Feature Selection for Discrete and Numeric Class Machine Learning

Mark A. Hall

Department of Computer Science
University of Waikato
Hamilton
New Zealand.

Abstract

Algorithms for feature selection fall into two broad categories: *wrappers* use the learning algorithm itself to evaluate the usefulness of features, while *filters* 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. Experiments using the new method as a preprocessing step for naive Bayes, instance-based learning, decision trees, locally weighted regression, and model trees show it to be an effective feature selector—it reduces the data in dimensionality by more than sixty percent in most cases without negatively affecting accuracy. Also, decision and model trees built from the pre-processed 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, 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* [9] 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* [9], 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 [1]. Another method [13] 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 [11][8]. 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. However, most filter algorithms work only on discrete class problems[1], 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

[1] A notable exception is RReliefF [22] which is an extension of Kira and Rendell's RELIEF [11] algorithm capable of working with continuous class problems.

applying suitable correlation measures. The next section describes the CFS algorithm. Section 3 describes the application of CFS to discrete class problems and presents experimental results of using CFS as a pre-processor for learning algorithms. Section 4 explains how the algorithm is extended to cope with continuous class problems and presents experimental results of using CFS on continuous class datasets. The last section summarises and discusses related work.

2 CFS: Correlation-based feature selection

2.1 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 [6], 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 variable 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 [6] 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, $\overline{r_{cf}}$ 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 standardised. 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.

2.2 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 [21] 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. Best first is also the preferred search strategy to use with the wrapper feature selector [12]. 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. Table 1 shows the best first search algorithm.

1.	Begin with the OPEN list containing the start state, the CLOSED list empty, and BEST←start state.
2.	Let s = arg max $e(x)$(get the state from OPEN with the highest evaluation).
3.	Remove s from OPEN and add to CLOSED.
4.	If $e(s) \geq e$(BEST), then BEST← s.
5.	For each child t of s that is not in the OPEN or CLOSED list, evaluate and add to OPEN.
6.	If BEST changed in the last set of expansions, goto 2.
7.	Return BEST.

Table 1: Best first search algorithm

2.3 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 [7]. 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.

3 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. Research on decision tree induction has provided a number of methods for estimating the quality of an attribute—that is, how predictive one attribute is of another [3][14][19][20]. For discrete class problems, CFS first discretises numeric features using the technique of Fayyad and Irani [4] and then uses a modified information gain measure (symmetrical uncertainty[18]) to estimate the degree of association between discrete 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* [20]. 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 normalised to ensure they are comparable and have the same effect. Symmetrical uncertainty [18] compensates for information gain's bias toward attributes with more values and normalises its value to the range $[0, 1]$:

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

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

3.1 Experiments (discrete class)

In order to evaluate the effectiveness of CFS as a global feature selector for common machine learning algorithms, experiments were performed using thirty-six standard datasets from the UCI collection [15]. The data sets and their characteristics are listed in Table 2. 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 6) and an instance-based learner (kNN[2]). The percentage of correct classifications, averaged over ten five-fold cross-validation runs, were calculated for each algorithm-dataset combination before and after feature selection by CFS. For each train-test split, a discretised copy of the training data was passed to CFS. The original, undiscretised train and test datasets had their dimensionality reduced with respect to the features selected by CFS before being passed to the learning algorithms. The same folds were used for each scheme.

Table 3 lists the results. A $\circ$ or $\bullet$ indicates whether results for CFS are significantly better or worse than when no feature selection is performed for each learning algorithm. In the C4.5 column, a $\surd$ or $\times$ indicates whether feature selection by CFS has significantly reduced or increased the *size* of the trees produced by C4.5 respectively. Throughout we speak of results being significantly different if the difference is statistically significant at the 5% level according to a paired two-sided t test, each pair of points consisting of the estimates obtained in one 5-fold cross validation run for before and after feature selection. Figure 1 shows how CFS compares with no feature selection for each learning algorithm. For each learner, the left bar shows the number of significant improvements and the right bar the number of significant reductions in accuracy.

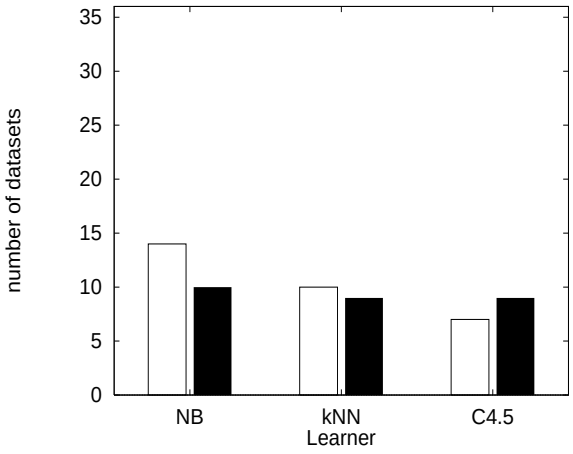

Figure 1: Results of paired t-tests ($p = 0.05$): left bar for each learner indicates how many times feature selection significantly improves performance; the right bar indicates how many times using the full feature set significantly outperforms feature selection.

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

Table 2: Discrete class datasets used in the experiments

	Dataset	Instances	Missing values (%)	Numeric attributes	Nominal attributes	Classes
1	anneal	898	0.0	6	32	5
2	audiology	226	2.0	0	69	24
3	australian	690	0.6	6	9	2
4	autos	205	1.1	15	10	6
5	balance-scale	625	0.0	4	0	3
6	breast-cancer	286	0.3	0	9	2
7	breast-w	699	0.3	9	0	2
8	dna-promoters	106	0.0	0	58	2
9	german	1000	0.0	7	13	2
10	glass (G2)	163	0.0	9	0	2
11	glass	214	0.0	9	0	6
12	heart-c	303	0.2	6	7	2
13	heart-h	294	20.4	6	7	2
14	heart-statlog	270	0.0	13	0	2
15	hepatitis	155	5.6	6	13	2
16	horse-colic	368	23.8	7	15	2
17	hypothyroid	3772	5.5	7	22	4
18	ionosphere	351	0.0	34	0	2
19	iris	150	0.0	4	0	3
20	kr-vs-kp	3196	0.0	0	36	2
21	labor	57	3.9	8	8	2
22	lymphography	148	0.0	3	15	4
23	mushroom	8124	1.4	9	22	2
24	pima-indians	768	0.0	8	0	2
25	primary-tumor	339	3.9	0	17	21
26	segment	2310	0.0	19	0	7
27	sick	3772	5.5	7	22	2
28	sonar	208	0.0	60	0	2
29	soybean	683	9.8	0	35	19
30	splice	3190	0.0	0	61	3
31	vehicle	846	0.0	18	0	4
32	vote	435	5.6	0	16	2
33	vote1	435	5.5	0	15	2
34	vowel	990	0.0	10	3	11
35	waveform-noise	5000	0.0	40	0	3
36	zoo	101	0.0	1	15	7

Table 3: Experimental results: discrete class datasets

Dataset	NB			kNN			C4.5		
	All	CFS		All	CFS		All	CFS	
anneal	86.80	85.86	•	98.72	96.60	•	98.78	97.49	•√
audiology	70.72	71.31		77.01	72.34	•	76.44	77.14	√
australian	77.02	74.61	•	85.79	85.61		84.61	84.16	√
autos	57.25	57.65		72.44	78.87	○	74.54	74.66	
balance-scale	90.08	89.67		89.86	89.41		78.35	78.14	
breast-cancer	73.12	72.53		73.05	73.26		72.16	70.82	•
breast-w	96.01	96.01		96.67	96.67		95.11	95.11	
dna-promoters	90.72	93.46	○	79.27	87.41	○	77.61	78.87	○√
german	74.85	72.96	•	73.36	73.43		71.08	72.36	○√
glass (G2)	62.36	65.51	○	76.46	80.95	○	76.12	80.40	○√
glass	48.10	48.70		68.61	74.00	○	68.31	68.59	√
heart-c	83.24	83.10		82.12	81.98		75.32	77.29	○√
heart-h	84.39	83.59	•	81.81	82.94		79.06	79.40	√
heart-statlog	84.19	83.78		81.48	80.48		76.15	78.45	○√
hepatitis	83.61	83.72		82.64	81.28		79.70	80.61	√
horse-colic	81.24	87.37	○	83.83	86.12	○	84.79	85.64	×
hypothyroid	95.04	94.53	•	93.24	92.76		99.48	97.60	•×
ionosphere	88.16	90.79	○	89.36	89.99		90.69	90.94	√
iris	95.33	95.73		95.33	95.07		94.54	94.67	√
kr-vs-kp	87.60	92.35	○	95.99	94.14	•	99.27	94.06	•√
labor	85.66	85.08		89.42	80.39	•	84.43	83.50	√
lymphography	83.43	79.13	•	81.25	80.02		74.12	74.30	√
mushroom	95.63	98.52	○	100.0	99.02	•	100.0	99.02	•√
pima-indians	75.34	76.22	○	73.99	76.06	○	71.38	71.41	√
primary-tumor	49.37	48.15	•	45.85	45.56		41.78	41.52	√
segment	78.99	85.27	○	96.95	96.94		96.28	96.47	
sick	92.03	96.12	○	96.04	96.63	○	98.76	97.43	•√
sonar	67.11	65.46	•	85.15	79.79	•	69.97	70.82	
soybean	92.56	91.94	•	90.90	90.45		90.49	89.51	•
splice	95.38	95.70	○	86.05	85.04	•	93.77	93.94	○√
vehicle	45.43	47.53	○	68.65	62.25	•	70.85	67.15	•×
vote	90.06	94.36	○	92.52	94.85	○	95.98	95.43	√
vote1	87.20	89.90	○	89.25	90.72	○	89.74	89.65	√
vowel	62.68	62.13		98.83	95.25	•	78.10	76.10	•
waveform-noise	80.05	80.19	○	82.96	85.59	○	74.65	76.76	○
zoo	95.06	92.89	•	95.03	94.25		93.01	92.72	√

○,•(√ ,×) statistically significant improvement or degradation

From Figure 1, it can be seen that feature selection improves the performance of naive Bayes on fourteen datasets and degrades performance on ten. For IB1, feature selection improves performance on ten datasets and degrades performance on nine. For C4.5, feature selection improves performance on seven datasets and degrades performance on nine. For all three algorithms feature selection safely removes features while not degrading the overall performance. Figure 2 shows the average number of features selected by CFS across the folds for each dataset. For seventy percent of the datasets CFS reduces the number features by more than half. Such a reduction in the number of features can have a dramatic affect on the training time of the learning algorithms.

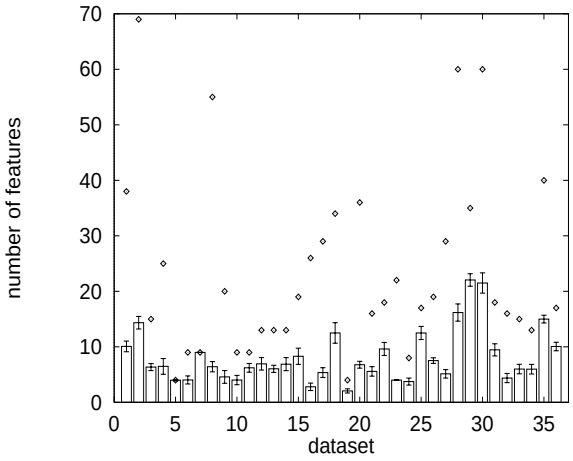

Figure 2: Average number of features selected by CFS with standard deviations. Dots show the original number of features in each dataset.

For C4.5, the size of the trees produced is also important because it has an influence on comprehensibility. From Table 3, it can be seen that feature selection by CFS significantly reduces the size of C4.5's trees for twenty four out of thirty six datasets and significantly increases the size of C4.5's trees on only three datasets.

4 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:

$$r_{XY} = \frac{\sum xy}{n\sigma_x\sigma_y}, \tag{6}$$

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

When one attribute is continuous and the other discrete, a weighted pearsons correlation is calculated as shown in Equation 7. 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.

$$r_{XY} = \sum_{i=1}^{k} p(X = x_i) r_{X_{bi}Y}, \tag{7}$$

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 8.

$$r_{XY} = \sum_{i=1}^{k} \sum_{j=1}^{l} p(X = x_i, Y = y_j) r_{X_{bi}Y_{bj}} \tag{8}$$

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

4.1 Experiments (continuous class)

Experiments on continuous class data follow a similar methodology to that described in Section 3.1 for the discrete case. The only difference is that features do not need to be discretised before being passed to CFS. Thirty five continuous class datasets [5] and their properties are listed in Table 4. Twenty of these datasets were used by Killpatrick and Cameron-Jones [10], seven are from the StatLib repository, and the remaining six were collected by Simonoff [23].

Three learning algorithms (close analogs of those used in the discrete class experiments) capable of learning in continuous class problems were used in the experiments: naive Bayes for regression [5] (NBR), M5′ [24], and locally weighted regression [2] (LWR). Naive Bayes for regression employes Gaussian kernal 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 state of the art technique that combines instance based learning and linear regression—a surface is fitted to neighbours of a target point using a distance weighted regression.

Table 5 summarises the relative root mean squared error of using the three learners with and without feature selection by CFS on the continuous class datasets. The relative root mean squared error of a method is its root mean squared error normalised by the root mean

Table 4: Continuous class datasets used in the experiments

	Dataset	Instances	Missing values (%)	Numeric attributes	Nominal attributes
1	auto93	93	0.7	16	6
2	autoHorse	205	1.1	17	8
3	autoMpg	398	0.2	4	3
4	autoPrice	159	0.0	15	0
5	baskball	96	0.0	4	0
6	bodyfat	252	0.0	14	0
7	bolts	40	0.0	7	0
8	breastTumor	286	0.3	1	8
9	cholesterol	303	0.1	6	7
10	cleveland	303	0.1	6	7
11	cloud	108	0.0	4	2
12	cpu	209	0.0	6	1
13	echoMonths	131	7.5	6	3
14	elusage	55	0.0	1	1
15	fishcatch	158	6.9	5	2
16	housing	506	0.0	12	1
17	hungarian	294	19.0	6	7
18	lowbwt	189	0.0	2	7
19	mbagrade	61	0.0	1	1
20	meta	528	4.3	19	2
21	pbc	418	15.6	10	8
22	pharynx	195	0.1	1	10
23	pollution	60	0.0	15	0
24	pwLinear	200	0.0	10	0
25	quake	2178	0.0	3	0
26	schlvote	38	0.4	4	1
27	sensory	576	0.0	0	11
28	servo	167	0.0	0	4
29	sleep	62	2.4	7	0
30	strike	625	0.0	5	1
31	veteran	137	0.0	3	4
32	vineyard	52	0.0	3	0

squared error of the sample mean[3]. Thus, a method that performs worse than the mean has a relative root mean squared error of more than 100. As before, a $\circ$ or $\bullet$ indicates whether results using CFS are significantly better or worse than when no feature selection is performed (all features are used) for each learning algorithm. For M5′, the number of linear models produced (there is one at each leaf) is also recorded. A $\surd$ or $\times$ indicates whether feature selection by CFS has significantly reduced or increased the number of linear models produced by M5′. Figure 3 shows how CFS compares with no feature selection for each learning algorithm. For each learner, the left bar shows the number of significant improvements (reductions of relative root mean squared error) and the right bar the number of significant degradations.

Table 5: Experimental results: continuous class datasets.

Dataset	NBR			LWR			M5′		
	All	CFS		All	CFS		All	CFS	
auto93	60.51	59.39		68.43	62.02	$\circ$	62.17	61.83	$\surd$
autoHorse	40.03	38.72		45.79	28.97	$\circ$	33.29	33.50	$\surd$
autoMpg	42.45	42.45		39.50	39.50		36.67	36.67	
autoPrice	41.68	37.47	$\circ$	38.79	35.76	$\circ$	39.55	37.40	$\circ\surd$
baskball	86.48	83.80	$\circ$	80.94	83.40	$\bullet$	81.16	84.40	$\bullet\surd$
bodyfat	26.46	14.85	$\circ$	12.73	11.62	$\circ$	12.24	23.25	
bolts	39.02	31.51	$\circ$	32.90	46.46	$\bullet$	30.35	49.67	$\bullet\times$
breastTumor	101.57	99.93	$\circ$	124.24	116.76	$\circ$	98.71	99.16	$\times$
cholesterol	104.02	104.33		120.89	103.44	$\circ$	100.46	98.67	$\circ\surd$
cleveland	76.13	72.97	$\circ$	84.84	76.82	$\circ$	74.54	73.39	$\surd$
cloud	52.45	47.58	$\circ$	39.66	39.25		38.87	38.11	$\surd$
cpu	37.05	32.90	$\circ$	19.43	11.54	$\circ$	20.21	20.10	$\surd$
echoMonths	78.56	71.69	$\circ$	76.33	70.10	$\circ$	67.90	68.16	$\surd$
elusage	53.58	47.53	$\circ$	55.56	48.14	$\circ$	46.54	44.52	$\circ\surd$
fishcatch	32.78	30.30	$\circ$	21.22	18.55	$\circ$	17.37	22.46	$\bullet\times$
housing	60.70	46.02	$\circ$	35.01	43.83	$\bullet$	45.17	47.42	
hungarian	72.54	68.11	$\circ$	76.72	75.21	$\circ$	77.38	75.20	$\circ\surd$
lowbwt	62.87	63.15		70.33	64.35	$\circ$	62.33	62.87	$\surd$
mbagrade	93.23	93.23		89.79	89.79		88.04	88.04	
meta	136.14	110.54	$\circ$	195.66	174.83		147.51	155.92	$\times$
pbc	86.43	88.50	$\bullet$	107.68	89.77	$\circ$	85.85	87.16	$\bullet\surd$
pharynx	82.32	75.37	$\circ$	104.47	77.87	$\circ$	72.32	72.16	$\surd$
pollution	84.77	77.91	$\circ$	73.62	65.13	$\circ$	71.08	68.52	$\surd$
pwLinear	52.19	49.21	$\circ$	38.72	35.33	$\circ$	32.54	32.47	$\surd$
quake	102.96	99.55	$\circ$	100.92	100.16	$\circ$	100.47	100.02	$\circ\surd$
schlvote	92.29	86.75	$\circ$	113.58	95.30	$\circ$	117.21	102.17	$\circ\surd$
sensory	92.59	91.09	$\circ$	117.03	98.84	$\circ$	87.10	87.13	$\surd$
servo	73.83	72.42	$\circ$	39.41	40.28		38.38	42.71	$\bullet\surd$
sleep	77.26	79.41	$\bullet$	76.19	78.84		96.57	81.15	$\surd$
strike	154.74	97.64	$\circ$	108.69	94.29	$\circ$	88.56	93.58	$\bullet\surd$
veteran	89.11	88.68	$\circ$	100.73	91.82	$\circ$	94.07	93.35	$\surd$
vineyard	66.76	62.59	$\circ$	65.21	63.24	$\circ$	72.69	68.61	$\circ\surd$

$\circ$,$\bullet$($\surd$, $\times$) statistically significant improvement or degradation

From Figure 3 it can be seen that CFS improves the accuracy of naive Bayes the most followed by LWR and then M5′. For naive Bayes, feature selection improves performance

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

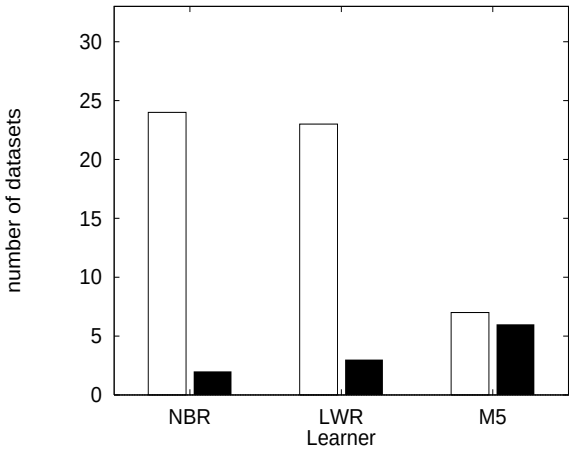

Figure 3: Results of paired t-tests ($p = 0.05$) for continuous class datasets: left bar for each learner indicates how many times feature selection significantly improves performance; the right bar indicates how many times using the full feature set significantly outperforms feature selection.

on twenty-four datasets and degrades performance on two. For locally weighted regression, feature selection improves performance on twenty-three datasets and degrades performance on three. For M5′, feature selection improves accuracy on seven datasets and degrades accuracy on six. Figure 4 shows the average number of features selected by CFS across the folds for each dataset. For sixty percent of the datasets CFS reduces the number of features by more than half.

Like C4.5, M5′ produces a structure that can be interpreted. The number of linear models produced by M5′ is related to the size of the tree as there is one model at each leaf of the tree. From Table 5, it can be seen that feature selection by CFS significantly reduces the number of linear models produced by M5′ for twenty four out of thirty two datasets and significantly increases the number of models produced on four datasets.

5 Related Work and Conclusions

While wrapper feature selection can be applied to regression problems with relative ease [16][17], few filter algorithms handle continuous class data. One notable exception is RReliefF (Regressional Relief) [22], which is an extension of Kira and Rendel's Relief [11] algorithm for classification problems. The Relief algorithms are quite different to CFS 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. Alternatively, cross validating the ranking can determine a cutoff point for a particular learner.

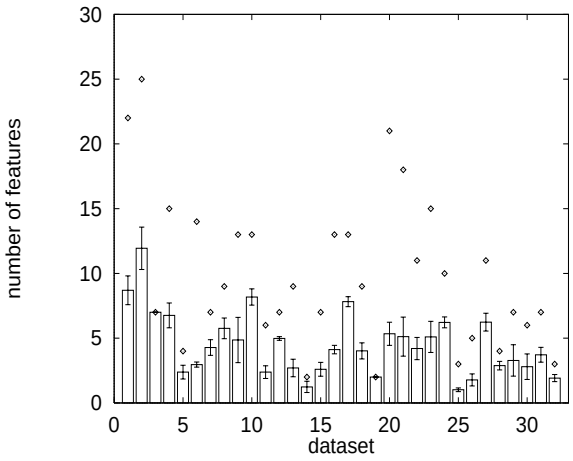

Figure 4: Average number of features selected by CFS with standard deviations on continuous class datasets. Dots show the original number of features in each dataset.

The instance-based nature of Relief makes it more computationally intensive than CFS. Relief works by randomly sampling training instances and using their nearest hits (instances of the same class) and nearest misses (instances of differing classes) to update the features' scores[4]. The complexity of Relief for m training instances, n attributes and s randomly sampled training instances is $O(s \times m \times n)$. Note that the value of s is normally much larger than the number of features. Increasing the value of s results in more reliable scores. For CFS computing the initial correlation matrix is $O(m((n^2 - n)/2))$ and the best first search is approximately quadratic in the number of features. When doing a forward search, the initial correlation matrix need not be pre-computed; correlations can be computed as required during the search.

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 datasets show that CFS can drastically reduce the dimensionality of datasets while maintaining or improving the performance of learning algorithms. Based on these results it can be concluded that CFS shows promise as a practical feature selector for common machine learning algorithms.

References

[1] Almuallim, H. and Dietterich, T. G. 1992. Efficient algorithms for identifying relevant

[4]See [22] for details on how this process is adapted to the continuous class case.

features. In *Proceedings of the Ninth Canadian Conference on Artificial Intelligence*, 38–45. Morgan Kaufmann.

[2] Atkeson, C. G., Moore, A. W. and Schaal, S. 1997. Locally weighted learning. AI Review 11: 11-73. Kluwer.

[3] Breiman, L. 1996. Technical note: Some properties of splitting criteria. *Machine Learning* 24: 41–47.

[4] 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.

[5] Frank, E., Trigg, L. E., Holmes, G. and Witten, I. H. 1998. Naive Bayes for regression. Working Paper 98/15. Hamilton, NZ: Waikato University, Department of Computer Science.

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

[7] Hall, M. A. 1998. Correlation-based Feature Selection for Machine Learning. Ph.D diss. Hamilton, NZ: Waikato University, Department of Computer Science.

[8] 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*.

[9] 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.

[10] Kilpatrick, D. and Cameron-Jones, M. 1988. Numeric prediction using instance-based learning with encoding length selection. In *Progress in Connectionist-Based Information Systems*. Singapore: Springer-Verlag.

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

[12] Kohavi, R. 1995. Wrappers for Performance Enhancement and Oblivious Decision Graphs. Ph.D diss. Menlo Park, CA: Stanford University, Deptartment of Computer Science.

[13] Koller, D. and Sahami, M. 1996. Towards optimal feature selection. In *Machine Learning: Proceedings of the Thirteenth International Conference* 294–292. Morgan Kaufmann.

[14] Kononenko, I. 1995. On biases in estimating multi-valued attributes. In *IJCAI95* 1034–1040.

[15] Merz, C. J. and Murphy, P. M. 1996. UCI *Repository of Machine Learning Databases* [http://www.ics.edu/~mlearn/MLRepository.html]. Irvine: University. of California., Deptartment of Information Science.

[16] Moore, A. W., Hill, D. J. and Johnson, M. P. 1992. An empirical investigation of brute force to choose features, smoothers and f unction approximators. In *Computational Learning Theory and Natural Learning Systems.* MIT Press.

[17] Moore, A. W. and Lee, M. S. 1994. Efficient algorithms for minimising cross validation error.

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

[19] Quinlan, J. R. 1989. Inferring decision trees using the minimum description length principle. *Information and Computation* 80:227–248.

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

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

[22] Robnik–Šikonja, M. and Kononenko, I. 1997. An adaption of Relief for attribute estimation in regression. In *Machine Learning: Proceedings of the Fourteenth International Conference* 296–304.

[23] Simonoff, J. S. 1996. *Smoothing Methods in Statistics.* New York: Springer-Verlag.

[24] Wang, Y. and 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.* Prague: University of Economics, Faculty of Informatics and Statistics.