Deliverable D14.1

Feature Selection

Luigi Portinale and Lorenza Saitta
Dipartimento di Informatica, Università del Piemonte Orientale
Spalto Marengo 33, Alessandria 15100, Italy

April 2, 2002

1 Feature Selection: State of the Art

The problem of *feature selection* is fundamental in a number of different tasks like classification, data mining, image processing, conceptual learning, etc... In recent times, the growing inportance of knowledge discovery and data-mining approaches in practical applications has made the feature selection problem a quite hot topic, especially when considering the mining of knowledge from real-world databases or warehouses, containing not only a huge amount of records, but also a significant number of features not always relevant for the task at hand.

Looking at the literature, there are essentially two main fields where the feature selection problem has been extensively studied:

- Statistical Pattern Recognition
- Machine Learning

In the first field, feature selection is considered from the classification point of view, i.e. the problem is approached having the construction of an efficient classifier (i.e. a pattern recognizer) as a target.

In the Machine Learning community, more emphasis is given to the more general problem of concept learning, even if classification still remain an important issue. However, this difference has provided different approaches aimed at solving the problem, addressing different perspectives and points of view.

In the following, we will report on the main proposals and approches developed inside the two communities.

1.1 Feature Selection in Statistical Pattern Recognition

Statistical Pattern Recognition is a sub-field of Pattern Recognition aimed at recognizing (classifying, describing or grouping) patterns inside available data, by using specific statistical techniques [2]. Even if they are around from almost 50 years, pattern recognition approaches have recently gained a new popularity, due to emerging applications which are not only challanging, but also computationally expensive and very demanding like data mining (identifying a pattern or a correlation among data or an outlier in millions of multidimensional patterns), document classification (searching text documents), forecasting, multimedia organization and retrieval in databases, flexible information retrieval (product retrieval in e-commerce applications, solution retrieval in help-desk support), etc...

The statistical approach to pattern recognition represents a pattern as a set of d features or attributes, by viewing it as a d-dimensional feature vector. Classical concepts from statistical decision theory [13] are then used to establish decision boundaries among pattern classes. The recognition system operates in two different modes: *training* or *learning* mode and *testing* or *classification* mode.

In the training mode, a feature extraction/selection module finds the suitable features for representing the input patterns and the classifier is trained to partition the feature space. A feedback path allows the designer to optimize both the preprocessing (required to segment the initial pattern from the background, to remove noise, etc...) and the feature extraction/selection strategies.

Two important phenomena that can be identified in statistical pattern recognition are the so called *curse of dimensionality* and the *peaking phenomenon*. The performance of a classifier depends on the interrelationship between sample size, number of features and classifier complexity. If one consider a very simple naive table-lookup technique consisting in partitioning the feature space into cells and associating a class label to each cell, it can be pointed out that this technique requires a number of training data points which is exponential in the feature space dimension [6]. This phenomen is termed the curse of dimensionality which produces as a consequence the peaking phenomenon in classifier design. This is a paradoxical effect that appears by considering the following; it is well-known that the probability of misclassification of a decision rule does not increase as the number of considered features increases, as long as the class-conditional densities are known (or alternatively the number of training samples is arbitrarly large and representative of the underlying densities). However, it has been often noticed in practice that increasing the features to be considered by a classifier may degrade its performance, if the number of traning examples that are used to design the classifier is small relative to the number of features. This paradoxical behavior is termed the *peaking phenomen* [23, 46, 45]. The explanation stands in the following: the most commonly used parametric classifiers estimate the unknown parameters and plug them in for the true ones in the class conditional densities. For a fixed sample size, as the number of features increases (and consequently the number of unknown parameters to be estimated from the sample), the reliability of parameter estimation decreases. As a consequence, the performance of the resulting classifier, for a fixed sample size, may degrade with an increase in the number of features.

The problem of dimensionality reduction is then important and must be addressed in the right way. There are two main resons to keep the dimensionality of the pattern representation (and so the number of features) as small as possible: the cost of the measurements and classification accuracy. Both the pattern representation and the classifier built on that are simplified in case a limited yet salient feature set is taken. Moreover, the paeking phenomenon can be alleviated in case just a small number of training examples is avalilable. Of course, it must be considered that, on the other hand, a reduction in the number of features may lead to a loss of the discriminatory power of the classifier and consequently lower the accuracy of the resulting system.

Concerning these aspects, it is important to distinguish between *feature extraction* and *feature selection* that are two steps that may be employed in order to address the above issue. Even if in the literature the two terms are often used interchangeably, in [2] it is suggested to use the first term for feature construction techniques. By adopting this view, the term *feature selection* refers to algorithms that identify and select tha hopefully best subset of the input feature set with respect the target task (e.g. classification accuracy). Methods that create (extract) new features based on transformation or combinations of the original features in the set are called *feature extraction algorithms*. Often, feature extraction preceds feature selection; first features are extracted from the sensed data and then, some of the extracted features with low discriminatory power are discarded, leading to the selection of the remaining features.

Notice that the two techniques are also complementary in their goals; feature selection leads to savings in measurement cost (some features are discarded and then there is no need to obtain them) and the selected features retain their original physical interpretation (that may be important in some cases for understanding the physical process that generates the pattern). On the other hand, the transformed features obtained by feature extraction techniques may provide a better discriminatory ability than the best selected subset, but these features fail in retaining the original physical interpretation and may not have a clear meaning.

The main issue in dimensionality reduction is the choice of a criterion function. A commonly used criterion is the classification error of a feature subset. However, it must be taken into account that this information by itself, cannot be reliably estimated when the sample size is small relative to the number of features. Furthermore, we cannot escape from the problem of the intrisic dimensionality of data; this refers to the fact that it is not always possible to adequately describe d-dimensional patterns in a subspace of dimensionality less than d.

Let us now describe some of the most used feature selection methods in statistical pattern recognition. As mentioned before, the problem can be summarized as follows: given a set of d features, select a subset of size m that leads to the smallest classification error. Let Y be the given set of features with cardinality d and let m represents the cardinality of the desired subset $X, X \subset Y$. Consider the feature selection criterion be identified by an evaluation function $J(X)$: higher is the value of $J(X)$, better is the set X. Following the above considerations, if P_ϵ is the classification error, a natural choice is $J = 1 - P_\epsilon$. The use of P_ϵ makes the criterion dependent on the specific classifier that is used and on the sizes of both training and test sets.

The most straightforward approach to the feature selection problem can be described as follows:

1. examine all possible subset of size m of the original feature set;

2. select the subset with the largest value of $J(\cdot)$.

In particular the first step would require examining $(\overset{d}{m})$ possible candidate subsets and consequently this exhaustive search becomes unfeasible and impractical even for moderate values of m and d.

Cover and Van Campenhous [15] have shown that no nonexhaustive sequential feature selection procedure can be guaranteed in general to produce the otimal subset. The only optimal (in terms of a class of monotonic criterion functions) feature selection method which avoids the exhaustive search is based on the branch and bound technology [40]. This method can avoid exhaustive search by using intermediate results for obtaining bounds on the final criterion value. It only works, however, with monotonic criterion functions. Most commonly used criterion functions do not satisfy the monotonicity property.

Since feature selection is typically dome off-line, it has quite often been argued that the execution time of the selection algorithm is not so critical; however, in data mining applications it is not rare to have thousands of features involved and the computational requirement of a feature selection algorithm becomes extremely important. Optimal but exhaustive strategies like breadth-first search cannot be considered in such applications [32]. Since this problem is not typical to data mining applications (see for instance document classification), a number of suboptimal feature selection techniques have been developed and proposed in the literature; they essentially tradeoff the optimality of the selected subset for computational efficiency [24]. They are essentially based on the simple method of selecting just the best individual features: however, the suboptimality of these strategies is due to the fact that the best pair of features need not contain the best single feature [14]. The approach might still be useful as a first step to select some individually good features in decreasing very large feature sets. Further selection has to be done by more advanced methods that take feature dependence into account. There are essentially two different modes of selection: *forward selection* when the approach operates by evaluating growing set of features and *backward selection* when it operates by evaluating shrinking set of features. The name (forward with respect to backward) arises from the fact that the search space can be considered as a graph with the nodes corresponding to feature subset; the starting state is the empty subset, so adding features corresponds to moving forward on the graph and deleting feature corresponds in moving backward starting from the whole feature set.

The most simple methods are the *Sequential Forward Selection (SFS)* and the *Sequential Backward Selection (SBS)*. SFS (respectively SBS) adds (respectively deletes) one feature at a time which in combination with the selected features maximizes the criterion function. In SFS, once a feature is retained, it cannot be discarded, while in SBS once a feature is deleted it cannot be brought back into the optimal subset. SFS is computationally attractive, since to select a subset of size 2 it examines only $(d-1)$ possible subsets.

More sophisticated techniques are the *Plus l-take away r* and the *Sequential Floating Search* [42] that may operate either forward (SFFS) or backward (SBFS). These methods backtrack as long as they find improvement compared to previous feature sets of the same size. The first kind of selection ("Plus l-take away r") first enlarge the current feature subset by l features using forward selection, then it deletes r features using backward selection. It avoids the problem of feature subset nesting encountered in SFS and SBS methods, but needs to select values of l and r. The floating search approach is essentially a generalization of this approach where the values of the parameters l and r are determined automatically and updated dynamically. It provides close to optimal solutions, but the number of feature evaluations may easily increase by a factor of 2 to 10.

Concerning the evaluation criterion, most feature selection methods used, as already mentioned, the classification error of a feature subset to evaluate its effectiveness. This could be done, for example, by a k-NN classifier using the leave-one-out cross validation method of error estimation. However, it is important to keep in mind that different classifiers and different methods to estimate the error rate could lead to a different feature subset being selected. In [24], several algorithms have been compared in terms of classification error and run time. The general conclusion is that the SFFS method performs almost as well as the branch and bound algorithm and demands lower computational resources. Still superior performance has been shown for an adaptive version of SFFS [43].

Other attempted approaches have tried to combine these kind of algorithms with neural network classifiers, where the node-pruning method simultaneosly determines both the optimal feature subset and the optimal network classifier [18, 10]. First the network is trained and the least salient node is removed. This procedure of training and node pruning is iterated until the desired trade-off between classification error and size of the network is achieved. The pruning of an input node in the network is equivalent to removing the corresponding feature.

1.2 Feature Selection in Machine Learning

The problem of feature selection has been deeply investigated also in Machine Learning, by exploiting new ideas as well as ideas presented and approached in the field of statistical pattern recognition. The emphasis of the machine learning approach is on conceptual learning; this task can be subdivided into two main subtasks: deciding which attributes or features to use for describing the concept; deciding how to combine such attributes to get the right concept induction. Because of that, the problem of feature selection is central to machine learning and to application of the field like data mining and knowledge discovery [34, 44].

As for statistical pattern recognition, the dimensionality reduction is essential for both complexity and accuracy issues; current machine learning application need algorithms able to scale-up to real-world problems and attaining high accuracy. As already noticed in the previous section, accuracy is in general not monotonic with respect to the addition of features, and even when the learning algorithm is robust with respect to irrelevant features (as for example the Naive-Bayes approach [38]), it is usually significantly affected by feature correlations. In addition, with the reduction of the number of features, it is more likely that the final learned concept il less complex and more understandable by humans.

As reported in [1], the objective of feature subset selection in machine learning is to reduce the number of features used to characterize a dataset so as to improve a learning algorithm's preformance on a given task. Feature selection in machine learning has shown its impressive performance gains in attacking large dimensionality with many irrelevant features [19, 41, 17], as well as in enhancing comprensibility of the learned result [52]. As already noticed, the problem can be exposed as a search problem, so that heuristic search techniques can be devised in order to face it. Each state in the search space is a subset of the original feature set and a partial ordering can be states, with each child in the ordering DAG, having exacttly one more feature than its parent. In [8] it is argued that the structure of this space suggests that any feature selection method must take into consideration four basic issues that determine the nature of the search process:

- a starting point in the search space;
- an organization of the search;
- an evaluation strategy of the selected subset;
- a stopping criterion for halting the search.

These issue are essentially the main issues of any heuristic search problem in Artificial Intelligence, but considering them directly, we can classify in a detailed way machine learning approaches to feature subset selection.

The starting point determines the directions of the search and provide the already discussed distinction between forward selection (starting with no feature and adding new features) and backward selection (starting with the whole feature sets and shrinking it until the desired subset is reached).

The organization of the search determines the strategy in the space of size 2^d if d is again the number of features available. We have already discussed that the only non exhaustive optimal strategy that is possible is the branch and bound algorithm [40], but optimality is guaranteed only if the evaluation function is monotonic. When monotonicity cannot be satsfied heuristic search is the only solution. Despite classic deterministic heuristic algorithms proposed in pattern recognition like SFS, SBS and floating methods, also best-first search has been attempted [28]. Results from [53] suggest that classic greedy hill-climbing approaches tend to get trapped on local peaks caused by interdependencies among features. As we have already noticed, this triggered the definition of more sophisticated strategies like the floating ones [42].

On the other side, non-deterministic approaches have also been investigated, in motivation to avoid getting stuck in local maximum. Randomness is used to escape from such local maxima and this implies that one should not expect the same solution from different runs. Different techniques has been adopted to define this kind of algorithms namely Genetic Algorithms (GA) [53, 55, 31], evolutionary computation [20], Simulated Annealing [25], Las Vegas Algorithms [51].

The evaluation function measures the effectiveness of a given selected feature subset and the objective of the search is its maximization. Depending on how a measure carries out this objective we can determine a distinction among the approaches that has become very important in the feature selection literature [28, 16]. Approaches can be classified as:

- wrapper approaches;
- filter approaches
- embedded approaches

This distinction arises because, in order to evaluate the selected subset both the characteristics of the data, the target concept and the learning algorithm can be taken into account.

As reported in [19], when the goal is the maximization of the accuracy of a given feature subset, the features selected should depend not only on the relevance of the data with respect to the target concept, but also on the learning algorithm. This defines the so-called *wrapper approach*. Kohavi and John [28] report domains in which a feature, altough being in the target concept to be learned, does not appear in the optimal feature subset that maximizes the predictive accuracy for the specific algorithm used. This occurs since feature relevance and accuracy optimality are not always coupled in feature selection. The wrapper approach implies that the selection algorithm searchs for a good subset of features using the induction algorithm itself as a part of the evaluation function, the same algorithm that will be used to induced the final target concept. Once the induction algorithm is fixed, the idea is to train it with the feature subset encountered by the search algorithm, estimate the error rate and assigning it as the value of the evaluation function of the feature subset. In this way, representational biases of the induction algorithm used to construct the final concept are included in the selection process.

Wrapper approaches usually need a high computational cost, but technical computer advances in the last years have made reasonable the wrapper strategy in several applications, calculating an amount of accuracy estimations (traning and testing on significant amount of data) not envisioned in the 1980s. Before applying the wrapper approach, an enumeration of the available resources is quite critical; two main factors can make the selection problem large [33]: the number of features and the number of instances. One must bear in mind that in the wrapper approach, every possible solution visited by the search engine requires the time needed by the learning algorithm in the training phase. Many approaches have been porposed in order to alleviate the excessive loading of the training phase, avoiding the evaluation of many subsets taking advantage of the intrinsic properties of the used learning algorithm [12] or reducing the burden of the cross-validation technique for model selection [39]. Recently, a wrapper approach over Naive-Bayes and ID3 learning algorithms has been proposed in conjuction with evolutionary computation, where the evolution is guaranteed by the factorization of the probaility distribution of the best solutions obtained by means of a Bayesian network [20]. Promising results have been achieved with this approach in a variety of tasks when domain knowledge is not available.

When the learning algorithm is not used in the evaluation function, the goodness of a feature subset can be assessed regarding only to the intrinsic properties of the data. This type of feature selection approach, which ignores the induction algorithm to assess the merits of a feature subset is known as the *filter approach*. The name is due to the fact that feature selection is done before applying the learning algorithm, so a “filter” is applied to the algorithm to select relevant features, looking just at the data and taking into account the target concept to be learned. The learning algorithm constucts the concept using the set of selected features, ignoring the others. As we have previously discussed, the statistical pattern recognition literature has proposed a number of measures for evaluating the goodness of a candidate feature subsets [5], trying to detect the feature subsets with higher discriminatory information with respect to the concept [27] considering the probability distribution of the data. These measures are usually monotonic and increase with the addition of features that afterwards can hurt the final accuracy.

Mainly inspired by these statistical measures, in the 90s more complex filer measures which do not use (as in the wrapper approach) the final induction algorithm, generated new selection algorithms such as Focus [3], Relief [26] or its extension ReliefF [30, 54], Cardie's algorithm [11], Koller and Sahami's work on probabilistic concepts [29], the *incremental feature selection method* [32], Bell and Wang's approach [4] or the mathematical programming characterization of [35].

Nowadays, the filter approach is receiving considerable attention from the data mining community to deal with huge databases when the wrapper approach is unfeasible [32]. However, when the size of the problem allows the application of the wrapper approach, works in the 90s have shown the superiority of the approach, in terms of predictive accuracy over the filter one.

Finally, another type of feature subset selection has been identified in [8]: the *embedded approach*. In this case the feature selection process is done inside the induction algorithm itself. For example, both partitioning and separate-and-conquer methods implicitely select features for inclusion in a branch or rule in preference to other features that appear less relevant, and in the final model some of the initial features might not appear. Classical induction algorithms like ID3, C4.5 or CART are in this category, while algorithms like Naive-Bayes or IB1 include all the presented features in the model and no selection is done.

The embedded approach is then done within the learning algorithm preferring some features instead of others and possibly not including all the available features in the final model induced by the learning algorithm. However, filter and wrapper approaches are located one abstraction level above the embedded one, performing a feature selection process for the final model apart from the embedded selection done by the learning algorithm itself.

A final remark should be done for a category of approaches not always considered in the classical classification of feature selection methods: *feature weighting approaches*. In this class of methods, actual feature selection is substituted by a feature weighting procedure able to weight the relevance of the features. The most important example can be found in the multi-layer perceptron, which weights network units depending on the error over the instances in the training set. Another example is the WINNOW system [21] able to adjust feature weights depending of the fact that a false positive or a false negative is discovered during learning. The method FEATUREBOOST [22] is also a weighting approach using boosting over the attributes; weights are updated at each iteration in such a way that focus is given to less used features. Finally, k-means clustering is exploited as a feature weighting method for feature selection in [37].

In conclusion, adopting a filtering approach means to have a quite good computational complexity, but the higher complexity of the wrapper approach will also produce higher accuracy in the final result. The filtering approach is a very flexible one, since any target learning algorithm can be used in conjunction, while the wrapper approach is strictly dependent on the learning algorithm; faster is the latter, better is the selection process from the computational point of view. Embedded approaches are intrinsic to some learning algorithm and so only those algorithm designed with this characteristic can be used. Finally, if a weighting scheme can be devised, feature selection can be implemented via feature weighting, by postponing the selection as a subsequent possible choice using the weights.

2 AdHoc Algorithm

As already mentioned, feature selection may be accomplished independently of the performance of the learning algorithm used in the knowledge extraction stage. Optimal feature selection is achieved by maximizing or minimizing a criterion function. Such approach are referred to as the *absolute* or *filter feature selection model*. Conversely, the effectiveness of the performance dependent or *feedback feature selection model* is directly related to the performance of the concept discovery algorithm, usually in terms of its predictive accuracy.

Kohavi and Pfleger [19] argued that feedback models are preferable for feature selection algorithms and supported their claims with empirical evidence. However, the literature do not address some important issues. First of all, it is not clear which is the best starting point for the search of a good subset of features. Starting the search on the whole set of original features usually turns out to be unfeasible due to combinatorial explosion when the number of features is not limited. Second, current feature selection algorithms do not help to answer a basic question that arises in a number of data analysis tasks, that is whether there exist some fundamental dimensions which underlie the given set of observed features. This is a major drawback in marketing applications, for example, in which gaining an insight of the deep structure of the data is as important as achieving a good generalization performance.

The attempt to address these open issues have been the basis of our research work on AdHoc (Automatic Discoverer of Higher- Order Correlations), a statistical algorithm that combines the advantages of both filter and feedback feature selection models to enhance the understanding of the given data and increase the efficiency of the feature selection process. AdHoc has been successfully used in fielded applications by TiLab, for instance to support the task of estimating the development effort of telecommunication software projects.

2.1 AdHoc Architecture

The architecture of AdHoc is shown in Figure 1.

Figure 1: AdHoc's architecture

AdHoc comprises two main steps, namely the Data Reduction step and the Feature Selection step. The Data Reduction step is concerned with reducing the dimensionality of the data and is independent of the knowledge discovery algorithm. An iterative process is applied to explore dependences between data and to partition the set of observed variables into a small number of clusters (factors), such that those variables in a given cluster are thought to be measures of the same underlying construct. Data reduction is beneficial under two point of views: on the one hand, it provides a deep insight into the structure of the problem at hand (each factor represents a data dimension); on the other hand, it delivers a very good starting point for the search of relevant features. Indeed, in the second step of AdHoc, namely the Feature Selection step, a genetic algorithm (GA) [36] is used to select the minimum number of most informative features from every factor. This approach has three major advantages:

- The likelihood of selecting good performing features grows markedly, as subsets which account for all the problem dimensions are formed;
- The complexity of search diminishes consistently, as the GA works through a far smaller space. The factor space is, usually, one order of magnitude smaller than the original feature space;
- The possibility of selecting a bad feature subset due to overfitting problems decreases.

AdHoc delivers three outputs:

- A hierarchy of factors, that is, a representation of the dimension underlying the data organized into level of abstractions;
- The best performing feature subset;
- A ranking of the observed features according to their informative power.

2.1.1 Data reduction step

Statistical techniques, like factor analysis, principal component analysis and cluster analysis (hereafter designated as Statistical Data Reduction Techniques or SDRTs), may not represent an optimal solution to the data reduction issue in the data mining framework [48] as they rely on a set of mathematical assumptions that diminish their applicability in a number of applications. Indeed, SDRTs:

- are fooled by spurious or masked correlations between features since they reduce data dimensionality on the basis of direct (1st-order) correlation only;
- are suitable to handle just numeric features;

- their outcome is rarely easy to interpret.

AdHoc provides a different approach to data reduction that overcomes some of the problems which degrade the performance of pure statistical techniques. AdHoc search for true association between the data is based on the concept of feature profile. The profile of a feature F denotes which features F is related to and which ones F is not related to. For example, let A, B, C, D, E, and F be six features that characterize a given data set. Also, let 0.2, 0.1, -0.8, 0.3, and 0.9, be estimates of the direct relationships between F and A, B, C, D, and E, respectively. F's profile is defined as the vector ¡0.2, 0.1, -0.8, 0.3, 0.9, 1.0¿.

Features which have similar profiles provide different measurement of the same concept for they are equally related (unrelated) to the rest of the features. AdHoc can handle both numeric and symbolic features: numeric features are automatically discretized if they need to be compared with symbolic features [49]. Comparing feature profiles may yield to a more reliable estimate of true association than a direct measure of association. Since components of the profile vector express correlations, comparing feature profiles may be viewed as correlating correlations. Nth-order correlations are recursively calculated by applying a statistical test that estimates profile similarity.

By examining higher-order correlations, one can determine the strength of relationship between features and group those features that are equivalent measures of some data dimension (factor). AdHoc produces a hierarchy of clusters which would resemble the concepts that characterize the observed phenomenon. Resulting clusters either represent a well defined concept in the data or hold features that do not contribute to a precise, unique concept.

2.1.2 Feature selection step

The problem of feature selection involves finding a good subset of features under some objective function, such as generalization performance. A feature subset cannot be truly informative and, consequently, good performing on unseen cases, unless it contains at least one feature which contribute to define every dimension underlying the data. To increase search efficiency and avoid data overfitting, AdHoc selects at most one feature from each of the factor (data dimension) that has been discovered in the data reduction step. As a consequence, feature subsets that reflect all the problem dimensions are formed.

We investigated several search heuristics to select the smallest number of features from each factor [50]. Among the others, genetic algorithms (GAs) turned out to be an excellent fit to this task [47]. In our experiments, fitness associated to a feature subset x was the ten-fold crossvalidated predictive accuracy of the C4.5 induction algorithm that would learn the data characterized by the x features only. The size of the space originated by factors turned out to be one order of magnitude less than the one originated by the whole set of original features, and the genetic search for good performing feature subsets over the factor space was three/four times faster than the search on the overall feature space. Most important, since factors accounts for all the dimensions in the data, genetic search over the factor space is equally likely to find the best performing feature subset than the search on the overall feature space, but is more likely to discover the leastsized feature subset.

2.1.3 Ranking of the features

By analyzing the distribution of features in the final population generated by the genetic algorithm, AdHoc is able to rank the features according to their informative content, that is, on the basis of their predictive power. For every feature, the ranking mechanisms accounts both for its occurrence frequency (number of its occurrences in the individuals within the population) and for its relevance, that is, for the number of occurrences in high-performing feature subsets. It is quite intuitive that a feature which appears many times in subsets that yield low predictive accuracy is less relevant than a feature that occurs a limited number but in the most predictive subsets.

2.2 Example Application

We conducted an extensive empirical analysis in order to evaluate the effectiveness of AdHoc. Several realworld data sets featuring different types of problematic features were selected from the U. C. Irvine Repository and the StatLog Repository. The COCOMO data set was provided by [9]. To estimate generalization performance of feature subsets, 10-fold cross-validation was used. AdHoc was first run on the training data; then the test set was used to evaluate the performance of the best feature subset learned by the GA.

Averaging over 10 runs, we noticed that the performance of feature subsets discovered by AdHoc improves C4.5 on 11 out of 14 domains. In particular, five times the improvement is significant at the 95and twice at the 90is worse than C4.5's on the remaining three domains, in one of which, namely Segment, the degradation was significant at the 95As far AdHoc 's efficiency, the data reduction step may take at most 10 minutes to yield the factor hierarchy on a SUN SparcStation 20. The feature selection step is more consuming: it may last from few minutes to 3/4 hours for the largest datasets.

2.2.1 A fielded application

Over the years, corporations have become increasingly dependent on software to meet their objectives. As a consequence, a great amount of research has been done to develop cost models for estimating the software development effort. CSELT has developed a multistrategy learning methodology aimed at assessing commercial cost estimation tools that are used to estimate the development effort of a software project. In case commercial tools turn out to provide unreliable estimates, the methodology is able to deliver a concise and highly-predictive cost models by investigating a database of past projects [7].

The methodology involves two stages. In the first stage, AdHoc explores the database of historical projects and identifies the project characteristics, called cost drivers, which have major influence on project cost. In the second stage, a case-based reasoning tool (cbr) is used to select the historical project which is the most similar one to the new project, i.e., the project whose development effort has to be estimated. The structure and parameters of the similarity measure embedded in the cbr system are set on the basis of the ranking of cost drivers produced by AdHoc. A statistical procedure yields the final estimate of the development effort for the new project.

References

[1] D.W. Ahah and R.L. Bankert. Feature selection for case-based classification of cloud types: an empirical comparison. In *Proc. AAAI 94*, pages 106–112, Seattle, 1994.

[2] R.P.W. Duin A.K. Jain and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[3] H. Almuallin and T.G. Dietterich. Learning with many irrelevant features. In *Proc. AAAI-91*, pages 547–552, Anhaim, CA, 1991.

[4] D.A. Bell and H. Wang. A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41:175–195, 2000.

[5] M. Ben-Bassat. Pattern recognition and reduction of dimensionality. In P.P. Krishnaiah and L.N. Kanal, editors, *Handbook of Statistics*, pages 773–791. North Holland, 1982.

[6] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford Clarendon Press, 1995.

[7] R. Bisio and F. Malabocchia. Cost estimation of software projects through case-based reasoning. In *Proc. of the 1st Int. Conf. on Case-Based Reasoning*, 1995.

[8] A.I. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997.

[9] B. W. Bohem. *Software Engineering Economics.* Prentice Hall, 1981.

[10] K.M. Mohiuddin C. Mao and A.K. Jain. Parsimonious network design and feature selection thorugh node pruning. In *Proc. 12th Intl. Conf. on Pattern Recognition*, pages 622–624, 1994.

[11] C. Cardie. Using decision trees to improve case-based learning. In *Proc. 10th Intl. Conf. on Machine Learning*, pages 25–32, Amherst, MA, 1993.

[12] R. Caruana and D. Freitag. Greedy attribute selection. In *Proc. 11th Intl. Conf. on Machine Learning*, pages 25–32, New Brunswick, NJ, 1994.

[13] H. Chernoff and L.E. Moses. *Elementary Decision Theory.* J. Wiley & Sons, 1959.

[14] T.M. Cover. The best two independent measurements are not the two best. *IEEE Transactions on System, Man and Cybernetics*, 4:116–117, 1974.

[15] T.M. Cover and J.M. VanCampenhout. On the possible orderings in the measurement selection problem. *IEEE Transactions on System, Man and Cybernetics*, 7(9):657–661, 1977.

[16] S. Das. Filters, wrappers and a boosting-based hybryd for feature selection. In *Proc. 18th Intl. Conf. on Machine Learning*, 2001.

[17] J.G. Dy and C.E. Brodley. Feature subset selection and order identification for unsupervised learning. In *Proc. 17th Intl. Conf. on Machine Learning*, pages 247–254, 2000.

[18] A.M. Fanelli G. Castellano and M. Pelillo. An iterative pruning algorithm for feedforward neural networks. *IEEE Transactions on Neural Networks*, 8(3):519–531, 1997.

[19] R. Kohavi G. John and K. Pfleger. Irrelevant features and the subset selection problem. In *Proc. 11th Intl. Conf. on Machine Learning*, pages 121–129, New Brunswick, NJ, 1994.

[20] R. Etxeberria I. Inza, P. Larranaga and B. Sierra. Feature subset selection by bayesian network-based optimization. *Artificial Intelligence*, 123:157–184, 2000.

[21] M.K. Warmouth J. Kivinen and P. Auer. The Perceptron algorithm versus Winnow: linear versus logarithimic mistake bounds when new input variables are relevant. *Artificial Intelligence*, 97:325–343, 1997.

[22] R. Caruana J. O'Sullivan, J. Langford and A. Blum. FeatureBoost: a meta-learning algorithm that improves model robustness. In *Proc. 17th Intl. Conf. on Machine Learning*, pages 703–710, 2000.

[23] A.K. Jain and B. Chandrasekaran. Dimensionality and sample size considerations in pattern recognition practice. In P.P. Krishnaiah and L.N. Kanal, editors, *Handbook of Statistics*, pages 835–855. North Holland, 1982.

[24] A.K. Jain and D. Zongker. Feature selection: evaluation, application and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

[25] J.Doak. An evaluation of feature selection methods and their application to computer security. Technical Report CSE-92-18, Univ. of California at Davis, 1992.

[26] K. Kira and L.A. Rendell. The feature selection problem: traditional methods and a new algorithm. In *Proc. AAAI-92*, pages 122–126, San Jose, CA, 1992.

[27] J. Kittler. Feature set search algorithms. In C.H. Chen, editor, *Pattern Recognition and Signal Processing*, pages 41–60. 1978.

[28] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[29] D. Koller and M. Sahami. Toward optimal feature selection. In *Proc. 13th Intl. Conf. on Machine Learning*, pages 284–292, Bari, Italy, 1996.

[30] I. Kononenko. Estimating attributes: analysis and extension of RELIEF. In *Proc. European Conf. on Machine Learning*, pages 171–182, Catania, Italy, 1994.

[31] L. Kuncheva. genetic algorithms for feature selection for parallel classifiers. *Information Processing Letters*, pages 163–168, 1993.

[32] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and data Mining*. Kluwer Academic Press, 1998.

[33] H. Liu and R. Sedtiono. Incremental feature selection. *Applied Intelligence*, 9(3):217–230, 1998.

[34] J. Han M. Chen and P. Yu. Data mining: an overview from database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–833, 1996.

[35] O.L. Mangasarian. Mathematical programming in data mining. *Data Mining and Knowledge Discovery*, 1:183–201, 1997.

[36] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 1994.

[37] B. Mirkin. Concept learning and feature selection based on square-error clustering. *Machine Learning*, 35:25–39, 1999.

[38] T. Mitchell. *Machine Learning*. Mc Graw Hill, 1997.

[39] A.W. Moore and M.S. Lee. Efficient algorithms for minimizing cross validation errors. In *Proc. 11th Intl. Conf. on Machine Learning*, pages 190–198, New Brunswick, NJ, 1994.

[40] P. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26(9):917–922, 1977.

[41] A.Y. Ng. On feature selection: learning with exponentially many irrelevant features as training examples. In *Proc. 15th Intl. Conf. on Machine Learning*, pages 404–412, 1998.

[42] J. Novovicova P. Pudil and J. Kittler. Floating search methods in feature selection. *Pattern Recognition*, 28(9):1389–1398, 1995.

[43] J. Novovicova P. Somol, P. Pudil and P. Paclik. Adaptive floating search methods in feature selection. *Pattern Recognition Letter*, 20(11-13):1157–1163, 1999.

[44] F. Provost and V. Kolluri. A survey of methods for scaling-up inductive algortihms. *Data Mining and Knowledge Discovery*, 2:131–169, 1999.

[45] S.J. Raudys and A.K. Jain. Small sample size effects in statistical pattern recognition: recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):252–264, 1991.

[46] S.J. Raudys and V. Pikelis. On dimensionality sample size, classification error and complexity of classification algorithms in pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:243–251, 1980.

[47] M. Richeldi and P. L. Lanzi. Improving genetic-based feature selection by reducing data dimensionality. In *Proc. of the Workshop on Evolutionary Computation, Int. Conf. on Machine Learning*, 1996.

[48] M. Richeldi and P. L. Lanzi. Performing effective feature selection by investigating the deep structure of the data. In *Proc. of the 2nd Int. Conf. on Knowledge Discovery and Data Mining*, 1996.

[49] M. Richeldi and M. Rossotto. Class-driven statistical discretization of continuous attributes. In *Proc. of the 8th European Conf. of Machine Learning*, 1995.

[50] M. Richeldi and M. Rossotto. Combining statistical techniques and search heuristic to perform effective feature selection. In *Machine Learning and Statistics: the Interface*, 1995.

[51] D.B. Skalak. Prototype and feature selection by sampling and random mutation hill-climbing algorithms. In *Proc. 11th Intl. Conf. on Machine Learning*, pages 293–301, New Brunwick, NJ, 1993.

[52] L. Talavera. Feature selection as a preprocessing step for hierarchical clustering. In *Proc. 16th Intl. Conf. on Machine Learning*, 1999.

[53] H. Vafaie and K. DeJong. Robust feature selection algorithms. In *Proc. 5th Intl. Conf. on Tools with Artificial Intelligence*, pages 356–363, Rockville, MD, 1993.

[54] I. Witten and E. Frank. *Data Mining: Praxtical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, 2000.

[55] J. Yang and V. Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13(2):44–49, 1998.