Working Paper Series
ISSN 1170-487X

Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning

by Mark A Hall

Working Paper 00/8
May 2000

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, New Zealand

Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning

Mark A. Hall MHALL@CS.WAIKATO.AC.NZ
Department of Computer Science, University of Waikato, Hamilton, New Zealand

Abstract

Algorithms for feature selection fall into two broad categories: *wrappers* that use the learning algorithm itself to evaluate the usefulness of features and *filters* that evaluate features according to heuristics based on general characteristics of the data. For application to large databases, filters have proven to be more practical than wrappers because they are much faster. However, most existing filter algorithms only work with discrete classification problems. This paper describes a fast, correlation-based filter algorithm that can be applied to continuous and discrete problems. The algorithm often outperforms the well-known ReliefF attribute estimator when used as a preprocessing step for naive Bayes, instance-based learning, decision trees, locally weighted regression, and model trees. It performs more feature selection than ReliefF does—reducing the data dimensionality by fifty percent in most cases. Also, decision and model trees built from the preprocessed data are often significantly smaller.

1. 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, that 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. Wrapper methods (Kohavi & John, 1997) employ—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 data sets containing many features. Filter methods, on the other hand (Kohavi & John, 1997), operate 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 & Dietterich, 1991). Another method (Koller & 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 & Rendell, 1992).

Filters have proven to be much faster than wrappers and hence can be applied to large data sets containing many features. Because they are more general they can be used with any learner, unlike the wrapper, which must be re-run when switching from one learning algorithm to another. However, most filter algorithms work only on discrete class problems, unlike the wrapper, which can be "wrapped" around any continuous or discrete class learner.

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 algorithm is simple, fast to execute and extends easily to continuous class problems by applying suitable correlation measures. The next section discusses related work. Section 3 describes the CFS algorithm. Section 4 describes the application of CFS to discrete class problems and presents experimental results comparing CFS to ReliefF as pre-processors for learning algorithms. Section 5 explains how the algorithm is extended to cope with continuous class problems and presents experimental results comparing CFS and ReliefF on continuous class data sets. The last section summarizes the findings.

2. Related Work

While wrapper feature selection can be applied to regression problems with relative ease, few filter algorithms handle continuous class data. The only exception is RReliefF (Regressional Relief) (Robnik-Šikonja & Kononenko, 1997), which is an extension of Kira and Rendell's (1992) Relief algorithm for classification problems. The Relief algorithms are quite different to the algorithm described in this paper in that they score (and hence rank) individual features rather than scoring (and hence ranking) feature subsets. To use Relief for feature selection, those features with scores exceeding a user-specified threshold are retained to form the final subset.

Relief works by randomly sampling an instance and locating its nearest neighbour from the same and opposite class. The values of the attributes of the nearest neighbours are compared to the sampled instance and used to update the relevance scores for each attribute. This process is repeated for a user specified number of instances m. The rationale is that a useful attribute should differentiate between instances from different classes and have the same value for instances from the same class. Relief was originally defined for two-class problems (Kira & Rendell, 1992) and was later extended (ReliefF) to handle noise and multi-class data sets (Kononenko, 1994). ReliefF smoothes the influence of noise in the data by averaging the contribution of k nearest neighbours from the same and opposite class of each sampled instance instead of the single nearest neighbour. Multi-class data sets are handled by finding nearest neighbours from each class that is different from the current sampled instance and weighting their contributions by the prior probability of each class. Robnik–Šikonja and Kononenko (1997) describe RReliefF, an extension of ReliefF to handle regression problems. The algorithm is essentially the same as ReliefF, but instead of requiring exact knowledge of whether two instances belong to the same class or not, it estimates a probability based on the distance between the class values of the two instances.

The implementation of ReliefF used for the experiments reported in this paper encompasses both ReliefF and RReliefF and from here-on is referred to simply as ReliefF. Kononenko (1994) notes that the higher the value of m (the number of instances sampled), the more reliable ReliefF's estimates are—though of course increasing m increases the running time. For all experiments, we set $m = 250$ (Robnik-Šikonja & Kononenko, 1997). For the discrete class experiments, k was set to 10 (Kononenko, 1994). For the numeric class experiments, we used the same parameter settings as Robnik–Šikonja and Kononenko (1997): $m = 250$, $k = 200$ and $\sigma = 20$ (a parameter that controls the exponential decrease in influence for more distant neighbours).

3. CFS: Correlation-based Feature Selection

Unlike ReliefF, CFS evaluates and hence ranks feature subsets rather than individual features. This section describes the CFS algorithm.

3.1 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 is:

Good feature subsets contain features highly correlated with the class, yet uncorrelated with each other.

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

able of interest (class). For example, a more accurate prediction of a person's success in a mechanics training course can be had from a composite of a number of tests measuring a wide variety of traits (ability to learn, ability to comprehend written material, manual dexterity and so forth), rather than from any one individual test which measures a restricted scope of traits.

Equation 1 (Ghiselli, 1964) formalizes the heuristic:

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

where $Merit_S$ is the heuristic "merit" of a feature subset S containing k features, $\overline{r_{cf}}$ the average feature-class correlation, and $\overline{r_{ff}}$ the average feature-feature intercorrelation. Equation 1 is, in fact, Pearson's correlation, where all variables have been standardized. The numerator can be thought of as giving an indication of how predictive 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. Because attributes are treated independently, CFS cannot identify strongly interacting attributes such as in a parity problem. However, it has been shown that it can identify useful attributes under moderate levels of interaction (Hall, 1998).

3.2 Searching the Feature Subset Space

The purpose of feature selection is to decide which of the initial (possibly large) number of 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 (Kohavi & John, 1997) are often applied to search the feature subset space in reasonable time. CFS first calculates a matrix of feature-class and feature-feature correlations from the training data and then searches the feature subset space using a best first search. Best first search was used in the final experiments as it gave slightly better results in some cases than hill climbing. The best first search starts with an empty set of features and generates all possible single feature expansions. The subset with the highest evaluation is chosen and expanded in the same manner by adding single features. If expanding a subset results in no improvement, the search drops back to the next best unexpanded subset and continues from there. Given enough time a best first search will explore the entire feature subset space, so it is common to limit the number of subsets expanded that result in no improvement. The best subset found is returned when the search terminates. CFS uses a stopping criterion of five consecutive fully expanded non-improving subsets.

3.3 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. Some machine learning algorithms are able to make use of locally predictive features and in these situations CFS has been shown to degrade their performance somewhat (Hall, 1998). The version of CFS used in the experiments described in this paper includes a heuristic to include locally predictive features and avoid the re-introduction of redundancy. After the feature subset space has been searched, the remaining unselected features are examined one by one to determine whether they are likely to be useful on a local rather than global basis. A feature will be admitted to the subset if its correlation with the class is higher than the highest correlation between it and any one of the already selected features.

4. Applying CFS to Discrete Class Data

In order to apply Equation 1 to estimate the merit of a feature subset, it is necessary to compute the correlation (dependence) between attributes. For discrete class problems, CFS first discretises numeric features using the technique of Fayyad and Irani (1993) and then uses symmetrical uncertainty (a modified information gain measure) to estimate the degree of association between discrete features (Press et al., 1988):

$$\mathrm{SU} = 2.0 \times \left[\frac{H(X) + H(Y) - H(X,Y)}{H(Y) + H(X)}\right] \quad (2)$$

Symmetrical uncertainty is used (rather than gain ratio) because it is a symmetric measure and can therefore be used to measure feature-feature correlations where there is no notion of one attribute being the "class" as such.

To handle missing data values in an attribute, CFS distributes their counts across the represented values in proportion to their relative frequencies.

In order to evaluate the effectiveness of CFS as a global feature selector for common machine learning algorithms, experiments were performed using sixteen standard data sets from the UCI collection (Blake, Keogh & Merz, 1998). The data sets and their characteristics are listed in Table 1. We also ran ReliefF with three relevance threshold settings: 0.0, 0.05, and 0.1. All features with a relevance score less than the specified threshold were removed (Kohavi & John, 1997). Three machine learning algorithms representing three diverse approaches to learning were used in the experiments—a probabilistic learner (naive Bayes), a decision tree learner (C4.5 release 8) and an instance-based learner (kNN[1]). The percentage of correct classifications, averaged over ten ten-fold cross validation runs, were calculated for each algorithm-data set combination before and after feature selection. For each train-test split, the dimensionality was reduced by each feature selector before being passed to the learning algorithms. In the case of CFS, a discretized copy of each training split was made for it to operate on. The same folds were used for each feature selector–learning scheme combination.

Table 1. Discrete class data sets.

	Data Set	Instances	Num.	Nom.	Classes
1	glass-2	163	9	0	2
2	anneal	898	6	32	5
3	breast-c	286	0	9	2
4	credit-g	1000	7	13	2
5	diabetes	768	8	0	2
6	horse colic	368	7	15	2
7	heart-c	303	6	7	2
8	heart-stat	270	13	0	2
9	ionosphere	351	34	0	2
10	labor	57	8	8	2
11	lymph	148	3	15	4
12	segment	2310	19	0	7
13	sick	3772	7	22	2
14	soybean	683	0	35	19
15	vote	435	0	16	2
16	zoo	101	1	15	7

Tables 2, 3 and 4 summarize the results of feature selection with naive Bayes, C4.5 and instance based learning respectively[2]. The tables show how often the method in a column significantly outperforms the method in a row. Figure 1 shows the number of features selected by ReliefF (thresholds 0 and 0.01) and by CFS as well as the number present in the full data set. Throughout we speak of results being significantly different if the difference is statistically significant at the 1% level according to a paired two-sided t test.

[1]The implementation of k-nearest neighbour used here sets k by cross validation on the training data.

[2]Full results are given in Table 9 at the end of the paper.

Table 2. Results of paired t-tests for naive Bayes.

	NB	CFS	Rlf	Rlf0.01	Rlf0.05	Rlf0.1
NB	-	6	1	4	4	5
CFS	4	-	4	5	3	4
Rlf	0	5	-	2	4	5
Rlf0.01	1	5	1	-	4	4
Rlf0.05	7	9	7	9	-	5
Rlf0.1	10	12	10	11	5	-

Table 3. Results of paired t-tests for C4.5. Figures in braces show the number of times method in column results in a smaller tree than method in row.

	C45	CFS	Rlf	Rlf0.01	Rlf0.05	Rlf0.1
C45	-	4 {9}	0 {2}	1 {6}	3 {11}	0 {10}
CFS	3 {0}	-	3 {0}	3 {1}	4 {7}	2 {10}
Rlf	0 {1}	5 {9}	-	1 {3}	4 {9}	2 {11}
Rlf0.01	1 {2}	2 {11}	1 {0}	-	3 {11}	2 {11}
Rlf0.05	5 {2}	8 {4}	5 {1}	5 {1}	-	2 {9}
Rlf0.1	13 {2}	11 {3}	11 {1}	13 {2}	8 {2}	-

From Tables 2, 3 and 4 it can be seen (by looking at the first row and second column) that CFS maintains or improves the accuracy of each learning scheme more often than not. Accuracy of naive Bayes is improved for six data sets and degraded on four. Accuracy of both C4.5 and IBk is improved for four data sets and degraded for three. On the remaining data sets for each learning scheme there is no change in accuracy after feature selection by CFS. When looking at the tree sizes for C4.5 (figures in braces in Table 3) it can be seen that CFS never increases the size of C4.5's trees and reduces tree size significantly on nine of the sixteen data sets.

ReliefF (with a relevance threshold of 0) results in no significant difference in accuracy for all three learning schemes on all data sets—the sole exception being an improvement for naive Bayes on glass-2. The explanation for this can be seen from Figure 1. ReliefF using a threshold of 0 performs very little feature set reduction. The largest reductions are for glass-2, anneal and sick; on the remainder of the data sets the number of features is reduced by only 2.6% on average. UCI data sets have been (for the most part) quite carefully engineered—that is domain experts have constructed features that they expect to be useful for predicting the class. Since Relief is very effective at identifying irrelevant attributes it is not surprising that most attributes are retained; especially since it is well known that Relief does not identify *redundant* attributes (Kira & Rendell, 1992).

Table 4. Results of paired t-tests for IBk.

	IBk	CFS	Rlf	Rlf0.01	Rlf0.05	Rlf0.1
IBk	-	4	0	3	2	3
CFS	3	-	3	3	1	0
Rlf	0	4	-	3	2	3
Rlf0.01	1	4	0	-	3	3
Rlf0.05	6	6	7	6	-	2
Rlf0.1	11	10	11	11	9	-

From Figure 1 it can be seen that CFS is a more aggressive feature selector than ReliefF. CFS reduces the number of features by 47% on average while maintaining or improving accuracy in most cases. This suggests that while the majority of attributes are useful for predicting the class, only a small subset are necessary in practice for generating an accurate model. Similar findings have been reported elsewhere, for example (Kohavi & John, 1997).

Tables 2, 3 and 4 show that using a threshold of 0.01 with ReliefF generally gives good results for all three learning algorithms. Although resulting in more feature set reduction, increasing the threshold beyond 0.01 leads to a substantial decrease in accuracy. From Figure 1 it can be seen that ReliefF selects fewer attributes with a threshold of 0.01 than with a threshold of 0, but CFS selects significantly fewer attributes than both on all data sets except diabetes.

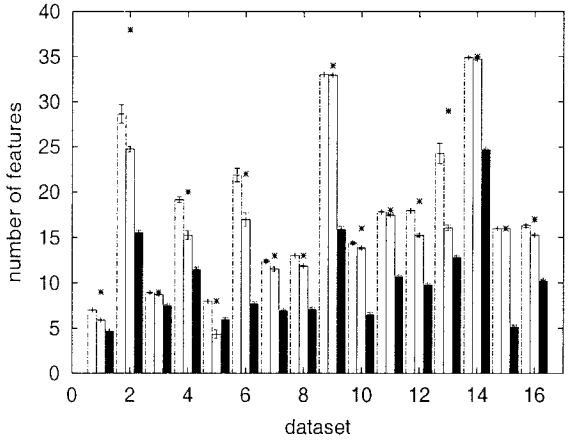

Figure 1. Average number of features selected by ReliefF with threshold 0 (left), ReliefF with threshold 0.01 (center) and CFS (right) on discrete class data sets. All differences between CFS and ReliefF are significant at the 1% level. Crosses show the original number of features in each data set.

5. Applying CFS to Continuous Class Data

For continuous class data the obvious measure for estimating the correlation between attributes in Equation 1 is standard linear (Pearson's) correlation. This is straightforward when the two attributes involved are both continuous:

$$r_{XY} = \frac{\sum xy}{n\sigma_x\sigma_y}, \tag{3}$$

where X and Y are two continuous variables expressed in terms of deviations.

When one attribute is continuous and the other discrete, a weighted Pearson's correlation is calculated as shown in Equation 4. Specifically, for a discrete attribute X and a continuous attribute Y, if X has k values, then k binary attributes are correlated with Y. Each of the $i = 1, ..., k$ binary attributes takes value 1 when the ith value of X occurs and 0 for all other values. Each of the $i = 1, ..., k$ correlations calculated is weighted by the prior probability that X takes value i.

$$r_{XY} = \sum_{i=1}^{k} p(X = x_i) r_{X_{bi}Y}, \tag{4}$$

where X_{bi} is a binary attribute that takes value 1 when X has value x_i and 0 otherwise.

When both attributes involved are discrete, binary attributes are created for both and all weighted correlations are calculated for all combinations as shown in Equation 5.

$$r_{XY} = \sum_{i=1}^{k} \sum_{j=1}^{l} p(X = x_i, Y = y_j) r_{X_{bi}Y_{bj}} \tag{5}$$

In this approach to calculating correlations, CFS replaces any missing values with the mean for continuous attributes and the most common value for discrete attributes.

Experiments on continuous class data follow a similar methodology to that described in Section 4 for the discrete case. The only difference is that features do not need to be discretised before being passed to CFS. Nineteen continuous class data sets and their properties are listed in Table 5 (Frank, Trigg, Holmes & Witten, in press).

Three learning algorithms (close analogs of those used in the discrete class experiments) capable of learn-

Table 5. Numeric class data sets.

	Data Set	Instances	Num.	Nom.
1	autoHorse	205	17	8
2	autoMpg	398	4	3
3	autoPrice	159	15	0
4	bodyfat	252	14	0
5	breastTumor	286	1	8
6	cholesterol	303	6	7
7	cloud	108	4	2
8	cpu	209	6	1
9	echoMonths	131	6	3
10	fishcatch	158	5	2
11	housing	506	12	1
12	hungarian	294	6	7
13	lowbwt	189	2	7
14	pharynx	195	1	10
15	meta	528	19	2
16	pbc	418	10	8
17	quake	2178	3	0
18	servo	167	0	4
19	veteran	147	3	4

Table 6. Results of paired t-tests for naive Bayes for regression.

	NBR	CFS	Rlf
NBR	-	9	8
CFS	1	-	2
Rlf	2	4	-

ing in continuous class problems were used in the experiments: naive Bayes for regression (Frank, Trigg, Holmes & Witten, in press) (NBR), model trees (Wang & Witten, 1997) (M5′), and locally weighted regression (Atkeson, Moore & Schaal, 1997) (LWR). Naive Bayes for regression employs Gaussian kernel density functions to estimate conditional probabilities. Model trees are the counterpart of decision trees for regression tasks. They have the same structure as decision trees but employ linear functions at each leaf node in order to predict continuous values. Locally weighted regression is a technique that combines instance based learning and linear regression—a surface is fitted to neighbours of a target point using a distance weighted regression.

Tables 6, 7 and 8 summarize the results of feature selection with naive Bayes for regression, model trees, and locally weighted regression respectively[3]. The tables show how often the method in a column outperforms the method in a row. In this case, the measure of performance is the relative root mean squared error (RRSE). The relative root mean squared error of a method is its root mean squared error normalized

[3]Full results are listed in Table 10 at the end of the paper.

Table 7. Results of paired t-tests for M5′. Figures in braces show the number of times method in column produces fewer linear models than method in row.

	M5′	CFS	Rlf
M5′	-	5 {8}	3 {7}
CFS	4 {2}	-	3 {3}
Rlf	2 {2}	3 {2}	-

Table 8. Results of paired t-tests for LWR.

	LWR	CFS	Rlf
LWR	-	8	8
CFS	4	-	6
Rlf	7	4	-

by the root mean squared error of the sample mean[4]. Thus, lower values for RRSE are better and a method that performs worse than the mean has a relative root mean squared error of more than 100. Figure 2 shows the number of features selected by ReliefF and CFS as well as the number present in the full data set.

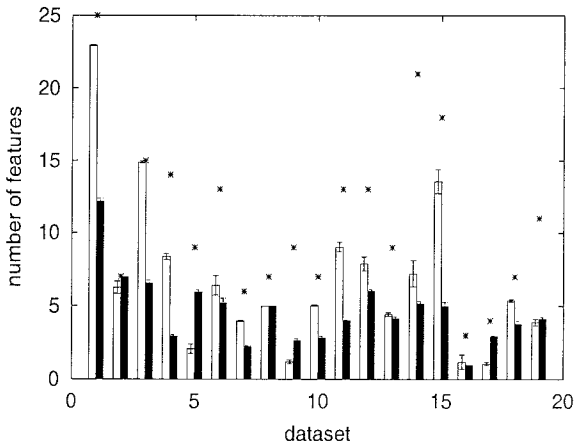

Figure 2. Average number of features selected by ReliefF (left) and CFS (right) on numeric class data sets. All differences between CFS and ReliefF are significant except for cloud (7), cpu (8), quake (16) and pharynx (19). Crosses show the original number of features in each data set.

Again—as in the discrete class case—CFS improves the accuracy of each learning scheme more often than not. For naive Bayes, accuracy is improved on nine data sets and decreased on only one. For M5′, accuracy is improved on five data sets and decreased on four. For locally weighted regression, accuracy is improved on eight data sets and decreased on four. On the remaining data sets for each learning scheme there is no significant change in performance. Like C4.5,

[4]The sample mean is computed from the test data.

M5′ produces a structure that can be interpreted. The number of linear models produced by M5′ is related to the size of the tree. From Table 7 it can be seen that CFS reduces the number of linear models produced by M5′ for eight out of the nineteen data sets and increases the number of linear models produced on two data sets.

ReliefF (with relevance threshold of 0) performs more feature selection on the numeric class data sets (as can be seen from Figure 2) than on the discrete class data sets, though not as much as CFS. ReliefF and CFS reduce the feature sets on average by 42% and 54% respectively. This indicates that there are more irrelevant attributes in these problems than in the discrete class data sets. From Tables 6, 7 and 8 (third column) it can be seen that, due to increased feature set reduction, there are more cases in which learning scheme's accuracy is changed significantly by ReliefF than there was for the discrete class data sets. For naive Bayes, accuracy is improved on eight data sets and decreased on two. For M5′, accuracy is improved on three data sets and decreased on two. ReliefF reduces the number of linear models produced by M5′ on seven data sets and increases the number of linear models on two data sets. For locally weighted regression, accuracy is improved on eight data sets and decreased on seven.

When CFS and ReliefF are compared (columns 2 and 3 in Tables 6, 7 and 8), it can be seen that the two methods perform comparably. For naive Bayes, CFS is better than ReliefF on four data sets and worse on two. For M5′, each method is better than the other on three data sets. For locally weighted regression, CFS is better than ReliefF on four data sets and worse on six.

6. Conclusions

This paper has presented a new correlation-based approach to feature selection (CFS) and demonstrated how it can be applied to both classification and regression problems for machine learning. CFS uses the features' predictive performances and intercorrelations to guide its search for a good subset of features. Experiments on discrete and continuous class data sets show that CFS can drastically reduce the dimensionality of data sets while maintaining or improving the performance of learning algorithms. Compared to ReliefF, CFS results in greater dimensionality reduction on both discrete and numeric class problems and performs comparably with respect to the accuracy of learning schemes using the reduced feature sets. Based on these results it can be concluded that CFS shows promise as a practical feature selector for common machine learning algorithms.

References

Almuallim, H. & Dietterich, T. G. (1991). Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence* (pp. 547–552). AAAI Press.

Atkeson, C. G., Moore, A. W. & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review, 11*, 11–73.

Blake, C., Keogh, E. & Merz, C. J. (1998). *UCI Repository of Machine Learning Data Bases.* Irvine, CA: University of California, Department of Information and Computer Science. [http://www.ics.uci.edu/~mlearn/MLRepository.html].

Fayyad, U. M. & Irani, K. B. (1993). Multi-interval discretisation of continuous-valued attributes. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 1022–1027). Morgan Kaufmann.

Frank, E., Trigg, L., Holmes, G. & Witten, I. H. (in press). Naive bayes for regression. *Machine Learning.*

Ghiselli, E. E. (1964). *Theory of psychological measurement.* McGraw-Hill.

Hall, M. A. (1998). *Correlation-based feature selection for machine learning.* PhD thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Kira, K. & Rendell, L. (1992). A practical approach to feature selection. In *Proceedings of the Ninth International Conference on Machine Learning* (pp. 249–256). Morgan Kaufmann.

Kohavi, R. & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97(1-2)*, 273–324.

Koller, D. & Sahami, M. (1996). Toward optimal feature selection. In *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 284–292). Morgan Kaufmann.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of relief. In *Proceedings of the Seventh European Conference on Machine Learning* (pp. 171–182). Springer-Verlag.

Press, W. H., Flannery, B. P., Teukolski, S. A. & Vetterling, W. T. (1988). *Numerical recipies in C.* Cambridge University Press.

Robnik-Šikonja, M. & Kononenko, I. (1997). An adaption of relief for attribute estimation in regression. In *Proceedings of the Fourteenth International Conference on Machine Learning* (pp. 296–304). Morgan Kaufmann.

Wang, Y. & Witten, I. H. (1997). Induction of model trees for predicting continuous classes. In *Proceedings of the poster papers of the European Conference on Machine Learning* (pp. 128–137). Prague, Czech Republic.

Table 9. Experimental results: discrete class data sets

Data Set	NB	CFS	Rlf0.01	C4.5	CFS	Rlf0.01	IBk	CFS	Rlf0.01
glass-2	62.36	63.85 ○	63.75 ○	75.73	80.63 ○	78.77	77.31	86.59 ○	82.41 ○
anneal	86.49	86.11	86.10 ●	98.68	98.78	98.77	99.04	98.69	98.85
breast-cancer	72.81	72.90	72.79	73.46	73.76	73.03	72.62	72.18	72.88
credit-g	74.88	74.42	74.39	70.70	72.67 ○	71.60 ○	73.42	73.28	73.09
diabetes	75.68	75.04	75.53	71.84	71.75	71.89	73.66	75.41 ○	74.54 ○
horse colic	81.04	86.46 ○	82.35 ○	85.35	85.09	85.10	83.64	86.32 ○	84.27
heart-c	83.63	84.04	83.38	74.56	75.83	75.24	82.04	81.84	82.10
heart-statlog	84.00	83.81	84.22	76.88	79.17 ○	77.13	81.70	79.89 ●	80.74 ●
ionosphere	82.61	88.63 ○	82.61	90.71	90.99	90.16	89.63	89.56	89.60
labor	94.57	86.98 ●	93.37	84.16	86.90 ○	84.16	90.13	82.30 ●	93.10
lymph	83.27	80.95 ●	83.08	75.34	74.28	74.84	80.78	80.92	81.67
segment	80.01	81.28 ○	83.32 ○	96.46	96.63	96.63	97.20	97.19	97.22
sick	92.70	94.48 ○	94.18 ○	98.73	97.49 ●	97.87 ●	96.09	96.12	96.47 ○
soybean	92.84	91.94 ●	92.72	91.89	90.66 ●	91.80	91.41	90.52 ●	91.37
vote	90.04	94.00 ○	90.04	96.37	95.52 ●	96.37	92.52	94.67 ○	92.25
zoo	95.58	94.59 ●	95.58	92.83	93.17	92.24	95.07	95.38	95.07

○,● statistically significant improvement or degradation

Table 10. Experimental results: numeric class data sets

Data Set	NBR	CFS	Rlf	M5′	CFS	Rlf	LWR	CFS	Rlf
autoHorse	39.17	40.88	39.78 ●	33.32	33.31	31.95	24.79	27.46	24.02
autoMpg	45.26	45.26	45.19	35.67	35.67	35.66	33.28	33.28	33.65
autoPrice	43.53	40.59	43.57	39.82	37.38 ○	38.67	40.69	40.17	40.76
bodyfat	26.08	13.61 ○	21.52 ○	11.15	10.72 ○	11.24	11.91	10.66 ○	11.22 ○
breastTumor	131.10	128.31 ○	122.63 ○	97.29	97.76	99.03 ●	103.06	102.19 ○	99.23 ○
cholesterol	114.45	109.91	112.71	101.62	100.15 ○	100.15	103.89	99.34 ○	100.45 ○
cloud	54.22	48.59 ○	50.72 ○	38.36	37.55	38.16	41.09	39.23 ○	40.00 ○
cpu	34.58	29.33 ○	29.33 ○	21.23	16.12 ○	16.35 ○	21.99	19.53 ○	19.53 ○
echoMonths	96.93	81.54 ○	75.34 ○	72.15	72.11	71.94	68.04	69.82	70.10 ●
fishcatch	32.16	29.15 ○	32.31	16.23	18.77 ●	16.45	22.45	30.69 ●	23.51 ●
housing	64.46	48.30 ○	58.30 ○	39.84	46.20 ●	41.11	39.93	48.29 ●	43.40 ●
hungarian	79.51	81.75	80.61	73.79	73.24	73.22	68.60	72.10 ●	70.74 ●
lowbwt	73.31	71.88	73.06	62.00	61.27	60.95 ○	62.66	63.36	61.75 ○
meta	152.87	141.41 ○	176.00	150.68	180.39	172.71	160.32	141.39 ○	135.76 ○
pbc	97.76	100.10	97.47	80.83	85.26 ●	82.28	81.38	86.60 ●	83.23 ●
pharynx	98.39	93.34	92.77 ○	105.87	71.53 ○	82.64 ○	118.05	74.25 ○	81.99 ○
quake	140.07	136.99 ○	137.76 ○	99.96	99.87	100.02	99.76	99.79	99.93 ●
servo	93.60	97.40 ●	104.84 ●	37.92	41.15 ●	65.15 ●	38.81	39.04	65.19 ●
veteran	94.21	95.20	104.28	90.53	90.86	90.66	97.77	93.66 ○	95.66

○,● statistically significant improvement or degradation

Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning

Mark A. Hall MHALL@CS.WAIKATO.AC.NZ
Department of Computer Science, University of Waikato, Hamilton, New Zealand

Abstract

Algorithms for feature selection fall into two broad categories: *wrappers* that use the learning algorithm itself to evaluate the usefulness of features and *filters* that evaluate features according to heuristics based on general characteristics of the data. For application to large databases, filters have proven to be more practical than wrappers because they are much faster. However, most existing filter algorithms only work with discrete classification problems. This paper describes a fast, correlation-based filter algorithm that can be applied to continuous and discrete problems. The algorithm often outperforms the well-known ReliefF attribute estimator when used as a preprocessing step for naive Bayes, instance-based learning, decision trees, locally weighted regression, and model trees. It performs more feature selection than ReliefF does—reducing the data dimensionality by fifty percent in most cases. Also, decision and model trees built from the preprocessed data are often significantly smaller.

1. 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, that 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. Wrapper methods (Kohavi & John, 1997) employ—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 data sets containing many features. Filter methods, on the other hand (Kohavi & John, 1997), operate 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 & Dietterich, 1991). Another method (Koller & 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 & Rendell, 1992).