WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia IJCNN

Combining feature ranking algorithms through rank aggregation

Ronaldo C. Prati
Centro de Matemática, Computação e Cognição (CMCC)
Universidade Federal do ABC (UFABC)
Santo André, SP, Brazil
Email: ronaldo.prati@ufabc.edu.br

***Abstract*—The problem of combining multiple feature rankings into a more robust ranking is investigated. A general framework for ensemble feature ranking is proposed, alongside four instantiations of this framework using different ranking aggregation methods. An empirical evaluation using 39 UCI datasets, three different learning algorithms and three different performance measures enable us to reach a compelling conclusion: ensemble feature ranking do improve the quality of feature rankings. Furthermore, one of the proposed methods was able to achieve results statistically significantly better than the others.**

I. Introduction

Feature selection has been an important research topic of pattern recognition, machine intelligence and data mining for a long time now [1], [2], [3]. There are plenty of studies pointing out that a subset of features may produce better predictive models (in terms of accuracy) than the entire feature set. This is because learning algorithms may be adversely affected by the presence of irrelevant and/or redundant features. Besides improving classification accuracy, feature selection significantly reduces the computational time necessary to induce the models, leading to simpler and faster classifiers for classifying new instances; facilitates data visualization and data understanding; and reduces the measurement and storage requirements.

Feature selection can be broadly divided into three categories. Filter methods are directly applied to datasets and generally provide an index for each feature quality by measuring some properties in the data. Wrapper and embedded methods, on the other hand, generally use a learning algorithm to indirectly assess the quality of each feature or feature sets. Wrapper uses the classification accuracy (or other performance measure) of a learning algorithm to guide a search process in the feature space and embedded methods use the internal parameters of some learning algorithms to evaluate features.

A relaxed formalization of feature selection is feature ranking. In the feature ranking setting, a ranked list of features is produced and one can select the top ranked features, where the number of features to select can be analytically or experimentally determined or set by the user. Many feature selection algorithms use feature ranking as a principal or auxiliary step because of its simplicity, scalability, and good empirical success. Furthermore, a ranked list of feature might be interesting by itself, as for instance in the microarray analysis, where the ranked list of features is used by biologists to find correlations among top ranked features and some diseases [3].

Likewise as finding the best subset of features is impractical in most domains, we may expect that algorithms that construct the best ranking of features are unfeasible. Inspired from ensemble learning, where a combination of models is used to improve predictive performance, in this paper we investigate methods whereby the combination of independent feature rankings would result into a (hopefully) more robust feature ranking. These methods are based on different ranking aggregation approaches, and may take as input rankings produced by any combination of different feature ranking algorithms.

The main contributions of this paper are:

1) A general and flexible framework for combining feature rankings. This framework is based into ranking aggregation mechanisms, and allows different instantiations depending on how the input base rankings are obtained as well as how the aggregation is carried out;
2) An empirical investigation of four concrete instantiations of this framework. These instantiations are based into four different aggregation approaches and are simple to implement, computationally cheap and in all but one case do not introduce new free parameters to set up;
3) An extensive empirical evaluation and analysis involving 39 datasets, three different learning algorithms and three different classification evaluation measures. Statistical analysis of the empirical evaluation enable us to reach a compelling conclusion: combining feature rankings do improve the quality of the ranked list of features. Furthermore, in our experiments, one of the proposed instantiations excels the others.

This paper is organized as follows: Section II presents related work. Section III presents a general framework for ensemble feature ranking. Section IV describes the four instantiations of this framework using four different ranking aggregation approaches studied in this paper. Section V describes the experimental setup used to evaluate the methods. Section VI presents and discusses the results, and Section VII concludes.

II. Related Work

Combining different feature selection methods has received considerable attention in recent years, thanks principally to the interest in the development of reliable and stable feature selection methods for gene selection in bioinformatics tasks [4], [5], [6]. Similar procedures based on ranking aggregation have been independently proposed by different authors. In [4], [5], rankings obtained from the same feature ranking algorithm ran in different samples of a dataset are combined through rank averaging — a ranking aggregation technique equivalent to the Borda method described in Section IV. The motivation is to construct a more robust ranking to avoid bias due to sampling variations, a critical issue in gene selection problem due to the high dimension and low sample size. In [6], a similar procedure using rank averaging is proposed, although using different ranking algorithms to obtain the input rankings rather than a single algorithm in different data samples. The motivation is also to soften the course of dimensionality/low sample size in gene selection tasks. Another algorithm similar do Borda was developed in [7], although the input rankings were based in a unsupervised approach rather than supervised ranking.

In [8] the ranking obtained by the aggregation of several rankings generated by different ranking algorithms, combined through rank averaging, are used to guide a sequential hill-climbing backward elimination procedure. The use of the aggregated ranking considerably outperforms the use of individual rankings on several UCI datasets. In [9] a similar approach is investigated, although using a forward feature inclusion rather than backward elimination. The author shows that the aggregated ranking is, in general, more stable than individual rankings according to the stability criterion she derived in the paper. In [10], the authors proposed an algorithm which combines different feature rankings generated by ROGER [11] — a genetic algorithm that outputs a linear combination of features which maximizes the area under the ROC curve — by reordering the features according to the frequency a feature appears better ranked than another into a paired analysis. This approach is similar to the Condorcet criterion described in Section IV, although the authors assume that a Condorcet winner always exists (a constraint that is not always satisfied in practice).

A somehow related approach is [12], where the authors use traditional ensemble approaches (in this case, ensembles of decision trees named Random forest (RF)) to perform feature selection. RF is a random subspace method that combines trees generated from random subsets of features from the original feature set, and is capable of efficiently ranking features for large data sets [13]. The authors exploit this property of RF, augmenting the original data with artificial contrast variables constructed independently from the target, and use their ranking for removal of irrelevant variables from the original set. Redundancy elimination is performed by using a variable masking measure that incorporates surrogate variable scores from ensembles of trees. Another approach has been recently proposed in [14], where a ensemble of probabilistic distance measures is proposed to evaluate features.

Our framework, described in the next section, generalizes these mentioned approaches by not being tailored to any particular feature ranking and by considering different aggregation approaches.

III. A General Framework for Ensemble Feature Ranking

Let $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be a dataset consisting of n examples or instances. Each instance $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ is a vector of m values, where each value x_{ij} of this vector represents a certain property (a feature) of that instance. The vector $\mathbf{f}_j = (x_{1j}, x_{2j}, \ldots, x_{nj})^t$ is the vector of values of a feature $\mathbf{f}_j$. In other words, D can be seen as a $n \times m$ matrix, where each row i is the instance $\mathbf{x}_i$ and each column j is the feature $\mathbf{f}_j$. Features can be either discrete or continuous. In so-called supervised problems, a feature of particular interest is called the target feature $\mathbf{f}_{target}$. When this target feature is discrete, this problem is known as a classification problem, and the value $x_{i,target}$ is called the class of the instance i.

A feature ranking algorithm R applied to the dataset D produces an ordered list of features $\pi = [\mathbf{f}_{-}^1, \mathbf{f}_{-}^2, \ldots, \mathbf{f}_{-}^j]$, where the superscript denotes the position in the ranked list of a certain feature $\mathbf{f}_{-}$, and this list is ordered by decreasing importance with respect to $\mathbf{f}_{target}$. Ties are allowed and are indicated by the same superscript for the tied features, that is, if features $\mathbf{f}_a$ and $\mathbf{f}_b$ are tied at the rank w, and they appear as $\mathbf{f}_a^w$ and $\mathbf{f}_b^w$ in the ranking. Furthermore, this ranking might be partial, that is, not all features appear in the ranking. Based on this ranking we can select a subset of the top k ranked features $[\mathbf{f}_{-}^1, \ldots, \mathbf{f}_{-}^k]$, $k \leq m$, where k can be set by the user or adjusted analytically or experimentally.

Different ranking algorithms may produce different rankings. Considering that many ranking algorithms are available, and that it is unlikely that exists an "universal" feature ranking algorithm, that is, a feature ranking that always produces the best ranking w.r.t. the target feature, it is plausible to assume that a combination of different feature rankings (called ranking aggregation) might produce a better ranking than individual rankings. The motivation behind combining different feature rankings is similar to ensemble methods in supervised learning, where multiple models were used to obtain better predictive performance than could be obtained from any of the constituent base models[15], [16].

In general, ensemble methods involve two steps: the first one is creating different base models, each providing their output, while the second step is to compose the different outputs into a single model. Different approaches can be used to generate the base input feature rankings. The most common are:

1) Variation of the ranking algorithm: Different feature ranking algorithms can be applied to the same dataset so that each ranking algorithm produces a different ranking.

2) Variation of the instances: Different subsamples of the dataset can be generated by sampling the instances with or without replacement, and the same feature ranking algorithm

be applied to these subsamples, producing a different ranking for each subsample.

3) Variation of the features: For feature ranking algorithms that are not univariate, that is, methods that not only uses information of the feature and the target feature but also take correlation to other features into account, different subsamples can be generated by taking different subset of features and the same feature ranking algorithm be applied to these subsamples, generating a different ranking for each subset of features.

4) Hybrid approaches: a combination of the previous mentioned methods.

The first two approaches generate complete rankings, that is, rankings contemplating all features while the third approach (or any combination which involves this approach) generates partial rankings.

The motivation for using feature ranking algorithms to construct ensembles is twofold. First, as stated before, many feature selection methods uses feature ranking as a principal or auxiliary step because of its simplicity, scalability, and good empirical success [3]. Filter methods naturally provide a feature quality score that can be used to rank features. For wrapper based methods, the predictive performance of the algorithm used as a wrapper can be used to rank features. For instance, the difference in accuracy of the predictive model with and without the feature could be used to rank features. For embedded methods, some information inside the model can be used for ranking. For instance, in decision trees, the level of the node in a tree can be used as a ranking and in SVMs, the ranking can be based on weights of the support vectors, as in [17]. Therefore, ensembles of feature ranking may be build upon a large number of methods already proposed in the literature.

Second, ranking aggregation methods can be used to combine the base rankings. Ranking aggregation is generally based on order-based aggregation, that is, only the orders of the attributes in the ranking are taken into account. There are several advantages of using order-based aggregation. One of them is that order-based aggregation is naturally calibrated and scale insensitive. If we want to do a score-based aggregation, we generally should firstly rescale the values to the same range (say, between 0 and 1) so that different absolute scales do not influence in the aggregate result. Furthermore, even though values might be in the same absolute scales, they may represent different relative scales. In other words, the same score (say 0.6) might represent different feature relevance for two different feature scoring algorithms. To avoid the influence of different relative scales when aggregating scores we must introduce some weight factor to "calibrate" scores. These weights are not generally easy to determine, and in general are set up empirically. Other advantages of rank aggregation are that they are generally computationally cheap, and have none or few free parameters to set up.

Within this framework, combining feature selection methods can be performed in a very general and flexible way. Our technique is not tailored nor limited to a specific feature ranking method. As long as a feature selection method produces a ranked list of features, this ranking can be used to construct a base ranking that can be used as input for constructing the ensemble. Furthermore, using ranking aggregation is a simple yet elegant and effective technique in the sense that they are easy to implement and use, and can naturally handle uncalibrated methods, partial lists and ties. The next section describes the four ranking aggregation methods we have implemented in this study.

IV. Ranking aggregation methods

Ranking aggregation is a classical problem from social choice and voting theory. Roughly speaking, the rank aggregation problem is to combine many different rank orderings on the same set of candidates (the base-rankings), in order to obtain a "better" ordering [18]. In our setting, candidates are features and base-rankings are different rank orderings of these features.

An aggregation method takes as input a number of (partial) ranked lists and produces as output another ranked list. We have implemented four different rank aggregation methods to test whether they produce good rankings for the problem of combining different feature rankings. The aggregation methods we have implemented are described next.

Borda (BC): The Borda count of an element is its mean position in the input rankings, that is, Borda(i) = $\sum_{j=1}^{n} \pi_j(\mathbf{f}_i)$, where $\pi_j(\mathbf{f}_i)$ is the rank of feature $\mathbf{f}_i$ in the ranking π_j. The Borda algorithm ranks the elements by increasing order of Borda counts.

Condorcet (CD): Consider a pairwise comparison between the ranks of two features. The Condorcet criterion states that if there is some alternative which defeats ever other in simple pairwise comparisons, then that alternative is the "Condorcet winner". Based on this property, the Condorcet aggregation method we have implemented works as follows: for each input ranking, compare the rank $\pi_j(\mathbf{f}_i)$ of a feature $\mathbf{f}_i$ on to the rank of every other feature, one pair at a time (pairwise), and tally a "win" for the higher-ranked feature. Sum these wins for all rankings, maintaining separate counts for each pairwise combination. The feature which wins every one else on their pairwise contests is the most preferred over all other features, and hence the feature which should appear at the top of the aggregated ranking. This feature is removed and another Condorcet winner can be calculated, which is ranked second in the aggregated ranking, and so forth. In case it is not possible to determine a single Condorcet winner, we opt for a random tie break among winners.

Schulze (SSD): Also known as Schwartz Sequential Dropping (SSD), this method also obeys the Condorcet criterion. For each pair $\mathbf{f}_x$ and $\mathbf{f}_y$ of features we count how many rankings ranks $\mathbf{f}_x$ over $\mathbf{f}_y$ and how many rankings ranks $\mathbf{f}_y$ over $\mathbf{f}_x$. If the first number is larger, then $\mathbf{f}_x$ defeats $\mathbf{f}_y$; if the second number is larger, $\mathbf{f}_y$ defeats $\mathbf{f}_x$; if both numbers are equal, there is a tie. We can construct a graph from this information as follows: the features are the vertices, and whenever $\mathbf{f}_x$ defeats $\mathbf{f}_y$ we add an edge from $\mathbf{f}_x$ to $\mathbf{f}_y$. This process leads to a graph that always has at least one cycle or single element,

which is not defeated by other options. The collection of all these elements is called the Schwartz set. If there are cycles in the Schwartz set, we remove edges from the graph to break these up. In a cycle, the edge connecting $\mathbf{f}_x$ to $\mathbf{f}_y$ is removed if the number of rankings where $\mathbf{f}_x$ is over $\mathbf{f}_y$ is minimal, considering the set of edges in the cycle. If there are several edges with this number of votes, they are all removed at once. Removing edges can make the Schwartz set smaller. We repeat this procedure until the Schwartz set contains no more cycles. After we have broken up all cycles, the Schwartz set consists of isolated nodes only, and the corresponding options are the winners. These nodes were removed from the graph and a new group of winners can be computed. Features are ranked in the aggregated ranking according to the winner sequence they appear. Typically there is only one winner per round. When there is more than one winner, ties are broken randomly.

Markov Chain(MC4): In [18] it is proposed methods to construct aggregate rankings based on Markov chains. One of these methods (MC4) is similar to the Google PageRank algorithm. In our feature ranking setting, the states of the chain correspond to the features to be ranked, the transition probabilities depend in some particular way on the input rankings, and the stationary probability distribution will be used to order the features to produce the aggregated ranking. The algorithm works as follows: the markov chain is represented by a graph. Each feature corresponds to a node in a graph. For each pair $\mathbf{f}_x$ and $\mathbf{f}_y$, if $\mathbf{f}_x$ appears above $\mathbf{f}_y$ in a ranking, a weighed directed edge joining the nodes $\mathbf{f}_x$ and $\mathbf{f}_y$ is created, with an weight proportional to the distance between these features in the ranking. These weights are rescaled to $0-1$ range to represent transition probabilities. If this node already is present in the graph, only the weight is updated by adding to the previous weight a value proportional to the distance between the two features in the ranking. After the graph is constructed, the PageRank algorithm is run in this graph till convergence, and feature are ordered in decreasing order according to the "prior importance" of each node in the graph. This method has a parameter α, which is the importance of each node at the begining of the execution of the PageRank algorithm.

CD, MC4 and SSD naturally handles partial lists and ties in the input rankings. In order to BC handle partial lists we must assume that the all items not ranked appear at the bottom of the ranked list and ties are handled by assigning average ranks[1]. Note however that although ties are allowed in the input ranking, they are randomly broken in the aggregate ranking. This tie breaking approach for the aggregate rankings was used due to the stepwise evaluation procedure described in Section V-B. BC was used in [4], [5], [6], [8], [9] and an approach similar to CD was used in [10] to combine rankings of features. We are not aware of the use of MC4 or SSD in the same context.

[1]If two features are tied at the 5^th position for instance, the average rank for both features is 5.5

V. Experimental Setup

A. Datasets

To evaluate the methods proposed in this paper we carried out an extensive experimental evaluation using 39 datasets from UCI [19]. The criterion to choose datasets was to select classification datasets having at least 10 features. A summary of the main characteristics of these datasets is shown in Table I. For each dataset, this table shows the dataset name, number of instances, number of features, number of classes and the percentage of instances in the majority class.

TABLE I: Summary of the datasets used in the experiments

name	# inst.	# feat	# classes	% maj. class
anneal	898	39 (6/33)	6	76,17
arrhythmia	452	280 (206/74)	16	54,20
audiology	226	70 (0/70)	24	25,22
autos	205	26 (15/11)	7	32,68
breast-cancer	286	10 (0/10)	2	70,28
wisc-breast-cancer	699	10 (9/1)	2	65,52
bridges-version1	107	12 (3/9)	6	41,12
bridges-version2	107	12 (0/12)	6	41,12
cmc	1473	10 (2/8)	3	42,70
horse-colic	368	23 (7/16)	2	63,04
horse-colic.ORIG	368	28 (7/21)	2	63,04
credit-rating	690	16 (6/10)	2	55,51
german-credit	1000	21 (7/14)	2	70,00
cylinder-bands	540	40 (18/22)	2	57,78
dermatology	366	35 (1/34)	6	30,60
glass	214	10 (9/1)	7	35,51
heart-disease	303	14 (6/8)	5	54,46
heart-statlog	270	14 (13/1)	2	55,56
hepatitis	155	20 (6/14)	2	79,35
ionosphere	351	35 (34/1)	2	64,10
synthetic-control	600	62 (60/2)	6	16,67
kr-vs-kp	3196	37 (0/37)	2	52,22
labor	57	17 (8/9)	2	64,91
letter	20000	17 (16/1)	26	4,07
lung-cancer	32	57 (0/57)	3	40,63
lymphography	148	19 (3/16)	4	54,73
mfeat	2000	217 (216/1)	10	10,00
promoters	106	58 (0/58)	2	50,00
mushroom	8124	23 (0/23)	2	51,80
optdigits	5620	65 (64/1)	10	10,18
page-blocks	5473	11 (10/1)	5	89,77
pendigits	10992	17 (16/1)	10	10,41
primary-tumor	339	18 (0/18)	22	24,78
segment	2310	20 (19/1)	7	14,29
sonar	208	61 (60/1)	2	53,37
soybean	683	36 (0/36)	19	13,47
spambase	4601	58 (57/1)	2	60,60
splice	3190	61 (0/61)	3	51,88
sponge	76	45 (0/45)	3	92,11

As can be seen in this table, the datasets have a great diversity, with the number of instances ranging from 32 to $20,000$, the number of classes ranging from 2 to 26, the number of features ranging from 10 to 280 and the prevalence of the most frequent class ranging from 4.07% to 92.11%.

B. Performance curves

The efficiency of a feature ranking algorithm can be directly measured by comparing the ranking it produces to the true ranking. A simple and effective direct evaluation criterion it to compute the rank correlation between the obtained ranking and the true ranking. However, this can only be done in synthetic data for which we know beforehand the true ranking. For real world data we often do not have this true ranking, and we have to evaluate it indirectly by using the performance of a predictive model as an indirect measure [20].

We follow the approach described in [6] to evaluate the feature rankings. We would expect that good rankings should put on top of the rank the most important features (w.r.t. some class feature), leaving at the bottom of the list the least important ones. Intermediate features would be somewhere in between, ordered by decreasing ranking importance. Therefore, to evaluate the ranking we perform a stepwise subset evaluation, generating a "performance curve[2] ". This stepwise evaluation was performed as follows: given a dataset D and an arbitrary ranking algorithm R, let $\pi = [\mathbf{f}_{-}^{1}, \mathbf{f}_{-}^{2}, \ldots, \mathbf{f}_{-}^{m}]$ be the ranked list obtained by applying R on D, where $\mathbf{f}_{-}^{1}$ denotes the top-ranked feature, $\mathbf{f}_{-}^{2}$ the second ranked feature and so forth. To evaluate this ranking, we generate n subsets $\{D_1, D_2, \ldots, D_n\}$ from the original dataset D, where each subset D_i is constructed using the top i ranked features, that is, $D_1 = \{\mathbf{f}_{-}^{1}\}$, $D_2 = \{\mathbf{f}_{-}^{1}, \mathbf{f}_{-}^{2}\}$, and so forth. For each dataset D_i, a predictive model is built using a learning algorithm L, and the a performance measure M_i is calculated into a separated test set T_i (that contains the same set of features as in D_i). The performance curve is formed by interpolating the $[M_1, M_2, \ldots, M_m]$ points of the M_m estimated performance measures.

To generate the performance curves we have used three different learning algorithms, all three implemented in the Weka toolkit [21], with default parameters: J4.8 (which is weka's C4.5 decision tree implementation), Naïve Bayes and SMO (weka's implementation of Support Vector Machines — SVM — using Sequential Minimal Optimization) and three different performance measures: error rate (the percentage of instances incorrectly classified by the induced predictive model), AUC (the area under the ROC curve; when multi-class datasets were used, the AUC was calculated using the M-measure [22]), and F_1 (the harmonic mean of precision and recall).

C. Input rankings

To generate the input rankings, we have used the following feature evaluation approaches often used to select subset of feaatures, as implemented in WEKA [21]. Features are ranked according to the descending order provided by the corresponding feature evaluation method.

[2]In [6], the curve is called "*error curve*," as only the error rate is used as performance measure. In this paper, as we consider three different performance measure (error rate, AUC and F1), we called it "*performance curve*."

1) Information Gain(IG): measures the gain in information entropy from using an attribute to split the instances into disjoint subsets according to the values of the attribute. It is used as a decision tree splitting criterion in ID3 [23].

2) Gain Ratio (GR): is the measure used as decision tree splitting criteria in C4.5. It is based in the information gain and was introduced by Quinlan [23] in order to avoid overestimation of multi-valued features. The gain ratio compensates for the number of attributes by normalizing by the information encoded in the split itself.

3) Symmetric uncertainty (SU): is also a measure based in the information gain, normalized by the feature entropy times the class entropy. This value is multiplied by two in order to be rescaled to the range between zero and one.

4) ChiSquared (χ^2): evaluate each feature individually by measuring the chi-squared statistic with respect to the class.

5) OneR (1R): evaluate each feature individually by using this feature alone to classify the target attribute.

6) ReliefF (RLF): was first developed in [24] and then substantially generalized and improved in [25]. It measures the usefulness of features based on their ability to distinguish between very similar neighbors examples belonging to different classes.

D. Experiments configuration

The experiments were carried out using 10-fold cross-validation. To avoid any bias in the experimental analysis due to sampling variances in the cross validation, the experiment are paired, that is, we used the same training and test sets for all methods. We ran all methods to generate input rankings as well as all classifiers using default parameters, as implemented in Weka [21]. For MC4, we used the implementation of PageRank available in Jung[3], with the α parameter set to 0.1. The training sets were used to generate the input rankings. These input rankings were given to the four ranking aggregation algorithms discussed in Section IV. Both input and aggregate rankings were used to generate classifiers using the stepwise subset evaluation procedure described in Section V-B, and these classifiers were evaluated in the test set to generate the performance curves.

We compare this approach to the approach proposed in [4]. This approach is referenced as linear (Lin), and the input rankings used were obtained by sampling with replacement at a rate of 90% the training set 10 times, and running Relief over each sample. In [4], both Relief and SVM recursive feature elimination (SVM-RFE) [17] were used to obtain the base rankings. We were not able to use the SVM-RFE to obtain the base rankings due to the computational complexity of this method. In [4], this was possible because they use microarray datasets, which have low sample sizes. Note that the base line methods contains 10 input rankings, while the other aggregation methods we have used contains only six.

[3]http://jung.sourceforge.net/

VI. Results and discussion

Due to lack of space numerical results are not presented in this paper. To have an overall idea of the performance of each approach, we calculated the area under each performance curve and in order to analyze whether there are differences among the methods, we ran the Friedman test, at 95% of confidence level. The Friedman test is the nonparametric equivalent of the repeated-measures ANOVA. Due to lack of space, only results of these tests are reported here. When the null-hypothesis is rejected by the Friedman test, we can proceed with a post-hoc test to detect what differences among the methods are significant. For this purpose, we ran the Nemenyi test, which is similar to the Tukey test for ANOVA and is used when all methods are compared to each other. See [26] for a thorough discussion regarding statistical tests in machine learning research.

To facilitate the analysis, we grouped the experiments into three groups. First, we grouped results by the performance measure used to draw the performance curve. The objective of this grouping is to test whether the performance of the feature ranking algorithms are robust to different performance measure or whether the use of a particular measure privileges a particular ranking method. The Friedman tests ran independently for all datasets grouped by the three different performance measures used to evaluate the results (F_1, error rate and AUC) rejects the null hypothesis that all ranking algorithms produce equivalent results and the results of the post-hoc Nemenyi test are summarized in Figure 1, where in (a), (b) and (c) are shown the graphical results for the measures F_1, error rate and AUC, respectively. In these graphs, results were ordered by decreasing performance, where the best ranking algorithms are placed to the right of the figure. A thick line joining two or more methods indicates that there is no statistical significance among those methods.

Overall, the graphs presented in Figure 1 show a clear pattern. The aggregated rankings perform better than base rankings, no matter the performance measure used to evaluate the models. Furthermore, apart from F1, where MC4 is not significantly better than 1R, all the aggregate rankings are significantly better than the input rankings. SSD was the best aggregation technique, no matter the measure used for evaluation. For error and AUC, SSD was significantly better than any other ranking, either the base rankings or the aggregate rankings. For F_1, SSD is not significantly better than CD, BC and Lin. For the other two evaluation metrics considered to evaluate performance, SSD is significantly better than the base line Lin. The other aggregated rankings, in general, form a second group although it is not possible to detect whether one method surpass another. The input rankings form a third group, and IG always appears with the worst performance, although it is not possible, in general, to point out significantly differences among them for all three measures.

In the second group, results were grouped according to the classifier used to generate the models. The objective of this grouping is to test whether the performance of of feature

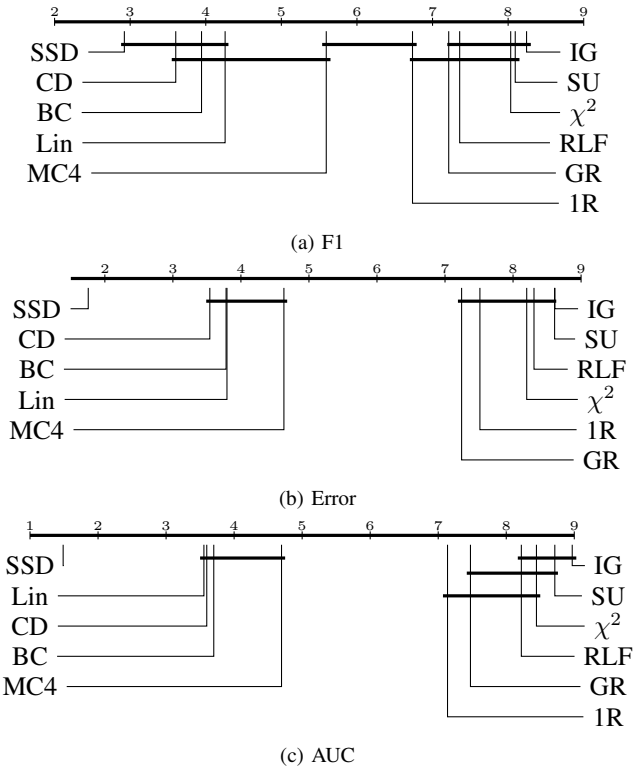
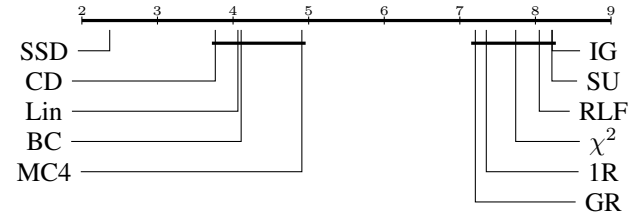


Fig. 1: Results of the Nemenyi test grouped by the three different measures used to evaluate the feature rankings. Methods are ordered by performance from left to wright. A thick line joining two or more methods indicates that there is no significantly difference among methods.

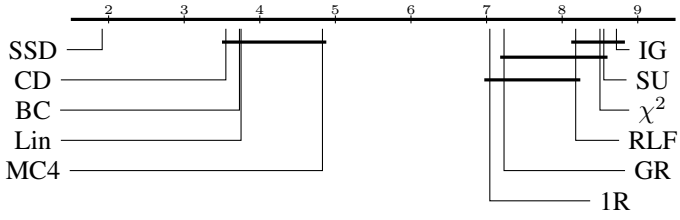
ranking methods are robust to different learning algorithms used to build the predictive models or whether the use of a particular learning algorithm privileges a particular ranking method. The Friedman tests ran independently for all datasets grouped by the three different learning algorithms used to evaluate the results (J48, Naïve Bayes and SVM) rejects the null hypothesis that all rankings algorithms produce comparable results and results of the post-hoc Nemenyi test are summarized in Figure 2, where in (a), (b) and (c) are shown the graphical results for the learning algorithms J48, SVM and Naïve Bayes respectively.

The results grouped by learning algorithms show a scenario very similar to the results grouped by performance measure. The aggregate rankings are clearly better than the base rankings, no matter the learning algorithm used to induce the models, as all aggregate rankings are significantly better than the base input rankings. Furthermore, SSD is significantly better than any other ranking method, either aggregated or base rankings, no matter the learning algorithm used to evaluate. Apart from Naïve Bayes, where MC4 is significantly worse than CD, BC and Lin, the other aggregation approaches, including the base line Lin, form a second group where no statistical significance could be identified. The base rankings form a third group, where for J48 it is not possible to detect

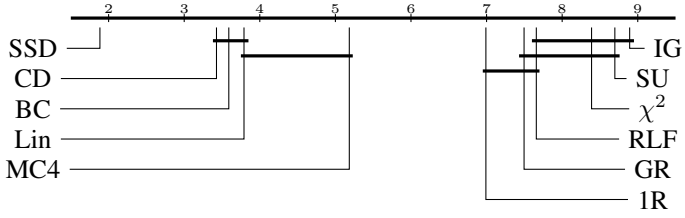
significant differences among all the base methods, while for SVM and Naïve Bayes it is not possible to detect differences among 1R, GR and RLF; GR, RLF, χ^2 and SU; as well as among RLF, χ^2, SU and IG.



(a) Decision Tree



(b) SVM



(c) Naïve Bayes

Fig. 2: Results of the Nemenyi test grouped by the three different algorithms used to evaluate the feature rankings. Methods are ordered by performance from left to wright. A thick line joining two or more methods indicates that there is no significantly difference among methods.

Finally, we test whether there are significant differences considering any learning algorithm and evaluation metric used to evaluate performance. The Friedman tests ran on all the results rejects the null hypothesis that all rankings algorithms produce comparable results and results of the post-hoc Nemenyi test are summarized in Figure 3. For this comparison, SSD excels any other ranking, either the aggregate rankings or the base rankings. CD, BC, MC4 and Lin forms a second group, where it is not possible to detect significant differences among them. The base input rankings form three groups where it is not possible to detect significant differences: 1R, GR, RLF and χ^2; GR, RLF, χ^2 and SU; and RLF, χ^2, SU and IG.

Overall, the aggregate rankings performed quite well, and are significantly better than any base input ranking. This is strong evidence that aggregating different feature ranking do improve performance. The performance of MC4 is somehow disappointing, as it was the worst method among the aggregate rankings. However, this may be due to the non attempt to adjust the α parameter of the PageRank algorithm. Although we used a value that is commonly used in the literature, it has

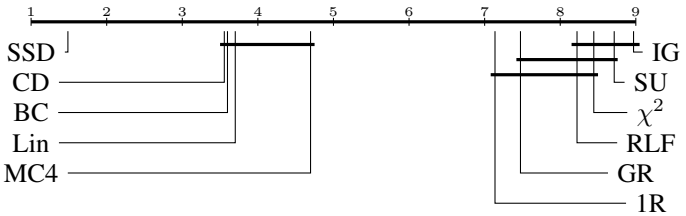


Fig. 3: Overall comparison among feature ranking methods. Methods are ordered by performance from left to wright. A thick line joining two or more methods indicates that there is no significantly difference methods.

not adjusted for this particular problem. Furthermore, SSD also was significantly better than the aggregated methods, including the base line. Based on this result, we may recommend SSD to combine feature rankings. However, it should be keep in mind that for some particular cases, other methods may produce better results than SSD. This is the case of the use the dataset cylinder-bands when J48 was used for evaluation, where MC4 produced good results and the other aggregation methods fail. For other methods, there is little or no differences among aggregated and input rankings, as in the case of sponge dataset using J48 for evaluation.

Another important aspect is the computational complexity of the ranking aggregation methods. Discounting the time to build the base rankings, BC only need to compute the average ranks of each features, and the computational cost is $O(j)$, where j the number of features. CD requires a computational cost of $O(j^2)$ to run, as it verifies all pairwise combination of features. A properly implementation of SSD have a computational cost of $O(j^3)$ [27]. MC4 also have a computational cost of $O(j^3)$, although in practice a linear time is required to converge in most of the cases [18]. Although the computational cost is considerably large for SSD compared to BC, for most applications where the number of features is not very large, the actual running time is not a problem. In our experiments, most of the datasets run in a couple of minutes in computer with a intel core i3 processor, 4GB of run, and iOS operating system. For problems with a large number of features, an hybrid approach can be considered, where a computationally cheap method is used to discard the lower portion of the feature ranking (and thus discarding the most irrelevant features) and a complex method is used to compute the aggregate ranking with the most relevant ones. We believe this would not decrease classification performance significantly, as in our experiments the larger variations among the methods are in the top part of the feature ranking, although additional experiments should be conducted to confirm this hypothesis.

VII. Conclusion

This paper investigates methods to construct ensembles of feature rankings. These methods take as input feature rankings that can be generated by a wide range of feature evaluation

methods and construct a new ranking using different ranking aggregation procedures. Four different rank aggregation methods were investigated. Empirical results of an extensive experimental evaluation using 39 datasets from UCI, three different learning algorithms and three evaluation measures for predictive classification performance shows the suitability of using rank aggregation to derive better feature rankings when compared to the base rankings taken as input. Furthermore, in the overall comparison, the Schulze aggregation method (SSD) excels the other methods using different learning algorithms and different performance measures to evaluate the ranked list of features, and may be considered as the first choice to compose ranking of features.

An interesting direction for future research is how to transform the ensemble feature ranking method into a ensemble feature selection method, that is, how to automatically select a subset of features based on the ranked list of features. One option is to use an approach similar to [8], using a separated validation set to evaluate the quality of the feature sets. Another option is to use an approach similar to [12], introducing artificially generated features that work as cut of points for discarding irrelevant features. Finally, it would be interesting to investigate adaptive aggregation methods that remove base rankings which may degrade performance, as well as investigates the stability of the aggregation methods evaluated in this paper.

Acknowledgments: Part of this work was carried out when the author was a post-doc researcher fellow at Instituto de Ciências Matemáticas e Computação of Universidade de São Paulo, Brazil. This work was supported by the Brazilian research councils FAPESP and CNPq.

REFERENCES

[1] K. Shima, M. Todoriki, and A. Suzuki, "SVM-based feature selection of latent semantic features," *Pattern Recogn. Lett.*, vol. 25, no. 9, pp. 1051 –1057, 2004.

[2] H. K. Ekenel and B. Sankur, "Feature selection in the independent component subspace for face recognition," *Pattern Recogn. Lett.*, vol. 25, no. 12, pp. 1377–1388, 2004.

[3] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[4] Y. Saeys, T. Abeel, and Y. V. de Peer, "Robust feature selection using ensemble feature selection techniques," in *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2008)*, ser. Lecture Notes in Computer Science, W. Daelemans, B. Goethals, and K. Morik, Eds., vol. 5212. Springer, 2008, pp. 313–325.

[5] T. Abeel, T. Helleputte, Y. V. de Peer, P. Dupont, and Y. Saeys, "Robust biomarker identification for cancer diagnosis with ensemble feature selection methods," *Bioinformatics*, vol. 26, no. 3, pp. 392–398, 2010.

[6] I. Slavkov, B. Ženko, and S. Džeroski, "Evaluation method for feature rankings and their aggregations for biomarker discovery," in *Third International Workshop on Machine Learning in Systems Biology*, S. Džeroski, P. Geurts, and J. Rousu, Eds. Helsinki University Printing House, 2009, pp. 115–125.

[7] Y. Hong, S. Kwong, Y. Chang, and Q. Ren, "Consensus unsupervised feature ranking from multiple views," *Pattern Recogn. Lett.*, vol. 29, no. 5, pp. 595–602, Apr. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2007.11.012

[8] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard, "A hybrid wrapper/filter approach for feature subset selection," *Electronic Journal of SADIO*, vol. 8, no. 1, pp. 12–24, 2008.

[9] L. I. Kuncheva, "A stability index for feature selection," in *Proc. IASTED, Artificial Intelligence and Applications*, Innsbruck, Austria, 2007, pp. 390–395.

[10] K. Jong, J. Mary, A. Cornuéjols, E. Marchiori, and M. Sebag, "Ensemble feature ranking," in *8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'2004)*, ser. Lecture Notes in Computer Science, vol. 3202. Springer, 2004, pp. 267–278.

[11] M. Sebag, J. Azé, and N. Lucas, "Impact studies and sensitivity analysis in medical data mining with ROC-based genetic learning," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003)*. IEEE Computer Society, 2003, pp. 637–640.

[12] E. Tuv, A. Borisov, G. Runger, and K. Torkkola, "Feature selection with ensembles, artificial variables, and redundancy elimination," *Journal of Machine Learning Research*, vol. 10, pp. 1341–1366, 2009.

[13] L. Brieman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[14] R. Harrison, R. Birchall, D. Mann, and W. Wang, "A novel ensemble of distance measures for feature evaluation: Application to sonar imagery," in *IDEAL*, ser. Lecture Notes in Computer Science, H. Yin, W. Wang, and V. J. Rayward-Smith, Eds., vol. 6936. Springer, 2011, pp. 327–336.

[15] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, J. Kittler and F. Roli, Eds., vol. 1857. Springer, 2000, pp. 1–15.

[16] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "A comparison of decision tree ensemble creation techniques," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 173–180, 2007.

[17] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Mach. Learn.*, vol. 46, no. 1-3, pp. 389–422, 2002.

[18] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in *Proceedings of the Tenth International World Wide Web Conference*. ACM, 2001, pp. 613–622.

[19] A. Asuncion and D. Newman, "UCI machine learning repository," http://www.ics.uci.edu/~mlearn/MLRepository.html, 2007.

[20] H. D. Lee, M. C. Monard, R. F. Voltolini, R. C. Prati, and F. C. Wu, "A simple evaluation model for feature subset selection algorithms," *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, vol. 10, no. 32, pp. 9–17, 2006.

[21] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, 2005.

[22] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Mach. Learn.*, vol. 45, no. 2, pp. 171–186, 2001.

[23] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[24] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proceedings of the Ninth International Workshop on Machine Learning (ML 1992)*. Morgan Kaufmann, 1992, pp. 249–256.

[25] I. Kononenko, "Estimating attributes: Analysis and extensions of relief," in *European Conference on Machine Learning (ECML'1994)*, ser. Lecture Notes in Computer Science, vol. 784. Springer, 1994, pp. 171–182.

[26] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[27] M. Schulze, "A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method," *Social Choice and Welfare*, vol. 36, no. 2, pp. 267–303, 2011-02-01. [Online]. Available: http://dx.doi.org/10.1007/s00355-010-0475-4