Model Mining for Robust Feature Selection

Adam Woźnica
University of Geneva
Department of Computer Science
7, route de Drize
Battelle - building A
1227 Carouge, Switzerland
Adam.Woznica@unige.ch

Phong Nguyen
University of Geneva
Department of Computer Science
7, route de Drize
Battelle - building A
1227 Carouge, Switzerland
Phong.Nguyen@unige.ch

Alexandros Kalousis
University Of Applied Sciences
School of Management
Department of Business Informatics
7, route de Drize
Battelle - building F
1227 Carouge, Switzerland
Alexandros.Kalousis@hesge.ch

ABSTRACT

A common problem with most of the feature selection methods is that they often produce feature sets–models–that are not stable with respect to slight variations in the training data. Different authors tried to improve the feature selection stability using ensemble methods which aggregate different feature sets into a single model. However, the existing ensemble feature selection methods suffer from two main shortcomings: (i) the aggregation treats the features independently and does not account for their interactions, and (ii) a single feature set is returned, nevertheless, in various applications there might be more than one feature sets, potentially redundant, with similar information content. In this work we address these two limitations. We present a general framework in which we mine over different feature models produced from a given dataset in order to extract patterns over the models. We use these patterns to derive more complex feature model aggregation strategies that account for feature interactions, and identify core and distinct feature models. We conduct an extensive experimental evaluation of the proposed framework where we demonstrate its effectiveness over a number of high-dimensional problems from the fields of biology and text-mining.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology—*Feature evaluation and selection*

General Terms

Algorithms, Performance, Experimentation

Keywords

feature selection, stability, model mining, high-dimensional data, classification

KDD'12, August 12–16, 2012, Beijing, China.

1. INTRODUCTION

High-dimensional datasets are becoming more and more abundant in the field of data mining. A traditional approach of tackling the high-dimensional learning problems is based on the application of feature selection methods to select a set of features–feature models–as small as possible that accurately describe the learning examples.

A common problem with most of the feature selection methods is that they often produce feature models that are not stable (or robust) with respect to slight variations in the training set. This can be problematic when we want to identify a handful of features which are important for the given mining problem and present them to the domain experts together with quantifiable evidence of their robustness and stability. A typical application domain in which the stability and robustness of the selected features are of paramount importance is biology. The analysis of biological samples using high-throughput technologies produces learning instances of very high dimensionality, tens of thousands or even hundreds of thousands. Very often this high dimensionality is coupled with a limited number of samples. In such cases, the low stability of the selected features, often coupled with their weak discriminatory power, raises questions about the scientific significance of these discoveries [10, 9].

We can identify two sources of feature model instability. The first is a high level of feature redundancy (a situation that is very typical in biological problems); this can make the feature selection methods to produce unstable feature sets simply because they can select different features among the redundant features. In such cases very different feature sets can be equivalent predictors of some outcome because they describe different aspects of the same phenomenon. The second major cause of instability is the "underspecification" in the sample space with respect to the feature dimensionality, a problem often described as the High Dimensionality Small Sample Size (HDSSS) problem. In the HDSSS setting we do not have adequate statistics, and very often slight variations in the training data can produce radical changes in the feature models. For example, it was shown in [6] that for typical biological problems at least one order of magnitude more training instances are needed to reach an acceptable level of feature stability. However, the HDSSS problem is here to stay since sample availability is often limited due to unsurpassable practical constrains, while our ability to measure different variables is ever increasing.

A common approach to derive more robust feature sets is

to use ensemble feature selection over bootstrap samples [18, 1, 9]. More specifically, the feature selection algorithm we want to "stabilize" is applied on a number of random subsamples of the training data, and the different outputs are subsequently aggregated to produce a new, hopefully more stable, feature set. Almost all the existing algorithms focus on feature selection methods that produce feature rankings (or feature scores that can be trivially converted to rankings) and output an aggregated ranking, based on which the desired number of features can be finally selected. The existing aggregation strategies are most often based on the averaging of the individual rankings [18, 1], or the (soft) frequency of selection of individual attributes [15, 16, 9]. A common characteristic of all these methods is that they treat the features independently and do not consider feature interactions. Moreover, from the existing literature there is no consensus of whether the aggregated feature sets, in addition to the boosted stability, also bring an improvement in terms of predictive performance.

Although the existing approaches based on ensemble feature selection are conceptually simple and were shown to give rise to more stable feature sets they have two main limitations. First, as already mentioned, the aggregation operators are usually based on simple averaging techniques that consider attributes independently, and as such they often generate "artificial" structures that would not have been produced by the original feature selection method. For example average feature ranks produced over the models of a sparsity-imposed algorithm, such as LASSO, most likely do not correspond to any output of that algorithm [21]. In general, for feature models whose elements are generated independently from each other, such as the ones obtained by univariate feature selection methods, the standard techniques based on simple averaging are expected to be appropriate. In cases where the components of the feature selection models are not independent, a typical example of these are models produced by multivariate feature selection methods, the simple strategies might not be sufficient, and other techniques that account for feature interactions is needed.

The second limitation is that the existing aggregation techniques return a unique feature model. Nevertheless, in problems with high levels of redundancy it might very well be the case that there are more than one feature sets, redundant between them, with similar information content. In such situations the domain experts might prefer the identification of *a number of alternative feature models*, all of which describe different aspects of the same problem.

In this work we address these limitations and present a general framework in which we mine over feature models to extract feature patterns that we will use to derive more stable feature models and to identify distinct but equivalent, in terms of predictive behavior, feature models. We define aggregation operators over the feature models which we exploit to aggregate and summarize the different feature models. The model aggregation operators we formalize range from the simplest strategies that treat attributes independently, to the more complex ones that are structure-preserving and account for feature interactions. We perform an in-depth empirical study where we evaluate our framework on a number of biological and text mining datasets where we demonstrate the effectiveness of our approach.

The remainder of this paper is organized as follows. In Section 2 we present the feature model mining framework. In Section 3 we present the experimental setup that we will use evaluate our framework and in Section 4 we present the experimental results. In Section 5 we review the related work, and we summarize our work in Section 6.

2. FEATURE MODEL MINING

In this section we present a general framework for feature model mining and describe a number of mining methods that operate over feature models. Our goal is to improve the stability and robustness of the feature selection as well as to provide the means to identify alternative feature selection models of equal quality.

We will first introduce some notation. Let $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ be a labeled dataset, where $\boldsymbol{x}_i \in \mathbb{R}^p$ is the i-th instance and $y_i \in \mathbf{Y}$ its class label. Feature selection identifies the most important features so that instances can be represented in a lower dimensional space without a significant loss of discriminatory power; most often, the number of desired features m is controlled by the user. Following the ideas that appeared recently in ensemble feature selection [18, 9] we generate b *base feature models* from b bootstrap subsamples of D using a given feature selection method. These base models will become the target of the feature model mining that we will present soon. Feature selection models come roughly in the following three flavors:

- feature weightings: $\boldsymbol{w} = (w_1, \ldots, w_p)^T,\ w_l \in \mathbb{R}$,
- feature rankings: $\boldsymbol{r} = (r_1, \ldots, r_p)^T,\ r_l \in \mathbb{N}^+$ (we assume that the values of r_l can be non-unique),
- feature subsets: $\boldsymbol{s} = (s_1, \ldots, s_p)^T,\ s_l \in \{0, 1\}$, with 0 and 1 indicating absence and presence of a feature.

In this work, we focus on the $\boldsymbol{r}$ and $\boldsymbol{s}$ representations; the output of a feature selection method that produces feature weightings can be always converted to a ranking.[1] We denote by $\mathcal{R} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_b\}$ and $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_b\}$ the feature rankings and subsets produced from the b bootstrap subsamples of the given dataset by the application of a given feature selection method. We will present a number of unsupervised feature model mining operators f which operate over $\mathcal{R}$ or $\mathcal{S}$, in fact most of the methods that we will present will work with the $\mathcal{S}$ representation. The different operators will aggregate the base feature models, while taking into account the feature model structures and systematically identify alternative feature models. By feature model structures here we mean the specific combinations of features, reflecting feature interactions and dependencies, that different feature selection algorithms are able to uncover according to their underlying learning bias. The signature of the feature model mining operators will be:

$$f\ :\ \mathcal{F} \longrightarrow \{\boldsymbol{s}'_i\}_{i=1}^{k}$$

where $\mathcal{F}$ is either $\mathcal{R}$ or $\mathcal{S}$, and k is a user specified parameter that defines the number of alternative feature models. We classify the operators to two groups according to the aggregation strategy they follow. In the first group, *Single Model Aggregation*, we have strategies that aggregate the different base feature models into a single model, i.e. $k = 1$. In the second group, *Multiple Model Aggregation*, we have strategies that produce a number of distinct models.

[1] [2] argues against directly aggregating weights–scores $\boldsymbol{w_i}$, especially those that correspond to p-values of univariate statistical tests.

2.1 Single Model Aggregation Strategies

We will start with a description of the first group which is the one that contains most of the proposed strategies. We separate the strategies of this group to different categories with respect to how they construct their final model, in terms of the model components of the base feature models, and how close is the final solution they produce to those that the feature selection algorithm would have generated.

Univariate Strategies.

In this category we have just two strategies *avgRank* and *mostFreq*, that produce the final model by looking at the individual feature scores, obtained either from their ranks or from their frequencies. More precisely, in *avgRank* [18, 1, 2, 9] the $score_l$ of the lth feature is the average of its rankings, i.e. $score_l = \sum_{i=1}^{b} r_{il}$. In *mostFreq* [15, 16, 9] we consider the frequency of the features that appear in $\mathcal{S}$, i.e. $score_l = \sum_{i=1}^{b} s_{il}$. In both cases, the final set of m features is selected according to their $score_l$. These two strategies make no effort what so ever to capture feature model structures since they treat the features independently. If one used a univariate feature selection to generate the base feature sets this is no problem, since no such structure exists. However, if the base feature sets have been produced by a multivariate feature selection algorithm, then it is probable that the final feature set will not reflect particular feature combinations that have been found informative by the feature selection algorithm since these aggregation strategies do not consider longer structures.

Model Component Combination Strategies.

Here we consider strategies that search for frequent itemsets over the different base feature sets, and then establish the union of the most frequent itemsets, until the number of distinct features in the union reaches m, i.e. the desired feature cardinality. One can think this strategy as the combination of frequent "model fragments" in to a final single feature model.

To limit the number of returned frequent itemsets, which depending on the dataset and the feature selection method can render the method computationally infeasible, we only focus on the frequent *maximal* [3] and *closed* [20] itemsets, giving respectively rise to the *maxApr* and *closedApr* strategies. An itemset is closed if no superset has the same support, and maximal if no superset is frequent. We note that for a given support threshold we have FMI $\subseteq$ FCI $\subseteq$ FI, where FCI, FMI and FI are respectively the set of all the frequent closed, all the frequent maximal, and all the frequent itemsets. The elements in FCI that are not in FMI necessarily have non-lower support with respect to those of FMI, and hence have on average a shorter in length than those of FMI. Additionally, the elements in FCI are more redundant (as measured by the average number of items they share) than the elements of FMI. Both *maxApr* and *closedApr* can be seen as an extension of *mostFreq*; in the latter method the cardinality of the returned itemsets is trivially restricted to one and the threshold is specified not on the minimum support but on the number of returned itemsets.

Exact Structure Preservation Strategies.

In this category we have a number of strategies which select in a principled manner one of the base feature models, namely the most representative one. Clearly, the solutions fully reflect the learning bias of the feature selection algorithm that generate the base feature sets. The different approaches rely on different combinations of frequent itemset discovery and clustering techniques.

The simplest strategy of this category is *mostRep*. Here, the final feature model is simply the median model of the set of base models S. We use the median instead of the average because the former preserves the model structure since the median is actually an element of the underlying set. The underlying assumption of the *mostRep* strategy is that there is a meaningful median element over S. This is a valid assumption only if the feature models form a single unimodal cluster. However, in general we cannot assume that S has such unimodal structure. We thus introduce the *largeMed* strategy which uses the k-medoid clustering algorithm and the average silhouette criterion [12] to determine the best number of clusters and returns as the final feature model the medoid of the largest cluster. As before we rely in k-medoids instead of k-means because the averaging process does not necessarily preserve the feature model structures discovered by the feature selection algorithm. In this category we have two more strategies, *medMaxApr* and *medClosedApr*, which can be seen as the counterparts of *maxApr* and *closedApr* that preserve exactly the feature model structure of the feature selection algorithm. Both of them define IS_{top} as the set that contains the frequent feature sets with the top support. Then, they determine all base feature models that are supersets of at least one element of the IS_{top}, and constitute the S' set of base feature models. The final feature model is the median feature set of S'. In that sense these two strategies try to find the most representative base feature set using as initial seeds the most frequent feature itemsets.

Overall, what we have in the single model aggregation strategies is a spectrum of different approaches which are distinguished according to the degree to which they respect the structure of the original base feature models and the bias of the feature selection algorithm that produced them. At the lower end we have the Univariate Strategies (*US*) which completely ignore these using an univariate approach to select the features that will be included in the final feature model. In the middle of the spectrum we have the Model Component Combination Strategies (*MCCS*) which detect frequent model fragments–subsets of the base feature sets–that appear frequently within the base feature sets and produce the final model by bringing together these model fragments. Finally, we have at the higher end the Exact Structure Preservation Strategies (*ESPS*) that output as a final model one of the base feature sets of S; the methods of this group differ on how they select the prototypical feature set. The simplest method returns just the median of S, followed by the method that returns the median of the largest cluster, to the most complicated that return the median of the base feature sets that are supersets of the most frequent itemsets.

2.2 Multiple Model Aggregation Strategies

The aggregation strategies of this group take as input the base feature models and produce as output k different feature models. The simplest strategy of the group, *allMed*, applies the k-medoid clustering algorithm on set of the base feature models S and returns the medoid of each cluster. Clearly, the cardinality of the k feature models is dictated

by that of the base feature models, so if it is m for the latter, it will also be m for the former. In addition to the clustering-based approach we also have two of frequent pattern-based approaches. The first one, *allClosedApr*, produces as output k feature models that correspond to the k highest support frequent closed itemsets.[2] Unlike *allMed* here we do not have control over the cardinality of the aggregated feature sets. It might very well be the case that this cardinality will be quite low since we are selecting the top frequent itemsets; depending on the dataset and the value of k the most frequent itemsets can easily consist of only single items—features. In order to be able to better control the cardinality of the final feature sets we derive an alternative of this strategy which we name *medClosedApr*. Here instead of simply returning as feature sets directly the top k itemsets we first get all the base feature models from S that contain the itemset(s) with the highest support, and return the corresponding k medoids. Like that each aggregated feature model will now have as many features as the base feature models.

3. EXPERIMENTS

In this section we investigate the behavior and performance of the different model aggregation strategies presented in Section 2. We have two suites of experiments dealing with the single and multiple model aggregation strategies.

We start with the single model aggregation strategies and evaluate them, over a panel of datasets and feature selection methods, both in terms of the stability of the aggregate feature models that they produce as well as in terms of the classification error they result to when these models are passed to a number of classification algorithms. Concretely, given a feature selection algorithm we generate b base feature models which will then be aggregated into a single one by each one of the different aggregation strategies. On each one of the aggregated models we train a classifier using a given classification algorithm. One of the primary goals of these experiments is to see whether the model aggregation strategies that account for the base feature model structures can bring an improvement over the univariate aggregation strategies.

In the second suite of experiments we examine the behavior of the multiple model aggregation strategies. We evaluate them with respect to a number of dimensions, namely the diversity of the multiple feature models they produce, the average errors of classifiers trained on them using some given classification algorithm, as well as the prediction agreement of the produced classifiers. What we want to examine is whether using the multiple model aggregation strategies we can produce very diverse feature models, i.e. feature models that deliver different descriptions of the classification problem, which give rise to accurate classifiers, i.e. the feature models are discriminatory, and that are "semantically similar", i.e. deliver the same predictions when they are asked to classify the same instance.

We set the cardinality of the base and the aggregate feature models m to 20. We performed additional experiments with different values of this parameter; however, these results reveal similar trends. We set the number of bootstrap samples b to 150. We implemented feature aggregation and performance computation (Section 3.1) using the R language.

[2]We also defined and experimented with a similar strategy obtained using the concepts of frequent maximal itemsets; however, its behavior was similar to that of *allClosedApr*.

3.1 Performance Measures

Stability Estimation.

To estimate the stability of feature models produced by a feature selection algorithm, or by a single model aggregation strategy coupled with a given feature selection algorithm, we compare the feature models generated over a number of variations of the input dataset. In this study we opted for N-fold stratified cross-validation (CV) with N = 10, resulting in a set of N feature sets $\mathcal{S}' = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N\}$. The different feature models are cross-compared and the average similarity sim_{avg} is computed:

$$sim_{avg}(\mathcal{S}') = \frac{2\sum_{1\leq i<j\leq N} sim(\boldsymbol{s}_i, \boldsymbol{s}_j)}{N(N-1)} \quad (1)$$

where $sim(\cdot,\cdot)$ is a similarity measure between two feature models. In this work we follow [11] and define $sim(\boldsymbol{s}_i, \boldsymbol{s}_j) = \frac{|\boldsymbol{f}_i \cap \boldsymbol{f}_j|}{|\boldsymbol{f}_i \cup \boldsymbol{f}_j|}$, where $\boldsymbol{f}_i$ is a set of identifiers of selected features from $\boldsymbol{s}_i$ and $|\cdot|$ denotes the set cardinality. The sim measure takes values in $[0, 1]$, zero when there is no overlap and one when the two sets are identical, and allows comparisons also between feature sets of different cardinalities. We also note that the performance measure (1) will be also used in the second set of experiments to measure the diversity of the feature models generated by methods from Section 2.2; in this case $sim(\cdot,\cdot)$ will correspond to the average similarity between all the k feature models.

Error Estimation.

As already mentioned, to asses the predictive performance of a feature model aggregation method we estimate the error of a classification algorithm trained over the feature model produced by the given aggregation method. We estimate this performance measure using exactly the same fold separation as the one in the 10-fold CV used in the stability estimation.

Predictive Agreement Estimation.

To quantify the predictive agreement of two classification models c_1 and c_2 trained over the same input dataset but using different feature sets we measure the percentage of identical predictions over some test dataset $\mathcal{D}' = \{\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_l\}$. Concretely, we compute $agree(c_1, c_2, \mathcal{D}') = \frac{\sum_{1\leq i\leq l} \delta(c_1(\boldsymbol{x}'_i), c_2(\boldsymbol{x}'_i))}{l}$ where $c_{\{1|2\}}(\boldsymbol{x}'_i)$ is the classification label assigned to the instance $\boldsymbol{x}'_i$ by the $c_{\{1|2\}}$ model; and $\delta(a, b) = 1$ if $a = b$, and 0 otherwise. The overall agreement of the k models produced by a multiple model aggregation strategy is then given by:

$$agree(\mathcal{D}') = \frac{2\sum_{1\leq i<j\leq k} agree(c_i, c_j, \mathcal{D}')}{k(k-1)} \quad (2)$$

The final agreement estimation is the average of (2) over the 10-CV folds (again, we have exactly the same folds as in the error and stability estimation).

Statistical Significance and Methods Ranking.

We control the statistical differences of the errors of two methods using the McNemar's test, and the paired T-test for

the stability and prediction agreement; for all of them we set the significance level to 0.05. To acquire a better picture of the relative performances of the different methods we establish a ranking schema for each one of the performance measures based on the results of the pairwise comparisons. More precisely, if one method is significantly better than another one, it is credited with one point; if there is no significant difference then both are credited with 0.5 points; finally, if it is significantly worse it is credited with zero points. Clearly, the more points one method scores the higher its ranking will be. If we compare n different methods then the maximum number of points that one can obtain is $n-1$ if it is significantly better than all the other methods; if there is no significant difference then each will get $(n-1)/2$ points.

3.2 Datasets

We experiment with high-dimensional data from three application domains: proteomics, genomics and text mining. The proteomics datasets, *ovarian*, *prostate*, and *stroke*, are mass spectrometry datasets. The genomics datasets, *leukemia*, *nervous*, *colon*, are DNA-microarray datasets. The text mining datasets, *disease, alt*ernative, describe classifications of sentences to relevant or non-relevant to given topics; features are word frequencies. The references to these datasets are in [11]. In Table 1 we give a short description of them.

Table 1: Statistics on the datasets. n, p and c denote respectively the number of training instances, data dimensionality and the number of classes.

Datasets	n	p	c
prostate	322	390	2
ovarian	253	385	2
stroke	208	172	2
leukemia	72	7129	2
colon	62	2000	2
nervous	60	7129	2
alt	4157	2112	2
disease	3237	2376	2

3.3 Feature Selection and Classification Algorithms

To create the base feature models we will use the following feature selection methods: *Information Gain (IG)*, *Chi-Square (CHI)* [5], *Symmetrical Uncertainty (SYM)* [5], *ReliefF (RELIEF)* [17], *SVMRFE* [7], *SVMONE* and *Correlation Based Feature Selection (CFS)* [8]. The first three methods are *univariate* feature selection methods; the remaining are *multivariate* methods that are in principle able to detect and exploit feature interactions. *RELIEF* delivers a weighting of the features by computing distances among each of the training instances and their 10-nearest neighbors, and estimating the contribution of each feature in these distances. *SVMRFE* is based on repetitive applications of linear *SVM* where the P% lowest ranked features are eliminated at each iteration. The ranks of the features are based on the order in which they are eliminated and the weights assigned to them by the linear *SVM*. In our experiments we set P to 10% and the complexity parameter C of the linear *SVM* to 0.5. We also included a simple linear support vector machine (*SVMONE*) which is equivalent to *SVMRFE* with a single iteration. *CFS* evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low inter-correlation are preferred. We note that *CFS* is the only method in this study which automatically determines the appropriate feature cardinality and hence we do not control for the m parameter. We used the WEKA implementation of these algorithms.

Since, as we mentioned, feature selection and feature model aggregation methods do not deliver classification models we had to use classification algorithms in order to train from their results classification models that we can use to estimate the discriminatory power of the feature models that they produce. We have chosen the classification algorithms in such a manner that they represent distinct learning paradigms. We experiment with *Decision Tree (J48)*, *SVM* and *1NN* learning algorithms. We used the WEKA implementation of the algorithms; we set the parameters to default values, except for the C parameter of *SVM* which was set to 0.5.

4. RESULTS

We will first experiment with and study the performance of the different single model aggregation strategies, and then that of the multiple model aggregation strategies.

4.1 Single Model Aggregation Strategies

Each one of the single model aggregation operators was applied on the b base feature models produced by each one of the seven different feature selection algorithms over a given dataset. The resulting aggregated feature models were subsequently passed to the three classification algorithms in order to have an estimate of their discriminatory power. The results over the different classification algorithms do not differ considerably so we will present only those of the *SVM*. We fix a feature selection algorithm and we rank the different aggregation operators with respect to the predictive error of the *SVM* models produced over their respective aggregated feature models, and with respect to the stability of their aggregated feature models. Ranking is done as described previously according to the points accumulated in terms of the significant wins and losses. Since for a given feature selection method we are comparing eight aggregation operators, plus one baseline which is the plain application of the given feature selection method (*nobagg*), the maximum number of points that one operator can score for a given dataset, if it is significantly better than all the rest, is eight; if there is no difference then everybody will all get four points.

In Table 2 we give these error and stability ranks for each aggregation operator averaged over all the different datasets with which we experimented. In this table the upper value in each cell is the error-based rank of the aggregation operator (indicated by the row) for the given feature selection algorithm (indicated in the column header); the bottom value is the respective stability rank. Clearly, error takes precedence over stability; a very high stability rank is useless if it is accompanied with a very low error rank.

Examining the averaged ranks (the last column of Table 2) we see that the best method in terms of its average error rank is *closedApr* that scores 5.6 points, followed by *maxApr* with 5.0 points, and *mostFreq* with 4.9; the latter has exactly the same number of points—rank—as the baseline method,

Table 2: Average prediction error & stability scores of the *single model aggregation operators* over all datasets when *SVM* is used as the classification algorithm. In each cell the upper value is the average error score of the model mining operator given in the row, over the different model mining operators, for the feature selection algorithm specified in the column header; the lower value is the respective stability score. The final column gives the respective averages for each model mining operator over the different feature selection algorithms. The larger the values the higher the rank of the method; the top rank per column is indicated in bold.

		Univariate Feature Selection			Multivariate Feature Selection				
		IG	*CHI*	*SYM*	*RELIEF*	*SVMONE*	*SVMRFE*	*CFS*	Avg
	nobagg	5.9	6.0	4.4	4.8	4.6	5.0	4.0	4.9
		6.1	**6.4**	5.8	5.9	4.3	3.7	4.7	5.3
US	*avgRank*	**6.1**	5.0	5.4	3.4	5.5	4.0	1.5	4.4
		7.2	6.2	**6.7**	5.6	6.1	**6.8**	4.8	6.2
	mostFreq	4.9	5.4	5.1	4.3	4.5	4.6	5.5	4.9
		5.9	6.3	6.4	**6.3**	**6.8**	6.7	**6.9**	**6.5**
MCCS	*maxApr*	4.9	4.8	4.6	4.6	5.2	5.2	5.8	5.0
		4.8	4.9	4.6	5.1	6.2	5.6	6.1	5.3
	closedApr	5.1	**6.4**	**6.1**	4.9	5.1	**5.8**	**6.1**	**5.6**
		5.8	6.0	6.3	6.0	6.6	6.7	6.5	6.3
ESPS	*medMaxApr*	0.9	0.9	0.9	1.5	1.0	1.1	2.3	1.2
		2.1	2.4	2.0	2.3	1.9	2.3	2.2	2.2
	medClosedApr	1.8	1.8	1.7	2.1	1.5	1.8	3.2	2.0
		2.1	2.0	2.0	2.2	1.9	2.1	2.2	2.1
	largeMed	3.5	3.1	4.4	5.0	2.6	3.5	4.5	3.8
		0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.1
	mostRep	3.0	2.8	3.4	**5.4**	**5.9**	5.0	3.1	4.1
		1.9	1.8	2.2	2.6	2.1	2.1	2.4	2.2

nobagg. Essentially, the two strategies that manage to improve the predictive performance over the baseline are the two strategies that combine model components. When it comes to the average stability rank, the top ranked aggregation operator is the *mostFreq* with 6.5 points, closely followed by *closedApr* with 6.3 points, and *avgRank* with 6.2 points; all three methods are better than the baseline in terms of the stability which scores 5.3 points. The four remaining aggregation operators, *medMaxApr, medClosedApr, largeMed* and *mostRep*, do not seem to bring any improvement neither with respect to error nor to stability compared to the baseline method. Note that all of them are exact structure preservation operators, i.e. they return as the final feature model one of the b base feature models.

The difference in the performance, both in terms of error and stability, of *closedApr* and *maxApr* is puzzling since the two methods follow the same principle to produce the final feature set, i.e. the combination of frequent feature model components. Their only difference is that the first makes use of frequent closed itemsets, while the latter relies on frequent maximal itemsets. In order to try to understand this difference, we took a look at the number of top itemsets that each of the two methods needs to combine, in order to reach the desired number of m selected features, as well as the average size and support of these itemsets. In Table 3 we give the results for the *alt* dataset; however, the patterns that we will right away describe are the same over the different datasets. First, observe that the average itemset cardinality of the two methods is very similar: around 12 and 13 for the univariate feature selection methods, six to seven for the multivariate feature selection methods, and four to five for the *CFS* feature selection method. This difference between the univariate and multivariate methods is logical given the fact that univariate feature selection methods do not model for feature interactions and select a number individual (likely redundant) features with a similar (and potentially high) support from the b samples of the data; consequently, the combinations of these individual features will also have a high support. On the other hand, the multivariate methods exploit feature interactions and hence the selected subsets of features will have a lower support and will be shorter in length. The difference in support of frequent itemsets between these two classes of feature selection methods can be observed in Table 3. The lowest average itemset cardinality for *CFS* could be explained by the fact in this algorithm we do not control for resulting feature cardinalities.

When we now look at the average number of itemsets that need to be merged in order to arrive at the desired number of features we see that *maxApr* gets there with much less itemsets, on average 20% of that *closedApr* needs. Thus, the itemsets produced by *maxApr* are much more diverse in terms of the features they contain compared to those of *closedApr*, which, as already mentioned in Section 2.1, is expected given that the itemsets produced by *closedApr* include those produced by *maxApr*. In the former method the itemsets are variants around a core set of features which explains why it needs so many more itemsets to get to the desired number of features. This also gives us a hint on why there is a difference in performance between *closedApr* and *maxApr*: the former method aggregates shorter and more general core sets of features with a higher support, while *maxApr* aggregates longer sets of features that are placed "lower" in the apriori lattice, but which are nevertheless overly specific and do not "generalize" well. The last column in Table 3 demonstrates the lower average support of itemsets produced by *maxApr* in comparison with *closedApr*.

Overall, the two univariate aggregation operators achieve an important improvement over the baseline in terms of the stability of the final feature model they produce; however, their predictive performance is the same, *mostFreq*, or even worse, *avgRank*, than the baseline. The operator that achieves the best predictive performance is *closedApr*;

Table 3: Averages of: number of frequent itemsets required to attain m distinct features, itemsets size, and support, for the *alt* dataset.

FS	# ItemSets	Size	Support
maxApr			
IG	12.50±3.10	13.91±0.35	0.38±0.08
CHI	12.30±3.02	14.18±0.35	0.41±0.04
SYM	16.80±4.47	13.78±0.38	0.35±0.05
RELIEF	32.20±8.61	7.49±0.57	0.37±0.03
SVMONE	68.90±32.06	7.26±0.39	0.21±0.06
SVMRFE	44.40±9.62	6.22±0.24	0.18±0.01
CFS	31.10±6.62	4.73±0.40	0.17±0.01
closedApr			
IG	60.90±19.62	12.92±0.35	0.52±0.07
CHI	64.00±26.60	13.18±0.32	0.56±0.04
SYM	159.30±58.51	12.22±0.42	0.51±0.05
RELIEF	118.40±31.16	6.66±0.57	0.46±0.03
SVMONE	220.70±107.08	6.63±0.34	0.30±0.06
SVMRFE	106.20±41.25	5.82±0.23	0.26±0.01
CFS	164.20±31.34	3.91±0.27	0.28±0.01

its score is better than the baseline, and in addition it has the second best stability score very close to the best. So, overall it comes as the best strategy for feature model aggregation. Remember here that *closedApr* works by combining frequent model fragments into a single model, while the two univariate approaches examine each feature on its own. While the univariate approach can improve the stability, it fails to improve the predictive performance. For the latter capturing larger model structures seems to be more important. The performance of the operators that preserve exactly the feature model structure by picking one of the original base models is quite disappointing. All of them are worse than the baseline method. Among them the ones that combine frequent itemsets and then find the medoids, *medMaxApr* and *medClosedApr*, have the worse performance. The two clustering medoid variants, *largeMed* and *mostRep*, fair a bit better, namely in terms of error, however their stability is equally low. The low stability of the exact structure preserving variants can be explained by the fact that actually they do not perform any kind of aggregation, but try to select among the base models the most representative one, however, by doing so the chances that there will be a larger overlap over the different final models are reduced.

We will now drill further down to the results given in Table 2 in order to get an idea of how the different model aggregation strategies fair with respect to the two distinct feature selection paradigms that we have here, namely univariate and multivariate. While the univariate aggregation strategy *avgRank* seems to have the strongest advantage in the case of univariate feature selection algorithm IG, this advantage is not persistent over the other two univariate feature selection algorithms, CHI and SYM. On the same time the *mostFreq* operator, while similar in spirit to *avgRank*, does not perform particularly well with the univariate feature selection algorithms; for IG and CHI it has a performance that is worse than the baseline with respect to both error and stability. The *closedApr* performance with CHI and SYM is quite good both in terms of error as well as stability. When we examine the four multivariate feature selection algorithms we see that *mostFreq* has an excellent stability performance with all of them, it is ranked top in three of the four. However, in terms of predictive performance it is quite poor, for three out of the four feature selection algorithms it is actually worse than the baseline. The *closedApr* has also a stability performance that is similar to that of *mostFreq*, however, in addition it also has a very good error performance, being better than the baseline method in all four multivariate feature selection algorithms, and the best in two of them.

Overall, while univariate feature aggregation strategies can improve the feature selection stability they fail to deliver similar gains in terms of the predictive performance, compared to the baseline method with no aggregation. Exploiting the feature model structure information, as it is done by the model component combination strategies, can improve not only the stability but also the predictive performance. However, what is important in the latter category is how these model components are discovered, as it is evident by the strong advantage of *closedApr* compared to *maxApr*.

4.2 Multiple Model Aggregation Strategies

We now turn to the experimentation with the multiple model aggregation strategies. We will examine the k multiple models that each strategy computes with respect to the accuracy they achieve, the agreement of their predictions, and their model similarity, i.e. the average number of features they have in common. We want to describe the degree to which we can have different feature models of good predictive performance, that produce similar predictions, and have a relatively small feature overlap. The eventual goal is to provide the domain experts with a more global picture of the mechanism that underlines the training data, than what they would have obtained with a single feature set.

As in the first suite of experiments we evaluate each model aggregation strategy with each one of the seven feature selection algorithms. Subsequently, on each one of the feature models that a given aggregation strategy will output for a given feature selection algorithm, we train a classifier using a given classification algorithm. We report the prediction agreement of the resulting classifiers, computed by (2), their average accuracy, and the average feature model similarity, computed by (1). We estimate these quantities using 10-fold CV. We present the results for all three classification algorithms that we used. For all the model aggregation strategies we set the number of different models to $k = 10$. We performed additional experiments with different values of this parameter; however, these results reveal similar trends. For each model aggregation strategy and classification algorithm we highlight the feature selection algorithm that achieves the top rank according to the statistical significance tests over the different performance measures.

The results for the *leukemia* dataset are given in Table 4 (for other datasets similar trends hold). First, we notice that the multiple models generated by *allClosedApr* are of lower predictive performance, as measured by the average classification error, in comparison with the models obtained by the clustering-based *allMed* strategy; the classification error for *allClosedApr* is on average three times higher than for *allMed*. As already noted in Section 2, this is a result of the fact that unlike *allMed*, in *allClosedApr* we do not control for the cardinality of the aggregated feature sets. In the examined dataset this cardinality is low which naturally leads to low discriminatory power of the feature models; in fact, most of the top frequent itemsets are of cardinality just

Table 4: Average and standard deviation of error and prediction agreement of classifiers trained on the $k = 10$ multiple models produced on the *leukemia* dataset by each one of the *multiple model aggregation strategies* over the different feature selection algorithms. Additionally, average feature model similarity over the k models of each model aggregation strategy and feature selection algorithm. We highlight the top ranked results for each strategy and classifier, for each performance measure.

FS	*J48*		*SVM*		*1NN*		
	agreement	error	agreement	error	agreement	error	similarity
			allMed				
IG	**0.97±0.04**	**0.14±0.08**	**0.98±0.03**	0.06±0.07	**0.93±0.06**	0.08±0.06	0.31±0.03
CHI	**0.97±0.05**	**0.14±0.08**	**0.98±0.02**	0.07±0.08	**0.94±0.05**	0.08±0.06	0.33±0.05
SYM	**0.98±0.04**	**0.15±0.09**	**0.98±0.02**	0.06±0.07	**0.94±0.05**	0.07±0.06	0.45±0.03
RELIEF	0.95±0.04	**0.13±0.09**	**0.97±0.05**	0.06±0.07	**0.94±0.06**	0.08±0.07	0.34±0.04
SVMONE	0.88±0.09	**0.12±0.09**	**0.97±0.03**	0.06±0.07	**0.95±0.05**	0.08±0.06	0.28±0.04
SVMRFE	0.92±0.07	**0.12±0.06**	**0.98±0.02**	**0.03±0.05**	**0.95±0.04**	**0.05±0.05**	0.21±0.03
CFS	0.91±0.03	**0.13±0.08**	0.93±0.04	0.07±0.05	**0.91±0.08**	0.07±0.06	**0.14±0.01**
			allClosedApr				
IG	0.87±0.06	**0.12±0.07**	0.87±0.05	**0.17±0.07**	0.89±0.05	**0.11±0.05**	0.20±0.05
CHI	0.87±0.06	**0.11±0.07**	0.87±0.05	0.18±0.07	0.89±0.05	**0.10±0.05**	0.19±0.06
SYM	**0.92±0.06**	**0.12±0.06**	**0.89±0.06**	0.16±0.09	**0.92±0.07**	**0.10±0.06**	0.29±0.16
RELIEF	0.84±0.10	**0.13±0.08**	0.84±0.08	**0.16±0.06**	0.84±0.09	**0.13±0.08**	0.23±0.07
SVMONE	0.79±0.10	**0.16±0.09**	0.81±0.06	0.20±0.06	0.78±0.11	0.16±0.08	0.20±0.05
SVMRFE	0.81±0.11	**0.13±0.08**	0.82±0.05	0.22±0.05	0.81±0.10	**0.12±0.07**	**0.16±0.05**
CFS	0.84±0.06	**0.12±0.06**	0.86±0.05	0.21±0.06	0.84±0.08	**0.11±0.06**	**0.15±0.03**
			medClosedApr				
IG	**0.97±0.04**	0.15±0.08	**0.98±0.03**	0.06±0.07	0.94±0.06	0.08±0.06	0.32±0.04
CHI	**0.97±0.05**	0.14±0.08	**0.98±0.03**	0.06±0.07	**0.95±0.05**	0.07±0.06	0.33±0.03
SYM	**0.98±0.04**	0.15±0.09	**0.98±0.02**	0.06±0.07	0.94±0.05	0.07±0.06	0.45±0.03
RELIEF	0.95±0.04	0.14±0.09	**0.97±0.05**	0.06±0.07	0.95±0.06	0.08±0.07	0.34±0.04
SVMONE	0.89±0.08	0.13±0.10	**0.97±0.03**	0.06±0.07	0.96±0.04	0.07±0.07	0.28±0.04
SVMRFE	0.92±0.06	**0.11±0.07**	**0.98±0.02**	**0.03±0.05**	0.96±0.04	**0.05±0.05**	0.21±0.03
CFS	0.91±0.04	0.13±0.08	0.93±0.04	0.07±0.05	0.91±0.08	0.07±0.07	**0.14±0.01**

one. The low cardinalities of the resulting multiple models in *allClosedApr* have also a direct effect of the prediction agreement and the average model similarity, which are consistently lower than it is the case for *allMed*. The second observation is that the performance of *medClosedApr*, as measured by the three evaluation metrics, is very similar to that of *allMed*. This a consequence of the fact that a number ($k = 10$) of (frequent closed) itemsets with highest support is contained in a large fraction of base feature models, and hence the input the k-medoids algorithm in both *allMed* and *medClosedApr* is similar, resulting in similar medoids generated by these strategies. Finally, we note the good performance of *SVM* coupled with *allMed* with respect to both agreement and error, and an acceptable level of model similarity. In particular coupling *allMed*, *SVM* and *SVMRFE*, it is possible to generate quite diverse feature signatures (average model similarity of 0.21) which nevertheless give rise to powerful (classification error of 0.03) and "semantically similar" (prediction agreement of 0.98) models. The classification error of *SVM* coupled with *allMed* is not statistically different compared to the corresponding baselines in which we only do standard feature selection and classification (these results are not reported in Table 4).

5. RELATED WORK

The traditional approach to stabilize feature selection algorithms in learning problems with redundant features focuses on selecting relevant and non-redundant feature subsets in a pre-processing step; examples include CFS and Markov blankets. [19] addressed the redundancy problem by grouping correlated features together and treating these groups as the entities over which feature selection will take place; this work was refined in [14]. A related idea was proposed in [21] where the authors identified the problem of instability of LASSO for problems with high feature correlations, and proposed a regularization technique based on mixing different norms ("elastic nets"). All these approaches return a unique set of features. However in many applications as argued in this paper it is more beneficial to be able to identify alternative but equivalent solution sets.

The main difference between the methods from [19, 14, 21] and our framework is that the former consider, explicitly or implicitly, groups of redundant features, which are included or excluded from the model simultaneously, whereas we aim at explicitly providing domain experts with groups of features that account for different aspects of a problem at hand. Moreover, the approaches from [19, 14, 21] consider specific notions of feature redundancy (e.g. the linear correlation in [19]), whereas our framework is more flexible as the notion of redundancy varies and depends on the feature selection algorithms (it is the linear correlation only for *CFS*). Finally, the previous works select the individual subsets within a specialized feature selection algorithm (e.g. regression regularized with the mixed norms [21]), while we identify the different solutions a-posteriori, based on sets of features which can be generated by using virtually any feature selection method.

As already mentioned, in the context of ensemble feature selection, the most popular methods are univariate aggregation strategies. Two more complex exceptions were presented in [4] and [13], both aggregating elements of $\mathcal{R}$. The former method is inspired by the well known PageRank algorithm and can be seen as a direct extension of *mostFreq* where all the *pairs* of top ranked features are considered, and the appearances of the different pairwise feature relationships (feature a ranked before feature b, or vice versa) are counted. These counts give rise to the corresponding

frequencies, based on which the final aggregate ranking of attributes is generated. Our aggregation strategies based on frequent itemsets are considerably more general as they can account for higher-order feature interactions (see Table 3). The method from [13] is similar to *mostRep* as it looks for a feature model whose *weighted distance* to all the input models is minimal; the returned ranking does not necessarily appear in the set of input transactions. The authors consider two different distance measures over rankings and weight the elements by various performance measures. In this study we have shown that the single model aggregation strategies based on clustering do not fare well in comparison with the other proposed strategies. We also note that the methods from [4, 13] give rise to an aggregate ranking only for the most discriminating features.

6. CONCLUSION

In this paper we presented a general framework in which we mine over different feature models produced from a given dataset in order to extract patterns over the models. We use these patterns to derive more complex feature model aggregation strategies that account for feature interactions, and identify core and distinct feature models.

We empirically examined our framework on a number of high-dimensional datasets. The empirical evidence suggests that our framework is effective in comparison with the existing aggregation techniques. We demonstrated that the existing univariate aggregation techniques, although appropriate in many cases, are not the best solutions overall. While univariate feature aggregation strategies can improve the feature selection stability they fail to deliver similar gains in terms of the predictive performance, compared to the baseline method with no aggregation. Exploiting the feature model structure information, as it is done by the model component combination strategies, can improve not only the stability but also the predictive performance. What is however important is how these model components are discovered, as it is evident by the strong advantage of *closedApr* compared to *maxApr*. We also observed the poor performance of the operators that preserve exactly the feature model structure by picking one of the original base models. Overall, we recommend the *closedApr* aggregation strategy that provides a good compromise between the stability and predictive performances. When it now comes to the multiple models aggregation strategies we demonstrated that it is possible to construct distinct feature models of good predictive performance, that produce similar predictions, and have a relatively small feature overlap. Such diverse and yet equivalent feature models describe different aspects of the same problem, and hence provide domain experts with a more global picture of the mechanism under study. We recommend the *allMed* technique that is based on the k-medoids clustering algorithm and hence easy to implement.

7. ACKNOWLEDGMENTS

This work was partially funded by the European Commission through EU projects DebugIT (FP7-217139) and e-LICO (FP7-231519). The support from the COST Action BM072 ("Urine and Kidney Proteomics") is also gratefully acknowledged.

8. REFERENCES

[1] T. Abeel et al. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. *Bioinformatics*, 26:392–398, 2010.

[2] A.-L. Boulesteix and M. Slawski. Stability and aggregation of ranked gene lists. *Briefings in Bioinformatics*, 10:556–568, 2009.

[3] D. Burdick, M. Calimlim, and J. Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *ICDE*, 2001.

[4] R. P. DeConde et al. Combining results of microarray experiments: A rank aggregation approach. *Stat. Appl. Genet. Molec. Biol.*, 5(1), 2006.

[5] R. Duda, P. Hart, and D. Stork. *Pattern Classification and Scene Analysis*. John Willey and Sons, 2001.

[6] L. Ein-Dor et al. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *PNAS*, 103(15):5923–8, 2006.

[7] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46:389–422, March 2002.

[8] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, 1998.

[9] A.-C. Haury, P. Gestraud, and J.-P. Vert. The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. 2011.

[10] J. P. Ioannidis. Microarrays and molecular research: noise discovery? *Lancet*, 365(9458):454–5, 2005.

[11] A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowl. Inf. Syst.*, 12:95–116, May 2007.

[12] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 1990.

[13] S. Lin and J. Ding. Integration of ranked lists via cross entropy monte carlo with applications to mrna and microrna studies. *Biometrics*, 65(1):9–18, 2009.

[14] S. Loscalzo, L. Yu, and C. Ding. Consensus group stable feature selection. In *KDD*, 2009.

[15] M. S. Pepe et al. Selecting differentially expressed genes from microarray experiments. *Biometrics*, 59(1):133–142, 2003.

[16] X. Qiu, Y. Xiao, A. Gordon, and A. Yakovlev. Assessing stability of gene selection in microarray data analysis. *BMC Bioinformatics*, 7(1), 2006.

[17] M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Mach. Learn.*, 53:23–69, October 2003.

[18] Y. Saeys, T. Abeel, and Y. Van de Peer. Robust feature selection using ensemble feature selection techniques. In *PKDD*, 2008.

[19] L. Yu, C. Ding, and S. Loscalzo. Stable feature selection via dense feature groups. In *KDD*, 2008.

[20] M. J. Zaki. Generating non-redundant association rules. In *KDD*, 2000.

[21] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J R Stat Soc Series B*, 67(2):301–320, 2005.