AUCS/TR9910

Dimensionality Reduction through Correspondence Analysis

Terry R. Payne† & Peter Edwards‡

†The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh PA 15232, USA
Terry.Payne@cmu.edu

‡Department of Computing Science
University of Aberdeen
King's College, ABERDEEN, AB24 3UE, Scotland
pedwards@csd.abdn.ac.uk

Abstract

Many learning algorithms make an implicit assumption that all the attributes of the presented data are relevant to a learning task. However, several studies on attribute selection have demonstrated that this assumption rarely holds. In addition, for many supervised learning algorithms such as nearest neighbour algorithms, the inclusion of irrelevant attributes can result in a degradation in the classification accuracy of the learning algorithm. Whilst a number of different methods for attribute selection exist, many of these are only appropriate for datasets which contain a small number of attributes (e.g. < 20). This paper presents an alternative approach to attribute selection, which can be applied to datasets with a greater number of attributes. We present an evaluation of the approach which contrasts its performance with one other attribute selection technique.

Submitted to Journal of Artificial Intelligence Research

October 14, 1999

1 Introduction

The dimensionality of a supervised learning task can be characterised in many ways. A dataset contains a number of situations or instances, each of which contain several attributes and a class value. The attributes may be considered to be *predictor* (relevant) attributes, as they may be used to induce a classification hypothesis[1] (sometimes represented as a set of rules or a decision tree) which is later used to predict the class of an instance. However, other attributes may be considered as *irrelevant* attributes, as they contribute nothing to the classification task, and may even degrade the accuracy of the resulting classifications. The time taken to induce a concept description from a training set, and to predict the class of a new instance is dependent on both the learning algorithm used, and the number of attributes present (i.e. the number of dimensions used to describe the data). Techniques that reduce the number of dimensions required to represent large, complex domains are becoming more sought after, as the number of these domains increases. Example domains include: finance, marketing, fraud detection, etc.

Determining which of the attributes are relevant to the learning task (i.e. identifying attributes which predict the class value) is a central problem in machine learning. In the past, domain experts selected the attributes believed to be relevant to the learning task. However, in the absence of such background knowledge, automatic techniques are required to identify such attributes. Rule induction algorithms have been developed which use a variety of metrics as part of their learning bias to select relevant attributes when building decision trees. Such metrics include the Information Gain metric (Quinlan, 1986) or the Distance-Based Gain Ratio (De Mántaras, 1991). However, studies have shown that the biases used by rule induction algorithms to favour smaller numbers of attributes and smaller decision trees fail to find the minimal subset of attributes necessary to identify the concept (Almuallim & Dietterich 1991).

The inclusion of irrelevant attributes can reduce the performance of different learning techniques. Nearest neighbour algorithms are especially prone to the inclusion of these attributes, as the metrics used calculate an average similarity measure across all of the attributes (Aha 1992). In addition to this, the sample complexity (i.e. the number of instances required to learn a concept) grows exponentially with the number of irrelevant attributes (Langley & Iba 1993), indicating that simple nearest neighbour algorithms may not scale up well if irrelevant attributes are present. For these reasons, various weighting techniques have been investigated in an attempt to reduce the contribution of irrelevant attributes with nearest neighbour algorithms (Wettschereck et al., 1997; Payne, 1999).

Empirical studies have also indicated that the type of concept learned influences the rate at which sample complexity grows when irrelevant attributes are present. (Langley & Sage) explored changes in sample complexity for conjunctive and parity concepts. With conjunctive concepts, they found that whilst sample complexity increased linearly with an increase in irrelevant attributes for the rule induction algorithm, C4.5 (Quinlan 1993), it grew exponentially for a simple nearest neighbour algorithm. However, the sample complexity grew exponentially for both algorithms with the parity concept.

A *redundant-attribute* set occurs when two or more relevant attributes exist, such that each makes an equal contribution towards learning some concept (John, Kohavi, & Pfleger 1994). In general, only a single member of this redundant-attribute set is required when learning the concept. The inclusion of more than one member will not only increases the time taken to induce the concept description, but may place emphasis on the part of the concept description the attributes in the set represent, and thus reduce the influence of other relevant attributes (Langley & Sage 1994a). The remaining

[1] A number of learning algorithms do not explicitly perform any induction until the classification stage.

attributes in this set are sometimes described as *redundant*.[2]

A number of different attribute selection techniques have been proposed which attempt to identify and eliminate those attributes which are either irrelevant or redundant. However, the number of different possible attribute subsets is exponential (2^n) with respect to the number of original attributes (n). As a result, many of the techniques that perform a search through a space of different attribute subsets do not scale up well when the number of original attributes is large (e.g. $n > 20$). An alternative approach to attribute selection is presented here. The instances in a dataset are represented as vectors within an instance space. An approximation of this space is then found, and the vectors are projected into this lower dimensional space. This is achieved by using the geometric technique, *Correspondence Analysis* (Greenacre, 1984), to identify and approximate the lower dimensional space (or *sub-space*). This sub-space can then be used by a nearest neighbour learning algorithm to perform class predictions for new instances. Two learning algorithms have so far been developed that utilise this approach to dimensionality reduction: *CA* and *CACP*.

Section 2 presents a brief survey of the different attribute selection methods currently in use. The principals behind the dimensionality reduction approaches used by *CA* and *CACP* are discussed in Section 3, and the algorithms themselves are presented in Section 4. The evaluation of these two learning algorithms, and a description of the datasets used as part of the evaluation are presented in Section 5, and then discussed in Section 6. The paper concludes with Section 7.

2 Dimensionality Reduction and Attribute Selection

The task of dimensionality reduction and attribute selection has been one of the central problems in machine learning, and to date, many techniques have been proposed (Payne 1999). Such techniques are also required within large Information Retrieval (IR) systems (Salton & McGill 1983; Deerwester, Dumais, Furnas, Landauer, & Harshman 1990), and text categorisation systems (Edwards, Bayer, Green, & Payne 1996; Yang & Pedersen 1997). These systems use large indexes to search and retrieve text documents stored in a database or *corpus*. Dimensionality reduction techniques are often used to reduce the number of words (or terms) used to index the documents, and hence improve the rate at which documents are retrieved. This section describes some of the different approaches used to reduce the dimensionality of datasets by a number of machine learning algorithms and IR/text categorisation systems.

The attribute selection techniques used by machine learning algorithms can be grouped into two broad categories: those that belong to the *filter* model, where the selection technique is independent of the learning algorithm used to learn the concept hypothesis; and those that belong to the *wrapper* model, where the learning algorithm is integral to the selection mechanism (John, Kohavi, & Pfleger 1994). Both models perform a search within a space of attribute subsets to determine the optimal (or sub-optimal) subset for the classification task. In contrast to these models, a number of nearest neighbour techniques utilise weights to identify irrelevant attributes. As these approaches do not fall into either previous category, we have suggested a third model which we refer to as the *weighted* model (Payne & Edwards, 1996).

The attribute selection approaches used for IR/text categorisation tasks differ slightly from those used by machine learning algorithms in that they are often applied to domains with huge numbers of attributes (often $> 5\,000$). Whilst many of the approaches used by most IR/text categorisation

[2]Redundancy is not a property of a single attribute; rather the conditional property on the remaining attributes once one member has been selected from the *redundant-attribute* set.

systems fall into the filter model, an alternative approach has recently been investigated that makes use of geometric tools. This approach is described in greater detail in Section 3.

The next four subsections briefly describe the different attribute selection models used by various machine learning algorithms, and the dimensionality reduction techniques used by IR/text categorisation systems. They are followed by a discussion of the relative merits of each approach.

Filter Model

The *filter* model (Figure 1) utilises an independent search criterion and evaluation function to find the appropriate attribute subset. This subset is then used to generate a reduced dataset which in turn is presented to a learning algorithm. The evaluation function is used to determine whether or not the inclusion of an attribute will affect the classification performance of the learner. For example, the consistency measure used by (Almuallim & Dietterich) and (Liu & Setiono) determines whether or not the removal of an attribute will result in the creation of instances that have identical attribute values but different class values. This filter model, however, does not take into account the learning biases used by the final learning algorithm, and thus may not select the subset most suitable for that algorithm.

Figure 1: The Filter Model.

Table 1 lists some of the attribute selection systems that use the filter model. The *Search* column represents the type of search used. The forward selection and backward elimination searches start from either no attributes, or a full complement of attributes, and then search for solutions by greedily selecting and adding (eliminating) attributes to (from) the attribute subset. (Cardie) and (Kubat et al.) perform searches by presenting data which includes all the attributes to a decision tree algorithm, and selecting the attributes which appear in the resulting decision tree. The *Evaluation* column refers to the evaluation function used, and the final column, *Testing Alg.* refers to the learning algorithm that utilised the attribute subset within each study.

Wrapper Model

In the *wrapper* model (Figure 2), the attribute selection algorithm utilises the final learning algorithm as part of its evaluation function. The training data is normally divided into two partitions: a training partition, and an evaluation partition. The attribute subset is used to reduce both data partitions to contain only those attributes within the subset. The learning algorithm is trained using the instances in the training partition. The instances in the evaluation partition are then classified, and an overall predictive accuracy is generated for that attribute subset. This accuracy measure is then used to guide the search.

(Aha & Bankert) compared the filter and wrapper models and found that the wrapper model performed best when applied to data on cloud patterns. This supports the original hypothesis that attribute selection should take biases used by the final learning algorithm into account. Whilst this model may yield better results than the filter model, the time taken to evaluate each attribute subset visited during the search makes this approach infeasible for problems with very large numbers of attributes.

Authors (System)	Search	Evaluation	Testing Alg.
(Aha & Bankert) (BEAM)	Beam variants of forward & backward selection/elimination	Calinski-Harabasz separability index	IB1
(Almuallim & Dietterich) (FOCUS)	Breadth-first	Consistency	ID3
(Cardie)	Forward Selection	Information Gain[a]	kNN
(Kubat et al.)	Forward Selection	Information Gain[b]	Naive Bayes
(Liu & Setiono) (LVF)	Las Vegas (random sampling)	Consistency	ID3
(Singh & Provan) (Info-AS)	Forward selection	Maximise 1 of 3 information metrics	Bayesian Network

[a]The C4.5 learning algorithm is used to induce a decision tree to identify the attribute subset.
[b]The ID3 learning algorithm is used to induce a decision tree to identify the attribute subset.

Table 1: Comparison of different attribute selection studies (filter model).

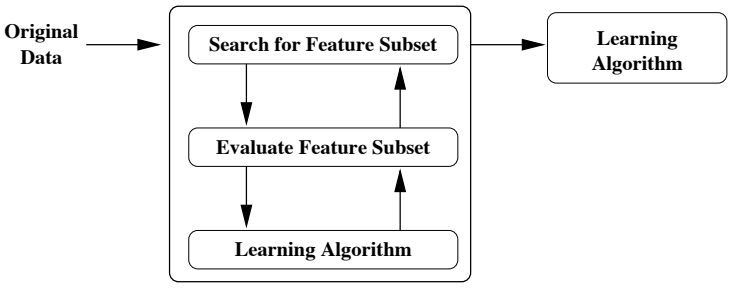

Figure 2: The Wrapper Model.

Table 2 lists some of the attribute selection systems that use the wrapper model. The *Search* column again represents the type of search used. The *Control* column refers to the control mechanism used when evaluating the attribute subsets. The last column refers to the learning algorithm used to evaluate each attribute subset.

Weighted Model

A number of learning algorithms make use of real-valued attribute vectors to weight attributes based on their past performance. A weighted attribute vector is generated, which initially gives each attribute an equal weight. The training set is then evaluated using a leave-one-out cross validation. After each instance has been evaluated, the weights are adjusted according to whether or not the classification was correct. We refer to this model as the *weighted* model (Figure 3).

The intuition behind this model is that irrelevant attributes will contribute very little overall to the classification task. The weighting strategies used normally reward attributes if they are responsible for correct predictions, and penalise them if they are responsible for incorrect ones. Thus, the contribution

Authors (System)	Search	Control	Learning Alg.
(Aha & Bankert) (BEAM)	Beam variants of forward & backward selection/elimination	leave-one-out cross validation	IB1
(Bala et al.) (GA-ID3)	Genetic algorithm	2 cross validated fixed partition tests	C4.5
(Caruana & Freitag)	Forward, backward & stepwise selection/ elimination variants	fixed size train/ evaluation partitions	ID3/C4.5
(Cherkauer & Shavlik) (SET-GEN)	Genetic algorithm	k-fold cross validation	C4.5
(John et al.)	Forward selection & backward elimination	k-fold cross validation	C4.5
(Kohavi) (BFS)	Best first search	k-fold cross validation	C4.5
(Kohavi) (IDTM)	Best first search	leave-one-out cross validation	Decision trees
(Langley & Sage) (Selective Bayes)	Forward selection	accuracy measure across training set	Naive Bayes
(Langley & Sage) (OBLIVION)	Backward elimination	k-fold cross validation	Oblivious decision trees[a]
(Langley) (CONDET)	Forward selection	leave-one-out cross validation	Determination table[b]
(Moore & Lee) (RACE)	Forward & backward selection, and schemata search	leave-one-out cross validation	1-NN
(Richeldi & Lanzi) (ADHOC)	Genetic algorithm	k-fold cross validation	C4.5
(Salzberg) (CSS)	Combined stepwise selection	fixed size train/ evaluation partitions	1-NN and EACH
(Singh & Provan) (K2-AS)	Forward selection	fixed size train/ evaluation partitions	Bayesian networks
(Skalak) (RMHC-PF1)	Random mutation search	k-fold cross validation	1-NN
(Terano & Ishino) (SIBILE)	Genetic algorithm	-	C4.5[c]
(Vafaie & De Jong)	Genetic algorithm	fixed size train/ evaluation partitions	AQ15

[a] "...equivalent to a nearest neighbour scheme that ignores some attributes..." (Langley & Sage, 1994b)

[b] A Determination table classifies unseen instances in a similar manner to a nearest neighbour scheme except that if no identical match can be found then the majority class is used.

[c] Decision trees induced by C4.5 are evaluated and rated by a domain expert.

Table 2: Comparison of different attribute selection studies (wrapper model).

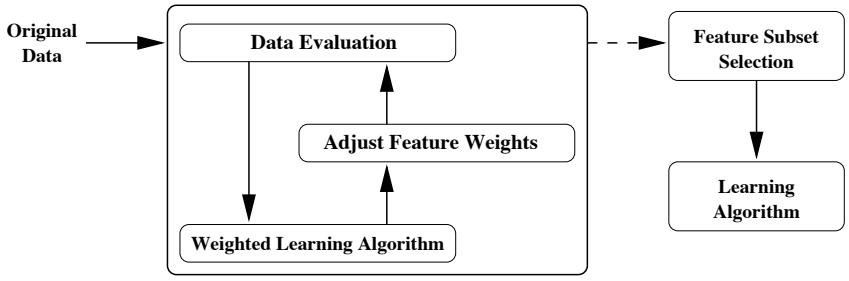

Figure 3: The Weighted Model.

of irrelevant attributes to the classification task falls as the contribution of other attributes rises. Those attributes that make a small contribution can then be eliminated.

Authors (System)	Selection	Evaluation	Testing Alg.
(Aha) (IB4)	Weighted attributes	Adjust weights wrt accurate or inaccurate predictions	Weighted nearest neighbour
(Kira & Rendell) (RELIEF)	Threshold selection	Adjust weights wrt closest +ve/-ve neighbours	ID3
(Kononenko) (RELIEF-extensions)	Threshold selection	Adjust weights wrt closest neighbours from each class	ID3
(Littlestone) (WINNOW)	Weighted attributes	Adjust weights wrt inaccurate predictions	Linear threshold classifier
(Payne & Edwards) (OMVW)	Weighted attributes	Weights based on class conditional probability	Weighted nearest neighbour
(Salzberg) (EACH)	Weighted attributes	Reduce weights wrt inaccurate predictions	Weighted nearest hyperrectangle

Table 3: Comparison of different attribute selection studies (weighted model).

Table 3 lists some of the systems that employ the weighted model. The *Selection* column indicates whether the attribute weights are used as part of the final learning algorithm or used to select an attribute subset. The *Evaluation* column refers to the way attribute weights are updated during the evaluation phase. The last column refers either to the type of algorithm used during attribute selection, or in the case of RELIEF (Kira & Rendell, 1992) and the extensions to RELIEF (Kononenko, 1994), to the learning algorithm used once the attributes have been selected.

IR/Text categorisation approaches

Dimensionality reduction techniques have been used by a number of IR systems to reduce the number of terms used to index the documents, resulting in an improvement in the rate at which documents are retrieved. These techniques have also been applied to the problem of reducing the number of terms presented to learning algorithms for text categorisation problems (Edwards, Bayer, Green, & Payne 1996; Yang & Pedersen 1997). (Moulinier) presents a framework for text categorisation, which includes a dimensionality reduction or attribute selection stage between the initial representation, that of textual data, and the final representation presented to the learning algorithm. Whilst some

studies have omitted this stage (Creecy, Masand, Smith, & Waltz 1992), the number of unique terms (typically in the region of tens or hundreds of thousands) is prohibitively high for most machine learning algorithms. For this reason, several different techniques have been developed specifically to reduce the dimensionality of the final data representation.

Technique	Study	Learning Algorithms
Information Gain	(Lewis & Ringuette)	PropBayes and DT-min10
	(Armstrong et al.)	Winnow, Wordstat and a Rocchio-based NN
	(Moulinier)	ID3, Charade, NN and Naive Bayes
	(Moulinier)	Ripper and Scar
	(Yang & Pedersen)	k-NN and a linear least squares fit mapping
Mutual Information	(Weiner et al.)	Neural Network classifier
	(Yang & Pedersen)	k-NN and a linear least squares fit mapping
χ^2 Statistic	(Schütze et al.)	Logistic regression, linear discriminant analysis and a Neural Network classifier
	(Yang & Pedersen)	k-NN and a linear least squares fit mapping
Frequency Measure	(Lang)	Rocchio-based NN and MDL
	(Edwards et al.)	IBPL and C4.5
	(Payne & Edwards)	IBPL and CN2
	(Yang & Pedersen)	k-NN and a linear least squares fit mapping

Table 4: A sample of different dimensionality reduction methods for text categorisation.

The techniques used by many text categorisation systems are similar to those categorised by the *filter* model. Table 4 lists a sample of the different evaluation methods used by various systems. *Latent Semantic Indexing* (LSI) (Deerwester et al., 1990; Schütze et al., 1995; Weiner et al., 1995) is an alternative approach for reducing the number of dimensions used to represent documents in a number of IR systems. Unlike the techniques presented in Table 4 which select a subset of terms to use when representing each document, LSI utilises an orthogonal decomposition technique to determine a smaller numeric representation for each document. A corpus is represented as a *term* $\times$ *document* matrix, where each row corresponds to a document, and each column to one of the terms appearing with the corpus. Thus, each document (i.e. row vector) is expressed as a point within some geometric space. An orthogonal decomposition technique is then applied to this matrix, resulting in a set of decomposed matrices that describe this space and the points within it. The space can then be approximated (by approximating the decomposed matrices) resulting in a lower dimensional representation of the points in the approximated space. This approach is described in greater detail in Section 3.

Singular Value Decomposition (SVD) (Press, 1992; Greenacre, 1984, Appx A.) is normally used to perform the matrix decomposition, although other orthogonal decomposition approaches, such as the ULV decomposition (Berry & Fierro 1996), can be used to replace SVD for this task. Studies have demonstrated that a significant reduction in dimensionality can be achieved when used within IR systems; for example from 5000-7000 terms to about 100 dimensions (Deerwester et al., 1990). SVD has also been successfully applied to the problem of reducing the dimensionality of protein sequence data for presentation to neural networks (Wu, Berry, Shivakumar, & McLarty 1995). The size of the input vectors presented to a backward propagation neural network was reduced from 9696 to 100. In addition to this, the predictive accuracy of the neural network improved when SVD was used.

Discussion

The various models described above differ in the way they reduce dimensionality. The filter and wrapper models perform a search through a space of possible attribute subsets. The number of states within this space is exponential; if there are n attributes in the original dataset, then there are a total of 2^n possible states in the search space. This exponential rise means that exhaustive, optimal searches are infeasible for all but simple problems involving few attributes. Therefore, most systems perform greedy or stochastic searches.

Several studies have shown that the wrapper model can identify better attribute sets, when compared with the filter model (Aha & Bankert 1994; John, Kohavi, & Pfleger 1994). However, induction is performed at every search state visited. This can result in an exponential rise in the time taken for an exhaustive search to locate an optimal subset of attributes. The number of instances, i, in the training set and the control mechanism used to evaluate each state will also influence the length of time taken to determine the final attribute subset. Many systems utilise a *k-fold cross validation* approach (Kohavi, 1995a) when testing each attribute state to reduce the number of times induction is performed (from i to k). Whilst this may reduce the time taken to generate the final attribute subset, this subset will be dependent on the order in which the training data is presented to the wrapper[3].

The weighted model differs from the other models in that no explicit search is performed. Instead, a weight vector is modified as each of the training instances is classified. Some weighted approaches utilise the weighted vector when evaluating each training instance, and hence the final weighted vector may be dependent on the order in which the training instances are evaluated. As this model generally evaluates the training data using a single leave-one-out cross validation approach, the time taken to generate the final attribute subset is linearly dependent on the number of instances in the training set and the total number of attributes used to describe the training set. For this reason, this model is more suited to performing attribute selection when the number of attributes is large.

The weighted model may also be unsuitable for numeric data. This model rewards attributes that are responsible for correct classifications, and penalises attributes that are responsible for incorrect classifications. For symbolic data, it is relatively simple to determine whether or not an attribute is responsible for a classification (if the overlap metric is used). In this case, the value (for each attribute) of the nearest neighbour is either equal or different to the corresponding value of the instance that is being classified. However, if a numeric distance metric is used, then the distance between these two attribute values can occur somewhere within a continuous range, and therefore correlation or regression techniques may be required to determine whether or not the attribute is responsible for the classification.

Although the filter and wrapper models involve a search through a large state space, the filter model generally takes less time to find a sub-optimal attribute subset than the wrapper model. This is due to the length of time taken to evaluate each attribute subset. However, if the number of attributes is very large, as in the case of IR and text categorisation problems, then performing any form of search becomes impractical. For this reason, a number of attribute selection approaches used by IR and text categorisation systems make an assumption that the relevance of each term is independent of the others. Although this assumption is counter-intuitive, it allows the relevance of each term (i.e. each attribute) to be assessed independently of the other terms. This reduces the number of possible evaluations that may be performed from 2^n to n, where n is the number of terms

[3] A deterministic approach should be used when partitioning the data into folds, so that the evaluated search states can be compared. If a stochastic approach is used, then each evaluated search state will not only be dependent of the attribute subset, but on the way in which the folds were partitioned.

extracted from the corpus of documents.

Latent Semantic Indexing (LSI) has been demonstrated to both improve performance of IR and text categorisation systems, and reduce the number of dimensions (i.e. attributes) required. This technique has also been used to reduce the dimensionality of data for other problems, such as within the task of protein sequence classification (Wu, Berry, Shivakumar, & McLarty 1995). However, such studies have demonstrated that LSI and the principals behind this method work for specific problems, but have not investigated the applicability of LSI to a broader range of classification tasks. For this reason, we have investigated a similar technique, based on *Correspondence Analysis* (Greenacre 1984), and have embedded two variations of this technique within a nearest neighbour learning algorithm. The resulting algorithms have been applied to a variety of classification problems found in the UCI Machine Learning Database Repository (Murphy & Aha 1994), and to artificial data (described in Section 5).

3 Subspace Approximation through Correspondence Analysis

In the previous section, a technique known as Latent Semantic Indexing (Deerwester et al., 1990) utilised various geometric techniques to reduce the dimensionality of data utilised by an Information Retrieval system. A related technique, known as *Correspondence Analysis* (Greenacre, 1984), is used to graphically display points within a dataset as a two or three dimensional data plot. This section summarises the theory behind Correspondence Analysis, and describes how *singular value decomposition* (SVD) can be used to reduce the number of dimensions required to represent the data points.

Correspondence analysis reduces the number of dimensions required to represent instances in a dataset. It achieves this by identifying an approximation of the Euclidean space that contains the instances (which are represented as vectors). This approximation is used to project the vectors from a J-dimensional instance space into a K-dimensional subspace, where J is the number of attributes of the dataset, and consequently the number of components of the vectors, and K (where $K \leq J$) is the rank of the approximated space.

The way in which the space can be approximated and the vectors projected may be explained by means of an example. Consider the values given in Table 5. Each value represents either the annual Profit or Debt of a fictitious company over the span of three consecutive years. These values can be represented as three vectors corresponding to Profit and Debt tuples for each year. They can then be plotted within a two-dimensional space. These three vectors, $\mathbf{y1}$, $\mathbf{y2}$ & $\mathbf{y3}$ are illustrated in Figure 4.

	Profit	Debt
Year 1	140	1580
Year 2	290	1310
Year 3	470	410

Table 5: Artificial data representing fictitious annual Profit and Debt figures over three years.

Any vector in a J dimensional space can be expressed as a linear combination of a *basis* and J scalar coefficients. A *basis* is a set of J linearly independent vectors that characterises a space. A particular basis is the canonical basis; it is this that characterises Euclidean space. For example, the canonical basis for the two dimensional Euclidean space $E^{(2)}$ consists of two basis vectors, $\mathbf{e_1}$ and $\mathbf{e_2}$, and can be expressed as:

$$E^{(2)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} \end{bmatrix}, \mathbf{e_1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{e_2} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Hence, the vector **y2** can be expressed as the linear combination of the canonical basis for a two-dimensional space, and of two scalar coefficients 290 and 1310, i.e.

$$\mathbf{y2} = \begin{bmatrix} 290 \\ 1310 \end{bmatrix} = 290 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 1310 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 290\mathbf{e_1} + 1310\mathbf{e_2}$$

In Figure 4 we can see that the three vectors **y1**, ..., **y3** exist close to the straight line **r**. It is possible to express each of the vectors **y1**, ..., **y3** as the combination of three new vectors: a vector from the origin to a fixed point (the *centroid*) on the line **r**; a vector along this line from the centroid; and a vector orthogonal to the line **r.** For example, it is possible to express the vector **y2** as a combination of the three vectors $\bar{\mathbf{y}}$, **b** and **c** (Figure 5), where $\bar{\mathbf{y}} = [\ 300 \quad 1100\ ]^{\mathsf{T}}$ is the vector[4] from the origin to the centroid, $\mathbf{b} = [\ 40 \quad -120\ ]^{\mathsf{T}}$ is the vector running along the line **r** (from top left to bottom right in Figure 5), and $\mathbf{c} = [\ 30 \quad 90\ ]^{\mathsf{T}}$ is a vector which is orthogonal to the line **r.**

$$\begin{aligned} \mathbf{y2} &= \bar{\mathbf{y}} + (-\mathbf{b}) + \mathbf{c} \\ &= [\ 300 \quad 1100\ ]^{\mathsf{T}} + \left(-[\ 40 \quad -120\ ]^{\mathsf{T}}\right) + [\ 30 \quad 90\ ]^{\mathsf{T}} \\ &= (300\mathbf{e_1} + 1100\mathbf{e_2}) + (-40\mathbf{e_1} + 120\mathbf{e_2}) + (30\mathbf{e_1} + 90\mathbf{e_2}) \\ &= 290\mathbf{e_1} + 1310\mathbf{e_2} \end{aligned}$$

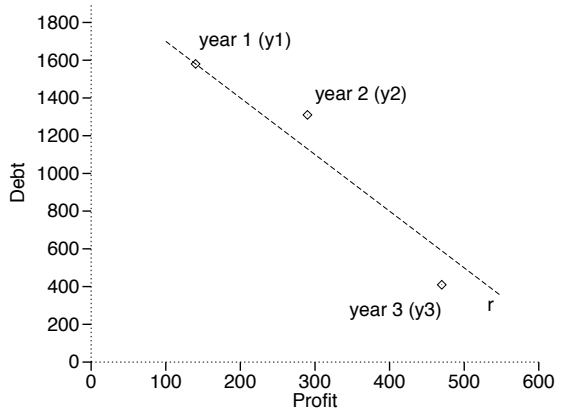

Figure 4: The three vectors, **y1**, ..., **y3**, plotted as points within a two-dimensional space. Note that each of the vectors lies close to the straight line **r.**

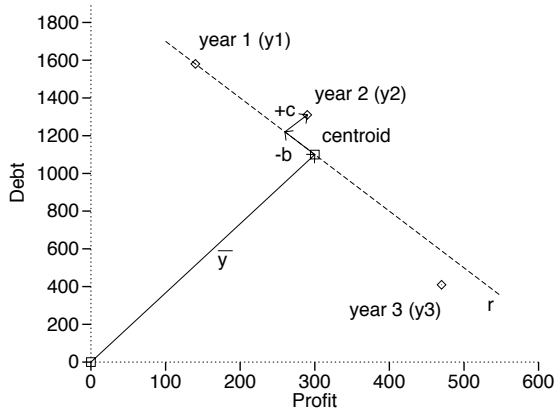

Figure 5: Each of the three vectors can be expressed as a combination of three new vectors: $\bar{\mathbf{y}}$, a vector running along the line **r**, and a vector orthogonal to the line **r.**

[4]The $^{\mathsf{T}}$ symbol is used here to refer to the transpose of each vector.

The coefficients of the vectors **b** and **c** (in conjunction with the centroid vector $\bar{\mathbf{y}}$) used to express the vector **y2** are -1 and 1 respectively. By varying these coefficients, it is possible to express every instance in the dataset using the centroid vector and these two vectors. The coefficient of the centroid vector $\bar{\mathbf{y}}$ is constant for all the instances in the dataset. Hence, the two vectors **b** and **c** can be used to construct the *basis* of a new space that has been translated from the origin by the vector $\bar{\mathbf{y}}$. Each of the instances in Table 5 can be expressed as a combination of the vector $\bar{\mathbf{y}}$ and a vector who's components are the coefficients of the basis vectors **b** and **c**. Figure 6 illustrates the three instances mapped onto this new two dimensional space.

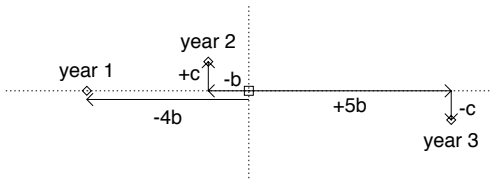

Figure 6: The three vectors **y1**, ..., **y3**, mapped onto a new, 2-dimensional space. The new space is characterised by the two basis vectors $\mathbf{b} = [40 \quad -120]^{\mathsf{T}}$ and $\mathbf{c} = [30 \quad 90]^{\mathsf{T}}$

Singular Value Decomposition (SVD) can be used to identify the basis {**b**, **c**}. The advantage of using this technique is that it is then possible to approximate the space that this new basis characterises using fewer dimensions. For example, the space illustrated in Figure 6 can be approximated by a space of only one dimension, characterised by the single basis vector **b**. The following subsection describes how SVD can be used to find a *best-fit* subspace within an n-dimensional space, and illustrates how the best *lower-rank* approximation of the subspace is found.

Determining the centroid and new basis

If SVD is used to identify a space that best fits the instances in the dataset, the matrices it generates can be used to calculate a good approximation to this space. This subsection shows how the centroid vector can be determined, and describes how SVD can be applied to the task of determining a new basis and finding an approximation of the space the basis characterises.

The space characterised by the basis {**b**, **c**} (see above), was translated from the origin by the centroid vector $\bar{\mathbf{y}}$. The centroid vector is used for two reasons: it is relatively simple to compute, and it is guaranteed to exist within a space containing the instances (Deerwester et al., 1990). Whilst any vector can be used to perform this translation, it is difficult to identify such vectors if the basis of the new space is unknown.

The centroid vector is calculated by determining the mean vector for all the vectors that represent the instances in the dataset. For example, the centroid vector $\bar{\mathbf{y}}$ for the three vectors **y1**, ..., **y3** can be calculated as follows:

$$\begin{aligned}
\bar{\mathbf{y}} &= \tfrac{1}{3}(\mathbf{y1} + \mathbf{y2} + \mathbf{y3}) \\
&= \tfrac{1}{3}(140\mathbf{e_1} + 1580\mathbf{e_2} + 290\mathbf{e_1} + 1310\mathbf{e_2} + 470\mathbf{e_1} + 410\mathbf{e_2}) \\
&= \tfrac{1}{3}(900\mathbf{e_1} + 3300\mathbf{e_2}) \\
&= 300\mathbf{e_1} + 1100\mathbf{e_2}
\end{aligned}$$

To simplify the process of identifying the new basis, the vectors representing the data can be translated by the centroid vector. This has the advantage of creating a new set of vectors whose centroid is located at the origin. Hence, these vectors can be expressed as coefficients of the new basis, without the need for an initial translation. For example, a new vector, **z2**, can be found by translating the vector **y2** with the centroid vector $\bar{\mathbf{y}}$. It can also be expressed with respect to the basis {**b**, **c**}, i.e.

$$\mathbf{z2} \;=\; \mathbf{y2} - \bar{\mathbf{y}} \;=\; -1\mathbf{b} + \mathbf{c}$$

Singular Value Decomposition is often used for solving most linear least squares problems, and for performing eigenvalue/eigenvector decomposition. However, it can also be used to construct an orthonormal basis of a best-fit space. The detailed theory behind SVD is not discussed here; for a more thorough discussion see (Greenacre, Appendix A.). The SVD of a matrix $\mathbf{X}$ of I rows and J columns, and of rank N (see below) can be expressed as:

$$\underset{I \times J}{\mathbf{X}} \;=\; \underset{I \times N}{\mathbf{L}} \quad \underset{N \times N}{\mathbf{D}} \quad \underset{N \times J}{\mathbf{R}^{\mathsf{T}}}$$

where $\mathbf{L}^{\mathsf{T}}\mathbf{L} = \mathbf{R}^{\mathsf{T}}\mathbf{R} = \mathbf{I}$ (the identity matrix).

The N orthonormal vectors of $\mathbf{L}$, called the left singular vectors, form an orthonormal basis for the columns of $\mathbf{X}$. Similarly, the N orthonormal vectors of $\mathbf{R}$, called the right singular vectors, form an orthonormal basis for the rows of $\mathbf{X}$. The diagonal matrix $\mathbf{D}$ contains the N singular values of $\mathbf{X}$, where the elements of $\mathbf{D}$: $d_1 \geq d_2 \geq \cdots \geq d_N > 0$. Figure 7 illustrates the singular value decomposition from an $I \times J$ matrix.

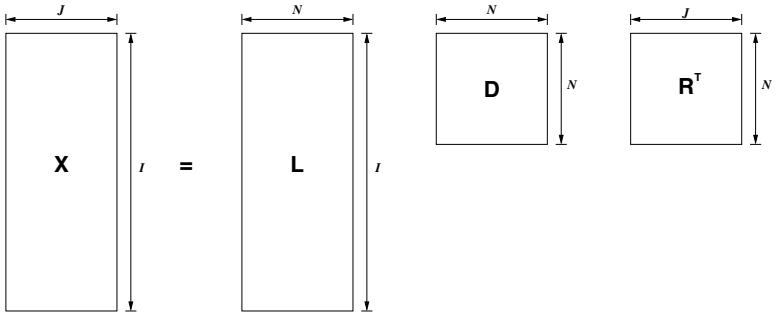

Figure 7: A Singular Value Decomposition of an $I \times J$ matrix.

The matrix $\mathbf{X}$ can be constructed such that each row corresponds to each of the translated vectors representing each instance in the dataset, and each column corresponds to one of the attributes of the dataset. For example, a matrix constructed from the example above would consist of three rows and two columns. Each row would correspond to the translated vectors **z1**, ..., **z3** (where $\mathbf{z1} = \mathbf{y1} - \bar{\mathbf{y}}$, etc.).

As stated above, the matrix $\mathbf{R}$ forms an orthonormal basis for the rows of $\mathbf{X}$. It is this matrix which characterises the best-fit space for the I instances in matrix $\mathbf{X}$. The rows of matrix $\mathbf{X}$ (corresponding to the instances in the dataset) can be projected into the new space by multiplying this matrix with the basis $\mathbf{R}$. The number of dimensions of the space characterised by the basis $\mathbf{R}$ will be equal to the *rank* (Fraleigh & Beauregard 1995) of the original matrix $\mathbf{X}$. The rank N of this matrix will be equal to or less than I or J, whichever is the smaller, i.e. $N \leq \min(I, J)$.

An advantage of using SVD is that the singular values in the diagonal matrix $\mathbf{D}$ can be used to determine which of the N columns of $\mathbf{R}$ can be omitted, and hence result in a lower rank approximation of the space characterised by $\mathbf{R}$. A basis that contains the columns of $\mathbf{R}$ corresponding[5] to the largest singular values in $\mathbf{D}$ will better approximate the space (characterised by $\mathbf{R}$) than one which contains columns corresponding to the smallest singular values. Therefore, if the basis $\mathbf{R}_{(K)}$ for a K dimensional approximation of an N dimensional space is required (where $0 < K \leq N$), then the K columns of $\mathbf{R}$ corresponding to the K largest singular values of $\mathbf{D}$ should be included in the basis for this approximation. This process is called *low rank approximation* (Greenacre, 1984).

To conclude, given a matrix $\mathbf{Y}$ representing I instances and J attributes (i.e. the rows of $\mathbf{Y}$ are the vectors $\mathbf{y}_1^\mathsf{T}, \mathbf{y}_2^\mathsf{T}, \ldots, \mathbf{y}_I^\mathsf{T}$), a centroid vector $\bar{\mathbf{y}}$ can be calculated from $\mathbf{Y}$. If a translated matrix $\mathbf{X}$ is defined as the matrix $\mathbf{Y} - 1\bar{\mathbf{y}}^\mathsf{T}$, then $\mathbf{X}$ can be decomposed into the three matrices $\mathbf{L}$, $\mathbf{D}$ and $\mathbf{R}^\mathsf{T}$. A $\mathbf{K}$ dimensional low rank approximation of $\mathbf{R}$ can be constructed, such that the row vectors[6] $r_1, r_2, \ldots, r_K$ of $\mathbf{R}^\mathsf{T}{}_{(K)}$ correspond to the $d_1, d_2, \ldots, d_K$ largest singular values of $\mathbf{D}$. Thus, the basis $\mathbf{R}_{(K)}$ characterises an approximated space of rank K. Once this basis has been determined, it can be used to project instances (represented as vectors) into the new space. For example, if there is a new vector $\mathbf{y}_i$ consisting of J components (i.e. dimensions), then we can find its projection, $\mathbf{f}_i$ which consists of K components in the space characterised by the basis $\mathbf{R}_{(K)}$ (with respect to the centroid vector $\bar{\mathbf{y}}$) as follows:

First, translate the vector $\mathbf{y}_i$ by the centroid vector:

$$\mathbf{x}_i^\mathsf{T} = \mathbf{y}_i^\mathsf{T} - 1\bar{\mathbf{y}}^\mathsf{T}$$

Then project the translated vector $\mathbf{x}_i$ into the new space, by finding the product of $\mathbf{x}_i$ and the basis $\mathbf{R}_{(K)}$:

$$\underset{1 \times K}{\mathbf{f}_i} = \underset{1 \times J}{\mathbf{x}_i} \quad \underset{J \times K}{\mathbf{R}_{(K)}}$$

It is possible to determine how good the approximated space is, by calculating the *variation* (as a percentage) of the space. The *total variation* of a space characterised by the basis $\mathbf{R}$ is given by $\sum_{k=1}^{N} d_k^2$, i.e. the sum of the squared singular values $d_1, d_2, \ldots, d_K$. Thus, the variation of the K-dimensional approximated space can be found by calculating the sum of the squares of the largest K singular values, $\sum_{k=1}^{K} d_k^2$, and expressing this value as a percentage of the total variation.

It is also possible to determine the importance, or *inertia* of each dimension $1 \leq k \leq K$ in the approximated space (of rank K) by squaring the corresponding singular value d_k, and expressing this value as a percentage of the total variation.

Thus, to find a K-rank approximation of a matrix $\mathbf{Y}$ containing I point vectors of dimension J:

1. Find the centroid vector $\bar{\mathbf{y}}$.
2. Find the translated matrix $\mathbf{X} = \mathbf{Y} - 1\bar{\mathbf{y}}^\mathsf{T}$.
3. Determine the basis $\mathbf{R}$ and the diagonal singular matrix $\mathbf{D}$ using singular value decomposition.
4. Select the K columns of $\mathbf{R}$ (or K rows of $\mathbf{R}^\mathsf{T}$) that correspond with the largest K singular values in the diagonal matrix $\mathbf{D}$.
5. Project the instances represented by the matrix $\mathbf{X}$ into the space characterised by $\mathbf{R}_{(K)}$, by multiplying $\mathbf{X}$ with $\mathbf{R}_{(K)}$.

[5] By corresponding, we mean that the nth singular value of $\mathbf{D}$, d_n, corresponds to the nth column in the matrix $\mathbf{R}$.
[6] The row vectors of $\mathbf{R}^\mathsf{T}_{(K)}$ are equivalent to the column vectors of $\mathbf{R}_{(K)}$.

The points plotted in Figure 8 illustrate how a 13 dimensional space can be approximated to a 2-dimensional subspace. The instances are represented as rows in the matrix $\mathbf{Y}$, and their attributes are represented as columns. The two dimensions correspond to the dimensions with the largest inertia values, 40.75% and 18.97%.

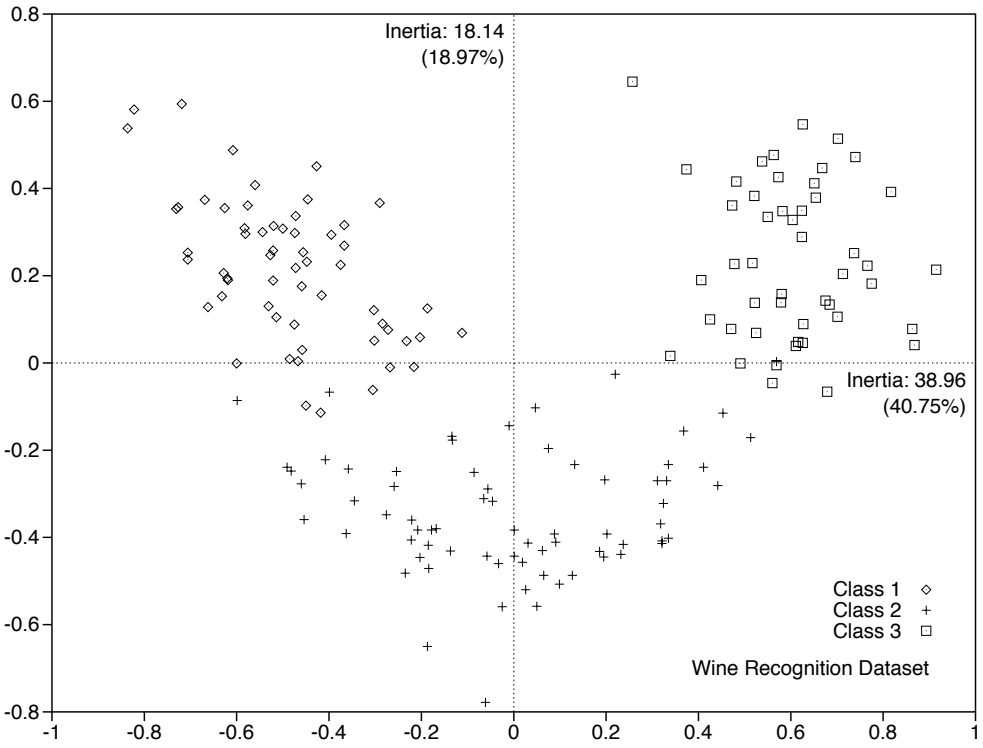

Figure 8: The 13-dimension Wine dataset approximated in a 2-dimensional subspace.

4 System Design

Many machine learning systems incorporate, or utilise some form of attribute selection to select an optimal (or sub-optimal) subset of the available attributes prior to induction (see Section 2). The sub-space approximation techniques described in the previous section project instances (represented as data points within some instance space) into a lower dimensional sub-space. To compare the benefits (in terms of predictive accuracy) of this approach with other attribute selection techniques, a suitable learning paradigm is required. Learning algorithms based on the Instance-Based Learning paradigm (IBL) (Aha, Kibler, & Albert 1991; Aha 1992; Salzberg 1991; Wettschereck & Dietterich 1995; Wilson & Martinez 1997) are ideal, as the accuracy of these techniques degrades in the presence of irrelevant or redundant data (Payne 1999), and they can be applied to problems where the domain contains numeric data.

Instance-based learning algorithms, which are sometimes referred to as Nearest Neighbour (NN) algorithms (Dasarathy, 1991), store and represent some or all the training instances as data points within a hyperdimensional instance space. Each instance consists of N attributes, and a class value. The instance space is usually described by N dimensions, where each dimension corresponds to a single

attribute. New (unseen) instances are classified by determining their location with this instance space, and identifying their nearest neighbour (or in the case of k-NN methods, the k nearest neighbours), using some *distance* function. The class value of the nearest instance (or instances) is then used to predict the class of the unseen instances.

Although a variety of distance functions exist (Wilson & Martinez, 1997), the *Euclidean Distance* function is traditionally used in most nearest neighbour algorithms. It is used to compare two numeric values within a Euclidean space. This distance function is a special case of the *Minkowskian Distance* function with $r = 2$, i.e.

$$D(i,j) = \left[\sum_{a=1}^{N} |i_a - j_a|^r\right]^{1/r}$$

where N is the number of attributes, and i and j are two instances. Another commonly used distance function, known as the *Manhattan Distance* (or *city-block*) function, is a special case of the Minkowskian Distance function with $r = 1$.

As these distance functions sum the difference between the values of each attribute, it is possible that attributes with large ranges can overwhelm those with relatively smaller ranges. For example, if one attribute has the range $[0..1000]$ and a second attribute has the range $[0..10]$, the relative difference between values for the first attribute will have a far greater effect on the final distance than the difference between values for the second distance. For this reason, the values are *normalised* so that they all lie in the range $[0..1]$. This can be achieved by dividing each value of each attribute by the range of that attribute[7]. Other normalisation functions ignore the values at the ends of the ranges, to avoid the effect of outliers. This may be achieved by, for example, eliminating the highest and lowest 5% of the values, or by selecting a range between two standard deviations either side of the mean value.

To compare the effects of using correspondence analysis for dimensionality reduction with more traditional approaches to attribute selection, a wrapper based attribute selection method was implemented. The search method used was a stochastic search known as the *Monte Carlo* method (Skalak 1994; Liu & Setiono 1996a; Liu & Setiono 1996b). This method was chosen as the number of search states visited can be controlled, and, unlike hill climbing approaches, it is not susceptible to local maxima (Payne & Edwards 1996). It is also possible to show that as the number of states visited increases, so does the probability of finding an optimal solution (Liu & Setiono 1996a). This method searches for the best attribute subset by selecting a random subset and evaluating it. The evaluation was performed using a leave-one-out cross validation with the nearest neighbour Euclidean distance learning algorithm on the training dataset (which contained only those attributes present in the attribute subset). A finite number of search states were visited (270 states for this study) and the attribute subset resulting in the highest cross validated accuracy were presented as the optimal subset.

5 Experimentation and Results

In order to determine whether or not the use of dimensionality reduction techniques can improve the performance of a nearest neighbour learning algorithm, experiments were run on twelve numerical datasets (Table 6) from the UCI Machine Learning Database Repository (Murphy & Aha 1994). Four nearest neighbour learning algorithms were evaluated (Table 7): *NN*, a basic Euclidean distance

[7]This function has been used to normalise the datasets presented in this paper.

nearest neighbour learning algorithm which utilised all the available attributes when classifying new instances; *MC*, which utilised a wrapper based attribute selection technique with a Monte Carlo search to locate optimal (or sub-optimal) attribute subsets prior to classifying new instances; and two new algorithms, *CA* and *CACP*, that employed the Correspondence Analysis techniques described in Section 3 to identify lower dimensional subspaces, and to map numeric data points into these subspaces. *CA* and *CACP* differed in the way the way the subspace mappings were generated: *CA* generated a subspace mapping from the entire training set, without utilising any class information; whereas *CACP* identified the mean or centroid data point for all the training instances of each class, and then used those centroids to generate the subspace mapping. Both *CA* and *CACP* used the Euclidean distance function to identify the nearest neighbours in the subspace.

Data Set		Attrs	Class	Inst	Inst/Class (%)
bupa	BUPA liver disorders	6	2	345	42.0 58.0
ionosp	JHU Ionosphere Database	34	2	351	64.0 36.0
pima	Pima Indians Diabetes Database	8	2	768	34.9 65.1
sonar	Sonar, Mines vs. Rocks	60	2	208	53.4 46.6
wiscon	Wisconsin Breast Cancer Database	9[a]	2	683[b]	65.0 35.0
wdbc	Wisconsin Diagnostic Breast Cancer	30[a]	2	569	37.3 62.7
wpbc	Wisconsin Prognostic Breast Cancer	33[a]	2	194	76.3 23.7
balance	Balance Scale Weight & Distance	4	3	625	7.8 46.1 46.1
glass	Glass Identification Database	9[a]	6	214	32.7 35.5 7.9
					6.1 4.2 13.6
iris	Iris Plants Database	4	3	150	33.3 33.3 33.3
shuttle	Challenger Space Shuttle O-Ring Data	4	3	23	74.0 21.7 4.3
wine	Wine Recognition Data	13	3	178	33.1 39.9 27.0

[a]The original dataset has an additional attribute which contains a unique identifier for each instance. This attribute was removed prior to use.

[b]There are 699 instances in the original database, but 16 had missing values and hence were removed.

Table 6: UCI Datasets used in this study.

A 20-fold cross validation strategy was used to evaluate the performance of the different learning algorithms on each of the datasets in Table 6. Several of these datasets each contained an attribute corresponding to a unique identification value. These attributes were removed from the datasets to prevent them affecting the classification accuracy. For example, the *glass* dataset contains an ordered numeric identifier, which is highly correlated (using Spearman's Rank Correlation, the coefficient is 0.958). If *NN* is used with a leave-one-out cross validation strategy, then the classification accuracy rises from 69.16% to 90.65% when this highly correlated attribute is included in the dataset.

Table 7 lists the different mapping functions used by the various learning algorithms. Each function generates a mapping by analysing the data in the training folds. This is done to prevent the data in the test folds affecting the way in which all the data is mapped. The *normalisation* function maps the range of numerical values for each attribute into the range [0..1]. Any values in the test folds that fall outside this range are mapped to the nearest boundary, i.e. $i_a = 1$ *iff* $(i_a > 1)$ *or* $i_a = 0$ *iff* $(i_a < 0)$ where i_a is the value corresponding to the attribute a within the instance i. The *class centroid identification* function creates a single centroid instance for each class, by averaging all the instances in the training folds belonging to that class (Kibler & Aha 1988). The *subspace mapping* function

uses the technique described in Section 3 to identify and approximate the basis for a subspace. The mapping is either generated from the instances in the training folds (*CA*) or from the class centroids (*CACP*). It is then used to project the instances in both the training and test folds into the approximated subspace prior to performing any classification tasks.

	Normalisation	Class Centroid Identification	Subspace Mapping
NN	×		
MC	×		
CA	×		×
CACP	×	×	×

Table 7: The pre-processing and mapping functions used by the four learning algorithms.

To determine the lowest number of dimensions that achieve the highest accuracy, the *CA* and *CACP* algorithms varied the number of dimensions to approximate the subspace for each dataset between 1 and n, where n was the total number of attributes available for the dataset. The results presented in the tables below refer to those tests that achieved the highest classification accuracy.

	NN	MC	$\Delta(MC)$
bupa	61.978 (6)	60.377 (4)	-1.601
ionosp	87.174 (34)	90.638 (14)	*3.464*
pima	70.994 (8)	67.958 (4)	**-3.036**
sonar	85.955 (60)	83.683 (28)	-2.272
wiscon	95.896 (9)	95.027 (5)	*-0.869*
wdbc	95.401 (30)	96.109 (14)	0.708
wpbc	69.056 (33)	71.168 (15)	2.112
balance	78.096 (4)	78.096 (4)	0.000
glass	68.093 (9)	71.001 (5)	2.908
iris	96.160 (4)	98.125 (2)	**1.965**
shuttle	75.000 (4)	60.000 (1)	**-15.000**
wine	94.862 (13)	94.792 (7)	-0.070

Table 8: Classification accuracies for the numeric UCI datasets for *NN* and *MC*. Values in bold or italic type indicate a significant difference at the 5% and 10% confidence levels respectively. The numbers of features used for each dataset are given in parenthesis.

Results from the UCI datasets

The results of the 20-fold cross validated tests for *NN* and *MC* are given in Table 8. The first two columns in this table refer to the classification accuracy of these two algorithms. The third column, $\Delta(MC)$, presents the difference in classification accuracy between these two algorithms, and hence changes in accuracy due to the feature selection component utilised by *MC*. The values in bold or italic indicate a significant difference with a 5% or 10% confidence interval respectively. The number of features used by each algorithm for each dataset is given in parenthesis.

The wrapper method succeeded in reducing the number of attributes for eleven of the twelve datasets. The number of attributes found for these datasets was typically half that of the original

	NN	CA	$\Delta(CA)$	CACP	$\Delta(CACP)$
bupa	61.978 (6)	61.978 (6)	0.000	61.978 (6)	0.000
ionosp	87.174 (34)	90.898 (22)	**3.724**	91.193 (11)	**4.019**
pima	70.994 (8)	70.994 (8)	0.000	70.994 (8)	0.000
sonar	85.955 (60)	86.955 (23)	1.000	85.995 (60)	0.000
wiscon	95.896 (9)	97.362 (6)	**1.466**	96.191 (3)	0.295
wdbc	95.401 (30)	96.651 (5)	*1.250*	96.294 (16)	0.893
wpbc	69.056 (33)	71.611 (16)	**2.555**	73.056 (15)	4.000
balance	78.096 (4)	78.122 (4)	0.026	88.952 (1)	**10.856**
glass	68.093 (9)	68.093 (8)	0.000	70.002 (8)	1.909
iris	96.160 (4)	96.160 (4)	0.000	96.696 (3)	0.536
shuttle	75.000 (4)	75.001 (1)	0.001	—	—
wine	94.862 (13)	97.084 (6)	*2.222*	97.639 (6)	**2.777**

Table 9: Classification accuracies for the numeric UCI datasets for *CA* and *CACP*. Values in bold or italic type indicate a significant difference at the 5% and 10% confidence levels respectively. The number of dimensions used to describe the final approximated subspace for each dataset is given in parenthesis.

number of attributes. There was a significant increase in classification accuracy for the *iris* dataset (at the 5% confidence level) and *ionosp* dataset (at the 10% confidence level). However, there was a significant decrease in classification accuracy for the *pima*, *shuttle* and *wiscon* datasets. No significant difference in classification accuracy was found between *NN* and *MC* for the remaining seven datasets. These results suggest that the wrapper method (used with a Monte Carlo search and a nearest neighbour algorithm) can successfully reduce the number of attributes in most cases, with little or no loss in classification accuracy, and that in some cases the classification accuracy can increase. Similar results were achieved by (Liu & Setiono) where a wrapper approach utilising the Monte Carlo search was evaluated with the two learning algorithms, C4.5 and ID3, on symbolic data. The *wiscon* dataset was used by another study of wrapper approaches (Kohavi & Sommerfield 1995), but found no significant difference in classification accuracy when a forward stepwise selection search was used to reduce the number of attributes with either ID3 or Naive-Bayes.

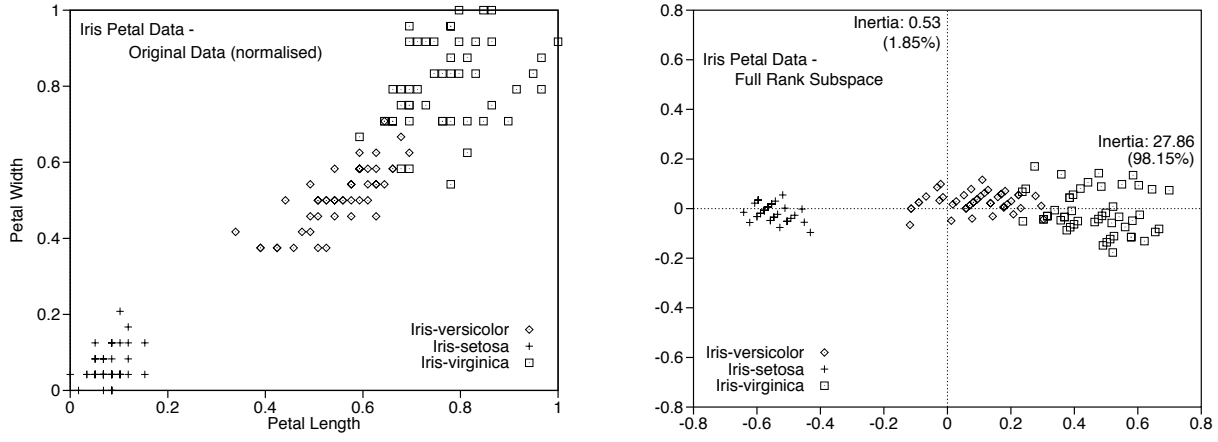

Figure 9: Mapping the two most relevant attributes of the *iris* dataset into a full ranked subspace.

Table 9 contains the results of the 20-fold cross validated tests for *CA* and *CACP*. Again, the results of the standard nearest neighbour algorithm, *NN* are presented, as are the changes in accuracy, $\Delta(CA)$ and $\Delta(CACP)$, for each dataset due to these two algorithms. The values in bold or italic indicate a significant difference between each approach employing the dimensionality reduction algorithms and *NN*, with a 5% or 10% confidence interval respectively. The number of dimensions used to describe the final approximated subspace for each dataset is given in parenthesis.

Both *CA* and *CACP* achieved a reduction in the number of dimensions required to represent the dataset for six of the twelve datasets. Again, the number of dimensions used was half that available for each dataset. The effects of the two algorithms differed for the *balance*, *iris*, *shuttle* and *sonar* datasets: *CA* failed to reduce the dimensionality of *balance* or *iris*; whereas *CACP* failed to reduce the dimensionality of *sonar*. No subspace mapping could be found for the class centroids of the *shuttle* dataset, and hence no results are given for *CACP* with this dataset. Neither dataset succeeded in reducing the dimensionality of the *bupa* or *pima* datasets.

The subspace mappings used by *CA* and *CACP* resulted in an increase in classification accuracy for most of the datasets, in addition to reducing the number of dimensions. The only dataset for which a reduction in the number of dimensions was achieved whilst the classification accuracy was unaffected was the *glass* dataset.

The results obtained for the *iris* dataset were lower than expected, as two of the four attributes (*petal length* and *petal width*) are known to be highly relevant to the classification task (Duda & Hart 1973; Michie, Spiegelhalter, & Taylor 1994). If only these two attributes are included in the dataset, then the accuracy for *NN* increases from 96.160% to 98.125%, and there is a corresponding increase in accuracy for *CA* (*CACP*) from 92.053% (93.927%) to 96.607% (96.607%) for a one dimensional approximated space. This suggests that the classification performance of both *CA* and *CACP* may degrade in the presence of irrelevant attributes. Figure 9 illustrates distribution of these two dimensional instances in the original (canonical) space, and their distribution within the new (full rank) subspace.

	NN	MC		CA		CACP	
	attrs	attrs	reduction	attrs	reduction	attrs	reduction
bupa	6	4	33.33%	6	0.00%	6	0.00%
ionosp	34	14	58.82%	22	35.29%	11	67.65%
pima	8	4	50.00%	8	0.00%	8	0.00%
sonar	60	28	53.33%	23	61.67%	60	0.00%
wiscon	9	5	44.44%	6	33.33%	3	66.67%
wdbc	30	14	53.33%	5	83.33%	16	46.67%
wpbc	33	15	54.55%	16	51.52%	15	54.55%
balance	4	4	0.00%	4	0.00%	1	75.00%
glass	9	5	44.44%	8	11.11%	8	11.11%
iris	4	2	50.00%	4	0.00%	3	25.00%
shuttle	4	1	75.00%	1	75.00%	-	—
wine	13	7	46.15%	6	53.85%	6	53.85%
Average Reduction		**11 datasets**	**51.22%**	**8 datasets**	**50.64%**	**8 datasets**	**50.06%**

Table 10: The number of attributes used by each algorithm and the corresponding reduction in dimensionality (given as a percentage of the original number of dimensions).

The result achieved by *CA* for the *balance* dataset suggests that when all the dimensions are present (i.e. no approximation is generated), the subspace mapping may still affect the classification accuracy of the learning algorithm. This is supported by the result obtained for the *iris* dataset when only the petal attributes are used. The effects of the mapping function generated by *CA* on this dataset are illustrated in Figure 9. In this case, the mapping function performs a rotation and a linear translation. The rotation should not affect the performance of the nearest neighbour algorithm. However, the varying translation of each dimension has the affect of distorting the subspace with respect to the original space, which is analogous to assigning relevance weights to each dimension.

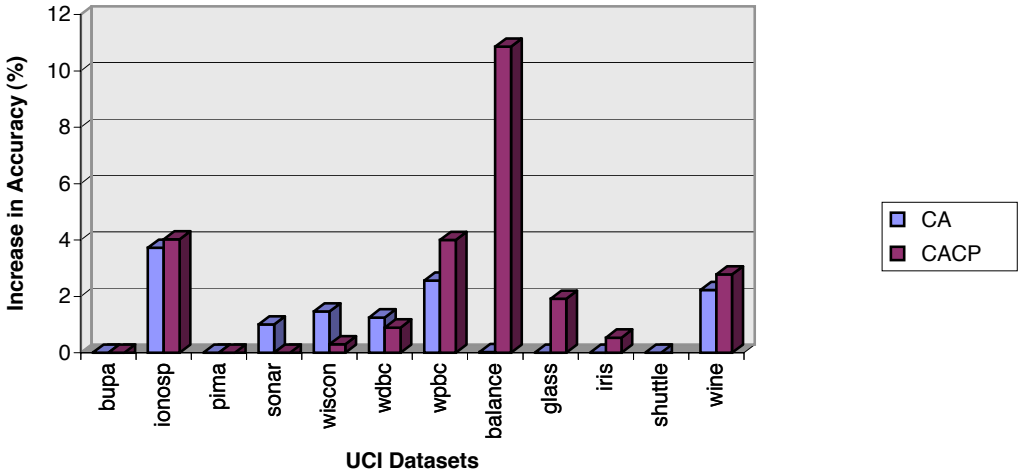

Figure 10: The difference in accuracy obtained when using the various dimensionality reduction techniques.

All three methods (*MC*, *CA* and *CACP*) succeeded in reducing the number of attributes required for the majority of the datasets used in this study. The reductions in dimensionality for each dataset (given as a percentage of the original number of dimensions) are listed in Table 10. *MC* reduced the number of attributes for eleven datasets by an average of 51.22% whereas *CA* and *CACP* reduced the dimensionality of eight datasets by an average of 50.64%, and 50.06% respectively. The reduction in dimensionality due to either *CA* or *CACP* resulted in an increase in classification accuracy for eight of the twelve datasets, although only four of these results (three for *CACP*) were significant at $\leq 10\%$ confidence level. *MC* achieved a significant increase in classification accuracy for only two of the datasets; but suffered a significant drop in classification accuracy for three others. Figure 10 charts the effects of the different methods of dimensionality reduction on the classification accuracy for each dataset.

Results from artificial datasets

The results for the *iris* dataset suggested that the performance of *CA* and *CACP* may degrade in the presence of irrelevant attributes. To investigate this hypothesis, two further datasets were created, consisting of 100 instances each. The datasets, illustrated in Figure 11, each consist of two numeric attributes and a boolean class label. The first dataset comprises of two linearly separable partitions. As *CACP* identifies and utilises class centroids, the second dataset contains four linearly inseparable partitions, two per class. A set of fifty, single attribute datasets were constructed, each containing a

single random value for each instance. New datasets were created for each experiment by combining one of the binary class datasets with a random sample of these irrelevant attribute datasets.

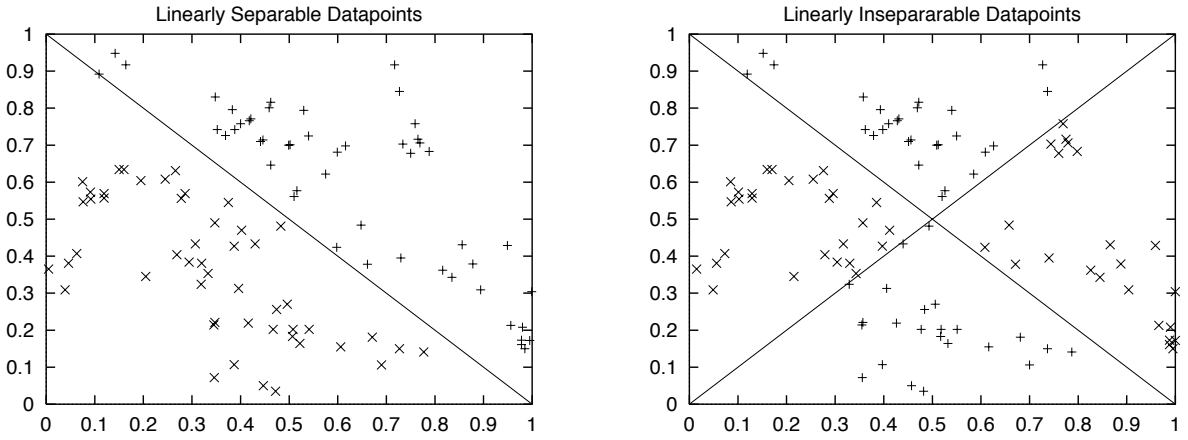

Figure 11: Two dimensional artificial data. Datapoints are either Positive (+) or Negative (×).

Various experiments were performed to investigate the behaviour of *CA* and *CACP* in the presence of irrelevant attributes. For each experiment, the two datasets containing the relevant attributes were augmented with an increasing number of irrelevant attributes. Each dataset was then tested with *NN*, *CA* and *CACP*. This was repeated fifteen times for different combinations of irrelevant attributes.

Figure 12 illustrates the results obtained from experiments on the linearly separable dataset. The classification accuracy of all three algorithms falls exponentially, as the number of irrelevant attributes increase. The classification accuracies for *NN* and *CA* are comparable with datasets containing small numbers of irrelevant attributes. However, after the number of irrelevant attributes exceeds fourteen, the difference in classification accuracy between the two algorithms becomes small but significant (a one-tailed t-test shows significance at the 5% level), with *CA* achieving a slightly higher accuracy than *NN*. The number of dimensions used by *CA* varies as the number of irrelevant attributes in the dataset increases. There is no reduction in dimensionality for datasets with few irrelevant attributes. As the number of irrelevant attributes increases beyond eight to forty-nine, the number of dimensions selected by *CA* increases slowly from eight to twenty-nine.

The error rate of *CACP* is much lower than that achieved by either *CA* or *NN*. *CACP* achieved a mean accuracy of 74.74% with forty-nine additional attributes, whereas *CA* and *NN* achieved mean accuracies of 57.47% and 55.93% respectively. The number of dimensions selected to approximate the space remained relatively constant (between three to five dimensions).

The results for the three algorithms on the linearly inseparable datasets are shown in Figure 13. Although *CACP* achieved superior results for these datasets, the overall performance was much lower than with linearly separable data. This drop in accuracy for *CACP* may be due to the method used to generate the centroids for each class. These centroids occur within a close proximity to each other, due to the distribution of the instances of each class. Although the instance space is divided into multiple partitions for each class, only a single centroid is generated for each.

The initial drop in accuracy in *NN* is not surprising, as there is an additional boundary separating the points of the two classes, and a small number of points lie along this new boundary. However, the results after the addition of only a few attributes (e.g. 11 attributes) are little better than that achieved by pure chance, indicating that any contribution that the relevant attributes have to any classification hypothesis has been obscured by the effects of the irrelevant attributes. The results

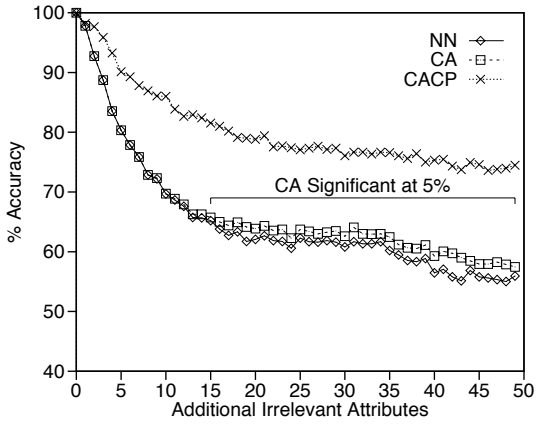

Figure 12: The effects of additional irrelevant attributes for a linearly separable dataset on three learning algorithms.

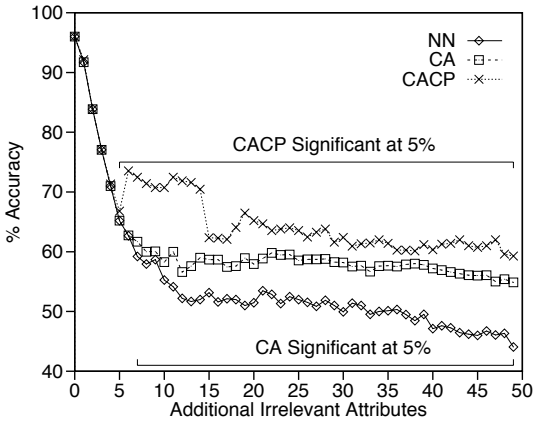

Figure 13: The effects of additional irrelevant attributes for a linearly inseparable dataset on three learning algorithms.

show an unusual increase in accuracy for *CACP* for datasets containing between five and fourteen additional attributes. As yet, no explanation has been found for this behaviour.

The above experiments were repeated to investigate the behaviour of both *CA* and *CACP* in the presence of redundant attributes. A set of forty-eight, single attribute datasets were constructed, each containing a single value for each instance. These values were based on those in the two datasets illustrated in Figure 11, and were calculated in one of several ways: values were copied from one of the dimensions of the original datasets; or values were calculated by inverting one of the dimensions using the function $f(x) = 1 - x$. In addition, some of the single attribute datasets were modified to introduce some variability to the similar dimensions. The function $f(x) = x \times (1 \pm rnd(\delta))$ was used, where $rnd(\delta)$ generates a small random number between 0 and δ; for this study we used $\delta = 0.05$.

All three algorithms achieved approximately 100% accuracy for the linearly separable dataset and 96.00% for the linearly inseparable dataset. A rank of two was always selected for *CA*, whereas the mean rank varied between one and four for *CACP*.

6 Discussion

The correspondence analysis techniques described in Section 3 were found to reduce the number of dimensions required by a nearest neighbour learning algorithm for eight of the twelve datasets studied. In addition, the resulting classification accuracy increased for all but one of these eight datasets. The techniques used by *CA* and *CACP* identified a new basis for a space that contained the instances in the training set, and then generated a lower dimension approximation to this space. As the number of dimensions of the approximated space (i.e. the rank of the space) was increased, the resulting classification accuracies were found to initially rise, and then fall slightly as a full rank subspace was used (i.e. no approximation performed).

In general, attribute selection techniques have been used to identify and eliminate both redundant and irrelevant attributes. The dimensionality reduction techniques used by *CA* and *CACP* appear to be very successful in removing redundant dimensions from the dataset. The data points are represented by an attribute-by-instance matrix. Once this matrix has been decomposed, the rank of

the matrix can be determined by the resulting diagonal matrix. This rank represents the number of linearly independent, orthogonal dimensions within a subspace. Therefore, the addition of any duplicate attributes, or any linear combination of attributes will not result in an increase in rank, and so will be eliminated by the decomposition. If two or more attributes contain very similar but not identical values, then there will be additional orthogonal dimensions to express the slight deviations between them (Figure 6). Because the inertia of such dimensions will be small, a lower rank subspace that excludes these dimensions will closely approximate the original subspace.

Unlike many of the existing attribute selection techniques, the dimensionality reduction techniques described here appear to have little impact in reducing the effects of irrelevant attributes. However, the performance of the class projected variant *CACP* degrades at a slower rate than either *CA* or a simple nearest neighbour in the presence of irrelevant attributes.

7 Conclusions

A number of different attribute selection techniques that reduce the dimensionality of a dataset have been investigated in recent years. These techniques not only reduce the number of dimensions required to learn a hypothesis, but can result in a classification increase for most learning algorithms (see Section 2). Various filter techniques have been proposed, but studies have shown that by including the learning algorithm in the selection process, better attribute subsets can be found. However, this wrapper approach cannot be scaled up to problems of more than a few attributes, due to the exponential increase in the size of the search required.

The text categorisation problem is one where attribute selection techniques have to function in the presence of tens or hundreds of thousands of attributes. The problem of scalability has been partially resolved by utilising simple variants of the filter method, or using a technique called Latent Semantic Indexing (Deerwester et al., 1990). We have studied the underlying principles upon which LSI was based, and developed two algorithms that reduce the dimensionality of data for small numeric classification tasks. The results from an evaluation of these algorithms suggest that this approach is successful in reducing the number of dimensions required for a learning task, and can result in an increase in the classification accuracy of the learning algorithm. However, the success of this approach appears to be due to the identification and elimination of redundant attributes. It fails to resolve the issue of removing irrelevant attributes. An investigation is required to determine the behaviour of this approach when used in conjunction with other attribute selection methods, such as weighted methods that identify and eliminate irrelevant attributes, but retain redundant ones. Further investigations are also required to determine the utility of correspondence analysis based approaches with other learning algorithms (such as rule induction algorithms), and to determine their performance in the presence of noise.

8 Acknowledgements

T.R. Payne acknowledges financial support provided by the UK Engineering & Physical Sciences Research Council (EPSRC). We also wish to thank Alberto Melacini for his helpful comments regarding SVD and low rank approximation.

References

Aha, D. (1992). Tolerating Noisy, Irrelevant and Novel Attributes in Instance-Based Learning Algorithms. *International Journal of Man-Machine Studies 36*, 267–287.

Aha, D. and Bankert, R. (1994). A Comparative Evaluation of Sequential Feature Selection Algorithms. In *Proceedings of the 5th International Workshop on Artificial Intelligence and Statistics*, pp. 1–7. Menlo Park, CA:AAAI Press.

Aha, D., Kibler, D., and Albert, M. (1991). Instance-Based Learning Algorithms. *Machine Learning 6*, 37–66.

Almuallim, H. and Dietterich, T. (1991). Learning With Many Irrelevant Features. In *Proceedings of the 9th National Conference on Artificial Intelligence (AAAI-91)*, pp. 547–552. MIT Press.

Armstrong, R., Freitag, D., Joachims, T., and Mitchell, T. (1995). WebWatcher: A Learning Apprentice for the World Wide Web. In *Working Notes of the AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments*. Menlo Park, CA:AAAI Press.

Bala, J., Huang, J., Vafaie, H., DeJong, K., and Wechsler, H. (1995). Hybrid Learning Using Genetic Algorithms and Decision Trees for Pattern Classification. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 719–724. San Mateo, CA:Morgan Kaufmann.

Berry, M. and Fierro, R. (1996). Low-Rank Orthogonal Decompositions for Information Retrieval Applications. *Numerical Linear Algebra with Applications 1*(1), 1–27.

Cardie, C. (1993). Using Decision Trees to Improve Case-Based Learning. In *Proceedings of the 10th International Conference on Machine Learning*, pp. 25–32. San Francisco, CA:Morgan Kaufmann.

Caruana, R. and Freitag, D. (1994). Greedy Attribute Selection. In *Proceedings of the 11th International Conference on Machine Learning*, pp. 28–36. San Francisco, CA:Morgan Kaufmann.

Cherkauer, K. and Shavlik, D. (1996). Growing Simpler Decision Trees to Facilitate Knowledge Discovery. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pp. 315–318. Menlo Park, CA:AAAI Press.

Creecy, R., Masand, B., Smith, S., and Waltz, D. (1992). Trading Mips and Memory for Knowledge Engineering. *Communications of the ACM 35*(8), 48–64.

Dasarathy, B. V. (1991). *Nearest Neighbor(NN) Norms: NN Pattern Classification Techniques*. Los Alamitos, California:IEEE Computer Society Press.

De Mántaras, R. (1991). A Distance-Based Attribute Selection Measure for Decision Tree Induction. *Machine Learning 6*, 81–92.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science 41*(6), 391–407.

Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. John Wiley and Sons.

Edwards, P., Bayer, D., Green, C., and Payne, T. (1996). Experience with Learning Agents which Manage Internet-Based Information. In *Machine Learning in Information Access: Papers from the 1996 AAAI Spring Symposium*, pp. 31–40. Menlo Park, CA:AAAI Press.

Fraleigh, J. and Beauregard, R. (1995). *Linear Algebra*. Menlo Park, CA:Addison-Wesley.

Greenacre, M. (1984). *Theory and Applications of Correspondence Analysis.* London, UK:Academic Press.

John, G., Kohavi, R., and Pfleger, K. (1994). Irrelevant Features and the Subset Selection Problem. In *Proceedings of the 11th International Conference on Machine Learning*, pp. 121–129. San Francisco, CA:Morgan Kaufmann.

Kibler, D. and Aha, D. (1988). Comparing Instance-Averaging with Instance-Filtering Learning Algorithms. In *Proceedings of the 3rd European Working Session on Learning, EWSL88*, pp. 63–88.

Kira, K. and Rendell, L. (1992). A Practical Approach to Feature Selection. In *Proceedings of the 9th International Workshop on Machine Learning*, pp. 249–256. San Mateo, CA:Morgan Kaufmann.

Kohavi, R. (1994). Feature Subset Selection as Search with Probabilistic Estimates. In *AAAI Fall Symposium on Relevance*, pp. 121–126. Menlo Park, CA:AAAI Press.

Kohavi, R. (1995a). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1137–1145. San Mateo, CA:Morgan Kaufmann.

Kohavi, R. (1995b). The Power of Decision Tables. In *Proceedings of the 8th European Conference on Machine Learning (ECML95)*, pp. 174–189. Berlin, Germany:Springer-Verlag.

Kohavi, R. and Sommerfield, D. (1995). Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, pp. 192–197.

Kononenko, I. (1994). Estimating Attributes: Analysis and Extensions of RELIEF. In *Proceedings of the 7th European Conference on Machine Learning*, pp. 171–182. Berlin, Heidelberg:Springer-Verlag.

Kubat, M., Flotzinger, D., and Pfurtscheller, G. (1993). Discovering Patterns in EEG-Signals: Comparative Study of a Few Methods. In *Proceedings of the 6th European Conference on Machine Learning*, pp. 366–371. Berlin, Heidelberg:Springer-Verlag.

Lang, K. (1995). NewsWeeder: Learning to Filter Netnews. In *Proceedings of the 12th International Machine Learning Conference (ML95)*, pp. 331–339. San Francisco, CA:Morgan Kaufmann.

Langley, P. (1996). Induction of Condensed Determinations. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pp. 327–330. Menlo Park, CA:AAAI Press.

Langley, P. and Iba, W. (1993). Average-case Analysis of a Nearest Neighbor Algorithm. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 889–894. San Mateo, CA:Morgan Kaufmann.

Langley, P. and Sage, S. (1994a). Induction of Selective Bayesian Classifiers. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pp. 399–406. Seattle, WA:Morgan Kaufmann.

Langley, P. and Sage, S. (1994b). Oblivious Decision Trees and Abstract Cases. In *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, pp. 113–117. Seattle, WA:AAAI Press.

Langley, P. and Sage, S. (1997). Scaling to Domains with Irrelevant Features. In R. Greiner (Ed.), *Computational Learning Theory and Natural Learning Systems*, Volume 4. Cambridge, MA:MIT Press.

Lewis, D. and Ringuette, M. (1994). A Comparison of Two Learning Algorithms for Text Categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, pp. 81–93.

Littlestone, N. (1988). Learning Quickly When Irrelevant Attributes Abound: A New Linear-threshold Algorithm. *Machine Learning 2*, 285–318.

Liu, H. and Setiono, R. (1996a). A Probabilistic Approach to Feature Selection - A Filter Solution. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 319–327. San Francisco, CA:Morgan Kaufmann.

Liu, H. and Setiono, R. (1996b). Feature Selection and Classification - A Probabilistic Wrapper Approach. In *The 9th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA-AIE'96)*, pp. 419–424.

Michie, D., Spiegelhalter, D., and Taylor, C. (Eds.) (1994). *Machine Learning, Neural and Statistical Classification.* UK:Ellis Horwood Ltd.

Moore, A. and Lee, M. (1994). Efficient Algorithms for Minimizing Cross Validation Error. In *Proceedings of the 11th International Conference on Machine Learning*, pp. 190–198. San Francisco, CA:Morgan Kaufmann.

Moulinier, I. (1996). A Framework for Comparing Text Categorisation Approaches. In *Machine Learning in Information Access: Papers from the 1996 AAAI Spring Symposium*, pp. 61–68. Menlo Park, CA:AAAI Press.

Moulinier, I. (1997). Feature Selection: A Useful Preprocessing Step. In *Proceedings of the 19th Annual BCS-IRSG Colloquium on IR Research*, pp. 140–158.

Murphy, P. and Aha, D. (1994). UCI Repository of Machine Learning Databases. Department of Information and Computer Science, University of California, Irvine, CA. [http://www.ics.uci.edu/∼mlearn/MLRepository.html].

Payne, T. (1999). *Dimensionality Reduction and Representation for Nearest Neighbour Learning.* Ph. D. thesis, The University of Aberdeen, Scotland.

Payne, T. and Edwards, P. (1996). A Survey of Feature Selection Methods. Unpublished Draft.

Payne, T. and Edwards, P. (1997). Interface Agents that Learn: An Investigation of Learning Issues in a Mail Agent Interface. *Applied Artificial Intelligence 11*(1), 1–32.

Payne, T. and Edwards, P. (1998). Implicit Feature Selection with the Value Difference Metric. In *ECAI 98 Conference Proceedings*, pp. 450–454.

Press, W. (1992). *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press.

Quinlan, J. (1986). Induction of Decision Trees. *Machine Learning 1*, 81–106.

Quinlan, J. (1993). *C4.5 Programs for Machine Learning.* San Mateo, CA:Morgan Kaufmann.

Richeldi, M. and Lanzi, P. (1996). Performing Effective Feature Selection by Investigating the Deep Structure of the Data. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pp. 379–382. Menlo Park, CA:AAAI Press.

Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval.* New York: McGraw-Hill.

Salzberg, S. (1991). A Nearest Hyperrectangle Learning Method. *Machine Learning 6*, 251–276.

Salzberg, S. (1992). Improving Classification Methods via Feature Selection. Technical Report TR JHU-92/12, Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218.

Schütze, H., Hull, D., and Pedersen, J. (1995). A Comparison of Classifiers and Document Representations for the Routing Problem. In *18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, pp. 229–237. New York, New York:ACM Press.

Singh, M. and Provan, G. (1995). A Comparison of Induction Algorithms for Selective and non-Selective Bayesian Classifiers. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 497–505. San Francisco, CA:Morgan Kaufmann.

Singh, M. and Provan, G. (1996). Efficient Learning of Selective Bayesian Network Classifiers. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 453–461. San Francisco, CA:Morgan Kaufmann.

Skalak, D. (1994). Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms. In *Proceedings of the 11th International Conference on Machine Learning*, pp. 293–301. San Francisco, CA:Morgan Kaufmann.

Terano, T. and Ishino, Y. (1996). Interactive Knowledge Discovery from Marketing Questionnaire Using Simulated Breeding and Inductive Learning Methods. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pp. 279–282. Menlo Park, CA:AAAI Press.

Vafaie, H. and De Jong, K. (1994). Improving a Rule Induction System using Genetic Algorithms. In R. Michalski and G. Tecuci (Eds.), *Machine Learning: A Multistrategy Approach. Vol 4*, pp. 453–469. San Francisco, CA:Morgan Kaufmann.

Weiner, E., Pedersen, J., and Weigend, A. (1995). A Neural Network Approach to Topic Spotting. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95)*, pp. 317–332.

Wettschereck, D., Aha, D., and Mohri, T. (1997). A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review 11*(1-5), 273–314. Special Issue on Lazy Learning.

Wettschereck, D. and Dietterich, T. (1995). An Experimental Comparison of the Nearest-Neighbor and Nearest-Hyperrectangle Algorithms. *Machine Learning 19*, 5–28.

Wilson, D. and Martinez, T. (1997). Improved Heterogeneous Distance Functions. *Journal of Artificial Intelligence Research 6*, 1–34.

Wu, C., Berry, M., Shivakumar, S., and McLarty, J. (1995). Neural Networks for Full-Scale Protein Sequence Classification: Sequence Encoding with Singular Value Decomposition. *Machine Learning 21*, 177–193.

Yang, Y. and Pedersen, J. (1997). A Comparative Study on Feature Selection in Text Categorisation. In *Proceedings of the 14th International Conference on Machine Learning.*, pp. 412–420. San Francisco, CA:Morgan Kaufmann.