Feature Selection in Classification Tasks

by
Farinaz Z. Pisheh

A dissertation submitted to the Department of Computer Science,

College of Natural Sciences and Mathematics

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Computer Science

Chair of Committee: Ricardo Vilalta

Committee Member: Christoph F. Eick

Committee Member: S. Stephen Huang

Committee Member: Hongyu Guo

University of Houston
December 2019

ACKNOWLEDGMENTS

During my Ph.D. journey, I have received support and encouragement from a number of individuals. My sincere gratitude goes to my advisor, Dr. Ricardo Vilalta for his patient supervision and valuable advice. Also, I would like to thank Dr. Christoph Eick, Dr. Hongyu Guo, and Dr. Stephen Huang, members of my advisory committee, for giving me helpful suggestions on my research.

I am deeply indebted to the Bisheh Family; Alex, Gaelan, Mary, Sahar, Chris, Natalie, and Jasmine. On my first day in the US, they accepted me into their family with open arms and love. I am grateful to them for always being there for me. Alex and Gaelan have been tremendous mentors to me, and their daughters have treated me as their sister.

Finally, my deep and sincere gratitude to my family; this journey would not have been possible without their support. I am extremely grateful to my parents, Mahin and Mostafa, for their love, prayers, care and sacrifices. They always encourage me to explore new directions in life and pursue my dreams. I am very much thankful to my brother, Alireza, for his empathy, constant encouragement and of course, his wonderful sense of humor!

ABSTRACT

Feature subset selection is an essential pre-processing task in machine learning and pattern recognition. In supervised learning, the goal of feature selection is to select the smallest subset of features that can predict the target class with high generalization performance, e.g., with high accuracy. Moreover, feature selection can also help to avoid over-fitting, reduce computational costs, and shorten training time. In this study, we introduce a new filter-based feature selection algorithm named EBFS, that is inspired by Relieff, a well-known distance-based method. Like Relieff, we use the nearest neighborhood of each instance to find the local weight of each feature. Unlike Relieff – and other filter-based methods that focus on mutual information, conditional mutual information, or interaction information – this study looks at the entropy of feature values in local neighborhoods effectively capturing information not used in previous distance-based methods. We introduce a new heterogeneous ensemble method based on Relieff that varies the size of the neighborhood and uses statistical hypothesis tests to find the number of relevant features. Both algorithms were tested on multiple datasets; results show the effectiveness of our approach when compared to other well-known methods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

1.1 Machine learning

Machine learning is a research area within the field of computer science; it is a sub-field of artificial intelligence in which algorithms are developed to make machines capable of learning patterns and structures in data and/or to induce models for prediction. Machine learning is applied to different structured and unstructured data and is extensively used in many areas, such as agriculture, finance, health, astronomy, education, and more. Roughly speaking, the main tasks in machine learning are divided into two categories: supervised and unsupervised learning [27]. In supervised learning, a set of labeled data (called the training data) is provided to the machine. The labeled data comprises examples, where each example contains a number of features and a target value. By applying machine learning algorithms to the training data, the machine learns a function that maps the space of features to the space of target values. The mapping function can then be used to predict the target value for new examples characterized by features. Classification is the most well-known task in supervised learning. In this task, all of the labeled values (classes) in the training set are known. The labels can be nominal, ordinal, intervals, or ratios. In unsupervised learning, in contrast, the labels or target values are not known; the algorithm aims to find the structure of unlabeled data based on the input features, normally by dividing examples into several categories or clusters.

1.1.1 Data preprocessing

No matter what task machine learning is used for, it is crucial to prepare the training data before invoking a learning algorithm. Depending on the data, several steps will be taken in the preprocessing phase. These steps include data cleaning, data transformation, and data reduction [22].

Data cleaning is responsible for dealing with errors in the data; this step needs to fill in missing values, resolve inconsistencies, and decide about outliers and noisy data. Data transformation

consists of feature aggregation/decomposition, scaling, and normalization of the data in accordance with the goal of the project. Finally, data selection focuses on selecting and extracting features that will help to predict the target label with higher accuracy and/or provide better performance. The next section discusses feature selection in greater detail.

1.1.2 Feature subset selection

Feature subset selection methods aim at identifying the smallest subset of features that preserves the true nature of the given data [41, 24]. In classification tasks, this means finding an optimal subset of features that are relevant to the target class. The main focus of these algorithms is to distinguish between relevant and irrelevant features according to some criterion.

1.2 Types of features in a dataset

During this process, it is important to identify the following groups of features [35]:

- Features that are strongly relevant to the target class: In many information-theoretic methods [12], an individual feature is considered strongly relevant to the target class when it shares a high degree of mutual information (or some derivation from mutual information, such as symmetrical uncertainty) with the class. Removing such features from the feature set will cause a noticeable drop in accuracy.

- Features that are weakly relevant: A feature is weakly relevant when it is not strongly relevant to the class but belongs to a subset of features that taken together are highly relevant to the class. Interactive features fall into this category.

- Features that are irrelevant: These are features that are neither strongly nor weakly relevant. It is expected that these features will be removed from the final optimal subset of features.

1.3 Types of feature selection algorithms

Feature selection algorithms can be divided into three models: filters, wrappers, and embedded methods [7, 66, 67]. Filter-based algorithms rely on the basic characteristics of the training data, and their performance is independent of the classifier, i.e., these algorithms do not classify the training data during the process of evaluating the selected features. Wrapper-based methods, in contrast, optimize a predictor as part of the selection process using different search strategies. Embedded methods perform feature selection in the process of training and are usually specific to the given learning machine [22, 24].

In the process of searching for the best subset of features, feature selection algorithms aim to remove irrelevant and redundant features. Irrelevant features add no useful information when predicting the class. Redundant features ought to be removed because other – previously chosen – features carry the same information about the target; including such features does not improve upon the final selected subset. However, there also exist features that, taken individually, might seem irrelevant but that are highly correlated with the target when they are combined with one or more other features. These features are called interactive features, and identifying them is a great challenge.

1.4 Research goal

Developing an algorithm to find an optimal subset of features for classification is still a challenging task. Several algorithms have been proposed; however, more innovation is needed to improve current state-of-the-art. Regarding filter-based methods, it is evident that most algorithms that use information theory are successful at detecting redundant features, while they are incapable of detecting high-order feature interactions. Distance-based methods, on the other hand, are better at detecting interaction but rather weak in recognizing redundant features. Moreover, setting the parameters correctly for each of these algorithms is an additional task that can have high impact on performance results.

Another issue with many of these algorithms is that while they normally return features ranked according to quality, it is not clear how many of the top features should be selected. In this study,

we address some of these issues. First, we introduce a new filter-based feature selection algorithm named EBFS, that resembles Relieff [39, 49], a well-known distance-based method. Like Relieff, our approach uses the nearest neighbors of each instance to find the local weight of each feature. The sum of the weights of a feature in all the selected neighborhoods gives the final weight of that feature. Unlike Relieff, our method uses an information-theoretical measurement to choose features. To the best of our knowledge, this is the first time that an algorithm uses both distance and information theory to find relevant features. While many filter-based methods focus on the relation between the target class and one or more features (mutual information, conditional mutual information, or interaction information), our approach looks at the entropy of feature values in local neighborhoods, effectively capturing information not used in previous distance-based methods. Experimental results on real datasets show the effectiveness of our approach to detect and identify relevant features.

We introduce another algorithm, also based on Relieff. Here, we use heterogeneous ensemble methods with Relieff while varying the number of nearest-neighbors. To aggregate results, we use a statistical approach and a post-hoc test, in such a way that our algorithm returns not only the final order of features according to their importance and relevance to the target label, but it also selects the final set of relevant features. We applied this method on several datasets, and the results show that the hypothesis-statistical method works efficiently. These statistical methods are used in the feature selection field for the first time and might be useful in other ensemble methods for feature subset selection.

1.5 Dissertation layout

The layout of this dissertation is as follows:

- **Chapter 2** presents a background and includes several feature selection techniques.
- **Chapter 3** explains the methodology used in this research.
- **Chapter 4** presents experiments and results.

- **Chapter 5** contains a summary of our work and suggestions for future work.

2 Related work

2.1 Basics of information theory

Feature subset selection methods aim at identifying the smallest subset of features that preserve the true nature of the given data. In classification tasks, this means finding an optimal subset of features that are relevant to the target class.

This section reviews some of the most important measurements that are used in filter based feature selection methods. The fundamental measure in IT is the entropy [52] of a discrete variable X:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i) \tag{1}$$

where X is a random variable, n is the number of unique values for X, and $P(x_i)$ is the probability of each possible value x_i. Entropy quantifies the amount of uncertainty in a random variable X (in bits). $H(X)$ is maximum when all $P(x_i)$ have the same value; in that case $H(X) = \log_2 n$. The term $\frac{H(X)}{\log_2 n} \in [0, 1]$ is referred to as *normalized entropy*. Additional measures are defined next. Let Y be a random variable with m unique discrete values. The joint entropy [52, 12] of X and Y is defined as:

$$H(X, Y) = -\sum_{i=1}^{n} \sum_{j=1}^{m} P(x_i, y_j) \log_2 P(x_i, y_j) \tag{2}$$

where joint entropy is less than or equal to the sum of the individual entropy values of X and Y.

Conditional entropy [12] is defined using the definition of joint entropy:

$$H(Y|X) = H(X, Y) - H(X) \tag{3}$$

where the entropy of Y given X shows the decrease in the entropy of Y when knowing the entropy of X.

The concepts of entropy and conditional entropy allow us to introduce mutual information [12], which quantifies the amount of information shared by the two variables in terms of entropy and joint entropy:

$$\begin{aligned} MI(X,Y) &\equiv H(X) + H(Y) - H(X,Y) \\ &\equiv H(X) - H(X|Y) \equiv H(Y) - H(Y|X) \end{aligned} \tag{4}$$

Jakulin [31, 30] presented this measure as a two-way interaction between two variables that can reduce the uncertainty about one of the two variables given knowledge about the other. Several forms of normalized MI have been proposed. One popular form in feature selection algorithm is as follows [12]:

$$U(X,Y) = 2\frac{MI(X,Y)}{H(X) + H(Y)} \tag{5}$$

Conditional mutual information [12] can be defined for three discrete random variables. Having three variables, X, Y and Z, where Z has q unique discrete values, the following shows the conditional mutual information of X and Y given Z:

$$\begin{gathered} MI(X;Y|Z) = \\ \sum_{k=1}^{q} P(z_k) \sum_{i=1}^{n} \sum_{j=1}^{m} P(x_i, y_j|z_k) \log_2 \frac{P(x_i,y_j|z_k)}{P(x_i|z_k)P(y_j|z_k)} \end{gathered} \tag{6}$$

Conditional MI [12] can be obtained using entropy, joint entropy, and conditional entropy:

$$\begin{aligned} MI(X;Y|Z) &= H(X|Z) + H(Y,Z) - H(X|Y,Z) \\ &\equiv H(X,Z) + H(Y,Z) - H(X,Y,Z) - H(Z) \end{aligned} \tag{7}$$

This measure is always non-negative. If it is zero, this means that X and Y are unrelated in the context of Z or that Z completely explains the association between X and Y. The amount of information that is common to all three variables can be quantified by introducing interaction information about them. This measurement is the result of expanding the mutual information [31]:

$$I(X;Y;Z) = I(X;Y|Z) - I(X;Y)$$

In [30] the authors formulate the k-way interaction information for all the attributes in v where $|V| = k$:

$$II(V) \stackrel{\Delta}{=} -\sum_{T \subseteq V} (-1)^{|V|-|T|} H(T) \tag{8}$$

In this paper, when we discuss interaction, we only take positive interaction into consideration.

When this value is positive it shows synergy [31], or information that is greater than the sum of the information about the individual variables. The parity function provides an example of synergy; in the parity between two features, the sum of the information that $X1$ and $X2$ individually provide is different from the positive interaction between them. On the other hand, a negative interaction value indicates that the features provide redundant information. The next section introduces some of the well-known algorithms for feature selection.

2.2 Algorithms based on information theory

Mutual information is commonly used to measure the average dependency between a feature and the target class; features with high MI have a greater chance to be included in the final subset of features. A popular forward feature selection technique using MI is called MRMR [46]; to avoid redundancy, the algorithm penalizes high values of MI between two features. Another algorithm is called CMIM [19]; it looks for features corresponding to high $MI(X_i, Y|X_j)$ where X_j is a previously selected feature, Y is the target class, and X_i is a candidate feature. In this way, the algorithm discards features bearing high information about the target class, but that have similar information as previously selected features. Feature selection based on Joint Mutual Information (JMI) [62] evaluates the quality of a candidate feature X_i, by joining it to each of the previously selected features $\{X_j\}$ and measuring $MI(X_i, X_j|Y)$.

Previous work has emphasized the importance of maintaining a balance between feature redundancy and relevancy to achieve competitive results [9]; as an example, the algorithm JMI provides

a trade-off between accuracy and stability for small datasets. CFC [26] uses Pearson's correlation coefficient to find features bearing high correlation with the target class; to detect redundant features, the algorithm excludes features highly correlated to other features. FCBF [64] follows a similar approach by gathering a set of features exhibiting high symmetrical uncertainty with the class, removing redundant features within the selected features set.

2.3 Feature interaction

Recent work advocates the search for interacting features jointly correlated with the target class. In [35], an interactive feature is called *weakly relevant* if is not highly correlated with the target class, but belongs to a subset of features that together are strongly correlated with the target class. In INTERACT [67], this definition is used to propose a criterion to remove features following a backward search technique. INTERACT uses symmetrical uncertainty and a scoring metric based on data consistency to eliminate all irrelevant features. As a result, only strongly and weakly related features remain.

In IWFS [66], a metric called interaction weight factor based on symmetrical uncertainty and conditional mutual information is proposed; the algorithm captures the interaction of up to three features and the target class. CMICOT [54] detects the interaction between multiple features under a greedy search and a sequential forward selection approach; the algorithm uses a binary feature representation to estimate the CMIM measure in high-dimensional problems, and can identify interactions of up to t features and the target class (t is an input parameter). In [57], a four-dimensional joint mutual information measure is used to detect high order interactions. In general, identifying high-order interactive features by using an efficient algorithm remains an open problem.

2.4 Algorithms based on distance

Another group of feature selection algorithms uses distances such as Euclidean and Manhattan distance to evaluate the relevance of features. These algorithms also go through an iterative process, but instead of selecting features in each iteration, they assign a weight to each feature. At the end,

the features with the highest weights are considered to be the most relevant to the target class. Although the descending order of the features' weights can show the importance of each feature, it is not clear how many of the features should be considered relevant. Another issue with this group of algorithms is that they are weak in recognizing redundancy. Although interaction is rarely addressed in connection with these methods, they implicitly give higher weights to interactive features (unless the interactive features include a duplicate or have a high correlation with another feature in the dataset).

2.4.1 Relieff

Relieff [39, 49] is one of the successful distance-based feature selection algorithms and is reminiscent of the KNN algorithm [27, 2]. The basic idea in this algorithm is to weigh each feature based on how well it distinguishes between samples that are similar to each other but are from different classes. In each round of the algorithm, Relieff selects a random sample Xi, then searches for the k nearest hits, which are its nearest neighbors from the same class, Hj, and the k nearest misses, which are its nearest neighbors from each of the different classes Mj. The distances are usually calculated using Euclidean or Manhattan distances [3]. The weight of each feature is updated according to the values of that feature for the current instance and its nearest hits and nearest misses.

$$\begin{aligned} W[f_j] = W[f_j] - \sum_{l=1}^{k} \text{diff}(Z_i[f_j], \text{Hit}_l[f_j])/(m.k) + \\ \sum_{C \neq \text{class}(Z_i)} [\tfrac{P(C)}{1-P(\text{class}(Z_i))} \sum_{l=1}^{k} \text{diff}(Z_i[f_j], \text{Miss}_l(C)[f_j])]/(m.k) \end{aligned} \tag{9}$$

where $Z_i[f]$ is the value of feature f for the ith instance in the dataset, Miss_j and Hit_j are the jth nearest miss and the jth nearest hit respectively. $\text{diff}(A[f_j], B[f_j])$ is the distance between the values of f_j in A and B; for nominal features it is 0 when the values are equal and 1 otherwise. For numerical attributes it is equal to $(A[f_j] - B[f_j])/\Gamma$ where Γ is the range of possible values $(\max - \min)$. Algorithm 1 shows pseudo code for the Relieff algorithm.

A feature is penalized each time it disagrees with the values that the nearest hits have for

Algorithm 1: Relieff

Input : dataset $D : \{Z_i | 0 \leq i \leq M\}$
with features: $F : \{f_j | 0 \leq j \leq n\}$ and
C: Classes.
k : Number of neighbors

Output: $W[f_j]$: Weight of each feature in the dataset

```
1  Set all W[f_j] = 0;
2  for i=1 to m do
3  |  Pick a random instance Z_i from D
4  |  Find k nearest Hits: Hit_l
5  |  foreach C ≠ class(Z_i) do
6  |  |  Find k nearest Misses: Miss_l(C)
7  |  end
8  |  for j=1 to n do
9  |  |  Update W[f_j] according to equation 9;
10 |  end
11 end
```

that feature. On the other hand, a feature is rewarded each time it disagrees with the values that nearest misses have for that feature. By looking at the local neighborhood of each instance, Relieff is able to update the weight of each feature. The update compares the value of a feature in the current instance with the value of the feature in each of the nearest miss and nearest hit neighbors. Features with values similar to nearest hits and farther away from the nearest misses will score higher. Jakulin [30] states that the common weaknesses of algorithms that use information theory to detect feature interaction does not exist in Relieff. By looking at the behavior of a feature in each neighborhood, Relieff is capable of identifying interactive features.

There are a number of challenges associated with using Relieff. First, there are two user-defined variables for which choosing the best value is not straightforward: the number of iterations m and the number of neighbors k. Second, Relieff is sensitive to duplicates, and in data sets with redundant features, Relieff cannot weigh them correctly, i.e., the actual weight that belongs to a feature is divided between the duplicates.

For cases where the user-defined parameters are set properly and there are no duplicate features in the dataset, Relieff is one of the best feature selection algorithms. In these cases, if a subset of features has a high relevance to the target due to interaction, Relieff gives them higher weights.

Several attempts have used the basic idea of Relieff to come up with more effective methods [23, 56, 55, 29, 60]. Two popular algorithms in this category are:

- Simba [23]: This algorithm is an iterative search margin-based feature selection algorithm. It is very similar to Relieff in the sense that both of these algorithms are reminiscent of KNN classifiers; however, the authors who developed Simba claim that it can discover duplicates. Unlike Relieff, when several duplicated (or correlated) features are related to the target, Simba includes only one of these in the final selected subset of features. The algorithm considers only one nearest hit and one nearest miss. The main idea in this algorithm is to turn the evaluation function into an optimization problem. Given the margins for one nearest neighbor, the algorithm calculates the hypothesis margin for a sample x with respect to a set of points P and a weight vector w for the features through the following formula:

$$\theta_p^w = \frac{1}{2}\{||x - nearmiss(x)||_w - ||x - nearhit(x)||_w\} \tag{10}$$

  where $||z||_w = \sqrt{\sum w_i^2 z_i^2}$. The feature set is generated by finding the weight vector w that maximizes the evaluation function. The authors use the stochastic gradient ascent over $e(w)$ and a user-defined variable that indicates the number of iterations in the algorithm. Relieff is also considered a margin-based feature selection algorithm. The margin of an instance x is defined with respect to its nearest hits and nearest misses as follows:

$$\theta = d(x, NM(x)) - d(x, NH(x)) \tag{11}$$

  where $d(.)$ is the distance function and $NH(x)$ and $NM(x)$ are the nearest hit and nearest

miss for sample x. Relieff attempts to find a weight for each feature that provides a maximal margin. The weight of each feature indicates how well it can distinguish between different classes.

- LOGO [56]: Another method that is similar to Relieff and Simba is the local-learning-based algorithm, or LOGO. This algorithm is related to SVMs [11] in the sense that both algorithms solve a nonlinear problem by first transforming it into a linear problem, then finding one nearest hit and one nearest miss in the new space using the expectation maximization technique, and then solving the linear problem using an optimization method that maximizes the margin. The method decomposes the nonlinear problem into a set of locally linear problems through local learning, and then learns feature relevance globally within the large margin framework. This algorithm has a strong performance when there is a large number of irrelevant features, giving them significantly lower weights than relevant features.

2.5 Ensemble methods in feature selection

Ensemble learning is a popular method in machine learning where the main idea is to combine multiple models instead of a using a single model to solve a problem. It is particularly common in classification, where the goal is to combine various learning algorithms to increase predictive performance outcomes. There are two main approaches in ensemble leaning [5, 6, 15]:

- Bagging: This method works by reducing the variance. This is done by random sampling with replacement in the dataset. This results in different training sets that are not statistically independent.

- Boosting: In boosting the main goal is to reduce bias by combining multiple weak classifiers and aggregate them to obtain a strong classifier.

The rationale of using ensembles for several learners is that by adding diversity, an increase in the performance of classifiers is expected. Several approaches are provided in order to combine the

results of learners [5, 4]. Using the ensemble method in feature selection is not a common practice. Generally, there are two types of ensemble feature selection algorithms:

- Homogeneous: In these algorithms, the same feature selection algorithm is applied to different subset of a dataset, so the result is different training data using the same feature selection method.
- Heterogeneous: In these algorithms, different feature selection methods are applied to a training dataset.

The result of either of these methods needs to be aggregated to obtain the final output. The approach to aggregate the results depends on the output of the feature selection algorithms in the ensembles [5]. When the output is a subset of features this can be done by finding intersection or union of all the obtained subset of features. The intersection contains only those features which are common among all the weak selectors. On the other hand, the union of the features gives a final set consisting of all the selected features by any of the weak selectors. In the case of wrapper selectors, it is possible to include partial results of each weak selector into the final selection only if it improves classification performance [5, 8].

In the case of combining the results of selectors that the output is the ranks or weights of features, the aggregation is more complicated. One idea is to use the Normalized Discounted Cumulative Gain algorithm (NDCG) [33]. It returns a value between 0 and 1 where 1 indicates that the rankings are identical. In [5] the authors show this method is not effective when it is used in ensemble feature selection algorithms. The other option to combine the results is to consider the min-rank, max-rank or the average task. Another search method that is used to improve the performance of search engines such as ranking SVM [34] also could be used to learn the ranking function.

In [50] the authors developed a homogeneous ensemble feature selection, which aggregates RELIEF and SVM RFE [25] with multiple bootstrapped samples of the training data. Their method showed that the performance of the ensemble feature selection method is almost the same or in some cases, slightly better than the version using a single feature selector, and it resulted in more robustness.

It is possible that several different feature subsets gives equally optimal results, and ensemble feature selection may reduce the risk of choosing an unstable subset. Moreover, different feature selection algorithms may obtain the ranks for the features based on local optima in the space of feature subsets. Ensemble feature selection might alleviate this problem and help to enhance the result of learning algorithm to achieve higher performance [51, 50].

In [16] the authors introduced a novel approach that employed the Neyman–Pearson lemma [44] to detect whether if a feature is statistically relevant. First, they implement a homogeneous feature selection using JMI. After that, they used the Neyman-Pearson lemma to identify the relevant features. First, it is assumed that there is an optimal number of features, K*, in the dataset. A Bernoulli random variable is defined for each feature, X_l. In every iteration of the ensemble method, the selector returns a set of indexes for the relevant features. Based on that the Bernoulli random variable of each feature is updated: if the feature is selected as relevant $X_l = 1$ otherwise $X_l = 0$. n bootstraps, obtained n Bernoulli random variables from a Binomial distribution with $Zn = X1 + ... + Xn$ successes. When a feature is selected as relevant by chance, the probability of it is $p_0 = k/K$, where k is the number of selected relevant features, and K is the total number of features. The observed probability of a feature selected as relevant in $p_1 = z_n/n$. To know if $p_1 > p_2$ the hypothesis is generated:

$$\begin{aligned} H0 &: p_0 = p_1 \\ H1 &: p_1 > p_0 \end{aligned} \tag{12}$$

According to the authors, the Neiman-Pearson lemma is the most powerful hypothesis test for solving this problem. They tested their method first by testing it on both toy datasets and real datasets. The selected features with their methods were evaluated on two classifiers, and the results were compared with the performance of the classifier on the full feature set and also on single JMI when the top 10 features in each dataset are selected. The experiments show promising results.

In this section we looked at some of the popular feature selection methods. We encourage avid

readers to look into [10, 40, 37] for more details.

3 Methodology

3.1 A novel entropy based feature selection

While our method is inspired by the original Relieff, we have used entropy to create a new metric called *e_measure*. In this section we explain this new metric and a novel feature selection algorithm called EBFS [47].

One of the critical properties of the Relief family is evaluating the relevancy of a feature based on the different values it takes on different classes. At its core, Relief estimates the following difference of terms to obtain the final weights:

$$\begin{aligned} W[f] \simeq P(\text{diff. values of } f|\text{nearest misses}) \\ -P(\text{diff. values of } f|\text{nearest hits}) \end{aligned} \tag{13}$$

With this interpretation of Relief, a feature is expected to attain a high weight if it leads to a set of different feature values among instances of different classes, while it leads to a set of similar feature values among instances of the same class. In other words, the formula penalizes features that separate instances of the same class. Now let Hits_i and Misses_i represent two sets that contain the nearest hits and nearest misses of Z_i, respectively. Let $\text{val}(F_j, \text{Hits}_i)$ and $\text{val}(F_j, \text{Misses}_i)$ be the set of values for feature F_j in the jth feature of all nearest hits and nearest misses of instance Z_i respectively. We can re-formulate the weight update formula for a feature f_j with respect to instance Z_i as follows:

$$W[f_j, Z_i] \simeq H\left(\text{val}(f_j, \text{Misses}_i)\right) - H(\text{val}(f_j, \text{Hits}_i)) \tag{14}$$

This formula shows the difference in uncertainty of the values of f_j in nearest misses and nearest hits. The formula shows two entropy terms: the first one corresponds to the amount of "choices" we have for the value of f_j in the set of nearest misses. In the case of non-binary classification problems, at least two classes are considered different from the class of instance Z_i. In relevant features, we expect to have different values for each of the classes in the set of nearest hits; as a

result we look for high entropy values where more choices of feature values lead to higher weight updates.

In the same way, the second term in equation 14 shows the amount of "choices" for f_j in the set of nearest hits. Here we look for low entropy values, meaning we expect most feature values to be about the same. A value of 0 means all nearest hits have the exact same value for f_j. In the next section we introduce e-measure, the main component of our algorithm.

3.1.1 e-measure

While the formula in equation 14 can quantify the uncertainty of feature values in the neighborhood of an instance Z_i, it may fail to capture relevant information. Consider the following example. Assume a binary classification and a neighborhood of size $k = 4$. Assume the feature values for nearest misses is $\{1, 1, 1, 1\}$ and values for nearest hits is $\{0, 0, 0, 0\}$. Although the feature separates nearest hits from nearest misses perfectly, both terms in equation 14 yield the same entropy value. To overcome this, we add the value of f_j of the current instance Z_i, $Z_i[f_j]$, to both terms of equation 14. We call this measurement *e_measure*:

$$E[f_j, Z_i] = H\big(\text{val}(f_j, \text{Misses}_i),\ Z_i[f_j]\big) - H\big(\text{val}(f_j, \text{Hits}_i),\ Z_i[f_j]\big) \tag{15}$$

By adding this additional term, we expect to reduce the value of $\text{val}(f_j, \text{Hits}_i)$ in relevant features as we are adding a value that is anticipated to have higher probability of occurrence. On the other hand, adding it to $\text{val}(f_j, \text{Misses}_i)$ leads to an increase of uncertainty as it is anticipated that this value will have lower chance of occurrence. We used normalized entropy in the formula to keep the range of entropy values between $[0, 1]$. Algorithm 2 shows pseudo code for EBFS [47].

3.2 A novel ensemble method using Relieff

In this section, we introduce another filter-based feature selection method. Like EBFS, this method is inspired by Relieff. Unlike EBFS, here we use ensemble methods to improve the performance of

Algorithm 2: EBFS

Input : dataset $D : \{Z_i | 0 \leq i \leq M\}$
with features: $F : \{f_j | 0 \leq j \leq n\}$ and
C : Classes.
k : Number of neighbors

Output: $W[f_j]$: Weight of each feature in D

Set all $W[f_j] = 0$
foreach Z_i *in* D **do**
    Hits = {}, Misses = {}
    Find k nearest Hits:
    Hits$_i$ ={Hit$_l$|$1 \leq l \leq k$}
    foreach $C \neq class(Z_i)$ **do**
        Find k nearest Misses: Miss$_l(C)$
        Misses$_i$ =Misses$_i \cup \{Miss_l(C) | 1 \leq l \leq k\}$
    end
    foreach f_j *in* F **do**
        Find values of f_j in Misses$_i$: val(f$_j$, Misses$_i$)
        Find values of f_j in Hits$_i$: val(f$_j$,Hits$_i$)
        $W[f_j] += e\_measure(Z_i, F_j)$ according to equation 15
    end
end

our algorithm. Although the benefits of ensemble methods have been widely studied in classifiers, their effects in the feature selection field need more attention. Moreover, we introduce a novel method to decide the final number of selected features.

Choosing the correct parameters for an algorithm is an important decision in any machine learning task. Some filter-based algorithms use iterative forward selection to include features incrementally, some use backward selection to remove irrelevant ones from an initial set containing all features. The final set in these scenarios includes all the relevant features according to that algorithm. On the other hand, in many feature selection algorithms, such as weight-based feature selection methods, one of the parameters is the appropriate subset size: the number of features that should be included in the final set. Determining a threshold or a cut-off point is not clear and is mostly done by choosing an arbitrary number. Including an arbitrary number of features in the final set may cause to discard some of the relevant features, or to include some of the irrelevant ones, which can degrade classification performance.

Another parameter-specific to Relieff that needs to be carefully considered is the number of nearest neighbors. If this number is set too small or too large, the algorithm will not be looking at the "right" neighborhood for each instance and the result might not be reliable. In most published papers, the number of neighbors is set to 5 or 10 for all the datasets. In the original paper [39], a value of $\log(n)$ is suggested, where n is the number of features. Finally some previous work [6] suggests choosing a percentage of features such as 10% or 25%. Choosing the size of the neighborhood depends on the nature of the learning task and of the number of available instances in the dataset. Different values for this parameter result in variations on the final weight for each feature. Since the ranking of features is set according to their final weights, changing the number of nearest neighbors will affect the final feature ranking. This effect is less noticeable in a noise-free dataset; however, in a dataset with a high percentage of noise, the final ranking of features could be significantly different with different values on the size of the neighborhood.

Our method tries to address these issues by 1) using ensemble methods and 2) choosing the best subset of features by using a statistical hypothesis test. To explain our method, we divide our

description into four parts. First we explain the ensemble method. We then go over a statistical hypothesis testing method which is followed by a post-hoc method applied to the output of the ensemble technique. After that we show a plot to visualize results. Finally we explain the rationale to choose the final subset of features.

3.2.1 The ensemble method

We use Relieff with different number of neighbors in an ensemble method setting. First, a range of numbers is selected $K = \{k_1, \ldots, k_p\}$ as the number of neighbors. Relieff is then applied to the dataset for each nearest-neighbor size. Features are listed in descending order according to their final weight (highest weight is rank 1). This gives p different orders of features. In the case of ties, the rank average (for features with the same weight) is assigned to each of the features. To find the final position of each feature and aggregate the results, there are a number of possibilities:

- Min-rank: The final rank of each feature is equal to the minimum rank obtained from p different nearest-neighbor sizes.
- Max-rank: The final rank of each feature is equal to the maximum rank obtained from p different nearest-neighbor sizes.
- Mean-rank: The final rank of each feature is equal to the average of all the positions obtained in p different nearest-neighbor sizes.
- Learning-the-rank: In methods such as SVM_rank [38], a variant of SVM classifier, the goal is to learn the ranking function.

Min-rank is the most conservative option, where it tries to reflect the best performance of a feature even when it happens only once in the 'wrong' neighborhood size. On the other hand, using max-rank means a feature may lose its chance of being included among the selected features even if only one neighborhood does not show its relevance. Thus both min-rank and max-rank can lead to a biased result. Learning-the-rank is another supervised task that requires large amounts of

training data in order to attain good results. In our case, where we deal with only a few samples, this is not a good choice. We decided to choose mean-rank to aggregate results; it is a technique that shows robustness in finding the correct order of each feature even under different neighborhood sizes. Algorithm 3 shows pseudo-code for our proposed mechanism.

Algorithm 3: Ensemble Relieff

Input : dataset $D : \{Z_i | 0 \leq i \leq M\}$
with features: $F : \{f_j | 0 \leq j \leq N\}$ and
C : Classes.
$K : \{K_p | 0 \leq p \leq P\}$ Range of size for nearest neighbors

Output: $Rank$: 2D P*N array where R_{pj}=Rank of feature j at k_p

```
Set all Rank_pj = 0;
foreach k_p in K do
    Run Relieff with neighborhood size k_p
    W[f_jp] : Weight of feature f_j
    Find R_pj : Rank of f_j according to its weight at neighbor size k_p
end
foreach f_j in F do
    Avg_Rank[f_j] = Average({R_pj | 0 ≤ p ≤ P}
end
Return Rank and Avg_Rank
```

3.2.2 The statistical method

The final set of features is obtained using a statistical hypothesis test. First, let us look at the output of our algorithm from another point of view. One output is through $Rank$, a 2D array in which the row p shows the weights of features according to Relieff(k_p), and each column contains the weights of one feature according to different neighbor sizes. We are interested to know if there is a significant difference between these features. When there is no significant difference, we can claim that all the features have the same importance and thus, all are relevant. If there is a significant difference between features, it is possible to use hypothesis testing to detect such difference. We define the null hypothesis and alternative hypothesis as follows:

$$
\begin{aligned}
H_0 &: \text{ The features have equal importance in predicting the class} \\
H_1 &: \text{The features do not have equal importance in predicting the class}
\end{aligned}
\tag{16}
$$

Here we need a statistical method for testing the differences between a group of related samples. Multiple hypothesis testing is a well-studied problem in statistics. In [14] two statistical methods are suggested for a very similar case when the goal is to compare the performance of a group of classifiers over a set of datasets: MANOVA Test and Friedman Test. We present more details about these two methods next.

- **MANOVA:** Multi-variate ANOVA or MANOVA is one popular method to compare differences of means among multiple dependent variables. If the variability between features is significantly larger than the error variability, MANOVA rejects the null hypothesis. Two assumptions are made here:

  Normal Distribution: ANOVA assumes that samples are coming from a normal distribution.

  Sphericity: This property is similar to the homogeneity of variance in ANOVA which means that the random variables exhibit equal variance.

  From all the available MANOVA tests, Tukey [59] is the most suitable one. It is a single-step multiple comparison procedure statistical test. It can be used to find means that are significantly different from each other.

  The first assumption is not valid in our case. The weights of features are not coming from a normal distribution. This can be proved by tests such as Jarque–Bera [32], Kolmogorov-Smirnov [42] or Shapiro-Wilks [53] tests. In [14], the author explains that even in the case of some form of abnormality in the population, many experts would not oppose the use of MANOVA. He argued that the second assumption is crucial; violating it would affect post-hoc results drastically. From this we conclude that MANOVA is not a good choice for our problem.

- **Friedman:** The Non-parametric analogue of MANOVA is the Friedman test [21, 20]. Let r_i^j be the rank of the jth (of k features) on the ith iteration (of N iterations) of Relieff. The Friedman test compares the average ranks of features, $R_j = \frac{1}{n}\sum_i r_i^j$. Under the null hypothesis, which states that all the features are equivalent and so their ranks R_j should be equal, the Friedman statistic is defined as:

$$\chi_F^2 = \frac{12N}{k(k+1)} \sum_j R_j^2 - \frac{k(k+1)^2}{4} \tag{17}$$

It is distributed according to χ^2 with $k-1$ degrees of freedom, when N and k are big enough (as a rule of a thumb, $N > 10$ and $k > 5$). For a smaller number of algorithms and data sets, exact critical values have been computed [36, 65, 14]. Iman and Davenport [13, 14] showed that Friedman's χ^2 is undesirably conservative and derived a better statistic,

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)\chi_F^2} \tag{18}$$

which is distributed according to the F-distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom. If the null-hypothesis is rejected by Friedman's test (p-value < 0.05), we can proceed with a post-hoc test. The Nemenyi test [43, 14], is similar to the Tukey test for ANOVA and is used when all features are compared to each other. The performance of two features is significantly different if the corresponding average ranks differ by at least the critical difference, where

$$CD = q_\alpha \frac{k(k+1)}{N} \tag{19}$$

critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$.

Considering MANOVA is not a good choice in our case, our approach is to first use Friedman's test and, in case p-value <0.05, we can proceed with the Nemenyi test. The result of the post-hoc

Nemenyi test is a 2D array containing the p-values which indicate if the difference between two features is significant. In the next section, we visualize this result.

3.2.3 Visualization of Nemenyi's matrix

To have a better understanding of the result of Nemenyi test, we proceed by converting the matrix to a heatmap. We generate a toy parity dataset as an example to explain the heatmap. 1000 instances are generated with 8 features $f_0, f_2, ..., f_7$ that are independently and identically distributed uniform random variables in the interval [0, 10]. The true target function is given by:

$$Y = f_0 \oplus f_1 \oplus f_2$$

In this scenario, the first three features are relevant. Using a neighbor size of $K = [2 : 15]$ we can obtain the weight matrix using the Friedman test to obtain the p-value. The p-value indicates that there is a significant difference between the features (p-value <0.05). We visualize the result of post-hoc Nemenyi using the heat-map in Figure 1:

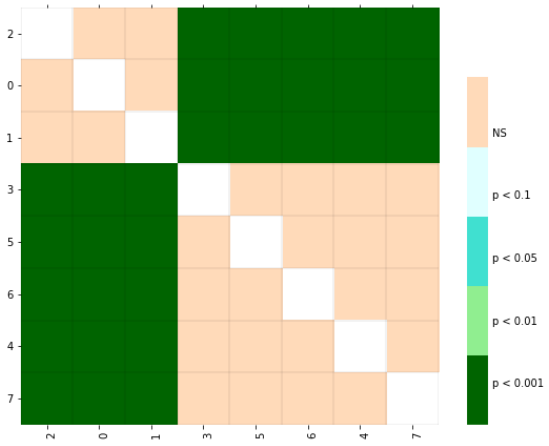

Figure 1: Heat-map of Nemenyi matrix for the parity problem

The features are sorted in descending order according to their average rank, where f_2 is ranked 1 (the highest weight), and f_7 is ranked 8. Since there is no noise in the dataset, the result clearly shows the separation of relevant and irrelevant features. There are no significant differences between

any pairs among the top three features as all the related cells are pink, which represents a p-value above 0.1. Moreover, no significant difference is observed between any pairs of the irrelevant features. The comparison between a relevant and an irrelevant feature always returned a value less than 0.001 (represented with dark green). The plot displays other p-values; in this example, all p-values are either above 0.1 or below 0.001. Figure 2 shows the result of a similar experiment with 50 features where only $f_0, ..., f_9$ are relevant features.

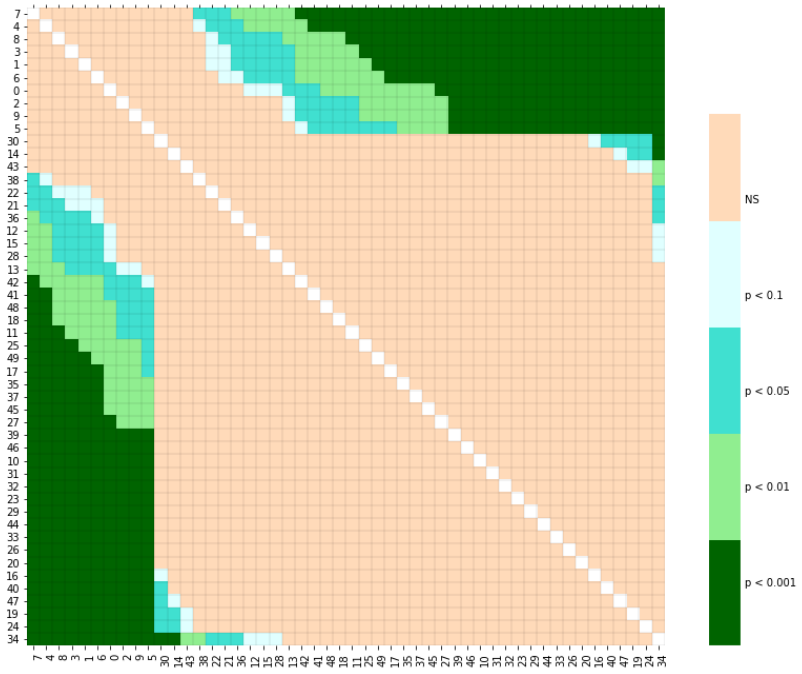

Figure 2: Example of visualization of Nemenyi matrix

In the experimental part of this work, other Nemenyi results will be visualized using similar heat-maps. Generally, using such a plot helps to observe the order of average ranks for the feature set. Moreover, the relationship between the features can now be elucidated more clearly. In the next section, we explain how our algorithm decides the number of features to be included in the final set of relevant features.

3.2.4 Forming the final subset

When working with feature selection algorithms where the output is the ranking or weights of all the features, finding the final relevant subset is challenging. In our case, feature selection is done by analyzing the Nemenyi matrix and evaluating the quality of the final set of features using three classifiers. Looking at Figures 1 and 2, it is obvious that we can expect no significant difference between any two relevant features. Also, no significant difference between two irrelevant features is observed. On the other hand, there is a significant difference between a relevant and an irrelevant feature. Note that the plot shows the features according to their rank, where we are certain the feature at the first row is relevant and the feature at the last row is irrelevant.

We begin by searching for a feature with the highest average rank that is not significantly different from the last feature, the feature in the last row. In the heat-map, we are looking for the first pink cell in the last column. We assume that this feature is the first irrelevant one and we call it First Possible Irrelevant (FPI). All features with lower average rank than FPI are considered irrelevant. Thus, for each feature f_j, it is possible to calculate the percentage of features with higher average rank than FPI that are not significantly different from f_j - for a p-value of 0.1; this means the percentage of pink cells in each row starting on column 0 to the column that belongs to FPI. Similarly, for each feature f_j, it is possible to calculate the percentage of features with a lower average rank than FPI that are not significantly different from f_j. Intuitively, we can say that a feature is more desirable when it has a high percentage of pink cells in the columns that belong to the hypothetical relevant features and a low percentage of pink cells in the columns of the hypothetical irrelevant features. We include the first feature f_0 in the relevant set and for all the other features $F : \{f_j | 1 \leq j \leq n\}$ we calculate the following :

- $NR[f_j]$: The percentage of features in $\{f_1, \ldots, f_{min(j-1,FPI-1)})\}$ that are not significantly different from f_j; the percentage of features that are **N**ot significantly different from f_j in the assumed **R**elevant set.

- $SI[f_j]$: The percentage of features in $\{f_{FPI}, \ldots, f_n\}$ that are significantly different from f_j;

the percentage of features that are **S**ignificantly different from f_j in the assumed **I**rrelevant set.

It should be noted that FPI is just an assumption about irrelevant feature with the highest rank, and it is always possible that the actual irrelevant feature with the highest rank be placed before it. Thus, in $NR[f_j]$ we are only considering features that have a better average rank that f_j and a better average rank that FPI. Our initial results showed that this approach yields better results.

We expect to see that by selecting a subset of features with a higher percentage in the first group and/or a lower percentage in the second group, we can reach better generalization performance. In the experimental part of this work we discuss and test different scenarios and evaluate results using three different classifiers.

4 Experiments and Results

The datasets employed in the experiments can be found at the UCI repository [17], except for the Colon dataset [1]. We report on datasets having a variety of size, no. of features, and no. of classes. All the datasets were pre-processed by removing index values. On all the algorithms that use information theoretic measures, we discretized conntinuous features using the MDL method [18]. We used three classifiers in Scikit-learn [45]: Linear SVM, KNN with three neighbors, and Decision tree. Table 1 shows the summary of the datasets in our experiments.

Table 1: Summary of the datasets in our experiments

dataset	#Features	#Samples	#Classes
Breast	9	699	2
CMC	9	1473	3
Wine	13	178	3
Zoo	16	101	7
Vehicle	18	846	4
Mushroom	22	8124	2
Kr	36	3196	2
Lung	56	32	3
Audio	69	200	24
Musk	166	476	2
SCADI	205	70	7
Colon	2000	62	2

4.1 Experiments on the novel entropy based feature selection

To assess the performance of our method, we compare it with five state-of-the-the art feature selection algorithms: Relieff [39], MRMR [46], CMIM [19], IWFS [66], and JMI [62]. We implemented EBFS [47] in Python 3.0. For Relieff, we used $k = 3$ and the built-in function from Matlab R2018b. IWFS was implemented in Matlab. The rest of the algorithms were provided by FEAST [9].

Similar to settings reported in recent publications [66, 54, 57], we planned our experiment as follows. We applied the feature selection algorithm to each of the datasets. Each algorithm was run n times on each dataset where $1 \leq n \leq \min(50, \#\text{features in dataset})$. After the ith run, the

top i feature(s) were selected and the rest were discarded. Then the dataset was fed to the three classifiers using ten-fold cross-validation. We report on average accuracy and standard deviation for the n runs. Results are shown in Table 2.

Each result is compared against EBFS for statistical significance using a two-tailed t-student test with $p = 0.05$. An (+) shows that EBFS is significantly better than the competitor, and an (-) shows the competitor is significantly better. For all the results with no signs, no significant difference were found. EBFS obtains the best results with decision tree classifier; no loss is seen against other feature selection algorithms in this classifier, and it achieved significantly better in fifteen cases.

In general, our method performs significantly better or equally well as other algorithms. When compared to Relieff, EBFS lost in just one dataset with the KNN classifier. Looking at the three classifiers results, EBFS achieved the most wins against IWFS; it won six, seven and eight times with Decision trees, KNN and SVM, respectively. In comparison to JMI and CMIM, EBFS performed mostly equal or better than these algorithms in all the classifiers. EBFS obtained similar results in comparison to MRMR when using decision trees and KNN. In SVM, however, EBFS lost three times and won twice, making MRMR our strongest competitor.

4.2 Experiments on the novel ensemble method using Relieff

4.2.1 Set up

In this section, we assess the performance of the proposed ensemble method. We use the same twelve datasets as the previous experiment. Relieff was implemented in Python. The ensemble method was built based on the neighborhood size in the range of 2 to 20. The result is a matrix that contains the weights of features where each row represents the result of Relieff in one of the nineteen selected neighborhoods.

Table 2: Average accuracy with standard deviations.

dataset	EBFS	Relieff	MRMR	CMIM	IWFS	JMI
C45						
Breast	92.22±0.03	92.03±0.03	92.44±0.01	92.39±0.01	92.58±0.01	92.61±0.01
CMC	49.38±0.04	46.79±0.02	47.65±0.01	46.94±0.02	48.44±0.01	48.12±0.01
Wine	93.45±0.04	88.48±0.09	94.21±0.04	94.58±0.04	90.26±0.05	94.42±0.04
Zoo	91.78±0.06	90.74±0.12	93.98±0.06	93.48±0.06	80.52±0.13(+)	93.92±0.06
Vehicle	66.35±0.07	65.55±0.09	66.22±0.05	66.6±0.05	63.4±0.07	64.34±0.06
Mushroom	96.4±0.01	91.69±0.11	96.5±0.01	96.6±0.0	96.34±0.01	96.56±0.0
Kr	92.35±0.05	93.68±0.06	93.49±0.06	93.29±0.06	90.53±0.06	93.87±0.06
Lung	54.85±0.1	52.03±0.09	53.32±0.09	50.72±0.09(+)	40.47±0.08(+)	54.23±0.11
Audio	72.11±0.11	70.35±0.13	73.93±0.06	74.12±0.06	41.13±0.05(+)	73.08±0.07
Musk	76.91±0.04	66.99±0.04(+)	64.61±0.03(+)	71.63±0.02(+)	66.25±0.04(+)	65.92±0.03(+)
SCADI	80.39±0.04	80.32±0.05	81.42±0.04	81.65±0.04	65.96±04(+)	81.61±0.04
Colon	88.7±0.03	71.68±0.04(+)	86.08±0.04(+)	85.3±0.04(+)	82.0±0.02(+)	86.19±0.04(+)
Win/Loss		2/0	2/0	3/0	6/0	2/0
KNN						
Breast	92.94±0.07	92.83±0.07	94.05±0.02	94.31±0.01	94.22±0.01	94.64±0.01
CMC	46.2±0.03	47.39±0.03	45.94±0.02	47.83±0.02	49.16±0.03(-)	49.11±0.03(-)
Wine	94.62±0.07	76.08±0.09(+)	96.26±0.06	96.74±0.06	90.15±0.08	96.3±0.06
Zoo	90.06±0.06	86.28±0.17	91.53±0.06	91.77±0.06	80.73±0.11(+)	91.42±0.06
Vehicle	64.32±0.09	64.42±0.09	60.55±0.04	63.41±0.04	60.28±0.06	61.49±0.04
Mushroom	94.77±0.01	88.93±0.12(+)	95.3±0.01(-)	94.59±0.01	94.52±0.01	94.9±0.01
Kr	82.33±0.09	84.93±0.07	83.46±0.08	83.24±0.07	74.17±0.09(+)	81.98±0.09
Lung	53.25±0.07	57.32±0.07(-)	55.72±0.11	50.2±0.07(+)	33.93±0.06(+)	55.33±0.09
Audio	57.02±0.08	58.37±0.11	53.76±0.04(+)	53.13±0.04(+)	40.71±0.06(+)	52.27±0.02(+)
Musk	75.26±0.05	76.39±0.06	66.3±0.04(+)	73.06±0.03(+)	68.11±0.03(+)	72.51±0.05(+)
SCADI	84.6±0.09	80.21±0.07(+)	84.85±0.08	83.38±0.07	35.02±0.05(+)	84.85±0.08
Colon	92.22±0.02	81.0±0.03(+)	94.9±0.02(-)	94.0±0.02(-)	84.57±0.06(+)	93.03±0.02
Win/Loss		4/1	2/2	3/1	7/1	2/1
SVM						
Breast	94.54±0.03	94.54±0.03	94.82±0.02	95.02±0.01	95.02±0.01	95.17±0.01
CMC	51.17±0.04	50.07±0.04	45.98±0.04(+)	50.29±0.03	50.29±0.03	50.3±0.03
Wine	95.66±0.06	91.93±0.07	96.59±0.05	96.49±0.05	90.08±0.07(+)	96.55±0.05
Zoo	90.2±0.07	90.03±0.12	90.92±0.08	92.74±0.06	81.17±0.12(+)	92.48±0.06
Vehicle	62.35±0.1	68.57±0.15	61.64±0.11	67.74±0.13	67.42±0.12	67.5±0.12
Mushroom	91.95±0.01	78.84±0.1(+)	85.9±0.09(+)	90.34±0.03(+)	88.44±0.02(+)	89.11±0.04(+)
Kr	91.75±0.05	91.16±0.05	90.28±0.05	91.96±0.05	89.84±0.05	92.06±0.05
Lung	50.9±0.07	53.4±0.08	54.35±0.08(-)	53.07±0.09	36.25±0.07(+)	53.2±0.1
Audio	73.31±0.12	68.63±0.13	72.69±0.05	72.75±0.06	41.7±0.07(+)	71.49±0.07
Musk	74.06±0.06	67.28±0.06(+)	73.35±0.03	64.81±0.04(+)	55.26±0.04(+)	65.04±0.05(+)
SCADI	81.33±0.05	79.21±0.04(+)	83.81±0.03(-)	82.89±0.03(-)	65.96±0.3(+)	83.41±0.03(-)
Colon	88.9±0.03	72.61±0.05(+)	94.52±0.02(-)	95.58±0.02(-)	84.57±0.1(+)	89.69±0.02
Win/Loss		4/0	2/3	2/2	8/0	2/1

4.2.2 Friedman's and Nemenyi's tests

For all the datasets, Friedman's test is applied on the matrix and for all it shows a p-value less than 0.1 which indicates there is a significant difference between the features. Table 3 shows the p-values for each dataset after applying Friedman's test using [28].

Table 3: P-value results of Friedman's test

dataset	p-value
Breast	9.71E-10
CMC	0.071201494
Wine	0.070914722
Zoo	1.48E-28
Vehicle	2.06E-06
Mushroom	0.092761454
Kr	7.40E-23
Lung	2.10E-40
Audio	4.45E-54
Musk	8.90E-226
SCADI	1.06E-85
Colon	0.04942286

4.2.3 Visualizing the Nemenyi matrix

After applying the Friedman test, we could move forward to the post-hoc Nemenyi test and visualize results using heat-maps. [58, 48] were used to calculate post-hoc Nemenyi. Figures 4 through 15 show the heat-maps of the twelve datasets in Table 1. The pink color indicates p-value < 0.1. Considering each figure as a matrix, it is possible to get all the results only from the upper triangle or the lower triangle; however, it is easier to study a relation between an individual feature with other features in a row. Some common patterns can be seen in these figures. The upper left starts with pink cells indicating no significant difference between the features with the highest average rank. Dark green cells can be seen on the upper right which means that features with higher average ranks are significantly different from the features with the lowest average rank. Also, the bottom right consists of mostly pink cells which indicate there is no significant difference between features with a higher chance to be considered as irrelevant. Although it is possible to study the difference between the features with different levels of significance(α), for the rest of this work we focus only

on two p-values: 0.001 and 0.1.

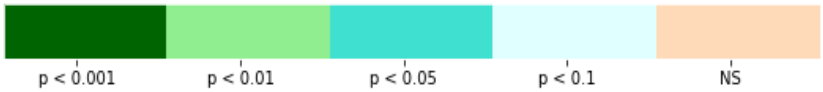

Figure 3: Color-bar of different p-values in the heat-maps: The pink represents no significant difference.

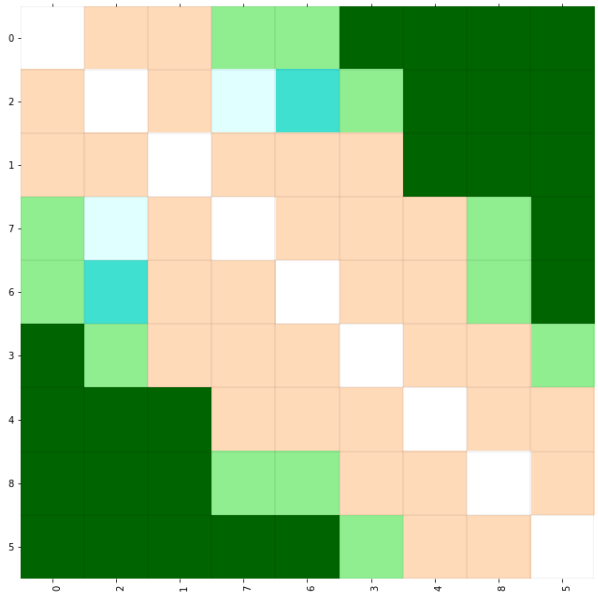

Figure 4: Nemenyi matrix of Breast

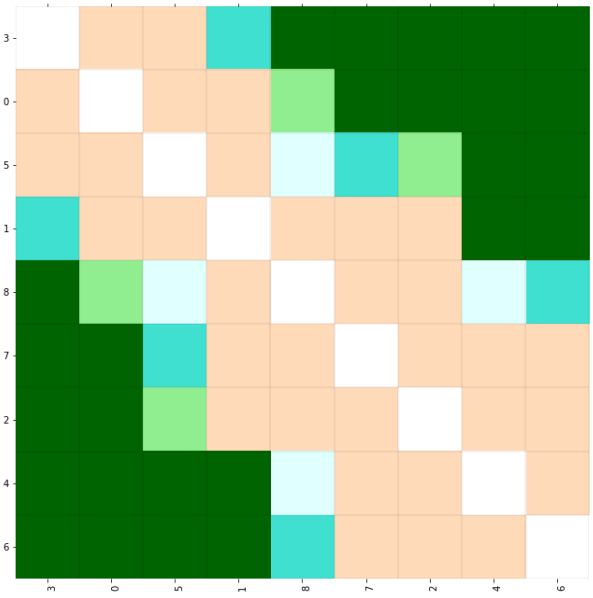

Figure 5: Nemenyi matrix of CMC

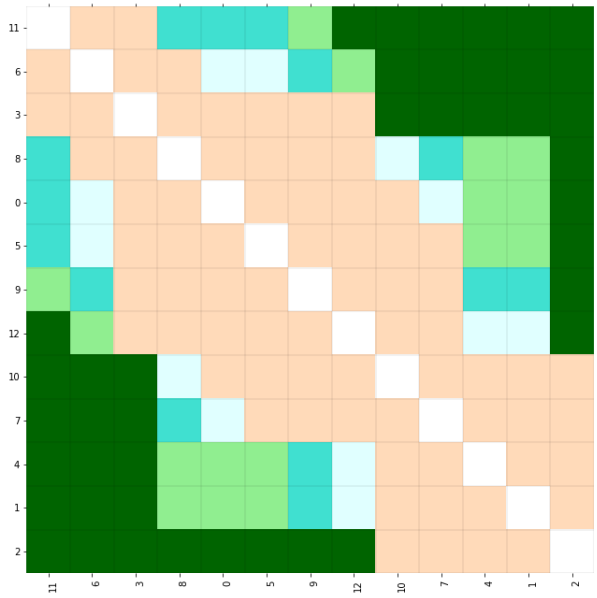

Figure 6: Nemenyi matrix of Wine

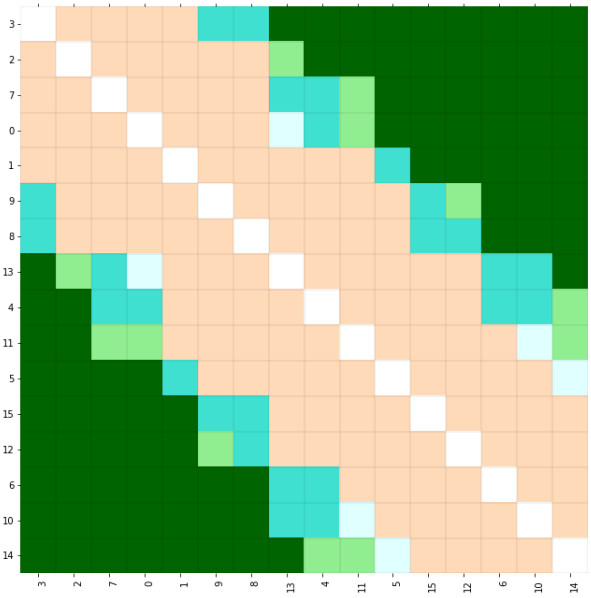

Figure 7: Nemenyi matrix of Zoo

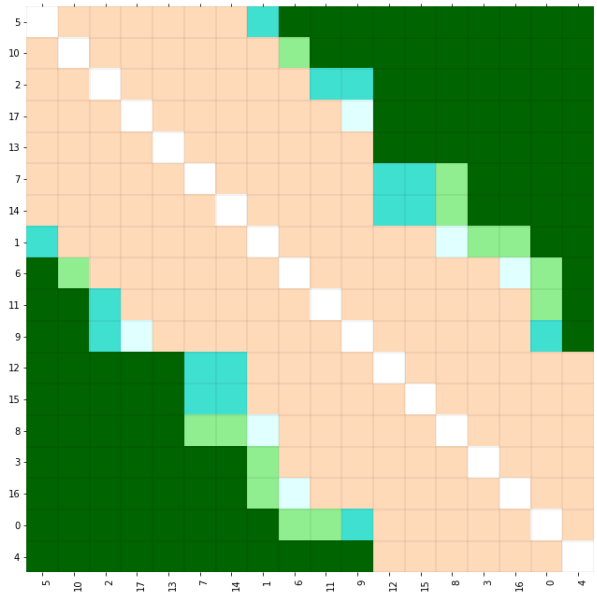

Figure 8: Nemenyi matrix of Vehicle

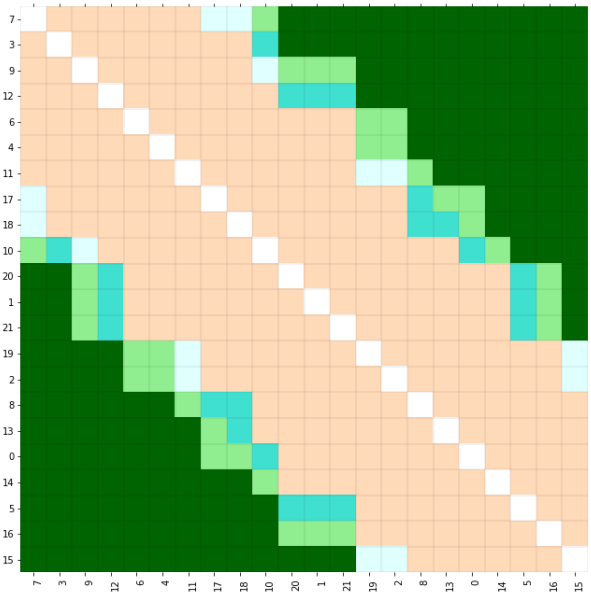

Figure 9: Nemenyi matrix of Mushroom

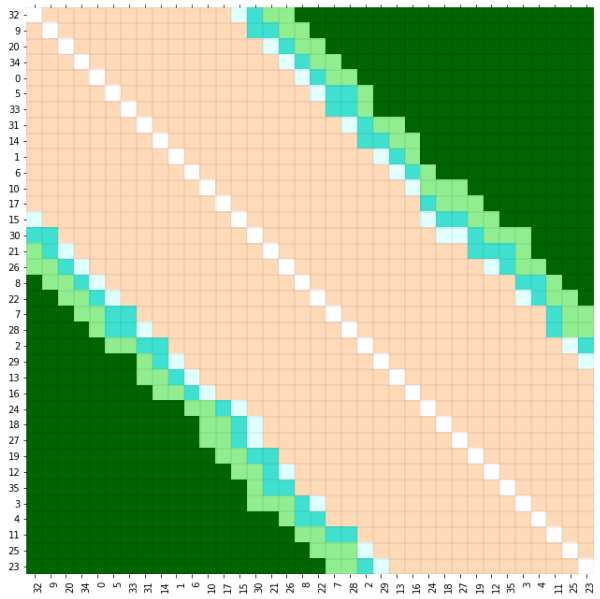

Figure 10: Nemenyi matrix of Kr

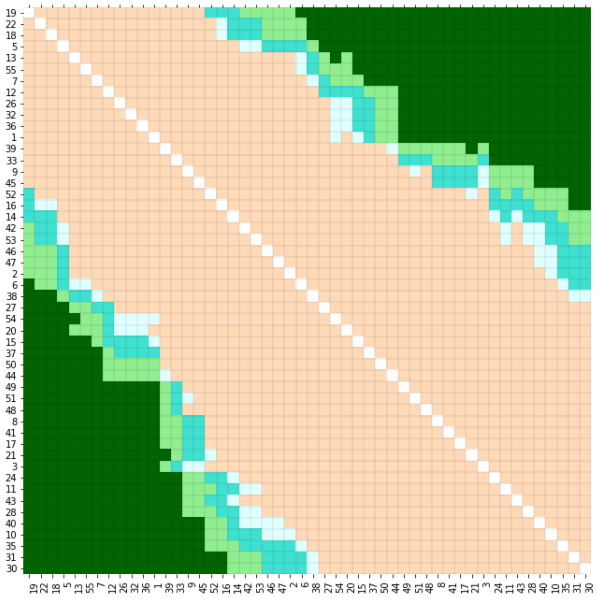

Figure 11: Nemenyi matrix of Lung

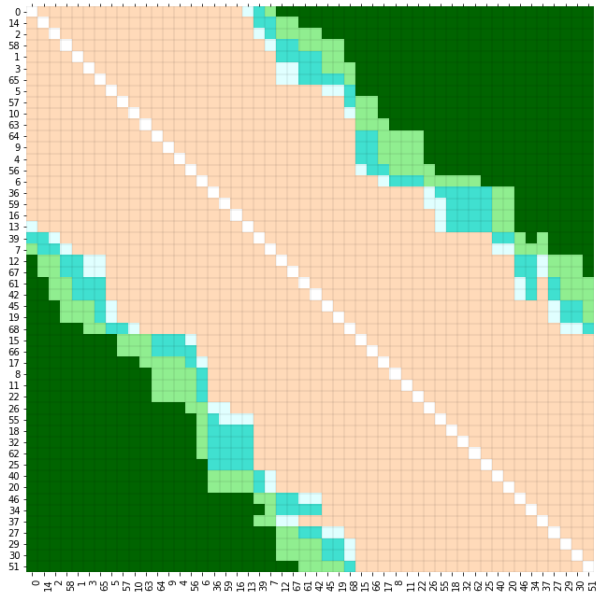

Figure 12: Nemenyi matrix of Audio

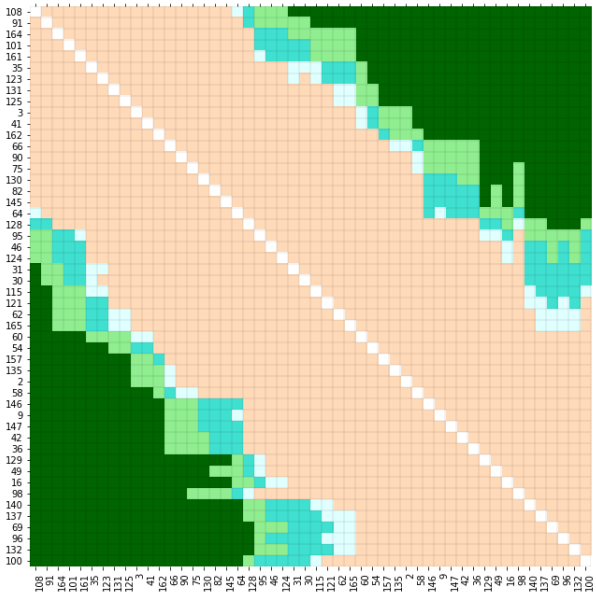

Figure 13: Nemenyi matrix of Musk

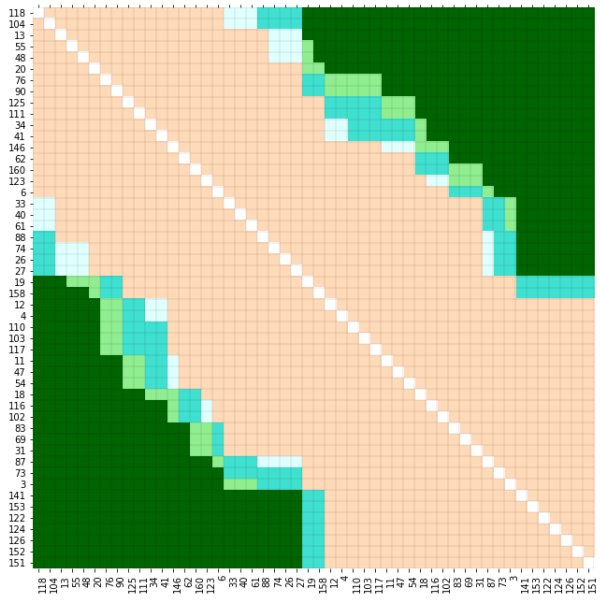

Figure 14: Nemenyi matrix of SCADI

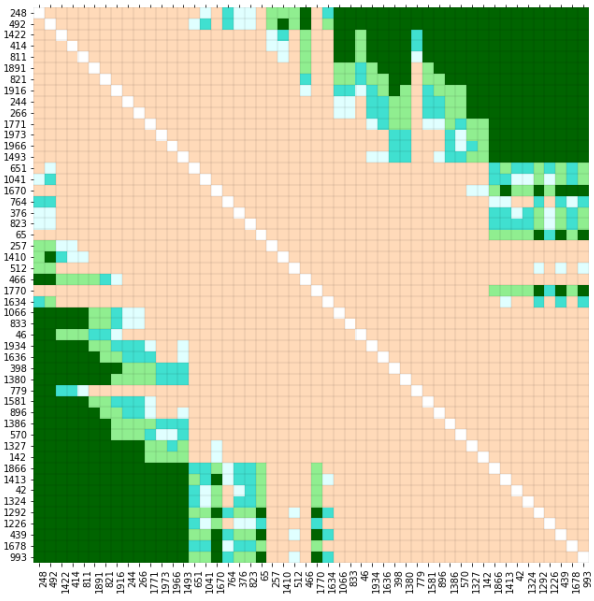

Figure 15: Nemenyi matrix of Colon

4.2.4 Experiment on NR, SI and α values

To test the performance of our method in selecting the final subset of features, we tried several sets of values for our parameters. Initial results showed that values less than 50% for NR or above 50% for SI will not increase the accuracy significantly in any of the classifiers. Table 4 shows the selected values for the parameters and the average selected number of features according to each set of parameters. To clarify the NR and SI concepts, we explain our selected parameters:

- NR=100 : Only features that have a 100% rate of non significant difference with assumed relevant features are selected.
- NR=50: Only features that have at least a 50% rate of non significant difference with assumed relevant features are selected.
- SI=10: Only features that have at least a 10% rate of significant difference with assumed irrelevant features are selected.
- SI=50: Only features that have at least a 50% rate of significant difference with assumed irrelevant features are selected.

Table 4: Tested parameters with the average number of selected features

NR	SI	α	Avg. number of selected features
-	10	0.001	13.58
		0.1	17.33
	50	0.001	8.17
		0.1	12.33
50	-	0.001	23.25
		0.1	19.92
	10	0.001	23.17
		0.1	20.17
	50	0.001	23.17
		0.1	20.0
100	-	0.001	14.67
		0.1	10.67
	10	0.001	15.17
		0.1	17.33
	50	0.001	14.42
		0.1	12.33

- NR=100 | SI=10: Only features that have at least a 100% rate of significant difference with assumed irrelevant features or at least a 10% rate of significant difference with assumed irrelevant features are selected.
- NR=100 | SI=50: Only features that have at least a 100% rate of significant difference with assumed irrelevant features or at least a 50% rate of significant difference with assumed irrelevant features are selected.
- NR=50 | SI=10: Only features that have at least a 50% rate of significant difference with assumed irrelevant features or at least a 10% rate of significant difference with assumed irrelevant features are selected.
- NR=50 | SI=50: Only features that have at least a 50% rate of significant difference with assumed irrelevant features or at least a 50% rate of significant difference with assumed irrelevant features are selected.

Figure 16 shows the average number of features in a bar chart. To show the selected features according to each set of parameters and compare them to the total number of features in each dataset, we generated Figures 17 and 18. Blue bars and orange bars show the total number of

features and the selected number of features respectively. It should be noted that since we set the maximum possible number of selected features to 50, these figures display 50 as the maximum number for a full feature set.

The range of selected features goes from 2 to 41, and we generally observe a reduction in the selected number of features. Of course, the important issue is to make sure that the selected subset is as close to optimal as possible. We evaluate the quality of selected features using three classifiers. As previously, we choose linear SVM, KNN with three neighbors, and C45. All classifiers are run using ten-fold cross-validation.

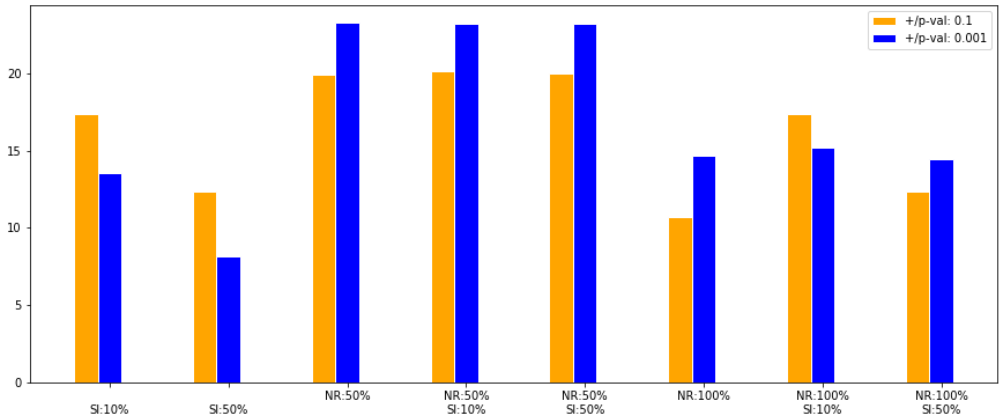

Figure 16: Average number of selected features according to each set of parameters

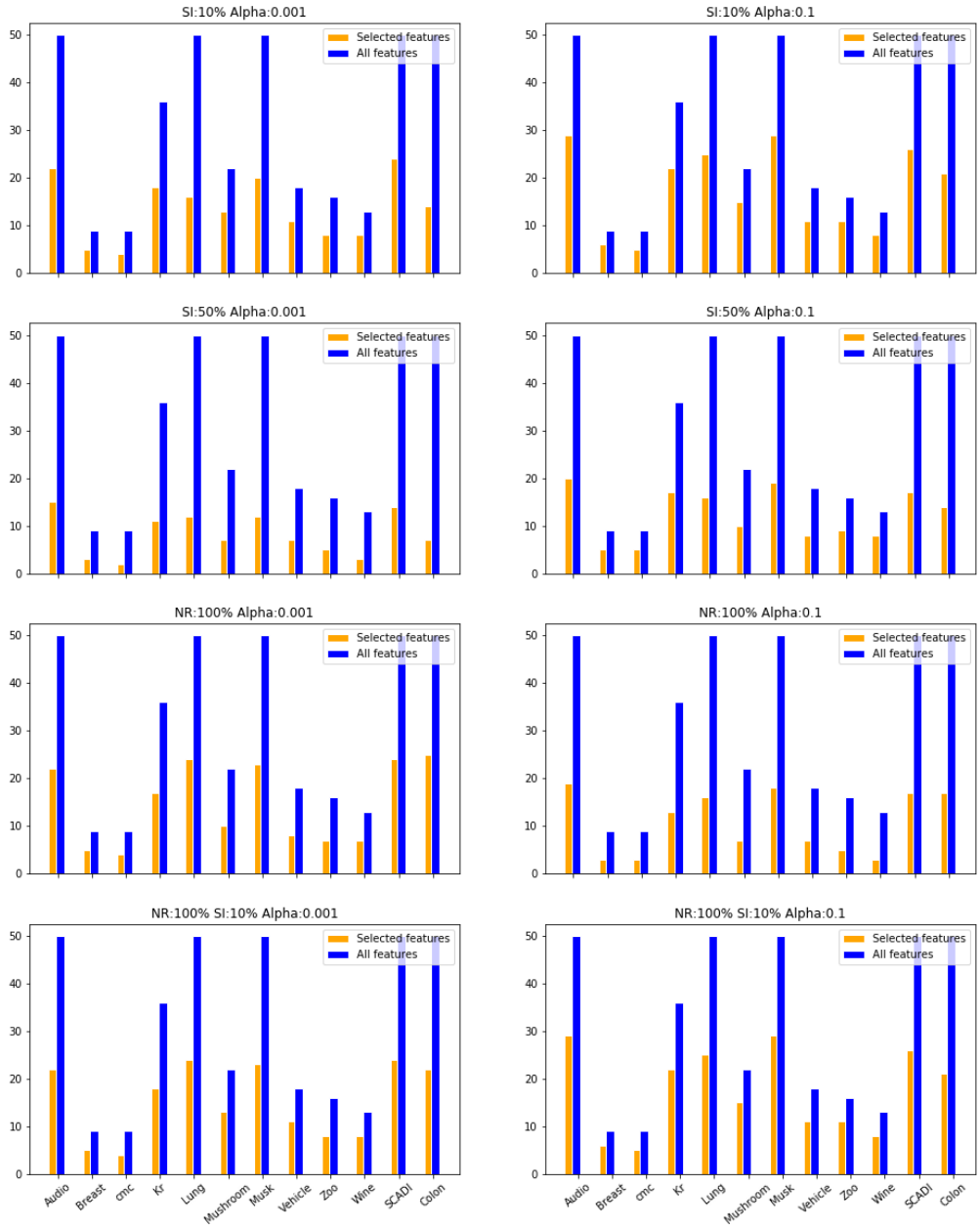

Figure 17: Number of selected features according to each set of parameters (part 1)

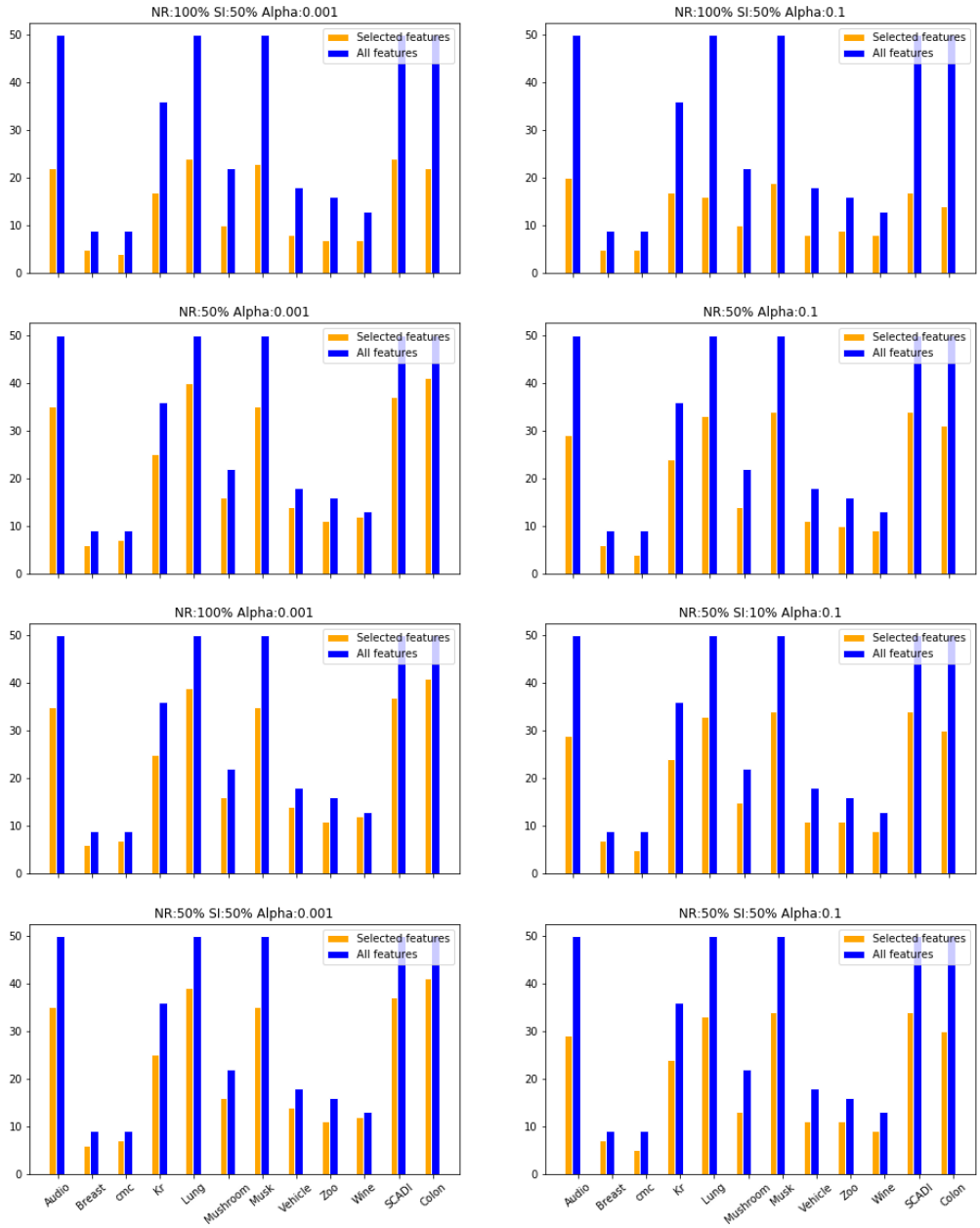

Figure 18: Number of selected features according to each set of parameters (part 2)

Tables 5, 6 and 7 show accuracy and standard deviation according to C45, KNN and SVM respectively. For each dataset we find the subset of features according to NI, SI and α and discarded the rest of features. Each result is compared against the full set of features for statistical significance using a two-tailed t-student test with $p = 0.1$. An (+) shows a win for our method and an (-) shows the full set of features is significantly better. For all other results with no signs, no significant difference was found.

The result on C45 shows that all the parameter settings performed mostly as well as the whole feature set even though our method never won. There is no significant difference between SI=10 and NR=100 | SI=10 and the full feature set in any of the datasets at any α level. At α=0.001, SI=50 lost three times which is the maximum number of losses in this table. At α=0.1, we generally observe less loss although NR=100 shows 2 losses and SI=50 and NR=100 | SI=50 each lost once.

The result of KNN shows that our method won in some settings with both α levels. It specifically always won in the dataset KR. The other win is in the Lung dataset with NR=100 (α=0.1), SI=10 (α=0.001) and SI=50 (both α values). Like in C45, SI=10 never lost but SI=50 has the maximum number of losses similar to its result at C45. NR=100 lost twice at α=0.1 and NR=50 lost once at α=0.001. SI=10 never lost but SI=50 has three losses at α=0.001 more than in any other setting similar to its performance with C45. For NR=100 | SI=10 and NR=100 | SI=50 no loss is observed.

Regarding the SVM table, more negative signs are observed. SI=10 has some losses with both α levels but still performs better than SI=50. NR=50 acted better than NR=100. Unlike the two other tables, here NR=100 has the worst performance. Each of NR=50, NR=50 | SI=10 and NR=50 | SI=50 lost once in each α value. NR=100 | SI=10 and NR=100 | SI=50 won once at α=0.001 but lost four and five times in total, respectively. Unlike C45 and KNN where SI=10 had the best performance, here NR=50 and all the combinations including NR=50 lost less than others including SI=10.

We also compare the result of each (NI, SI, α) combination with the full feature set using the Wilcoxon signed-rank test [14, 61], which is a non-parametric alternative to the paired t-test. It

Table 5: C45: Average accuracy with standard deviations in comparing full feature set and the proposed method

Name	All Features	NR=100	NR=50	SI=10	SI=50	NR= 100, SI=10	NR= 100 SI=50	NR=50 SI=10	NR=50 SI=50
P-value=0.001									
Audio	0.78±0.09	0.77±0.1	0.76±0.1	0.77±0.1	0.77±0.1	0.77±0.1	0.77±0.1	0.76±0.1	0.76±0.1
Breast	0.92±0.04	0.92±0.04	0.93±0.04	0.92±0.04	0.94±0.03	0.92±0.04	0.92±0.04	0.93±0.04	0.93±0.04
CMC	0.48±0.04	0.46±0.04	0.46±0.02	0.46±0.04	0.48±0.04	0.46±0.04	0.46±0.04	0.46±0.02	0.46±0.02
Kr	0.98±0.03	0.97±0.03	0.97±0.03	0.97±0.03	0.96±0.03	0.97±0.03	0.97±0.03	0.97±0.03	0.97±0.03
Lung	0.37±0.28	0.47±0.27	0.47±0.38	0.58±0.28	0.55±0.31	0.47±0.27	0.47±0.27	0.43±0.27	0.43±0.27
mushroom	0.97±0.09	0.99±0.02	0.97±0.09	0.97±0.09	0.97±0.07	0.97±0.09	0.99±0.02	0.97±0.09	0.97±0.09
Musk	0.69±0.1	0.67±0.15	0.7±0.11	0.65±0.15	0.62±0.11	0.67±0.15	0.67±0.15	0.7±0.11	0.7±0.11
Vehicle	0.72±0.03	0.68±0.03(-)	0.68±0.03(-)	0.7±0.03	0.68±0.04(-)	0.7±0.03	0.68±0.03(-)	0.68±0.03(-)	0.68±0.03(-)
Zoo	0.94±0.06	0.92±0.04	0.93±0.05	0.94±0.05	0.89±0.1	0.94±0.05	0.92±0.04	0.93±0.05	0.93±0.05
Wine	0.95±0.06	0.96±0.05	0.94±0.04	0.95±0.05	0.85±0.08(-)	0.95±0.05	0.96±0.05	0.94±0.04	0.94±0.04
SCADI	0.82±0.09	0.83±0.11	0.82±0.09	0.83±0.11	0.69±0.12(-)	0.83±0.11	0.83±0.11	0.82±0.09	0.82±0.09
Colon	0.84±0.15	0.76±0.2	0.8±0.18	0.85±0.19	0.83±0.19	0.76±0.23	0.76±0.23	0.8±0.18	0.8±0.18
P-value=0.1									
Audio	0.78±0.09	0.78±0.11	0.75±0.1	0.75±0.1	0.77±0.1	0.75±0.1	0.77±0.1	0.75±0.1	0.75±0.1
Breast	0.92±0.04	0.94±0.03	0.93±0.04	0.93±0.04	0.92±0.04	0.93±0.04	0.92±0.04	0.92±0.05	0.92±0.05
CMC	0.48±0.04	0.49±0.03	0.46±0.04	0.47±0.04	0.47±0.04	0.47±0.04	0.47±0.04	0.47±0.04	0.47±0.04
Kr	0.98±0.03	0.95±0.03	0.97±0.03	0.96±0.03	0.97±0.03	0.96±0.03	0.97±0.03	0.97±0.03	0.97±0.03
Lung	0.37±0.28	0.58±0.28	0.51±0.37	0.43±0.27	0.58±0.28	0.43±0.27	0.58±0.28	0.51±0.37	0.51±0.37
mushroom	0.97±0.09	0.97±0.07	0.97±0.09	0.97±0.09	0.99±0.02	0.97±0.09	0.99±0.02	0.97±0.09	0.97±0.09
Musk	0.69±0.1	0.66±0.12	0.7±0.14	0.66±0.14	0.66±0.13	0.66±0.14	0.66±0.13	0.7±0.14	0.7±0.14
Vehicle	0.72±0.03	0.68±0.04(-)	0.7±0.03	0.7±0.03	0.68±0.03(-)	0.7±0.03	0.68±0.03(-)	0.7±0.03	0.7±0.03
Zoo	0.94±0.06	0.89±0.1	0.95±0.05	0.93±0.05	0.94±0.05	0.93±0.05	0.94±0.05	0.93±0.05	0.93±0.05
Wine	0.95±0.06	0.85±0.08(-)	0.94±0.06	0.95±0.05	0.95±0.05	0.95±0.05	0.95±0.05	0.94±0.06	0.94±0.06
SCADI	0.82±0.09	0.74±0.12	0.82±0.09	0.83±0.11	0.74±0.12	0.83±0.11	0.74±0.12	0.82±0.09	0.82±0.09
Colon	0.84±0.15	0.85±0.19	0.74±0.16	0.76±0.2	0.85±0.19	0.76±0.2	0.85±0.19	0.73±0.21	0.73±0.21

Table 6: KNN: Average accuracy with standard deviations in comparing full feature set and the proposed method

Name	All Features	NR=100	NR=50	SI=10	SI=50	NR= 100, SI=10	NR= 100 SI=50	NR=50 SI=10	NR=50 SI=50
P-value=0.001									
Audio	0.51±0.12	0.53±0.11	0.53±0.11	0.53±0.11	0.58±0.12	0.53±0.11	0.53±0.11	0.53±0.11	0.53±0.11
Breast	0.95±0.03	0.95±0.04	0.95±0.04	0.95±0.04	0.95±0.03	0.95±0.04	0.95±0.04	0.95±0.04	0.95±0.04
CMC	0.49±0.04	0.49±0.03	0.49±0.05	0.49±0.03	0.47±0.03	0.49±0.03	0.49±0.03	0.49±0.05	0.49±0.05
Kr	0.69±0.09	0.93±0.04(+)	0.92±0.06(+)	0.93±0.05(+)	0.95±0.05(+)	0.93±0.05(+)	0.93±0.04(+)	0.92±0.06(+)	0.92±0.06(+)
Lung	0.41±0.23	0.6±0.34	0.45±0.38	0.72±0.33(+)	0.67±0.31(+)	0.6±0.34	0.6±0.34	0.42±0.4	0.42±0.4
mushroom	0.95±0.11	0.96±0.09	0.95±0.1	0.96±0.09	0.93±0.11	0.96±0.09	0.96±0.09	0.95±0.1	0.95±0.1
Musk	0.77±0.09	0.68±0.14	0.72±0.13	0.67±0.17	0.68±0.14	0.68±0.14	0.68±0.14	0.72±0.13	0.72±0.13
Vehicle	0.65±0.04	0.68±0.05	0.62±0.04(-)	0.64±0.05	0.69±0.07	0.64±0.05	0.68±0.05	0.62±0.04(-)	0.62±0.04(-)
Zoo	0.95±0.05	0.92±0.04	0.94±0.05	0.94±0.05	0.87±0.1(-)	0.94±0.05	0.92±0.04	0.94±0.05	0.94±0.05
Wine	0.98±0.03	0.96±0.05	0.99±0.02	0.98±0.04	0.82±0.14(-)	0.98±0.04	0.96±0.05	0.99±0.02	0.99±0.02
SCADI	0.86±0.11	0.89±0.1	0.86±0.11	0.89±0.1	0.65±0.1(-)	0.89±0.1	0.89±0.1	0.86±0.11	0.86±0.11
Colon	0.89±0.18	0.92±0.13	0.87±0.19	0.85±0.16	0.82±0.19	0.9±0.17	0.9±0.17	0.87±0.19	0.87±0.19
P-value=0.1									
Audio	0.51±0.12	0.55±0.13	0.53±0.11	0.53±0.11	0.53±0.12	0.53±0.11	0.53±0.12	0.53±0.11	0.53±0.11
Breast	0.95±0.03	0.95±0.03	0.95±0.04	0.95±0.04	0.95±0.04	0.95±0.04	0.95±0.04	0.95±0.03	0.95±0.03
CMC	0.49±0.04	0.47±0.03	0.49±0.03	0.49±0.03	0.49±0.03	0.49±0.03	0.49±0.03	0.49±0.03	0.49±0.03
Kr	0.69±0.09	0.92±0.04(+)	0.92±0.06(+)	0.93±0.06(+)	0.93±0.04(+)	0.93±0.06(+)	0.93±0.04(+)	0.92±0.06(+)	0.92±0.06(+)
Lung	0.41±0.23	0.72±0.33(+)	0.51±0.37	0.6±0.34	0.72±0.33(+)	0.6±0.34	0.72±0.33(+)	0.51±0.37	0.51±0.37
mushroom	0.95±0.11	0.93±0.11	0.96±0.09	0.96±0.09	0.96±0.09	0.96±0.09	0.96±0.09	0.96±0.09	0.96±0.09
Musk	0.77±0.09	0.73±0.18	0.7±0.14	0.7±0.14	0.7±0.2	0.7±0.14	0.7±0.2	0.7±0.14	0.7±0.14
Vehicle	0.65±0.04	0.69±0.07	0.64±0.05	0.64±0.05	0.68±0.05	0.64±0.05	0.68±0.05	0.64±0.05	0.64±0.05
Zoo	0.95±0.05	0.87±0.1(-)	0.93±0.05	0.94±0.05	0.93±0.05	0.94±0.05	0.93±0.05	0.94±0.05	0.94±0.05
Wine	0.98±0.03	0.82±0.14(-)	0.97±0.04	0.98±0.04	0.98±0.04	0.98±0.04	0.98±0.04	0.97±0.04	0.97±0.04
SCADI	0.86±0.11	0.78±0.12	0.86±0.11	0.89±0.1	0.78±0.12	0.89±0.1	0.78±0.12	0.86±0.11	0.86±0.11
Colon	0.89±0.18	0.87±0.16	0.9±0.17	0.9±0.17	0.85±0.16	0.9±0.17	0.85±0.16	0.92±0.13	0.92±0.13

Table 7: SVM: Average accuracy with standard deviations in comparing full feature set and the proposed method

Name	All Features	NR=100	NR=50	SI=10	SI=50	NR= 100, SI=10	NR= 100 SI=50	NR=50 SI=10	NR=50 SI=50
P-value=0.001									
Audio	0.78±0.09	0.74±0.07	0.75±0.08	0.74±0.07	0.76±0.09	0.74±0.07	0.74±0.07	0.75±0.08	0.75±0.08
Breast	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03	0.95±0.04	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03
CMC	0.52±0.03	0.52±0.04	0.52±0.04	0.52±0.04	0.48±0.04(-)	0.52±0.04	0.52±0.04	0.52±0.04	0.52±0.04
Kr	0.93±0.07	0.94±0.06	0.93±0.07	0.94±0.06	0.94±0.06	0.94±0.06	0.94±0.06	0.93±0.07	0.93±0.07
Lung	0.48±0.21	0.68±0.25(+)	0.5±0.17	0.65±0.28	0.63±0.36	0.68±0.25(+)	0.68±0.25(+)	0.5±0.17	0.5±0.17
Mushroom	0.92±0.12	0.92±0.12	0.93±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.93±0.12	0.93±0.12
Musk	0.73±0.12	0.71±0.08	0.72±0.08	0.73±0.09	0.68±0.12	0.71±0.08	0.71±0.08	0.72±0.08	0.72±0.08
Vehicle	0.8±0.05	0.69±0.05(-)	0.74±0.03(-)	0.73±0.05(-)	0.68±0.03(-)	0.73±0.05(-)	0.69±0.05(-)	0.74±0.03(-)	0.74±0.03(-)
Zoo	0.95±0.05	0.91±0.05(-)	0.93±0.05	0.95±0.05	0.87±0.1(-)	0.95±0.05	0.91±0.05(-)	0.93±0.05	0.93±0.05
Wine	0.99±0.02	0.96±0.04(-)	0.99±0.02	0.97±0.04(-)	0.82±0.12(-)	0.97±0.04(-)	0.96±0.04(-)	0.99±0.02	0.99±0.02
SCADI	0.83±0.07	0.85±0.11	0.82±0.09	0.85±0.11	0.72±0.11(-)	0.85±0.11	0.85±0.11	0.82±0.09	0.82±0.09
Colon	0.9±0.15	0.9±0.11	0.84±0.18	0.88±0.17	0.84±0.19	0.92±0.13	0.92±0.13	0.84±0.18	0.84±0.18
P-value=0.1									
Audio	0.78±0.09	0.75±0.09	0.73±0.07	0.73±0.07	0.74±0.09	0.73±0.07	0.74±0.09	0.73±0.07	0.73±0.07
Breast	0.96±0.03	0.95±0.04	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03	0.96±0.03
CMC	0.52±0.03	0.48±0.04(-)	0.52±0.04	0.53±0.04	0.53±0.04	0.53±0.04	0.53±0.04	0.53±0.04	0.53±0.04
Kr	0.93±0.07	0.94±0.06	0.93±0.08	0.93±0.07	0.94±0.06	0.93±0.07	0.94±0.06	0.93±0.08	0.93±0.08
Lung	0.48±0.21	0.65±0.28	0.47±0.22	0.62±0.25	0.65±0.28	0.62±0.25	0.65±0.28	0.47±0.22	0.47±0.22
Mushroom	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12	0.92±0.12
Musk	0.73±0.12	0.7±0.1	0.73±0.08	0.72±0.07	0.7±0.11	0.72±0.07	0.7±0.11	0.73±0.08	0.73±0.08
Vehicle	0.8±0.05	0.68±0.03(-)	0.73±0.05(-)	0.73±0.05(-)	0.69±0.05(-)	0.73±0.05(-)	0.69±0.05(-)	0.73±0.05(-)	0.73±0.05(-)
Zoo	0.95±0.05	0.87±0.1(-)	0.94±0.05	0.93±0.05	0.93±0.05	0.93±0.05	0.93±0.05	0.93±0.05	0.93±0.05
Wine	0.99±0.02	0.82±0.12(-)	0.97±0.05	0.97±0.04(-)	0.97±0.04(-)	0.97±0.04(-)	0.97±0.04(-)	0.97±0.05	0.97±0.05
SCADI	0.83±0.07	0.78±0.12	0.83±0.07	0.84±0.1	0.78±0.12	0.84±0.1	0.78±0.12	0.83±0.07	0.83±0.07
Colon	0.9±0.15	0.88±0.17	0.84±0.18	0.87±0.16	0.88±0.17	0.87±0.16	0.88±0.17	0.85±0.16	0.85±0.16

ranks the differences in performances for each data set, ignoring the signs, and compares the ranks for the positive and the negative differences. Table 8 shows these results; a negative or positive sign was added when Wilcoxon shows the significant difference at the 0.1 level to indicate a loss or win for our method. In C45, only SI=50 lost. In KNN no win or loss can be seen. In SVM, at α=0.1, NR=100 and any combination including NR=50 are a loss. Also at α=0.1, only SI=50 lost. For all the other combinations no significant difference is seen.

Table 8: P-values using Wilcoxon's test when comparing our method with the full feature set

α	NR	SI	C45	KNN	SVM
0.1	-	10	0.1361	0.2721	0.2721
0.1	-	50	0.5303	0.8753	0.1167
0.1	50	-	0.3668	0.5049	0.0376(-)
0.1	50	10	0.1467	0.2896	0.0455(-)
0.1	50	50	0.1696	0.2896	0.0376(-)
0.1	100	-	0.3078	0.7537	0.0414(-)
0.1	100	10	0.1361	0.2721	0.2721
0.1	100	50	0.5303	0.8753	0.1167
0.001	-	10	0.5563	0.5303	0.5303
0.001	-	50	0.0995(-)	0.6379	0.0414(-)
0.001	50	-	0.2894	0.9063	0.1167
0.001	50	10	0.2894	0.9063	0.1167
0.001	50	50	0.2894	0.9063	0.1167
0.001	100	-	0.5303	0.2721	0.4101
0.001	100	10	0.2240	0.2393	0.7537
0.001	100	50	0.5303	0.3078	0.5303

In another test, we compared our approach against Relieff with a neighbour size in the range [2,20]. After running Relieff on each neighborhood, the top 10%, 25%, and 40% of features in each dataset were selected. Accuracy acts as the performance metric, with Wilcoxon's test used for comparison. Table 9 shows the number of wins/losses according to Wilcoxon's at the 0.1 significance level.

We observe that for all the NI and SI values, we gained on identical results for both α levels. When 10% of features were selected, no loss can be seen. Our method always won according to SVM. SI=50 has the weakest result here, with the least number of wins. No loss is seen when the top 25% of features are selected; we see no win in KNN. In C45, we can see a number of wins

at SI=10 and NR=50. In SVM, again SI=10 and all the combinations of NR=100 won 19 times, however the combination of NR=50 shows only 6 wins. We see a few losses when the top 40% of features are selected. These losses happened in C45 and KNN when SI=50. We still get the most number of wins in SVM. SI=10 and NR=100 | SI=10 resulted in 17/19 wins.

Generally, we obtained the best result with SI=10. Interestingly, for each feature, having a significant difference with at least 10% of assumed irrelevant features means higher chance to be considered as relevant. We showed that our method at SI=10 is an effective way of selecting features and is capable of improving accuracy when compared to the full feature set when running C45, KNN, and linear SVM.

Table 9: P-values obtained from Wilcoxon's test in comparison of our method with Relieff

Feature %			10						25						40					
			C45		KNN		SVM		C45		KNN		SVM		C45		KNN		SVM	
NR	SI	α	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-
0	10	0.001	19	0	19	0	19	0	8	0	0	0	19	0	7	0	0	0	17	0
0	10	0.1	19	0	19	0	19	0	8	0	0	0	19	0	7	0	0	0	17	0
0	50	0.001	9	0	6	0	19	0	0	0	0	0	0	0	0	2	0	4	0	0
0	50	0.1	9	0	6	0	19	0	0	0	0	0	0	0	0	2	0	4	0	0
50	0	0.001	18	0	11	0	19	0	1	0	0	0	6	0	0	0	0	0	5	0
50	0	0.1	18	0	11	0	19	0	1	0	0	0	6	0	0	0	0	0	5	0
50	10	0.001	18	0	10	0	19	0	0	0	0	0	6	0	0	0	0	0	5	0
50	10	0.1	18	0	10	0	19	0	0	0	0	0	6	0	0	0	0	0	5	0
50	50	0.001	18	0	10	0	19	0	0	0	0	0	6	0	0	0	0	0	5	0
50	50	0.1	18	0	10	0	19	0	0	0	0	0	6	0	0	0	0	0	5	0
100	0	0.001	12	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	7	0
100	0	0.1	12	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	7	0
100	10	0.001	13	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	17	0
100	10	0.1	13	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	17	0
100	50	0.001	12	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	7	0
100	50	0.1	12	0	19	0	19	0	0	0	0	0	19	0	0	0	0	0	7	0

To sum up, in most cases, there is no significant difference between our method and the full feature set. To compare the different combinations of NI, SI, and α, we can see SI=10 gives one of the best results, however increasing SI to 50 is not helpful; the outcome is poor for all classifiers. Generally, NR=50 performs better when compared to NR=100 according to t-test win/loss results across the three classifiers. Although we have more negative signs at α=0.1, according to Wilcoxon's

test, we cannot choose a winner according to t-test results.

5 Conclusion and Future Work

5.1 Summary of contributions

Feature subset selection is a crucial task during the data pre-processing phase of almost any classification task. Finding high-order interactive features is an important challenge that needs further attention. In this work, we first introduce EBFS, an entropy-based feature selection algorithm that analyzes the neighborhood of a random instance (as originally proposed in the well-known Relieff algorithm) to estimate feature quality. Different from previous work, EBFS focuses on the entropy of feature values for nearest hits and misses. In comparison with five popular feature selection algorithms, our method shows competitive results and is capable of finding relevant features, including interactive features. This is the first time that a feature selection method based on a distance-metric uses information theory and specifically entropy to find the optimal subset of features. The result shows that in all three classifiers - C45, KNN and SVM - EBFS, obtained better results compared to the original Relieff.

The second contribution is the introduction of a new heterogeneous ensemble feature selection algorithm. The method exploits the benefits of ensemble methods in classification tasks, but this time to improve the performance of feature selection algorithms. This is the area of research that needs more attention. We introduced a new method to choose a final subset of features using a hypothesis method, Friedman and its post-hoc test, Nemenyi. The results show that our method is capable of capturing relevant subsets of features effectively. We used heatmaps to visualize the results of post-hoc Nemenyi. Visualization in feature selection could be an asset in the hands of data scientists to get a better sense of the relations between features. This is, for example, very important in micro-array data and in many scientific fields, such as biomedical research, where the study of gene interactions is an important task. The proposed statistical test can be used with different ensemble feature selection algorithms to determine the final subset of relevant features. We also evaluated our approach against Relieff with the number of neighbors in [2,20]. After running Relieff with one neighbor size, the top 10%, 25%, and 40% of features in each dataset were selected.

Accuracy was recorded, and Wilcoxon's test was performed in order to compare results. Table 8 shows the number of wins/losses according to Wilcoxon's test at the 0.1 significance level.

We observe that for all NI and SI values, we gained the same result under both α levels. When 10% of features were selected, no loss can be seen. Our method always won according to SVM. SI=50 has the weakest result. No loss is seen when the top 25% of features are selected; we see no win in KNN. In C45, we can see a number of wins at SI=10 and NR=50. In SVM, again SI=10 and all the combinations of NR=100 won 19 times, however the combination of NR=50 shows only 6 wins. We see a few losses when the top 40% of features are selected. These losses happened in C45 and KNN when SI=50. We still get the most number of wins in SVM. The settings of SI=10 and NR=100 | SI=10 results in 17/19 wins.

Generally, we obtained best results with SI=10. Interestingly, having a significant difference of at least 10% makes a feature a confident candidate for the relevant set. We showed that our method at SI=10 is an effective way of selecting features and is capable of gaining the highest accuracy at C45, KNN, and linear SVM.

5.2 Future work

There are a number of ideas that can be developed to improve the EBFS algorithm to improve on generalization performance. The first is to improve the e-measure. Exploring other information theory metrics in the formula could make the algorithm more effective. In addition, it is important to investigate how to find neighborhoods in a selective manner, to increase the ability to identify interacting features. To gain better results, one idea is to determine which subset of the training data should be included as the most effective neighborhood for the selected instance in each iteration. By finding the best neighborhood, we can figure out the number of nearest neighbors dynamically. Here the goal is finding the "best neighborhood" rather than a subset of "good neighbors". Some work has been done using active learning in this area but to the best of our knowledge the idea has not been applied to the feature selection field [63]. Like other distance-based methods, our method is incapable of detecting duplicates among the features and this could be improved

using information-theoretic metrics. Moreover, finding high order interactive features effectively is another area that can be explored.

The use of ensemble methods in feature selection is a new research area; it is possible to improve our proposed method by searching for the optimal combination of parameter values using ensemble techniques. It is also worth exploring how to find the final subset of relevant features using statistical methods, using different homogeneous and heterogeneous ensemble feature selection algorithms.

References

[1] Alon, U., Barka, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the United States of America* (1999).

[2] Altman, N. S. An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician* (1992).

[3] Asquith, J., and Kolman, B. Elementary Linear Algebra. *The Mathematical Gazette* (1987).

[4] Bagui, S. C. Combining Pattern Classifiers: Methods and Algorithms. *Technometrics* (2005).

[5] Bolón-Canedo, V., and Alonso-Betanzos, A. *Recent Advances in Ensembles for Feature Selection*, vol. 147 of *Intelligent Systems Reference Library*. Springer International Publishing, Cham, 2018.

[6] Bolón-Canedo, V., and Alonso-Betanzos, A. Ensembles for feature selection: A review and future trends. *Information Fusion 52* (12 2019), 1–12.

[7] Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. A review of feature selection methods on synthetic data. *Knowledge and Information Systems 34*, 3 (2013), 483–519.

[8] Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. Distributed feature selection: An application to microarray data classification. *Applied Soft Computing Journal* (2015).

[9] Brown, G., Pocock, A., Zhao, M.-J., and Luján, M. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *Journal of Machine Learning Research 13*, Jan (2012), 27–66.

[10] Chandrashekar, G., and Sahin, F. A survey on feature selection methods. *Computers and Electrical Engineering* (2014).

[11] Cortes, C., and Vapnik, V. Support-Vector Networks. *Machine Learning* (1995).

[12] Cover, T. M., and Thomas, J. A. *Elements of Information Theory*. 2012.

[13] Davenport, J. M. Approximations of the critical region of the friedman statistic. *Communications in Statistics - Theory and Methods* (1980).

[14] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research 7*, Jan (2006), 1–30.

[15] Dietterich, T. G. Ensemble methods in machine learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2000).

[16] Ditzler, G., Polikar, R., and Rosen, G. A bootstrap based Neyman-Pearson test for identifying variable importance. *IEEE Transactions on Neural Networks and Learning Systems 26*, 4 (4 2015), 880–886.

[17] Dua, D., and Graff, C. UCI machine learning repository. *University of California, School of Information and Computer Science* (2019).

[18] Fayyad, U., and Irani, K. Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of Thirteenth International Joint Conference on Artificial Intelligence* (1993), 1022–1027.

[19] Fleuret, F. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research 5* (2004), 1531–1555.

[20] Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association 32*, 200 (1937), 675–701.

[21] Friedman, M. A Comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics* (1940).

[22] García, S., Luengo, J., and Herrera, F. Data preprocessing in data mining. *Intelligent Systems Reference Library* (2015).

[23] Gilad-Bachrach, R., Navot, A., and Tishby, N. Margin based feature selection - theory and algorithms. In *Proceedings of the 21st International Conference on Machine Learning* (2004), p. 43.

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

[25] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine Learning* (2002).

[26] Hall, M. Correlation-based Feature Selection for Machine Learning. *Methodology 21i195-i20*, April (1999), 1–5.

[27] Hastie, T., Tibshirani, R., and Friedman, J. The elements of statistical learning. *New York: Springer Series in Statistics 1*, 10 (2001).

[28] Hill, C., and Hill, C. SciPy. In *Learning Scientific Programming with Python*. 2016.

[29] Huang, Y., McCullagh, P. J., and Black, N. D. An optimization of ReliefF for classification in large datasets. *Data and Knowledge Engineering* (2009).

[30] Jakulin, A. Attribute interactions in machine learning. Master's thesis, University of Ljubljana, Department of Computer and Information Science, Slovenija, 2003.

[31] Jakulin, A. *Machine learning based on attribute interactions*. PhD thesis, Univerza v Ljubljani, Department of Computer and Information Science, Slovenija, 2005.

[32] Jarque, C. M., and Bera, A. K. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics Letters* (1980).

[33] Järvelin, K., and Kekäläinen, J. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* (2002).

[34] Joachims, T. Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2002).

[35] John, G. H., Kohavi, R., and Pfleger, K. Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 121–129.

[36] Kafadar, K., and Sheskin, D. J. Handbook of parametric and nonparametric statistical procedures. *The American Statistician* (1997).

[37] Khalid, S., Khalil, T., and Nasreen, S. A survey of feature selection and feature extraction techniques in machine learning. In *Proceedings of 2014 Science and Information Conference, SAI 2014* (2014).

[38] Kolde, R., Laur, S., Adler, P., and Vilo, J. Robust rank aggregation for gene list integration and meta-analysis. *Bioinformatics* (2012).

[39] Kononenko, I. Estimating attributes: analysis and extensions of RELIEF. In *European Conference on Machine Learning* (1994), pp. 171–182.

[40] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR) 50*, 6 (2018).

[41] Liu, H., and Hiroshi, M. *Computational Methods of Feature Selection*. CRC Press, 2007.

[42] Massey, F. J. The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of the American Statistical Association* (1951).

[43] Nemenyi, P. B. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, Department of Mathematics, 1963.

[44] Neyman, J., and Pearson, E. S. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* (1933).

[45] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* (2011).

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

[47] Pisheh, F., and Vilalta, R. Filter-based information-theoretic feature selection. In *Proceedings of 3rd International Conference on Advances in Artificial Intelligence* (2019).

[48] Pohlert, T. The pairwise multiple comparison of mean ranks package (PMCMR). `https://cran.r-project.org/web/packages/PMCMR/index.html`, 2014. Accessed: 2019-09-12.

[49] Robnik-Šikonja, M., and Kononenko, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning 53*, 1-2 (2003), 23–69.

[50] Saeys, Y., Abeel, T., and Van De Peer, Y. Robust feature selection using ensemble feature selection techniques. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2008), Springer.

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

[52] Shannon, C. E. A mathematical theory of communication. *The Bell System Technical Journal 27*, July 1928 (1948), 379–423.

[53] Shapiro, S. S., and Wilk, M. B. An analysis of variance test for normality (Complete samples). *Biometrika* (1965).

[54] Shishkin, A., Bezzubtseva, A., and Drutsa, A. Efficient high-order interaction-aware feature selection based on conditional mutual information. *Advances in Neural Information Processing Systems*, Nips (2016), 1–9.

[55] Sun, Y., and Li, J. Iterative RELIEF for feature weighting. In *Proceedings of the 23rd International Conference on Machine Learning* (2006), ACM, pp. 913–920.

[56] Sun, Y., Todorovic, S., and Goodison, S. Local-learning-based feature selection for high-dimensional data analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 9 (2010), 1610–1626.

[57] Tang, X., Dai, Y., Xiang, Y., and Luo, L. An interaction-enhanced feature selection algorithm. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2018), Springer, pp. 115–125.

[58] Terpilowski, M. scikit-posthocs: Pairwise multiple comparison tests in Python. *The Journal of Open Source Software 4*, 36 (2019), 1169.

[59] Tukey, J. W. Comparing individual means in the analysis of variance. *Biometrics* (1949).

[60] Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., and Moore, J. H. Relief-based feature selection: Introduction and review. *Journal of Biomedical Informatics 85* (2018), 189–203.

[61] Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biometrics Bulletin* (1945).

[62] Yang, H., and Moody, J. Feature selection based on joint mutual information. In *Proceedings of International ICSC Symposium on Advances in Intelligent Data Analysis* (1999), Citeseer, pp. 22–25.

[63] Yu, A., and Grauman, K. Predicting useful neighborhoods for lazy local learning. *Advances in Neural Information Processing Systems* (2014), 1916–1924.

[64] Yu, L., and Liu, H. Feature selection for high-dimensional data: A fast correlation-based filter solution. *International Conference on Machine Learning (ICML)* (2003), 1–8.

[65] Zar, J. Biostatistical analysis. 4nd. *Prentice Hall USA* (1999).

[66] Zeng, Z., Zhang, H., Zhang, R., and Yin, C. A novel feature selection method considering feature interaction. *Pattern Recognition 48*, 8 (2015), 2656–2666.

[67] Zhao, Z., and Liu, H. Searching for interacting features. In *IJCAI International Joint Conference on Artificial Intelligence* (2007), pp. 1156–1161.