Feature Selection for Machine Learning: Comparing a Correlation-based Filter Approach to the Wrapper

Mark A. Hall, Lloyd A. Smith {mhall, las}@cs.waikato.ac.nz
Department of Computer Science
University of Waikato
Hamilton
New Zealand.
Phone +64 7 856 2889 extn 6017

Abstract

Feature selection is often an essential data processing step prior to applying a learning algorithm. The removal of irrelevant and redundant information often improves the performance of machine learning algorithms. There are two common approaches: a *wrapper* uses the intended learning algorithm itself to evaluate the usefulness of features, while a *filter* evaluates features according to heuristics based on general characteristics of the data. The wrapper approach is generally considered to produce better feature subsets but runs much more slowly than a filter. This paper describes a new filter approach to feature selection that uses a correlation based heuristic to evaluate the worth of feature subsets When applied as a data preprocessing step for two common machine learning algorithms, the new method compares favourably with the wrapper but requires much less computation.

Introduction

Many factors affect the success of machine learning on a given task. The quality of the data is one such factor—if information is irrelevant or redundant, or the data is noisy and unreliable, then knowledge discovery during training is more difficult. Feature subset selection is the process of identifying and removing as much of the irrelevant and redundant information as possible. Machine learning algorithms differ in the amount of emphasis they place on feature selection. At one extreme are algorithms such as the simple nearest neighbour learner, which classifies novel examples by retrieving the nearest stored training example, using all the available features in its distance computations. Towards the other extreme lie algorithms that explicitly try to focus on relevant features and ignore irrelevant ones. Decision tree inducers are examples of this approach. By testing the values of certain features, decision tree algorithms attempt to divide training data into subsets containing a strong majority of one class. This necessitates the selection of a small number of highly predictive features in order to avoid overfitting the training data. Regardless of whether a learner attempts to select features itself or ignores the issue, feature selection prior to learning can be beneficial. Reducing the dimensionality of the data reduces the size of the hypothesis space and allows algorithms to operate faster and more effectively. In some cases accuracy on future classification can be improved; in others, the result is a more compact, easily interpreted representation of the target concept.

Algorithms that perform feature selection as a preprocessing step prior to learning can generally be placed into one of two broad categories. One approach, referred to as the *wrapper* (John, Kohavi, and Pfleger, 1994) employs—as a subroutine—a statistical re-sampling technique (such as cross validation) using the actual target learning algorithm to estimate the accuracy of feature subsets. This approach has proved useful but is very slow to execute because the learning algorithm is called repeatedly. For this reason, wrappers do not scale well to large datasets containing many features. Another approach, called the *filter* (John, Kohavi, and Pfleger, 1994), operates independently of any learning algorithm—undesirable features are filtered out of the data before induction commences. Filters typically make use of all the available training data when selecting a subset of features. Some look for consistency in the data—that is, they note when every combination of values for a feature subset is associated with a single class label (Almuallim and Dietterich, 1992). Another method (Koller and Sahami, 1996) eliminates features whose information content is subsumed by some number of the remaining features. Still other methods attempt to rank features according to a relevancy score (Kira and Rendell, 1992; Holmes and Nevill-Manning, 1995). Filters have proven to be much faster than wrappers and hence can be applied to large data sets containing many features. Their general nature allow them to be used with any learner, unlike the wrapper, which must be re-run when switching from one learning algorithm to another.

This paper presents a new approach to feature selection, called CFS, (Correlation-based Feature Selection) that uses a correlation based heuristic to evaluate the worth of features. The effectiveness of CFS is evaluated by comparing it with a well known wrapper feature selector that uses a specific learning algorithm to guide its search for good features. The results presented in this paper show that CFS compares favourably with the wrapper but requires far less computation.

CFS: Correlation-based Feature Selection

Feature evaluation

At the heart of the CFS algorithm is a heuristic for evaluating the worth or merit of a subset of features. This heuristic takes into account the usefulness of individual features for predicting the class label along with the level of intercorrelation among them. The hypothesis on which the heuristic is based can be stated:

Good feature subsets contain features highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other.

In test theory (Ghiselli, 1964), the same principle is used to design a composite test for predicting an external variable of interest. In this situation, the "features" are individual tests which measure traits related to the variable of interest (class).

Equation 1 (Ghiselli, 1964) formalises the heuristic:

$$Merit_s = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{1}$$

where $Merit_S$ is the heuristic "merit" of a feature subset S containing k features, $\overline{r_{cf}}$ is the mean feature-class correlation ($f \in S$), and $\overline{r_{ff}}$ is the average feature-feature intercorrelation. Equation1 is, in fact, Pearson's correlation, where all variables have been standardised. The numerator can be thought of as giving an indication of how predictive of the class a group of features are; the denominator of how much redundancy there is among them. The heuristic handles irrelevant features as they will be poor predictors of the class. Redundant attributes are discriminated against as they will be highly correlated with one or more of the other features.

Feature Correlations

Classification tasks in machine learning often involve learning from categorical features, as well those that are continuous or ordinal. In order to have a common basis for computing the correlations necessary for Equation 1, continuous features are transformed to categorical features in a preprocessing step using the supervised discretisation method of Fayyad and Irani (1993). A measure based on information theory estimates the degree of association between nominal features.

If X and Y are discrete random variables, Equations 2 and 3 give the entropy of Y before and after observing X.

$$H(Y) = -\sum_{y \in Y} p(y) \log_2 p(y), \tag{2}$$

$$H(Y|X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log_2 p(y|x). \tag{3}$$

The amount by which the entropy of Y decreases reflects the additional information about Y provided by X and is called the *information gain* (Quinlan, 1993). Information gain is given by

$$\begin{aligned} \text{gain} &= H(Y) - H(Y|X) \\ &= H(X) - H(X|Y) \\ &= H(Y) + H(X) - H(X,Y). \end{aligned} \tag{4}$$

Information gain is a symmetrical measure—that is, the amount of information gained about Y after observing X is equal to the amount of information gained about X after observing Y. Unfortunately, information gain is biased in favour of features with more values, that is, attributes with greater numbers of values will appear to gain more information than those with fewer values even if they are actually no more informative. Furthermore, the correlations in Equation 1 should be normalized to ensure they are comparable and have the same effect. Symmetrical uncertainty (Press et al. 1988) compensates for information gain's bias toward attributes with more values and normalises its value to the range $[0, 1]$:

$$\text{symmetrical uncertainty} = 2.0 \times \left[\frac{\text{gain}}{H(Y) + H(X)} \right] \tag{5}$$

Searching the Feature Subset Space

The purpose of feature selection is to decide which of the initial features to include in the final subset and which to ignore. If there are n possible features initially, then there are 2^n possible subsets. The only way to find the best subset would be to try them all—this is clearly prohibitive for all but a small number of initial features.

Various heuristic search strategies such as hill climbing and best first (Rich and Knight, 1991) are often applied to search the feature subset space in reasonable time. CFS starts from the empty set of features and uses a forward best first search with a stopping criterion of five consecutive fully expanded non-improving subsets. Best first search is also the preferred search strategy to use with the wrapper (Kohavi, 1995).

Applying CFS to Machine Learning Problems

Figure 1 shows the stages of the CFS algorithm and how it is used in conjunction with a machine learning algorithm. A copy of the training data is first discretized using the method of Fayyad and Irani (1993), then passed to CFS. CFS calculates feature-class and feature-feature correlations using symmetrical uncertainty and then searches the feature subset space. The subset with the highest merit (as measured by Equation 1) found during the search is used to reduce the dimensionality of both the original training data and the testing data. Both reduced datasets may then be passed to a machine learning algorithm for training and testing.

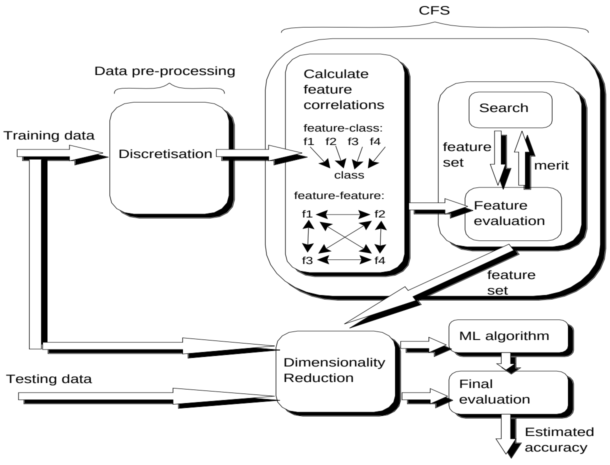

Figure 1: The components of CFS.

Locally Predictive Features

Because correlations are estimated globally (over all training instances), CFS tends to select a "core" subset of features that has low redundancy and is strongly predictive of the class. In some cases however, there may be subsidiary features that are *locally predictive* in a small area of the instance space. CFS includes a heuristic to include locally predictive features and avoid the re-introduction of redundancy.

Experimental Results

Experiments with the basic version of CFS described above have shown that it can be of use to machine learning algorithms in terms of improving accuracy and comprehensibility of induced models (Hall and Smith, 1998; Hall 1998). In this paper we look at how CFS compares with a well known wrapper feature selector. This section present the results of experiments designed to compare the performance of common machine learning algorithms after feature selection by CFS with their performance after feature selection by the wrapper. In particular, the accuracy of learners and the size of models produced after feature selection are compared. In addition, the execution time of CFS is compared with the wrapper.

Machine Learning Algorithms

Two machine learning algorithms representing two diverse approaches to learning were used in the experiments—a probabilistic learner (naive Bayes) and a decision tree learner (C4.5).

Naive Bayes employs a simplified version of Bayes formula to classify each novel example. The posterior probability of each possible class is calculated using conditional probabilities for feature values and prior probabilities of classes estimated from the training data; each novel instance is assigned the class with the highest posterior probability. Due to the assumption that feature values are independent given the class, the naive Baye's classifier's predictive performance can be adversely affected by the presence of *redundant* features in the training data.

C4.5 (Quinlan, 1993) is an algorithm that summarises the training data in the form of a decision tree. Along with systems that induce logical rules, decision tree algorithms have proved popular in practice. This is due in part to their robustness and execution speed, and to the fact that explicit concept descriptions are produced, which users can interpret.

C4.5 "grows" decision trees recursively, using a greedy approach to decide which attributes to test at the nodes of the tree. An information theoretic measure similar to symmetric uncertainty guides the process. C4.5 can sometimes overfit training data, resulting in large trees. In many cases, feature selection can result in C4.5 producing smaller trees.

Wrapper Feature Selection

The $\mathcal{MLC}$++ machine learning library (Kohavi et al. 1994) was used to provide results for the wrapper. Tenfold cross validation of the training data was used to provide an estimate of the accuracy of feature sets with respect to a particular learning algorithm. The same search strategy and stopping criterion as CFS are employed by the wrapper.

Experiments

Twelve standard datasets drawn from the UCI collection (Merz and Murphy, 1996) were used in the experiments: they are summarised in Table 1. These datasets were chosen because of the prevalence of nominal features and their predominance in the literature. Three of the datasets (australian, lymphography, and horse-colic) contain a few continuous features; the rest contain only nominal features.

Fifty runs were done for each machine learning algorithm on each dataset with features selected by CFS and by the wrapper. In each run, a dataset was randomly split into a training and testing set (sizes given in Table 1). CFS and the wrapper were applied in turn to the full training set to select features. Separate training and testing sets consisting of features selected by CFS and features selected by the wrapper were created and each machine learning algorithm was applied to these dimensionally reduced datasets.

Table 2 shows the results of feature selection for naive Bayes; results for naive Bayes with no feature selection (All features) are shown as well. Accuracies give the percentage of correct classifications, averaged over the fifty trials. Results for CFS are shown in bold if they show significant improvement over the corresponding result for the wrapper, and *vice versa*. A "+" or "−"

Table 1: Datasets used in the experiments

Dataset	Features	Max/min feature values	Classes	Train size/ test size
mushroom	23	12/1	2	1000/7124
vote	17	2/2	2	218/217
vote1	16	2/2	2	218/217
australian	16	23/2	2	228/462
lymph	19	8/2	4	98/50
primary-tumor	18	3/2	23	226/113
breast-cancer	10	11/2	2	191/95
dna-promoters	56	4/4	2	69/37
audiology	70	6/2	24	149/77
soybean	36	7/2	19	450/223
horse-colic	28	346/2	2	242/126
kr-vs-kp	37	3/2	2	2110/1086

sign shows where results for CFS are significantly better or worse than when no feature selection is performed (all the features are used), and similarly for the wrapper. Throughout, we speak of results being "significantly different" if the difference is statistically different at the 5% level according to a paired two-sided *t*-test.

Similarly, Table 3 shows the results of feature selection for C4.5.

Table 2: Accuracy of naive Bayes with feature selection by CFS and the wrapper.

Dataset	CFS	Wrapper	All features
mushroom	98.53+	**98.86+**	94.75
vote	95.20+	95.24+	90.25
vote1	89.51+	88.95+	87.20
australian	85.90+	85.16+	78.21
lymph	**83.92+**	76.00−	82.12
primary-tumor	**46.73**	42.32−	46.87
breast-cancer	**72.06**	70.96−	72.16
dna-promoters	**90.58**	82.05−	89.21
audiology	75.04−	**79.33**	80.24
soybean	92.69+	92.99+	91.30
horse-colic	86.24+	**87.70+**	83.13
kr-vs-kp	94.24+	94.36+	87.33

Discussion of Results

CFS outperforms the wrapper four times for naive Bayes and five times for C4.5, while the wrapper outperforms CFS three times for both learning algorithms. Furthermore, as shown by the entries marked with "+" or "−" in the tables, CFS improves the accuracy of the learning algorithms more times and degrades accuracy fewer times than the wrapper does. For naive Bayes, CFS improves accuracy eight times and degrades accuracy only once; the wrapper improves accuracy seven times but degrades accuracy four times. For C4.5, CFS improves accuracy twice and degrades accuracy twice; the wrapper improves accuracy three times but degrades accuracy five times.

It appears that the wrapper has some difficulty on datasets with fewer examples. Cross validation accuracy estimates can exhibit greater variability when the number of examples is small (Kohavi, 1995), and the wrapper may be overfitting these datasets in some cases. CFS, on the other hand, does not need to reserve part of the training data for evaluation purposes, and, in general, tends to do better on smaller datasets than the wrapper.

Figure 2 shows how feature selection by the wrapper and the CFS affects the size of the trees induced by C4.5. Bars below the zero line indicate that feature selection has *reduced* the size of the trees. The graph shows that both feature selectors reduce the size of the trees induced by C4.5 more often than not. The wrapper tends to result in somewhat smaller trees than CFS.

Figure 3 shows the average number of features selected on each dataset by the wrapper using naive Bayes and by CFS. CFS generally selects a similar sized feature set as the wrapper[1]. In many cases the number of features is reduced by more than half by both methods.

CFS executes many times faster than the wrapper. On a Sparc server 1000, a single trial took one cpu unit or less to complete for all datasets except kr-vs-kp, which averaged 8 cpu units. By comparison, the wrapper ranged from 123 cpu units to complete one trial on breast-cancer to over 9000 cpu units to complete one trial on kr-vs-kp. The wrapper is cubic in the number of features, whereas CFS is squared in the number of features.

Table 3: Accuracy of C4.5 with feature selection by CFS and the wrapper.

Dataset	CFS	Wrapper	All features
mushroom	98.55−	**99.03−**	99.59
vote	95.39	95.33	95.23
vote	**88.66**	87.82−	88.71
australian	**85.89+**	84.75+	83.69
lymph	76.67	**80.86+**	75.80
primary-tumor	**41.00**	39.53−	40.99
breast-cancer	70.97	71.11	71.77
dna-promoters	**76.89+**	74.21	74.58
audiology	**77.60**	75.50−	78.48
soybean	88.88	89.65	89.16
horse-colic	84.79	85.56+	84.02
kr-vs-kp	94.23−	**97.19−**	99.16

Conclusion

This paper has presented a correlation-based approach to feature selection for machine learning and compared it with the wrapper—a well known feature selection technique that uses the target learning algorithm to guide its search for good features. The experiments have shown that, in many cases, CFS gives results that are comparable or better than the wrapper. Because CFS makes use of all the training data at once, it can

[1]The number of features selected by the wrapper using C4.5 is very similar. Note that because CFS is a filter, the feature sets it selects are the same regardless of the final learning algorithm.

give better results than the wrapper on small datasets. CFS is much faster than the wrapper (by more than an order of magnitude), which allows it to be applied to larger datasets than the wrapper.

Many applications of machine learning involve predicting a "class" that takes on a continuous numeric value. Future work will aim at extending CFS to handle problems where the class is numeric.

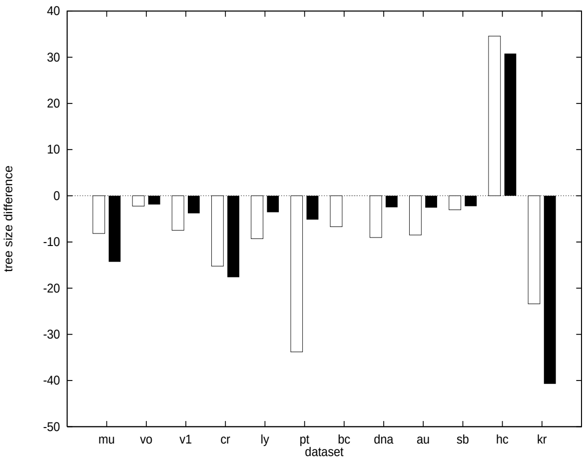

Figure 2: Average change in the size of the trees induced by C4.5 when features are selected by the wrapper (left) and CFS (right).

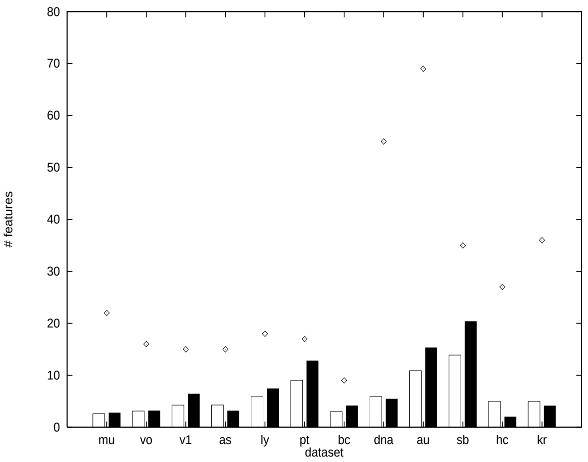

Figure 3: Number of features selected by the wrapper using naive Bayes (left) and CFS (right). Dots show the number of features in the original dataset.

References

Almuallim, H. and Dietterich, T. G. 1992. Efficient Algorithms for Identifying Relevant Features. In *Proceedings of the Ninth Canadian Conference on Artificial Intelligence*, 38–45. Morgan Kaufmann.

Fayyad, U. M. and Irani, K. B. 1993. Multi-interval Discretisation of Continuous-valued Attributes for Classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence.* Morgan Kaufmann.

Ghiselli, E. E. 1964. *Theory of Psychological Measurement.* McGraw-Hill.

Hall, M. A. 1998. Correlation-based Feature Selection for Machine Learning. Ph.D diss. Dept. of Computer Science, Waikato Univ.

Hall, M. A. and Smith, L. A. 1998. Practical Feature Subset Selection For Machine Learning. In Proceedings of the 21st Australasian Computer Science Conference, 181–191. Springer.

Holmes, G. and Nevill-Manning, C. G. 1995. Feature Selection via the Discovery of Simple Classification Rules. In *Proceedings of the International Symposium on Intelligent Data Analysis.*

John, G. H.; Kohavi, R.; and Pfleger, P. 1994. Irrelevant Features and the Subset Selection Problem. In *Machine Learning: Proceedings of the Eleventh International Conference.* Morgan Kaufmann.

Kira, K. and Rendell, L. 1992. A Practical Approach to Feature Selection. In *Machine Learning: Proceedings of the Ninth International Conference.* Morgan Kaufmann.

Kohavi, R. 1995. Wrappers for Performance Enhancement and Oblivious Decision Graphs. Ph.D diss. Dept. of Computer Science, Stanford Univ.

Kohavi, R.; John, G.; Long, R.; Manley, D.; and Pfleger, K. 1994. $\mathcal{MLC}$++: A machine learning library in C++. In *Tools with Artificial Intelligence,* 740–743. IEEE Computer Society Press

Koller, D. and Sahami, M. 1996. Towards Optimal Feature Selection. In *Machine Learning: Proceedings of the Thirteenth International Conference*, 294–292. Morgan Kaufmann.

Merz, C. J. and Murphy, P. M. 1996. UCI *Repository of Machine Learning Data-bases* [http://www.ics.edu/~mlearn/MLRepository.html]. Irvine Calif.: Univ. of Calif., Dept. of Information Science.

Press, W. H.; Flannery, B. P.; Teukolski, S. A.; and Vetterling, W. T. 1988. *Numerical Recipes in C.* Cambridge University Press.

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

Rich, E. and Knight, K. 1991. *Artificial Intelligence.* McGraw-Hill.