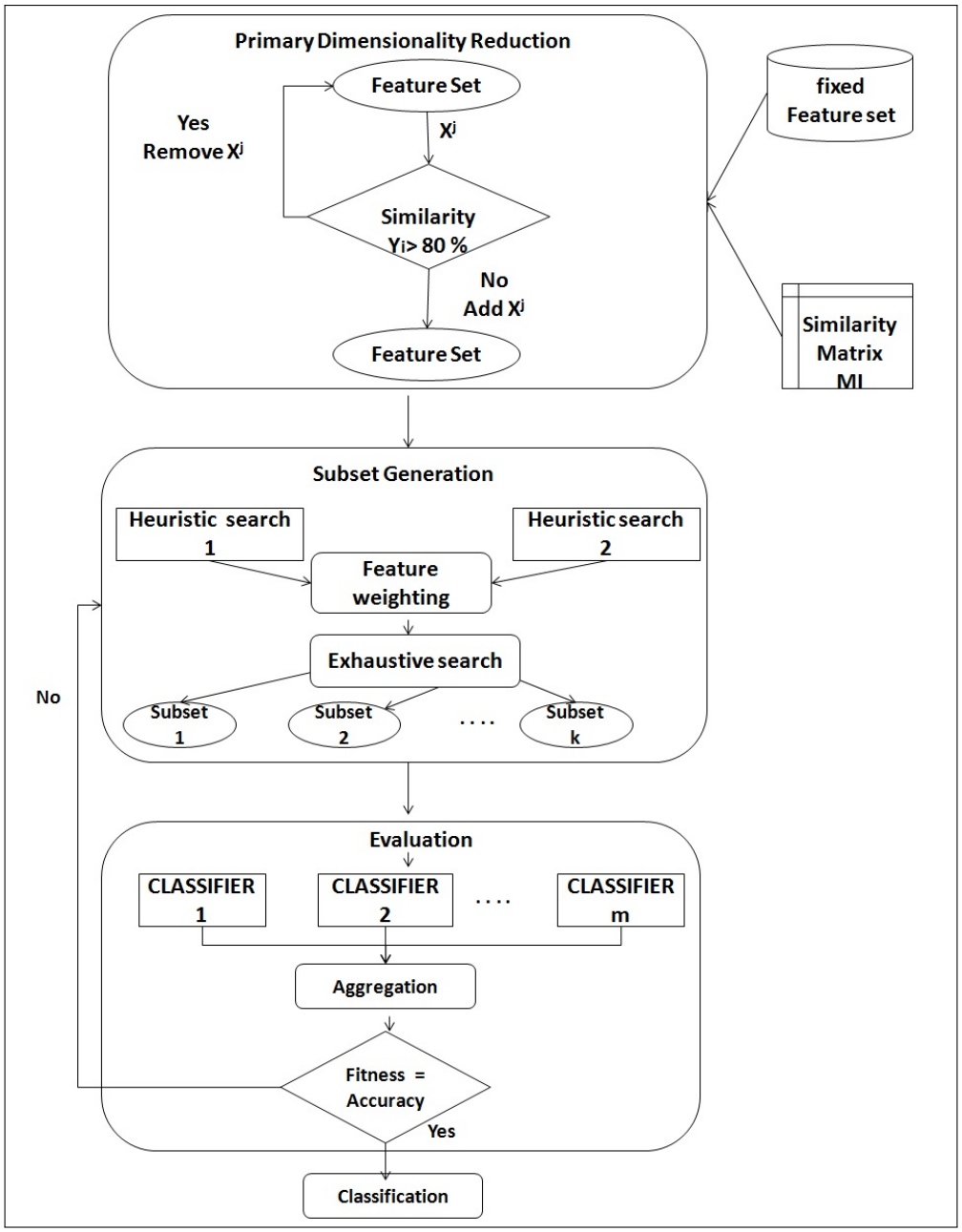


Figure 3.2: A wrapper approach combing multiple classifiers for feature selection.

of classifier models that were selected to form committees of experts in order to test various classifier combination schemes. Therefore, this section focuses only on the general aspect of each family. A conceptual description of the chosen algorithms is given in Table (3.1) more details are given in Appendix (B).

Among the most popular four classifier models were selected namely DT, SVM, KNN, and ANN.

Table 3.1: General properties of some classification algorithms.

	Classification Algorithms						
	Properties	**DA**	**LR**	**DT**	**SVM**	**ANN**	**KNN**
Assumptions	numeric variables	yes					yes
	normally distributed variables	yes					
	equal covariance matrices	yes					
	problem of interaction	yes	yes				
	problem of multicollinearity	yes	yes				
	normalization of variables				yes		yes
Output	Score	yes	yes	yes		yes	yes
	Class			yes	yes	yes	yes

Classifier arrangement approaches

In this section two different classifier arrangement approaches are used within the wrapper evaluation process, namely the same-type approach and the mixed-type approach. The same-type approach combines classifiers from the same family and uses them within the wrapper framework to select the relevant features. For example, classifiers belonging to SVM family are combined together. The mixed-type approach combines classifiers from different families.

Table 3.2: Summary of used classifiers within each family.

DT	ANN	KNN	SVM
J48	MultilayerPerceptron (MP)	K=1 (1NN)	Polynomial (SVMP)
RandomForest (RF)	VotedPerceptron (VP)	K=5 (5NN)	Radial (SVMR)

The same-type combinations use classifiers from the four different families discussed before. More precisely, Two classifiers from DT family, two from ANN family, one from KNN family is used with two different number of neighbors K= 1 and K=5 and one classifier is used from SVM family. The chosen SVM classifier is used with two different kernels, namely the polynomial and radial basis function kernel. In this way, features that are related to both linear aspects and non-linear aspects can be identified. All considered classifiers are summarized in Table (3.2).

By using the second arrangement approach we investigate how classifiers from different families work together and how their interaction affects features' selection.

Classifiers are combined using an exhaustive approach so that each classifier is used with every other classifier from a different nature. This leads to the construction of a total of 76 mixed-type classifier combinations, described in Tables (3.3)-(3.4) including 24 for 2-classifier mixed-type combinations and 52 for 3-mixed-type combinations. In this way, both approaches help us obtain a complete picture of the effects of the nature and number of classifiers on feature selection.

Table 3.3: Summary of all possible combination of two classifiers.

Possible combinations
(J48+ SVMP), (J48+ SVMR), (J48+ MP), (J48+ VP), (J48 +1NN), (J48+5NN), (RF+ SVMP), (RF+ SVMR), (RF+ MP), (RF+ VP), (RF +1NN), (RF+5NN),(SVMP + MP), (SVMP + VP), (SVMP +1NN), (SVMP +5NN), (SVMR + MP), (SVMR + VP), (SVMR +1NN), (SVMR +5NN), (MP +1NN), (MP +5NN), (VP +1NN), (VP +5NN).

Aggregation rules

Traditionally, the approach used to build a multi-classifiers system is to experimentally compare the performance of several classifiers and select the best one. However, many alternative approaches based on combining multiple classifiers have emerged (Kuncheva et al. 2001). There are basically two classifier combination scenarios. In the first, all classifiers use the same representation of the input example. In this case, each classifier, for a given input example, produces an estimate of the same posteriori class probability. In the second scenario, each classifier uses its only representation of the input example. For multiple classifiers using distinct representations, many existing schemes can be considered, where all the representations are used jointly to make a decision. We can derive the commonly used classifier combination schemes such as the product rule, average rule, minimum rule, maximum rule and majority voting schemes.

Table 3.4: Summary of all possible combination of three classifiers.

Possible combinations
(J48 +RF + SVMP), (J48 +RF+ SVMR), (J48 +RF + MP), (J48 +RF +VP),
(J48 +RF +1NN), (J48 +RF +5NN), (J48+ SVMP+ SVMR), (J48+ MP + VP),
(J48 +1NN+5NN), (J48+ SVMP + MP),(J48+ SVMP + VP), (j48+SVMP +1NN),
(J48+ SVMP +5NN), (J48+ SVMR + MP), (J48+ SVMR + VP), (J48+ SVMR +1NN),
(J48+ SVMR +5NN), (J48+ MP +1NN), (J48+ MP +5NN), (J48+ VP +1NN),
(J48+ VP +5NN), (RF + SVMP+ SVMR), (RF + MP + VP), (RF +1NN+5NN),
(RF + SVMP + MP), (RF + SVMP + VP), (RF+SVMP +1NN), (RF + SVMP +5NN),
(RF + SVMR + MP), (RF + SVMR + VP), (RF+SVMR +1NN), (RF + SVMR +5NN),
(RF + MP +1NN), (RF + MP +5NN), (RF + VP +1NN), (RF + VP +5NN),
(SVMP+SVMR + MP), (SVMP+SVMR +VP), (SVMP+SVMR +1NN), (SVMP+SVMR +5NN),
(SVMP + MP + VP), (SVMP +1NN+5NN), (SVMP + MP +1NN), (SVMP + MP +5NN),
(SVMP + VP +1NN), (SVMP + VP +5NN), (SVMR + MP + VP), (SVMR +1NN+5NN),
(SVMR + MP +1NN), (SVMR + MP +5NN), (SVMR + VP +1NN), (SVMR + VP +5NN).

The simplest and most common way for aggregation is to use a simple arithmetic mean also known as the average. This operator is interesting because it gives an aggregated value that is smaller than the greatest argument and bigger than the smallest one. Then, the resulting aggregation is "a middle value". This property

is known as the compensation property. The minimum and the maximum are also basic aggregation operators. The minimum gives the smallest value of a set, while the maximum gives the greatest one (Kittler 1998). Majority vote is also a common classifier combination method, particularly used in classifier ensembles when the class labels of the classifiers are crisp (Kuncheva et al. 2001). In general, majority voting is a simple method that does not require any parameters to be trained or any additional information for later results.

3.4 Experimental Investigations

The precision, recall, F-measure and ROC area of feature subsets selected from different combinations are given in Tables (3.5)-(3.8) for the four datasets using a 10-fold cross validation. The best results are shown in bold.

Two approaches for wrapper evaluation are presented, namely the same-type approach and the mixed-type approach. Results for the first approach are investigated in Section (3.4.1) and those for the second in Section (3.4.2).

3.4.1 Results and discussion for the same-type approach

Analysis of features selected by DT family combination

Looking to the results produced by DT family in Tables (3.5)-(3.8), we notice that the J48 classifier achieves in most cases the best individual results for the German, HMEQ and the Tunisian datasets, expect for the Australian dataset where the individual results produced by SVM were slightly better. The good performance of the wrapper using DT classifiers is guided by the nature of this family which is well known for its highly accurate performance on financial data (Piramuthu 2004).

A closer look at Tables (3.5)-(3.8) shows that results are much better within the combination process. Actually, some DT algorithms adopt local search strategy while others are global optimized algorithms. Then, combing a set of DT algorithms may avoid some of their drawbacks and experimental results show that combination results are more effective than individual ones.

Table 3.5: Performance comparison of the new wrapper method and the other feature selection methods for the Australian dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
J48	0.867	0.855	0.855	0.862
RF	0.863	0.851	0.851	0.858
Average	0.782	**0.925**	0.848	0.863
Product	0.864	0.852	0.853	0.859
Maximum	**0.930**	0.794	**0.856**	0.859
Minimum	0.866	0.855	0.855	0.862
Majority Vote	0.782	0.922	0.846	**0.865**
		Support Vector Machine		
SVMP	0.921	0.794	0.853	0.855
SVMR	**0.930**	0.799	**0.860**	0.862
Average	0.787	**0.925**	0.850	**0.864**
Product	0.866	0.855	0.855	0.861
Maximum	0.859	0.848	0.848	0.856
Minimum	0.927	0.794	0.855	0.858
Majority Vote	0.781	0.915	0.848	0.857
		Artificial Neural Network		
MP	0.860	0.849	0.850	0.856
VP	0.859	0.848	0.848	0.855
Average	0.862	0.851	0.851	0.857
Product	0.783	**0.919**	**0.861**	**0.860**
Maximum	0.862	0.851	0.851	0.857
Minimum	0.862	0.851	0.851	0.857
Majority Vote	**0.864**	0.853	0.854	0.858
		K-Nearest Neighbor		
1NN	**0.865**	0.852	**0.852**	0.860
5NN	0.859	0.848	0.848	0.855
Average	0.812	**0.890**	0.849	0.877
Product	0.811	0.866	0.838	**0.883**
Maximum	0.820	0.880	0.849	0.875
Minimum	0.824	0.823	0.822	0.876
Majority Vote	0.853	0.851	0.851	0.882

As expected, combination schemes have approximately the same performance. Although the product, minimum and the maximum rules, seem to have the best precision rates while the average rule and the majority vote rule give the best recall and ROC area for DT family.

DT classifiers are sometimes considered as embedded methods. These kind of

methods essentially perform feature selection within the learning process, which means that they are able to select relevant features on their own: using their own search strategy and splitting mechanism. In other words DT classifiers select relevant features at two different stages. In the first stage features are selected by individual classifiers and in the second features are selected by the combination of DT classifiers. In this way, only features that are selected at both stages will form the final feature subset which is very likely to include features of high relevance.

Analysis of features selected by SVM family combination

We notice from Tables (3.5)-(3.8) some differences among the individual results of polynomial and radial SVM. For the four datasets, we notice that the performance with the radial SVM is slightly better. This result is due to the nature of the two kernels. In general the polynomial kernel looks for linear characteristics within datasets while the radial kernel identifies linear and non-linear aspects of the datasets.

Overall we notice that the same-type combinations with SVM improves the performance, meaning that the selected features within the combination process are more suitable for the CS task. Tables (3.5)-(3.8) show that the model performance changes with the different combination rules. We notice that the four performance measures increase with the combination. In fact, majority vote, minimum and average rule combination give significantly higher ROC area and F-measure.

Analysis of features selected by KNN family combination

Tables (3.5)-(3.8) show that KNN classifiers give good results within the combination framework. The high performance of the obtained combinations using the KNN family is the result of its natural simplicity. In fact KNN is a non-parametric classification method that does not assume any parametric distribution of the random variables. Non-parametric models are very flexible making them usually good classifiers for many situations (Li et al. 2011). The main advantage of KNN is that it can learn from a small set of examples explaining the good performance with the Australian and the

Table 3.6: Performance comparison of the new wrapper method and the other feature selection methods for the German dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
J48	0.735	0.750	0.723	**0.635**
RF	0.686	0.716	0.665	0.570
Average	0.740	0.930	0.824	0.583
Product	0.732	0.933	0.820	0.568
Maximum	0.741	0.930	0.825	0.585
Minimum	**0.744**	0.929	**0.826**	0.591
Majority Vote	0.740	**0.934**	**0.826**	**0.635**
		Support Vector Machine		
SVMP	0.490	0.700	0.576	0.500
SVMR	**0.708**	**0.728**	**0.709**	**0.627**
Average	0.695	0.722	0.678	0.583
Product	0.682	0.714	0.664	0.568
Maximum	0.697	0.723	0.680	0.585
Minimum	0.702	0.726	0.685	0.591
Majority Vote	0.699	0.724	0.679	0.584
		Artificial Neural Network		
MP	0.719	0.738	0.717	0.634
VP	0.703	0.726	0.701	0.614
Average	**0.769**	0.896	0.827	0.634
Product	**0.769**	0.894	0.825	**0.645**
Maximum	0.758	0.894	0.820	0.643
Minimum	0.717	0.737	0.712	0.625
Majority Vote	0.764	**0.904**	**0.828**	0.625
		K-Nearest Neighbor		
1NN	0.699	0.724	0.677	0.582
5NN	0.691	0.718	0.688	0.598
Average	0.745	0.917	0.822	0.592
Product	0.739	**0.937**	**0.826**	**0.601**
Maximum	**0.749**	0.899	0.817	0.597
Minimum	0.745	0.917	0.822	0.592
Majority Vote	0.742	0.914	0.819	0.587

German datasets. On the other hand, its major disadvantage is being computationally intensive for large datasets since it uses all training data as the examples (Thomas et al. 2002).

Table 3.7: Performance comparison of the new wrapper method and the other feature selection methods for the HMEQ dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
J48	0.859	0.864	0.844	0.795
RF	0.857	0.860	0.838	0.785
Average	0.867	0.982	**0.921**	0.793
Product	0.863	**0.983**	0.918	0.787
Maximum	**0.914**	0.899	0.906	**0.809**
Minimum	0.855	0.852	0.853	0.806
Majority Vote	0.868	0.979	0.920	0.797
		Support Vector Machine		
SVMP	0.633	0.796	0.705	0.555
SVMR	**0.843**	0.804	0.724	0.619
Average	0.827	0.977	0.896	0.701
Product	0.809	0.815	0.759	0.662
Maximum	0.816	0.822	0.774	0.683
Minimum	0.800	0.819	0.778	**0.691**
Majority Vote	0.824	**0.987**	**0.898**	0.682
		Artificial Neural Network		
MP	0.693	0.638	0.664	0.677
VP	0.81	0.827	0.789	0.607
Average	0.868	0.871	0.869	0.877
Product	0.835	**0.977**	**0.902**	0.602
Maximum	0.811	0.829	0.793	0.734
Minimum	0.838	0.974	0.901	0.732
Majority Vote	**0.911**	0.930	0.920	**0.879**
		K-Nearest Neighbor		
1NN	0.852	0.837	0.791	0.803
5NN	0.837	0.824	0.766	0.812
Average	0.821	**0.998**	0.901	**0.891**
Product	0.850	0.825	0.766	0.881
Maximum	**0.889**	0.997	**0.940**	0.889
Minimum	0.821	0.996	0.900	0.842
Majority Vote	0.832	0.996	0.907	0.844

Analysis of features selected by ANN family combination

From Tables (3.5)-(3.8) we notice that, as with the combination from other classifier families, the final result has improved using ANN combinations. ANN classifiers are excellent to extract information from a dataset. During the training process ANN can

be used to map an input to desired output, classify data or learn patterns. Hence, ANN can also be used to perform indirectly feature selection (Ledesma et al. 2008).

Table 3.8: Performance comparison of the new wrapper method and the other feature selection methods for the Tunisian dataset.

	Precision	**Recall**	**F-Measure**	**ROC Area**
		Decision Tree		
J48	0.722	0.850	0.781	0.597
RF	0.797	0.846	0.801	**0.695**
Average	0.858	0.985	0.917	0.652
Product	0.859	0.985	0.918	0.655
Maximum	**0.866**	0.985	**0.921**	0.653
Minimum	0.861	**0.986**	0.919	0.644
Majority Vote	0.858	0.987	0.917	0.649
		Support Vector Machine		
SVMP	0.722	0.850	0.781	0.500
SVMR	0.797	0.837	0.805	0.566
Average	**0.861**	0.962	0.909	**0.666**
Product	0.710	0.842	0.770	0.500
Maximum	0.860	**0.968**	**0.911**	0.563
Minimum	0.798	0.839	0.803	0.661
Majority Vote	0.859	**0.968**	0.910	0.656
		Artificial Neural Network		
MP	0.802	0.843	0.800	0.577
VP	0.826	0.857	0.816	0.562
Average	0.856	0.979	0.913	0.677
Product	0.865	**0.984**	**0.921**	0.659
Maximum	0.867	0.975	0.918	0.668
Minimum	**0.888**	0.855	0.871	**0.731**
Majority Vote	0.866	0.981	0.920	0.657
		K-Nearest Neighbor		
1NN	0.785	0.843	0.794	0.680
5NN	0.792	0.844	0.800	0.685
Average	0.855	0.977	0.912	**0.775**
Product	0.852	0.993	0.917	0.756
Maximum	0.864	0.925	0.893	0.746
Minimum	0.863	0.932	0.896	0.704
Majority Vote	**0.866**	**0.985**	**0.921**	0.753

In this section we investigated the effect of classifiers nature on the final feature selection results. It is interesting to know if the observed results are only due to classifiers nature or to their interactions with the aggregation methods. Hence

we use a two-way ANOVA to analyze if the mean values of F-measure significantly change along with the levels of the two independent variables classifier and aggregation method. The first independent variable classifier presents the first factor in ANOVA where DT, SVM, ANN and KNN are the levels of this variable. Aggregation method denote the second factor in ANOVA where $\{Average, Product, Maximum, Minimum, MajorityVote\}$ are the levels of this second factor. To test the interaction we use the hypotheses presented below:

For the first factor, i.e. Classifier, H_0 and H_1 are given by:

$$\begin{cases} H_0 : \mu^1_{DT} = \mu^1_{SVM} = \mu^1_{ANN} = \mu^1_{KNN} \text{ Performances of classifiers are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the classifer mean performance is different from the others} \end{cases}$$

H_0 and H_1 for factor 2, i.e. aggregation method, would be

$$\begin{cases} H_0 : \mu^2_{Aver} = \mu^2_{Prod} = \mu^2_{Max} = \mu^2_{Min} = \mu^2_{MajV} \text{ Performances of aggregation methods are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the aggregation methods mean performance is different from the others} \end{cases}$$

Interaction between the two 2 factors:

$$\begin{cases} H_0 : \text{There is no interaction between the two factors,} \\ \text{versus} \\ H_1 : \text{There is an interaction between the two factors mean performance} \end{cases}$$

To set up a two-way ANOVA we use the data in Table (3.9) and obtained results are summarized in Table (3.10)

The obtained result of the two-way ANOVA in Table (3.10) show that we don't have a significant interaction between the two factors which means that the effect on the outcome of any specific level of F-measure change for one factor is the same for

Table 3.9: Summary of F-measures for all aggregation methods with the four classification methods in wrapper framework.

	Average	**Product**	**Maximum**	**Minimum**	**Majority Vote**
DT	0.848	0.853	0.856	0.855	0.846
	0.824	0.820	0.825	0.826	0.826
	0.921	0.918	0.906	0.853	0.920
	0.917	0.918	0.921	0.919	0.917
SVM	0.850	0.855	0.848	0.855	0.848
	0.678	0.664	0.680	0.685	0.679
	0.896	0.759	0.774	0.778	0.898
	0.909	0.770	0.911	0.803	0.910
ANN	0.851	0.861	0.851	0.851	0.854
	0.827	0.825	0.820	0.712	0.828
	0.869	0.902	0.793	0.901	0.920
	0.913	0.921	0.918	0.871	0.920
KNN	0.849	0.838	0.849	0.822	0.851
	0.822	0.826	0.817	0.822	0.819
	0.901	0.766	0.940	0.900	0.907
	0.912	0.917	0.893	0.896	0.921

Table 3.10: Tests of between-subjects effects in wrapper framework.

Source	**Type III Sum of Squares**	**DF**	**Mean Square**	**F**	**Sig. (p-value)**
Corrected Model	0.090^a	19	0.005	1.154	0.326
Intercept	57.826	1	57.826	14028.038	0.000
Aggregation Method	0.013	**4**	0.003	0.768	**0.550**
Classifier	0.063	**3**	0.021	5.081	**0.003**
Aggregation Method * Classifier	0.015	**12**	0.001	0.301	**0.987**
Error	0.247	60	0.004		
Total	58.163	80			
Corrected Total	0.338	79			

Dependant Variable : F-measure

every fixed setting of the other factor. We notice from Table (3.10) that there is no statistically significant difference in mean interest in F-measure between the different aggregation methods (p-value= 0.550), but there is statistically significant difference between different classifiers (p-value =0.003).

When ANOVA gives a significant result for one classification methods, this indicates that at least one classifier results differs from the other classifiers. Yet, ANOVA test does not indicate which classifier results influenced the reject of H_0. In order to analyze the pattern of difference between means, we conduct a pairwise comparison. Results of pairwise comparisons for classifiers are given in Table (3.11).

Table 3.11: Multiple comparisons table for classifier levels in wrapper framework.

Classifier (I)	**Classifier (J)**	**Mean difference (I-J)**	**Sig.**
ANN	DT	-0.01405	0.900
	KNN	-0.003	0.999
	SVM	0.05790*	0.030
DT	ANN	0.01405	0.900
	KNN	0.01105	0.948
	SVM	0.07195*	0.004
KNN	ANN	0.003	0.999
	DT	-0.01105	0.948
	SVM	0.06090*	0.020
SVM	ANN	-.05790*	0.030
	DT	-0.07195*	0.004
	KNN	-0.06090*	0.020

From Table (3.11) we notice that there is a statistically significant difference between the obtained results from SVM and the others classifications.

3.4.2 Results and discussion for the mixed-type approach

Because of the large number of combinations, the mixed-type approach is examined using only the Australian dataset and results are summarized in Tables (3.12) and (3.13).

Table (3.12) presents results for 2-classifiers mixed combination while Table (3.13) presents those for 3-classifiers mixed combination. We need to investigate the effect of classifiers nature on feature selection and if the number of classifiers within the combination framework also affects the feature selection. Tables (3.12) and (3.13) give:

- The measured F-measure for the features generated by different combinations
- The mean number of evaluated subsets, i.e. the first number between the parentheses and the associated mean number of selected features, i.e. the second number between the parentheses.

From Tables (3.12) and (3.13) we notice that the combination with few classifiers selected the features that achieved the best F-measure with a smaller number of

Table 3.12: Total number of evaluated subsets and selected features by 2 classifiers mixed-type combinations and associated F-measure rates for the Australian Dataset.

Lowest F-measure lies between 0.847 **and** 0.855	**Intermediate F-measure lies between** 0.856 **and** 0.859	**Highest F-measure lies between** 0.860 **and** 0.874
j48+1NN (79,3)	J48+ SVMP(106,4)	J48+ MP(116,7)
RF+SVMR(82,2)	J48+ SVMR(106,4)	RF+SVMP(79,3)
RF+MP(111,6)	J48 + VP(120,7)	RF+1NN(96,4)
RF+VP(104,5)	j48+5NN(105,4)	SVMP+VP(88,4)
RF+5NN(96,4)	SVMP+MP(112,5)	SVMP+1NN(79,3)
SVMP+5NN(116,6)	SVMR+MP(112,7)	MP+5NN(121,7)
SVMR+1NN(79,3)	SVMR+VP(116,7)	VP+5NN(117,7)
SVMR+5NN(127,9)		
MP+1NN(107,6)		
VP+1NN(127,6)		

evaluated subsets. More specifically, the 2-classifiers' combinations produce an F-measure in the range $[0.860, 0.874]$ with a number of evaluated subsets that do not exceed 121 evaluations. On the other hand the 3-classifiers' combination gives the same rate but with a much higher number of evaluated subsets.

Table (3.12) shows that combining DT classifiers with ANN or KNN classifiers generally yields the lowest F-measure (RF+MP, RF+VP, RF+5NN, J48+1NN) and this is due to the difference in nature between these three types of classifiers. Actually, ANN classifiers identify relationships between features based on the available prior knowledge about the actual features in the dataset. However, KNN classifiers select the most relevant features with the closest distance to a set of specified features called neighbors. For this family the resulting features depend of the number of chosen neighbors. DT classifiers nature is very dissimilar to the nature of ANN and KNN. In fact, they use a statistical measure to evaluate the relevance of features.

Table (3.13) shows that the majority of combinations with SVM classifiers selected a set of features that achieved the best rates of F-measure, specially the case when SVM classifiers are combined with KNN classifiers. The fact that these combinations lead to high F-measure, despite the fact that they consider classifiers from different families, could be due to the existence of particular similarities between these two families. KNN classifiers use a distance metric to decide which are the most relevant

Table 3.13: Total number of evaluated subsets and selected features by 3 classifiers mixed-type combinations and associated F-measure rates for the Australian Dataset.

Lowest F-measure lies between 0.847 **and** 0.855	**Intermediate F-measure lies between** 0.856 **and** 0.859	**Highest F-measure lies between** 0.860 **and** 0.874
J48+RF+MV(136,7)	J48+RF+SVMP(82,2)	J48+RF+1NN(79,3)
J48+RF+5NN(131,9)	J48+RF+SVMR(75,2)	J48+VP+1NN(139,7)
J48+1NN+5NN(114,7)	J48+RF+MP(144,10)	RF+MP+VP(111,6)
J48+SVMR+MP (146,7)	J48+MP+VM(126,7)	RF+SVMP+MP(132,6)
J48+SVMR+1NN(75,2)	J48+SVMP+SVMR(75,2)	RF+SVMP+VP(120,8)
J48+SVMR+5NN(141,10)	J48+SVMP+MP(126,7)	RF+SVMP+1NN (116,7)
J48+MP+5NN(122,7)	J48+SVMP+VP(139,6)	RF+SVMR+MP (126,9)
J48+VP+5NN(112,7)	J48+SVMP+1NN(118,6)	RF+SVMR+VP(135,7)
RF+1NN+5NN (79,3)	J48+SVMP+5NN(139,10)	SVMP+1NN+5NN(108,7)
RF+SVMP+5NN (94,5)	J48+SVMR+VP(189,7)	SVMP+MP+1NN(132,8)
RF+SVMR+1NN (88,3)	J48+MP+1NN(165,10)	SVMR+MP+5NN(132,10)
RF+SVMR+5NN (117,8)	RF+SVMP+SVMR(82,2)	SVMP+VP+1NN(118,10)
RF+MP+1NN(130,7)	MP+SVMP+SVMR (139,6)	SVMR+1NN+5NN(108,7)
RF+MP+5NN (133,10)	VP+SVMP+SVMR (120,5)	SVMR+MP+1NN (132,9)
RF+VP+1NN(130,7)	1NN+SVMP+SVMR (82,2)	SVMR+MP+5NN(149,9)
RF+VP+5NN(123,10)	5NN+SVMP+SVMR (75,2)	SVMR+VP+5NN (149,9)
	SVMP+MP+VP (109,5)	
	SVMP+VP+5NN (153,11)	
	SVMR+MP+VP (109,5)	
	SVMR+VP+1NN(122,8)	

features to the target variable. SVM classifiers use a distance to select the most relevant features by measuring the distance between each feature in accordance with the hyper-plane that separates the best class from the target concept.

3.5 Conclusion

In this chapter we developed an ensemble wrapper feature selection approach for a CS application. The proposed approach is composed of three stages. In the first one we performed a dimensionality reduction using bank experts' knowledge. In the second stage a heuristic is used to reduce the search space to less than 10 features, which easer

the exhaustive search. In the final stage the generated subsets are evaluated using a multi-classifiers process involving two arrangement approaches, namely the same-type and mixed-type approach. From the three stages, we show that the use of prior information on relevant features effectively induces a significant gain in complexity with improved generalization. Also, we have shown that the number of classifiers and their nature have an important effect on wrapper feature selection results. This chapter and the previous one discussed two different concepts of feature selection namely filter and wrapper methods, their combination will be investigated in the next chapter.

Chapter 4

A Three-Stage Feature Selection Using Quadratic Programming for Credit Scoring

Contents

4.1 Introduction

Chapters (2) and (3) reviewed the two most important methods for feature selection, respectively the filter and the wrapper feature selection methods with proposed modifications for improvement. In general, we cannot show the superiority of one approach over the other, because of the fact that there are strong mixed arguments in favor of

both methods. This chapter explores a variety of filter and wrapper feature selection methods to reduce non relevant features. These two types of selection methods are complementary to each other. A fusion strategy is then proposed to sequentially combine the ranking criteria of multiple filters and a wrapper method. Evaluations are conducted on four credit datasets. This chapter is organized as follows. Section (4.2) describes hybrid feature selection. Section (4.3) proposes a three-stage fusion combining both strategies. Then, our proposed method is compared with some existing selection methods in Section (4.4), and conclusions are drawn in Section (4.5).

4.2 Hybrid Framework

As discussed in Chapter 1 there are two main classes of feature selection methods: the filter and the wrapper. Both approaches have their merits and shortcomings and the superiority of one approach over the other is not settled. Rather than trying to optimize just one approach, it is better to integrate both in one compacted feature selection model. Several merging approaches can be used for feature selection (Wu et al. 2009; Mak and Kung 2008): a fusion of several filters or wrappers, wrappers and filters merged in a parallel way, or a sequential combination of filters and wrappers.

Filter and wrapper methods require various resources and lead to differed results. Combining both methods seems a natural choice to benefit from their advantages and avoid their shortcomings. Since both methods consider two different selection criteria, and we have no knowledge about the number of relevant features, a combination of both methods as a hybrid approach is proposed.

4.3 New Approach for hybrid feature selection

In order to improve the significance of selected features, we propose a three-stage approach as a combination of filter and wrapper methods. In the first stage we use a set of filter-based methods to classify candidate attributes based on their relevance level into three main categories. We obtain the following feature relevance categories: high, average and poor. Highly relevant features are kept as input to the second stage,

average ones are kept as input to the third stage and the last category is eliminated. In the second stage an efficient method dealing with both redundancy and relevance is considered. At this stage we minimize the redundancy among most relevant features while maximizing their relevance to the target class. To find the best combination between relevance and redundancy we formulate this problem as an optimization of a quadratic multi-objective function. Once the most relevant features are separated from redundant ones we move to stage three. The latter takes as input the selected features of Stage II and combines them with the average relevant features of Stage I. Then, a wrapper approach is trained on the resulting features. Figure (4.4) presents a flowchart of the different stages of the proposed approach.

4.3.1 Stage I: feature-based filtering

As discussed in Chapters 1 and 2, there are many ranking criteria for filters: MI, T-statistics, PCC, Relief, entropy and many others. Unfortunately, choosing the best one is a difficult task and depends on many factors such as the amount of available data, the data distribution and types of descriptive features among others. Rather than to optimize one single filter, we combine results of multiple filters in the pre-selection process. Many methods can be adopted to find the best combination. In this work, we fuse individual filters' output, i.e. final ranking of each filter, while assuming that the effect of each filter on the final decision is the same.

For the pre-filtering stage and for simplicity the number of used filters is fixed to three. Each filter ranks features according to their particular criterion, resulting into three different rankings. Then, the result of each filter is divided into three subsets of identical size according to their level of relevance to the target variable. Figure (4.1) shows the different relevance categories. Three groups of features are obtained for each filter: the highly significant, the average ones and those with poor relevance.

Once the relevance levels are identified for each filter, and as a first step the most relevant features are merged. They are produced by the three filters in one single group yielding a subset with the most relevant features. At a second step, the three

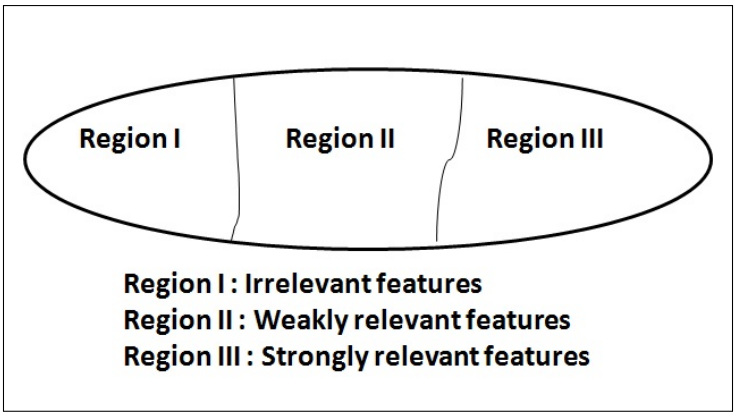


Figure 4.1: A view of feature relevance categories

groups of features with average relevance are grouped and redundant features, or the ones which appear in the first resulting subset, are also eliminated. Since the remaining features lack relevance and they are not adequate for our study they are eliminated. Figure (4.2) illustrates Stage I for an example of 22 features from the Tunisian dataset.

4.3.2 Stage II: reduction of redundant features using quadratic programming

Filter algorithms frequently do not consider interaction between features. Moreover, resulting ranking lists from Stage I may contain redundant information. Therefore, one common improvement direction for filter algorithms is to consider dependencies among variables, and an approach based on MI is proposed. This approach studies the redundancy among features starting from the highly ranked features selected in Stage I. The problem of feature redundancy is considered by statistical machine learning methods as well as mathematical ones. Mathematical programming based approaches have been proven to be successful in terms of classification accuracy for a wide range of applications. The proposed mathematical method is a quadratic programming formulation. Quadratic optimization process uses an objective function with quadratic and linear terms. Here, the quadratic term denotes the similarity among each pair of variables whereas the linear term captures the correlation between each feature and the target variable.

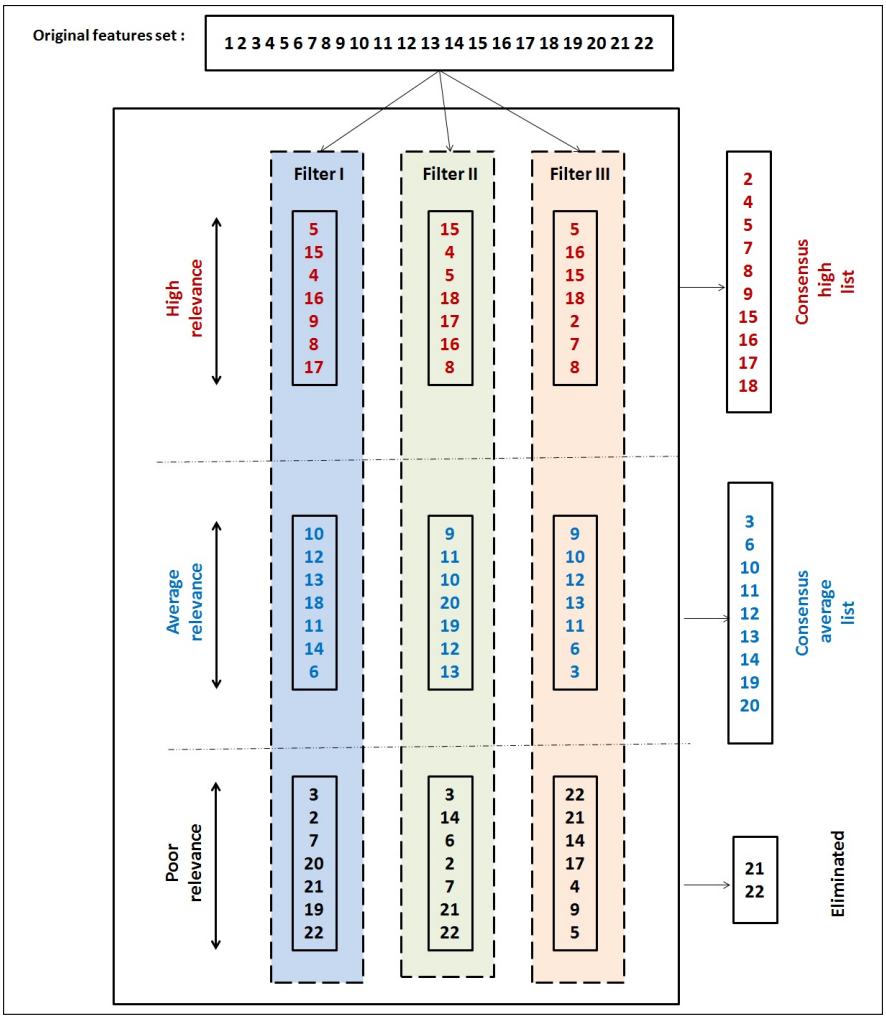


Figure 4.2: The proposed process of merging features selected by three filters in the fusion method

Let's assume that the classifier learning problem involves N training samples and d variables. A quadratic programming problem minimizes a multivariate quadratic function subject to linear constraints (Rodriguez et al. 2010) is given by:

$$\begin{aligned} &Min\ f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T Q\mathbf{w} - Z^T\mathbf{w}. \\ &\text{Subject to } w_i \geq 0 \text{ for all } i = 1, \ldots, d \\ &\text{and } \sum_{i=1}^{d} w_i = 1, \end{aligned} \tag{4.1}$$

where Z is a d-dimensional row vector with non-negative entries describing the coefficients of the linear terms in the objective function measuring how each feature is correlated with the target class (relevance), Q a $(d \times d)$ symmetric positive semi-definite matrix describing the coefficients of the quadratic terms representing the similarity among variables (redundancy) and the weights of variables are denoted by an d-dimensional column vector $\mathbf{w}$.

Bazaraa et al. (1993) and Rodriguez et al. (2010) showed that a feasible solution exists for this kind of problem and that the constraint region is bounded. When the objective function $f(\mathbf{w})$ is strictly convex for all feasible points the problem has a unique local minimum which is also the global minimum. The conditions for solving quadratic programming, including the Lagrangian function and the Karush-Kuhn-Tucker conditions are explained in details in Bazaraa et al. (1993).

Depending on the learning problem, the two conditions can have different relative purposes in the objective function. Therefore, a scalar parameter α is introduced as follows:

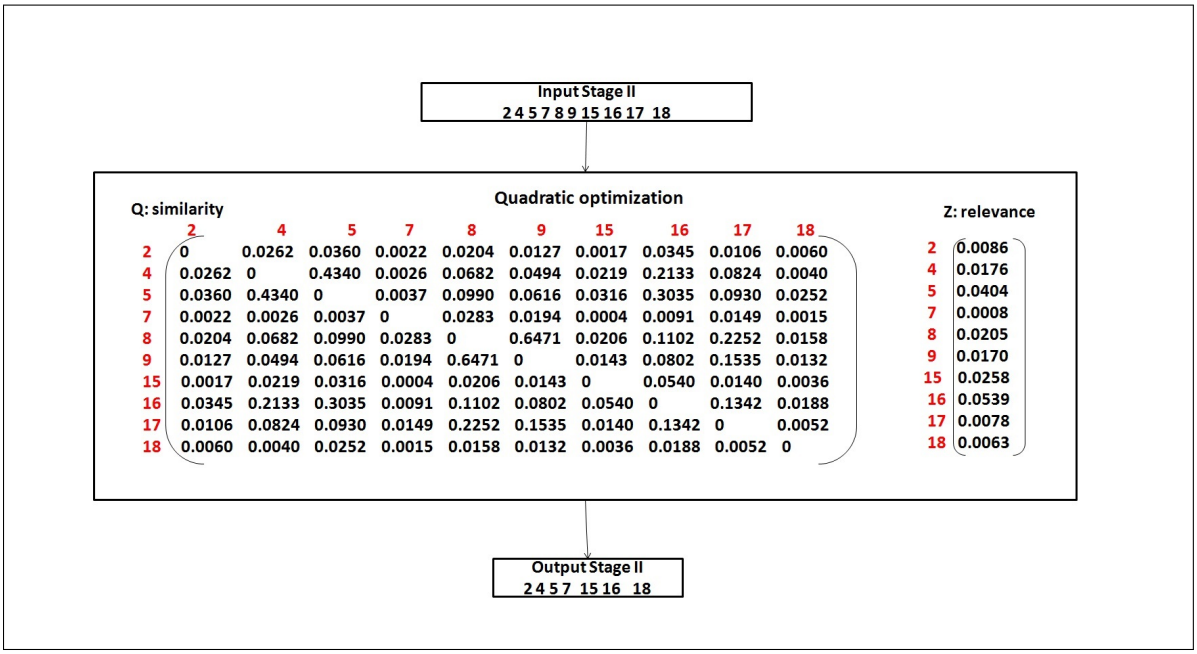
$$Min\ f(\mathbf{w}) = \frac{1}{2}(1-\alpha)\mathbf{w}^T Q\mathbf{w} - \alpha Z^T\mathbf{w}, \tag{4.2}$$

where $\mathbf{w}$, Q and Z are defined as before and $\alpha \in [0,1]$, where if $\alpha = 1$ only relevance is considered. On the opposite, if $\alpha = 0$ then only independence between features is considered. That is features with higher weights are those which have lower similarity coefficients with the remaining features. Every data set has its best choice of the scalar α. However, a reasonable choice of α should balance the relation between relevance and redundancy. Thus, a good estimation of α is needed. We know that the relevance and redundancy terms in Equation (4.2) are balanced when $(1-\alpha)\overline{Q} = \alpha\overline{Z}$, where $\overline{Q}$ is the estimate of the mean value of the matrix Q and $\overline{Z}$ is the estimate of the mean value of vector Z. Hence, we propose a practical estimate of α as follows:

$$\hat{\alpha} = \frac{\overline{Q}}{\overline{Q} + \overline{Z}}. \tag{4.3}$$

After solving the quadratic programming optimization the features with higher weights are considered to be better variables for subsequent classifier training. Figure

(4.3) illustrates Stage II.



Q: similarity	2	4	5	7	8	9	15	16	17	18	Z: relevance
2	0	0.0262	0.0360	0.0022	0.0204	0.0127	0.0017	0.0345	0.0106	0.0060	0.0086
4	0.0262	0	0.4340	0.0026	0.0682	0.0494	0.0219	0.2133	0.0824	0.0040	0.0176
5	0.0360	0.4340	0	0.0037	0.0990	0.0616	0.0316	0.3035	0.0930	0.0252	0.0404
7	0.0022	0.0026	0.0037	0	0.0283	0.0194	0.0004	0.0091	0.0149	0.0015	0.0008
8	0.0204	0.0682	0.0990	0.0283	0	0.6471	0.0206	0.1102	0.2252	0.0158	0.0205
9	0.0127	0.0494	0.0616	0.0194	0.6471	0	0.0143	0.0802	0.1535	0.0132	0.0170
15	0.0017	0.0219	0.0316	0.0004	0.0206	0.0143	0	0.0540	0.0140	0.0036	0.0258
16	0.0345	0.2133	0.3035	0.0091	0.1102	0.0802	0.0540	0	0.1342	0.0188	0.0539
17	0.0106	0.0824	0.0930	0.0149	0.2252	0.1535	0.0140	0.1342	0	0.0052	0.0078
18	0.0060	0.0040	0.0252	0.0015	0.0158	0.0132	0.0036	0.0188	0.0052	0	0.0063

Figure 4.3: Redundancy analysis for highly ranked features

At this stage, given its efficiency MI is chosen as a similarity measure. Hence, the quadratic term is $q_{jj'} = MI(X^J, X^{j'})$ and the linear one is $z_j = MI(X^J, Y)$. Using the quadratic approach based on MI provides a new ranking of the highly ranked feature selected in Stage I. This new ranking takes into account simultaneously the MI between all pairs of features and the relevance of each feature to the target one.

4.3.3 Stage III: feature-based wrapping

In Stages I and II we selected top-ranked features and removed redundant ones based on MI and quadratic programming. Many studies such as those conducted by Peng et al. (2005) showed that simply combining highly discriminant features often does not give a better feature set that yields the best classification performance. The reason behind this is that the feature set is not an inclusive representation of the characteristics of the target feature. Because features are selected according to their discriminative powers, they are not maximally representative of the original space covered by the entire dataset. The feature set may represent one or several dominant

characteristics of the target class, but these could still be small regions of the relevant space covering the target class. Thus, the generalization ability of the feature set could be limited.

Based on these facts, we propose to combine features selected in Stage II and those having average relevance in Stage I. This combination aims to expand the space covered by the feature set. The resulting feature set is the input to a wrapper algorithm. In wrapper based methods, feature selection is powered by the learning method, and features' relevance is evaluated by the given accuracy of the classification method. Generally, we obtain a set with a very small number of non-redundant features giving a high accuracy, since the characteristics of the features match very well with the characteristics of the learning method. Figure (4.4) illustrates Stage III.

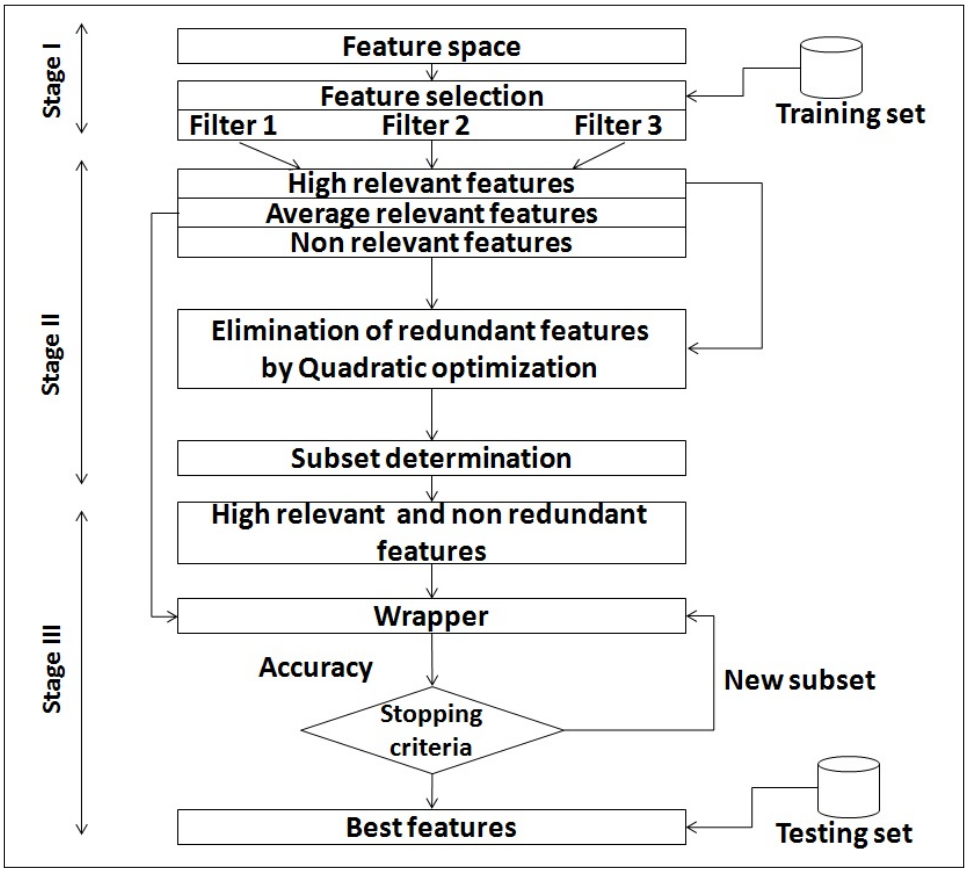


Figure 4.4: Flowchart of the proposed three-stage feature selection fusion

4.4 Experimental investigations

In Stage I, we used three filters to rank features according to their level of significance. Matlab used in this step provides a whole set of tools for selecting diversity and discriminating features. We used the "Rankfeatures" function as a simple way to rank features using an independent evaluation criterion for binary classification. To assess the significance of every feature in separating two labeled groups many criterions could be used such as: χ^2, Relief, MI, relative entropy, and others. For simplicity, the three first criterions are selected for our task. Numbers of selected features in Stage I are given in Table (4.1).

In Stage II, the redundancy is reduced using the quadratic optimization and MI as a similarity measure. This stage is implemented in R software using the "Quadprog" package (Goldfarb and Idnani 1983), where results are obtained with $\alpha = 0.501$ for the Australian dataset, $\alpha = 0.511$ for the German dataset, $\alpha = 0.509$ for the HMEQ and $\alpha = 0.514$ for the Tunisian dataset. This means that the best value of α is obtained when there is an equal tradeoff between relevance and redundancy. The MI for all features is measured using the function "Mutualinfo" in Matlab. Stage III takes as output the features selected in Stage II and those classified as average significance features. Then, a Bi-Directional wrapper is used.

Table 4.1: Number of remaining features after Stage I

High	Average	Poor	Retained
		Australian	
7	3	4	10
		German	
7	9	4	16
		HMEQ	
5	5	2	10
		Tunisian	
10	9	2	19

The following filter and wrapper methods have been considered in the described experiments for the comparisons:

- Maximal Relevance (MaxRel) feature selection selects those features that have

highest relevance to the target class (Peng et al. 2005).

- minimal-Redundancy-Maximum-Relevance (mRMR) algorithm chooses a subset of features with both minimum redundancy and maximum relevance . The mRMR algorithm selects features greedily, minimizing their redundancy with features chosen in previous steps and maximizing their relevance to the class (Ding and Peng 2005).

- Relief, χ^2 and MI are given in Chapter 1.

- Kulback-leibler is used as ranking criteria. Features with the maximum Kullback-Leibler distance are selected as the most significant features.

- Forward, backward, bi-Directional wrapper details are given in Chapter 1.

4.4.1 Results and discussion

Results for the three credit datasets using the previously quoted feature selection methods are summarized in Tables (4.3)-(4.5). Classification results represent the performance of each feature selection method for four different classification methods: DT, SVM, ANN and KNN, where the best results are shown in bold.

Performance of filters and wrappers

Tables (4.3)-(4.5) show the performances achieved by ANN, KNN, SVM and DT using six filters and three wrappers. Both filters and wrappers perform well as feature selectors for the scoring task. They may not always give the best set of features for the classification algorithm but in most cases they do. There is obviously a strong similarity in the feature sets selected by different approaches. A more detailed picture of the achieved results shows that the precision of wrappers is better than some of the studied filters. These results are confirmed by the AUC rate for the three datasets, proving the superiority of wrappers in terms of precision. In some cases using wrappers is advantageous since they are able to achieve the same performance

as filters with a more reduced subset. In other cases, filters do a better job with a significant lower complexity than wrapper even by using limited information.

Performance of the New Approach

We notice from Tables (4.3)-(4.6) that in most cases the new approach achieves the best rate of AUC. The higher is the value of AUC the better is the distinguishing capacity of the classifier. This means that the chosen features set by the new approach provides the best combination of features given that it improves the capability of a credit model to correctly identify the behavior of an applicant to pay back a loan.

Table 4.2: Summary of the best performance results archived by the set of feature selection methods for the four datasets within the hybrid framework.

	DT	**SVM**	**ANN**	**KNN**
MaxRel Features	⊙		⊗	
mRMR Features			⊖ ⊖	
χ^2		⊙		
Kullback-leibler		⊖ ⊖		
Relief				
MI	⊙	⊕	⊕	
Wrapper Bi-Directional				
Wrapper Forward				
Wrapper Backwards				
Three-Stage Approach	⊖⊗⊕ ⊖⊗⊕ ⊖⊗ ⊕⊙ ⊗⊗⊕⊙	⊗⊙ ⊖⊗⊕ ⊖⊗⊕⊙ ⊖⊗⊕⊙	⊗⊕⊙ ⊖⊙ ⊗⊕⊙ ⊗⊕⊙	⊖⊗⊕⊙ ⊖⊗⊕⊙ ⊖⊗⊕⊙ ⊖⊗⊕⊙

⊖ Precision, ⊗ Recall, ⊕ F-measure, ⊙ ROC Area.
Color: - Australian, - German , - HEMQ, - Tunisian.

From Table (4.2) we see that for the ANN and KNN classifiers the three stage approach always achieves the highest rate of area under the ROC curve. This was not the case for SVM where the proposed approach achieves three times the highest area under the ROC curve and for DT where the proposed approach achieves twice the highest area under the ROC curve. We also notice from Table (4.2) that overall our proposed approach achieves 15 times the best recall, 13 times the best ROC area and 14 times the best F-measure and 11 times the highest precision.

Consistent with the theoretical analysis for feature selection, the fusion approach usually outperforms single wrappers or filters.

Table 4.3: Classification results for the three stage feature selection for the Australian dataset.

	Precision	Recall	F-Measure	ROC Area
		Decision Tree		
MaxRel Features	0.603	0.672	0.636	0.812
mRMR Features	0.557	0.589	0.573	0.797
χ^2	0.603	0.673	0.640	0.916
Kullback-leibler	0.721	0.661	0.690	0.819
Relief	0.586	0.560	0.546	0.799
MI	0.601	0.600	0.600	**0.837**
Wrapper Bi-Directional	0.739	0.771	0.750	0.798
Wrapper Forward	0.737	0.775	0.755	0.788
Wrapper Backwards	0.749	0.769	0.749	0.796
Three-Stage Approach	**0.857**	**0.889**	**0.873**	0.797
		Support Vector Machine		
MaxRel Features	0.839	0.871	0.854	0.792
mRMR Features	0.902	0.852	0.876	0.813
χ^2	0.629	0.670	0.655	0.818
Kullback-leibler	**0.931**	0.861	0.894	0.820
Relief	0.795	0.898	0.843	0.803
MI	0.930	0.870	**0.900**	0.817
Wrapper Bi-Directional	0.912	0.845	0.876	0.807
Wrapper Forward	0.919	0.840	0.879	0.810
Wrapper Backwards	0.915	0.843	0.878	0.808
Three-Stage Approach	0.900	**0.901**	0.880	**0.823**
		Artificial Neural Network		
MaxRel Features	0.892	0.917	0.904	0.855
mRMR Features	**0.931**	0.880	0.905	0.847
χ^2	0.790	0.720	0.700	0.950
Kullback-leibler	0.931	0.870	0.900	0.826
Relief	0.685	0.626	0.605	0.845
MI	0.729	0.673	0.702	0.844
Wrapper Bi-Directional	0.896	0.898	0.898	0.843
Wrapper Forward	0.899	0.892	0.897	0.845
Wrapper Backwards	0.900	0.889	0.898	0.842
Three-Stage Approach	0.895	**0.944**	**0.919**	**0.856**
		K-Nearest Neighbor		
MaxRel Features	0.782	0.843	0.815	0.757
mRMR Features	0.795	0.844	0.819	0.756
χ^2	0.687	0.739	0.715	0.755
Kullback-leibler	0.831	0.770	0.800	0.724
Relief	0.674	0.726	0.701	0.750
MI	0.789	**0.972**	0.871	0.742
Wrapper Bi-Directional	0.893	0.924	0.903	0.754
Wrapper Forward	0.790	0.826	0.809	0.752
Wrapper Backwards	0.797	0.820	0.800	0.750
Three-Stage Approach	**0.897**	0.944	**0.919**	**0.758**

Table 4.4: Classification results for the three stage feature selection for the German dataset.

	Precision	Recall	F-Measure	ROC Area
		Decision Tree		
MaxRel Features	0.620	0.632	0.620	**0.727**
mRMR Features	0.496	0.509	0.502	0.562
χ^2	0.594	0.532	0.561	0.681
Kullback-leibler	0.624	0.589	0.606	0.710
Relief	0.582	0.555	0.568	0.691
MI	0.616	0.634	0.625	0.725
Wrapper Bi-Directional	0.776	0.751	0.782	0.705
Wrapper Forward	0.773	0.789	0.780	0.700
Wrapper Backwards	0.770	0.787	0.786	0.703
Three-Stage Approach	**0.878**	**0.890**	**0.882**	0.723
		Support Vector Machine		
MaxRel Features	0.506	0.565	0.583	0.713
mRMR Features	0.527	0.498	0.512	0.693
χ^2	0.649	0.543	0.591	**0.718**
Kullback-leibler	0.667	0.532	0.590	0.708
Relief	0.513	0.532	0.522	0.679
MI	0.606	0.566	0.585	0.710
Wrapper Bi-Directional	0.858	0.693	0.760	0.677
Wrapper Forward	0.853	0.695	0.759	0.673
Wrapper Backwards	0.856	0.690	0.758	0.679
Three-Stage Approach	**0.956**	**0.805**	**0.859**	0.677
		Artificial Neural Network		
MaxRel Features	0.720	**0.634**	0.670	0.767
mRMR Features	0.697	0.555	0.616	0.742
χ^2	0.729	0.600	0.657	0.749
Kullback-leibler	0.736	0.577	0.645	0.757
Relief	0.656	0.611	0.633	0.749
MI	0.712	0.634	**0.672**	0.767
Wrapper Bi-Directional	0.681	0.589	0.631	0.727
Wrapper Forward	0.683	0.583	0.635	0.729
Wrapper Backwards	0.680	0.585	0.629	0.720
Three-Stage Approach	**0.781**	0.589	0.651	**0.769**
		K-Nearest Neighbor		
MaxRel Features	0.703	0.630	**0.668**	0.775
mRMR Features	0.684	0.611	0.645	0.754
χ^2	**0.718**	0.634	0.673	0.750
Kullback-leibler	0.716	0.634	0.670	0.762
Relief	0.617	0.611	0.614	0.759
MI	0.703	0.634	0.666	0.775
Wrapper Bi-Directional	0.651	0.577	0.616	0.761
Wrapper Forward	0.653	0.552	0.612	0.760
Wrapper Backwards	0.650	0.570	0.618	0.756
Three-Stage Approach	0.663	**0.677**	0.619	**0.776**

To be more precise if we look into the results of Table (4.3) we notice that with the Australian datasets the proposed approach achieves the highest recall with DT, SVM

and ANN classifiers and the best F-measure with DT, ANN and KNN classifiers.

Table 4.5: Classification results for the three stage feature selection for the HMEQ dataset.

	Precision	Recall	F-Measure	ROC Area
		Decision Tree		
MaxRel Features	0.808	0.558	0.660	0.725
mRMR Features	0.730	0.558	0.632	0.713
χ^2	0.782	0.598	0.677	0.757
Kullback-leibler	0.732	0.552	0.629	0.703
Relief	0.590	0.542	0.565	0.753
MI	0.790	0.598	0.680	0.757
Wrapper Bi-Directional	0.750	0.570	0.647	0.762
Wrapper Forward	0.748	0.564	0.627	0.767
Wrapper Backwards	0.742	0.568	0.643	0.760
Three-Stage Approach	**0.788**	**0.794**	**0.791**	**0.888**
		Support Vector Machine		
MaxRel Features	0.806	0.607	0.692	0.723
mRMR Features	0.843	0.530	0.650	0.707
χ^2	0.813	0.579	0.616	0.752
Kullback-leibler	**0.848**	0.540	0.659	0.710
Relief	0.563	0.593	0.577	0.715
MI	0.823	0.577	0.678	0.737
Wrapper Bi-Directional	0.730	0.573	0.642	0.745
Wrapper Forward	0.739	0.570	0.643	0.755
Wrapper Backwards	0.735	0.579	0.647	0.759
Three-Stage Approach	0.721	**0.846**	**0.778**	**0.839**
		Artificial Neural Network		
MaxRel Features	0.691	0.561	0.619	0.720
mRMR Features	**0.740**	0.556	0.634	0.710
χ^2	0.681	0.588	0.631	0.751
Kullback-leibler	0.737	0.558	0.635	0.707
Relief	0.563	0.515	0.537	0.653
MI	0.681	0.588	0.631	0.751
Wrapper Bi-Directional	0.670	0.589	0.626	0.750
Wrapper Forward	0.676	0.588	0.628	0.753
Wrapper Backwards	0.674	0.600	0.636	0.751
Three-Stage Approach	0.614	**0.722**	**0.663**	**0.792**
		K-Nearest Neighbor		
MaxRel Features	0.688	0.564	0.619	0.730
mRMR Features	**0.694**	0.564	0.622	0.707
χ^2	0.671	0.701	0.685	0.747
Kullback-leibler	0.698	0.563	0.623	0.710
Relief	0.538	0.515	0.529	0.655
MI	0.675	0.599	0.634	0.750
Wrapper Bi-Directional	0.675	0.585	0.626	0.752
Wrapper Forward	0.672	0.588	0.627	0.755
Wrapper Backwards	0.671	0.583	0.623	0.753
Three-Stage Approach	0.674	**0.704**	**0.688**	**0.774**

Table 4.6: Classification results for the three stage feature selection for the Tunisian dataset.

	Precision	Recall	F-Measure	ROC Area
		Decision Tree		
MaxRel Features	0.611	0.614	0.612	0.700
mRMR Features	0.620	0.623	0.628	0.701
χ^2	0.610	0.615	0.613	0.702
Kullback-leibler	0.796	0.800	0.798	0.688
Relief	0.501	0.502	0.501	0.690
MI	0.590	0.620	0.604	0.679
Wrapper Bi-Directional	0.730	0.790	0.759	0.703
Wrapper Forward	0.706	0.722	0.713	0.702
Wrapper Backwards	0.689	0.736	0.702	0.678
Three-Stage Approach	**0.862**	**0.960**	**0.908**	**0.716**
		Support Vector Machine		
MaxRel Features	0.707	0.700	0.733	0.725
mRMR Features	0.805	0.787	0.795	0.700
χ^2	0.650	0.742	0.693	0.670
Kullback-leibler	0.744	0.853	0.794	0.673
Relief	0,522	0.650	0.581	0.670
MI	0,604	0,647	0,608	0.708
Wrapper Bi-Directional	0.711	0.750	0.710	0.706
Wrapper Forward	0.706	0.722	0.713	0.710
Wrapper Backwards	0.774	0.846	0.786	0.680
Three-Stage Approach	**0.852**	**0.990**	**0.916**	**0.752**
		Artificial Neural Network		
MaxRel Features	0.812	0.826	0.818	0.712
mRMR Features	0.820	0.830	0.825	0.715
χ^2	0.775	0.790	0.782	0.650
Kullback-leibler	0.805	0.805	0.805	0.699
Relief	0.788	0.870	0.827	0.702
MI	0.816	0.820	0.818	0.687
Wrapper Bi-Directional	0.889	0.856	0.872	0.713
Wrapper Forward	0.809	0.850	0.806	0.690
Wrapper Backwards	0.815	0.852	0.811	0.703
Three-Stage Approach	**0.864**	**0.973**	**0.915**	**0.730**
		K-Nearest Neighbor		
MaxRel Features	0.818	0.850	0.827	0.706
mRMR Features	0.795	0.846	0.807	0.710
χ^2	0.766	0.789	0.777	0.623
Kullback-leibler	0.754	0.800	0.776	0.641
Relief	0.700	0.802	0.747	0.690
MI	0.722	0.850	0.781	0.650
Wrapper Bi-Directional	0.818	0.854	0.813	0.700
Wrapper Forward	0.789	0.840	0.800	0.686
Wrapper Backwards	0.800	0.848	0.802	0.680
Three-Stage Approach	**0.859**	**0.981**	**0.916**	**0.723**

Table (4.4) shows that for German dataset the new approach achieves the highest recall with DT, SVM and KNN classifiers, the best precision with DT, SVM and

ANN and the highest F-measure with DT and SVM and produces the best AUC for ANN and KNN witch means that the selected features by this approach allow finding the best middle ground between specificity and sensitivity.

We notice from Tables (4.5)-(4.6) that the new approach achieves the best precision, recall, F-measure and ROC area with DT and SVM classifiers with both the HMEQ and Tunisian datasets. Table (4.5) shows that for the HMEQ dataset the new approach achieves the best recall and F-measure and AUC with all datasets.

A two-way ANOVA is performed on the F-measure results in order to test the difference between the features selection methods and classification methods. The first factor is represented through the different feature selection methods, where $\{Relief, MI, MaxRel, mRMR, \chi^2, Kullback, Bi-Directional, Forward\,, Backward, ThreeStage\}$ present the levels of the first factor. The second factor is represented by the different classification methods including $\{DT, SVM, ANN, KNN\}$. Several hypotheses are jointly tested in a two-way ANOVA. H_0 and alternative hypothesis H_1 for the first factor presenting all feature selection methods would be

$$\begin{cases} H_0 : \mu^1_{Relief} = \mu^1_{MI} = \mu^1_{MaxRel} = \mu^1_{mRMR} = \mu^1_{\chi^2} = \mu^1_{Kullback} = \mu^1_{Directional} = \mu^1_{Forward} \\ = \mu^1_{Backward} = \mu^1_{ThreeStage} \text{ Performances of selection methods are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the feature selection methods mean performance is different} \\ \text{from the others} \end{cases}$$

For the second factor, i.e. Classifier, H_0 and H_1 are given by:

$$\begin{cases} H_0 : \mu^2_{DT} = \mu^2_{SVM} = \mu^2_{ANN} = \mu^2_{KNN} \text{ Performances of classifiers are equal,} \\ \text{versus} \\ H_1 : \text{At least one of the classifer mean performance is different from the others} \end{cases}$$

Interaction between the two factors:

$$\begin{cases} H_0 : \text{There is no interaction between the two factors,} \\ \text{versus} \\ H_1 : \text{There is an interaction between the two factors} \end{cases}$$

To set up a two-way ANOVA we use the data in Table (4.7), the obtained results are summarized in Table (4.8)

Table 4.7: Summary of F-measures for all feature selection methods with the four classification methods in hybrid framework.

	χ^2	**Relief**	**MI**	**MaxRel**	**mRMR**
DT	0.640	0.546	0.600	0.636	0.573
	0.561	0.568	0.626	0.620	0.502
	0.677	0.565	0.680	0.660	0.632
	0.613	0.501	0.604	0.612	0.628
	Kullback	**Directional**	**Forward**	**Backward**	**Three Stage**
	0.690	0.750	0.755	0.749	0.873
	0.606	0.782	0.780	0.786	0.882
	0.629	0.647	0.627	0.643	0.791
	0.698	0.759	0.713	0.702	0.908
	χ^2	**Relief**	**MI**	**MaxRel**	**mRMR**
SVM	0.655	0.843	0.900	0.854	0.876
	0.591	0.522	0.585	0.583	0.512
	0.616	0.577	0.678	0.692	0.650
	0.693	0.581	0.608	0.733	0.795
	Kullback	**Directional**	**Forward**	**Backward**	**Three Stage**
	0.894	0.876	0.879	0.878	0.880
	0.590	0.760	0.759	0.758	0.859
	0.659	0.642	0.643	0.647	0.778
	0.794	0.710	0.713	0.786	0.916
	χ^2	**Relief**	**MI**	**MaxRel**	**mRMR**
ANN	0.700	0.605	0.702	0.904	0.905
	0.657	0.633	0.672	0.670	0.616
	0.631	0.537	0.631	0.619	0.634
	0.782	0.827	0.818	0.818	0.825
	Kullback	**Directional**	**Forward**	**Backward**	**Three Stage**
	0.900	0.898	0.897	0.898	0.919
	0.645	0.631	0.635	0.629	0.651
	0.635	0.626	0.628	0.636	0.663
	0.805	0.872	0.806	0.811	0.915
	χ^2	**Relief**	**MI**	**MaxRel**	**mRMR**
KNN	0.715	0.701	0.871	0.815	0.819
	0.673	0.614	0.666	0.668	0.645
	0.685	0.529	0.634	0.619	0.622
	0.777	0.747	0.781	0.827	0.807
	Kullback	**Directional**	**Forward**	**Backward**	**Three Stage**
	0.800	0.903	0.809	0.800	0.919
	0.670	0.616	0.612	0.618	0.619
	0.623	0.626	0.627	0.623	0.688
	0.776	0.813	0.800	0.802	0.916

The items of primary interest in Table (4.8) are the effects listed under the "Source" column and the values under the "Sig." column. As in the previous hypothesis tests, if the p-value is less than 0.05, as set by the experimenter, then that

Table 4.8: Tests of between-subjects effects in hybrid framework.

Source	**Type III Sum of Squares**	**DF**	**Mean Square**	**F**	**Sig. (p-value)**
Corrected Model	0.646	39	0.017	1.518	0.045
Intercept	81.140	1	81.140	7432.361	0.000
Selection Method	0.418	**9**	0.046	4.253	**8.37151E-05**
Classifier	0.095	**3**	0.032	2.913	**0.037**
Selection Method * Classifier	0.133	**27**	0.005	0.451	**0.991**
Error	1.310	120	0.011		
Total	83.096	160			
Corrected Total	1.956	159			

Dependant Variable : F-measure

effect is significant. From Table (4.8) we notice that we don't have a statistically significant interaction between the factor selection method and the factor classifier, but there are statistically significant differences between classifier levels and selection method levels where p-value is less than 0.05 for both factors. A more detailed picture is given in Tables (4.9) and (4.10).

Table 4.9: Multiple comparisons table for different classifiers in hybrid framework.

Classifier (I)	**Classifier (J)**	**Mean difference (I-J)**	**Sig.**
ANN	DT	0.06180*	0.045
	KNN	0.01028	0.971
	SVM	0.00803	0.986
DT	ANN	-0.06180*	0.045
	KNN	-0.05153	0.128
	SVM	-0.05378	0.103
KNN	ANN	-0.01028	0.971
	DT	0.05153	0.128
	SVM	-0.00225	1.000
SVM	ANN	-0.00803	0.986
	DT	0.05378	0.103
	KNN	0.00225	1.000

Table (4.9) shows that there is a significant difference between the results produced by DT and ANN classifiers where the computed p-value is 0.045.

we notice from Table (4.10) that there is a statistically significant difference between the obtained results from (1) the three stage approach and MI where the p-value = 0.017, (2) the three stage approach and mRMR where the p-value = 0.015, (3) the three stage approach and χ^2 where the p-value = 0.002 and (4) the three stage approach and relief where the p-value < 0.05.

Table 4.10: Multiple comparisons table for feature selection methods in hybrid framework.

Selection method (I)	**Selection method (J)**	**Mean difference (I-J)**	**Sig.**	**Selection method (I)**	**Selection method (J)**	**Mean difference (I-J)**	**Sig.**
Backward	Directional	-0.00906	1	MaxRel	Backward	-0.02725	0.999
	χ^2	0.06875	0.695		Directional	-0.03631	0.993
	Forward	0.00519	1		χ^2	0.0415	0.981
	Kullback	0.022	1		Forward	-0.02206	1
	MaxRel	0.02725	0.999		Kullback	-0.00525	1
	MI	0.04438	0.971		MI	0.01713	1
	mRMR	0.04531	0.967		mRMR	0.01806	1
	Relief	0.11687	0.059		Relief	0.08962	0.32
	Three stage	-0.08819	0.343		Three stage	-0.11544	0.066
Directional	Backward	0.00906	1	MI	Backward	-0.04438	0.971
	χ^2	0.07781	0.527		Directional	-0.05344	0.91
	Forward	0.01425	1		χ^2	0.02438	1
	Kullback	0.03106	0.998		Forward	-0.03919	0.987
	MaxRel	0.03631	0.993		Kullback	-0.02237	1
	MI	0.05344	0.91		MaxRel	-0.01713	1
	mRMR	0.05437	0.9		mRMR	0.00094	1
	Relief	.12594*	0.029		Relief	0.0725	0.627
	Three stage	-0.07913	0.502		Three stage	-.13256*	0.017
χ^2	Backward	-0.06875	0.695	mRMR	Backward	-0.04531	0.967
	Directional	-0.07781	0.527		Directional	-0.05437	0.9
	Forward	-0.06356	0.782		χ^2	0.02344	1
	kullback	-0.04675	0.959		Forward	-0.04012	0.985
	MaxRel	-0.0415	0.981		Kullback	-0.02331	1
	MI	-0.02438	1		MaxRel	-0.01806	1
	mRMR	-0.02344	1		MI	-0.00094	1
	Relief	0.04812	0.951		Relief	0.07156	0.644
	Three stage	-.15694*	0.002		Three stage	-0.13350*	0.015
Forward	Backward	-0.00519	1	Relief	Backward	-0.11687	0.059
	Directional	-0.01425	1		Directional	-0.12594*	0.029
	χ^2	0.06356	0.782		χ^2	-0.04812	0.951
	Kullback	0.01681	1		Forward	-0.11169	0.086
	MaxRel	0.02206	1		Kullback	-0.09487	0.244
	MI	0.03919	0.987		MaxRel	-0.08962	0.32
	mRMR	0.04012	0.985		MI	-0.0725	0.627
	Relief	0.11169	0.086		mRMR	-0.07156	0.644
	Three stage	-0.09338	0.265		Three stage	-.20506*	0
Kullback	backward	-0.022	1	Three stage	Backward	0.08819	0.343
	Directional	-0.03106	0.998		Directional	0.07913	0.502
	χ^2	0.04675	0.959		χ^2	0.15694*	0.002
	Forward	-0.01681	1		Forward	0.09338	0.265
	MaxRel	0.00525	1		Kullback	0.11019	0.095
	MI	0.02237	1		MaxRel	0.11544	0.066
	mRMR	0.02331	1		MI	0.13256*	0.017
	Relief	0.09487	0.244		mRMR	0.13350*	0.015
	Three stage	-0.11019	0.095		Relief	0.20506*	0

4.5 Conclusion

Feature selection is an important task in CS. We propose in this chapter fusing in a first stage a set of filters methods as a pre-selection step. The first stage is followed by a filter selection based on a quadratic optimization and a similarity study. Finally, the fusion is refined by a wrapper selection. Results show that the fusion performance is either superior to or at least as good as either filter or wrapper methods.

Conclusion

Credit-risk evaluation involves processing huge volumes of data. Consequently it requires powerful data mining tools. Several methods developed in machine learning have been used for financial credit-risk evaluation and especially for CS. However, the majority of these tools are affected by the curse of dimensionality and irrelevant features. These facts often degrade the performance of predictive models both in speed and in predictive accuracy. Hence, the use of optimal feature subset becomes essential.

In this thesis, we review the framework of feature selection and disuss the basic concepts of different feature selection models: filter, wrapper and hybrid. Some research questions related to each one of these three categories are examined.

In Chapter 2 we investigate filter feature selection. We present a brief reminder of the filter framework and two major issues when dealing with filtering methods, respectively the selection trouble and the issue of disjoint ranking for similar features. Then, a new approach is introduced with experimental investigations in Chapter 2 based on three steps: in the first we present the feature selection problem as an optimization problem with the aim of finding the best list, which would be the closest possible to all individual ordered lists. Then, in the next step we presented a solution to the optimization problem. The solution consists on using GAs. In the final step we used similarity in order to resolve the problem of disjoint ranking for similar features. The results for this chapter were evaluated on four credit datasets. We compare the new approach with some well known aggregation methods and some individual filtering methods. Results show that ensemble methods improve precision, recall and

F-measure, especially when the similarity is considered.

The proposed approach in Chapter 3 is an ensemble method based on wrapper feature selection which is a complete search and a multiple classifiers system. We first focus on the search strategy and the choice of the starting point and we propose reducing the search space to a manageable size based on similarity study with prior knowledge. Then, a hybrid search strategy mixing heuristic and complete search is performed.

The last part of Chapter 3 is dedicated to the evaluation process. In this step two different classifier arrangement approaches are used within the wrapper evaluation process, namely the same-type approach and the mixed-type approach. By using the second arrangement approach we need to investigate how classifiers from different families work together and how their interaction influences the feature selection. Then, by using both approaches we obtained a complete picture of the influences of the nature of classifiers on feature selection. Results show that the use of prior information and heuristics in the complete search induces a significant gain in complexity with improved generalization. Furthermore, we show that the number of classifiers and their nature have an important effect on wrapper feature selection.

The final contribution of this thesis is in Chapter 4, where a three-stage feature selection fusion using quadratic programming is proposed. In the first stage we used a set of filter-based methods to classify candidate attributes based on their relevance level into three main categories, to get the following feature relevance categories: high, average and poor. Highly relevant features are kept as input to the second stage, average ones are used as input to the third stage and the last category is eliminated.

In the second stage an efficient method dealing with both redundancy and relevance is considered. In this stage we minimize the redundancy among the most relevant features while maximizing their relevance to the target class. To find the best combination between relevance and redundancy we formulate this problem as an optimization of a quadratic multi-objective function. Once the most relevant features are

separated from the redundant ones we move to stage three. The latter took as input the selected features of stage two and combine them with the average relevant features of the first stage, then a wrapper approach is trained on the resulting features. Results show that the fusion performance is either superior to or at least as good as either filter or wrapper methods.

The used datasets were relatively small in size, i.e., less than 100 features. In fact, the considered datasets in the evaluation have a maximum of 23 features (i.e. Tunisian dataset). It would be of interest to test our three proposed methods on large datasets to provide more insight into their performance.

Bibliography

Al-Ani, A. and M. Deriche (2001). An optimal feature selection technique using the concept of mutual information. In *Proceedings of the Sixth International Symposium on Signal Processing and its Applications*, pp. 477–480.

Arauzo-Azofra, A., J. M. Benitez, and J. L. Castro (2008). Consistency measures for feature selection. *Journal of Intelligent Information Systems 30*(3), 273–292.

Bardos, M. (2001). *Analyse discriminante: Application au risque et scoring financier*. Dunod.

Bazaraa, M., H. Sherali, and C. Shetty (1993). *Nonlinear Programming Theory and Algorithms*. New York: John Wiley.

Bellotti, T. and J. Crook (2009). Support vector machines for credit scoring and discovery of significant features. *Expert Systems with Applications 36*(2), 3302–3308.

Blum, A. L. and P. Langley (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence 97*(2), 245–271.

Bonev, B. (2010). *Feature Selection based on Information Theory*. Ph. D. thesis, University of Alicante.

Bouckaert, R. R., E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald, and D. Scuse (2009). *Weka manual (3.7.1)*.

Breiman, L., J. Friedman, R. Olshen, and C. Stone (1984). *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks.

Burges, J. (1998). A tutorial on support vector machines for pattern recognition. *data mining knowledge discovery 2*(2), 121–167.

Carterette, B. (2009). On rank correlation and the distance between rankings. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, New York, NY, USA, pp. 436–443. ACM.

Chan, Y. H., W. Y. N. Wing, S. Y. Daniel, and P. P. K. Chan (2010). Empirical comparison of forward and backward search strategies in l-gem based feature selection with rbfnn. In *Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC)*, pp. 1524–1527.

Chen, F. L. and F. C. Li (2010). Combination of feature selection approaches with svm in credit scoring. *Expert Systems with Applications 37*(7), 4902–4909.

Cho, S., H. Hong, and B. C. Ha (2010). A hybrid approach based on the combination of variable selection using decision trees and case-based reasoning using the mahalanobis distance: For bankruptcy prediction. *Expert Systems with Applications 37*(4), 3482–3488.

Chrysostomou, K., S. Y. Chen, and X. Liu (2008). Combining multiple classifiers for wrapper feature selection. *International Journal of Data Mining, Modelling and Management 1*(1), 91–102.

Chrysostomou, K. A. (2008). *The Role of Classifiers in Feature Selection: Number vs Nature*. Ph. D. thesis, School of Information Systems, Computing and Mathematics. Brunel University.

Clegg, J., J. F. Dawson, S. J. Porter, and M. H. Barley (2009). A genetic algorithm for solving combinatorial problems and the effects of experimental error - applied to optimizing catalytic materials. *QSAR & Combinatorial Science 28*(9), 1010–1020.

Dash, M. and H. Liu (2003). Consistency-based search in feature selection. *Artificial Intelligence 151*(1-2), 155–176.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, London, UK, pp. 1–15. Springer-Verlag.

Ding, C. and H. Peng (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology 3*(2), 185–206.

Dinu, L. P. and F. Manea (2006). An efficient approach for the rank aggregation problem. *Theoretical Computer Science 359*(1), 455–461.

Dittman, D. J., T. M. Khoshgoftaar, R. Wald, and A. Napolitano (2013). Classification performance of rank aggregation techniques for ensemble gene selection. In C. Boonthum-Denecke and G. M. Youngblood (Eds.), *Proceedings of the International Conference of the Florida Artificial Intelligence Research Society (FLAIRS)*. AAAI Press.

Durand, D. (1941). *Risk elements in consumer instalment financing.* National bureau of economic research [New York].

Falangis, K. and J. Glen (2010). Heuristics for feature selection in mathematical programming discriminant analysis models. *Journal of the Operational Research Society 61*(5), 804–812.

Fernandez, G. (2010). *Statistical Data Mining Using SAS Applications*. Chapman & Hall/Crc: Data Mining and Knowledge Discovery. Taylor and Francis.

Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics 7*(2), 179–188.

Forman, G. (2008). Bns feature scaling: an improved representation over tf-idf for svm text classification. In *Proceedings of the 17th ACM conference on Information and knowledge mining*, New York, NY, USA, pp. 263–270. ACM.

Frydman, H., E. I. Altman, and D. L. Kao (1985). Introducing recursive partitioning for financial classification: The case of financial distress. *Journal of Finance 40*(1), 269–91.

Giudici, P. (2003). *Applied Data Mining: Statistical Methods for Business and Industry*. West Sussex PO19 8SQ, England: John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester.

Goldfarb, D. and A. Idnani (1983). A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming 27*(1), 1–33.

Gotshall, S. and B. Rylander (2000). Optimal population size and the genetic algorithm.

Guyon, I. and A. Elisseeff (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research 3*(9), 1157–1182.

Haizhou, W. and L. Jianwu (2011). Credit scoring based on eigencredits and svdd. In *Proceedings of the International Conference on Applied Informatics and Communication*, pp. 32–40.

Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the International Conference on Machine Learning(ICML)*, pp. 359–366.

Hand, D. J. and W. E. Henley (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society Series A 160*(3), 523–541.

Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning*. Springer series in statistics. Springer New York Inc.

Holland, J. H. (1992). *Adaptation in natural and artificial systems*. Cambridge, MA, USA: MIT Press.

Howley, T., M. G. Madden, M. L. O'Connell, and A. G. Ryder (2006). The effect of principal component analysis on machine learning accuracy with high-dimensional spectral data. *Knowledge Based Systems 19*(5), 363–370.

Hsieh, N. C. and L. P. Hung (2010). A data driven ensemble classifier for credit scoring analysis. *Expert Systems with Applications 37*(1), 534–545.

Huang, C. L., M. C. Chen, and C. J. Wang (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications 33*(4), 847–856.

Jiang, Y. (2009). Credit scoring model based on the decision tree and the simulated annealing algorithm. In *Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering*, Washington, DC, USA, pp. 18–22. IEEE Computer Society.

Kira, K. and L. A. Rendell (1992). A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning*, San Francisco, CA, USA, pp. 249–256. Morgan Kaufmann Publishers Inc.

Kittler, J. (1998). Combining classifiers: A theoretical framework. *Pattern Analysis & Applications 1*(1), 18–27.

Kohavi, R. and G. H. John (1997). Wrappers for feature subset selection. *Artificial Intelligence 97*(1).

Kolde, R., S. Laur, P. Adler, and J. Vilo (2012). Robust rank aggregation for gene list integration and meta-analysis. *Bioinformatics 28*(4), 573–580.

Kumar, G. and K. Kumar (2011). A novel evaluation function for feature selection based upon information theory. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 395–399.

Kumar, R. and S. Vassilvitskii (2010). Generalized distances between rankings. In *Proceedings of the 19th international conference on World wide web*, WWW'10, New York, NY, USA, pp. 571–580. ACM.

Kuncheva, L. I., J. C. Bezdek, and P. W. Duin (2001). Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition 34*(2), 299–314.

Kwang, L. (2002). Combining multiple feature selection methods. In *Proceedings of the Mid-Atlantic Student Workshop on Programming Languages and Systems*.

Ledesma, S., G. Cerda, G. Aviña, D. Hernández, and M. Torres (2008). Feature selection using artificial neural networks. In *Proceedings of the 7th Mexican International Conference on Artificial Intelligence: Advances in Artificial Intelligence*, MICAI '08, Berlin, Heidelberg, pp. 351–359. Springer-Verlag.

Legrand, G. and N. Nicoloyannis (2005). Feature selection method using preferences aggregation. In *Proceedings of the 4th international conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'05, Berlin, Heidelberg, pp. 203–217. Springer-Verlag.

Li, S., E. J. Harner, and D. Adjeroh (2011). Random KNN feature selection - a fast and stable alternative to Random Forests. *BMC Bioinformatics 12*(1), 450–461.

Liu, H. and H. Motoda (1998). *Feature Selection for Knowledge Discovery and Data Mining*. Norwell, MA, USA: Kluwer Academic Publishers.

Liu, H. and L. Yu (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering 17*(4), 491–502.

Liu, Y. and M. Schumann (2005). Data mining feature selection for credit scoring models. *Journal of the Operational Research Society 56*(9), 1099–1108.

Mak, M. W. and S. Y. Kung (2008). Fusion of feature selection methods for pairwise scoring svm. *Neurocomputing 71*(16-18), 3104–3113.

Matjaz, V. (2012). estimating probability of default and comparing it to credit rating classification by banks. *Economic and business review 14*(4), 299–320.

Merbouha, A. and A. Mkhadri (2006). Méthodes de scoring non-paramètriques. *Revue de Statistique Appliquée 56*(1), 5–26.

Molina, L. C., L. Belanche, and A. Nebot (2002). Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of the IEEE International Conference on Data Mining*, pp. 306 – 313. IEEE Computer Society.

Paleologo, G., A. Elisseeff, and G. Antonini (2010). Subagging for credit scoring models. *European Journal of Operational Research 201*(2), 490–499.

Peng, H., F. Long, and C. Ding (2005). Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*(8), 1226–1238.

Pihur, V., S. Datta, and S. Datta (2009). RankAggreg, an R package for weighted rank aggregation. *BMC Bioinformatics 10*(1), 62–72.

Piramuthu, S. (2004). Evaluating feature selection methods for learning in data mining applications. *European Journal of Operational Research 156*(2), 483 – 494.

Piramuthu, S. (2006). On preprocessing data for financial credit risk evaluation. *Expert Systems with Applications 30*(3), 489–497.

Rodriguez, I., R. Huerta, C. Elkan, and C. S. Cruz (2010). Quadratic Programming Feature Selection. *Journal of Machine Learning Research 11*(4), 1491–1516.

Sadatrasoul, S. M., M. Gholamian, M. Siami, and H. Z. (2013). Credit scoring in banks and financial institutions via data mining techniques: A literature review. *Journal of Artificial Intelligence and Data Mining 1*(2), 119–129.

Saeys, Y., T. Abeel, and Y. Peer (2008). Robust feature selection using ensemble feature selection techniques. In *Proceedings of the European conference on Machine Learning and Knowledge Discovery in Databases - Part II*, ECML PKDD '08, Berlin, Heidelberg, pp. 313–325. Springer-Verlag.

Saeys, Y., I. n. Inza, and P. Larrañaga (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics 23*(19), 2507–2517.

Schebesch, K. B. and R. Stecking (2005). Support vector machines for classifying and describing credit applicants: detecting typical and critical regions. *Journal of the Operational Research Society 56*(9), 1082–1088.

Sculley, D. (2007). Rank aggregation for similar items. In *Proceedings of the Seventh SIAM International Conference on Data Mining*, pp. 587–592.

Thomas, L. (2009). *Consumer credit models: pricing, profit, and portfolios*. Oxford University Press.

Thomas, L., J. Crook, and D. Edelman (2002). *Credit Scoring and Its Applications*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Thomas, L. C. (2000). A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. *International Journal of Forecasting 16*(2), 149–172.

Tsai, C. F. and J. W. Wu (2008). Using neural network ensembles for bankruptcy prediction and credit scoring. *Expert Systems with Applications 34*(4), 2639–2649.

Tufféry, S. (2007). *Data mining et statistique décisionnelle: l'intelligence des données*. Editions Ophrys.

Tuv, E., A. Borisov, G. Runger, K. Torkkola, I. Guyon, and A. R. Saffari (2009). Feature selection with ensembles, artificial variables, and redundancy elimination. *Journal of Machine Learning Research 10*(9), 1341–1366.

Vapnik, V. (1995). *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc.

Šušteršič, M., D. Mramor, and J. Zupan (2009). Consumer credit scoring models with limited data. *Expert Systems with Applications 36*(3), 4736–4744.

Wang, J., A. R. Hedar, S. Wang, and J. Ma (2012). Rough set and scatter search metaheuristic based feature selection for credit scoring. *Expert Systems with Applications 39*(6), 6123–6128.

Wu, O., H. Zuo, W. Zhu, Mingliangand Hu, J. Gao, and H. Wang (2009). Rank aggregation based text feature selection. In *Proceedings of the Web Intelligence*, pp. 165–172.

Yang, L. (2001). New issues in credit scoring application. Working Papers 16/2001, Institut Fur Wirtschaftsinformatik.

Yu, L. and H. Liu (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 856–863.

Yu, L. and H. Liu (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research 5*(1), 1205–1224.

Yun, C., D. Shin, H. Jo, J. Yang, and S. Kim (2007). An experimental study on feature subset selection methods. In *Proceedings of the 7th IEEE International Conference on Computer and Information Technology*, CIT '07, Washington, DC, USA, pp. 77–82. IEEE Computer Society.

Zhang, D., X. Zhou, S. C. H. Leung, and J. Zheng (2010). Vertical bagging decision trees model for credit scoring. *Expert Systems with Applications 37*(12), 7838–7843.

Publications

The following are published articles out of this thesis.

Journal Papers

1. Bouaguel, W., Bel Mufti, G. and Limam, M. (2013), A three-stage feature selection using quadratic programming for credit scoring. Applied Artificial Intelligence: An International Journal, Vol. 27, No. 8, September 2013, pp. 721-742.

2. Bouaguel, W. and Bel Mufti, G. (2012), An improvement direction for filter selection techniques using information theory measures and quadratic optimization. International Journal of Advanced Research in Artificial Intelligence, Vol. 1, No: 5, August 2012, pp. 7-11.

Conference Papers

1. Bouaguel, W. and Limam, M. (2014), An Ensemble Wrapper Feature Selection for Credit Scoring. Advances in Intelligent Systems and Computing (AISC) Series of Springer, Vol. 335, pp. 619-631.

2. Bouaguel, W., Bel Mufti, G. and Limam, M. (2013), Similarity aggregation a new version of rank aggregation applied to credit scoring case. Mining Intelligence and Knowledge Exploration, Lecture Notes in Computer Science, Vol. 8284, pp. 618-628.

3. Bouaguel, W., Bel Mufti, G. and Limam, M. (2013), Rank aggregation for

filter feature selection in credit scoring. Mining Intelligence and Knowledge Exploration, Lecture Notes in Computer Science, Vol. 8284, pp. 7-15.

4. Bouaguel, W., Ben Brahim, A. and Limam, M. (2013), Feature selection by rank aggregation and genetic algorithms, Proceedings of the 5th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2013), Vilamoura, Algarve, Portugal, 19-22 September 2013, pp. 33-40.

5. Ben Brahim, A., Bouaguel, W. and Limam, M. (2013), Feature selection aggregation versus classifiers aggregation for several data dimensionalities, Proceedings of the International Conference on Control, Engineering & Information Technology (CEIT 2013), Sousse, Tunisia, 4-7 June 2013, pp. 10-15.

6. Bouaguel, W., Bel Mufti, G. and Limam, M. (2013), On the effect of search strategies on wrapper feature selection in credit scoring, Proceedings of the International Conference on Control, Decision and Information Technologies (CODIT 2013), Hammamet, Tunisia, 6-8 May 2013, pp. 60-67.

7. Bouaguel, W., Bel Mufti, G. and Limam, M. (2013), A fusion approach based on wrapper and filter feature selection methods using majority vote and feature weighting, Proceedings of the International Conference on Computer Applications Technology (ICCAT 2013), Sousse, Tunisia, 20-22 January 2013, pp. 47-53.

8. Bouaguel, W., Bel Mufti, G. and Limam, M. (2012), Quadratic Programming for Feature Selection in Credit Scoring, Proceedings of the Third Meeting on Statistics and Data Mining (MSDM 2012), Hammamet, Tunisia, 14-15 March 2012, pp. 7-14.

Book Chapters

1. Bouaguel, W., Bel Mufti, G. and Limam, M. (2015), A new feature selection technique applied to credit scoring data using a rank aggregation approach based on: optimization, genetic algorithm and similarity. Knowledge Discovery & Data Mining (KDDM) for Economic Development: Applications, Strategies and Techniques, Taylor & Francis (Accepted).

2. Ben Brahim, A., Bouaguel, W. and Limam, M. (2014), Combining feature selection and data classification using ensemble approaches: application to cancer diagnosis and credit scoring. Case Studies in Intelligent Computing Achievements and Trends, Taylor & Francis, August 2014, pp. 517-532.

Appendix A

Feature categories and datasets description

A.1 Feature categories

In general, a feature can describe either a qualitative or quantitative characteristic of a credit applicant. Examples of qualitative characteristics are gender, occupation and marital status. Examples of quantitative characteristics are age, amount of a loan. Qualitative and quantitative features can each be divided into two main categories, as depicted in Figure (A.1).

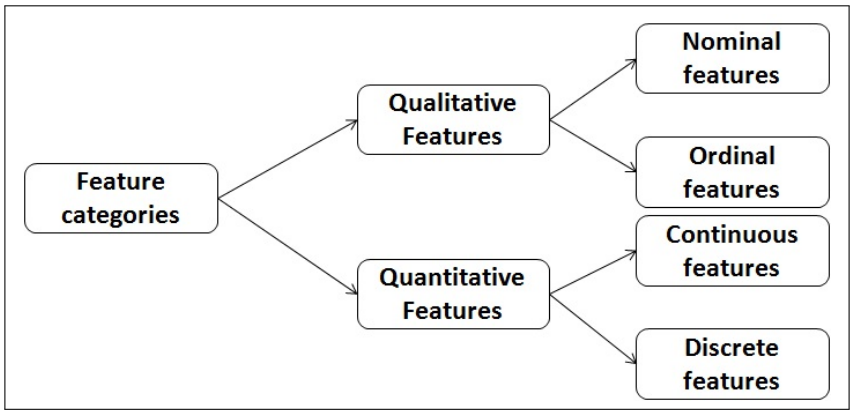


Figure A.1: Features categories.

A.1.1 Qualitative features

Qualitative features are also called categorical, describe a quality of the credit applicant. Categorical features can be either nominal or ordinal. A nominal feature is a categorical feature that has two or more categories (levels), with no intrinsic ordering to the categories. Purpose of credit is an example of nominal features with four categories {new car, household appliances, education, business } and there is no intrinsic

ordering to the categories.

An ordinal feature is a categorical feature where the categories are ordered or hierarchical i.e. we have a rank for each category (1^{st}, 2^{nd}, 3^{rd}, ...). For example, the feature educational level is an ordinal feature with three categories {elementary school, high school, college}. These categories can be easily ordered.

A.1.2 Quantitative features

Quantitative features describe some quantity about the credit applicant and are often measured or counted. These features can be either continuous or discrete. A continuous variable is one that could take any value in an interval. A discrete feature is one that can only take specific numeric values but those numeric values have a clear quantitative interpretation.
Because qualitative data always have a limited number of alternative values, such variables are also described as discrete. All numeric qualitative features are discrete, while some quantitative features are discrete and some are continuous. For statistical analysis, qualitative features can be converted into discrete numeric data by simply counting the different values that appear.

A.2 datasets description

This section reports some benchmark data sets that are used to evaluate the performance of different feature selection methods. While the Australian and German datasets are downloaded from the UCI Machine Learning Repository, The HMEQ dataset is available at SAS software and the Tunisian dataset is the result of collected information from a Tunisian bank.

A.2.1 Australian dataset

As discussed earlier in Chapter 1 The Australian dataset is composed of 690 instances where 307 ones are creditworthy while 383 are not.There are 6 numerical and 8 categorical features and all feature names and values have been changed to meaningless symbols for confidentiality. The labels have been changed for the convenience of the

statistical algorithms. For example, attribute 4 originally has 3 labels p,g,gg and theses have been changed to labels 1,2,3.

Variable	Description	Type	Description of modalities
A1	No description is available	Categorical	This feature has 2 modalities $\{0, 1\}$, where no description is available about the significance of each modality
A2	No description is available	Continuous	No modalites
A3	No description is available	Continuous	No modalites
A4	No description is available	Categorical	This feature has 2 modalities $\{1, 2, 3\}$, where no description is available about the significance of each modality
A5	No description is available	Categorical	This feature has 14 modalities $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$, where no description is available about the significance of each modality.
A6	No description is available	Categorical	This feature has 9 modalities $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, where no description is available about the significance of each modality
A7	No description is available	Continuous	No modalites
A8	No description is available	Categorical	This feature has 2 modalities $\{0, 1\}$, where no description is available about the significance of each modality
A9	No description is available	Categorical	This feature has 2 modalities $\{0, 1\}$, where no description is available about the significance of each modality
A10	No description is available	Continuous	No modalites
A11	No description is available	Categorical	This feature has 2 modalities $\{0, 1\}$, where no description is available about the significance of each modality
A12	No description is available	Categorical	This feature has 3 modalities $\{1, 2, 3\}$, where no description is available about the significance of each modality
A13	No description is available	Continuous	No modalites
A14	No description is available	Continuous	No modalites
A15	Creditability	Categorical	1: good applicant 2: bad applicant

A.2.2 German dataset

The German dataset covers a sample of 1000 credit consumers where 700 instances are creditworthy and 300 are not. For each applicant 21 numeric input variables are available. More details are given in the table below.

Variable	Description	Type	Description of modalities
Alter	Age	Continuous	No modalites
Beruf	Occupation	Categorical	1: unemployed / unskilled with no permanent residence 2: unskilled with permanent residence 3: skilled worker / skilled employee / minor civil servant 4: executive / self-employed / higher civil servant
Beszeit	Has been employed by current employer for	Categorical	1: unemployed 2: ≤ 1 year 3 : $1 \leq .. < 4$ years 4 : $4 \leq .. < 7$ years 5 :≥ 7 years
Beurge	Further debtors / Guarantors	Categorical	1: none 2: Co-Applicant 3: Guarantor
Bishkred	Number of previous credits at this bank (including the running one)	Categorical	1: zero 2: one or two 3: three or four 4: five or more
Famges	Marital Status / Sex	Categorical	1: male: divorced / living apart 2: female: divorced / living apart / married 3: male: single /married /widowed 4: female: single
Gastarb	Foreign worker	Categorical	1: yes 2: no
Hoehe	Amount of credit in "Deutsche Mark"	Continuous	No modalites
Kredit	Creditability	Binary	0: not credit-worthy 1: credit-worthy

Laufkount	Balance of current account	Categorical	1: no running account 2: no balance or debit 3: $0 \leq .. < 200$ DM 4 : 200 DM / checking account for at least 1 year
Laufzeit	Duration in months	Categorical	No modalites
Moral	Payment of previous credits	Categorical	0: hesitant payment of previous credits 1: problematic running account / there are further credits running but at other banks 2: no previous credits / paid back all previous credits 3: no problems with current credits at this bank 4: paid back previous credits at this bank
Pers	Number of persons entitled to maintenance	Categorical	1: 0 to 2 2: 3 or more
Rate	Installment in % of available income	Categorical	1: ≥ 35 2: $25 \leq ... < 35$ 3: $20 \leq ... < 25$ 4 : < 20
Telef	Telephone	Categorical	1: yes 2: no
Sparkont	Value of savings or stocks	Categorical	1:not available / no savings 2: < 100 DM 3: $100 \leq ... < 500$ DM 4: $500 \leq ... < 1000$ DM 5 : ≥ 1000 DM
Verm	Most valuable available assets	Categorical	1: not available / no assets 2: Car / Other 3: Savings contract with a building society / Life insurance 4: Ownership of house or land
Verw	Purpose of credit	Categorical	1: new car 2: used car 3: items of furniture

			4: radio / television 5: household appliances 6: repair 7: education 8: vacation 9: retraining 10: business
Weitkred	Further running credits	Categorical	1: at other banks 2: at department store or mail order house 3: no further running credits
Wohn	Type of apartment	Categorical	1: free apartment 2: rented flat 3: free apartment
Wohnzeit	Living in current household for	Categorical	1:< 1 year 2: $1 \leq ... < 4$ years 3: $4 \leq ... < 7$ years 4 : ≥ 7 years

A.2.3 HEMQ dataset

The HMEQ dataset is composed of 5960 instances where 4771 instances are creditworthy and 1189 are not. For each applicant, 12 input variables are available, more descriptions are given below.

Variable	Description	Type	Description of modalities
BAD	Creditability	Binary	1: applicant defaulted on loan or seriously delinquent 0: applicant paid loan
CLAGE	Age of oldest credit line in months	Continuous	No modalites
CLNO	Number of credit lines	Continuous	No modalites
DEBTINC	Debt-to-income ratio	Continuous	No modalites
DELINQ	Number of delinquent credit lines	Continuous	No modalites
DEROG	Number of major derogatory reports	Continuous	No modalites

JOB	Occupational categories	Categorical	This feature has 6 modalities $\{1, 2, 3, 4, 5, 6\}$, where no description is available about the significance of each modality
LOAN	Amount of the loan request	Continuous	No modalites
MORTDUE	Amount due on existing mortgage	Continuous	No modalites
NINQ	Number of recent credit inquiries	Continuous	No modalites
REASON	reason for credit	Categorical	DebtCon=debt consolidation HomeImp=home improvement
VALUE	Value of current property	Continuous	No modalites
YOJ	Years at present job	Continuous	No modalites

A.2.4 Tunisian dataset

Tunisian dataset covers a sample of 2970 instances of credit consumers where 2523 instances are creditworthy while 446 are not. Each credit applicant is described by 22 input variables as described below.

Variable	Description	Type	Description of modalities
ZONE GEO	Geographic zone	Categorical	1: North 2: South 3: Center
GEN	Gender of the credit applicant	Categorical	1:female 2:mal
MKT	Non description is available	Categorical	1:PAR 2:PRF
NOUV SM	Profession of the applicant	Categorical	1: Liberal 2 : Lawyer and likened 3: Private employee 4 : Students / others 5: Doctor and likened 6: Retreat 7: Government employee
AGE	Age	Continuous	No modalites

EPARGNE	saving account	Categorical	1: Saving account 0: No saving
TOTENG	Amount of other credits	Continuous	No modalites
DUR	Duration in months	Continuous	No modalites
CMR	No description is available	Continuous	No modalites
CMR2	No description is available	Continuous	No modalites
AUTOR	No description is available	Continuous	No modalites
MNTDEBLOC	No description is available	Continuous	No modalites
ENCOUR	Amount of further running credits	Continuous	No modalites
MULTIBANC	Multiple running account in other banks	Categorical	1: Yes 2: No
DOMICIL	No description is available	Categorical	1 :Transfer of delegation 2: Pension account 3: No domiciliation 4: Directly paid wages
Safir	The applicant has a safir account	Categorical	1: Normal account 2: Safir
STAT FAM	Marital Status	Categorical	1: Single 2: Divorced 3: Married 4: Widower or Widow
SAL MEN	Net monthly salary	Continuous	No modalites
REV MEN	Net monthly income	Continuous	No modalites
NBR	Number of dependants under the age of 18	Continuous	No modalites
RVS	No description is available	Continuous	No modalites

Appendix B
Classification methods

This appendix presents classification algorithms which are used to evaluate the candidate subsets in the wrapper framework. Understanding the foundations of these algorithms can be helpful for the comprehension of this work.

B.1 Artificial Neural Network

ANN were originally developed in machine learning field and become an important data mining method. A neural network is composed of a set of elementary computational units, called neurons, connected together through weighted connections. Every neuron, also called a node, represents an autonomous computational unit and receives as inputs the description of an observation x_i, $(x_i^1, ..., x_i^d)$ called signal. Each signal is attached with an importance weight after that the neuron elaborates the input signals, their importance weights and the threshold value through something called a combination function. The combination function produces a value called potential. An activation function transforms the potential into an output signal. The activation function is defined as follows:

$$f(x_i) \quad = \quad \sum_{j=1}^{d} \beta_j x_i^j + \beta_0 \quad = \quad \beta^T x_i, \tag{B.1}$$

where β_j, $j = 1, ..., d$ is the weight associated for each signal. The final output y_i of the neurone is decided according to $f(x_i)$ sign.

$$y_i = \begin{cases} 0 \ if \ \beta^T x_i \leq 0 \\ 1 \ Otherwise \end{cases} \tag{B.2}$$

ANNs are found to be a powerful solution. Their performance is dependent on initial condition, network topologies and training algorithms. This may be one reason why the results of ANN vary for credit scoring.

B.2 Support Vector Machines

Among the new methods for credit scoring, SVM is one of most promising methods. The use of SVM in financial application has been previously examined by several works (Schebesch and Stecking 2005; Huang et al. 2007; Bellotti and Crook 2009). SVM was first proposed by Vapnik (1995) and recently becomes one of the most applied methods in data mining. There are many reasons for choosing SVM (Burges 1998), it requires less prior assumptions about the input data and can perform a nonlinear mapping from an original input space into a high dimensional feature space, in which it constructs a linear discriminant function to replace the nonlinear function in the original low-dimension input space. A simple description of the SVM algorithm is provided as follows. Given a training set $\{x_i, y_i\}_{i=1}^n$ with input vector $x_i = [x_i^1, x_i^2, \ldots, x_i^d]^T$ and target variable $y_i \in \{+1, -1\}$, the original formulation of SVM algorithm satisfies the following conditions:

$$\begin{cases} \beta^T.\phi(x_i) + b \geq +1 \ if \ y_i = +1 \\ \beta^T.\phi(x_i) + b \leq -1 \ if \ y_i = -1 \end{cases} \tag{B.3}$$

which is equivalent to

$$y_i(\beta^T.\phi(x_i) + b) - 1 \geq 0, \ i = 1, ..., n, \tag{B.4}$$

where β represents the weight vector and b the bias. $\phi(x_i)$ is a nonlinear mapping function. From equation (B.4) we come down to the construction of two parallel bounding hyperplanes at opposite sides of a separating hyperplane $\beta^T.\phi(x_i) + b = 0$ in the feature space with the margin width between both hyperplanes equal to $\frac{2}{||w||^2}$. the classifier then takes the decision function form $sgin(\beta^T.\phi(x_i) + b)$.

B.3 Decision Trees

According to Thomas et al. (2002) the idea of DT is to split the set of applications into different sets and then identify each of these sets as good or bad depending on what the majority in that set is. The idea was developed for general classification problems by Breiman et al. (1984) and was used for the first time by Frydman et al. (1985).

This method is very simple and can be described according to this scheme (Giudici 2003): A DT consists of nodes and edges, the root node defines the first split of the credit applicants sample. Each internal node splits the instance sample into two subsets. Each node contains individuals of a single class. The operation is repeated until the division in sub-populations is no more possible.

B.4 K-nearest-Neighbor

The main idea of KNN is to choose a distance measure on the space of application data in order to measure how distant any two applicants are (Thomas et al. 2002). Then, using a learning sample of past applicants presented by the couples (x_i, y_i), a new applicant $x_{i\prime}$ is classified as good or bad depending on the proportions of goods and bads among the k nearest applicants from the learning sample. The two parameters needed to run this approach are the distance metric and how many applicants k constitute the set of nearest neighbors. A commonly used distance metric with KNN is the Euclidean distance given by :

$$d(x_i, x_{i\prime}) = \sqrt{\sum_{j=1}^{d}(x_i^j - x_{i\prime}^j)^2} \tag{B.5}$$

This method is usually used for heterogeneous data with missing data. Although simple, the choice of the number of neighbors k is still a difficult task. This number is either fixed beforehand or chosen by crossed validation (Merbouha and Mkhadri 2006).