2-2021

A New Feature Selection Method Based on Class Association Rule

Sami A. Al-Dhaheri
The Graduate Center, City University of New York

A NEW FEATURE SELECTION METHOD BASED ON CLASS ASSOCIATION RULE

by

SAMI AL-DHAHERI

A dissertation submitted to the Graduate Faculty in Computer Science

in partial fulfillment of the requirements for the degree of Doctor of Philosophy,

The City University of New York

2021

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirements for the degree of Doctor of Philosophy.

Professor Abdullah Uz Tansel

Date | Chair of Examining Committee

Professor Ping Ji

Date | Executive Officer

Professor William Sakas

Professor XiangDong Li

Professor Reda Alhajj

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

A New Feature Selection Method based on Class Association Rule

by

Sami Al-Dhaheri

Adviser: Professor Abdullah Uz Tansel

Feature selection is a key process for supervised learning algorithms. It involves discarding irrelevant attributes from the training dataset from which the models are derived. One of the vital feature selection approaches is Filtering, which often uses mathematical models to compute the relevance for each feature in the training dataset and then sorts the features into descending order based on their computed scores. However, most Filtering methods face several challenges including, but not limited to, merely considering feature-class correlation when defining a feature's relevance; additionally, not recommending which subset of features to retain. Leaving this decision to the end-user may be impractical for multiple reasons such as the experience required in the application domain, care, accuracy, and time. In this research, we propose a new hybrid Filtering method called Class Association Rule Filter (CARF) that deals with the aforementioned issues by identifying relevant features through the Class Association Rule Mining approach and then using these rules to define weights for the available features in the training dataset. More crucially, we propose a new procedure based on mutual information within the CARF method which suggests the subset of features to be retained by the end-user, hence reducing time and effort. Empirical evaluation using small, medium, and large datasets that belong to various dissimilar domains reveals that CARF was able to reduce the dimensionality of the search space when contrasted with other common Filtering methods. More importantly, the classification models devised by the

different machine learning algorithms against the subsets of features selected by CARF were highly competitive in terms of various performance measures. These results indeed reflect the quality of the subsets of features selected by CARF and show the impact of the new cut-off procedure proposed.

Acknowledgments

I express my deepest gratitude to my supervisor Professor Abdullah Uz Tansel for his continuous encouragement, guidance, and support during the preparation of the dissertation.

I thank my supervisory committee, Professor William Sakas, Professor Reda Alhajj, and Professor XiangDong Li, for providing helpful suggestions and constructive criticisms. I also thank Dilvania Rodriguez for all her help with administrative matters.

Kudos to all my friends and family who helped me in a multitude of ways to reach this stage in my life. My heartfelt appreciation goes to my parents who have instilled within me a love for intellectual pursuits.

Table of Contents

List of Figures

List of Tables

Chapter One

1. Introduction

1.1 Introduction

In the era of big data, the importance of generating intelligence from large databases is growing significantly due to advances in computer networks, hardware, and mobile technology. Datasets of different business domains are often associated with high dimensionality (they consist of a large number of features), contain noise (duplicate instances, missing values, inconsistencies, etc.), and consist of unstructured features (attributes that are hard to be represented in a relational database such as email content). These data characteristics make identifying the best set of features during the process of feature selection challenging as this task requires thorough pre-processing and domain knowledge availability [25].

Before delving into the feature selection topic, it is important to understand what can be gained from machine learning. The key element of this process is that learning is from a massive amount of historical data. We label human and animal behaviors as intelligent as they include learning experiences, thus learning is intrinsic to human life. As humans we adjust and adapt to new scenarios daily; remembering, adapting, and generalizing are important parts of learning [108]. Generalizing is identifying similarities between different circumstances and applying the knowledge derived from one place in other appropriate places [22]. Learning becomes incredibly important as it involves knowledge and intelligence, so modelling these aspects into computers becomes fundamental [108]. The success of machine learning depends on many factors, primarily on the quality of input data, the number of data observations, type of learning, and variable types, among others. Data pre-processing to deal with noisy data is crucial for the machine learning

algorithm to ensure output quality and a smooth learning process. The data pre-processing phase comprises 90% of the work during the learning process and includes data cleansing, data normalization, data transformation, data balancing, feature extraction, feature selection, and others [122].

Feature selection is a primary process that affects the outcome of predictive machine learning algorithms as it discards irrelevant features as early as possible and reduces the search space for the problem [80,169]. The aim of feature selection is to determine a set of features automatically based on a measure of relevance [79]. To be exact, in supervised learning tasks in which the goal is to predict a certain variable, users look for improvements in the classification learning algorithm in terms of predictive accuracy, training speed, and simplicity of the data representation. This can be accomplished using feature selection methods in which the efficiency of the learning process and building the classification models will be improved as a smaller number of features will be processed by the learning algorithm [21].

The quality of the data being processed may affect the learning process of the model of any classification algorithm based on the available noise which may deteriorate the model's performance in terms of classification accuracy [50]. Hence, by adopting feature selection before learning, noisy attributes are usually removed to achieve desirable outcomes. More importantly, feature selection improves the performance by reducing the model's overfitting; it establishes efficient learning, since only the selected attributes are considered during the learning phase while providing a deeper understanding of the underlying processes that are involved in data processing [123].

Figure 1 shows the feature selection process (dashed blue line) as part of the supervised learning lifecycle (classification problems). Once the training dataset is loaded and data cleansing is

applied, the process of feature selection starts. Using feature selection methods, relevance for all features in the input dataset, other than the class, are detected. The output will be a complete set of features along with their corresponding relevance and a subset of which is passed to the classification algorithm for training to derive classification models.

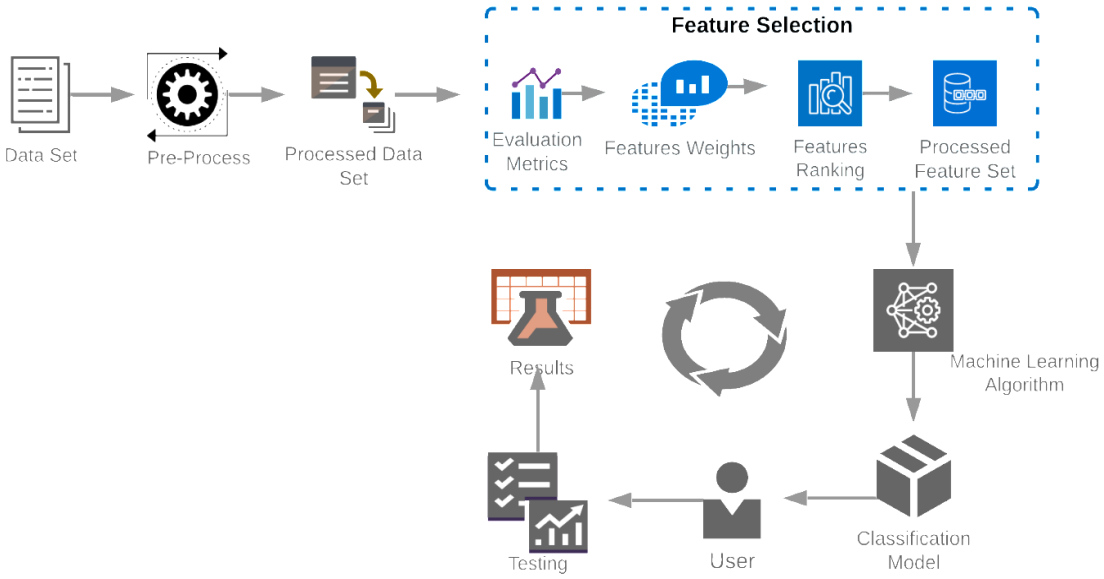

Figure 1: Feature Selection Phase of the Entire Learning Process

One of the common feature selection methods used to efficiently identify the subset of features from the dataset prior to learning is the Filtering approach [27]. Using this approach, the relevance (score) of each feature in the training dataset is determined using a mathematical model and then the available features sorted based on the computed scores. The user is then left to choose which subset of features to retain in a difficult process that usually requires experience, accuracy, and care [72,140]. Common Filtering methods are Information Gain (IG), Chi-square testing (CST),

Odds Ratio (OR), Gain Ratio (GR), and Symmetrical Uncertainty (SU), plus many others [138,98,137,184].

Most Filtering methods merely consider the correlations between the available features and the class label to define relevance and ignore features' similarity [19,35,149,161]. It is imperative to capture feature-to-feature correlations to discard redundant features otherwise the retained features will have a high level of redundancy; this may lead to increasing the search space for the machine learning algorithms as well as complicating the models derived [52,65,191]. In addition, the end-user will have to manually check all possible features to reduce the similarity among features in the result—an onerous task if a massive number of features is found [26,140].

Class Association Rule (CAR) mining is a promising classification approach that was initially developed by Liu, Hsu, and Ma (1998) [95] in an algorithm called Classification-based Association (CBA). This approach utilizes Association Rule mining methods to discover a subset of rules that can be represented as a classification model, and then in turn be used for forecasting purposes in classification problems [4,11,101,165]. Since the CAR mining approach ensures that each generated rule covers at least one data observation, this approach with some modifications can be utilized for feature selection to minimize feature similarity.

This research investigates the problem of feature selection by developing a new Filtering method based on the CAR mining approach. We develop a new Class Association Rule Filter method (CARF), which efficiently produces simple rules from the set of the available features in the training dataset, and then assigns scores to the available features. More importantly, CARF is associated with a new cut-off procedure that smartly recommends which subsets of features should be chosen thus saving the end-user vital resources including time, effort, and domain experience. Section 1.3 provides more details on the key contributions of this research.

CARF investigates features' correlations in the training dataset to define their relevance based on learning rules. Then it prunes all irrelevant correlations, i.e. rules that are redundant, and keeps rules associated with just one feature value in the rule's body. CARF ensures that rules learnt have no common training instances hence minimizing the similarities among features in the result set. These rules are then utilized to define the feature's relevance. More details on the CARF method are given in Chapter Three.

Some of the chapters in this dissertation are submitted for dissemination in reputable academic journals related to data processing and data mining.

1.2 Research Problem, Aims, and Objectives

Dimensionality reduction involves the pre-processing of high-dimensional data to eliminate unimportant features to produce the same or better performance in evaluating, visualizing, and modelling. The best way to reduce dimensionality is by feature selection i.e. choosing the input dimensions that represent the relevant features to solve a particular problem [79,190].

Feature selection can be considered part of the learning process of classification in machine learning. The main goal of classification algorithms is to construct a classification system (model) from an input dataset (training dataset). The model is then evaluated in terms of its predictive power against another dataset (test dataset) that the model did not encounter during the learning process [1]. During the supervised learning process, the complete set of attributes (features) is used to train the model from the training dataset. The role of feature selection, using a relevancy measure, comes prior to learning, and mainly to select the smallest relevant subset of features from the training dataset. This enhances the efficiency of the training phase in terms of computing resources with any features that are irrelevant (based on the defined relevancy measure) being discarded at an early stage. However, the definition of relevancy in supervised learning is not yet

well defined [19,107,117]. For example, relevancy could be defined as dissimilarity among features other than the class label, or it could be related to a certain user objective.

The mathematical definition of feature selection is given below [79]:

Let T_i, with $1 \leq i \leq n$, be the search space of features (independent variables) $F_i = \{f_1, f_2, f_2, \ldots, f_n\}$. The data instance E_i is the domain of F_i .A data instance in the space is defined as $E_i = E_1 \times E_2 \times E_3 \times \ldots \times E_n$, and the independent variable (class label) C has a space of class labels. The aim is to construct a model as a function $F:E \rightarrow C$ according to the relevant features using the relevancy measures defined by the user. The subset of features chosen should preseve the complete features set in the training dataset.

The ultimate aims of this research are to develop:

1) A new feature selection method that reduces feature similarity and defines feature relevance using rules devised by the CAR approach.
2) A new filtering method that provides the end-user with a clear indicative measure of how many features to retain thus simplifying the manual process of feature selection, at least in Filtering methods.

The aims can be fulfilled using the below list of objectives:

- To survey feature selection approaches in the literature
- Critically analyze Filtering methods and show their advantages and disadvantages and the different mathematical models they utilize
- Understand the CAR mining approach especially its way of discovering the rules
- Investigate information theory methods to define a new metric for differentiating relevant from irrelevant features within Filtering methods

- Amend CAR mining to reduce feature-feature correlations based on data coverage
- Design and implement a new Filtering method using high level language such as Java
- Conduct in-depth experimentations on different classification datasets by evaluating the CARF and other state-of-the-art Filtering methods according to various performance metrics such as error rate, accuracy, recall, precision, and harmonic mean.

The key research questions that this dissertation will answer are:

1) How can we develop a new Filtering method that reduces features' similarity using simple rules derived by CAR mining?
2) How can we develop a new Filtering method that provides the end-user with an indicative measure of the number of features to choose in an automated manner?

1.3 Class Association Rule Mining

CAR mining is a special type of Association Rule Mining that deals with classification problems [119]. The input dataset will contain labelled instances, i.e. features as independent variables, and a class label as a dependent variable. The aim is to build a classification model that contains rules in the form $Cl: F \rightarrow C$ in which the antecedent of the rule contains conjunctive feature values, and the consequent is a class value [90]. Figure 2 depicts the main steps in a CAR mining algorithm.

The CAR mining approach generates the rules, usually in two steps [2]:

1) Discovering frequent itemsets (feature values with high frequency) (Definition 1)
2) Generating the rules.

The first step requires setting a threshold called the minimum support (minsupp) to a specific value as a measure for deciding which itemsets to keep during the training phase. An itemset (Definition

#5 in Chapter 3) consists of a combination of features values; for example, if the itemset contains 1 feature value it is said to be 1-itemset and when it contains two features values it is said to be 2-itemset, and so forth. Each itemset in the training dataset will be associated with a support, which denotes the frequency of the itemset in the training data set from the cardinality of that dataset. When the computed support of the itemset is larger than or equal to the minsupp threshold then that itemset is considered frequent, otherwise it will be infrequent. All frequent itemsets are stored in a data structure so they can be used to produce the rules in a later step; all infrequent itemsets are discarded as they do not hold any significant frequency power.

The first step involves discovering frequent itemsets using algorithms such as Apriori, Frequent Pattern Growth (FP-Growth), CHARM, and DECLAT [7,56,186,187]. Frequent itemsets are features' values with frequencies above a user-defined threshold called the minsupp. This step is exhaustive since it requires passing over the training dataset multiple times and a huge number of computations, i.e. calculating itemsets' supports (frequencies), especially when the minsupp threshold is set by the user to a low value [124,166,167]. Most existing algorithms discover frequent itemsets in a repetitive manner starting with itemsets that contain a single feature value, i.e. frequent 1-itemset, then from these candidate 2-itemsets (2 features values) and those that have supports above the minsupp will be declared as frequent 2-itemsets. Then, frequent 2-itemsets are used to produce the candidate 3-itemset and so forth. The algorithm terminates when no more frequent itemsets are found in the training dataset.

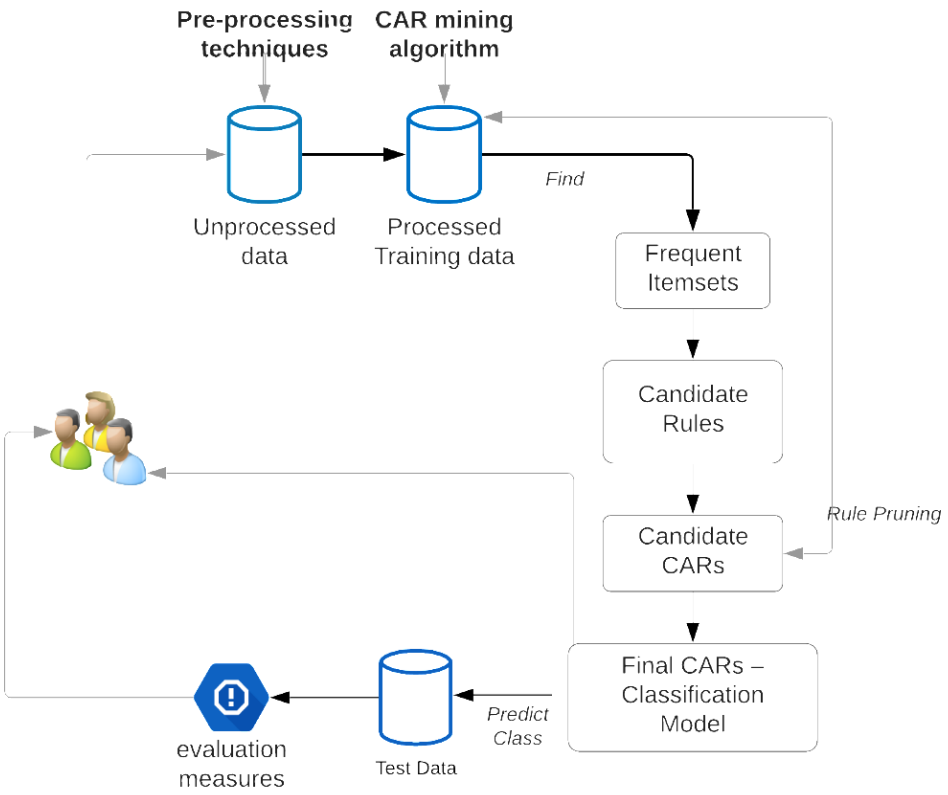

Figure 2: Class Association Rule Mining Main Steps

In generating the rules (Step 2), another threshold, called the minimum confidence (minconf), is employed. Each frequent itemset discovered in step (1) with two or more feature values will be considered a rule if its confidence value is larger than the minconf threshold. The confidence value of itemset ($A \rightarrow B$) denotes the frequency of A & B occurring together in the training dataset from the frequency of A occurring by itself. Any paternal rule that has a confidence value above the minconf is generated, whereas any potential rules with confidence values below the minconf will be removed. The confidence value is a goodness measure for the rules where only rules with high correlations between the features' values are kept. However, since the problem under consideration is a classification problem where the purpose is predicting the target class, then not all rules

generated (those which pass the minconf thresholds) are relevant. Only rules that have the class value on their consequent are required for prediction and therefore, all other rules will be ignored.

Once rules are generated, then the CAR mining algorithm sorts them and invokes a rule-pruning procedure to further reduce redundant rules to retain only those that are highly predictive [13]. This pruning procedure is implemented against the training dataset to evaluate each rule by checking whether it has an actual data coverage. The procedure begins by sorting the available rules in descending order according to their confidence and support values. For each rule, the pruning procedure then checks if it has successfully covered any training instances; if the rule has no data coverage (the check returns a false value), it will be removed. However, if the check returns a true value then,

a) The evaluated rule is inserted into the classification model
b) All training data instances covered by the evaluated rule will be marked for deletion.

The pruning procedure continues the process of evaluation until one of the two below conditions is met:

1) All training instances are marked for deletion OR
2) All candidate rules have been evaluated.

When one of these two possible conditions is met, the pruning procedure terminates, and the CAR mining algorithm returns all rules that have been stored in the classification model. These rules are then used for predicting the class value of test data instances [10].

The CAR mining approach has advantages and disadvantages summarized in Table 1.1 [3,11,159,164].

Table 1.1: Advantages and Disadvantages of the Class Association *Rule Mining Approach*

Class Association Rule Mining Advantages	**Class Association Rule Mining Disadvantages**
Produces simple classification models consisting of easy-to-interpret rules making this approach attractive for decision makers.	The number of items in the rule can be massive.
Easy to manually control the classification model.	Rules cover limited numbers of data instances.
Highly predictive performance in terms of classification accuracy in multiple application domains.	Often the numbers of retained rules in the classification model is large when compared to Decision Trees or Rule Induction approaches.

1.4 Issues and Contributions

1.4.1 Issue 1: Feature Relevance

The primary aim of performing feature selection is to discover a smaller set of features to represent the entire input dataset according to a specific measure. This is cost-effective in terms of computational cost, and more importantly, may lead to reduced overfitting of the model learnt by the learning algorithms [178,185]. In the Filtering approach, relevance is measured by how correlated the feature is with the class label by utilizing various measures for deriving the final subset of features [191]. For example, the ReliefF method [142], estimates a feature's score by computing the difference between the randomly chosen instance from the training dataset and its two closest instances with identical and opposite class labels. The IG method [137] computes a feature's score using the entropy, which shows how well a feature splits the data samples using the class values in the training dataset [152]. Since most Filtering methods merely evaluate feature-class correlations, there is potential for having redundancy among retained features. It will then be

the responsibility of the end-user to manually check these retained features to further filter them out, which is impractical when the number of retained features is high [26,140,168]. A more promising approach is to measure relevance based on both feature class and feature-feature correlations using supervised learning or new mathematical models.

We intend to develop a new hybrid Filtering method that makes use of learning rules via CAR mining to compute the feature's relevance. Using this approach, itemsets (feature values plus class value) are discovered from the training dataset in one single scan and then stored in an efficient vertical layout data structure (Line space). Within the data structure, each itemset is represented by ColumIDs:LineID to automatically locate its support and confidence value without having to scan the input dataset multiple times. In our proposed solution, we are only interested in finding frequent 1-itemsets in the format of (item value, class value), i.e. any itemset that has a single item (feature value) plus a class value. These frequent 1-itemsets are then converted into rules when they pass the minimum confidence threshold. They are then tested against the training dataset in an incremental manner, starting with the best one in terms of confidence, to eliminate any redundant rules. Once the complete non-redundant rules are retained then we use them to assign weights to the features in the training data.

1.4.2 Issue 2: Indicative Cut-off Measure

Most Filtering methods provide the end user with a complete set of sorted features [62,160]. Features sorted at the top probably have higher merit, yet this requires manual validation by a domain expert involving time and expertise [26]. To be exact, the availability of someone with in-depth knowledge and an understanding of the data and the application domain is vital for the success of Filtering methods, particularly when dealing with high-dimensional datasets or applications with complex features. The domain expert defines features' relevance and decides

which features should be discarded (irrelevant features) and those to be retained (relevant features). Relevance in this context does not correspond to the features set that yields best performance in terms of machine learning, but rather the smallest subset that the domain expert identifies from the results produced by the Filtering methods. It will be advantageous to have an indicative measure as to which features are important to ease the manual process of checking all features.

For example, when dealing with domain applications such as for credit card scoring, Filtering methods such as IG favor features with many possible values, such as credit card number, as this uniquely occurs with every instance in the dataset. Therefore, this feature presumably will be positioned at the top of the result. Nevertheless, when the user carefully examines the resulting subset of features, the credit card number could be dropped as a useless feature. Other features such as age, gender, marital status, owning a house, owning a car, and having an unpaid student loan will be more useful for the domain expert and hence utilized to build classification models. This example, if limited, shows that the presence of a domain expert can be vital in filtering out results offered by Filtering methods in feature selection. Nevertheless, the availability of domain experts is not cost or time effective for institutions and businesses. A more realistic solution is to provide a cut-off to distinguish between potential relevant and irrelevant features in the results set.

In this research we propose a new indicative procedure that simplifies the difficult process of manually checking each feature and it can be embedded within the proposed CARF method to calculate a cut-off point based on the information in the input dataset and using mutual information to give the user an indication of how many features should be retained. This cut-off procedure ensures that fewer, yet impactful, features are retained by the end-user thus improving the efficiency of the feature selection process and without having to drastically impact on the quality of the classification models devised against the retained features set. Empirical results in Chapter

Four show that the proposed cut-off procedure provides end-users with highly relevant yet small subsets of features that, when processed by machine learning algorithms, derive competitive classification models in terms of predictive accuracy and other performance measures.

1.4.3 Issue 3: Trial and Error

Since relevance is defined according to various mathematical equations, different scores can be obtained. Consequently, the outcome provided to the end-user may vary substantially from one Filtering method to another and from one dataset to another, making it difficult for the user to decide which method to use [72,139]. For instance, we ran three different Filtering methods namely IG, CFS, and Pearson Correlation [137,52,15] against the Sick dataset from UCI [94]. This dataset consisted of 30 features including the class label. Table 1.2 depicts the top five selected features of the considered Filtering method. The results, if limited, clearly reveal that the order of the features by the Filtering methods is different. Specifically, ReliefF selected 'Referral Source' as the best feature whereas the Gain Ratio and Pearson Correlation methods selected 'T3'. More importantly, the subsets of results produced by the considered Filtering methods are also different. For example, ReliefF selected 'TSH Measured' and 'On Thyroxine' as relevant features; however, these features were not considered to be relevant by the Pearson Correlation or Gain Ratio Filtering methods. Additionally, 'Age' was considered relevant by the Pearson Correlation method, yet this feature was not chosen by the Gain Ration or ReliefF methods.

Table 1.2: Top Five Features Selected by the IG, CFS, and PC Methods from *the Sick Dataset*

	ReliefF	**Gain Ratio**	**Pearson Correlation**
1.	Referral Source	T3	T3
2.	T3 Measured	Hypopituitary	Referral Source
3.	TSH Measured	Referral Source	T4U
4.	On Thyroxine	T4U	Age

5.	T4U Measured	T3 Measured	T3 Measured

The problem of score inconsistency may confuse the end-user. Recent approaches have dealt with results of instability for Filtering methods [72,168,139], but these also produce variations especially in how to discriminate between useful and useless features in the final subset which is not defined. The relevance can only be quantitively measured in the presence of a classification algorithm, as in the Wrapping approach [27,49,100,151,191], so the optimal features sets can be determined according to specific evaluation metrics (recall, accuracy, precision, harmonic mean, etc.). However, in the Filtering approach, the presented subset of features cannot guarantee the best classification performance as in the Wrapping approach. Therefore, measuring feature relevance becomes a difficult task that requires extensive trial and error by the user; the domain expert can also be relied on to manually assess the Filtering method's outcome. The latter was discussed in the previous sub-section.

We deal with the above issue by defining feature relevance based on non-overlapping simple rules induced from the training datasets. These rules are non-overlapping since they have been learnt from different parts of the training dataset, so they do not share training instances. The goodness of these rules is measured by the actual data coverage. Each feature of the training dataset is assigned a score using the items (feature values) appearing in the rules, and these rules have already been evaluated against the training dataset to reveal their goodness (confidence the relevance of the features has been measured). This approach does not guarantee an optimal subset of features as in the Wrapping approach, though it provides at least a measure of goodness to overcome the problem of trial and error.

1.5 Dissertation Structure

The dissertation comprises five chapters. Chapter Two reviews common approaches related to Filtering methods within the learning lifecycle and critically analyzes these approaches. Chapter Three proposes the new Filtering method and the cut-off procedure. Chapter Four is devoted to the implementation, testing, empirical results, and analysis of the proposed methods. In Chapter Four, we test CARF and the cut-off procedure on a large number of datasets and compare the results with the state-of the-art Filtering methods using machine learning. Finally, we conclude in Chapter Five.

Chapter Two

2. Literature Review

2.1 Introduction

Feature selection helps domain experts and users understand which features are available in the dataset that could improve the performance of the classification models [78]. Feature selection methods are normally divided into three main categories: Filtering, Wrapping, and Embedded [26]. Filtering uses mathematical methods to generate weights for the available features in the dataset. These weights are calculated against the training dataset using a static mathematical equation such as mutual information [41], Bayes Theorem [102], Relief [174] or probabilities [38]. These methods are highly efficient since they choose the features set without having to utilize any learning algorithm. Conversely, Wrapping methods employ a learning algorithm (often a classification method) to produce classification models using all possible feature-class combinations [74]. They ultimately select the features set that yielded the highest performance in terms of classification accuracy when processed by the learning algorithm. Despite the fact that Wrapping methods derive the best performance, they may not be feasible for data with high dimensionality; besides, they rely on the type of learning algorithm used, so when changing the learning algorithm the outcome will change (a different features set will be produced) [21]. Finally, Embedded methods combine both Wrapping and Filtering methods in a way that the learning algorithm has its own feature selection during the model-building phase [105,106]. An example of an embedded method is the Least Absolute Shrinkage and Selection Operator (LASSO) method for linear regression [195].

This chapter highlights feature selection methods and common approaches within Supervised Learning. Unsupervised feature selection is beyond the scope of this research - the focus is primarily Filtering methods. The chapter will critically analyze Filtering methods and pinpoint issues that researchers need to address to further enhance the process of selecting features and its impact on the outcome of learning approaches. We build upon recent research works related to Filtering methods [27,37,133,140,147,155,157] and reveal new issues such as Features' Relevance which is based on the feature's score, complex data, score variations, how to discriminate between useful and useless features, and live data processing.

The chapter comprises six sub-sections. Section 2.2 introduces learning types in machine learning and then briefly discusses the main steps of any Supervised Learning algorithm. Section 2.3 is devoted to evaluation measures in Supervised Learning, and Section 2.4 surveys recent literature related to Filtering methods and discusses common Wrapping and Embedded approaches. In Section 2.5, we highlight the challenges for Filtering methods, and finally, the chapter summary is given in Section 2.6.

2.2 Learning Types in Machine Learning

Machine learning is a part of Artificial Intelligence that, without any human assistance, improves the learning process of computers based on past experiences [136]. Machine learning algorithms such as Artificial Neural Network, Support Vector Machines (SVM), Decision Trees, Probabilistic Instance-based Learning [30,38,138,143], and others attempt to estimate potentially significant relationships between input and output variables by analyzing large datasets [114]. Two common learning types in machine learning that are pursued by machine learning algorithms are predictive and descriptive analysis. The former involves constructing a model to predict a specific variable value, called the class label, using historical labelled instances and a number of independent

variables (features) [88]. The latter involves producing certain knowledge to describe the dataset or grouping data instances to a set of specific groups. The learning process starts with the input of data instances with specific features and then exploring that data to discover useful patterns for decision-making using different learning schemes [5]. When Supervised and Unsupervised are integrated for specific datasets, we call the learning process Semi-Supervised Learning [108]. An example of a Semi-Supervised Learning is to use clustering algorithms for tasks that involve classification or prediction. Lastly, Reinforcement Learning involves learning from the surrounding environment by trial and error (learning curves from decisions and actions taken throughout the learning process) [66]. Figure 2.1 depicts the common types of learning which we intend to explain in subsequent sections.

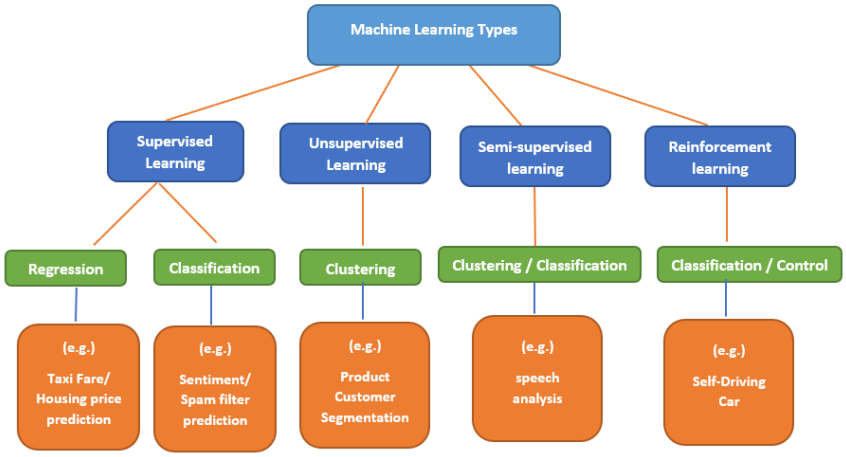

Figure 3: Machine Learning Types [110]

2.2.1 Supervised Learning Tasks

Supervised Learning is the most common type of learning and an important element of data science [85]. In Supervised Learning, the input will be a classification task represented by a training dataset with labelled variable, the purpose of which is to guess the value of the class label in an unseen test dataset as accurately as possible. The way Supervised Learning works is by modelling a classification model using the training dataset and the machine learning algorithm

(often a classification method); forecasting the class label in the test dataset is then performed by the classification model. The evaluation of the classification model is usually done using the test dataset, and when this is not possible, then testing methods such as ten-fold cross validation are used [73]. If the results of the evaluation are acceptable (good predictive accuracy is obtained), then we say that the model can be accepted and generalized; otherwise we reject the model. There are various evaluation metrics for Supervised Learning including classification accuracy, error rate, precision, recall, harmonic mean, and others [68]. Overall, Supervised Learning algorithms are modelled to learn from past experience, i.e. learn by data examples, using certain learning schemes such as statistical, probabilistic, information theory, and others. Thus, the analysis often involves constructing a mapping function that can be used to map new data examples, i.e. test instances [85].

There are several techniques used in Supervised Learning to construct classification models such as k-Nearest Neighbor (kNN) [59], Incremental Reduced Error Pruning (IREP) [29], Naïve Bayes [38], Back Propagation [144], C4.5 [137], Multi-Class Classification-based Association (MCAR) [165], AdaBoost [44], Random Forest [58], RIPPER (Furia)[28], Linear SVM [30], and others.

There are two primary tasks in Supervised Learning: classification and regression, as shown in Figure 4. A classification task mainly comprises two parts: model construction i.e. training on the input dataset to learn a mapping function and using the model for forecasting the class of new instances [126]. During the model's construction, the classification algorithm utilizes the relevant features that best describe the model [73]. There are several applications where Supervised Learning can be used including fraud detection [154], phishing detection [4], credit card scoring [154], medical diagnosis such as cancer detection [51], behavioral applications such as autism detection [164,170], dementia prediction [9], image recognition [183], consumer retention [70], weather forecasting [117], sport games prediction [23], and others.

One special case of classification tasks is regression, which occurs when the class label in the training dataset is continuous (numeric or fraction) [118]. Regression basically employs a statistical method for analyzing a labelled dataset to discover the relationship between the class label and independent variables. For example, if one would like to forecast volume of sales, income, temperature, stock price, etc. In its simplest form, regression analyzes the correlations between two variables in a dataset to check whether one can explain the other by linearly modelling their relationship [33]. Regression relies on a linear, quadratic, polynomial, nonlinear hypothesis [99]. There are many

different types of regression analysis such as linear regression [171], logistic regression [86], and polynomial regression [33].

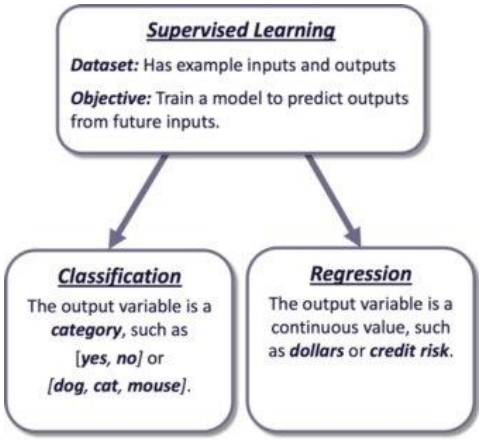

Figure 4: Supervised Learning [69]

2.2.2 Unsupervised Learning

Unsupervised Learning does not require class labels in the input dataset, and algorithms try to find patterns and similarities of the available variables to identify previously unknown knowledge or clusters [24]. This learning refers to the possibility of understanding and organizing data to decide the potential response. The lack of context in Unsupervised Learning for the learning algorithm can often be useful, as it allows the algorithm to revisit patterns that have previously not been considered [148]. Unsupervised Learning algorithms are powerful in detecting patterns in multidimensional data that cannot be detected by humans. Unsupervised Learning appears to be more complex, as the analysis does not have a specific goal as do classification tasks in Supervised Learning [156]. The most common tasks in Unsupervised Learning involve Association Rule and clustering [150] (See Figure 5).

Clustering refers to a wide range of techniques used to identify subgroups or clusters within a dataset using a similarity measure [89]. This allows observations to be divided into separate groups, such that each group comprises similar observations. This technique can therefore detect

outliers, i.e. the points falling outside of the clusters [178]. Since the clustering task is subjective, there are several methods that can be used to accomplish this goal. Each approach applies a different set of guidelines to describe the 'similarity' between data points [156]. Clustering can be used in multiple real applications such as customer segmentation [48], targeted marketing [16], and recommender systems [132], among others. There are many clustering techniques, but the most widely used algorithms in cluster analysis are k-means, hierarchical clustering, and expectation maximization (EM) [156].

Another common Unsupervised Learning task is Association Rule mining which involves exploring sales transactions to produce hidden correlations among items in the format of 'If-Then' rules. The Association Rule mining algorithm usually relies on two main thresholds: minimum support and minimum confidence. The minimum support is used to check which items are eligible to be part of the rules by comparing their frequencies in the transactional database. An item with a frequency above the minimum support threshold is called a large / frequent item while any items below the minimum support threshold will be ignored. Once the complete large items are found, the Association Rule mining algorithm then evaluates the confidence value to convert those items into rules. Association rules are crucial for making planning and marketing decisions such as items that go on sale, item shelving, promotion strategies, and others. The typical Association Rule applications include basket data analysis [76], cross-marketing [17], catalog design [156], and loss-leader analysis [83]. Generally, Apriori [7], Frequent Pattern Growth [56], dEclat [187], CHARM [186], LCM [173], Map Reduce ARM [25], and Apriori-Feed Forward [7] are common Association Rule mining algorithms.

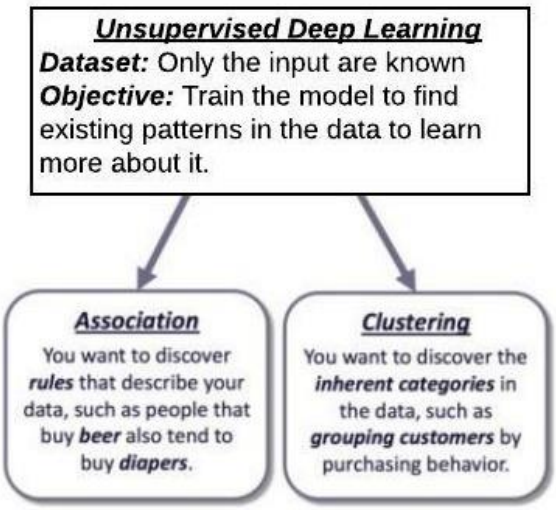

Figure 5: Unsupervised Learning [69]

2.2.3 Supervised Learning Lifecycle

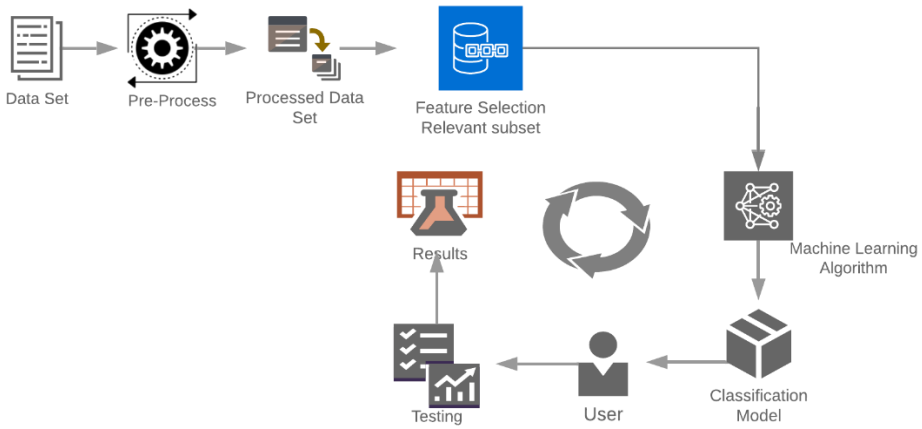

Figure 6: Learning Lifecycle

The Supervised Learning lifecycle shown in Figure 6 consists of preprocessing the input data, applying classification algorithms on the processed data, building classification models, and evaluating the models against test data. The underlying goal of the process is to extract knowledge from raw data by using intelligent algorithms, along with the required pre-processing and transformation of data [82]. The overall process involves repeatedly applying the following steps:

2.2.3.1 Data Collection and Understanding

Data collection is the process by which data is gathered from numerous external and internal sources for analysis. Often, there will be a specific scope, task, and purpose for data analysis based on the need of the business or the user; these are used to map the data into specific AI and machine learning techniques [8]. For example, if the purpose is to predict a value of a variable and that value is discrete, then classification algorithms seem more suitable. However, if the task is to reveal certain correlations in the form of easy-to-interpret knowledge, then Association Rule mining can be an appropriate approach. Collected data must be understood and treated in a manner that makes sense for the specific purpose and tasks [8].

Prior to performing any data manipulation and pre-processing operations on the collected data, the user or the data engineer must understand the data. Data understanding in the context of data statistics and features is essential to ensure that methods selected for learning are appropriate and compatible. More importantly, it gives the user an in-depth understanding of the data characteristics, feature types, machine learning requirements in terms of types of features needed, and any requirement for data transformation such as normalization, discretization, and others if any. By using the scope of the work and purpose, the user will be able to identify to a certain degree any possible features to be collected from the external and internal sources. Often when users are faced with big data with a large number of features, the process of determining which features to collect is difficult to manage [122]. Data collection helps users and data engineers to capture the needed features and instances for analysis using the scope and purpose of the analysis [82].

2.2.3.2 Data Pre-processing

Data pre-processing is a phase that involves data preparation and transformation into a form suitable for modelling the task under investigation using machine learning [192]. Data pre-processing is a key step since the quality of the data when modelled directly impacts the outcome (knowledge) goodness and consequently the reliability of the outcome and its use [8]. Pre-processing aims to reduce data size, find discretized features, normalize features, remove noise and outliers, while handling missing data, dealing with data inconsistences, smoothing data, data balancing, and other tasks [122]. Common data pre-processing operations are discussed below.

A. Data Cleansing

Data cleaning is the process of identifying and deleting (or fixing) incorrect data from a dataset, i.e. identifying inconsistencies, redundancies, missing or obsolete parts of the dataset, and then repairing, remodeling, or eliminating data to gain reliable information [39]. Incomplete data is an inevitable problem when dealing with any of the real-world data sources. In general, certain crucial considerations need to be addressed when cleaning unknown feature values, i.e. causes for missing data (input errors, system errors, overlooked or lost data, some features are not applicable for a given instance, etc.) [82]. Therefore, missing data, noisy data, and data inconsistencies need to be treated carefully before learning steps are activated. There are various techniques to cleanse the data and to deal with missing values such as deleting instances with missing values, imputing using average or median, using most common value in the feature, zero/ constant, etc. [192]. Noise data can be corrected using the binning method, clustering, sorting, and regression, etc. [8].

B. Data integration

Data Integration is the process of combining data from multiple sources into a unified view to support the data analyst in making smarter business decisions [192]. Schema integration and

redundancy are two major issues of data integration [156]. The key benefits of data integration are enhancing system collaboration, saving time, increasing performance, minimizing errors, and delivering more valuable knowledge [156]. There are several data integration techniques including, Manual Integration or Common User Interface, Application-Based Integration, Middleware Data Integration, Uniform Data Access or Virtual Integration, Common Data Storage or Physical Data Integration [128].

C. Data Transformation

Data transformation is a process of converting data format, structure, or values to make the learning process more efficient. Data transformation may involve a variety of tasks such as change of data type, data cleaning by eliminating null or obsolete data, data enrichment, or aggregation, depending on project goals [156]. Smoothing [120], Aggregation [145], Generalization [49], and Normalization [153] are a few methods to perform transformation. Smoothing allows binning, clustering, and regression techniques to eliminate the noise data [120]. Aggregation is the method by which statistical measures such as mean, median, and variance are applied to summarize data [145]. Generalization involves replacing lower level data (primitive) by higher level using hierarchical principles [49]. Normalization is the process of adjusting data into specific ranges such as 0 to 1 or -1 to 1 [153].

D. ***Discretization***

Discretization is the process through which continuous variables, models, or features are converted into a discrete form [98]. This is accomplished by generating a series of adjacent intervals (or bins) which extend beyond the range of variable/ model/ function. Overall, discretization algorithms can be divided into two depending on if they function as supervised (uses class label) or unsupervised discretization (top down or bottom up discretization) [8].

Binning [67], histogram [188], entropy-based [34], and clustering [102] are a few discretization techniques that can be easily performed. Binning, histogram, and clustering are unsupervised methods whereas the popular entropy-based is a supervised method. Equal-width or equal-frequency binning can be used to discretize attributes by replacing values by mean or median. Binning and entropy-based partitioning measure the number of partitions, independent of the other features [82].

E. Dimensionality Reduction

In cases when there is a massive number of features in applications related to bioinformatics, text categorization, and medical diagnosis, reducing the number of input features in the dataset becomes vital [21]. Dimensionality reduction methods, also known as feature selection methods, reduce the search space of the learning problem (number of input features) while trying to maintain model performance [6]. This step involves finding relevant features from the original set of features in the dataset to represent the whole dataset, a) to make the training time efficient, b) to simplify the output, c) to simplify the input, and d) to eliminate useless features [79].

Examples of common approaches to feature extraction are Maximum Relevancy Minimum Redundancy (mRMR) [35], ReliefF [81], Conditional Mutual Information Maximization (CMIM) [41], Pearson Correlation Coefficient (PCC) [130], Principal Component Analysis (PCA) [129], Non-Linear Principal Component Analysis (NLPCA) [135], Independent Component Analysis (IPA) [61], and Correlation-based feature selection (CFS) [52]. Many researchers have been studying how these approaches help to improve the predictive accuracy of the classification algorithm. Hence, in Chapter Three, we discuss the feature selection problem and its related literature.

2.2.3.3 Training using a Machine Learning Algorithm

After data is pre-processed, a machine learning algorithm, such as classification or clustering, will be applied on the dataset depending on the learning purpose. Careful attention is then required to decide which algorithm and parameters are appropriate and matching the overall data processing process. The success of the selected algorithm relies on different factors such as the data characteristics, setting values of the algorithm's parameters, input features, and available computing resources among others [103].

2.2.3.4 Testing and Evaluation

The outcome of the training phase consists of the patterns or models learned. In the case of classification tasks such as credit card scoring or medical diagnosis, the model learnt is evaluated on test data to reveal its performance in terms of predictive power. Often models are measured in terms of common evaluation metrics such as accuracy, sensitivity, specificity, F-measure, training time, etc. If the model does not show the expected result, we can use a different learning algorithm or possibly tune the parameters of the original algorithm. The outcome of evaluating multiple algorithms against test data lays the groundwork for discovering which algorithms could be worth tuning on the problem [68]. There are various performance measures to validate the model, and these measurements provide a reliable score; these evaluation measures are discussed below.

2.3 Evaluation Step and Measures

Evaluation measures play a significant role in machine learning as they are not only used to compare various learning algorithms but also as goals to optimize in developing learning models on unseen data [60]. Therefore, frequent evaluation and focusing on the outcome of what was more and less effective will help identify areas for improvement and consequently help meet the goals. The information gained helps in the understanding of the model's impact for organizations to make

informed decisions. Thus, identifying performance metrics to evaluate the models derived by machine learning is vital. Metrics such as accuracy, precision, specificity, sensitivity, and others are described below [68]:

Accuracy: Measures the proportion of correctly classified observations to the total observations. In a classification task, this is the ability to predict the class label. Accuracy is recognized as the simplest measure among all the others.

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

Precision: The ratio of correctly predicted positive observations to the total predicted positive observations.

$$\text{Precision} = TP/(TP+FP) \quad (2)$$

Specificity: The ratio of correctly predicted negative observations to all of the observations in the actual class – no. It is also the true negative rate.

$$\text{Specificity: } TN / (TN+FP) \quad (3)$$

Sensitivity/ Recall: The ratio of correctly predicted positive observations to all of the observations in actual class - yes. This is the true positive rate.

$$\text{Recall} = TP/TP+FN \quad (4)$$

AUROC curve (Area Under the Receiver Operating Characteristics) depicted in Figure 7 shows the relationship between false positives and true positives. A true positive is the result when the classification algorithm predicts the positive class correctly. A false positive is the result when the algorithm predicts the positive class incorrectly. The field tests bias, the capacity of the algorithm to accurately classify the test results.

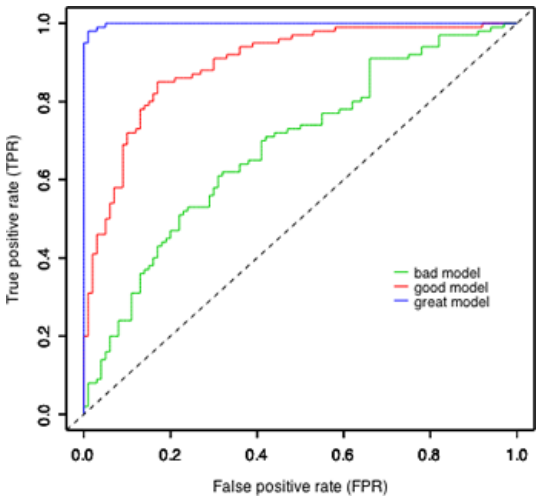

Figure 7: AUROC Curve [158]

Time: The amount of time in milliseconds (ms) taken to build the classifier model.

2.4 Common Feature Selection Approaches

Lei [87] proposed a new approach for selecting text features based on the IG and Genetic Algorithm (GA). The main feature selection techniques such as IG, generic coding, fitness function, selection, crossover, and mutation have been tested. The fitness function has been enhanced for the information filtering systems to properly understand the qualities of weight, text, and vector similarity, etc. This approach selects the feature with the frequency of items based on the information gained. It has shown that IG improved values, however it minimized the text vector dimensions and maximized text classification accuracy, i.e. result in effective feature selection method.

Khalid, Khalil, and Nasreen [79] investigated feature selection steps to minimize the effect of irrelevant, redundant, and noisy data on the classification task of biological medical data. The authors studied search strategy, subset evaluation, and stopping criteria by comparing results obtained using feature selection techniques such as Correlation Coefficient [130], mRMR [35], PCA [129], Prediction Analysis of Microarray (PAM) [32]. The comparative analysis found that

feature selection approaches that manage both redundant and irrelevant features at once, such as mRMR, are far more reliable and effective for the learning process relative to approaches that treat redundancy features and/or irrelevant features discretely.

Jenke, Peer, and Buss [63] reviewed feature selection techniques on emotion recognition from EEG signals based on 33 studies. Various feature extraction methods have been studied to choose appropriate features as well as electrode locations relying on neuroscientific observations. The study was performed on multivariate and univariate feature selection techniques. The research indicated that multivariate selection techniques such as mRMR responded slightly better than univariate techniques, often requiring fewer than 100 features on average. A total of 16 participants (seven females, nine males) aged between 21 and 32 took part in this experiment. The recorded data set for each subject contains eight trials of 30-second EEG testing for five different emotions (happy, curious, angry, sad, quiet). The authors compared the most impactful features and the electrodes that are often chosen for them. Advanced extraction methods for features such as Higher Order Crossings (HOC) [125], Higher Order Spectra (HOS) [109], and Hilbert-Huang Spectrum (HHS) [97] outperformed widely-used spectral power bands.

Zena and Gillies [189] reviewed feature selection methods to reduce the dimensionality of high-dimensional microarray cancer data. Microarrays, a biological medium to collect gene expressions, are a common source of data. Analyzing microarrays can be challenging due to two major factors, i.e. the scale of the data and complex relationships between the various genes [6]. Therefore, removal of unnecessary features will improve the output. The authors critically analyzed a few common methods for choosing significant features, i.e. Filtering, Wrapping, and Embedding, and subsequently more focus has been given to filtering where univariate and multivariate methods

were discussed and evaluated. In the next section we review common feature selection methods based on the taxonomy shown in Figure 8.

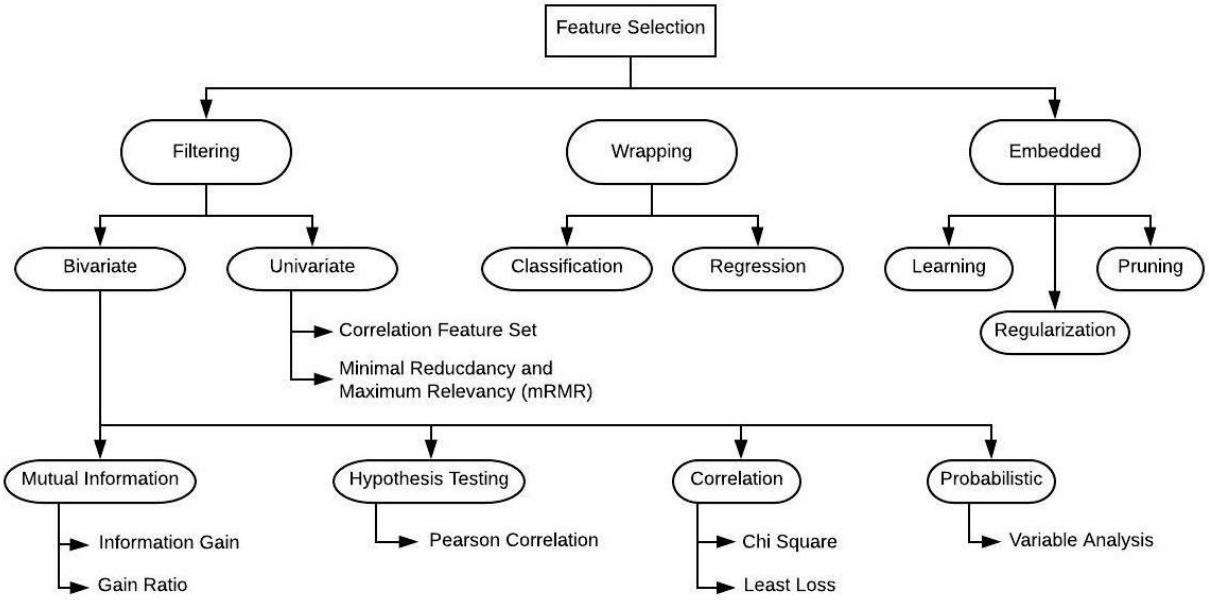

Figure 8: Taxonomy of Feature Selection Methods

2.4.1 Filtering Methods

Filtering (Figure 9), the most widely used feature selection technique, uses various relevance metrics to identify a smaller subset of features and filter out irrelevant features [42]. Once a relevance metric is chosen, a score per available feature is calculated using a mathematical model and a ranking procedure is invoked to rank features. Normally, relevancy metrics rely primarily on the features' frequencies and the correlations between the feature and the class labels available in the training dataset [45]. The process of determining relevant features does not require the involvement of a supervised learning model [60].

There are two common categories of Filtering methods: Univariate and Multivariate. Univariate techniques involve ranking each feature individually based only on its relevance to class label [5]. Thus, most of the Univariate Filtering methods, e.g. Mutual information, Variance, Fisher [41,47,105], do not consider feature redundancies, i.e. feature-to-feature correlation to determine an optimum set of features [33, 35, 74, 178]. In contrast, subset-based evaluation techniques utilize Multivariate schemes to determine the ranking considering the entire feature subset as one. Thus, although Univariate techniques ignore feature spaces/ dependencies in ranking the features, Multivariate techniques take feature spaces into consideration to identify and eliminate redundant features [111,121]. However, Univariate techniques are more efficient in terms of computational complexities. Multivariate techniques outperform Univariate techniques because they consider the mutual relationship between features, hence minimizing the presence of feature-to-feature relationships in the results set to offer a less redundant outcome [107]. In the next subsections we review common Filtering methods.

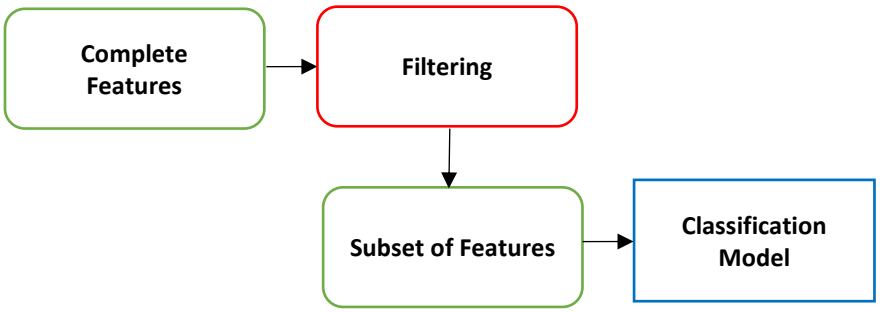

Figure 9: Filtering Method Process

2.4.1.1 Univariate Filtering Methods

Univariate Filtering methods consider the relationship between each single feature in the training dataset with the dependent variable [5]. These methods usually employ statistical tests to compute weights for each feature from the training dataset based on the feature's frequency, the feature's frequency with the class labels, and class label's frequency, among others. Univariate Filtering methods do not seek relationships between all features in the training dataset [79]. In the next subsections, we review common Univariate Filtering methods.

A. Correlation-Based Methods

Correlation is a key metric in statistics which measures the relationship between two variables [45]. The correlation coefficient is ±1 when two variables are dependent. If the features are not correlated, then the correlation coefficient is 0. Two types are typically utilized to measure the correlation of two random features; the first one uses information theory, and the second is more common and based on linear correlation [130]. As per the standard literature, for a pair of variables (X, Y), the linear Pearson correlation coefficient 'r' is given by:

$$r = \frac{\sum(X_i - \bar{X}_i)(Y_i - \bar{Y}_i)}{\sqrt{(X_i - \bar{X}_i)^2}\sqrt{(Y_i - \bar{Y}_i)^2}} \quad (2.1)$$

where X and Y are the two random quantitative features. The magnitude of r pinpoints to the correlation strength. When r is negative, this indicates a negative association and when r is positive then there is a positive association. Other correlation-based Filtering methods that are Multivariate are discussed in Section 3.2.

Fisher-Score (F-Score)

F-score is a Univariate Filtering method mostly used in binary classification problems to determine the optimum feature subsets using the distances between the features and the class labels. F-score

is derived by dividing the sample mean of each class per feature by their variances. Therefore, F-Score presents a list of features that are ranked based on their importance within the class label. Features with high F-score are listed as the top-ranked features and have the ability to predict the class variable [69]. However, since F-Score weighs each feature individually, it completely ignores the features that indicate a lower score when considered individually but a higher score with the class label when combined with another feature [47,93].

$$FiR_i = \frac{\left|\bar{x}_i^{(0)} - \bar{x}_i^{(1)}\right|}{\sqrt{var(x_i)^{(0)} + var(x_i)^{(1)}}} \qquad (2.2)$$

B. Mutual Information

Information Gain (IG)

IG uses the amount of information each feature holds to predict the class label as the measure of goodness. The purer the outcome of a data split using a feature and the class labels, the higher the gain associated with that feature. IG estimates the expected decrease in Entropy when splitting the data instances using a feature [137]. Entropy estimates the uncertainty between features [41]. IG calculates the amount of information between available features and class labels, and then produces scores for the available features. Shown below are the mathematical equations for IG [138].

$$\text{IG}(T, F) = \text{Entropy}(T) - \sum ((|T_f| / |T|) * \text{Entropy}(T_f)) \qquad (2.3)$$

where

$$\text{Entropy}(T) = \sum -P_{cl} \log_2 P_{cl} \qquad (2.4)$$

where P_v = probability that T belongs to class cl.

T_f = subset of T for which F has value f

$|T_f|$ = number of examples in T_f, and $|T|$ = Size of T.

Maximal Information Coefficient (MIC)

MIC estimates the linear or non-linear correlations between variables and seeks other hidden relationships such as superposition of functions. Both maximum information and grid partition methods can be used in calculating MIC [157]. The MIC score basically ranges from 0-1 where 1 indicates high dependency between two considered variables and 0 indicates that the considered variables are purely independent. Although MIC performs well on both numerical and categorical data, it does not perform as well as other dynamic feature selection algorithms such as T-Test [45,64].

$$MIC(D) = \max_{xy<B(n)} \{M(D)_{x,y}\} \quad (2.5)$$

Where the X/Y grid size < B(n), D is given a fixed set and X and Y represents various column and row values of the maximum grid. According to Sunab, Liab, Daiab, Songa, and Lang (2018) [157], MCI performs well when setting $B(n) = n^{0.6}$.

Gain Ratio (GR)

One of the issues associated with the IG Filtering method is that it is biased toward features that are linked with more values [41,78]. To deal with this bias, GR was proposed. GR is a normalized form of IG which is computed by dividing the IG with the Entropy of the feature in regard to the class label (See Equation 2.6). Since IG is employed to build tree-based classifiers, GR measures the IG ratio of each node in the decision tree where non-terminal nodes signify the tests on one or

more feature, and terminal nodes signify outcomes of the decision [74]. Basically, this Filtering method considers both the size and number of the branches in the decision tree and determines the optimum feature subsets [74]. GR normalizes and balances IG by dividing it by Entropy of X (H(X) as shown in Equation (2.7) when class 'Y' is to be predicted. The value of GR typically ranges from 0-1, where 1 indicates 'X' and 'Y' which are dependent on each other. Hence 'X' has the ability to predict 'Y' and 0 indicates that 'X' and 'Y' are completely independent variables thus, 'X' does not have the ability to predict 'Y'. One drawback of GR is that it does not take feature dependencies into consideration when determining the weights of the features [161].

$$GR = \frac{IG}{IntrinsicInfo(S,A)} \quad (2.6)$$

$$IntrinsicInfo(S,A) = -\sum \frac{S_i}{S} log_2 \frac{S_i}{S} \quad (2.7)$$

Where IG is the information gain and *IntrinsicInfo* is the Entropy of attribute 'A' over a set of examples 'S'.

Symmetrical Uncertainty (SU)

SU is another Filtering method that uses the merit of each feature to determine the fitness of features in predicting the class label [116]. It symmetrically compensates for the bias of MI that occurs due to a large number of different values and presents a normalized score within the range of 0-1 [133]. A score of 1 indicates higher merit with regards to the class label, whereas 0 indicates that the considered features do not have any merit. As shown in Equation (2.8), SU between features X and Y can be obtained by dividing twice MI, after observing Y from the Entropy of the features [155].

$$SU(A, B) = \frac{2 \times IG(A|B)}{E(A)+E(B)} \quad (2.8)$$

Where $IG(A|B)$ the gain of information of A after knowing Y; E(A) and E(B) are the Entropy values of A and B respectively.

ReliefF

ReliefF computes the scores (proxy statistics estimates) of each available feature with the class label in the training dataset and then ranks features based on computed scores [129]. The original ReliefF Filtering method was unable to deal with missing values and only considered datasets with binary classification (two class labels) [174]. The mathematical notations for the ReliefF Filtering method are given below and the complete pseudocode of the original algorithm (Relief) is shown in Figure 10.

The Relief algorithm iterates over *m* random instances (set up by the end-user) in the target training dataset (*Ri*) without replacement. During each iteration, Ri and the vector W of the feature score are amended according to W[A] as shown in Figure 10. W[A] denotes the differences between the neighboring data instances and the target instances in feature value. For each target instance, the algorithm determines two neighbors, i.e. hit (H)- the one of similar class label, and miss (M)- the one with different class label. Lastly, the algorithm amends A's weight in W when the feature values of Ri do not match those of the nearest H or nearest M. Therefore, any feature with a value different from that of Ri and M triggers the W[A] to increase as this is an indication of informative value.

$$W[A] = W[A] - \frac{\left(diff\frac{A,R_i,H}{m}\right)}{\left(diff\frac{A,R_i,M}{m}\right)} \qquad (2.9)$$

Where,

W[A]= feature weights

A = number of features

m = number of random training instances out of 'n' number of training instances used to update W

R_i = randomly selected target instance

H/M = nearest hit and nearest miss

Algorithm 1 Pseudo-code for the original Relief algorithm

Require: for each training instance a vector of feature values and the class value
  $n \leftarrow$ number of training instances
  $a \leftarrow$ number of features (i.e. attributes)
  Parameter: $m \leftarrow$ number of random training instances out of n used to update W

  initialize all feature weights $W[A] := 0.0$
  for i:=1 **to** m **do**
    randomly select a 'target' instance R_i
    find a nearest hit 'H' and nearest miss 'M' (instances)
    for A:= 1 **to** a **do**
      $W[A] := W[A] - diff(A, R_i, H)/m + diff(A, R_i, M)/m$
    end for
  end for
  return the vector W of feature scores that estimate the quality of features

Figure 10: Original ReliefF Algorithm Pseudocode [152]

C. Probabilistic Methods

Chi Square Testing (CST)

CST uses the difference between the observed frequencies and expected frequencies of each categorical variable and the class labels to determine the association among variables [66]. To use this method for feature selection, the variables considered should be categorical, sampled independently, and the value of expected frequency should be greater than 5 [98].

A higher CST value indicates close dependency between the variable and the class label. Therefore, features with higher CST values are selected for model-fitting purposes. Below is the formula for the CST [98].

$$X^2 = \frac{(O-E)^2}{E} \quad (2.10)$$

Where

O denotes the Observed Frequency, and *E* denotes the Expected frequency, for the considered features' values. CST has been applied widely including text mining [12,113].

Least Loss (L^2)

L^2 is a new Filtering method that uses the distance between observed and expected frequencies of the variables with the class labels to rank the features [168]. L^2 values are computed based on the expected and observed frequencies of the features; each feature is ranked in ascending order based on the computed L^2 values. This method identifies and removes redundant features without having to alter the model construction phase [168]. The authors evaluated the L^2 Filtering method on a large number of datasets downloaded from the UCI data repository using the Naïve Bayes algorithm. The results produced were compared with the IG and CST methods using the same classification algorithm and they pinpointed that L^2 can reduce data dimensionality and maintain stable models in terms of accuracy. The mathematical notation of the L^2 method is given below.

$$L^2(Y,X) = \sum_{i,j}[P(Y_i,X_j) - P(Y_i)P(X_j)]^2 \quad (2.11)$$

Where,

X = Independent feature class

Y = Class label

$P(Y_i)$ = Theoretical marginal distributions of Y

$P(X_j)$ = Theoretical marginal distributions of X

$P(Y_i, X_j)$ = Theoretical joint probability distributions of X and Y

Variable Analysis (VA)

Kamalov and Thabtah (2017) [72] developed a Filtering method to reduce the results variations of the IG and CST methods. The authors presented the VA method which uses a vector of scores of both IG and CST results, normalizes the scores, and then computes the vector magnitude (V_score). The V_score along with Correlation Feature Set (CFS) method's results [72,52] are then combined to produce a new metric for filtering out relevant features. The V_score for a feature X is defined in Equation (2.12) and can be represented as the square root of the sum of the square of its IG and CST results (its coordinates) as in Equation (2.13).

$$V_a = \begin{pmatrix} IG_x \\ CST_x \end{pmatrix} \quad (2.12)$$

$$|V_a| = \sqrt{(IG)^2 + (TST)^2} \quad (2.13)$$

The VA Filtering method utlizes CFS results to compute the V_score in its subset and then discards features with V_scores 50% less than the top V_score in the CFS feature set. Experimental results using 15 datasets from the UCI data repository reveal that the Va Filtering method reduces the number of chosen features for the majority of the datasets considered when contrasted with IG and CST results.

Distinguishing Feature Selector (DFS)

DFS ranks features based on their probability of occurrence in number of classes. If a feature occurs rarely in a single class and does not occur in the other classes, that type of feature is scored low and considered irrelevant. Features that occur frequently in a single class and are not present in the other classes are ranked high since they are highly distinguishing. Hence, features that are

ranked high are considered to have merits and are used in the predictive models. The DFS formula is given below:

$$\text{DFS(t)} = \sum_{j=1}^{M} \frac{P(C_j|t|)}{P(|\bar{t}|C_j)+P(|\text{t}|\bar{C}_j)+1} \quad (14)$$

Where,

M = number of classes

$P(C_j)$ = Probability of j^{th} class

$P(|\bar{t}|C_j)$ = Probability of absence of term 't' when class 'C_j' is given

$P(|\text{t}|\bar{C}_j)$ = Feature likelihood when classes other than 'C' are given

2.4.1.2 Multivariate Filtering Methods

Most of the available Filtering methods consider the univariate relationship between two variables and do not consider correlation among sets of features during feature selection. Evaluating combinations of feature-to-feature relationships is vital in reducing the redundancy in the optimal features subset to discard similar features from playing any role in the learning process [19,26,161]. Therefore, the removal of the redundant features will minimize the model's overfitting and improve training time efficiency for the classification algorithm since fewer features that are dissimilar will be processed. In the next sub-sections, we review common multivariate Filtering methods that deal with the issue of reducing feature-to-feature correlations during the feature selection process.

A. Correlation-Based Methods

A heuristic evaluation function based on correlations is used in the Correlation Feature Set (CFS) Filtering method to rank features. CFS selects features that are highly correlated with the class

label, but independent from each other [52]. It ignores the irrelevant features that demonstrate a lower correlation value [123]. The CSF criterion is computed by the following mathematical equations:

$$\text{Merit}_s = \frac{k\overline{w_{cf}}}{\sqrt{k+k(k-1)\overline{w_{ff}}}} \qquad (2.15)$$

Where Merit is the hypothesis of a feature set S containing k subset of features in Equation (2.15). $\overline{\omega cf}$ is the feature-class correlation average, and $\overline{\omega ff}$ the feature-feature inter-correlation average.

The correlation between two subsets *i* and *j*, w_{ij} is computed per Equation (2.16)

$$\text{Merit}_s = \frac{\sum(i-\bar{\imath})\,(j-\bar{\jmath})}{\sqrt{[\sum(i-\bar{\imath})^2]\,[\sum(j-\bar{\jmath})^2]}} \qquad (2.16)$$

Where *i* is the instance set that belong to the feature and *j* is the instances set that belong to the class label, and $\bar{\imath}$ and $\bar{\jmath}$ are the mean values of the class and features, respectively.

One of CFS's limitations is that it usually derives an outcome that contains features with relatively low IG and V-scores [19].

Fast Correlation-Based Filter (FCBF)

The study by Yu and Liu (2004) [184] is one such attempt to address the need to incorporate a redundant feature analysis process; considering feature relevancy only is insufficient to determine the best feature subsets. The authors introduced a novel mechanism called Fast Correlation-Based Filter (FCBF), which involves first selecting relevant features and then through a relevance and redundancy analysis, identifying predominant features from the selected set to enhance the feature selection process. For each available feature in the input dataset, FCBF computes the Symmetrical

Uncertainty (SU) (The IG of a feature F given a class C divided by the sum of entropies of A and C together) (See Equation 2.8), then ranks the SU scores to eliminate redundant features.

Yu and Liu (2004) [184] applied FCBF on gene expression microarray data analysis to show the merit of eliminating redundant features to accurately classify the diseases or phenotypes.

Minimal-Redundancy and Maximal-Relevancy (mRMR)

mRMR involves selecting the feature subsets with maximum relevancy to the class label and minimum redundancy among the features [35]. In doing so, mRMR considers the tradeoff between redundancy and relevance features, using F-Statistics to determine the relevancy of the features with the class labels, and Person Correlation to determine the feature redundancy as shown in Equation 2.17 [191,111].

$$f^{mRmR}(X_i) = I\,(Y, X_i) - \frac{1}{|s|^2}\sum_{X_s \varepsilon S} I\,(X_s, X_i) \qquad (2.17)$$

Where,

Y = Class label

S = Set of selected features

$|S|$ = Number of features

I = Mutual Information

X_i = Features that are not selected now

$X_s \varepsilon S$ = One feature out of the feature set 'S'

Maximum Marginal Relevance (MMR)

MMR is a divergence-based Filtering method that is mostly used in text classification and document retrieval problems for re-ranking the feature subsets. In a text summarization problem, MMR is used to keep the feature redundancies of the results to a minimum while maintaining the query relevance of the already ranked documents to increase the results' diversity. MMR considers the similarities of the key phrases along with the similarities of already selected phrases [191]. The mathematical notation of MMR is given in Equation 2.18.

$$\text{MMR} = \arg \max_{\substack{D_i \varepsilon R \\ S}} [\lambda \, Sim_1(D_i, Q) - (1 - \lambda) \max_{D_j \varepsilon S} [\, Sim_2(D_i, D_j)] \quad (2.18)$$

Where,

$\bar{D}_\iota$= Documents in the data collection 'C'

Q = Query

R= Relevant documents in collection 'C'

S = Current results set

There is other multivariate feature selection for Unsupervised Learning tasks specifically clustering problems such as Localized Feature Selection Based on Scatter Separability (LFSBSS), Multi-Cluster Feature Selection (MCFS), and Feature weighting Kmeans [89,114]. However, these methods are out of the scope of this research. Table 2.1 depicts a summary of the Filtering methods which we have discussed.

Table 2.1: Common Filtering Methods of Feature Selection

Name	Type	Mathematical Equation	Relevance Measure	Strength	Weakness
F-score	Univariate	$FiR_i = \frac{\left\|\bar{x}_i^{(0)} - \bar{x}_i^{(1)}\right\|}{\sqrt{var(x_i)^{(0)} + var(x_i)^{(1}}}$	Sample mean	Simple and straightforward to define correlation and easy to implement	It cannot consider the effect of combined features and only considers features-class correlation individually
IG	Univariate	IG (T, F) = Entropy (T) – $\sum$ ((| T_f | / | T |) * Entropy (T_f))	Mutual information	It shows the worthiness of a feature by using the class information as an efficient measure	Biased results for features with large numbers of distinguished values It only considers class-feature correlations
Pearson Correlation	Univariate	$r = \frac{\sum(X_i-\bar{X}_i)(Y_i-\bar{Y}_i)}{\sqrt{(X_i-\bar{X}_i)^2}\sqrt{(Y_i-\bar{Y}_i)^2}}$	Linear correlation in statistics	Simple to compute and it shows the linearity of a relationship	Works only on continuous features and does not consider feature redundancy
MIC	Univariate	$MIC(D) = \max_{xy<B(n)}\{M(D)_{x,y}\}$	Correlation between variables	It measures the non-linear and linear correlations of variables	It does not consider feature redundancy issue
GR	Univariate	GR = $\frac{IG}{IntrinsicInfo(S,A)}$	Mutual information	It removes the bias of favoring features with a large number of distinct values inherited from mutual information	It prefers features with limited values and does not consider redundant features
SU	Univariate	SU(A, B) = $\frac{2\times IG(A\|B)}{E(A)+E(B)}$	Mutual information	It overcomes the bias of the mutual information method by not favoring features with a large number of distinct values	It prefers features with limited values and does not consider redundant features
ReliefF	Multivariate	W[A] = W[A] - $\frac{\left(diff\frac{A,R_i,H}{m}\right)}{\left(diff\frac{A,R_i,M}{m}\right)}$	Proxy statistics	For each instance, it repeatedly allocates the largest score to the feature which discriminates it from the neighboring instances of a different class	It is sensitive to noisy data and does not reduce feature-feature redundancy
CST	Univariate	$X^2 = \frac{(O-E)^2}{E}$	Observed and expected frequencies	Straightforward calculations and robust in terms data distribution	Sensitive to dataset set size and cannot establish casual correlation between two features

L^2	Univariate	$L^2(Y,X) = \sum_{i,j}[P(Y_i,X_j) - P(Y_i)P(X_j)]^2$	Distance between the observed and expected frequencies	It simplifies the hypothesis testing of CST method by providing a more generic hypothesis	Does not consider feature-feature correlations and relies on a cut-off score
VA	Univariate	$\|V_a\| = \sqrt{(IG)^2 + (TST)^2}$	Vector of scores of both IG and CST results	It defines a new vector based on the aggregation of the scores of multiple filtering methods to reduce result variability	Does not consider feature-feature correlations
GI	Univariate	$Gini = 1 - \sum_{i=1}^{n}(p_i)^2$	Statistically significant measure	Straightforward to compute	It does not consider feature redundancy and it does not consider the changes in a data sample
OR	Univariate	$OddsRatio(F) = \log\frac{odds(W\|C_1)}{odds(W\|C_2)} = \log\frac{P(W\|C_1)(1-P(W\|}{(1-P(W\|C_1))P(W\|}$	Probabilities of two features appearing and non-appearing together in the dataset	It is a versatile method besides one can easily present the sampling distribution of the odds ratio	The odds ratio is like the relative risk if the target class is rare
ECE	Univariate	$ECE(t_k) = p(t_k)\sum_{i=1}^{\|c\|}p(c_i\|t_k)\log\frac{p(c}{p($	Mutual information	It can be used as a loss function for measuring the performance of classification problems	It only considers the correlation between the feature and the class label
DFS	Univariate	DFS(t) = $\sum_{j=1}^{M}\frac{P(C_j\|t\|)}{P(\|\bar{t}\|C_j)+P(\|t\|\bar{C}_j)+1}$	Probability of occurrences	It chooses unique features while discarding uninformative features	It has been evaluated mainly in text-related applications, thus highly frequent unrelated text can be accounted for during the computations
FCBF	Multivariate	$f^{mRmR}(X_i) = I(Y,X_i) - \frac{1}{\|s\|^2}\sum_{X_s\varepsilon S} I(X_s,X_i)$	Symmetrical Uncertainty	It removes pairwise based on symmetrical uncertainty scores to enhance efficiency for datasets with large numbers of features	When two redundant features, A and B, have good correlation with another selected feature (V), then one of the two features (A or B, with the larger rank) will be chosen and the other one will be discarded. In this case, FCBF is not considering which of the two features better contributes to the V based on the relevancy. FCBF deals with continuous features, therefore discretization is done on the features prior to feature selection

mRMR	Multivariate	$f^{mRmR}(X_i) = I(Y, X_i) - \frac{1}{\lvert S\rvert^2}\sum_{X_s \varepsilon S} I(X_s, X_i)$	F-Statistics and a measure of correlation	It detects feature-feature correlations but in a bivariate manner	The high sensitivity of redundancy and relevance measures to outliers in the data. In addition, it does not account for the coherence in combinations of features, i.e. how two or more features work together with the class label.
MMR	Multivariate	MMR = arg $max_{\frac{D_i \varepsilon R}{S}}[\lambda\, Sim_1(D_i, Q) - (1 - \lambda)\, max_{D_j \varepsilon S}[Sim_2(D_i, D_j)]$	Feature similarities	It considers minimizing redundancy of features while sustaining relevance	It requires parameter tuning for sustaining good results
CFS	Multivariate	Merit$_s$ = $\frac{k\overline{w_{cf}}}{\sqrt{k+k(k-1)\overline{w_{ff}}}}$	Symmetrical uncertainty and specific heuristic	It ignores the irrelevant features that demonstrate a lower correlation value	Does not consider class labels that are continuous

2.4.2 Wrapping Approach

Wrapper approach in feature selection is mainly associated with Supervised Learning algorithms where all possible combinations of features with the class label in the training dataset are evaluated to determine the smallest optimal subset of features. Unlike the Filtering approach, Wrapper methods define the relevance of a subset of features based on the ability of that subset in predicting the class labels [100]. They also take both feature correlations and feature redundancies into consideration to evaluate the optimum feature subsets [175]. Therefore, the Wrapper method uses a classification algorithm to determine the quality and accuracy of the selected feature subsets [19]. Depending on the type of data used, the classification data processing is repeated on the data until the best feature subsets are obtained. However, although the Wrapper method is better than the Filtering method in terms of optimizing the performance of the resulting model, it is slower and computationally expensive [19,139,140]. Moreover, it has an increased risk of model overfitting and requires a lot of time and resources as it seeks optimum feature subsets from a massive space of dimensionality [73]. Thus, an independent validation data sample is needed to evaluate the selected subset of features for generalization. Figure 11 shows the lifecycle of a Wrapper.

There are several literatures by various scholars which address the usefulness and effectiveness of the various Wrapper methods in feature selection and classification [100,174]. Some of them utilize those methods individually or in combination to produce new feature selection algorithms that outperform the conventional Wrapper methods by overcoming the weaknesses associated with the existing methods. Taradeha et al. (2019) [162] proposed HGSA (Hybrid Gravitational Search Algorithm), a Wrapper-based feature selection algorithm that utilizes the Gravitational Search Algorithm (GSA) [141] with Crossover and Mutation evolutionary operators as the search strategy, and using the k-nearest neighbor algorithm (kNN) [59] and decision tree [137] as classifiers. The primary purpose of mutation evolutionary operators here is to maximize the algorithm performance. To validate the proposed mechanism, experiments were carried out using 18 different datasets obtained from the UCI data repository [94]. The evaluation results showed that HGSA achieved superior performance with 88% and 83% accuracies when used with decision tree and kNN classifiers, respectively.

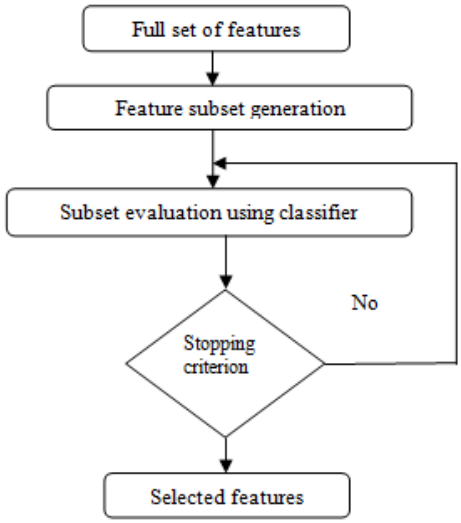

Figure 11: Wrapper Approach [127]

Wang, Khoshgoftaar, and Napolitano [175] focused on Wrapper-based feature selection approaches using real-world software metrics to assess the performance of the classification model. The relationship between internal learner and external learner were measured using four versions of telecommunications data containing 42 software metrics along with five machine learning algorithms, i.e. Naïve Bayes (NB) [38], Perceptron Multilayer (MLP) [143], K-nearest neighbors (KNN) [59], Support Vector Machine (SVM) [30], and Logistic Regression (LR) [171]. The result showed (1) the performance is rare when internal and external learner match; (2) NB shows the best performance within the Wrapper; (3) LR is often the best learner for classification models irrespective of the learner within the Wrapper.

Zena and Gillies [189] presented Wrapper-based feature selection using two categories i.e. Randomized and Deterministic Wrapper for breast cancer prediction. The authors used SVM [30] to distinguish between cancerous and non-cancerous breast tumors—SVM reported very precise results. Gradient-based leave-one-out gene selection (GLGS) [46] and Leave-one-out calculation sequential forward selection (LOOCSFS) [172] are two approaches performed on SVM, whereby GLGS was found to be better than LOOCSFS. The same data was trained on the randomized Wrapper method and genetic algorithms (GA) along with simulated annealing. It was identified that like all other Wrappers, randomized algorithms occupy more CPU time and more memory to run.

Xue, Yao, and Wu [180] presented a Wrapper-based ensemble feature selection method called hybrid Genetic Algorithm (GA)- and Extreme Learning Machine (ELM)-, based on the HGEFS algorithm, to improve accuracy. This proposed model adopts GA [87] and ELM [14] as search mechanisms and a modified extreme learning machine called EM-ELM [40] as the classifier to identify the optimal feature subsets. This method utilizes the ensemble approach to further stabilize

and improve the performance. Few sets of candidate features are chosen based on the computed accuracies and the smallest feature subset is identified using the EM-ELM output weight. Several UCI datasets and two microarray datasets are used to test the effectiveness of HGEFS along with other machine learning algorithms. The results demonstrated HGEFS to be a competitive and robust Wrapping method in terms of the derived models' accuracy.

2.4.3. Embedded Methods

Distinct to both Filter and Wrapping methods, the Embedded method has an inbuilt mechanism to perform both classification and model construction tasks together (see Figure 12). The techniques used to evaluate the optimal feature subset depend on the classification algorithm given. Embedded methods overcome some of the computational complexities associated with the Wrapper method by performing both feature selection and model construction together [93]. L1 (LASSO) [192] regularization and decision tree (DT) [138] with linear classifiers like support vector machine (SVM) are examples of Embedded methods. For a decision tree, the common Embedded method involves selecting a feature at every stage of the tree growing process and grouping features into smaller subsets.

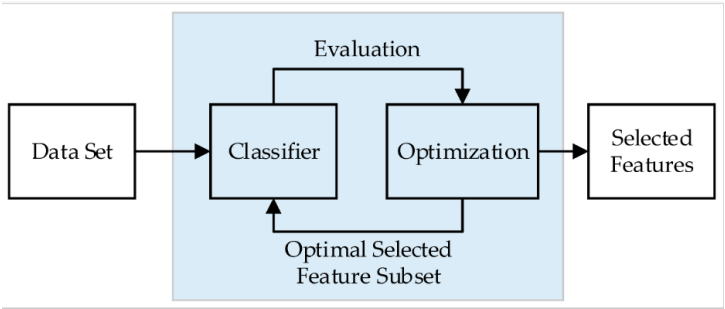

Figure 12: Embedded Approach [147]

Peng and Xu (2013) [131] developed a local information-based Embedded method for regression analysis to address drawbacks such as high computational complexities, algorithm implementation

difficulties, high computational cost, and convergence. Incorporating local data information helps to weight and rank each feature using their least squares loss value. The L1-regularization method was used to further enhance the performance by improving the feature rankings derived from different datasets. To evaluate the model's performance, it was applied on three different synthetic datasets and ten UCI datasets and the generated results were compared against other feature selection methods such as Neighbor-based feature selection (NNFS) [118], Feature Vector Machine (FVM), and mRMR [35]. The results suggested that the regression analysis-based Embedded method performs well on regression problems and data with high dimensionality and convergence.

Hernandez et al. (2007) [57] developed a machine learning algorithm to help molecular cancer diagnosis, that presented an Embedded-based feature selection approach for gene selection and classification. The proposed approach consisted of two main phases. The first phase involved reducing the feature space through conventional pre-selection of the genes using any Filtering criteria. The second phase involved further reducing of the feature space through identifying the predictive gene subsets. The fitness of the identified smaller gene subsets was then measured through an SVM classifier. The method was trained over three datasets consisting of gene expression data pertaining to acute lymphoblastic leukemia (ALL), colon cancer, and lymphoma using SVM-based selection algorithms [51]. The comparison showed that the proposed method performs more favorably on the gene expression data than the reference methods.

Discovering bio-makers, characteristics of genes or molecules that constitute a certain illness or a disease, is a hot topic that has gained the attention of many scholars. A study by Abeel, Helleputte, Peer, Dupont, and Saeys (2009) [5] focused on discovering an efficient feature selection technique for microarray data to identify potential biomarkers. Since most of the available bio-maker

selection algorithms produce different results when applied on the same dataset due to different parameter settings, this problem can negatively impact the selection of features. Therefore, the authors proposed a general experimental system based on ensemble learning with an SVM classifier to be integrated into the biomarker identification process to increase robustness. The system was applied on four datasets of cancer diagnosis microarray data obtained from previous literature to evaluate the stability and performance of the established method in gene selection. The results discovered that the stability of RFE tends to diminish when the number of selected features decreases, whereas the proposed ensemble method tends to have an enhanced stability.

Min and Fangfang, (2010) [112] proposed a Filter-Wrapper-based hybrid feature selection method called FWHM (Filter-Wrapper Hybrid Method) to enhance the feature selection efficiency. The proposed method has two basic phases: Filter and Wrapper. The first phase involves ranking the feature using six different criteria namely Correlative Family Selection, ReliefF, Class Separability, Mahalanobis Distance, Multivariate Correlation Coefficient, and Mutual Information. The second phase involves deciding on the final feature subset using improved clonal selection algorithms such as the Adaptive Genetic Algorithm (AGA) [193], Chaotic Binary Particle Swarm Optimization (CBPSO) [31], Restrictive Multi-classes Classification Model Based on Chaotic Binary Particle Swarm Optimization, and the Clonal Selection Algorithm (CSA) [193].

Hamed, Dara, and Kremer [54] investigated a new embedded feature selection called Recursive Feature Addition (RFA) to improve classification accuracy. This process begins with an empty set of features and proceeds to add more features until it meets a threshold. The authors applied the RFA algorithm on five different benchmark datasets, by comparing results obtained using other feature selection algorithms, i.e. two Wrappers: Wrapper subset evaluation [100], Classifier subset evaluation [123]; two filters: CFS subset evaluation [52], Consistency subset evaluation [160], and

one embedded: Recursive Feature Elimination (RFE) [182]. The comparative analysis found that RFA achieved superior accuracy and time efficiency compared to Filter, Wrapper, and other embedded approaches.

Maldonado and Lopez [106] investigated embedded feature selection strategy to solve the class-imbalanced issue [37] by designing two types of Support Vector Machine (SVM) namely, Cost Sensitive SVM (CS-SVM) [176] and Support Vector Data Description (SVDD) [89]. The proposed methods used a Quasi-Newton update and Armijo line search. The experiments were performed on twelve highly imbalanced microarray datasets using linear and Gaussian kernels. The generated result showed that CS-SVM achieved the highest overall predictive accuracy relative to other well-known feature selection approaches such as Fisher [77], RFE [182], BFE [105], KP-CSSVM [105], and KP-SVDD [91].

2.5 Discussion

Feature selection has some challenges including relevance definition for generating the features' scores, domain expert availability to choose which features to keep, and dealing with complex data, among others.

2.5.1 Complex Data

Complex data refers to any type of data that requires complex sequential processes to convert it into normal data, which can then be used to derive meaningful information [134]. Multimedia data, such as audio and video files, images, sound files with text and motion, 2D or 3D coordinates, mammographic data, or data files with a combination of two or more data types are some examples of complex data [115]. This data can be either static media such as texts, images, and graphs, or dynamic media involving audio, video, and speech. Domain experts often find it difficult to work with complex data as it requires a massive storage facility and advanced tools and technology to

process the information. The various forms, formats, dimensions, and the dynamic nature of complex data make it further challenging to handle [18].

For instance, consider a video file: a single video frame can comprise a large amount of data in various formats. Each frame can have different people and objectives that provide various information within seconds such as location, size of the entity, speed, etc. Extracting features from such media is a tedious task [18]. To enable the feature selection process, these complex file formats must be converted into readable and processable formats. Panchromatic images, digital images, graphics, or animations can be converted into an N x M matrix of pixels (numerical or representational forms) through image mining techniques such as compressive sampling [36], Clustering, and Association Rules [104]. Converting the original data into different formats may confuse the mathematical metrics used for ranking making the feature selection process more challenging as it can alter the important links between the candidate features and the class labels by changing the data structures. To yield better prediction outcomes, it is vital to take data structures into consideration when determining the optimal feature subsets [92].

Tang et al. (2014) [161] discussed the challenges associated with social media data in the feature selection process. There are different types of data attributes present in various social media domains. A typical social media platform has posts and followers as the main attributes with behaviors of following each other and making posts. But there is additional information and data linkages, such as who follows whom (user-user relations), who generates the posts (ser-post relations), and who likes and comments on the social media posts, which significantly differs from traditional attribute-value data. According to Tang et al. (2014) [161], this attributes and linkages data can be a challenge as well as an opportunity for the feature selection process. It can be challenging because most of the available feature selection methods choose features based on the

assumption that the attribute data values are independent and distributed identically. This is not always the case for complex information such as social media data. However, it can also be an opportunity for more investigations and research on how to use linkage data to develop advanced feature selection methods that yield better results than the conventional methods. Tang et al. (2014) [161] proposed a new feature selection mechanism called LinkedFS, that integrates attribute data values with linkage data for selecting features in social media data. The experimental results indicated that the number of selected features, percentage of labeled data, and the relationships/ linkage among the attribute values play a massive role in optimizing the feature selection performance.

A feature selection method using GR to select influential features that could be used to classify audio data was proposed by [84]. Initially the audio data is extracted into numerical format (.Wav) using the MIRToolBox tool [84]. The optimal features are then selected based on the GR of each extracted numerical feature with respect to the pulse clarity. The authors derived approximately a 90% success rate in music genre classification using the proposed method.

Saravanan (2018) [146] addressed complex data with feature selection and extraction in video data files and proposed an automated key frame-based shot method to effectively extract and select features from the key frames. In this mechanism, video files are first converted into image frames based on the image pixel values and the errors of the images are eliminated using image histograms. The error removal process uses the RGB values of each image's features to remove the redundant images and continues until the optimal image features are determined. Foschi et al. (2002) [43] presented a novel approach to extract features and patterns from images. The proposed model was intended to identify the most relevant features through the information derived from

the images and enable the users to define new features to determine undefined regions. The proposed mechanism exhibited high performance in identifying the patterns on the images.

2.5.2 Results Variations

In many situations, when Filtering methods are applied on the same dataset, different results are generated [72]. This situation arises mainly because most of the Filtering methods use different mathematical and ranking techniques to measure correlations. For instance, IG uses Entropy, and CST uses observed and expected frequency to determine the relevancy of the features to the class label. This produces different results, which in turn may create confusion for the end-user particularly when there is a massive number of features in the result set.

To deal with the issue of results variation, researchers such as 72,151,168 created feature selection methods that integrate the results offered by multiple Filtering methods to stabilize results. Kamalov and Thabtah (2017) [41] created a new Filtering method called VA which normalizes the scores obtained by CST and IG and then creates a new vector score (V_score) per feature in the dataset. Features with a V_score less than 50% of the top V_score obtained by the CFS Filtering method are discarded to refine the final feature set. Filtering methods such as VA or L2 can be seen as a promising solution to the results variation issue; nevertheless, such methods necessitate computing the scores using multiple Filtering methods, normalizing the scores, integrating the scores to obtain new global scores, and then involve ranking and a computational cost. A more realistic approach is to create a new model based on existing mathematical models of multiple Filtering methods and then use that model to produce the results thus saving on many computations.

2.5.3 Real-Time Processing

There are certain scenarios, such as medical diagnosis tools and spam detection systems, where real time processing of data is crucial. Usually in such situations data continues to accumulate and the amount of data or features to be processed is infinite and unpredictable [92]. For example, consider an online spam detection system developed to distinguish genuine e-mails from spam e-mails. The number of e-mails received per given period is always unknown and the classification task should be performed in real-time. In such situations it is not possible to wait until the data-gathering process is completed to perform the feature selection task. This issue has created the need for a feature selection method that can maintain and update feature subsets while the data is streaming [75,179].

Xuegang et al. (2016) [181] proposed online feature selection as an efficient technique to handle large data streams without the knowledge of the entire feature space. The proposed mechanism assumes that even though several features keep changing over the time, the number of data instances is fixed. The authors used online group feature selection approaches such as GFSSF (Group Feature selection with Streaming Features) and OGFS (Online Group Feature Selection) to select optimal feature subsets while eliminating the feature redundancies in data streams.

Wu et al. (2010) [179] also proposed a novel approach called Online Streaming Feature Selection (OSFS) to distinguish between redundant, weakly relevant, weakly redundant, and non-redundant features of streaming data to identify the Best Candidate Feature sets (BCF). The performance of the proposed model was compared to two well-known techniques: Grafting and Alpha investing. Further, an additional algorithm called Fast-OSFS which involves re-examining the selected features in terms of their inner redundancies and outer redundancies was introduced to improve the efficiency and performance of the original OSFS. The experimental results showed that the

proposed models perform better than the existing online streaming feature selection methods in terms of compactness and better accuracy. Zhou et al. (2005) [194] suggested Alpha-investing as an effective technique to deal with challenges associated with classification of large streams of text data while maintaining high Incremental Feature Selection (IFS) performance.

2.6 Chapter Summary

Feature selection in the context of Supervised Learning tasks is the process by which suitable subsets of features are selected based on a relevance metric to efficiently learn a model using machine learning techniques. Filtering, Wrapping and Embedded are the main approaches of feature selection that utilize different schemes to select the final subset of features. This chapter reviewed and critically analysed various feature selection methods with a focus on Filtering approach in the context of Supervised Learning. We also highlighted the learning lifecycle within machine learning and described the common learning types. More importantly, we critically analyzed Filtering methods by highlighting issues to be addressed when developing new feature selection methods. Specifically, we discussed common Filtering methods and identified their mathematical models and the numerous ways they compute scores to determine the subset of features. This chapter also highlighted challenges associated with the Filtering-based feature selection approach including result variations, data complexity, and real time data processing. The discussion showed that there are two primary factors needed for the success of feature selection: an intelligent method to assist the user by defining a cut-off that splits relevant and irrelevant features, and a normalized mathematical method to reduce the volatility of existing scores. The former factor can be dealt with by developing a cutoff procedure independent from the Filtering method with an ability to suggest the subset of features to the end user saving extensive manual processing that requires experience and care. Whereas the latter factor can be dealt with by

developing a new Filtering method that involves learning from the data, especially feature-to-feature correlations to minimize feature-to-feature redundancy.

In the next chaper, we propose a new feature selection method called CARF and a new cut-off procedure that recommends the ideal number of features to the end-user.

Chapter Three

3. The Proposed Feature Selection Method based on Class Association Rules

3.1 Introduction

In this chapter, we introduce the CARF (Class Association Rule Filter) method and the new cut-off procedure that is based on Information Theory approach. These two methods combined determine small subsets of features by calculating a cut-off value to distinguish between relevant and irrelevant features in the feature selection process. The methodology followed is based on CAR mining and depicted in Figure 13. We assume that the input dataset has been pre-processed in terms of missing values, discretization, normalization, etc. The first phase in the methodology is to transform the processed dataset into Line Space (LS) data format to efficiently discover the rules. All items will be represented by ColumnID:LineID, which simplifies the rule discovery step by efficiently computing itemsets' support and confidence values (See Section 3.2 for definitions). The CARF method is focused on discovering rules, for which we need one data scan.

The choice of the Vertical CAR approach for 1-rule discovery was based on several factors including:

1) It represents the input dataset in a compact data structure, i.e. LS and Item Space (IS) layouts, that enables an efficient search for items' information during the process of building the rules

2) It improves the computation process as discussed in Chapter One.

3) It has proven experimentally to be better than the Horizontal CAR approach in multiple studies as discussed in Chapter One.

The CARF method does not implement all steps of the Vertical CAR approach as we will see later on, rather it adopts this approach for data representation to enable efficient 1-rule discovery; thus, classifier building, and prediction phases are excluded since they are not part of the feature selection process. Once the rules are discovered, we utilize their features and data coverage information to calculate the scores of the available features. Each feature appearing in a rule will have a score assigned to reflect its relevance. Features that appeared in the rules will a have better chance of being chosen since they will have a high probability of classifying training instances. Once CARF is selected, the end-user can choose MutInfoMethod (the new cut-off procedure) to compute a cut-off score based on all ranked features. The cut-off score is an indicative measure that shows how many features should be retained by the user. The details on how the cut-off is computed will be provided in Section 3.5.3. Before explaining the main steps of CARF and the search method, we introduce the CAR mining approach adopted to design and implement CARF. The concentration will be on CAR mining terms, vertical mining, and data layouts.

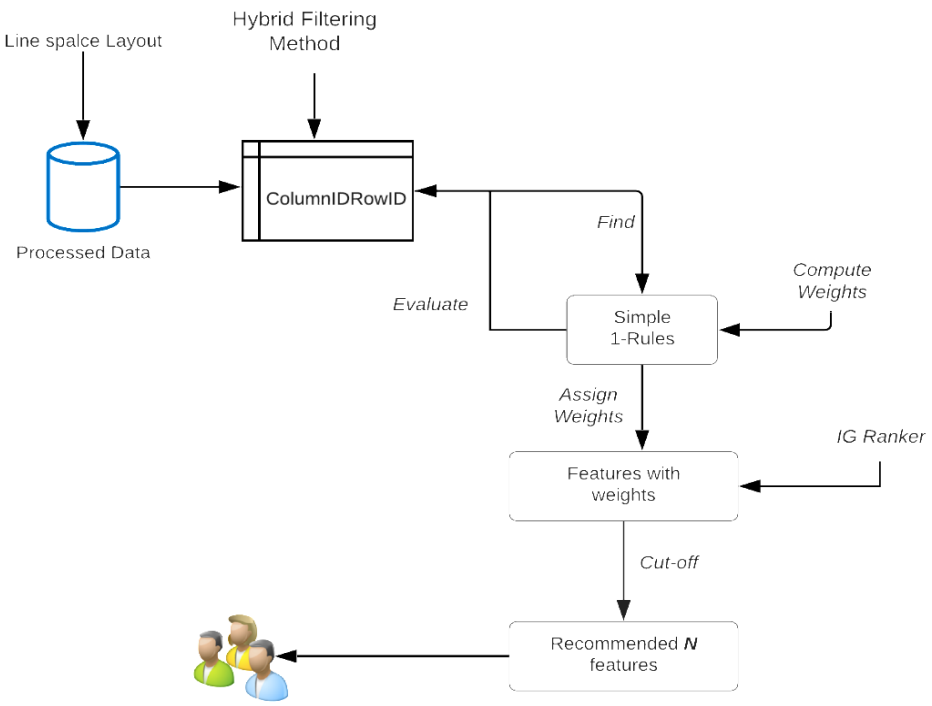

Figure 13: CARF Method

3.2 Terms and Example of CAR Mining

Typically, the CAR mining approach adapts an Association Rule Mining algorithm to discover a complete set of rules. A pruning step is then invoked on the consequent part of the rules to filter out the discovered rules into a subset that contains the class label. All other rules that have feature values on the consequent part of the rules are discarded since they add no value during the prediction step. The retained subset of rules is then used to forecast test instances. Since the proposed filtering method adopts CAR mining to devise the rules, the below terms are used by CARF when a training dataset T with labelled class C is used as an input:

1- An itemset **items** is a form of frequent pattern that consist of a feature F_i and its value f_{ij}, denoted as (F_i, f_{ij}).

2- The i_{th} *training instance* in T is a list of feature values (F_{i1}, f_{i1}), …, (F_{in}, f_{jn}), plus a class label denoted by c_i.

3- An itemset is a set of disjoint feature values occurring in a training instance, denoted as < (F_{i1}, f_{i1}), …, (F_{in}, f_{in})>.

4- A 2-itemset is a set of two disjoint feature values occurring in a training instance, denoted as < (F_1, f_1), (F_2, f_2)>.

5- The frequency (*freq*) of itemset items in T is the number of instances in T that match the item.

6- An itemset's items passes the minimum support threshold (*minsupp*) if, freq(*items*)/ |T| ≥ *minsupp*. Such itemset is said to be a frequent itemset.

7- From an item AB we can generate a rule R: $A \rightarrow B$. The rule passes the minimum confidence threshold (*minconf*) if *freq* (A B)/ *freq*(A) ≥ *minconf*.

8- 1-rule is a special case of an itemset that passed *minsupp* and *minconf* of the format <*antecedent*, *class*>, where antecedent is 1-temset and the consequent is a class value.

3.3 CAR Mining Example

We demonstrate an example to show how Class Association Rule mining algorithms work to elaborate on how the CARF method learns the rules. Consider the dataset displayed in Table 3.1A and suppose that the minimum support count is set to 2; in other words, for a feature value to be considered in a rule it must occur in the dataset at least two times, and the minimum confidence (minconf) is set to 70%. Table 3.1B depicts that all features' values with support frequencies larger than the minimum support count, and therefore the complete candidate 1-itemset, are kept as they are all frequent. Table 3.1C displays the candidate 2-itemset that resulted from Table 3.1B along with their support frequencies from which any 2-candidate itemset that has not passed the minimum support count has been removed. The remaining frequent 2-itemset which passed the

minimum support requirement is utilized to generate the candidate 3-itemset as shown in Table 3.1D. In that table, only two candidate 3-itemsets are frequent and the rest are removed.

Table 3.1A: Sample Dataset of Eight Instances

Feature 1: Smoker	**Feature 2: Gender**	**Feature 3 (Class): Potential Heart Problems in the Future**
False	Female	No
True	Female	No
True	Male	Yes
True	Male	Yes
False	Male	No
True	Female	Yes
False	Female	No
True	Male	Yes

Table 3.1B: Candidate 1-itemset from Table 2A

Candidate 1-itemset	**Support Count**
False	3
True	5
Female	4
Male	4
Yes	4
No	4

Table 3.1C: Candidate 2-itemset from Table 2A

Candidate 1-itemset	**Support Count**
~~False, Male~~	~~1~~
False, Female	2
True, Male	3
True, Female	2
~~False, Yes~~	~~0~~
False, No	3
True, Yes	4
~~True, No~~	~~1~~
Male, Yes	3
~~Male, No~~	~~1~~
~~Female, Yes~~	~~1~~
Female, No	3

Table 3.1D: Candidate 3-itemset from Table 2A

Candidate 1-itemset	**Support Count**
False, Female, NO	2
True, Male, Yes	3
~~True, Female, Yes~~	~~1~~
~~True, Female, No~~	~~1~~
~~False, Male, Yes~~	~~0~~

Table 3.1E: Association Rules Produced from Table 2A and Potential Class Association Rule in Yellow

Rule #	Rules	Support Count	Confidence	
1	Yes →True	4	100%	
2	False → No	3	100%	
3	Male, Yes → True	3	100%	
4	True, Male → Yes	3	100%	
5	False, Female → No	2	100%	Remove
6	True → Yes	4	80%	
7	No →False	3	75%	
8	Male → True	3	75%	
9	No → Female	3	75%	
10	Female → No	3	75%	
11	Yes → Male	3	75%	
12	Male → Yes	3	75%	Remove
13	True, Yes → Male	3	75%	
14	Yes → True, Male	3	75%	
15	Male → True, Yes	3	75%	

Table 3.1F: Rule Evaluation Process Against the Dataset of Table 2A

Feature 1: Smoker	Feature 2: Gender	Feature 3 (Class): Potential Heart Problems in the Future	Rule # Covering the Instance
False	Female	No	2
True	Female	No	10
True	Male	Yes	4
True	Male	Yes	4
False	Male	No	2
True	Female	Yes	6
False	Female	No	2
True	Male	Yes	4

Table 3.1G: Class Association Rule After Evaluation

Rule #	Rules	Confidence
2	False → No	100%
4	True, Male → Yes	100%
6	True → Yes	80%
10	Female → No	75%

The complete frequent itemsets generated from the dataset (Table 3.1A) are now available, so association rules can be produced using the confidence test as shown in Table 3.1E. In that table, rules that pass the minconf threshold are displayed. However, a filtering process is applied to keep the rules with class value on its consequent (the highlighted yellow rules) as these are the CARs that can be employed for prediction, and all other rules will be pruned.

To evaluate the retained rules' goodness, they will be tested on the training dataset one by one in a top down fashion using their confidence values. This pruning step is crucial as we are only interested in rules that have classified data instances and differ from other rules to ensure a reduction in rules similarity. Table 3.1F depicts the training data instances along with the rules used to correctly classify; these are the final CARs that we keep as shown in Table 3.1G. Potential similar rules such as #5 and #12 have been removed since other similar rules with a higher position classified their instances and thus this procedure minimizes rule redundancy. It should be noted that we do not follow the rule generation and evaluation process described earlier in the proposed CARF method, rather we simplify it to produce rules with 1 feature value in its antecedent part (left hand side) only, as discussed later in Section 3.5.2.

3.4 Proposed Filtering Method Data Layout (Vertical Mining)

Most CAR mining algorithms adapt the horizontal data format model to represent the input dataset. When using the horizontal data model, the input dataset is represented by sets of rows and columns where each row corresponds to a data example and each column corresponds to a feature (see Table 3.1A). Usually the CAR mining algorithm examines the data examples in top down fashion during the learning step (discovering frequent itemsets). It computes the frequency (support) of the candidate itemsets storing them within a data structure. For example, the CBA algorithm [48] iterates over the training dataset *m* times using the Apriori step-wise search method [7], where *m* corresponds to the number of data scans needed to produce the entire frequent itemsets; this is not cost-effective in regards to computation complexity as stated earlier. CBA is one of the successful CAR mining algorithms that implements the Apriori itemset_generation method by taking at iteration *m* the set of frequent (*m*-1)-itemset to produce candidate *m*-itemset and from which it

derives m+1 the frequent itemsets. The CBA algorithm and its successors utilize horizontal data layout thus they inherit problems from Association Rule Mining such as:

- Computation complexity
- Exponential growth of candidate itemsets and hence increasing the possible number of rules
- Do not deal with datasets with imbalanced class labels
- Rules overlapping in data instances causing large classification models in terms of the number of rules.

Vertical Mining enhances the efficiency of discovering frequent itemsets and simplifying the process of rule generation used by algorithms that employ the Horizontal Mining approach. This approach uses an efficient data layout called Tidlist (T-list) or LS to process large datasets in parallel and distributed data processing applications [55,163,167]. A dataset in Vertical Mining is represented by items, along with their line numbers that they incurred in the training dataset, in a special data structure called T-list (see Tables 3.2B and 3.2C). Using this data format, the process of computing support values for itemsets and confidence values for the rules will be simple using intersections of the T-lists of disjoint itemsets. When compared with the Horizontal Mining approach, the Vertical Mining approach reduces the number of passes over the dataset to a single time [78,99].

CARF uses a more efficient data layout based on the LS format that maps each feature and the class labels to ease support and confidence computations by repeatedly transforming the data between LS and Item Space (IS). A LS basically represents a transaction's identification (TID) and its corresponding entries (feature values), in which each entry (item) is represented as ColumIDs:LineID. Whereas IS takes one entry of the LS and transforms it into a key:value entry in which the key denotes the item (ColumIDs:LineID) and the value corresponds to the lines that

appeared with that item. More importantly, transforming between LS and IS can help the algorithm discover frequent itemsets, then efficiently generate and rank the rules before constructing the classification model.

Column IDs: are the features / column numbers in the training dataset in which the item appeared.

LineID: The data instance (row) number in which the itemset first occurred in the training dataset.

When the input dataset is converted into LS and then to IS, all intermediate data retain the same data formats until the algorithm terminates, simplifying the rule discovery phase. We have followed this approach when discovering the rules of CARF as discussed in Section 3.5.2.

Vertical Mining approach uses IS and LS to identify the frequent itemsets by simply intersecting the lines of the disjoint itemsets. The result of a single intersection between two frequent 1-itemsets will be the lines of the new candidate 2-itemset in which frequent 1-itemsets have occurred together in the dataset. By taking the size of the resulting lines of the new candidate 2-itemset we obtain its support value and decide whether it is frequent or not without having to go back to the dataset and compute its support as required in Horizontal Mining algorithms.

Overall, the CAR mining approach has shown promising results in terms of performance for classification accuracy involving multiple applications including fraud detection, cybersecurity, the stock exchange, medical diagnosis, and others. However, a notable issue related to the rules produced by this approach is that they may share training instances since these rules utilize shared instances. To be precise, the support counts for multiple rules would include data instances that these rules have in common in the training dataset. Therefore, creating a rule redundancy that is obvious when models generated by CAR mining are compared with those of Decision Trees, Covering, and Rule Induction [3,11]. If overlapping of rules is reduced, then we believe that this

approach can be used to measure the significance of the features within these rules and thus chosen as a feature selection method. In Section 3.5, we show our proposal for the new feature selection method and how we reduced the issue of rules overlapping.

3.5 **The Proposed Class Association Rule Filter (CARF) Method**

CARF's main steps are depicted in Figure 14. In the next sub-sections, we discuss these steps in more detail.

Input: Dataset, minsupp and minconf thresholds
Output: Sorted features with weights

1. Invoke LS & IS Conversion procedure to covert the input dataset into a ColumnID:LineID data format
2. Invoke the Rule_Induction and pruning procedures to compute frequent 1-itemset and candidate 1-rule
3. Invoke Feature_Weight_Assignment procedure
4. Invoke CARF's procedure to determine the cut-off.

Figure 14: Steps in CARF method

3.5.1 Data Conversion

The input data is normally represented by features and instances (columns and rows). To simplify the process of rule generation, each instance will be represented by a line number and will be linked with two integer values (the column ID in which it occurs and the first line in which it

Mapping Procedure (convert dataset into LS)

Input: A training dataset

Output: A LS data formats

```
1 for each distinct feature value in the training dataset
2       generate the entry key <feature value, LineID>
3 end
4 for each cluster of entries with same feature value as key
5       choose the lowest LineID to represent the feature value.
6       generate <Feature value, LineID>
7 End.
```

Algorithm 3.1: Data Transformation Procedure

appears). This process is demonstrated in Algorithm 3.1 above which needs to iterate once over the training dataset. Therefore, the CARF method transforms the training data into LS to prepare it for the training step in which the induction procedure to discover the rules is invoked. The LS will be transformed into IS, i.e. items in the LS format will have corresponding keys in the ColumIDs:LineIDs format, and corresponding values (Line:Class) in line format to automatically locate the relevant information, i.e. their frequencies, without having to scan the input dataset repeatedly.

To illustrate how the CARF method uses LS and then transforms it into IS, consider the training dataset shown in Table 3.2A (horizontal data format) and its equivalent LS data format shown in

Table 3.2A: A Training Dataset

Line#	Features			
	F1	F2	F3	Class
0	A_1	Y_1	A_2	C_1
1	A_2	Y_1	A_2	C_1
2	A_2	Y_2	A_2	C_2
3	A_2	Y_2	A_2	C_1
4	A_1	Y_1	A_1	C_2
5	A_1	Y_2	A_1	C_1
6	A_2	Y_2	A_1	C_1
7	A_2	Y_1	Y_2	C_1
8	A_1	Y_1	A_1	C_3

Table 3.2B: LS Data Format

LineID	Features			Line: Class
0	(0)0	(1)0	(2)0	0:0
1	(0)1	(1)0	(2)0	1:0
2	(0)1	(1)2	(2)0	2:2
3	(0)1	(1)2	(2)0	3:0
4	(0)0	(1)0	(2)4	4:2
5	(0)0	(1)2	(2)4	5:0
6	(0)1	(1)2	(2)4	6:0
7	(0)1	(1)0	(2)7	7:0
8	(0)0	(1)0	(2)4	8:3

Table 3.2C: IS Data Format

	Features	Line: Class
$F1=A_1$	(0)0	0:0, 4:2, 5:2, 8:3
$F1=A_2$	(0)1	1:0, 2:2, 3:0, 6:0, 7:0
$F2=Y_1$	(1)0	0:0, 1:0, 4:2, 7:0, 8:3
$F2=Y_2$	(1)2	2:2, 3:0, 5:0, 6:0
$F3=A_2$	(2)0	0:0, 1:0, 2:2, 3:0
$F3=A_1$	(2)4	4:2, 5:0, 6:0, 8:3
$F3=Y_2$	(2)7	7:0

Table 3.2B. In Table 3.2B [55], Itemset ($F1=A_1$) was represented by 0:0 (the first column and line that $F1=A_1$ has appeared) and itemset ($F3=A_1$) was denoted 2:4. It should be noted that if the

dataset is not a classification task (no class value), then A_1 will be represented by 0:0. Table 3.2B depicts the LS representation allowing for the class label in which each row represents the occurrences of the itemset value. If we group the entries of the line in the IS (Table 3.2C), then the support value can be obtained taking the class into account. For instance, row 1 in Table 3.2C shows that itemset (F1 =A_1), i.e. 0(0), has a support of 4 without considering the class information as it appears four times in the dataset, twice with class C_1 and once with class labels C_3 and C_2, respectively. Using the IS data layout, we can easily use the class information and thus compute the support for the relevant items thus further reducing the search space of itemsets' discovery and improve the mining approach.

3.5.2 Rule Discovery and Weight Calculations

Initially, the CARF method uses a rule induction procedure (according to Algorithm 3.2) to scan the training dataset to construct the LS data structure (as per Algorithm 3.1). The LS is then converted into an IS where keys are now items IDs (ColumnID: LineID) and values are lines where the items appeared in the training dataset. Using this IS data format, the frequent 1-itemset is now available i.e. a feature value plus a class value, by aggregating the entries that belong to the same item key to compute the support values. Any 1-itemset that has a frequency below the minsupp will be discarded at an early stage and not stored for later use (Line 8). The number of iterations to discover the frequent 1-itemset and the 1-rules is just one (as discussed earlier) since we only perform one data scan to perform the data transformation between a LS and an IS.

CARF computes the best 1-rule (rules with one feature value in the antecedent and one class value in the consequent) in terms of confidence after removing all infrequent 1-itemsets. Any candidate

rule (frequent 1-itemset) with a confidence below the minconf will be removed. Once the best 1-rule (rules having just 1 item in their antecedent) is found (Algorithm 2- Line 12), then

a) Its weight is computed based on Equation (3.1) (Line 13)
b) The 1-rule is recorded in the 1-rule set (Line 14)
c) The training instances covered by this rule are erased from the IS data structure (Line 15)
d) Repeats 1-15 of Algorithm (3.2)

The process is iterative and ensures that

a) Only the fittest 1-rule in terms of confidence is accounted for at each step of the rule induction process
b) Once a 1-rule is derived, its training instances will not be counted for any possible remaining candidate 1-rule.

The rule generation process guarantees that the produced 1-rules are evaluated against the training dataset and cover training instances. The outcome will be a set of rules of size 1 (1-rules) derived from the training dataset and without data redundancy. The Rule Generation method at early stages prevents any possible participation of weak features (those that belong to weak candidate rules) from taking part in the rule-growing process by discarding them at preliminary stages. More importantly, the process considers statistically correlated features with the class (1-rules) to reduce the search space, and only those will be used to compute the features weights as we will discuss soon. We only induce 1-rules from non-overlapping instances of the input dataset ensuring high feature-class correlation and low feature-feature correlation.

Mapping Procedure (convert LS into IS) and Discovering Frequent Itemsets and 1-rules

Input: A training dataset T

Output: IS data format with frequent 1-itemset and candidate 1-rule

```
1 LS ←Map(T)
2 for each entry in LS do
3       generate <Feature value, LineID >                          // set a key for the feature value
4       Reduce (entries of same key):
5       Support_count = number of LineIDs
6       Item_confidence = (number of LineIDs with the largest class/ number of LineIDs)
7       LineIDs_min = the smallest value of group of LineIDs
8             If Support_count >= minsupp
9                   begin
10                        generate <Feature value, LineIDs_min >
11                              If confidence >= minconf
12                                    Produce R
13                                    Compute R's weight
14                                    CAR_Set ← R : (< Feature value, LineIDS > <confidence, support_count>)
15                                    Update <Feature values, LineIDs > by erasing R's lines
16
17                  end
18    end
```

Algorithm 3.2: Proposed Frequent Itemset and 1-Rules

In the above rule-growing procedure, we ensure that 1-rules are produced from different data instances to reduce the possibility of data overlapping which may cause the generation of redundant rules or overfitting. Since features are assigned weights based on their appearance within the derived rules and these rules are not redundant, feature-to-feature correlations (at least within shared instances) are minimized.

The features that belong to the derived 1-rules will then be assigned weights based on a) The order in which the rule is produced; b) The number of data instances covered by the rules; c) The number of uncovered data instances before the rule was evaluated. The process of computing the weights

for the available feature is performed by examining the 1-rule set. The function checks the feature within the rule, i.e. R, and assigns the feature a score using Equation (3.1)

$$Weight\ (\ F_i\ ) = freq\ (\ R\ )\ \text{X}\ UDV\ (\ R\ ) \qquad (3.1)$$

Where F_i is a feature, *freq (R)* is the number of instances covered by R, and *UDV (R)* denotes the number of uncovered data values in the training dataset before R was evaluated.

For a rule such as R: $A_1 \rightarrow Class_2$, If (A_1, $Class_2$) appeared 15 times in a training dataset with 100 total instances and R was the highest-ranked rule obtained, then the weight of feature A_1 will be calculated (15x100 = 1500). The next candidate rule will then be evaluated on the remaining 85 instances. For each 1-rule generated, the CARF automatically estimates the weight of the feature that occurred in the rule's antecedent during the rule induction process, which is highly efficient. All information related to the set of 1-rules is available in the IS data structure (ColumnID; LineID; Line:Class) therefore, accessing the number of instances covered by the rule and the remaining uncovered instances is straightforward.

Using the CARF method, not only rules with higher confidence will potentially contribute to computing the scores of the features, but also rules with actual data coverage. Hence, dissimilar features will potentially have high weights and thus a larger possibility of being retained by the user.

3.5.3 The CARF's Cut-off Procedure (MutInfoMethod)

Most filtering methods use a naïve method to order the list of features retained based on the scores computed from large–small. These methods fail to answer questions such as, "How many features should be retained?"

We propose a new search procedure called MutInfMethod that will help answer the above question.

Most Filtering methods suffer from not providing a clear cut-off point between useful and useless features. They tend to sort all available features from scores computed by the mathematical model, complicating the task for the user. For instance, if IG is applied on a dataset of 1000 features, it will estimate the scores using Entropy and then sort the features depending on their correlation to the target label (scores obtained). The task of which features to choose is left for the domain expert as the outcome given by IG does not provide any relevant information. The domain expert or the user has the freedom to decide if the top 10, 50, or 100 features should be chosen, which can cause important features to be overlooked and less informative features to be incorporated into the prediction models; this leads to poor classification performance.

A smart mechanism should be integrated into the existing Filtering methods to define the boundary between informative and less informative features independent from domain expert, data characteristics, and the mathematical models used. The mechanism should be robust so that it can be integrated with any Filtering method to guide the user by reducing the search space of the results offered. This would make the user's examination process less complicated and more focused. We show in Section 3.5.3 our contribution on how to recommend ***N*** specific subset of features to the end-user by the CARF method.

One possible way to build a new procedure to identify the cut-off between irrelevant and relevant features is by utilizing the Mutual Information concept. This was originally proposed to effectively determine the worthiness of the features in a dataset using features and class information, and latterly was used in Supervised Learning applications by the Decision Trees classification approach.

Mutual Information is a foundation concept based on the information theory research discipline in mathematics. It was initially investigated by Shannon (1948) [152] on how to represent data messages in a compact manner to efficiently use this in communication and signal processing applications concerning data transmission and data compression operations. A question such as "How informative is the message / data in terms of information?" is fundamental. Mutual Information tries to answer such a question by quantifying features to reveal how much information they can offer. In a typical classification task, Decision Tree algorithms such as C4.5 [137] utilize IG to reveal how much information each feature supplies by measuring the Shannon entropy, i.e. uncertainty of the dataset before a split and after a split using that feature. The algorithm usually selects the feature that has the largest information minimizing the Shannon entropy so the data can be split into effective groups.

The idea of using IG is quantified using events and probabilities. Information within an event can be calculated using the probability of that event; normally frequently occurring events require less information to represent them when compared with rare events since the latter is associated with a higher degree of uncertainty, i.e. entropy. Moreover, rare events, i.e. events with lower probabilities, require more information in bits. For instance, if we flipped a fair coin, the probability of that event will be 1/2 and the information obtained using H(A) will be 1.000 bit. Whereas the probability of rolling a number on a dice will be less than that of flipping a coin, i.e. 1/6, and thus the expected amount of information would be larger, 3.222 bits since the probability is lower (more supervising event).

In the context of classification of data for feature selection, features that are more correlated with the class label, i.e. have higher information gains, are able to split the dataset using the class

information since they offer higher average reduction in the Shannon entropy. For a discrete feature *A* whose values are $\{a_1, a_2, a_3, \ldots, a_n\}$, the entropy can be estimated using Equation 3.2.

$$\text{Entropy(A)} = \sum_{i=1}^{n} -P_{ai}\, log_2\, P_{ai} \qquad (3.2)$$

Where the *logarithm* is the base-2 and *P(ai)* is the probability of the feature *A* occurring with *class i* in the training dataset. The reason for using logarithm base 2 is that information is measured in bits.

In the proposed cut-off procedure, the set of all features along with the assigned scores by the CARF method are used as input. Since the weights may vary significantly, where high correlated features with the target class have larger weights than low correlated features, then the features should be ordered. The cut-off procedure's first step is to arrange features based on their assigned computed scores in descending order. Then it performs the following main steps to compute the cut-off score to isolate relevant features from the irrelevant features and offer only the relevant ones to the end-user. The following main steps are then performed to compute the cut-off score to isolate relevant features from the irrelevant features and offer only the relevant ones to the end-user.

1) The scores assigned to the features are normalized as the cut-off procedure can be integrated with any feature selection method. In performing normalization, the cut-off procedure ensures that each data observation is within a range of 0.0 to 1.0 and that all data observations are within the same scale. This indeed makes the data observations within a commons scale, reduces data distortion, and simplifies the process of computation regarding the cut-off value. This step is done using the below normalization equation

$$\text{Normalize (A)} = \frac{A_{weight}}{\sum_{1}^{n} A_{n-weights}}$$

2) The normalized weights associated with the features are then processed by the cut-off procedure to calculate how each of them provides information. In this way, each feature's score is treated as a probability and denotes the correlation between that feature and the class label computed based on the mathematical model of the feature selection method used. For instance, in the CARF method, the scores of the features are computed using the rules discovered. The cut-off procedure uses the Mutual Information approach and calculates the entropy for each normalized feature's score. In this way, each feature is now associated with the computed score as well as its corresponding degree of information in bits. This step is done using the below entropy equation

$$\text{Entropy(A)} = \sum_{i=1}^{n} -P_{ai}\, log_2\, P_{ai}$$

3) The degree of information in bits for all features is then totalled to provide one measure for all of the available features in respect to their degree of information. Then 2 to the power of the total is calculated to reflect the number combinations in bits of the computed so features based on their computed scores can be selected. By calculating the 2 power of the total of information for the available features' scores, an indicative measure is obtained on how many features can be chosen by the end-user. This step is done by taking 2 to the power of the total entropy

$$\text{Cut-off} \leftarrow 2^{\text{E\_F\_total}}$$

In the proposed search procedure (Algorithm 3.3), the features will be associated with the computed weights and sorted in descending order. We normalize the weights to ensure that no matter what range of values is used as input for the CARF's cut-off procedure, these scores will be between 0–1 according to the normalization equation (Line 4). To elaborate on the

normalization issue, Filtering methods often produce different scores using various mathematical models. For example, CST produces scores above one, and GR derives scores between 0–1. We unify the scores, which can be done using normalization (Equation 3.3). Once the weights of the features are normalized, then we calculate the IG for each feature based on its normalized weight and for all features with weights above zero. The total of the IG values is then obtained for the available features and utilized to compute the cut-off value of the number of indicative features that the user should use. This is computed using the equation at line 11 in Algorithm 3.3.

$$\text{Normalize (A)} = \frac{A_{weight}}{\sum_1^n A_{n-weights}} \qquad (3.3)$$

By using the new search procedure, features with a limited number of distinct values in the training dataset will be associated with higher weights due to the fact that they provide more information as they represent 1-rules with high data coverage yet non-overlapping instances. Thus, these features will probably have a higher chance of being retained by the CARF method, i.e. their position will be within the suggested cut-off value. Features that are common will have light weights and hence their position in the retained subset of features will be below the suggested cutoff

The New Cut-off Procedure

Input: A set of features weight F with their corresponding weights

Output: A cut-off

1. N_F ← Ø //Empty set to hold the normalized weights
2. N_F_total ← 0
3. for each F_i in F do
4. N_F ← Normalize (F_i) // Normalize (F_i) using Equation (3.3)
5. N_F_total ← N_F_total + N_F
6. end
7. E_F ← Ø //Empty set to hold the entropy of each weight
8. E_F_total ← 0
9. for each N_F_i in N_F do
10. E_F ← Entropy (N_F_i)
11. E_F_total ← E_F_total + E_F
12. end
13. Cut-off ← $2^{E\_F\_total}$
14. Return (Cut-off)

Algorithm 3.3: The CARF's Cut-off Procedure (MutInfMethod)

3.5.4 Example of CARF

Revising the dataset of Table 3.1A, and assuming that the minimum support count = 2 and the minimum confidence = 75%, the algorithm presents the data in a vertical format by Line:Class and the ColumnID:LineID as shown in Table 3.3A to simplify the processes of locating frequent items in the dataset and generating the 1-rules. In Table 3.3A, the first column represents the line# and the class ID and the second column contains the features and their ColumnID:LineID representation. CARF algorithm coverts the data into IS as shown in Table 3.3B, which can be considered as a map in which items are keys and their corresponding lines with class labels are

their values. This data representation makes the process of locating frequent 1-items and generating the 1-rules straightforward. To be precise, and to find frequent 1-item (feature values with occurrences larger than or equal the minimum support threshold), CARF only groups the lines associated with each 1-item to compute its frequency. For instance, 1-item 0(0)= "Smoker=False" has appeared in Line:Class set {0:(0),4:(0),6:(0)}, which means it has occurred in lines (0,4,6) and with class ID = 0 (No). Based on such information, the support of this itemset is 3 and its confidence is 3/3 as it only occurs with class 0 (No).

CARF discovers the possible frequent 1-itemset obtained after the initial scan of CARF, in which itemset "Smoker=False", i.e. 0(0), as discussed earlier has only occurred with class "No", so it has 3/3 confidence (100%) and therefore it is the strongest 1-item. Therefore, 1-item 0(0) will be generated as a 1-rule, i.e. False ⇾ No, and all data instances connected with it is discarded from the dataset, i.e. Lines (0,4,6). The weight of this rule is also computed in the induction procedure as $3 \times 8 = 24$ and assigned to feature "Smoker". Once the instances of the first 1-rule are discarded then CARF derives the possible remaining 1-item from the uncovered instances as shown in Table 3.3C. In Table 3.3C, "Gender=Female" has only one occurrence with class "Yes" and "No" respectively, so it failed to pass the minimum frequency and therefore it is discarded. The best 1-item remains from the uncovered training data is "Gender=Male" with class label "Yes" as it has 100% confidence. This item is then converted into 1-rule, i.e. Male ⇾ Yes, and its data (lines 2,3,7) are removed from the training dataset as shown in Table 3.3B. After removing the second rule from the remaining uncovered data only two instances remain uncovered and no more frequent 1-item can be discovered, so the rule discovery method terminates.

Using this rule discovery procedure by CARF there is no need to go back to the dataset to calculate support and confidence. More importantly, the process of computing both support and confidence for 1-rules is simple and efficient and does not require another phase for rule generation as in most existing CAR mining algorithms.

Table 3.3A: LS format of Table 3.1A

Line: Class	Features	
0:(0)	(0)0	(1)0
1:(0)	(0)1	(1)0
2:(2)	(0)1	(1)3
3:(2)	(0)1	(1)3
4:(0)	(0)0	(1)3
5:(2)	(0)1	(1)0
6:(2)	(0)0	(1)0
7:(0)	(0)1	(1)3

Table 3.3B: IS format of Table 3.1A

Feature Name	ColumnID:LineID	Line:Class
FALSE	0(0)	0:(0),4:(0), 6:(0)
FEMALE	1(0)	0:(0), 1:(0),5:(2),6:(0)
TRUE	0(1)	1:(0), 2:(2), 3:(2), 5:(2), 7:(2)
MALE	1(2)	2:(2), 3:(2), 4:(0),7:(2)

Table 3.3C: IS format of Table 3.1A

Feature	ColumnID:LineID	Line:Class
~~FEMALE~~	~~1(0)~~	~~1:(0),5:(2)~~
TRUE	0(1)	1:(0), 2:(2), 3:(2), 5:(2), 7:(2)
MALE	1(2)	2:(2), 3:(2), 7:(2)

Once the rules are generated then their weights are assigned to the feature as follows:

Normalized Features Ranks:

Feature Weight

Smoker 0.615

Gender 0.385

The cut-off point using Shannon entropy is then calculated using its designated mathematical formula to recommend two features to the end user. This example, if limited, demonstrates how the filtering method works.

3.6 Chapter Summary

In this chapter we proposed a new Filtering method based on inducing simple rules from the classification datasets called CARF and a new search method that helps the user identify how many features to retain called MutInfoMethod. The CARF method learns rules with 2-item in the dataset and from these computes weights that are intelligently assigned to the features in the dataset. The transformation of data format from LS to IS during the rule discovery process makes this a simple and efficient approach. The distinguishing characteristic of the CARF method is that it erases redundant rules as early as possible; this has ensured that rules generated are not redundant. We think that will be advantageous as fewer features will be retained, making the process more efficient. We also investigated a vital issue in feature selection—reducing the time for checking the results of Filtering methods—and have proposed MutIfMethod, an automatically generated cut-off threshold procedure to differentiate between relevant and irrelevant features. This new cutoff procedure suggests the number of features to be chosen by the end-user without the results having to be manually checked. In the next chapter, we show the implementation of CARF and MutInf Methods in Java and their integration in the Weka open source machine learning platform. More importantly, we reveal the true perfromance of CARF on a large number of real datasets and compare its perofrmance with other known filtering methods using machine learning algorithms.

Chapter Four

4. Implementation and Experimental Analysis

4.1 Introduction

The proposed feature selection method, i.e. Class Association Rule Filter (CARF), was implemented in the Java platform so it can be integrated easily within the Waikato Environment for Knowledge Analysis (Weka) data mining tool [53,156]. The reasons for choosing the Weka data mining tool as a primary environment for running CARF are fourfold:

1) Weka consists of a large number of feature selection methods and learning algorithms therefore the experiments can run without having to re-code the methods utilized in the experiments
2) Weka is an open-source tool, and no cost is incurred when running the experiments
3) Familiarity of the Java programming language makes it easier and faster to design, implement, test, and integrate the new source code of CARF and the new mutual information search method within Weka's feature selection package
4) Several other machine learning tools can be accessed through Weka such as R and Deep learning 4j.

In this chapter, the implementation, testing, and validation details of CARF are discussed. We examine the datasets, experimental settings, methods, and evaluation metrics used in the experiments, and present results analysis. We initially describe Weka's environment and structure, then highlight CARF's threshold setting and user interface (UI). The data and results analysis of CARF and other feature selection methods are presented in detail according to various evaluation metrics including, but not limited to, recall, precision, and accuracy.

4.2 Testing Environment

CARF is coded in Java and implemented within the Weka environment, which is available free to developers, researchers, and students for contribution. Weka consists of a large number of data mining and machine learning packages related to correlation analysis, regression, classification, clustering, association rule, feature selection, data visualization, discretization, and other tasks. The tool was initially developed at the University of Waikato in Hamilton, New Zealand and multiple versions have subsequently emerged; currently Weka 3.8 is the version that is mainly used for testing while Weka 3.9 can be used for the development of new intelligent learning algorithms such as ours. We have used the development version of Weka to integrate CARF Java source code.

Figure 15 depicts Weka's main modules including 'Explorer', 'Experimenter', 'KnowledgeFlow', 'Workbench', and 'Simple CLI'. The 'Explorer' module provides users with an easy-to-interact UI to perform various data pre-processing and learning tasks. It contains six major tab pages: 'Preprocess', 'Classify', 'Cluster', 'Associate', 'Select Attributes' and 'Visualize' for data pre-processing (missing values treatment, data normalization, data discretization, attribute type conversion, etc.), classification (decision tree, probabilistic, statistical, etc.), clustering (hierarchical, partitioning, overlapping, etc.), association rule, feature selection (filters, wrapper) and visualization (graphs, trees) respectively (See Figure 16 for the 'Explorer' tab options). The 'Knowledge Flow' module can be used for the same purpose as 'Explorer', but in a 2-dimensional view. Unlike the 'Explorer', which is used for single algorithm applications, the 'Experimenter' is used to design and run multiple experiments in parallel thus offering a robust way to deal with large experiments. The 'Simple CLI' module provides an editor interface that users can interact with to invoke Weka's functions directly using Java commands. Lastly, the 'Workbench' module

enables access to all modules within a single interface in Weka, for instance, users can use ‘Explorer’ and ‘Experimenter’ modules at the same time.

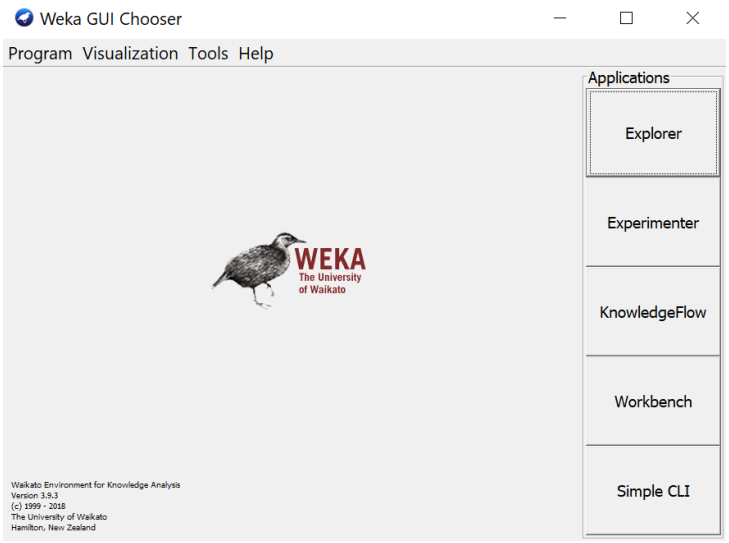

Figure 15: Weka Landing Page

Weka archives learning and data pre-processing resources such as source codes, compiled codes, library files, and meta data, i.e. description files, inside packages. The packages are mainly saved in WEKA_HOME, which can be amended by the developer. In WEKA_HOME, there are directories including packages, props, systemDialogs and repCache. When a developer or a contributor writes a new package and wants to share it with other contributors in the Weka community, then the contributor may contact the Weka administrator and provide the description file along with the source code for testing. Once the code of the new package is verified, it will be compiled by the administrator and archived in Weka’s repository to allow users to access it from the Weka’s ‘Package Manager’. This process can take a long time until the new package can become part of Weka’s central repository. The developer is liable for hosting the new package’s archive. There is an alternative way to make the new package source code available to the

community, in an unofficial manner, by making the new package/code available online—advertising is the responsibility of the contributor.

To utilize certain methods related to data processing in Weka, users can install the method via Weka's 'Package Manager'. Methods are organized in Weka based on certain characteristics. For example, classification algorithms are grouped in Weka's 'Explorer' based on the output format they offer and the learning scheme they adopt such as trees, rules, and probabilities, among others. Conversely, clustering algorithms are presented based on their names which is also the case for feature selection methods.

4.3 CARF UI and Mutual Information Search Method Implementation

Since the CARF feature selection method and the mutual information search method have been integrated into the Weka environment, then datasets in comma-separated format (CSV) and text format (ARFF), are accepted. The proposed search method which works with CARF, i.e. MutInfoMethod is independent from CARF and can be utilized by any other filter-based feature selection methods. The MutInfoMethod method when attached with any filter offers a cut-off value as well as recommending which features should be selected by the end-user. CARF ensures to produce scores linked with the available features and computed using 1-rules derived during the

learning step. Based on Figure 16, we selected CARF as the feature seletion method within the 'Attribute Evaluator' and MutInfoMethod as a search method.

The CARF package contains the necessary source codes, the proposed feature selection and

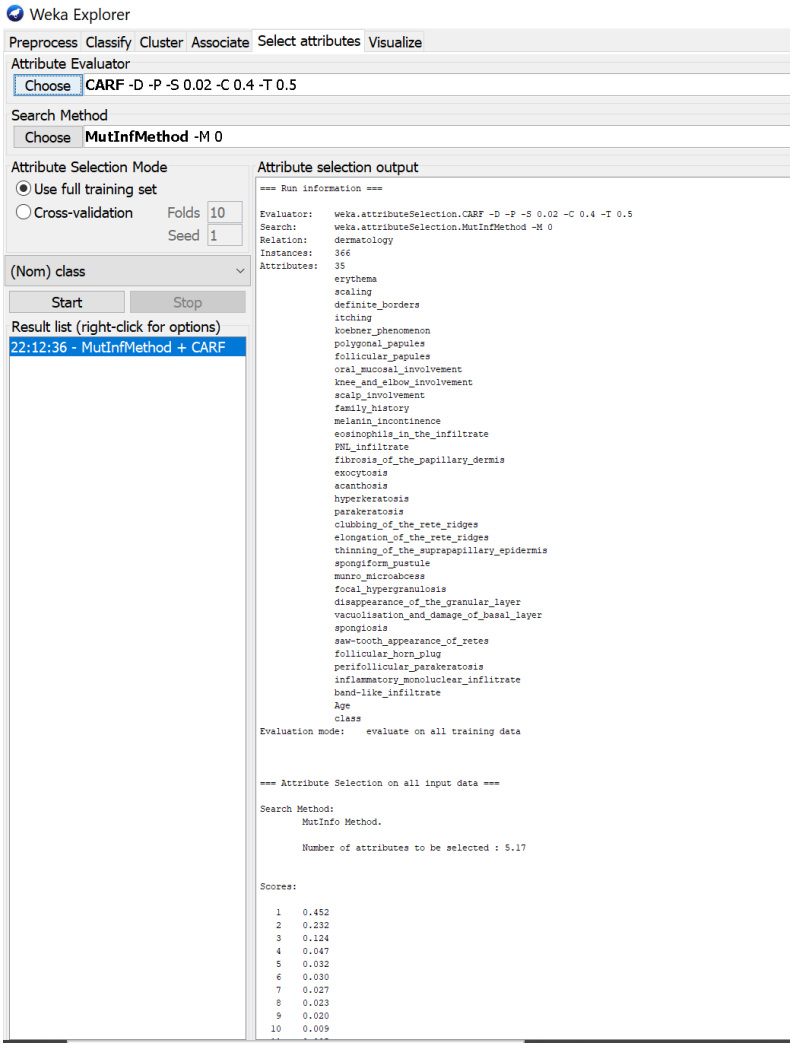

Figure 16: The 'Explorer' Tab Options when CARF is Selected

the search methods are compiled and then archived in Weka's home directory. The implementation version of the MutInfoMethod search method is embedded within the feature selection UI, i.e. 'Attributes Selection' and can be seen within Weka's 'Explorer' module. This tab page contains

the implementation versions of wrapper and filter methods in Weka such as Symmetrical Uncertainty, Information Gain, Chi-Square testing, Gain Ratio, Fisher score, and many others. Furthermore, there are search methods implemented inside the 'Attributes Selection' UI such as the 'Ranker', and others.

We have integrated CARF within Weka modules including 'Explorer', 'Experimenter', 'Command PLI', etc. When using the 'Explorer' module, the user can access CARF from the 'Select Attributes' tab page and 'Attribute Evaluator' (See Figure 17). The user must then choose

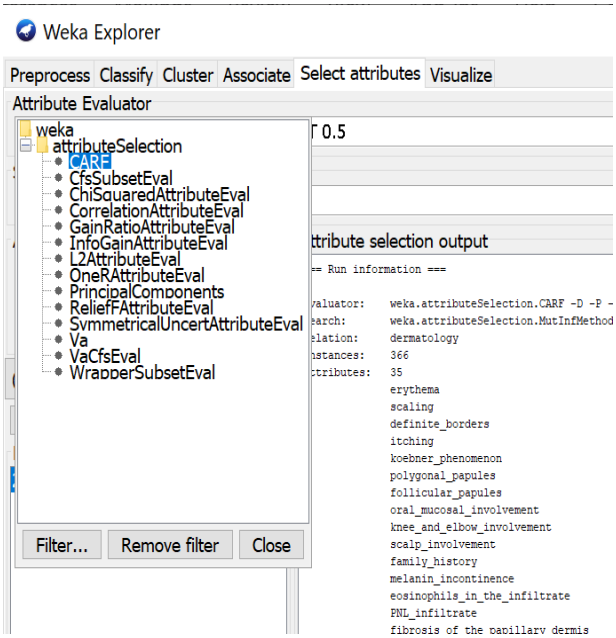

Figure 17: CARF in the Sample of Weka's Feature Selection Methods

MutInfoMethod as the search method. By doing so, the proposed search method will be the metric to decide the number of suggested features when CARF is applied on any given dataset. Therefore, when in Weka 3.8 the user should always amend the default search method from 'Ranker' into 'MutInfoMethod to ensure that the cut-off threshold, which separates influential features from those that are non-influential, is automatically calculated by CARF.

Simple customized UIs are designed and implemented as shown in Figures 4.4a and 4.4b for CARF and MutInfoMethod to enable users to set the minimum support and the minimum confidence thresholds. In the current CARF implementation, we use the default values for the minimum support and the minimum confidence of 2% and 40% respectively based on warming up experiments and previous research works discussed earlier in Chapter Three. The minimum support value is utilized as pre-pruning to filter out as early as possible any attribute value that has low frequency in the training dataset during the process of growing the rule. Therefore, only high frequency attribute values can be used in building rules; all attribute values with inadequate frequencies (frequencies lower than the minimum support thresholds) are discarded by CARF. Equally, the minimum confidence threshold plays a significant role in deciding which derived rules participate in the process of computing features scores, and which rules are removed early. The correlation between the attribute values and the class labels is evaluated when discarding rules below the minimum confidence thresholds thus ensuring that highly correlated rules remain for the computation of features scores.

4.3.1 Demonstrated Example

Figure 18 shows the UI of CARF, in which the minimum support and the minimum confidence values are set to 2% and 40% respectively, and Figure 19a depicts the results obtained by CARF when applied against the 'Anneal' dataset [94]. The 'Anneal' dataset is a multi-class dataset that consists of 38 features excluding the class label and contains 898 data instances and 6 possible class values. When employing the MutInfoMethod search method, CARF was able to reduce the dimensionality of the 'Anneal' dataset significantly by ignoring 33 features and keeping 5 features only as apparent in Figure 19a. We ran the Bayesian Network (Bayes Net) classification algorithm [20] on the subsets of features recommended by the CARF method and the derived model showed

95.10% predictive power. We repeated the same experiment on the 'Anneal dataset' by coupling the MutInfoMethod search method with Component Analysis (PCA) and Chi-Square testing feature selection methods [98,129] and their results are shown in Figures 19b and 19c, respectively. PCA and CST methods derived 30 and 17 subsets of features from the 'Anneal' dataset, respectively. When the Bayes Net classification algorithm processed the two subsets of features chosen by the PCA and CST testing method, the models generated have 94.87% and 96.21% respectively. This experiment, if partial, reveals that, the proposed feature selection and search methods not only substantially reduce the search space of features, but also this has little impact on the predictive power performance of the models derived by classification algorithms. More experiments' results, and analysis are given in Section 4.6.

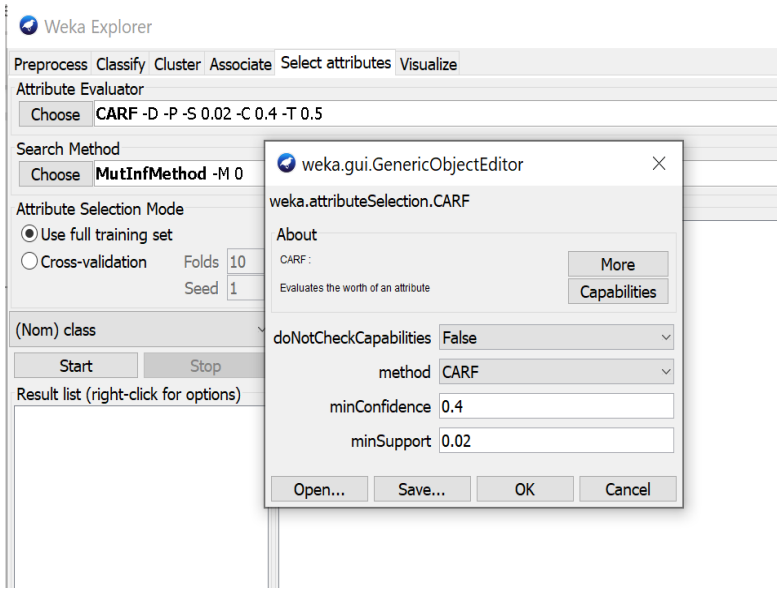

Figure 18: CARF UI

Selected features:
0.5232 33 thick
0.1968 5 hardness
0.0913 1 family
0.0611 3 steel
0.046 12 surface-quality

Figure 19a: Set of Ranked Features Produced by CARF from the 'Anneal' Dataset

Ranked features:
937.8061 1 family
905.7163 9 strength
866.7645 5 hardness
618.1328 33 thick
582.4798 3 steel
530.4829 12 surface-quality
385.0935 8 formability
384.9498 25 ferro
332.0928 20 chrom
274.4497 7 condition
260.141 10 non-ageing
256.7951 34 width
227.1319 6 temper_rolling
222.9967 24 exptl
144.0954 27 blue/bright/varn/clean
87.503 21 phos
83.376 13 enamelability

Figure 19b: Set of Ranked Features Produced by CST from the 'Anneal' Dataset

4.4 Performance Measures used for Testing

```
Ranked features:
 0.901   1 -0.343condition=?+0.342condition=S-0.31formability=?+0.279formability=2-0.196temper_rolling=T...
 0.83    2 -0.396family=TN+0.365family=?-0.312non-ageing=N-0.277formability=3-0.262ferro=Y...
 0.77    3 0.331steel=A+0.294hardness-0.292surface-quality=?+0.265surface-finish=?-0.265surface-finish=P...
 0.713   4 0.307surface-finish=?-0.307surface-finish=P-0.279blue/bright/varn/clean=B+0.27blue/bright/varn/clean=?-0.242strength
 0.664   5 0.456enamelability=?-0.456steel=V-0.358enamelability=2-0.278enamelability=1-0.234bw/me=?...
 0.621   6 -0.404formability=1-0.376steel=?-0.342condition=A-0.242surface-quality=G+0.241steel=A...
 0.58    7 -0.332bw/me=B-0.318packing=?+0.316bw/me=?+0.313surface-quality=E+0.304packing=3...
 0.546   8 -0.484oil=?+0.365oil=N+0.322oil=Y-0.207bw/me=B-0.195cbond=Y...
 0.513   9 0.307steel=M+0.306strength-0.246steel=?-0.238bore=600+0.233bore=0...
 0.483  10 -0.46packing=?+0.426packing=3+0.273surface-quality=D-0.249family=ZS-0.219surface-quality=E...
 0.455  11 0.3  family=ZS+0.286bore=0-0.27bore=600+0.24 carbon-0.228steel=W...
 0.429  12 -0.298packing=?+0.296oil=Y+0.291packing=3+0.258len+0.254bf=Y...
 0.404  13 -0.51surface-quality=F-0.427bf=Y+0.299steel=R+0.212formability=3+0.211surface-quality=D...
 0.38   14 0.376formability=5-0.299temper_rolling=T+0.293surface-quality=D+0.238family=ZS+0.237steel=M...
 0.356 150.496blue/bright/varn/clean=V+0.387steel=S+0.307blue/bright/varn/clean=C-0.275blue/bright/varn/clean=?-0.215surface-quality
 0.335  16 -0.322surface-quality=D+0.294oil=Y+0.288cbond=Y-0.239blue/bright/varn/clean=V+0.22 steel=K...
 0.315  17 -0.362blue/bright/varn/clean=C-0.341bt=Y+0.326steel=W-0.318steel=M+0.254steel=S...
 0.296  18 0.488steel=W+0.343blue/bright/varn/clean=C-0.32bt=Y-0.277steel=S-0.271steel=M...
 0.279  19 -0.302cbond=Y-0.302blue/bright/varn/clean=C-0.273len-0.271bt=Y+0.258bc=Y...
 0.263  20 -0.479bc=Y+0.381exptl=Y-0.356packing=2-0.264ferro=Y-0.234bore=500...
 0.246  21 0.571chrom=C-0.428ferro=Y+0.407packing=2+0.277exptl=Y-0.2lustre=Y...
 0.23   22 -0.545packing=2-0.418exptl=Y+0.379chrom=C+0.311bore=500-0.279ferro=Y...
 0.214  23 -0.77phos=P+0.332ferro=Y-0.266enamelability=1-0.243width+0.206enamelability=2...
 0.198  24 -0.732enamelability=1+0.571enamelability=2+0.301phos=P+0.119bc=Y-0.113ferro=Y...
 0.183  25 0.485bore=500+0.484exptl=Y-0.261chrom=C-0.228formability=5+0.192formability=3...
```

Figure 19c: Set of Ranked Features Produced by PCA from the 'Anneal' Dataset

All experiments will be run using the 10-fold cross-validation data resampling method in the Weka environment [177]. Hence the models produced by the classification algorithms are tested on different samples of the data ten times to produce the performance measures so users can decide to accept or reject models. Initially, the input dataset is split into ten partitions where the classification algorithm trains on nine partitions to generate the model, and the generated model is tested in terms of predictive power on the remaining partition. The process is repeated 10 times and error rate/classification accuracy computed at each time is averaged and provided to the end-user. Cross-validation resampling is used commonly in the machine learning community when evaluating models derived by classification algorithms since it often generates less biased results when compared with other testing methods [177]. The ten-fold cross validation method's steps are briefly summarized below:

1. Randomly shuffle the training dataset

2. Partition the training dataset into 10 parts with stratification (ensuring all class values appear in each part)
3. For each part:
    - Choose one part as testing dataset (hold out part)
    - Use the remaining nine parts for training the model
    - Evaluate the derived model's hold out part
    - Produce error rate / classification accuracy
4. Repeat the process (steps 1-3) 10 times
5. Aggregate the results and produce the average error rate / classification accuracy

To evaluate the effectiveness of the subset of features chosen by CARF, we adopted various performance measures related to supervised learning tasks, i.e. classification benchmarks. These methods include classification accuracy, precision, and recall (see Equations 4.1–4.3), which have been derived from the contingency table shown in Figure 22. The reason for choosing these evaluation measures because the effectiveness of the subsets of features will be tested using classification algorithms. The subsets of features of the considered feature selection methods, including ours, are chosen from binary and multi-class types of datasets as discussed in Section 4.6. Table 4.1 depicts the likely outcomes for a model's binary classification problem using the model's predicted value vs. the class true value [177]. In particular, the below four scores are calculated and visualized:

True Positive (TP) = data instances that are truly positive and were accurately predicted as positive by the classification algorithm.

True Negative (TN) = data instances that are truly negative and were accurately predicted as negative by the classification algorithm.

False Negative (FN) = data instances that are truly positive but were incorrectly predicted as negative by the classification algorithm.

False Positive (FP) = data instances that are truly negative but were incorrectly predicted as positive by the classification algorithm.

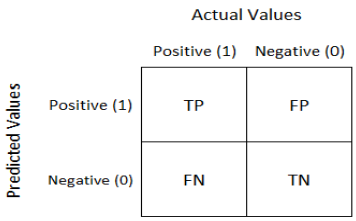

*Figure 20: Contingency Table Sho*wing Possible Outcome of a Binary Classification Problem

The accuracy, precision, and recall measures are described below.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (4.1)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (4.2)$$

$$\text{Recall} = \text{True Positive Rate (TPR)} = \frac{TP}{TP+FN} \quad (4.3)$$

Classification accuracy (Equation 4.1) is the most common evaluation measure in machine learning and represents the proportion of correct classifications from the size of the test dataset—in other words, how many times the test instances have been predicted correctly by the machine learning algorithm. However, in certain datasets, such as when the class labels' frequency varies

significantly (imbalanced dataset), classification accuracy may no longer be considered as a primary evaluation measure.

Other measures, including precision (Equation 4.2), and recall (Equation 4.3), can be used in addition to classification accuracy. Precision and recall are considered as quality and quantity metrics, respectively. Good precision in information retrieval is indicated when a query result has more relevant than irrelevant results; when the query results are mainly relevant, this indicates high recall. The same analogy can be seen in classification tasks within machine learning; precision counts the number of instances that are predicted to be positive and truly linked with the positive class, whereas recall counts the number of instances that are predicted to be positive from all truly positive data instances in the test dataset. Precision quantifies the number of positive class predictions that actually belong to the positive class. In other words, when the FPs are minimized, this maximizes precision, and when the FNs are minimized, this maximizes recall.

4.5 Settings, Methods Used and Data

All feature selection and classification experiments have been conducted in the Weka 3.8 environment where CARF and the MulInfMethod search methods' source codes have been integrated. We have contrasted CARF with three popular feature selection methods of type filtering, namely GR, CST, and ReliefF. These methods have been selected since they exhibit different schemes in the way they select features from classification benchmarks. CST evaluates the correlations between the feature and the target class label using expected and observed frequencies as per Equation 4.4. GR treats features equally and does not favor features with more possible values by dividing the information gained from the data by the entropy of the feature as shown in Equations 4.5–4.6. Lastly, ReliefF computes the merit of the feature by calculating the

difference of its nearest instance pairs and with respect to their class labels according to Equation 4.7. A hit is considered to be when the difference in the feature value and a neighboring instance pair with the same class label is seen, and the feature score gets reduced. A miss is considered to be when there is a difference between a feature and a neighboring instance pair with different labels and the feature score rises.

$$\text{CST}/X^2 = \frac{(O-E)^2}{E} \quad (4.4)$$

Where

O denotes the Observed Frequency, and *E* denotes the Expected Frequency, for the considered features' values.

$$\text{GR} = \frac{IG}{IntrinsicInfo(S,A)} \quad (4.5)$$

$$IntrinsicInfo(S,A) = -\sum \frac{S_i}{S} log_2 \frac{S_i}{S} \quad (4.6)$$

Where IG is the information gain and *IntrinsicInfo* is the Entropy of attribute 'A' over a set of examples 'S'.

$$\text{ReliefF} = W[A] = W[A] - \frac{\left(diff\frac{A,R_i,H}{m}\right)}{\left(diff\frac{A,R_i,M}{m}\right)} \quad (4.7)$$

Where,

W[A]= feature weights

A = number of features

m = number of random training instances from 'n' number of training instances used to update 'W'

R_i = randomly selected target instance

H/M = nearest hit and nearest miss

To assess the quality of the feature sets selected by the feature selection methods we have utilized two different classification algorithms namely decision tree (C4.5) and probabilistic (Bayes Net) [137,20]. These algorithms have been selected due to their applicability in various classification domains and the different learning methodologies they use in constructing the classification model. Specifically, the Bayes Net algorithm computes the probabilities of each possible class from a directed acyclic graph built by the algorithm to assign the largest probability class to the test data instance. The algorithm utilizes the chain rule by searching the graph to build a Bayes network using the conditional probabilities of the possible feature values in the test data and the information within the training dataset. The graph consists of nodes with each denoting a feature and arcs that represent how the node and its parents are correlated. The C4.5 constructs a tree structure as a classification model using different data quantification splitting metrics such as Shannon entropy [152]. The algorithm chooses the feature with the highest information gain as a root, splits the data using that feature's values, and repeats the same process until each path in the tree ends up in a leaf. Once this occurs, C4.5 then trims the tree using error-based estimation methods.

The default parameter setting of the Bayes Net and C4.5 algorithms within the Weka environment have been used. Specifically, the C4.5 algorithm used subtree-raising pruning and a confidence factor of 0.25 and set the minimum description length principle (MDL) to true. For Bayes Net, the kernel estimator used was "SimpleEstimator" with alpha=0.5 for calculating the conditional probabilities during building the model. The search method adopted by Bayes Net was hill climbing. For the proposed feature selection method, the minimum support and the minimum confidence were set to 2% and 40% respectively, following warming up experimentation and

according to previous research works in CAR mining, i.e. [166,167]. Lastly, all the experimental runs have been executed on a personal computer with a processor speed of 2.7 GHz and 8 GB Random Access Memory (RAM).

4.6 Results Analysis

4.6.1 Dimensionality Reduction Results

A total of 15 different datasets from the Kaggle and UCI data repositories [71,94] have been chosen to evaluate the goodness of the subsets of features chosen by CARF when contrasted with other common feature selection methods. Table 4.2 displays the datasets used in the experiments along with the characteristics of these datasets including, but not limited to, the dataset size in terms of the number of data instances, the number of features, whether the dataset is binary or multi-class, the class distribution variable, and primarily the feature selection results in terms of the number of retained features per dataset. The choice of selecting the classification datasets was made for several reasons:

1. They are publicly available for free
2. They have been used by previous researchers in machine learning for analysis
3. They represent different classification applications, for example medical diagnosis, cyber security, finance, engineering, and others
4. Some of the datasets contain missing values and others do not
5. They have different size in terms of number of data instances
6. The dimensionality varies significantly; for instance, ‘Cleve’ contains 12 features whereas ‘PD’ contains 755 features.

All continuous attributes within the selected datasets have been implicitly discretized during the feature assessment process. Moreover, any feature in the chosen datasets with missing values has been treated implicitly using the feature selection methods by 'ReplaceMissingValue' filter in Weka.

The last four columns of Table 4.2 show the dimensionality reduction results after applying the considered feature selection methods, including CARF, on the 15 datasets. It is apparent from the results derived, i.e. features' subsets, by the considered feature selection methods that CARF when employing the MutInfoMethod search method produced fewer features and for all the considered datasets when contrasted with the GR, ReliefF, and CST methods. For example, for the 'Cleve' low dimensional dataset (12 features), the GR, ReliefF, and CST methods derived 9, 10, and 9 subsets of features respectively, whereas CARF only retained 5 features reducing the search space by around 60%. More importantly, and for a large dimensional dataset such as 'Arrhythmia' which consists of 280 features, only 7 features were retained by CARF and this is substantially less than the subsets of features selected by the GR, ReliefF, and CST methods which are 118, 162, and 86 respectively.

Based on the results sets produced by the feature selection methods, CARF consistently selected fewer features regardless of whether the dataset was binary, multi-class, or even imbalanced. These results clearly show that CARF was not sensitive to data imbalance or to the type of the application domains to which the dataset belongs. The proposed feature selection method was not sensitive to noise such as missing values within the dataset or to the dimensionality level, e.g. the number of features. In fact, CARF significantly minimized the search space for the large dimensional datasets we consider including 'Arrhythmia', 'ECG Heath Categorization', 'Email Spam' and 'PD'. CARF

was able to satisfy the user with just 7, 17, 8, and 13 from the 'Arrhythmia', 'ECG Heath Categorization', 'Email Spam', and 'PD' datasets, respectively.

Table 4.2: Characteristics of the Datasets and the Subsets of Features Selected by Feature Selection Methods

Dataset	Number of Instances	Number of Features	Missing Values	Class Distribution	Number of features selected			
					GR	ReliefF	CST	CARF
Anneal	898	39	Yes	8:99:684:0:67:40	22	16	17	5
Arrhythmia	452	280	Yes	245:44:15:15:13:25:3:2:9:50:0:0:0:4:5:	118	162	86	17
Cleved	303	12	Yes	138:165	9	10	9	5
Colic	368	23	Yes	232:136	11	12	12	5
Credit-g	1000	21	No	700:300	12	14	11	3
Cylinder-bands	540	40	Yes	228:312	19	20	10	12
Dermatology	366	35	Yes	112:61:72:49:52:20	29	27	30	6
ECG Heartbeat Categorization	21892	188	No	18118:556:1448:162:1608	166	150	149	17
Email Spam	5172	3002	No	3672:1500	1606	1217	1146	16
Hepatitis	155	20	Yes	32:123	13	15	12	7
Ionosphere	351	35	No	126:225	32	33	32	8
Optdigits	5620	65	No	554:571:572:568:558:558:566:554:56	56	42	50	13
Parkinson's Disease (PD)	756	755	No	192:564	457	444	446	12
Sonar	208	61	No	97:111	20	46	20	9
Waveform	5000	41	No	1692:1653:1653	18	20	18	7
Wine	178	14	No	59:71:48	13	12	12	3

Figure 4.7 shows the minimization of the search space per dataset by the CARF feature selection method. The results have been calculated using the percentage difference in the selected features of the other considered feature selection methods and CARF. In all datasets the search space reduction difference is apparent as CARF continuously minimized the search space more than GR, ReliefF, and CST. The reduction of the search space by CARF was at least 41.67% in the case of the 'Hepatitis' dataset when compared with the CST method, and 99% in the case of high dimensional dataset like 'Email Spam'. The range of the search space reduction when adopting

CARF versus the other feature selection methods on the 15 datasets is between 41% and 99%. This shows that CARF only selects highly influential features with the least feature-to-feature correlations and therefore is able to reduce the search space substantially.

The way that CARF considers one item only per rule during the process of building the rules (which in turn is utilized for the features score calculation), has contributed to fewer features being retained. CARF favors the best attribute value associated with the rule during the induction process and discards all others ensuring that significant attribute values that have correlation with the class are those used for rule induction. More importantly, CARF ensures that once each rule is induced then its data instances are discarded, thus for the induced rules there is no chance of data overlapping. This process reduces feature-to-feature correlation and therefore minimizes results redundancy in terms of features retained during the feature assessment process.

In general, CARF has been effective in minimizing data dimensionality when compared to other feature selection methods such as GR, ReliefF, and CST. The new search methods, i.e. MutInfoMethod, when integrated with these feature selection methods, showed good performance

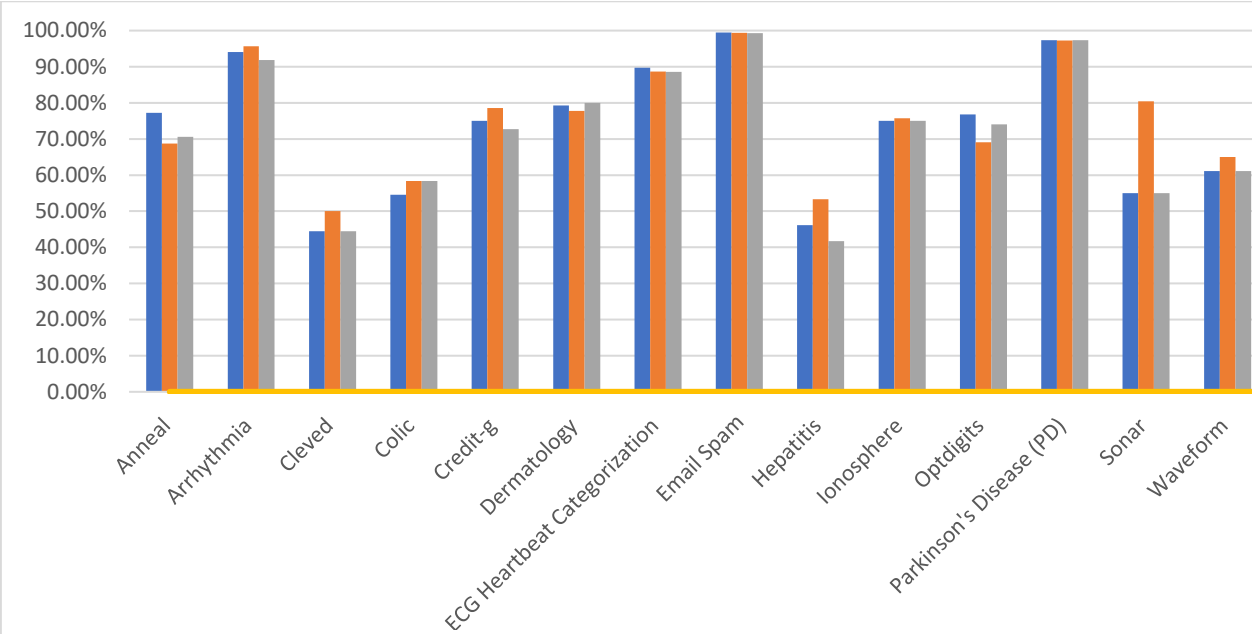

Figure 21: Search Space Reduction of CARF Compared with Other Feature Selection Methods

identifying cut-off values for users, including novices, to assess even if they have little knowledge about the dataset characteristics. In the next section we evaluate the performance of the retained features by the considered feature selection methods, including CARF, and reveal the goodness of the cut-offs suggested by the mutual information search method.

Overall, the CARF feature selection method was consistently able to offer much smaller subsets of features from the 15 datasets—this is highly beneficial for decision makers and for users to explain a dataset in a much more concise manner. In addition, offering fewer features empowers users as they will be able to control and understand them more easily than a large number of features as offered by the current feature selection methods. The intelligent cut-offs proposed by the MutInfoMethod search methods showed a significant decrease in the number of features that

can be retained by the end-user. In the next subsection we show the predictive power obtained by ML techniques when processing the retained features subsets.

4.6.2 Predictive Accuracy, Precision, and Recall Results Analysis

We evaluate the quality of the subsets generated by CARF, GR, ReliefF, and CST feature selection methods by investigating classifiers generated against these subsets of features with two common classification algorithms, namely Bayes Net and C4.5. Table 4.3 shows the performance measure results of the classifiers produced by the Bayes Net algorithm against the subsets of features selected by CARF, GR, ReliefF, and CST from the 15 datasets and with respect to classification accuracy, recall, and precision. The last 5 columns of table 4.4 shows the dimensionality reduction for each method. You can see from the below result set that CARF significantly reduce the dimensionality without sacrificing the accuracy.

Table 4.3: Bayes Net Algorithm Results on the Considered Feature Selection Methods

	Accuracy				Recall				Precision				Number of Features Selected			
Dataset	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**
Anneal	95.88	94.77	96.21	94.99	95.90	94.80	96.20	95.00	96.80	96.20	97.00	95.30	22	16	17	5
Arrhythmia	69.91	71.46	71.02	63.94	69.90	71.50	71.00	63.90	66.70	70.10	69.30	57.40	118	162	86	17
Cleve	83.50	83.50	84.16	83.17	83.50	83.50	84.20	83.20	83.50	83.50	84.20	83.20	9	10	9	5
Colic	82.61	81.79	82.07	82.88	82.60	81.80	82.10	82.90	82.70	81.80	82.10	82.70	11	12	12	5
Credit-g	73.30	74.80	74.00	71.20	73.30	74.80	74.00	71.20	72.00	74.00	72.90	69.10	12	14	11	3
Dermatology	98.09	98.09	97.81	89.07	98.10	98.10	97.80	89.10	98.10	98.10	97.90	90.10	29	27	30	6
ECG Heartbeat Categorization	62.39	62.03	62.03	67.43	62.40	62.00	62.00	67.40	86.10	86.10	86.10	85.10	166	150	149	17
Email Spam	92.69	91.84	91.84	83.60	92.70	91.80	91.80	83.60	92.60	91.75	91.80	88.50	1606	1217	1146	16
Hepatitis	83.23	83.87	83.87	83.23	83.20	83.90	83.90	83.20	84.50	85.40	84.90	82.20	13	15	12	7
Ionosphere	89.74	89.46	89.46	90.88	89.70	89.50	89.50	90.90	89.70	89.40	89.40	90.80	32	33	32	8
Optdigits	92.19	91.76	92.22	83.02	92.20	91.80	92.20	83.00	92.40	92.00	92.40	83.10	56	42	50	13
Parkinson's Disease (PD)	75.93	77.91	76.85	81.22	75.90	77.90	76.90	81.20	79.30	80.20	79.60	81.60	457	444	446	12
Sonar	80.29	79.81	79.81	74.04	80.30	79.80	79.80	74.00	80.50	80.00	80.00	74.30	20	46	20	9
Waveform	79.86	79.84	79.86	79.30	79.90	79.80	79.90	79.30	83.10	83.10	83.10	81.10	18	20	18	7
Wine	98.88	98.31	98.31	94.94	98.90	98.30	98.30	94.90	98.90	98.40	98.40	95.00	13	12	12	3

The predictive accuracy results show that Bayes Net was able to construct more predictive models from the small subsets of features selected by the proposed feature selection method on 4 out of the 15 datasets we consider. To be more specific, for the 'Colic', 'ECG Heartbeat Categorization', 'Ionosphere', and 'PD' datasets, the Bayes Net algorithm was able to derive classifiers from CARF's data subsets with0.27%, 1.09%, 0.82%; 5.04%, 5.40%, 5.39%; 1.14%, 1.42%, 1.42%; and 5.29%, 3.31%, 4.36% higher accuracies respectively than those derived by the same algorithm from the GR, ReliefF, and CST method subsets. These results, especially for high dimensional datasets such as 'PD' and 'ECG Heartbeat Categorization', are promising particularly when cutting down the search space significantly. CARF was able to reduce the number of features of 'Colic', 'ECG Heartbeat Categorization', 'Ionosphere', and 'PD' when coupled with the MutInfoMethod search method to 5, 17, 8, and 12 features, respectively. Conversely, GR, ReliefF, and CST reduced

the number of features from the same datasets to 11, 166, 32, 457; 12, 150, 32, 446; and 12, 149, 32, 446; respectively. These results indeed reveal the goodness of the subsets of features offered by CARF and showed that in many cases not only the search space was minimized by CARF, but also the quality of the classifiers derived from these subsets are of high quality at least in terms of predictive accuracy, recall, and precision measures. Overall, the search space on these datasets has been reduced massively when CARF was applied compared to using the GR, ReliefF, and CST methods.

In general, the performance results in regard to predictive accuracy showed consistency when Bayes Net algorithms processed the feature subsets of the considered feature selection methods on the 15 datasets. There was a drop of approximately 9% when Bayes Net processed the subsets of CARF from the 'Dermatology' dataset, and the same pattern was noticed in the C4.5 results (Table 4.4). We investigated this case and noticed that the dataset is multi-class in nature (consists of more than two class labels) and contains primarily linear features and just one categorical feature. When compared to GR, ReliefF, and CST, CARF was able to reduce the search space of this dataset by 79.31%, 77.78%, and 80.00% respectively, identifying just 6 features out of 34 using the cut-off value suggested by the MutInfoMethod search method. Three of the 6 features identified by CARF are of high significance, i.e. 'thinning_of_the_suprapapillary_epidermis', 'band-like_infiltrate', and 'fibrosis_of_the_papillary_dermis', and three of lowsignificance, i.e. 'disappearance_of_the_granular_layer', 'melanin_incontinence', and 'perifollicular_parakeratosis' as shown in Figure 24. Whereas the other feature selection methods retained 29, 27, and 30 features, respectively, many of which have low yet close scores. For this dataset, it seems that a large number of clinical features with little differences contribute minimally

to the diagnosis of 'Erythemato-squamous' disease in dermatology. Since this disease overlaps in many histopathological features, and CARF retains only non-overlapping features, then many of these overlapping attributes are not retained by the CARF method. Another probable reason could be associated with the progression of the disease and that features at one stage can be insignificant but later on can be significant, thus the longitudinal property may play a role in feature assessment.

Table 4.4 shows the performance results of the C4.5 algorithm with respect to classification accuracy, recall, and precision against the subsets of features chosen by GR, Relief=F, CST, and CARF from the 15 datasets. Based on the predictive accuracy results of C4.5, this algorithm produced better performance when it processed CARF feature subsets on 6 out of the 15 datasets we consider, namely 'Cleve', 'Credit-g', 'Hepatitis', 'Ionosphere', 'Waveform', and 'Wine'. For the remaining 9 datasets, C4.5 produced more accurate classifiers from one or more of the other

```
Ranked features:
 0.4521   22 thinning_of_the_suprapapillary_epidermis
 0.2319   33 band-like_infiltrate
 0.1242   15 fibrosis_of_the_papillary_dermis
 0.0469   26 disappearance_of_the_granular_layer
 0.0323   12 melanin_incontinence
 0.0296   31 perifollicular_parakeratosis
```

Figure 22: Features Selected by CARF from the 'Dermatology' Dataset

features' subsets selected by GR, ReliefF, or CST. For the 'Cleve', 'Credit-g', 'Hepatitis', 'Ionosphere', 'Waveform' and 'Wine' datasets, the C4.5 algorithm was able to generate classifiers from CARF's data subsets with 1.65%, 0.67%, 1.66%; 0.20%, 0.30%, 0.30%; 1.94%, 0.00%, 1.29%; 0.86%, 0.86%, 0.29%; 0.50%, 0.26%, 0.58%; and 1.69%, 1.69%, 1.69%)higher accuracies than those derived by the same algorithm from the GR, ReliefF, and CST methods respectively.

The last 5 columns of table 4.4 shows the dimensionality reduction for each method. You can see from the below result set that CARF significantly reduce the dimensionality without sacrificing the accuracy.

Table 4.4: The C4.5 Algorithm Results on the Considered Feature Selection Methods

	Accuracy				Recall				Precision				Number of Features Selected			
Dataset	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**	**GR**	**ReliefF**	**CST**	**CARF**
Anneal	98.10	98.66	98.10	96.10	98.10	98.70	98.10	96.10	98.10	98.70	98.10	96.20	22	16	17	5
Arrhythmia	64.15	65.04	64.15	62.83	64.20	65.00	64.20	62.80	62.20	61.60	62.70	58.80	118	162	86	17
Cleve	77.23	78.21	77.22	78.88	77.20	78.20	77.20	78.90	77.20	78.20	77.20	78.90	9	10	9	5
Colic	85.86	85.59	85.59	83.97	85.90	85.60	85.60	84.00	86.00	85.80	85.80	83.80	11	12	12	5
Credit-g	71.70	71.60	71.60	71.90	71.70	71.60	71.60	71.90	70.60	70.00	70.10	69.90	12	14	11	3
Dermatology	94.71	93.71	94.26	86.89	93.70	93.70	94.30	86.90	93.70	93.70	94.20	87.90	29	27	30	6
ECG Heartbeat Categorization	94.85	94.89	94.91	93.90	94.90	94.90	94.90	93.90	94.60	94.70	94.70	93.50	166	150	149	17
Email Spam	92.69	91.84	91.84	83.60	92.70	91.80	91.80	83.60	92.60	91.75	91.80	88.50	1606	1217	1146	16
Hepatitis	80.64	82.58	81.29	82.58	80.60	82.60	81.30	82.60	78.70	80.90	79.30	80.60	13	15	12	7
Ionosphere	91.45	91.45	92.02	92.31	91.50	91.50	92.00	92.30	91.50	91.60	92.30	92.50	32	33	32	8
Optdigits	90.69	90.46	90.60	84.29	90.70	90.50	90.60	84.30	90.70	90.50	90.60	84.20	56	42	50	13
Parkinson's Disease (PD)	83.33	82.93	82.93	82.14	83.30	82.90	82.90	82.10	83.20	83.20	82.90	81.30	457	444	446	12
Sonar	74.04	73.55	76.92	72.12	74.00	73.60	76.90	72.10	74.00	73.50	76.90	72.20	20	46	20	9
Waveform	76.32	76.56	76.24	76.82	76.30	76.60	76.20	76.80	76.30	76.60	76.20	76.80	18	20	18	7
Wine	93.82	93.82	93.82	95.51	93.80	93.80	93.80	95.50	94.00	94.00	94.00	95.60	13	12	12	3

Tables 4.3 and 4.4 show the recall and precision results produced by the C4.5 and Bayes Net algorithms from the subsets selected by the feature selection methods on the 15 datasets using the suggested cut-offs of the MutInfoMethod search method. To clarify, and for each dataset, the subsets of features based on the computed cut-offs were retained to be processed by C4.5 and Bayes Net to derive classifiers. The precision and recall results are consistent with the predictive accuracy results derived using both C4.5 and Bayes Net. To be specific, the won-tied-lost record of the C4.5 classifiers' performance with respect to recall rates when processing the CARF-

selected features sets are contrasted with those of GR, ReliefF, and CST, are 6-0-9, 5-1-9, and 6-0-9 respectively. The won-tied-lost record remains unchanged for the precision results obtained by the same classification algorithm. Furthermore, the won-tied-lost records of the Bayes Net classifiers' performance with respect to recall rates when processing the CARF selected features sets and contrasted with those of GR, ReliefF, and CST, are 5-1-9, 5-0-10 and 5-0-10 respectively. The won-tied-lost record remains unchanged for the precision results obtained by the same classification algorithm.

A series of one-sample t tests were performed to investigate whether the CARF's accuracy, recall, and precision values differed form a calculated mean based on the sum of GR, ReliefF, CST, and CARF. On accuracy, CARF's values are statistically significant at the 0.01 level on four datasets: Dermatology, ECG Heartbeat- Categorization, Email Spam, and Parkinson's Disease. On two of the datasets, the increase in accuracy on CARF is statistically significant. Those are ECG Heartbeat- Categorization, and Parkinson's Disease. On two of the datasets, the decrease in accuracy on CARF is statistically significant. Those include Dermatology, and Email Spam. On the remaining eleven datasets, the difference in accuracy in CARF did not significantly differ from the mean of all methods indicating a statistically insignificant difference.

Concerning Recall, CARF's values are statistically different from the average of all methods on four datasets. On ECG Heartbeat- Categorization and Parkinson's Disease, the increase on Recall associated with CARF is not due to chance. By the same token, Dermatology, and Email Spam Recall decreases observed on CARF in comparison to the mean of other methods are judged to be statistically significant. On the remaining eleven datasets, the mean difference between CARF, and the average of other methods are found to be statistically insignificant at the 0.01 level.

Considering the precision of the four methods, CARF performs better than the other methods on two datasets and underperforms on two datasets given the average of all methods as a reference metric. On ECG Heartbeat- Categorization and Parkinson's Disease datasets, CARF values are statistically different from the mean of all methods recording a slight improvement not due to chance. Simultaneously, CARF precision values on Dermatology, and Email Spam are statistically different from the mean of all methods indicating that the decrease in precision is real, and sampling error or chance have little to do with it. On remaining eleven datasets, however, CARF values do not differ systematically from the mean of all methods, and the four methods are said to yield similar precision values. Table 4.5 shows the p-value statistical significance of CARF based on Table 4.3 Bayes Net algorithm accuracy, recall and precision results on all of the 15 datasets

Table 4.5: Statistical Significance of CARF based on Bayes Net algorithm results

Datasets	Accuracy	Recall	Precision
Anneal	0.08	0.14	0.19
Arrhythmia	0.13	0.19	0.21
Cleve	0.11	0.14	0.18
Colic	0.12	0.16	0.17
Credit-g	0.09	0.14	0.17
Dermatology	0.01 *	0.01*	0.01*
ECG Heartbeat Categorization	0.01 *	0.01*	0.01*
Email Spam	0.01*	0.01*	0.01*
Hepatitis	0.14	0.17	0.12
Ionosphere	0.11	0.13	0.15
Optdigits	0.17	0.12	0.16
Parkinson's Disease (PD)	0.01*	0.01*	0.01*
Sonar	0.13	0.17	0.21
Waveform	0.09	0.08	0.15
Wine	0.13	0.18	0.21

The differences in statistical significance among the datasets stem from many plausible sources. First, the nature of the datasets differs, and the measurement scales of features within each of them may have caused the increase or decrease in accuracy, recall, and precision. For instance, some

datasets possess dependent variables with many values while others reflect binary measurements of outcomes. Further, in many datasets, one may find linear combinations of variables already included within such data. This may have caused some fluctuations in estimating the coefficients on accuracy, recall, and precision. In addition, many features within datasets consist of minimal variability. This limits the ability of statistical estimation to detect empirical associations causing biases in the calculation of accuracy, recall, and precision.

Another culprit causing differences between CARF, and other methods in terms of generating varying accuracy, recall, and precision is data imbalance. In many of the datasets, the properties of the domain cause a natural data imbalance due to the nature of occurrence of events on a variable or feature. Some events or values occur more frequently compared to others, and in turn influence the predictive modelling ability of statistical estimation. Moreover, sampling error, and measurements biases account for some of the differences in values among the four methods. In many datasets, many values are overrepresented or underrepresented in few variables causing bias in estimating robust measurements on accuracy, recall, and precision.

Missing values present another source of bias causing differences in accuracy, recall, and precision between CARF, and other methods. In some data sets, missing values are large, in other datasets, missing completely at random data is not satisfied, and missing values are potentially concentrated in sub-samples within the given data. Imputation of missing values may result in differing results from those observed and reported. The type of imputation could also skew the estimates of accuracy, recall, and precision in one way or another. All such changes to be performed on missing values by Weka would carry significant consequences on the estimation of predictive modelling goodness or fit.

The orthogonality of the datasets also is potentially responsible for the differing accuracy, recall, and precision values indicated. Some datasets have scales, indexes, and summated rating averages, which are combinations of existing features. CARF does not perform as well as the other methods when sums or linear scales exist within the dataset. Therefore, on the ill-performing datasets observed, many features included were not oblique. CARF performs better when features are independent of each other, and correlations amongst them are low producing high factorial structures.

For instance, there was a notable drop of approximately 9% when Bayes Net processed the subsets of CARF from the 'Dermatology' dataset, and the same pattern was noticed in the C4.5 results. We investigated this case and noticed that the dataset is multi-class in nature (consists of more than two class labels) and contains primarily linear features and just one categorical feature. For this dataset, it seems that a large number of clinical features with little variability contributed minimally to the diagnosis of 'Erythemato-squamous' disease in dermatology. Since this disease overlaps in many histopathological features, and CARF retains only non-overlapping features, many overlapping attributes are not retained by the CARF method. Another probable reason could be associated with the progression of the disease, and that features at one stage can be insignificant, but later on, it can be significant. Thus, the longitudinal property may play a role in feature assessment.

The obtained results of recall and precision rates are consistent and pinpoint that both classification algorithms were able to identify the proportion of positive predictions that were actually positive from the different subsets of features offered using the cut-off values of the MutInfoMethod search methods, and for all considered feature selection methods. This indeed shows the effectiveness of the proposed search procedure, not only in cutting down the search

space of datasets considered, but also in offering high impactful features and for all feature selection methods. This can provide the below key competencies, especially when integrated with the CARF feature selection method:

- Substantial reduction in the search space of the dataset
- Few different features are retained yet with competitive performance in terms of recall, precision, and accuracy
- Expert and novice users can exploit the few number of features easily
- Simpler classification models are offered when processing the few features which can be easily understood by users.

Overall, when employing the MutInfoMethod search method, CARF was superior to the other feature selection methods in minimizing the search space of the various datasets considered. More importantly, models derived against the subsets of features selected by CARF from two different machine learning algorithms showed good performance with respect to precision, recall, and accuracy when compared with models derived by the same algorithms against much larger features sets, i.e. sets selected by other feature selection methods. These results revealed that CARF was able to remove feature-to-feature similarity due to the non-overlapping induction process it adopts in growing and pruning the rules, which is subsequently employed to assign scores to the available features in the training dataset.

4.7 Chapter Summary

In this chapter we presented the implementation, testing, and validation of the proposed feature selection and search methods, i.e. CARF and MutInfoMethod. Specifically, we show the UIs for

CARF and the MutInfoMethod and how to set up its primary parameters prior to experimentation. We show how CARF was integrated within the Weka environment to enable reusability of different existing methods and evaluation measures without having to re-code everything from scratch. This chapter also introduced evaluation measures such as recall, precision, and predictive accuracy in the experiments as well as in the datasets, and characteristics such as their size, number of features, type, and class balance status.

In the next chapter we highlight the conclusions of the dissertation.

Chapter Five

5 Conclusions

A key factor that influences the quality of the classification outcome in machine learning tasks is the choice of a relevant set of features from the input dataset for data processing. This process, known as feature selection, aims to discard irrelevant features as early as possible and then offer the smallest subset of relevant features to the learning algorithm to simplify the learning phase and improve the performance. This research investigated different issues related to an important feature selection approach named Filtering. It utilizes mathematical models to determine each feature's merits during data pre-processing and leaves the difficult task of choosing the final subset of features to the end-user. These issues included, but were not limited to, the reduction of the similarity among retained subsets of features by the Filtering approach, computing an indicative measure that can differentiate between relevant and irrelevant features, and providing the end-user with fewer yet influential subsets of features to improve the manual process of selecting features.

In response to the aforementioned issues, which have been explained in detail in Chapter One, this research proposed a new Filtering method called CARF which is based on inducing simple rules from the classification datasets. CARF is coupled with a new cut-off procedure called MutInfMethod which calculates an indicative threshold to help the user identify the number of features. The CARF method learns rules with 1-item in the dataset and from these computes weights that are intelligently assigned to the features in the dataset. The transformation of the data format from LS to IS during the rule discovery process makes the proposed method simple and efficient. The distinguishing characteristic of the CARF method is that it erases redundant rules as early as possible; this has ensured that the rules generated are useful. We think that will be

advantageous as fewer features will be retained, making the process more efficient. We also investigated the vital issue in feature selection of reducing the time for checking the results of Filtering methods and have thus proposed MutInfMethod—an automatically generated cut-off threshold procedure to distinguish between features. This new cut-off procedure suggests the number of features to be chosen by the end-user without the results having to be manually checked.

In Chapter Four, we show the implementation of CARF and MutInfMethods in Java and their integration into the Weka open source machine learning platform. More importantly, we reveal the true perfromance of CARF on a large number of real datasets with 12–3002 available features and compare its performance with other known Filtering methods using machine learning algorithms.

Experimentation using different datasets from the Kaggle and UCI data repositories, three common feature selection methods, and two classification algorithms, has been executed to show the true performance of the proposed feature selection method. The classification algorithms were adopted to derive classifiers from the different feature sets recommended by the considered feature selection methods. The results analysis shows that CARF continuously selects fewer features than GR, ReliefF, and CST on all datasets considered reducing the search space significantly and with a relative difference of 41%–99%. The classifiers produced by the C4.5 and Bayes Net algorithms showed that the subsets of features retained by CARF when using the proposed cut-off procedure are highly competitive in terms of recall, precision, and accuracy rates when contrasted with models selected by the other feature selection methods. These empirical results reveal that despite fewer features being retained by CARF, the models derived from machine learning algorithms against these subsets maintained competitive predictive power.

Additionally, the experimental results showed that the cut-off procedure is independent from any feature selection methods, and when integrated with methods such as GR, CST, ReliefF, and

CARF, computes a cut-off that recommends fewer features—this can be useful for the end-user in understanding the dataset and its impactful features. These cut-offs that are computed by the proposed method in an automated manner are helpful for users to decide which features to select, relaxing the complicated process of filtering the results of feature selection. In the case of CARF and the new cut-off procedure, the search space of features was reduced for most of the considered datasets and without influencing the models' performance when classification algorithms were applied. In addition, the search space minimization impacted the models derived positively by simplifying them for decision makers and users, making them easy to understand and manage when compared with models derived from a large number of features.

References

1. N. Abdelhamid, A. Ayesh, and W. Hadi. Multi-label rules algorithm based associative classification. *Parallel Processing Letters, 24*(01), 2014.
2. N. Abdelhamid, A. Ayesh, F. Thabtah, S. Ahmadi, and W. Hadi, W. MAC: A multiclass associative classification algorithm. *Journal of Information and Knowledge Management* (*JIKM). 11*(2), pages 1250011-1–1250011-10. WorldScinet, 2012.
3. N. Abdelhamid, and F. Thabtah. Associative classification approaches: Review and comparison. *Journal of Information and Knowledge Management (JIKM). 13*(3) 1450027, 2014.
4. N. Abdelhamid, F. Thabtah and H. Abdel-jaber. Phishing detection: A recent intelligent machine learning comparison based on models content and features. *2017 IEEE International Conference on Intelligence and Security Informatics (ISI).* https://doi.org/10.1109/ISI.2017.8004877). Beijing, China. IEEE Explore digital Library, 2017.
5. T. Abeel, T. Helleputte, Y. Peer, P. Dupont, and Y. Saeys. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. *Bioinformatics*, pages 392–398, 2009.
6. H. Abusamra. A comparative study of feature selection and classification methods for gene expression data. *King Abdullah University of Science and Technology Thuwal, Kingdom of Saudi Arabia,* 2013.
7. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Verkamo. Fast discovery of association rules. *Advances in Knowledge Discovery and Data Mining, 12*(1), pages 307–328, 1996.
8. S. Alasadi and W. Bhaya. Review of data pre-processing techniques in data mining. *Journal of Engineering and Applied Sciences, 12*, pages 4102–4107, 2017.
9. M. Angelillo, F. Balducci, D. Impedovo, G. Pirlo, and G. Vessio. Attentional pattern classification for automatic dementia detection. IEEE Access, 7, pages 57706–57716, 2019.
10. M. Antonie and O. Zaïane. *An associative classifier based on positive and negative rules.* Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, pages 64 – 69, 2004.
11. I. Aqra, T. Herawaan, N. Ghani, A. Akhunzada, A. Ali, R. Razali, M. Ilahi and K-W. Choo, K-W. A novel association rule mining approach using TID intermediate itemset. *PLoS One 13*(1), 2018.
12. S. Bahassine, A. Madani, A. M. Al-Saremand M. Kissi. Feature selection using an improved chi-square for Arabic text classification. *Journal of King Saud University – Computer and Information Sciences, 32*, pages 225–231, 2020.
13. E. Baralis, S. Chiusano, and P. Graza. A lazy approach to associative classification. *IEEE Transactions on Knowledge and Data Engineering, 20*(2). *ISSN:* 10414347, 2018.

14. P. Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Transactional Information Theory 44*(2), pages 525–536, 1998.
15. J. Benesty, J. Chen, Y. Huang, and I. Cohen. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*, pages 37–40. Springer, 2009.
16. I. Bentolila, Y. Zhou, I. Ismail, and R. Humpleman. *System and method for behavioral model clustering in television usage, targeted advertising via model clustering, and preference programming based on behavioral model*, 2013.
17. C. Bharath and J. Ganesh. A proposal to enhance cross-marketing via social networks by modelling and analyzing market basket like patterns across social network groups. *IEEE International Workshop on: Business Applications of Social Network Analysis (BASNA)*, pages 1-4. IEEE, 2010.
18. C. Bhatt and M. Kankanhalli. Multimedia data mining: State of the art and challenges. *Journal of Multimedia Tool Applications*, pages 35–76, 2011.
19. A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence, 97*(1-2), pages 245–271, 1997.
20. R. Bouckaert. Bayesian network classifiers in Weka. (Working paper series. University of Waikato, Department of Computer Science. No. 14/2004). Hamilton, New Zealand: University of Waikato, 2004.
21. T. Brick, R. Koffer, D. Gerstorf, and N. Ram. Feature selection methods for optimal design of studies for developmental inquiry. *Journals of Gerontology: Psychological Sciences*, pages 113–123, 2018.
22. T. Brosch and R. Tam. Manifold learning of brain MRIs by deep learning. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 633–640. Berlin & Heidelberg: Springer, 2013.
23. R. Bunker and F. Thabtah. A machine learning framework for sport result prediction. *Applied Computing and Informatics, 15*(1), pages 27-33, 2019.
24. Y. Cai, T. Huang, L. Hu, X. Shi, L. Xie, and Y. Li. Prediction of lysine ubiquitination with mRMR feature selection and analysis. *Amino Acids*, pages 1387-1395, 2012.
25. L. Chen, X. Huo, and G. Agrawal. Accelerating MapReduce on a coupled CPU-GPU architecture. *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1-11. IEEE, 2012.
26. M. Cherrington, J. Lu, D. Airehrour, F. Thabtah, Q. Xu, and S. Madanian. *Particle swarm optimization for feature selection: A review of filter-based classification to identify challenges and opportunities*. In Proceedings of IEEE IEMCON, pages 0523-0529. Vancouver, Canada, 2019a.
27. M. Cherrington, F. Thabtah, J. Lu, and Q. Xu. *Feature selection: Filter methods performance challenges.* IEEE 2019 International Conference on Computer and Information Sciences (ICCIS), 1-4. Aljouf, KSA. April, 2019b.
28. W. Cohen. Fast effective rule induction. *In: Prieditis A, Russell S (eds) Proceedings of the 12th International Conference on Machine Learning, ICML*, pages 115–123. Tahoe City. Morgan Kaufmann, 1995.

29. W. Cohen. Efficient pruning methods for separate-and-conquer rule learning systems. *In Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 988-994. France, 1993.

30. C. Cortes and V. Vapnik. Support vector machines. *Machine Learning, 20*(3), pages 273–297, 1995.

31. Z. Da-wei, W. Geng-feng, and H. Min. Improved clonal selection algorithm. *Journal of Computer Engineering and Design, 30*(11), pages 2741–2748, 2009.

32. A. Dabney. Classification of microarrays to nearest centroids. *Bioinformatics, 21*(22), pages 4148–4154, 2005.

33. D. Campbell and S. Campbell. Introduction to regression. *StatLab Workshop Series*, pages 1–15, 2008.

34. C. De Sá, C. Soares, and A. Knobbe. Entropy-based discretization methods for ranking data. *Information Sciences, 329*, pages 921–936, 2016.

35. C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *J. Bioinform. Comput. Biol. 3*, pages 185–205, 2005.

36. M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, pages 83–92, 2008.

37. E. Duchesnay, A. Cachia, N. Boddaert, C. Nadia, J-F. Mangin, J-L. Martinot, F. Brunelle, and M. Zilbovicius. Feature selection and classification of imbalanced datasets application to PET images of children with autistic spectrum disorders. *NeuroImage*, pages 1003–14, 2011.

38. R. Duda and P. Hart. Naive Bayes. *Pattern classification and scene analysis, 3,* 1973.

39. O. Embarak. *Data analysis and visualization using Python: Analyze data to create visualizations for BI systems.* Apress, 2018.

40. G. Feng, G. Huang, Q. Lin, R. Gay. Error minimized extreme learning machine with growth of hidden nodes and incremental learning. *IEEE Transactional Neural Network 20*(8), pages 1352–1357, 2009.

41. F. Fleuret and I. Guyon. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research, 5*, pages 1531–1555, 2004.

42. J. Forst, A. Tombros, and T. Rölleke. Less is more: Maximal marginal relevance as a summarisation feature. *Second International Conference on the Theory of Information Retrieval, ICTIR*, pages 350–353. Cambridge, UK, 2009.

43. P. Foschi, D. Kolippakkam, H. Liu, and A. Mandvikar. Feature extraction for image mining. *Multimedia Information Systems*, pages 56–69, 2002.

44. Y. Freund and R. Schapire. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research, 4*, pages 933–969, 2003.

45. R. Ge, M. Zhou, Y. Luo, Q. Meng, G. Mai, D. Ma, G. Wang, and F. Zhou. McTwo: A two-step feature selection algorithm based on maximal information coefficient. *BMC Bioinformatics, 17*, Article 142, 2016.

46. A. Ghosh, B. Dhara, and R. De. Selection of genes mediating certain cancers, using a neuro-fuzzy approach. *Neurocomputing, 133*, pages 122–140, 2014.

47. Q. Gu, Z. Li, and J. Han. Generalized Fisher score for feature selection. *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 266–273, 2011.

48. H. Güçdemir and H. Selim. Integrating multi-criteria decision making and clustering for business customer segmentation. *Industrial Management & Data Systems*, 2015.

49. R. Guercke, C. Brenner, and M. Sester. Data integration and generalization for SDI in a grid computing framework. *Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 3-–1. Beijing, China, 2008.

50. X. Guo, K. Dominick, A. Minai, H. Li, C. Erickson, and L. Lu. Diagnosing autism spectrum sisorder from brain resting-state functional connectivity patterns using a deep neural network with a novel feature selection method. *Frontiers in Neuroscience*, pages 1–19, 2017.

51. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, pages 389–422, 2002.

52. M. Hall. *Correlation-based feature selection for machine learning*. [PhD Thesis]. New Zealand Department of Computer Science, Waikato University, 1999.

53. M. Hall, G. Holmes, and E. Frank. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter 11(1)*, pages 10–18, 2009.

54. T. Hamed, R. Dara, and S. Kremer. An accurate, fast embedded feature selection for SVMs. *International Conference on Machine Learning and Applications*, pages 135–140, 2014.

55. S. Hammoud. *A MapReduce classifier based on association rule data mining.* [PhD Thesis], Brunel University, 2010.

56. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 1–12, 2000.

57. J. Hernandez, B. Duval, J-K. Hao, E. Marchiori, J. Moore, and J. Rajapakse. A genetic embedded approach for gene selection and classification of microarray data. *EvoBIO-07, Lecture Notes in Computer Science, 4447*, pages 90–101.Springer, 2007.

58. T. Ho. Random forest - Document analysis and recognition. *Proceedings of the Third International Conference, 1*, pages 278-282, 1995.

59. J. Hodges and E. Lehmann. KNN - Some applications of the Cramer-Rao inequality. *In Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 233–238. The Regents of the University of California, 1951.

60. J. Huang and C. Ling. Constructing new and better evaluation measures for machine learning. *International Joint Conference on Artificial Intelligence*, pages 859–864. Chicago, 2007.

61. A. Hyvarinen. Fast ICA for noisy data using Gaussian moments. *Proceedings of the IEEE International Symposium on Circuits and Systems VLSI (Cat. No. 99CH36349), 5*, pages 57–61. IEEE, 1999.

62. M. Jalil, F. Mohd, and M. Noor. A comparative study to evaluate filtering methods for crime data feature selection. *International Conference on Computer Science and Computational Intelligence*, pages 114–120. Bali: Science Direct, 2017.

63. R. Jenke, A. Peer, and M. Buss. Feature extraction and selection for emotion recognition from EEG. *IEEE Transactions on Affective Computing, 5*, pages 327–339, 2014.

64. B. Jiang, C. Ye, and J. Liu. Non-parametric K-sample tests via dynamic slicing. *Journal of the American Statistical Association,* 2014.

65. S-Y. Jiang, and L-X. Wang. Efficient feature selection based on correlation measure between continuous and discrete features. *Information Processing Letters, 116*(2), pages 203–215, 2016.

66. X. Jin, A. Xu, R. Bie, and P. Guo. Machine learning techniques and chi-square feature selection for cancer classification using SAGE gene expression profiles. *Data Mining for Biomedical Applications: PAKDD 2006 Workshop*, pages 106–115. Singapore: BioDM, 2006.

67. S. Jishan, R. Rashu, N. Haque, and R. Rahman. Improving accuracy of students' final grade prediction model using optimal equal width binning and synthetic minority over-sampling technique. *Decision Analytics, 2*(1), *1,* 2015.

68. R. Joshi. *Accuracy, precision, recall & F1 score: Interpretation of performance measures,* 2016. Retrieved from https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

69. C. Juliano. *Machine learning: What developers and business analysts need to know,* 2018. Retrieved from https://www.infoworld.com/article/3259512/machine-learning-what-developers-and-business-analysts-need-to-know.html

70. B. Jullien, Y. Lefouili, and M. Riordan. Privacy protection and consumer retention, 2018. Retrieved from https://www.tse-fr.eu/sites/default/files/TSE/documents/doc/wp/2018/wp_tse_947.pdf

71. Kaggle, 2020. https://www.kaggle.com/datasets [accessed June 5, 2019]

72. F. Kamalov and F. Thabtah. *A feature selection method based on ranked vector scores of features for classification*. Annals of Data Science, pages 1–20. Springer, 2017.

73. P. Kamavisdar, S. Saluja, and S. Agrawal. A survey on image classification approaches and techniques. *International Journal of Advanced Research in Computer and Communication Engineering, 2*(1), pages 1005–1009, 2013.

74. A. Karegowda and M. Jayaram. Comparative study of attribute selection using gain ratio and correlation based feature selection. *International Journal of Information Technology and Knowledge Management*, pages 271–277, 2010.

75. I. Katakis, G. Tsoumakas, and I. Vlahavas. *Dynamic feature space and incremental feature selection for the classification of textual data streams*. In: ECML/PKDD-2006 international workshop on knowledge discovery from data stream, pages 107–116, 2006.

76. M. Kaur and S. Kang. Market basket analysis: Identify the changing trends of market data using association rule mining. *Procedia Computer Science, 85*, pages 78–85, 2016.

77. T. Kaur, B. Saini, and S. Gupta. A novel feature selection method for brain tumor MR image classification based on the Fisher criterion and parameter-free Bat optimization. *Neural Computing and Applications, 29*(8), pages 193–206, 2018.

78. U. Khaireand and R. Dhanalakshmib. Stability of feature selection algorithm: A review. *Journal of King Saud University - Computer and Information Sciences,* 2019. https://doi.org/10.1016/j.jksuci.2019.06.012.

79. S. Khalid, T. Khalil, and S. Nasreen. A survey of feature selection and feature extraction techniques in machine learning. *Science and Information Conference,* 2014. https://doi.org/10.1109/SAI.2014.6918213.

80. T. Khoshgoftaar, A. Fazelpour, H. Wan, and R. Wald. A survey of stability analysis of feature subset selection techniques. *IEEE 14th International Conference on Information Reuse & Integration (IRI),* pages 424–431. San Francisco, CA, 2013.

81. K. Kira and L. Rendell. A practical approach to feature selection. *In Machine Learning Proceedings,* pages 249–256, 1992.

82. S. Kotsiantis, D. Kanellopoulos, and P. Pintelas. Data preprocessing for supervised learning. *International Journal of Computer Science, 1*, pages 111–117, 2006.

83. T. Kumbhare and S. Chobe. An overview of association rule mining. *International Journal of Computer Science and Information Technologies, 5*, pages 927–930, 2014.

84. Lartillot, O., & Toiviainen, P. (2007). *A Matlab toolbox for musical feature extraction from audio*. In Proceedings of the 10th International Conference on Digital Audio Effects (DAFx-07), 10–15.

85. E. Learned-Miller. *Introduction to supervised learning*. Pages 1–5, 2014.

86. S. Le Cessie and J. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, pages 191–201, 1992.

87. S. Lei. A feature selection method based on information gain and genetic algorithm. *International Conference on Computer Science and Electronics Engineering*, pages 355–358, 2012.

88. S. Levy, M. Duda, N. Haber, and D. Wall. Sparsifying machine learning models identify stable subsets of predictive features for behavioral detection of autism. *MolecularAutism*, 8, 65, 2017.

89. Y. Li, M. Dong, and J. Hua. Localized feature selection for clustering. *Pattern Recognition Letters, 29*(1), pages 10–18, 2008.

90. W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple-class association rule. ***Proceedings of the*** *IEEE International Conference on Data Mining – ICDM*, pages 369–376, 2001.

91. G. Li, Y. Hu, H. Chen, L. Shen, H. Li, M. Hu, and K. Sun, K. . An improved fault detection method for incipient centrifugal chiller faults using the PCA-R-SVDD algorithm. *Energy and Buildings, 116*, pages 104–113. Mladenić, 2016.

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

93. C. Li and J. Xu. Feature selection with the Fisher score followed by the Maximal Clique Centrality algorithm can accurately identify the hub genes of hepatocellular carcinoma. *Scientific Reports,* 9, Article 17283, 2019.

94. M. Setiono. *UCI machine learning repository.* University of California, School of Information and Computer Science, 2013. [http://archive.ics.uci.edu/ml].

95. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. *Proceedings of the Knowledge Discovery and Data Mining Conference- KDD*, pages 80–86. New York, 1998.

96. B. Liu, Y. Ma, and C-K Wong. Classification using association rules: weakness and enhancements. *In Vipin Kumar, et al., (eds) Data mining for scientific applications*, 2001.

97. L. Liu, R. Mehl, W. Wang, and Q. Chen. Applications of the Hilbert-Huang transform for microtremor data analysis enhancement. *Journal of Earth Science, 26*(6), pages 799–806, 2015.

98. H. Liu and R. Setiono. Chi2: Feature Selection and Discretization of Numeric Attributes. *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pages 388–391. Herndon, 1995.

99. A. Lorberfeld. *Machine learning algorithms in layman's terms, Part 1,* 2019. Retrieved from https://towardsdatascience.com/machine-learning-algorithms-in-laymans-terms-part-1-d0368d769a7b

100. J. Loughrey and P. Cunningham. Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets. *International Conference on Innovative Techniques and Applications of Artificial Intelligence,* pages 33–43. London: Springer, 2004.

101. J. Lucas, A. Laurent, M. Moreno, and M. Teisseire. fuzzy associative classification approach for recommender systems. *Int. J. Uncertainty Fuzziness Knowl.-Based Syst., 20*(4), pages 579–617, 2012.

102. A. Mabrouk, C. Gonzales, K. Jabet-Chevalier, and E. Chojnaki. Multivariate cluster-based discretization for Bayesian network structure learning. *In International Conference on Scalable Uncertainty Management*, pages 155–169. Springer, 2015.

103. O. Maimon and L. Rokach. Introduction to soft computing for knowledge discovery and data mining. *Soft Computing for Knowledge Discovery and Data Mining*, pages 1–13, 2008.

104. A. Makode and A. Kapse. Knowledge discovery using various multimedia data mining technique. *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)*, pages 1138–1141, 2015.

105. S. Maldonado and J. López, J. An embedded feature selection approach for support vector classification via second-order cone programming. *Intelligent Data Analysis, 19*(6), pages 1259–1273, 2015.

106. S. Maldonado and J. Lopez. Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for SVM classification. In *Applied Soft Computing, 67,* pages 94–105, 2018.

107. M. Mandal and A. Mukhopadhyay. An improved minimum redundancy maximum relevance approach for feature selection in gene expression data. *Procedia Technology, 10*, pages 20–27, 2013.

108. S. Marsland. *Machine Learning: An algorithmic perspective, Second Edition.* CRC Press, 2014.

109. R. Martis, U. Acharya, K. Mandana, A. Ray, and C. Chakraborty. Cardiac decision making using higher order spectra. *Biomedical Signal Processing and Control, 8*(2), pages 193–203, 2013.

110. Microsoft. *Introduction to Machine Learning and ML.NET Part 1*, 2015. Retrieved from https://social.technet.microsoft.com/wiki/contents/articles/53298.introduction-to-machine-learning-and-ml-net-part-1.aspx

111. R. Milos, G. Mohamed F. Nenad, and Z. Obradovic. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinformatics*, 2017. https://doi.org.10.1186/s12859-016-1423-9.

112. H. Min and W. Fangfang. *Filter-wrapper hybrid method on feature selection.* Intelligent Systems (GCIS) 2010 Second WRI Global Congress, 3, pages 98–101, Dec 2010.

113. D. Mladenić. *Feature subset selection in text-learning.* European Conference on Machine Learning, pages 95–100, 1998.

114. D. Modha and W. Spangler. Feature weighting in k-means clustering. *Machine Learning*, pages 217–237, 2003. https://doi.org/10.1023/A:1024016609528.

115. S. More and D. Mishra. Multimedia data mining: A survey. *Prathibha: International Journal of Science, Spirituality, Business and Technology (IJSSBT), 1*(1), pages 2277–7261, 2012.

116. M. Nashwan and S, Shahid. Symmetrical uncertainty and random forest for the evaluation of gridded precipitation and temperature data. *Atmospheric Research, 230*, pages 1–30, 2019.

117. A. Naik and S. Pathan. Weather classification and forecasting using back propagation feed-forward neural network. *International Journal of Scientific and Research Publications, 2*(12), pages 1–3, 2012.

118. A. Navot, L. Shpigelman, N. Tishby, and E. Vaadia. Nearest neighbour-based feature selection for regression and its application to neural activity. *Advances in Neural Information Processing Systems, 18,* pages 995–1002, 2006.

119. D. Nguyen, B. Vo, and B. Le. CCAR: An efficient method for mining class association rules with itemset constraints. *Eng. Appl. Artif. Intell., 37*, pages115–124, 2015.

120. H. Noh. Frontier estimation using kernel smoothing estimators with data transformation. *Journal of the Korean Statistical Society, 43*, pages 503–512, 2014.

121. J. Novaković, P. Strbac, P. and D. Bulatović. Towards optimum feature selection using Ranking method and classification algorithm. *Yugoslav Journal of Operations Research*, pages 119–135, 2011.

122. H. Obaid, S. Ahmed Dheyab, and S. Sabah Sabry. The impact of data pre-processing techniques and dimensionality reduction on the accuracy of machine learning. *Annual*

Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), pages 279–283. Jaipur, 2019.

123. A. Onan. A fuzzy-rough nearest neighbor classifier combined with consistency-based subset evaluation and instance selection for automated diagnosis of breast cancer. *Expert Systems with Applications, 42*(20), pages 6844–6852, 2015.

124. F. Padillo, J. Luna, and S. Ventura. Evaluating associative classification algorithms for big data. *Big Data Anal., 4(*1), page 2, 2019.

125. A. Patil, C. Deshmukh, and A. Panat. *Feature extraction of EEG for emotion recognition using Hjorth features and higher order crossings*. In 2016 Conference on Advances in Signal Processing (CASP) pages 429–434. IEEE, 2016.

126. P. Patil, S. Thube, B. Ratnaparkhi, and K. Rajeswari. Analysis of different data mining tools using classification, clustering and association rule mining. *International Journal of Computer Applications, 93*, pages 35–39, 2014.

127. K. Pavya and B. Srinivasan. Enhancing wrapper-based algorithms for selecting optimal features from thyroid disease dataset. *Computer Science,* 2018.

128. S. Pearlman. *What is data integration*, 2019. Retrieved from https://www.talend.com/resources/what-is-data-integration

129. K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2*(11), pages 559–572, 1901.

130. K. Pearson. Notes on the history of correlation. *Biometrika, 13*(1), pages 25–45, 1920.

131. X. Peng and D. Xu. A local information-based feature-selection algorithm for data regression. *Pattern Recognition, 46*(2), pages 519–2530, 2013.

132. A. Pereira and E. Hruschka. Simultaneous co-clustering and learning to address the cold start problem in recommender systems. *Knowledge-Based Systems, 82*, pages 11–19, 2015.

133. M. Piao, Y. Piao, and J. Lee,. Symmetrical uncertainty-based feature subset generation and ensemble learning for electricity customer classification. *Symmetry*, 1–11, 2019.

134. B. Picinbono and P. Chevalier. Widely linear estimation with complex data. *IEEE Transactions on Signal Processing, 43*(8), pages 2030–2033, 1995.

135. K. Poole, and H. Rosenthal. A spatial model for legislative roll call analysis. *American Journal of Political Science, 29*(2), pages 357–384, 1985.

136. I. Portugal, P. Alencar, and D. Cowan. The use of machine learning algorithms in recommender systems: A systematic review. *Expert Systems with Applications*, pages 205-227, 2018.

137. J. Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann, 1993.

138. J. Quinlan. Induction of decision trees. *Machine Learning*, pages 81–106, 1986.

139. K. Rajab. New hybrid features selection method: A case study on websites phishing. *Security and Communication Networks*, 2017. https://doi.org/10.1155/2017/9838169

140. M. Rajab, and D. Wang. Filter methods practical challenges for feature selection. *Journal of Information and Knowledge Management*, pages 1–12, 2020.
141. E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, pages 2232–2248, 2009.
142. M. Robnik-Šikonja, I. Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning 53*, pages 23–69 (2003). https://doi.org/10.1023/A:1025667309714
143. F. Rosenblatt. Multi-layered perceptron & artificial neural networks. *The Perceptron, a Perceiving and Recognizing Automaton Project Para, Cornell Aeronautical Laboratory*, 1957.
144. D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing*, pages 318–362, 1986.
145. A. Saber, A. Al-Zoghby, and S. Elmougy. Big-data aggregating, linking, integrating and representing using semantic web technologies. *The International Conference on Advanced Machine Learning Technologies and Applications* pages 331–342. Springer, 2018.
146. D. Saravanan. *Effective video content retrieval using image attribute standards.* Hyderabad, India: EAI Endorsed Transactions on Energy Web, 2018.
147. D. Sargsyan, S. Jagannatha, N. Manyakov, A. Skalkin, A., Bangerter, S. Ness, K. Durham, D. Amaratunga, J. Cabrera, and G. Pandina. Feature selection with weighted importance index in an autism spectrum disorder study. *Statistics in Biopharmaceutical Research*, pages 118–125, 2019.
148. R. Sathya and A. Abraham. Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence, 2*,(3) pages 4–38, 2013.
149. M. Savić, V. Kurbalija, M. Ivanović, and Z. Bosnić. *A feature selection method based on feature correlation networks*. In: Y. Ouhammou, M. Ivanovic, A. Abelló, & L. Bellatreche, (eds) Model and Data Engineering. MEDI 2017. Lecture Notes in Computer Science, (10563). Springer, Cham., 2017.
150. J. Senders, P. Staples, A. Karhade, M. Zaki, W. Gormley, M. Broekman, T. Smith, and O. Arnaout. *Machine learning and neurosurgical outcome prediction: A systematic review*, pages 1–11, 2017.
151. B. Senliol, G. Gulgezen, L. Yu, and Z. Cataltepe. Fast correlation based filter (FCBF) with a different search strategy. *Computer and Information Sciences*, pages 1–4, 2008.
152. C. Shannon. The mathematical theory of communication. *Bell Syst. Tech. J. 27*, pages 379–423, 1948.
153. X. Shen, X. Gong, Y. Cai, Y. Guo, J. Tu, H. Li, and Z. Zhu. Normalization and integration of large-scale metabolomics data using support vector regression. *Metabolomics, 12*(5), 89, 2016.
154. A. Shen, R. Tong, and Y. Deng. Application of classification models on credit card fraud detection. *International Conference on Service Systems and Service Management*, pages 1–4. IEEE, 2007.

155. B. Singh, N. Kushwaha, and O. Vyas. A feature subset selection technique for high dimensional data using symmetric uncertainty. *Journal of Data Analysis and Information Processing,* 2014. https://doi.org/10.4236/jdaip.2014.24012.

156. S. Srivastava. Weka: A tool for data preprocessing, classification, ensemble, clustering and association rule mining. *International Journal of Computer Applications, 88,* pages 26–29, 2014.

157. G. Sunab, J. Liab, J. Daiab, Z. Songa, and F. Lang. Feature selection for IoT based on maximal information coefficient. *Future Generation Computer Systems*, pages 606–616, 2018.

158. A. Swalin. Choosing the right metric for evaluating machine learning models, 2018. Retrieved from https://medium.com/usf-msds/choosing-the-right-metric-for-evaluating-machine-learning-models-part-2-86d5649a5428

159. S. Syed Ibrahim, K. Chandran, and R. Nataraj. LLAC: Lazy Learning in Associative Classification in the Springer Lecture Series in Communications in Computer and Information Science (CCIS), Advances in Communication and Computers, 190, Part I, pages 631–638, 2011.

160. A. Tallón-Ballesteros, J. Riquelme, and R. Ruiz Merging subsets of attributes to improve a hybrid consistency-based filter: A case of study in product unit neural networks. *Connection Science, 28*(3), pages 242–257, 2016.

161. J. Tang, S. Alelyani, and H. Liu. Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, pages 115–148, 2014.

162. M. Taradeha, M. Mafarja, A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, and H. Fujita. An evolutionary gravitational search-based feature selection. *Information Sciences,* pages 219–239, 2019.

163. D. Tayal and K. Meena. A new MapReduce solution for associative classification to handle scalability and skewness in vertical data structure. *Future Generation Computer Systems, 103*, pages 44–57, 2020.

164. F. Thabtah, N. Abdelhamid, and D. Peebles. A machine learning autism classification based on logistic regression analysis. *Health Information Science and Systems*, pages 7–12, 2019.

165. F. Thabtah, P. Cowling, and Y. Peng. MCAR: multi-class classification based on association rule. *In The 3rd ACS/IEEE International Conference on Computer Systems and Applications,* page 33. IEEE, 2005.

166. F. Thabtah, and S. Hammoud. Parallel and distributed single and multi-label associative classification data mining frameworks-based MapReduce. *Journal of Parallel Processing Letter*. *Parallel Process. Lett.* 25, 1550002. World Scientific, 2015.

167. F. Thabtah, S. Hammoud, and H. Abdeljaber. MR-ARM: A MapReduce association rule mining. *Journal of Parallel Processing Letter,* 23, 1350012. World Scientific, 2013.

168. Thabtah, F., Kamalov, F., Hammoud, and S. Shahamiri. A new feature selection method based on simplified observed and expected likelihoods distance. *Information Science,* 2020.

169. F. Thabtah, F. Kamalov, and K. Rajabc. A new computational intelligence approach to detect autistic features for autism screening. *International Journal of Medical Informatics*, pages 1386–56, 2018.

170. F. Thabtah, and D. Peebles. A new machine learning model based on induction of rules for autism detection. *Health Informatics Journal*, pages 1–23, 2019.

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

172. D. Tong. Extracting informative genes from unprocessed microarray data. *In 2010 International Conference on Machine Learning and Cybernetics, 1*, pages 439–443. IEEE, 2010.

173. T. Uno, T. Asai, Y. Uchida, and H. Arimura. LCM: An efficient algorithm for enumerating frequent closed item sets. *Fimi, 90*, 2003.

174. R. Urbanowicz, M. Meeker, M. Cava, R. Olson, and J. Moore. Relief-based feature selection: Introduction and review. *Journal of Biomedical Informatics*, pages 189–203, 2018.

175. H. Wang, T. Khoshgoftaar, and A. Napolitano. An empirical study on wrapper-based feature selection for software engineering data. *International Conference on Machine Learning and Applications*, pages 84–89. Miami: IEEE Computer Society, 2013.

176. S. Wang, Y. Yuan, C. Zhu, D. Kong, and Y. Wang. Discrimination of polycyclic aromatic hydrocarbons based on fluorescence spectrometry coupled with CS-SVM. *Measurement, 139*, pages 475–481, 2019.

177. I. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques, 2005.

178. A. Wosiak and D. Zakrzewska. Integrating correlation-based feature selection and clustering for improved cardiovascular disease diagnosis. *Overcoming 'Big Data' Barriers in Machine Learning Techniques for the Real-Life Applications* 2018. https://doi.org/10.1155/2018/2520706.

179. X. Wu, K. Yu, H. Wang, W. Ding. *Online streaming feature selection*. International Conference on Machine Learning, Haifa, Israel, pages 1159–1166, 2010.

180. X. Xue, M. Yao, and Z. Wu. A novel ensemble-based wrapper method for feature selection using extreme learning machine and genetic algorithm. *Knowledge and Information Systems*, pages 389–412, 2018.

181. H. Xuegang, P. Zhou, P. Li, and J. Wang. A survey on online feature selection with streaming features. *Frontiers of Computer Science*, 2016. https://doi.org/10.1007/s11704-016-5489-3.

182. K. Yan and D. Zhang. Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B: Chemical, 212*, pages 353–363, 2015.

183. C. Yang, S Prasher, J. Landry, and A. DiTommaso. Application of artificial neural networks in image recognition and classification of crop and weeds. *Canadian Agricultural Engineering, 42*(3), pages 147–152, 2000.

184. L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res. 5*, pages 1205–1224, 2014a.
185. L. Yu H. Liu. *Redundancy based feature selection for microarray data*. KDD '04 Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 737–742, 2014b.
186. M. Zaki and C. Hsiao. CHARM: An efficient algorithm for closed itemset mining. P*roceedings of the 2002 Siam International Conference on Data Mining (SDM'02)*, pages 457–473, 2002.
187. M. Zaki and K. Gouda. Fast vertical mining using diffsets. *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 326–335, 2003.
188. E. Zdravevski, P. Lameski, R. Mingov, A. Kulakov, D. Gjorgjevikj. Robust histogram-based feature engineering of time series data. *Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 381–388). IEEE, 2015.
189. M. Zena and D. Gillies. A review of feature selection and feature extraction methods applied on microarray data. *Hindawi Publishing Corporation Advances in Bioinformatics*, pages 1–13, 2015.
190. G. Zeng. A unified definition of mutual information with applications in machine learning. *Mathematical Problems in Engineering*, pages 1–12, 2015.
191. Z. Zhao, R. Anand, and M. Wang. *Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform.* Cornell University, 2019.
192. X. Zheng, M. Wang, and J. Ordieres-Meré. Comparison of data preprocessing approaches for applying deep learning to human activity recognition in the context of industry 4.0. *Sensors*, pages 1–13, 2018.
193. J. Zhi-wei, W. Geng-feng, and H. Min. Feature selection based on adaptive genetic algorithm and SVM. *Journal of Computer Engineering*, pages 200–203, 2009.
194. J. Zhou, D. Foster, R. Stine, and L. Ungar. Streaming feature selection using alpha investing. KDD '05 Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pages 384-–93, 2005.
195. H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistics Society*, pages 301–320, 2005.