FSS-OBOP: Feature subset selection guided by a bucket order consensus ranking

Juan A. Aledo
Departamento de Matemáticas, Universidad de Castilla-La Mancha, Albacete 02071, Spain
juanangel.aledo@uclm.es

José A. Gá
Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Albacete 02071, Spain
jose.gamez@uclm.es

David Molina
Departamento de Matemáticas, Universidad de Castilla-La Mancha, Ciudad Real 13071, Spain
david.molina@uclm.es

Alejandro Rosete
Instituto Superior Politécnico José Antonio Echeverría (Cujae), Marianao 19390, Havana, Cuba
rosete@ceis.cujae.edu.cu

***Abstract*—Several authors have ton the importance of aggregating the results of different feature selection methods in order to improve the solutions obtained. To the best of our knowledge, the consensus rankings obtained in all of these proposals do not allow that some variables are tied. This paper studies the advantages of allowing ties in the consensus ranking obtained from aggregating several features selection methods. This implies that the consensus ranking is modeled as the problem of obtaining the Optimal Bucket Order instead of solving the Rank Aggregation Problem. In this paper we propose a filter-wrapper algorithm, that we will call FSS-OBOP, which uses a filter-based consensus ranking with ties to guide the posterior wrapper phase. By using a benchmark with 12 high-dimensional datasets, we show that allowing ties in the consensus rankings leads to subsets that, when used to induce a classifier, obtain at least the same, when not better, accuracy. Furthermore, and what is actually more significant, they reduce the number of wrapper evaluations extraordinarily.**

1. Introduction

Given a dataset with n predictive variables, attributes or features $\mathcal{X} = \{X_1, \dots, X_n\}$ and a discrete (nominal) *target* variable known as *class* C, supervised classification consists in the induction from the available data of a *classifier* or function

$$\mathcal{C} : X_1 \times X_2 \times \cdots X_n \longrightarrow C$$

which generalizes well on unseen (new) data. *Supervised Feature Selection* (SFS) or *Feature Subset Selection* (FSS) is the problem of selecting a subset $\mathcal{S} \subseteq \mathcal{X}$ which will be used to induce the classifier instead of $\mathcal{X}$. When n is large, then $|\mathcal{S}| \ll |\mathcal{X}|$.

FSS is a very important problem in data mining and pattern recognition because of the increasing size of the attributes (variables) of databases [1], e.g. in biology [2]. FSS is also relevant since several studies have demonstrated that a subset of features may produce more accurate predictive models than the entire set [3]. Furthermore, feature selection is convenient to simplify (in terms of time, space and comprehensibility) the induced models [3].

In SFS the information provided by the class variable is used to guide the feature selection process. There are two main schemes: *filter* and *wrapper*, although they can be successfully combined in the so-called *filter-wrapper* approach [4], [5]. In the univariate filter approach, the predictive variables are scored according to their merit with respect to the class. Thus, information, distance or error-based measures are used, and then the variables are ranked from best to worse score and the best k variables are retained, k being a user-defined value. Multivariate filter approaches are also available. They work by approximating the merit of a subset by aggregating in some way the merit of the pairs or triplets of variables it contains [6]. On the other hand, in the *wrapper* approach the merit of a subset is directly assessed by inducing a classifier over the selected subset and (cross)validating it. Obviously, both approaches have their own (dis)advantages, mainly related with CPU time and generalization capability. Finally, the filter-wrapper approach arises as a combination that tries to take advantage from the two base methods. It starts by creating a ranking of variables by using univariate filter criteria, and then runs over the (first $k \leq n$) variables of the ranking by scoring in a wrapper way the subsets formed by the first r variables of the ranking ($r = 1, \dots, k$). In the simplest filter-wrapper approach, it means to reduce the number of induced/evaluated classifiers from polynomial ($\geq n^2$) to linear.

In this paper we focus on the univariate filter+wrapper approach, but instead of using a single ranking we produce a set of them by using re-sampling techniques. Then we aggregate these sets of rankings into a single one, but with the novelty of allowing ties in the obtained consensus, that is, variables can be arranged in groups in such a way that there is no preference among them. This *bucket order*-based ranking will be used to guide the wrapper phase. As a result, we expect to maintain or even increase the predictive power of the obtained classifier; but, over all, what we pursue is to reduce the number of (highly time consuming) wrapper evaluations. Our expectations are corroborated by the experiments carried out over 12 high-dimensional datasets, ranging from 500 to more than 50000 variables.

The rest of the paper is organized as follows. Section 2 briefly recaps some of the proposals in the literature dealing with rank aggregation-based FSS. Section 3 introduces the optimal bucket order problem. In Section 4 we formally present the FSS-OBOP algorithm, which is our proposal to approach the FSS problem by using a bucket order-based consensus ranking. In Section 5 we carry out an extensive experimental comparison over a benchmark of 12 high-dimensional datasets and, finally, in Section 6 we conclude and identify some possibilities to continue with this research.

2. Related studies

Recently, researchers in FSS have considered the use of a ranking of predictive variables which comes from the aggregation of a set of different rankings of variables, obtained by using different feature selection techniques of different samples of the original dataset. In this section we briefly review (classify) some of the works in this direction.

Several papers have been devoted to study the way in which the ranking is made. Thus, in [7] permutations of the most important variables (top-k lists) are aggregated, while in [8] and [9] the aim of the study is to compare the mean and the median as the criterion to be used in the aggregation process. In [10] a more flexible approach is followed, by allowing weighted input rankings and so using a weighted rank aggregation method.

Regarding the algorithms used to carry out the aggregation process, heuristic and *fast* greedy methods like Borda are mainly used [11]. However, in some papers more sophisticated search engines have been used, obtaining better results but needing, by far, more CPU time. This is the case of [12] and [13] where genetic algorithms are used to guide the search, or [14] where a Markov chain-based rank aggregation algorithm is applied.

With a different goal, several papers have devoted their effort to study the effect that the use of a consensus or aggregated ranking has in terms of stability and robustness. For example, [15], [16], [17], [18] claim that the use of a consensus ranking reduces the unstability of the selected subset with respect to sampling variations. The robustness against small variations in the scores used to construct the original rankings is analyzed in [3], [19], concluding that the consensus ranking is a more robust approach.

With respect to applications, consensus based feature selection has been applied to different real world problems: medical data [20], credit scoring [21], micro-array data [22], text sentiment classification [13], mass spectometry data [16], etc.

Finally, it is interesting to remark that in many of the previous papers, it is explicitly stated that ties in the final ranking could be of interest, but in practice they *are resolved randomly if necessary*. In this paper, we focus our study on filling this gap.

3. The optimal bucket order problem

In all the papers cited in the previous section, the goal of the aggregation is to obtain a ranking without ties, that is, a strict preference relation is defined over all the items (features) appearing in the ranking. Therefore, the problem approached is the one known as the *Rank Aggregation Problem* (RAP) [23]. Basically, the RAP can be defined as follows (see e.g. [24] for the details):

- Let $[[n]]$ be a set of items.
- Let $\Sigma = \{\sigma_1, \ldots, \sigma_r\}$ be a set of rankings, σ_i establishing a preference order for a subset $[[m_i]]$ of $[[n]]$, $i = 1, \ldots, r$.
- The solution to the RAP is the *consensus* ranking π_0, which is a permutation of (all) the elements in $[[n]]$ (or complete ranking without ties).
- π_0 is computed as the permutation having the smallest average distance to the rankings in Σ. Usually, the distance measures the number of disagreements between two given rankings.

If all the rankings in Σ are permutations of the elements in $[[n]]$, then the problem is known as the *Kemeny Problem* (KP) [25] and the Kendall distance is used as measure. If the rankings in Σ can be incomplete and/or have ties, then we are in the general setting or RAP, and a generalized Kendall distance is used as measure [26]. Both problems are NP-complete and usually tackled by using greedy heuristic algorithms [24], [27].

Although RAP has more degrees of freedom regarding the type of rankings in the input, both problems, KP and RAP, require to obtain a permutation as output. In this paper we argue in favor of allowing ties in the obtained consensus ranking. To the best of our knowledge, this constitute a novel approach to deal with the FSS problem. In feature selection, allowing ties may be convenient because of the following reasons:

- Semantics. Suppose we use several methods and obtain different rankings for a subset of variables[1]: $\{1|2|3|4, 2|1|3|4, 1|2|4|3, 2|1|4|3\}$. Then, if we want to aggregate them into a consensus one, it is obvious that all of them agree in that i is better that j for $i \in \{1,2\}$ and $j \in \{3,4\}$, but there is no consensus with respect to the preference between 1 and 2, and between 3 and 4. Hence, the most reasonable solution in this case would be $1,2|3,4$. However, by applying RAP (KP in this case) this solution is not allowed and the ties must be arbitrarily or randomly broken.
- Coherence. Suppose that in all the obtained rankings 1 and 2 are tied, that is, no preference relation between them is expressed. Even in this case, where there is no doubt, the algorithm solving RAP must break the tie in order to fulfill the requirements of RAP.

[1] Items between vertical bars are equally preferred or tied;$a|b$ means that a is preferred to b.

- Computational advantages. If we take the ranking to guide an ulterior wrapper FSS stage, then, as we will detail in Section 4, the presence of ties in the resulting ranking will reduce the number of wrapper evaluations.

To allow ties in the solution we rely on a different approach to aggregate the rankings: the *Optimal Bucket Order Problem* (OBOP) [28], where a bucket is a set of items that are tied. The output of the OBOP is a *bucket order*, instead of a strict preference ordering or permutation.

Next we formally introduce the OBOP, but adapting the presentation to the FSS problem, e.g. we talk of *features* instead of *items*, etc.

- Given a set of features $[[n]] = \{1, ..., n\}$, a *bucket order* $\mathcal{B}$ is an ordered partition of $[[n]]$ [28], [29]. More precisely, it is a linear ordering of disjoint subsets (*buckets*) $B_1, B_2, \ldots, B_k$ of $[[n]]$, $1 \leq k \leq n$, with $\cup_{i=1}^{k} B_i = [[n]]$.
- Given two buckets B_i, B_j in $\mathcal{B}$, we will write $B_i \prec_{\mathcal{B}} B_j$ to indicate that B_i precedes B_j according to the bucket order $\mathcal{B}$. Analogously, given two features $u \in B_i, v \in B_j$, we will write $u \prec_{\mathcal{B}} v$ if $B_i \prec_{\mathcal{B}} B_j$. All the features that belong to the same bucket are considered *tied*. Thus, if u, v are tied regarding $\mathcal{B}$, we will write $u \sim_{\mathcal{B}} v$.
- Associated to a bucket order $\mathcal{B}$ we will consider its associated *bucket matrix* B [28], which is the square matrix $n \times n$ such that $B(u,v) = 1$ if $u \prec_{\mathcal{B}} v$, $B(u,v) = 0$ if $v \prec_{\mathcal{B}} u$ and $B(u,v) = 0.5$ if $u \sim_{\mathcal{B}} v$. In particular, all the entries in the main diagonal of B are equal to 0.5 and $B(u,v) + B(v,u) = 1$ for all $u, v \in [[n]]$, $u \neq v$. Alternatively, we will write $u \prec_B v$, $v \prec_B u$ and $u \sim_B v$ to express that $B(u,v) = 1$, $B(u,v) = 0$ and $B(u,v) = 0.5$, respectively.
- A *pair order matrix* M of dimension $n \times n$ is a matrix with entries in the interval $[0, 1]$ and such that $M(u,v) + M(v,u) = 1$ for whichever $u, v \in [[n]]$, $u \neq v$, and $M(u,u) = 0.5$ for all $u \in [[n]]$. Usually $M(u,v)$ is interpreted as the probability that the feature u precedes the feature v regarding a given set of rankings.
- Given a pair order matrix M, the OBOP consists in finding a bucket matrix B $n \times n$ (that represents a bucket order) which minimizes

$$D(B, M) = \sum_{u,v} |B(u,v) - M(u,v)|. \qquad (1)$$

Since any solution of RAP is in the space of solutions of OBOP, OBOP can be considered as a generalization of RAP, and it is also NP-complete [29].

In [29] the *Bucket Pivot Algorithm* (BPA) was introduced to tackle with the OBOP and an approximation study of its performance was presented in [28]. BPA is a recursive algorithm that works similarly to the quicksort algorithm. It picks a pivot at random and then places the other features (items) before, after or in the same bucket as the pivot according to the value in the pair order matrix M. In particular, given a parameter $\beta \in [0, 0.5]$, we put a feature u before the feature-pivot p if $M(p,u) \leq 0.5 - \beta$, we put a feature u after the feature-pivot p if $M(p,u) \geq 0.5 + \beta$, and in the same bucket as p otherwise. Then, recursive calls are carried out for the lists *before* and *after*.

4. FSS based on the consensus bucket order

As mentioned in the introduction, perhaps the simplest filter-based feature selection method consists in the application of an evaluator $eval(f_i, C)$ which measures the discriminative power of feature f_i with respect to the class C in some way. Then, a *ranker* is used as search method, which simply sorts the features according to their $eval(\cdot, C)$ value. Finally, only the first k features in the rank are retained to induce the classifier. This method has the advantage of being computationally efficient, as only n filter evaluations are needed. However, it has two important drawbacks: (1) it does not consider interactions between the features; and (2) the value of k must be manually set, with the risk of being too small or too big.

A potential solution to the aforementioned problems is to use the *filter-wrapper* approach. In this approach, first a ranking is obtained, which is later refined by using wrapper evaluations. Thus, the value of k is less important and an enoughly large value is selected, since it would be reduced later in the wrapper phase. The wrapper process is guided by the filter-based ranking, which leads to a reduction in the number of wrapper evaluations in comparison to pure wrapper methods. In this work we focus on the simplest approach, which carries out k wrapper evaluations exactly:

```
1) Let $\sigma = f_1|f_2|\ldots|f_k$ be the ranking obtained for the best $k$ features according to the filter measure.
2) For $i = 1, \ldots, k$ do
       evaluate the subset $S_i = \{f_j \ : f_j \in \sigma \wedge j \leq i\}$ in a wrapper way.
3) Return the subset $S_i$ with the best evaluation score.
```

As have been pointed out in Section 2, dealing with the aggregation of several rankings has some advantages with respect to dealing with a single one, e.g. stability and robustness against the small variations introduced by the evaluation measures and the train/test partitions. In this paper we advocate for the use of *bucket orders* instead of strict preference orders. Some of the reasons have been detailed in the previous section. Now we discuss on the computational efficiency expected gain, which consists in saving a considerably number of wrapper evaluations.

If we use a bucket order to guide the wrapper evaluation process, all the features in the same bucket are equally preferred, so there is no need to deal with them one by one. Thus, instead of adding a single variable to the selected subset at each iteration, we add in a single step all the variables placed in the next bucket. For example, if the obtained ranking (bucket order) is $1|2,3|4,5,6|7$, then only 4 subsets instead of 7 will be evaluated in the wrapper phase: $\{1\}$, $\{1,2,3\}$, $\{1,2,3,4,5,6\}$, and $\{1,2,3,4,5,6,7\}$. As wrapper evaluations mean to train and validate a classifier, even many times depending on the validation carried out, the process is computationally time demanding, and so reducing the number of wrapper evaluations represents a great saving in CPU time.

The proposed method is shown in Table 1. Now we briefly explain it:

- Lines 1-5. Following the idea used in ensemble theory [30], we draw b samples from $\mathbf{D}$ and compute a ranking of size k for each one. As the k features ranked in each σ_i can be different, the number of features selected in at least one ranking, $k^* = |\cup_{i=1}^{b} \sigma_i|$, is larger than k (usually, in practice $k^* \gg k$).
- Line 6. M, the square matrix of size $k^* \times k^*$, is constructed from $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_b\}$. $M(u, v)$ is the proportion of rankings where u precedes v given the number of rankings in Σ where both features appears.
- Line 7. BPA is stochastic because of the random selection of the pivot. We observed that the initial pivot plays a major role in the final bucket order. Thus, by BPA$_{it}$ we refer to iterate BPA it times by using different seeds, so that the best one is returned. Two remarks must be done here: (1) BPA is very fast $O(k^* log(k^*))$; and (2) the goodness of the bucket order returned by each iteration of BPA is scored by using the matrix C and equation 1. Thus, this step is independent of the size of the dataset.
- Line 8. The bucket order is truncated to its first k' buckets. k' is selected such that $|\cup_{i=1}^{k'-1} B_i| < k$ and $|\cup_{i=1}^{k'} B_i| \geq k$.
- Lines 9-13. This is the wrapper phase guided by the bucket order. k' wrapper evaluations are carried out, each one using as input features the union of those included in the first i buckets, $i = 1, \ldots, k'$. We use accuracy as goodness score and each subset is assessed by using a cross validation process (5cv in our experiments) in order to prevent overfitting.
  A particular situation arises in this phase. Usually $|\cup_{i=1}^{k'} B_i| > k$, and therefore including all the features in the last bucket means using more than k features. For a fair comparison with the approaches using rankings without ties, we only add to $S_{k'}$ the first $k - |\cup_{i=1}^{k'-1} B_i|$ features of bucket $B_{k'}$. Notice that the position inside of a bucket is arbitrary and depends on the random seed used by the particular iteration of BPA.
- Lines 14-15. Finally the subset having the best accuracy is returned.

FSS-OBOP requires $b{\cdot}n$ filter evaluations and k' wrapper evaluations. In fact, in this case, each wrapper evaluation means to learn and test $n \cdot f$ classifiers, where $n \cdot f$ is the number of folds of the inner cross-validation used to fight overfitting. Therefore, wrapper evaluations are much more computationally expensive than filter ones. The reduction of wrapper evaluation calls is the goal of this paper.

TABLE 1. ALGORITHM FSS-OBOP

	Input
$\mathbf{D}$	a dataset of m instances over variables $\{f_1, \ldots, f_n, C\}$
$eval$	a univariate evaluator to score the features
k	the number of features to select in the filter phase
b	the number of bootstrap samples to create the rankings
β	the β value for BPA
it	the number of times BPA is iterated
$\mathcal{A}$	the classification algorithm be used in the wrapper phase
	Ouput
$\mathcal{S}$	the subset of selected features

```
 1  for $i = 1, \ldots, b$ do
 2      Obtain a sample $\mathbf{T_i}$ of size $m$ from $\mathbf{D}$ by using sampling
          with replacement
 3      Create a ranking $\pi_i$ of the features by using $\mathbf{T_i}$ and $eval$
 4      $\sigma_i \leftarrow$ Truncate($\pi_i, k$)
 5  endfor
 6  $M \leftarrow ConstructMatrix(\{\sigma_1, \sigma_2, \ldots, \sigma_b\})$      // $M_{k^* \times k^*}$
 7  $\mathcal{B}_0 \leftarrow$ BPA$_{it}(M, \beta)$
 8  $\mathcal{B}_0^k \leftarrow$ Truncate($\mathcal{B}_0, k$)          // $\mathcal{B}_0^k = B_1|\ldots|B_{k'},\ k' \leq k$
 9  for $i = 1, \ldots, k'$ do
10      $S_i \leftarrow \bigcup_{j=1}^{i} B_j$
11      $\mathbf{D}_i \leftarrow \mathbf{D}^{\downarrow S_i \cup \{C\}}$
12      $acc_i \leftarrow$ CrossValidate($\mathcal{A}, \mathbf{D}_i$)
13  endfor
14  $s = \arg\max\{acc_1, \ldots, acc_{k'}\}$
15  return $\mathcal{S} = S_s$
```

5. Experimental results and discussion

The goal of this paper is to study if the use of the consensus bucket order represents a significant reduction in the number of wrapper evaluations without decreasing the accuracy of the obtained model. Therefore, our experiments are designed in order to test these two issues. As benchmark, we use a set of 12 high-dimension datasets usually considered in recent FSS literature [11], [16], [19].

In Table 2 we show a brief description of these datasets: number of rows (instances) m, number of variables n and number of labels for the class variable c.

For the comparison, we use the following algorithms:

- *Ranker (R)*. In this case the ranking (strict preference order) directly computed by using the evaluator is used to guide the wrapper search. This method requires n filter evaluations and k wrapper evaluations
- *FSS-OBOP($\beta = 0$)*. When $\beta = 0$ BPA returns a strict preference order, that is, a bucket order in which every bucket contains a single feature. This approach, which uses a strict preference order as consensus, has been the one used in the literature to deal with consensus-based FSS. It requires $b \cdot n$ filter evaluations and k wrapper evaluations

TABLE 2. DATABASES USED IN THE EXPERIMENTS

DB	m	n	c
madelon	501	2000	2
colon	2001	62	2
dlbcl	4027	47	2
lymphoma	4027	96	9
gisette	5001	6000	2
leukemia	7130	72	2
arcene	10001	100	2
lung	12534	181	2
prostate	12601	136	2
gcm	16064	190	14
dexter	20001	300	2
psoriasis	54676	180	3

- *FSS-OBOP($\beta > 0$)*. The approach proposed in this paper. In this case, a bucket order is used to guide the wrapper search. The value of β controls the number of bucket in the bucket-order, larger values of beta produce a small number of buckets. We have tried $\beta = 0.25$ (as it is the value recommended in [28]), $\beta = 0.2$ and $\beta = 0.15$.

We perform 8 experiments over each dataset by changing some of the parameters of the FSS-OBOP algorithm:

- Two evaluators are used, one based on information theory and the other one on the classification error:
  - Symmetrical Uncertainty (SU) [31] is a sort of normalized mutual information between the class C and a given feature f_i. It is given by the expression
$$SU(f_i, C) = \frac{2(H(C) - H(C|f_i))}{H(C) + H(f_i)},$$
where $H(\cdot)$ stands for Shannon entropy. In our case, if f_i is numeric, then it is discretized by using entropy-based multi-interval Fayad and Irani discretization [32]. The discretization is only used to compute $SU(\cdot)$. In the wrapper phase the original numeric values are used.
  - OneR (1R) [33] scores each variable by applying the 1R classification algorithm. This algorithm generates a set of rules which only uses a single variable as predictor, then the classification error using these rules is used to score the given feature.
- Two different classifiers are used.
  - Naive Bayes (NB) [34] is a computationally very efficient probabilistic classifier which is known to be affected by irrelevant and redundant features.
  - C4.5 [35] is a state-of-the-art decision tree-based classifier. It carries out its own embedded feature selection process, but it is largely benefited from a previous FSS process, in particular when it has to deal with many numerical variables.
- The number of features that are selected in the filter step (k). We use $k = 50$ and $k = 100$.
- We always generate $b = 50$ samples and then 50 rankings are used to obtain the consensus.
- BPA is always iterated 100 times to select the best bucket order (according to the input matrix C).

5.1. Results

To study the performance of the proposed algorithm with these parameterizations, we run a 10 folds cross-validation for each parameterization and dataset. The averaged accuracy for the test folds when using SU as evaluator are reported in Table 3, while the results when using 1R as evaluator are reported in Table 4. The best result(s) for each experiment is(are) in bold.

Following the recommendations in the literature [36], we carry out a statistical analysis based on the application of Friedman test ($\alpha = 0.05$) and a post-hoc Holm test ($\alpha = 0.05$) for the cases in which Friedman has detected that at least one algorithm is significantly different to the others. We compare the algorithms globally, considering all the results for each algorithm (8 experiments × 12 datasets). We also carry out particular comparisons by taking into account only the results for SU, 1R, NB, C4.5, $k = 50$ or $k = 100$. The p-values and rankings obtained by the Friedman tests are shown in Table 5, with the averaged position between brackets. If the Friedman test detects that at least one algorithm is different, then we highlight the p-value by using bold face. In these (3) cases, the post-hoc Holm test is carried out by using as control the algorithm in the first position of the ranking. We highlight those algorithms which are significantly worse than the control according to Holm test in bold face.

5.2. Discussion

From the statistical analysis we can extract the following conclusions:

- The use of bucket orders to guide the search is never worse than using a strict preference order-based consensus or the ranking directly obtained by ranker+evaluator.
- More precisely, the use of the bucket orders obtained by using $\beta = 0.2$ or $\beta = 0.25$ is significantly better in 3 out of the 7 cases studied. Between this two algorithms (betas), even significant difference is never observed, we should select $\beta = 0.2$ as it is placed in the first position of the ranking obtained by Friedman test in 6 out of the 7 cases, and it is always in a better position than $\beta = 0.25$.
- Our experiments do not report any supremacy regarding accuracy of using the consensus ranking without ties instead of the ranking directly obtained

TABLE 3. ACCURACY RESULTS WHEN USING SYMMETRICAL UNCERTAINTY AS EVALUATOR

		β					β					β					β			
Dataset	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25
arcene	**0,79**	0,72	0,75	0,75	0,76	**0,8**	0,76	0,76	0,77	0,79	**0,79**	0,75	0,74	0,76	0,74	0,77	**0,81**	0,8	**0,81**	0,8
colon	**0,84**	0,8	0,79	0,79	0,79	0,8	0,79	0,73	0,76	0,76	**0,84**	0,8	0,75	0,82	0,79	**0,82**	0,78	**0,82**	0,79	0,79
dexter	0,86	0,86	0,87	**0,87**	0,87	**0,87**	0,87	0,87	0,82	0,83	0,9	0,91	0,91	0,91	0,9	0,83	0,84	**0,87**	0,86	0,83
dlbcl	**0,93**	**0,93**	0,91	0,91	0,89	0,7	0,7	0,71	**0,74**	0,71	**0,93**	0,89	**0,93**	**0,93**	0,91	0,7	0,68	0,68	0,71	**0,78**
gcm	0,52	0,56	**0,58**	0,55	0,57	0,43	0,39	0,44	0,38	0,45	0,53	0,57	**0,58**	0,54	0,57	0,43	0,43	0,43	**0,48**	0,46
gisette	0,89	0,89	**0,89**	0,89	0,88	0,92	0,92	0,92	**0,92**	0,92	0,89	**0,89**	0,89	0,89	0,89	0,93	0,93	0,93	0,94	0,93
leukemia	0,91	0,94	0,92	**0,97**	0,96	0,86	**0,88**	**0,88**	0,86	0,86	0,91	0,94	0,94	**0,97**	0,96	0,86	**0,89**	0,88	0,86	0,86
lung	0,98	0,98	**0,99**	**0,99**	**0,99**	0,93	0,91	0,93	0,93	**0,95**	0,98	0,98	0,99	0,99	**1**	0,93	0,91	0,94	0,94	**0,95**
lymphoma	0,79	0,8	0,79	**0,8**	0,78	**0,8**	0,76	0,77	0,77	0,77	0,78	0,78	0,78	0,79	0,8	0,8	0,77	0,79	0,78	**0,81**
madelon	0,61	0,61	0,6	0,6	0,6	0,75	0,76	0,76	0,76	**0,77**	**0,61**	0,6	0,6	0,59	0,59	0,75	0,76	0,76	0,77	**0,77**
prostate	**0,7**	0,65	0,64	0,66	0,61	0,85	0,87	0,84	0,88	0,87	**0,7**	0,65	0,63	0,64	0,62	0,85	0,88	0,88	0,87	0,86
psoriasis	0,69	0,71	0,72	0,69	**0,77**	0,67	0,67	0,64	**0,73**	0,71	0,71	0,72	0,73	0,72	**0,77**	0,68	0,7	0,66	**0,71**	**0,71**
	$k = 50$, $\mathcal{A} =$ Naive Bayes					$k = 50$, $\mathcal{A} =$ C4.5					$k = 100$, $\mathcal{A} =$ Naive Bayes					$k = 100$, $\mathcal{A} =$ C4.5				

TABLE 4. ACCURACY RESULTS WHEN USING ONER AS EVALUATOR

		β					β					β					β			
Dataset	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25
arcene	**0,7**	0,65	0,64	0,65	0,62	0,68	**0,75**	0,66	0,73	0,65	0,7	0,68	0,67	**0,73**	0,7	0,67	0,69	0,65	**0,75**	0,72
colon	**0,84**	0,8	0,82	0,82	0,75	0,78	0,82	**0,85**	0,82	0,71	**0,84**	0,8	0,75	0,79	0,74	0,78	0,82	0,82	0,85	0,84
dexter	**0,85**	0,81	0,83	**0,85**	0,84	0,84	0,84	0,86	0,85	0,85	0,89	0,89	**0,9**	0,89	0,89	0,84	**0,85**	**0,85**	0,84	0,83
dlbcl	0,91	0,89	**0,93**	**0,93**	0,91	0,78	0,76	0,74	0,78	**0,82**	0,93	0,89	0,91	0,91	0,91	0,78	0,74	0,7	0,83	**0,86**
gcm	0,54	0,57	0,56	0,52	0,52	**0,48**	0,43	0,47	0,46	0,45	0,56	0,56	0,55	0,53	0,55	0,45	0,41	0,44	**0,46**	0,46
gisette	**0,87**	0,87	0,87	0,87	0,87	0,91	0,91	0,91	**0,91**	0,91	**0,88**	0,88	0,88	0,88	0,88	0,93	0,93	0,93	0,93	**0,93**
leukemia	0,94	0,94	0,97	**0,97**	0,97	**0,88**	0,86	0,86	0,86	0,86	0,94	0,96	**0,97**	0,94	0,94	**0,88**	0,86	0,86	0,86	0,86
lung	**0,99**	0,98	0,99	0,99	0,98	**0,96**	0,93	0,94	0,95	0,95	0,99	0,98	0,99	0,99	**1**	**0,96**	0,94	0,93	0,95	0,95
lymphoma	0,75	0,78	**0,82**	0,75	0,77	0,6	0,75	0,69	**0,78**	0,73	0,77	0,77	0,75	**0,81**	0,79	0,64	0,74	0,69	0,73	0,75
madelon	**0,62**	0,6	0,6	0,61	0,6	0,69	0,75	0,75	0,75	0,74	**0,62**	0,6	0,6	0,6	0,61	0,71	0,74	0,74	0,75	0,74
prostate	0,67	**0,71**	0,62	0,66	0,63	0,85	**0,91**	0,87	0,87	0,86	0,67	**0,7**	0,6	0,62	0,65	0,85	0,89	0,83	0,85	0,85
psoriasis	0,78	0,78	**0,81**	0,79	**0,81**	0,67	**0,71**	0,71	0,68	0,69	0,79	0,78	0,78	**0,81**	**0,81**	0,69	0,67	0,71	**0,74**	0,73
	$k = 50$, $\mathcal{A} =$ Naive Bayes					$k = 50$, $\mathcal{A} =$ C4.5					$k = 100$, $\mathcal{A} =$ Naive Bayes					$k = 100$, $\mathcal{A} =$ C4.5				

TABLE 5. RESULTS FOR THE STATISTICAL ANALYSIS ON ACCURACY

All **(p=0,033)**	SU (p=0,114)	1R (p=0,143)	C4.5 **(p=0,000)**	NB (p=0,343)	$k = 50$ (p=0,526)	$k = 100$ **(p=0,021)**
$\beta = 0,20$ (2,60)	$\beta = 0,20$ (2,65)	$\beta = 0,20$ (2,53)	$\beta = 0,20$ (2,34)	R (2,67)	$\beta = 0,20$ (2,69)	$\beta = 0,20$ (2,5)
$\beta = 0,25$ (2,92)	$\beta = 0,25$ (2,77)	R (2,92)	$\beta = 0,25$ (2,59)	$\beta = 0,20$ (2,85)	R (2,97)	$\beta = 0,25$ (2,68)
R (3,11)	$\beta = 0,15$ (2,97)	$\beta = 0,25$ (3,07)	$\beta = \mathbf{0,15}$ **(3,21)**	$\beta = 0,15$ (3,06)	$\beta = 0,15$ (2,99)	**R (3,25)**
$\beta = \mathbf{0,15}$ **(3,14)**	R (3,29)	$\beta = 0$ (3,17)	$\beta = \mathbf{0}$ **(3,31)**	$\beta = 0$ (3,18)	$\beta = 0,25$ (3,15)	$\beta = \mathbf{0,15}$ **(3,28)**
$\beta = \mathbf{0}$ **(3,25)**	$\beta = 0$ (3,32)	$\beta = 0,15$ (3,31)	**R (3,55)**	$\beta = 0,25$ (3,24)	$\beta = 0$ (3,20)	$\beta = \mathbf{0}$ **(3,29)**

by the evaluator. Things may be different if other filter-wrapper strategies were used or if other parameters were studied, e.g. size of the selected subset or *stability*, as reported in the literature.

Once we have concluded that using FSS-OBOP ($\beta = 0.2$) is the preferred algorithm with respect to accuracy, we pay attention to the number of wrapper evaluations carried out by the different algorithms (Tables 6 and 7). Now the difference is clear. The algorithms using the ranking produced by the evaluator+ranker or using the consensus represented by the ranking without ties need exactly 50 or 100 wrapper evaluations for $k = 50$ and $k = 100$ respectively. These numbers reduce to 5.04 ($k = 50$) and 6.46 ($k = 100$), which represents the 10.08% and the 6.46% with respect to the rankings non allowing ties. As wrapper evaluations are much more costly in terms of CPU time than filter ones, this fact clearly compensates the number of extra filter evaluations carried out with respect to the direct use of the evaluator-based ranking.

Finally, due to the lack of space, we only briefly comment on the number of selected features. In this case, we

TABLE 6. NUMBER OF WRAPPER EVALUATIONS WHEN USING SYMMETRICAL UNCERTAINTY AS EVALUATOR

		β					β					β					β			
Dataset	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25
arcene	50	50	4,1	2,9	2,1	50	50	4,1	2,9	2,1	100	100	4,8	4,1	3,1	100	100	4,8	4,1	3,1
colon	50	50	4,6	4	3,7	50	50	4,6	4	3,7	100	100	5,4	4,4	4	100	100	5,4	4,4	4
dexter	50	50	10,5	8,7	7,1	50	50	10,5	8,7	7,1	100	100	12,5	10	8,5	100	100	12,5	10	8,5
dlbcl	50	50	5,1	4,3	3,4	50	50	5,1	4,3	3,4	100	100	6,2	5,1	3,9	100	100	6,2	5,1	3,9
gcm	50	50	2,9	2,5	2	50	50	2,9	2,5	2	100	100	3,8	3,1	2,2	100	100	3,8	3,1	2,2
gisette	50	50	15,2	12,2	10,1	50	50	15,2	12,2	10,1	100	100	23,5	18,5	14	100	100	23,5	18,5	14
leukemia	50	50	7,1	4,8	4,5	50	50	7,1	4,8	4,5	100	100	8,4	6,4	5,3	100	100	8,4	6,4	5,3
lung	50	50	6	4,5	3,5	50	50	6	4,5	3,5	100	100	8,1	6,2	4,6	100	100	8,1	6,2	4,6
lymphoma	50	50	3,6	2,9	3,2	50	50	3,6	2,9	3,2	100	100	5	3,6	3,2	100	100	5	3,6	3,2
madelon	50	50	8,8	6,7	5,3	50	50	8,8	6,7	5,3	100	100	9,2	7,2	5,5	100	100	9,2	7,2	5,5
prostate	50	50	4,8	4,1	3,5	50	50	4,8	4,1	3,5	100	100	6,5	4,7	3,8	100	100	6,5	4,7	3,8
psoriasis	50	50	2,6	2,3	2,1	50	50	2,6	2,3	2,1	100	100	4,1	2,6	2,5	100	100	4,1	2,6	2,5
	$k = 50$, $\mathcal{A} =$ Naive Bayes					$k = 50$, $\mathcal{A} =$ C4.5					$k = 100$, $\mathcal{A} =$ Naive Bayes					$k = 100$, $\mathcal{A} =$ C4.5				

TABLE 7. NUMBER OF WRAPPER EVALUATIONS WHEN USING ONER AS EVALUATOR

		β					β					β					β			
Dataset	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25	R	0	0.15	0.20	0.25
arcene	50	50	3	2,5	1,9	50	50	3	2,5	1,9	100	100	3,9	3,1	2	100	100	3,9	3,1	2
colon	50	50	3,8	3	2,8	50	50	3,8	3	2,8	100	100	4,6	3,8	2,9	100	100	4,6	3,8	2,9
dexter	50	50	9,8	7,4	5,8	50	50	9,8	7,4	5,8	100	100	10,9	8,5	7,3	100	100	10,9	8,5	7,3
dlbcl	50	50	4,5	3,2	2,8 1	50	50	4,5	3,2	2,8	100	100	5,9	4,6	3,9	100	100	5,9	4,6	3,9
gcm	50	50	3	2,4	1,6	50	50	3	2,4	1,6	100	100	3	2,6	1,9	100	100	3	2,6	1,9
gisette	50	50	19,3	17,1	15,1	50	50	19,3	17,1	15,1	100	100	31,1	26,4	21,3	100	100	31,1	26,4	21,3
leukemia	50	50	5,7	4,4	3,7	50	50	5,7	4,4	3,7	100	100	6,8	5,4	4,3	100	100	6,8	5,4	4,3
lung	50	50	5,7	4,4	3,4	50	50	5,7	4,4	3,4	100	100	8,3	6,1	4,6	100	100	8,3	6,1	4,6
lymphoma	50	50	2,5	2,9	2,4	50	50	2,5	2,9	2,4	100	100	4,1	3,2	3	100	100	4,1	3,2	3
madelon	50	50	5,7	4,6	3,5	50	50	5,7	4,6	3,5	100	100	6,4	5,1	4,2	100	100	6,4	5,1	4,2
prostate	50	50	4,7	3,7	2,7	50	50	4,7	3,7	2,7	100	100	5,2	3,9	3,3	100	100	5,2	3,9	3,3
psoriasis	50	50	3,4	3,1	2,1	50	50	3,4	3,1	2,1	100	100	5	3,7	2,8	100	100	5	3,7	2,8
	$k = 50$, $\mathcal{A} =$ Naive Bayes					$k = 50$, $\mathcal{A} =$ C4.5					$k = 100$, $\mathcal{A} =$ Naive Bayes					$k = 100$, $\mathcal{A} =$ C4.5				

observe that the use of bucket orders to guide the search selects a larger number of features. This fact was expected, as when using bucket orders variables are added in blocks, not individually. In particular, when using the ranking produced by the evaluator+ranker, 18.57 (33.48) features are selected when $k = 50$ ($k = 100$). These numbers are similar to the ones when using a strict preference order as consensus: 19.68 (35.04). Finally, these numbers are bigger when using the bucket order-based consensus to guide the search: 27.72 (48.56). Therefore, we can conclude that the number of selected features by FSS-OBOP ($\beta = 0.2$) is about 1.5 times the number of features selected by the methods that use rankings without ties to guide the search. However, these number of features is extremely small if compared to the high dimension of the used datasets (12380 features on average).

6. Conclusions

In this paper we show the advantage of allowing ties in the filter ranking used to guide the search. This advantage is mainly due to the reduction on the number of wrapper evaluations which has a direct effect in CPU time. Furthermore, the algorithm FSS-OBOP ($\beta = 0.2$) becomes the right choice from the point of view of accuracy. On the other hand, this method selects more features, by a factor of 1.5. Anyway, the number of selected variables is too small (27.72 when $k = 50$ and 48.56 when $k = 100$) if we take into account that we have dealt with high-dimensional datasets, having between 500 and 54676 features.

Although this is a first approach in using rankings *with ties* to guide the wrapper phase, many possibilities arise for the future: trying more sophisticated filter-wrapper strategies than a linear search; using different evaluators to produce the set of rankings to be aggregated; studying other measures different to accuracy (e.g. stability and/or robustness); and, what we consider the most interesting one, allowing ties also in the input rankings.

Acknowledgments

This work was partially financed by the *Ministerio de Economía y Competitividad, Junta de Comunidades de Castilla-La Mancha, Universidad de Castilla-La Mancha* and *FEDER funds* by means of the projects *TIN2013-46638-C3-3-P*, *PEII-2014-049* and *CCI-2014ES16RFOP010*.

References

[1] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[2] I. Kavakiotis, A. Triantafyllidis, G. Tsoumakas, and I. Vlahavas, "Ensemble feature selection using rank aggregation methods for population genomic data," in *9th Hellenic Conference on Artificial Intelligence*, 2016, p. 22.

[3] R. C. Prati, "Combining feature ranking algorithms through rank aggregation," in *International Joint Conference on Neural Networks*, 2012, pp. 1–8.

[4] P. Bermejo, J. A. Gámez, and J. M. Puerta, "Incremental wrapper-based subset selection with replacement: An advantageous alternative to sequential forward selection," in *IEEE Symposium on Computational Intelligence and Data Mining*, 2009, pp. 367–374.

[5] M. Gütlein, E. Frank, M. A. Hall, and A. Karwath, "Large-scale attribute selection using wrappers," in *IEEE Symposium on Computational Intelligence and Data Mining*, 2009, pp. 332–339.

[6] G. Brown, A. Pocock, M. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *Journal of Machine Learning Research*, vol. 13, pp. 27–66, 2012.

[7] R. P. DeConde, S. Hawley, S. Falcon, N. Clegg, B. Knudsen, and R. Etzioni, "Combining results of microarray experiments: a rank aggregation approach," *Statistical Applications in Genetics and Molecular Biology*, vol. 5, no. 1, 2006.

[8] I. Slavkov, B. Zenko, and S. Dzeroski, "Evaluation method for feature rankings and their aggregations for biomarker discovery." in *3rd International Workshop on Machine Learning in System Biology*, 2010, pp. 122–135.

[9] R. Wald, T. M. Khoshgoftaar, and D. Dittman, "Mean aggregation versus robust rank aggregation for ensemble gene selection," in *11th International Conference on Machine Learning and Applications*, vol. 1, 2012, pp. 63–69.

[10] Y. Zhang and F. J. Verbeek, "Comparison and integration of target prediction algorithms for microrna studies," *Journal Integrative Bioinformatics*, vol. 7, no. 3, p. 127, 2010.

[11] C. Sarkar, S. Cooley, and J. Srivastava, "Robust feature selection technique using rank aggregation," *Applied Artificial Intelligence*, vol. 28, no. 3, pp. 243–257, 2014.

[12] W. Bouaguel, A. B. Brahim, and L. Mohamed, "Feature selection by rank aggregation and genetic algorithms." in *International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, 2013, pp. 74–81.

[13] A. Onan and S. Korukoğlu, "A feature selection model based on genetic rank aggregation for text sentiment classification," *Journal of Information Science*, 2015.

[14] J. Dutkowski and A. Gambin, "On consensus biomarker selection," *BMC Bioinformatics*, vol. 8, no. 5, p. 1, 2007.

[15] Z. He and W. Yu, "Stable feature selection for biomarker discovery," *Computational Biology and Chemistry*, vol. 34, no. 4, pp. 215–225, 2010.

[16] Y. Saeys, T. Abeel, and Y. Van de Peer, "Robust Feature Selection Using Ensemble Feature Selection Techniques", in *European Conference on Machine Learning*, 2008, pp. 313–325.

[17] R. Wald, T. M. Khoshgoftaar, D. Dittman, W. Awada, and A. Napolitano, "An extensive comparison of feature ranking aggregation techniques in bioinformatics," in *IEEE 13th International Conference on Information Reuse and Integration*, 2012, pp. 377–384.

[18] A. Woznica, P. Nguyen, and A. Kalousis, "Model mining for robust feature selection," in *18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 913–921.

[19] F. Yang and K. Mao, "Robust feature selection for microarray data based on multicriterion fusion," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 8, no. 4, pp. 1080–1092, 2011.

[20] C. Sarkar, S. Cooley, and J. Srivastava, "Improved feature selection for hematopoietic cell transplantation outcome prediction using rank aggregation." in *International Workshop on Artificial Intelligence in Medical Applications*, 2012, pp. 221–226.

[21] W. Bouaguel, G. B. Mufti, and M. Limam, "Rank aggregation for filter feature selection in credit scoring," in *Mining Intelligence and Knowledge Exploration*, 2013, pp. 7–15.

[22] D. J. Dittman, T. M. Khoshgoftaar, R. Wald, and A. Napolitano, "Classification performance of rank aggregation techniques for ensemble gene selection." in *26th International Florida Artificial Intelligence Research Society Conference*, 2013, pp. 420–425.

[23] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in *10th International Conference on World Wide Web*, 2001, pp. 613–622.

[24] J. A. Aledo, J. A. Gámez, and D. Molina, "Using extension sets to aggregate partial rankings in a flexible setting," *Applied Mathematics and Computation*, vol. 290, pp. 208 – 223, 2016.

[25] J. Kemeny and J. Snell, *Mathematical Models in the Social Sciences*. Blaisdell-New York, 1962.

[26] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1-2, pp. 81–93, 1938.

[27] A. Ali and M. Meila, "Experiments with Kemeny ranking: What works when?" *Mathematical Social Sciences*, vol. 64, no. 1, pp. 28 – 40, 2012.

[28] A. Ukkonen, K. Puolamäki, A. Gionis, and H. Mannila, "A randomized approximation algorithm for computing bucket orders," *Information Processing Letters*, vol. 109, no. 7, pp. 356 – 359, 2009.

[29] A. Gionis, H. Mannila, K. Puolamäki, and A. Ukkonen, "Algorithms for discovering bucket orders from data," in *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006, pp. 561–566.

[30] D. W. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[31] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *20th International Conference on Machine Learning*, 2003, pp. 856–863.

[32] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuousvalued attributes for classification learning," in *13th International Joint Conference on Artificial Intelligence*, vol. 2, 1993, pp. 1022–1027.

[33] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine Learning*, vol. 11, no. 1, pp. 63–90, 1993.

[34] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *11th Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 338–345.

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

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.