Weighted Tree-Based Cluster Ensembles for High Dimensional Data

Thesis submitted by

Christine Wendy SMYTH   BSc (Hons) / BA

in March 2007

for the degree of Doctor of Philosophy
in the School of Mathematics, Physics, and Information Technology
James Cook University

STATEMENT OF SOURCES

DECLARATION

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references given.

_____________________ _______________

Signature Date

STATEMENT ON THE CONTRIBUTION OF OTHERS

The main body of this thesis is comprised of six publications, completed during the period of candidature. I, the undersigned, conducted the relevant research, designed the experiments, wrote all the necessary computer code, and drafted and edited the manuscripts. The contributions of others included knowledgeable discussions and proofreading of the manuscripts.

Stipend support was also received in the form of a James Cook University Prestige Postgraduate Research Scholarship for the period of candidature.

____________________ ________________

Signature Date

ACKNOWLEDGEMENTS

I don’t think that I really lost the plot too many times over the last three years. This is in no doubt due to the concerted efforts of many. Firstly, I’d like to thank my parents and grandmother. They kept me grounded (Mum), entertained (Dad), and well-fed (Goo). My sister, Helen, prevented me from becoming a boring statistician for which I will be eternally grateful. I also thank my friends for keeping me up to date with the outside world.

I’d also like to thank my academic colleagues, particularly my knowledgeable supervisors, Professor Danny Coomans and Dr Yvette Everingham, for expert advice and guidance.

To all these people, I say “thank you”.

ABSTRACT

The increasing size of datasets is particularly evident in the field of bioinformatics. It is unlikely that analyzing these large datasets with a single model will produce an accurate solution. This has led to the ensemble approach, where many models are averaged to give a consensus representation of the data. Taking a weighted average of the individual models has improved the accuracy of both classification and regression ensembles. However, weighting models within a cluster ensemble has remained relatively undeveloped because there is no gold standard available for comparison.

This thesis explores a technique of weighting cluster ensembles. A regression technique, multivariate regression trees, is shown to produce an accurate clustering solution. Each solution (tree) is then weighted purely in terms of its predictive accuracy. Various weighting strategies are trialed to determine the superior technique. After each individual tree is assigned a weight, the trees' co-occurrence matrices are obtained. The co-occurrence matrices are then aggregated together, weighted according to the trees' predictive weights. The final result is a single weighted co-occurrence matrix.

A new technique, similarity-based k-means, is developed in order to partition the weighted co-occurrence matrix. Similarity-based k-means is demonstrated to produce accurate partitions of similarity matrices. The resulting clusters agree with the known groups in the investigated datasets.

Furthermore, this thesis develops two other techniques so that maximal information can be obtained in conjunction with the weighted cluster ensemble. The first method suggests an estimate of the natural number of clusters in a dataset, by assessing the predictive performance and variability of similarity-based k-means for various numbers of clusters. The estimates agree with the known numbers of groups within the investigated datasets. The second method elucidates the variables that define the clusters. These variables have high classification power within the studied datasets.

Therefore, this thesis presents a holistic cluster analysis: clusters are accurately unearthed within large datasets; an estimate of the natural number of clusters is obtained; and the variables important in defining the clusters are also established. The weighted cluster ensemble technique is applied to a variety of small and large datasets. All results demonstrate the power of weighting the individual models within the ensemble: the developed weighted cluster ensemble technique consistently outperforms the other techniques. The results of analyzing two DNA microarray datasets are particularly promising. The discovered clusters overlap with the known diagnoses in the datasets, and the variables deemed important in defining the clusters have previously been suggested as biomarkers.

Whilst the size of contemporary datasets presents unique statistical challenges, the potential information within them is immense. Statistical techniques must be developed in order to accurately analyze these datasets. Motivated by the success of weighted regression and classification ensembles applied to large datasets, this thesis suggests a technique of weighting models within a cluster ensemble. The results highlight the potential of weighted cluster ensembles in high dimensional settings, such as the analysis of DNA microarrays.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

BACKGROUND

Large datasets are becoming more commonplace, as technology continues to improve. It is unlikely that a single statistical model will capture all the inherent information when applied to a large dataset. This has led to the ensemble approach. Ensembles combine many individual models in order to represent the dataset effectively. To ensure that the individual models within the ensemble are different to each other, the models are usually created by either:

(a) Supplying different input parameters such as starting seeds to the model (Greene, Tsymbal, Bolshakova and Cunningham 2004, Hadjitodorov, Kuncheva and Todorova 2006).
(b) Sampling the observational units and/or variables and growing the models on the subsampled datasets. Commonly used sampling schemes are bootstrapping, random features and random projection.
(c) Using different statistical models within the ensemble.
(d) Or any combination of the first three strategies.

Once enough individual models are obtained, it is then necessary to combine them into an ensemble. In the regression and classification settings, the "simple average" approach is very common. Here the average of the predictions of the M models is taken as the ensemble prediction for regression; and the majority vote of the classifications of the M models is taken as the ensemble classification. Simple average ensembles are known to be more stable and accurate than a single model for regression and classification problems (Krogh and Vedelsby 1995, Breiman 2001).

Recently, it has been shown that even greater accuracy is attainable if the models within the regression or classification ensemble are weighted more intelligently than the "simple average" (Friedman and Popescu 2003). Post processing is the name given in this thesis to the process of suggesting weights for each model in the ensemble to reflect the model's importance and relevance. The post processing paradigm is intuitive: by dampening the influence of the bad models and enhancing the influence of the good models the ensemble's accuracy is ultimately improved. In the classification and regression sense, a model can be defined as "good" or "bad" by comparing its predictions or classifications to the actual response (the "gold standard") via some loss criterion. The post processing technique will assign weights on the basis of the loss criterion.

Cluster ensembles have also proved to be more stable and accurate than a single clustering solution (Strehl and Ghosh 2002, Dudoit and Fridlyand 2003, Topchy, Jain and Punch 2003, Weingessel, Dimitriadou and Hornik 2003, Greene, et al. 2004, Hu and Yoo 2004). A single clustering solution attempts to partition the observational units into groups or clusters so that observational units within a cluster are more similar to each other than to observational units in another group (Hastie, Tibshirani and Friedman 2001). A single clustering solution may be erroneous or unstable when obtained from a very large dataset, because some variables within the dataset are superfluous and serve only to distort the true structure by creating noise. Consequently, cluster ensembles are well suited to clustering high dimensional datasets, because the ensemble approach is robust against large amounts of noise.

There are numerous methods of combining individual models to create a cluster ensemble. The simplest technique involves the creation of a co-occurrence matrix. The co-occurrence matrix is an $n*n$ matrix where the $(i,j)^{th}$ element of the matrix is the number of times observational units i and j are clustered together over the set of base clusterings divided by the number of models within the ensemble (Dudoit and Fridlyand 2003, Greene, et al. 2004). Fred and Jain (2003) suggest using a smaller co-occurrence matrix to reduce the computational complexity. The authors replace the traditional $n*n$ co-occurrence matrix with an $n*p$ matrix where the $(i,j)^{th}$ element is the percentage of times observational unit i has been clustered with its j^{th}, $j=1,...,p$ nearest neighbour. In contrast, Al-Razgan and Domeniconi (2006) define the elements of a co-occurrence matrix more probabilistically. Each observational unit in a base clustering solution has a vector of posterior probabilities associated with it. Each element of the vector gives the probability the observational unit belongs to a particular cluster. The similarity between two observational units i and j is the cosine similarity between their probability vectors. The similarities are averaged over the base clusterings to give the $(i,j)^{th}$ element of the co-occurrence matrix.

The co-occurrence matrix is then clustered by a suitable clustering algorithm: Greene, et al. (2004) and Fred and Jain (2002, 2003) use hierarchical algorithms. Hadjitodorov, et al. (2006) stated that using single linkage hierarchical clustering on the co-occurrence matrix and cutting the dendrogram at similarity θ was equivalent to thresholding the co-occurrence matrix (setting all values of the co-occurrence matrix greater than θ as one and all others as zero). Dudoit and Fridlyand (2003) convert the co-occurrence matrix to a dissimilarity matrix by subtracting each element from one, and then apply partitioning around medoids (PAM) (Kaufman and Rousseeuw 1987).

The co-occurrence matrix approach is not the only technique of creating cluster ensembles. Leisch (1999), in one of the original papers involving cluster ensembles combines the cluster centres obtained from each base clustering solution into one overall dataset. This dataset is then split using hierarchical clustering. Each observational unit is assigned to the cluster that contains the centre that the observational unit is closest to (over all centres). Topchy, et al. (2003) convert the cluster labels of the individual solutions to binary variables and perform k-means clustering on the new variables.

Another cluster ensemble combination technique of Dudoit and Fridlyand (2003) permutes the cluster labels of each clustering solution (obtained by applying the clustering algorithm to a bootstrap learning set). The objective is to find maximum overlap between the individual clustering solutions' labels and the labels obtained by clustering the entire dataset. The final ensemble labels are decided by a majority vote of the permuted labels of each clustering solution. Similarly, Weingessel, et al. (2003) implement a sequential voting technique, albeit slightly more complex than Dudoit and Fridlyand's (2003) bagging. Frossyniotis, Pertselakis and Stafylopatis (2002) iteratively permute the cluster labels so that there is maximum agreement with the previous cluster solution, and then takes a majority vote to obtain the overall partition.

Strehl and Ghosh (2002) in a seminal paper formally demonstrated how cluster ensembles could be expressed as an optimization problem. The optimal cluster ensemble labels shared maximal average mutual information with the individual solutions' labels. However, solving this optimization criterion is computationally infeasible. By converting the cluster labels to a hypergraph model, three methods of combining cluster labels were produced. Fern and Brodley (2006) also introduced another formulation of the cluster combination problem as a graph partitioning problem.

Similarly, both Hu and Yoo (2004) and Al-Razgan and Domeniconi (2006) used graph partitioning to split their "co-occurrence" matrices.

Topchy, Jain and Punch (2004) present a more statistical approach to cluster ensembles. The dataset is clustered M times and M cluster labels are obtained for each observational unit. A mixture model is fit to the cluster labels. The components of the mixture model are considered to be multivariate multinomial. Each observational unit is assigned to the component for which it has the highest posterior probability of belonging. Topchy, et al. (2004) highlight the low computational complexity of the mixture model approach.

Whilst techniques of creating cluster ensembles are multiplying, techniques of post processing cluster ensembles have remained relatively unexplored. The post processing technique should assign high weights to accurate partitions and zero weights to poor partitions and the consensus across the weighted partitions should then be found (Topchy, Jain and Punch 2005). However, it is difficult to assess the accuracy of an individual partition, thereby rendering post processing virtually impossible, because of the lack of a "gold standard" (actual response) associated with cluster analysis (Topchy, et al. 2003, Frossyniotis, Likas and Stafylopatis 2004, Al-Razgan and Domeniconi 2006). The few techniques of creating weighted cluster ensembles have been suggested by Kaufman and Rousseeuw (1990), He, Xu and Deng (2002), Modha and Spangler (2000), and Frossyniotis, et al. (2004).

Kaufman and Rousseeuw (1990) initially suggest taking a weighted sum of dissimilarity matrices created from different sources, using weights chosen in a "subjective way" but do not investigate the topic further. He, et al. (2002) convert the mixed type clustering problem into a cluster ensemble, where the labels obtained from clustering the categorical variables only must be combined with the labels obtained from clustering the numerical variables only. The two sets of labels are clustered using a weighted categorical clustering algorithm where the weights correspond to the proportion of categorical and numerical variables. It is suggested that this approach could be used as a generic weighted cluster ensemble technique where labels from different solutions are combined using a weighted categorical clustering algorithm. However, the method of calculating the weights when the sets of labels do not correspond to simply clustering different attributes of a dataset is not further explicated.

Modha and Spangler (2000) also introduced an algorithm that clusters observational units using weighted sets of variables simultaneously. The weights are calculated by maximizing the product of the resulting average within cluster coherence divided by the average between cluster coherence using each set of variables. The methodology could theoretically be extended to cluster ensembles, where each clustering solution is equivalent to a set of variables; however it becomes computationally infeasible as the number of clustering solutions increases.

A major advancement in weighting cluster ensembles was made by Frossyniotis, et al. (2004). Frossyniotis, et al. (2004) produce both boosted and bagged cluster ensembles. To generate the boosted cluster ensembles, boosted classification ensemble methodology is followed closely. At each iteration, a training set is selected, where samples which have previously been clustered poorly have a higher chance of being selected (boosted sample). The clustering algorithm is applied and the labels permuted such that they have the highest overlap with the previous labels. For each observational unit, $\underset{\sim}{x}_i$, a degree of cluster membership to each cluster k is calculated using the Euclidean distance, $dist(\underset{\sim}{x}_i, \underset{\sim}{x}_{S_r})$:

$$h_{i,k} = \frac{1}{\sum_{r=1}^{K} \frac{dist(\underset{\sim}{x}_i, \underset{\sim}{\bar{x}}_{S_k})}{dist(\underset{\sim}{x}_i, \underset{\sim}{\bar{x}}_{S_r})}}$$

where $\underset{\sim}{\bar{x}}_{S_r}$ is the mean vector of the r^{th} cluster. The $h_{i,k}$ are then analyzed to update the sampling weights of each observational unit: higher weights are given to observational units that have been clustered poorly. A weight is also given to the current clustering solution: a high weight is given to a solution that produces a "good" partition measured by a function of the $h_{i,k}$'s. The final partition is obtained by the weighted vote of the base clustering solutions. The boosted method outperforms the studied bagged clustering ensemble method, providing evidence that intelligently weighted cluster ensembles outperform simply averaging clustering solutions.

Whilst Frossyniotis, et al. (2004) offer a technique of weighting cluster ensembles, the strategy requires Euclidean distances. It is unlikely that this weighting strategy will produce accurate results in the high dimensional setting, where distances between any two points are relatively similar (Beyer, Goldstein, Ramakrishnan and Shaft 1999). However:

- the empirical evidence presented by Frossyniotis, et al. (2004) and
- the increased accuracy evidenced when weighting regression and classification ensembles and
- the ideology that within a cluster ensemble there will be both "good" and "bad" partitions (Fern and Brodley 2006)

all motivate further research into weighting cluster ensembles.

This thesis suggests a methodology for creating post processed cluster ensembles suitable for high dimensional data via the following steps:

- **Finding individual clustering solutions using a regression technique.** This will ensure that clusters are predictable and create a makeshift gold standard.
- **Weighting the individual solutions using regression post processing techniques.**
- **Partitioning the weighted cluster ensemble to obtain an overall clustering solution.**

These points are elaborated below.

FINDING INDIVIDUAL CLUSTERING SOLUTIONS USING A REGRESSION TECHNIQUE

This thesis conjectures that if the cluster solutions could be obtained as a by-product of a regression technique, the inherent "lack of gold standard" problem could be solved. Traditional regression loss criteria (such as residual sums of squares) could be utilized to measure the accuracy of the individual models. Cluster quality will then be measured in terms of predictive error as has been suggested by Yeung, Haynor and Ruzzo (2001), Dudoit and Fridlyand (2002), and Tibshirani, Walther, Botstein and Brown (2005). Therefore, this thesis first shows that a recent regression technique, multivariate regression trees (Segal 1992), can double successfully as a clustering technique in both the low and high dimensional settings.

Univariate regression trees were first introduced by Breiman, Friedman, Olshen and Stone (1984), as a technique of predicting a continuous response variable. In the beginning, all the data are contained in a single node. Regression trees recursively seek a binary partition of the dataset: at each stage a node is divided into two nodes. At each partition, the dataset is divided on the basis of the value of an explanatory variable, so as to make the new subsets as homogeneous as possible with respect to the response variable (Ploner and Brandenburg 2004). Observational units are dropped down the tree until they fall into a terminal node. Terminal nodes are a point at which a further partition of the data would result in only a very small improvement in the predictive ability of the tree and can be considered predictable by a single mean. Each terminal node contains a constant predicted value for the observational units falling into that node.

Multivariate regression trees are analogous to univariate regression trees. The predicted value at each terminal node is the mean response vector of the observational units in the learning sample that fall into that node. The notion of multivariate regression trees was first mooted by Segal (1992). The concept of multivariate regression trees was further explored by De'Ath (1999), who examined the splitting criterion of each node, and widened their scope to incorporate dissimilarity response matrices. De'Ath (1999) also found that multivariate regression trees are robust to noisy response and explanatory variables. A more theoretical explanation of multivariate regression trees can be found in the Supporting Theory appendix.

Regression trees have many advantages. Firstly, variables are not assumed to follow a particular distribution (Yohannes and Hoddinott 1999). Secondly, regression trees allow for mixed data types in the explanatory dataset (Yohannes and Hoddinott 1999). Finally, regression trees have the ability to assess the "importance of variables" and use these "important variables" to make the splits at each node.

The observational units in the terminal nodes are homogenous about the mean response vector. Therefore, it can be assumed that the observational units in each of the terminal nodes form a cluster, which can be characterised by the node's mean response vector. Thus, multivariate regression trees present a visually interpretable cluster analysis, and the investigator can see intuitively which explanatory variables are the most important.

A typical dataset designated for cluster analysis consists entirely of explanatory variables. However, multivariate regression trees find groups within response variables based on predictor variables. This thesis proposes two modifications that enable multivariate regression trees to be applied to datasets containing only explanatory variables. Firstly, the explanatory variables are replicated as response variables to create an "auto-associative multivariate regression tree" (Questier, Put, Coomans, Walczak and Vander Heyden 2005). In this way terminal nodes will be homogeneous about the mean of the explanatory variables. Secondly, to facilitate the application of the technique to high dimensional data, the response set is reduced using either principal components analysis or factor analysis (see Supporting Theory – Dimension Reduction Techniques appendix for more information). The resulting principal component scores or factor scores are used as the response set.

This thesis is the first investigation into the capability of multivariate regression trees as a clustering technique. It exploits multivariate regression trees so that clusters are produced by performing a regression analysis on the dataset. Each observational unit will have an associated predicted value (the mean of its terminal node). By sampling the dataset, many regression trees can be grown, and thereby many cluster solutions obtained. Because the multivariate regression trees are regression models, regression post processing methods can be applied to weight the clustering solutions within the ensemble.

WEIGHTING THE INDIVIDUAL SOLUTIONS USING REGRESSION POST PROCESSING TECHNIQUES

Each regression tree within the ensemble must then be weighted according to its accuracy. This is commonly referred to as post processing the ensemble. Post processing (regression ensembles) aims to assign large weights to models with low prediction error and small weights to inaccurate models, in order to increase the performance of the ensemble. This thesis investigates the forward stagewise approximation to the lasso, Bayesian linear regression, quadratic programming and evolutionary algorithms as potential post processing techniques.

Assume the regression ensemble (here an ensemble of regression trees) to be given by:

$$\hat{f}_{\text{ens}}(\underset{\sim}{x}) = \sum_{\text{m}=1}^{\text{M}} \omega_{\text{m}} \hat{f}_{\text{m}}(\underset{\sim}{x})$$

where $\hat{f}_{\text{m}}(\underset{\sim}{x})$ is the prediction of an observational unit $\underset{\sim}{x}$ by the m^{th} model; ω_{m} is the weight assigned to $\hat{f}_{\text{m}}(\underset{\sim}{x})$; and M is the number of models. An obvious choice for $\underset{\sim}{\omega}$, the vector of all weights, is to choose $\underset{\sim}{\omega}$ to minimize a generic loss criterion:

$$\underset{\sim}{\hat{\omega}} = \min_{\underset{\sim}{\omega}} \sum_{i=1}^{n} L\left(\underset{\sim}{y}_i, \sum_{m=1}^{M} \omega_m \hat{f}_m(\underset{\sim}{x}_i) \right)$$

where n is the number of observational units. If the generic loss criterion is given by the squared error loss, the post processing process reduces to an ordinary least squares problem. However, the ordinary least squares weights are prone to overfitting. The problem of overfitting is mitigated by the addition of a penalty function $\lambda \cdot P(\underset{\sim}{\omega})$ to the squared error loss:

$$\underset{\sim}{\hat{\omega}} = \min_{\underset{\sim}{\omega}} \sum_{i=1}^{n} L\left(\underset{\sim}{y}_i, \sum_{m=1}^{M} \omega_m \hat{f}_m(\underset{\sim}{x}_i) \right) + \lambda \cdot P(\underset{\sim}{\omega}).$$

The above equation has its origins in regularized regression. The parallels between regularized regression and post processing are obvious; the data matrix is replaced by the predictions matrix[1] and the regression coefficients are replaced by the ensemble weights.

The penalty function:

$$P(\underset{\sim}{\omega}) = \sum_{m=1}^{M} |\omega_m|$$

is referred to as the lasso (Tibshirani 1996). The lasso is well known, it produces sparse estimates of coefficients and increases accuracy. For these reasons it is used as a "benchmark" in this thesis to compare other common post processing techniques against.

This thesis implements the forward stagewise estimate as an approximation to the solution of the lasso because if offers considerable computational savings[2]. The forward stagewise estimates are remarkably similar to the true solution of the lasso objective

[1] The predictions matrix is an $n * M$ matrix with each column containing the predictions of the n observational units for a single model.
[2] The estimates are often referred to in this thesis as the "lasso" estimates.

function, and this similarity has been quantified using the least angle regression (LARS) (Efron, Hastie, Johnstone and Tibshirani 2004) algorithm. Indeed the forward stagewise algorithm has been shown to be a very successful post processing technique by Friedman and Popescu (2003). The algorithm (specifically for regression) is as follows (Hastie, et al. 2001):

1) Set all weights to zero. Choose ε as a small number greater than zero, and choose the number of iterations, its, to be quite large.

2) $\text{For } \mathrm{j} = 1 : \text{its}$

$$\left(\beta^*, \delta^*\right) = \arg\min_{\beta,\delta} \sum_{i=1}^{n} \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m \hat{f}_m(\underset{\sim}{x}_i) - \beta \hat{f}_\delta(\underset{\sim}{x}_i) \right)^T \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m \hat{f}_m(\underset{\sim}{x}_i) - \beta \hat{f}_\delta(\underset{\sim}{x}_i) \right).$$

$$\hat{\omega}_{\delta^*} = \hat{\omega}_{\delta^*} + \varepsilon \cdot sign(\beta^*).$$

3) Finally,

$$\hat{f}_{\text{ens}}(\underset{\sim}{x}) = \sum_{m=1}^{M} \hat{\omega}_m \hat{f}_m(\underset{\sim}{x}).$$

Using Bayesian linear regression theory (see Supporting Theory – Bayesian Linear Regression appendix), the lasso coefficients are equivalent to the modes of the conditional posteriors for the regression coefficients given independent double exponential prior distributions for each coefficient. This thesis also trials different priors (multivariate normal, multivariate t, and Weibull) to ascertain if other priors can give results superior to the lasso.

It was initially proposed to solve the lasso using the optimization technique “quadratic programming” (Tibshirani 1996). Quadratic programs solve a quadratic loss subject to a set of constraints. The lasso can be reformulated as a quadratic program:

$$\underset{\sim}{\omega}^{\text{lasso}} = \arg\min_{\underset{\sim}{\omega}} \left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}\right)^{T} \left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}\right)$$

subject to

$$\sum_{\mathrm{m}=1}^{\mathrm{M}} \left|\omega_{\mathrm{m}}\right| \leq \mathrm{s}$$

where $\hat{\boldsymbol{Y}}$ is the predictions matrix (an $\mathrm{n}*\mathrm{M}$ matrix with each column containing the predictions of the n observational units for a single model); and the loss function is the squared error loss. Quadratic programs are described in more detail in the Supporting Theory appendix.

Quadratic programming is obviously not restricted to the lasso however, and in this thesis quadratic programming is trialed as a potential post processing technique using different quadratic loss functions and constraints to ascertain if other loss functions and constraints can give results superior to the lasso. Quadratic programs (and also linear programs) have previously been used successfully as a post processing technique for regression and classification ensembles (Krogh and Vedelsby 1995, Heskes 1997, 1998).

To contrast to the more formal optimization techniques, purely stochastic procedures, evolutionary algorithms, are also explored in this thesis. Evolutionary algorithms encode a potential set of weights via a chromosome (see Supporting Theory – Evolutionary Algorithms appendix). The resulting ensemble is assessed via a fitness function and the best chromosomes reproduce into the next generation. Over generations, all chromosomes begin to approach the best set of weights. Evolution strategies have previously been used successfully as a post processing technique for regression ensembles (Zhou, Wu, Jiang and Chen 2001).

PARTITIONING THE WEIGHTED CLUSTER ENSEMBLE TO OBTAIN AN OVERALL CLUSTERING SOLUTION

After post processing the ensemble of regression trees, each tree will have an associated weight. To create the post processed cluster ensemble, each tree's resulting co-occurrence matrix is multiplied by the tree's weight, and the weighted co-occurrence matrices are aggregated to give an overall co-occurrence matrix. The weighted co-occurrence matrix approach was chosen here because it does not require an optimization algorithm to permute the cluster labels of individual solutions (Dudoit and Fridlyand 2003, Hadjitodorov, et al. 2006). Furthermore, the assumption that observational units in a "natural" cluster will commonly be grouped together by the individual solutions (Greene, et al. 2004) is intuitively appealing.

The weighted co-occurrence matrix must then be divided into clusters. This thesis suggests a new technique for clustering the co-occurrence matrix with a focus on predictability: similarity-based k-means (SBK). SBK takes a similarity matrix as input (see Supporting Theory – Similarity-Based K-Means appendix). The notion of SBK was inspired by the sums of squares splitting criterion (SSR) of multivariate consensus trees (Hancock 2006). SBK enforces the predictability of the solution by explicitly predicting the group structure found within the entire similarity matrix (including the covariance submatrices) shown to be important by Hancock (2006). The algorithm incorporates a validity criterion to ensure that clusters are meaningful. Previous similar efforts reported in the literature include splitting a dissimilarity matrix using k-means (Nguyen and Worring 2004, Lai, Huang, Wibowo and Tanaka 2005, Jacobs 2006) and applying k-means to the eigenvectors of a co-occurrence matrix (Jonasson, Hagmann, Thiran and Wedeen 2005).

By predicting the co-occurrence matrix, SBK allows for an estimate of the natural number of clusters in a dataset, based on figures of merit (FOMs) (Yeung, et al. 2001). FOMs (see Supporting Theory – Determining the Natural Number of Clusters appendix) use the ensemble ideology to assess the number of clusters and this point is elaborated below. Before describing FOMs, other techniques that exploit the ensemble ideology to assess the number of clusters are described briefly.

EXPLOITING THE CLUSTER ENSEMBLE METHODOLOGY TO ASSESS THE NATURAL NUMBER OF CLUSTERS

Cluster ensemble methodology can be used to assess the number of clusters in a dataset and to validate clustering results. In fact, some of the first forays into cluster ensembles focussed on these two goals rather than finding an overall partition. For example, Levine and Domany (2001) obtain base solutions by subsampling the data and then constructing base co-occurrence matrices for each solution. Each base co-occurrence matrix is compared via an appropriate function to the single co-occurrence matrix obtained when the entire dataset is clustered. This process is repeated for different values of parameters to ascertain which parameter gives rise to the most stable solution.

Similarly, Smolkin and Ghosh (2003) suggest a technique for assessing the stability of clusters. Smolkin and Ghosh (2003) cluster the dataset into K clusters. Then, they take a random subset of variables and re-cluster the observational units into K clusters using only these variables. This process is repeated. The stability of an original cluster (obtained from the entire dataset) is the proportion of times it is found as a cluster using the reduced subsets of variables.

The stability of the solution can also elude to the correct number of clusters. For example, Monti, Tamayo, Mesirov and Golub (2003) compared the cumulative distribution functions of the K co-occurrence matrices obtained by clustering the data into $k = 1,...,K$ clusters. In a similar fashion, Ben-Hur, Elisseeff and Guyon (2002) computed the pairwise similarity between two base co-occurrence matrices created by clustering two subsamples of the data. The process was repeated to obtain a cumulative distribution function of pairwise similarities. This cumulative distribution function was computed for each k, $k = 1,...,K$ and these were investigated to assess the optimal number of clusters.

Monti, et al. (2003) and Ben-Hur, et al., (2002) both assume that by assessing the stability of the cluster solutions within an ensemble, the number of clusters can be deduced. An alternative assumption is that by assessing the "predictability" of the individual solutions the number of clusters can be found or the clustering results can be validated. Dudoit and Fridlyand (2002) randomly divide the observational units into two distinct sets. A clustering algorithm is applied to the first and second subsets to obtain k clusters for each subset. The resulting labels of the first subset are used to form a classifier. The classifier is applied to the second subset and the similarity between its classifications and the cluster labels is calculated. The procedure is repeated to obtain a set of similarities for each number of clusters. The median similarity for each k is compared to the average median similarity using a null distribution and an estimate of the number of clusters is obtained.

Tibshirani, et al. (2005) suggest a similar procedure to Dudoit and Fridlyand (2002). The authors randomly divide the dataset in two and cluster both subsets to k clusters. The centroids of the first set are used to assign the observational units of the second subset to clusters. The minimum overlap of the actual cluster labels and predicted cluster labels of the second subset gives the "prediction strength". The process is repeated and an average prediction strength for each k is obtained. The maximum prediction strength over $k = 1,...,K$ is deemed to give the optimal number of clusters.

Yeung, et al. (2001) also take a predictive approach to validating cluster results with a figure of merit. The authors leave out a variable, cluster the data into k clusters, then calculate the root mean square deviation of the left out variable within each cluster. By summing over all variables an average estimate of the predictive power (figure of merit) of the algorithm can be obtained for each k. The performance of clustering algorithms can be compared by comparing their predictive powers at each k. Also the "elbow" of the figure of merit (FOM) curve is shown to often coincide with the known number of clusters in the dataset. This thesis extends Yeung, et al.'s (2001) FOM methodology in conjunction with the SBK algorithm to provide an estimate of the number of clusters in a dataset.

AN EXAMPLE

The size of datasets is increasing dramatically as technology improves (Ribarsky 2003), and this is particularly evident in the field of bioinformatics. For example, with the recent advent of DNA microarrays, measurements on the expression levels of tens of thousands of genes can be made on numerous samples (Jinang and Zhang 2004).

Recently, cluster analysis has been applied to DNA microarray data (Alon, et al. 1999, Alizadeh, et al. 2000, Segal, Taskar, Gasch, Friedman and Koller 2001) and it is one of the most commonly applied techniques to such data (Ben-Hur and Guyon 2003). Clustering genetic data has an important role to play in the discovery, validation, and understanding of various diseases (Do, McLachlan, Bean and Wen 2002). The aim of clustering DNA microarray data is to understand how genes and samples are organized. It is often of interest to cluster both genes and samples to answer questions such as:

- Which samples are most similar to each other in terms of their expression profiles across genes? (Hastie, et al. 2001)
- Which genes are most similar to each other in terms of their expression profiles across samples? (Hastie, et al. 2001)

A DNA microarray data set typically has a large number of variables (genes), a small number of observational units (samples) and possibly overlapping clusters (Do, et al. 2002). Traditional clustering algorithms are not effective when applied to datasets with these characteristics for numerous reasons. Firstly, with a large number of variables, some may be irrelevant and/or redundant, and only serve to obscure the "true clusters" (Milligan 1980). Secondly, as the number of variables (genes) is greater than the number of observational units (samples) it is a non-standard problem in cluster analysis (McLachlan, Bean and Peel 2002) and hence suffers the "curse of dimensionality". Finally, most clustering algorithms are "effective in unmasking global structure", but they are much less effective when the dataset contains complex cluster shapes, such as overlapping clusters (Do, et al. 2002, Pollard and van der Laan 2002). Therefore, the necessary characteristics of algorithms used to cluster genetic data include: robustness against large amounts of noise and outliers; the ability to handle high dimensional data; and the ability to detect irregular shapes (Jinang and Zhang 2004).

Consequently, cluster ensembles are well suited to clustering large DNA microarray datasets because the ensemble approach is robust against large amounts of noise. In fact, cluster ensembles have been applied with success to DNA microarray datasets (Dudoit and Fridlyand 2003, Grotkjaer, Winther, Regenberg and Nielsen 2006). Thus, as an illustration the methods created in this thesis are also trialed with two well known microarray datasets.

BIOMARKERS

Finding the variables that best discriminate clusters within the DNA microarrays warrants investigation. The ultimate outcome is the discovery of 'biomarker' variables that define clusters (such as aggressive and non-aggressive cancer). For example, if two clusters correspond to early stage cancer and healthy patients, genes that define the clusters could be investigated as potential 'early detection' biomarkers. In the future biomarkers (genes or proteins) may answer:

(a) Can the disease be detected early in its onset?

(b) How far has the disease progressed at its time of detection?

(c) What is the best course of treatment for the individual patient? (Wadsworth, et al. 2004)

(d) What is the likelihood of recurrence?

(e) What is the nature of any recurrence (for example, aggressive)? (Duggan, et al. 2003)

The multivariate regression trees used in this thesis give variable importance measures. A weighted variable importance measure can be obtained over the ensemble. Variables that are deemed ‘important’ by the ensemble have been used by the trees to distinguish accurate clusters. Those variables that are found to have high importance when the technique is applied to DNA microarrays could warrant further investigation as biomarkers.

OVERVIEW OF THESIS LAYOUT

The following three sections in this thesis are devoted to introducing and testing the various components of the post processed cluster ensemble methodology. The first section introduces multivariate regression trees, shows how they can double successfully as a clustering technique, and illustrates the extension that makes them successful in the high dimensional setting. The next section trials regression post processing methods and attempts to identify a superior technique. The following section takes the weighted co-occurrence matrices given by the post processed multivariate regression trees on various small datasets and splits them using SBK. The variable importance lists and cluster number curves are also illustrated. The post processed cluster ensembles are also demonstrated on two large DNA microarray datasets using the modification required to make multivariate regression trees successful in the high dimensional setting.

Each section contains two manuscripts submitted for publication in chosen journals. This implies that nomenclature is not consistent across manuscripts, because it is specific to the designated journal. However, all nomenclature is explained within the manuscript. For the same reason, formatting is not consistent across manuscripts because it complies with the specific journal.

The manuscripts are summarized at the beginning of the section and additional information is given at the end of each section. The major results from all sections are recapitulated in the "Concluding Summary" section. Potential avenues of future research are given in the "Future Work" section. Although each manuscript contains all the necessary theory, some theory is elaborated in the "Supporting Theory" appendix.

MULTIVARIATE REGRESSION TREES FOR CLUSTER ANALYSIS

OVERVIEW

To allow solutions within a cluster ensemble to be weighted according to their relevance, each solution's accuracy must be ascertained. Because there is no "gold standard" associated with cluster analysis, assessing the accuracy of an individual clustering solution is difficult. This thesis suggests that if the clustering solution were obtained as a result of performing a regression analysis on the data, the solution could be evaluated in terms of its prediction error. This section shows that multivariate regression trees can double successfully as a clustering technique and compares them to other successful clustering algorithms. The first manuscript explores auto-associative multivariate regression trees (AAMRTs) in the low dimensional setting and also illustrates figures of merit for cluster number selection.

AAMRTs cluster a dataset by replicating the explanatory variables of the dataset as the response variables. The trees recursively split the dataset to maximize the homogeneity of the observational units in the nodes about their means. As the dataset becomes larger (the number of variables increases), the distances between any two points are relatively similar (Beyer, et al. 1999) so searching for clusters that minimize their within sums of squares becomes pointless. Therefore, the second manuscript extends multivariate regression trees such that they are a suitable clustering technique for large, noisy datasets. The dataset is reduced via principal components analysis or factor analysis to mitigate the "curse of dimensionality" and clustering is performed in the reduced dimension space via the original variables. The trees are trialed on datasets increasingly perturbed by noise and are shown to outperform other clustering techniques. Although each manuscript contains the necessary theory, some theory is elaborated in the "Supporting Theory" appendix.

Auto-Associative Multivariate Regression Trees for Cluster Analysis

Christine Smyth[*], Danny Coomans, Yvette Everingham and Timothy Hancock

Statistics and Intelligent Data Analysis Group,

School of Mathematical and Physical Sciences,

James Cook University,

Australia

Chemometrics and Intelligent Laboratory Systems

Volume 80

Pages 120-129

2006

SUMMARY

Multivariate Regression Trees, an intuitive and simple regression technique, intrinsically produce homogenous subsets of data. These characteristics imply that Multivariate Regression Trees have the potential to be utilised as an easily interpretable clustering method. The suitability of Multivariate Regression Trees as a clustering technique is investigated with two real datasets containing only explanatory variables. The preliminary results show that Multivariate Regression Trees as a clustering algorithm produce clusters of similar quality to the well-known K-means technique, and more recent approaches to Cluster Analysis including Mixture Models of Factor Analysers and Plaid Models. The study also evaluates the suitability of various criteria used to describe cluster solutions.

1. Introduction

Multivariate Regression Trees [1] are a multivariate extension of Classification and Regression Trees [2] and have typically been used in the regression context. After growing the tree, an observational unit is dropped down the tree, following the binary decision functions at each split, until it comes to rest in a terminal node. Each terminal node contains a predicted value for all observational units falling into that node.

Multivariate Regression Trees are a relatively intuitive and simple technique that produces constrained clusters via the terminal nodes [3]. The clusters are homogenous with respect to the response variables and are defined by the explanatory variables that form the tree [3-5]. Questier et al. [3] used identical explanatory and response variables to create an Auto-Associative Multivariate Regression Tree (AAMRT). The authors investigated AAMRT as an unsupervised variable selection technique and indicated that

AAMRT can be considered a “non-constrained” clustering technique. Here, we investigate further the capabilities of AAMRT as a clustering method.

We explore the suitability of the proposed technique using two real datasets underpinned by biological phenomena. To help benchmark the capabilities of this extended approach of Multivariate Regression Trees, we employ three other clustering techniques, K-means, Mixture Models of Factor Analysers [6, 7], and Plaid Models [8].

K-means is deemed the “reference technique” because it is a widely-used, traditional clustering method. Moreover, Milligan [9], studying the effects of error perturbation on clustering algorithms, found K-means accompanied with “rational starting procedures” to be the technique most resilient to the addition of spurious noise variables.

The recent Mixture Models of Factor Analysers is a statistical approach to Cluster Analysis. The model is a sum of component density functions. Each component represents a cluster, and observational units have an associated probability of belonging to each cluster. Mixture Models of Factor Analysers extend Normal Mixture Models by incorporating Factor Analysis. The dimensionality reducing property of Factor Analysis implies that this clustering technique lends itself to analysing high dimensional data.

Similarly, the recent Plaid Models were designed for use with large datasets such as DNA microarrays. This technique is a novel form of two-way cluster analysis: variables and observational units are clustered simultaneously. Observational units are not restricted to being in exactly one cluster. The model is a sum of “layers”, and each layer represents a cluster.

One of the main issues associated with Cluster Analysis is estimating the true number of clusters in a dataset [10, 11]. To address this issue, Tibshirani et al. [12] have proposed the Gap Statistic. The Gap Statistic outperforms the traditional Silhouette Statistic [11] and the Hartigan [13] Statistic [12]. The Gap Statistic has also been shown to perform well against more recent techniques, such as a stability based method [14], and a prediction-based resampling method [15]. Because of its apparent success, we used the Gap Statistic to estimate the number of clusters in the datasets using each of the clustering methods. These estimates were then compared to the known numbers of classes within the datasets.

The Gap Statistic supplemented the statistic commonly used in association with each technique to select the number of clusters. The relative error curve of a multivariate regression tree is a monotonically decreasing function that shows the homogeneity of the terminal nodes within a tree as it is successively split. The size of the decrease in relative error between a tree with k splits and $k+1$ splits is proportional to the amount of heterogeneity removed by the addition of the new split. As the decrease in relative error becomes less obvious, any additional split will result in only a small decrease in the heterogeneity of the terminal nodes, and at this point the clusters are sufficiently homogeneous. Therefore, we suggest using the elbow of the relative error curve as a heuristic to choose the number of clusters (similar to the heuristic of using the elbow of a scree plot to choose the number of principal components).

The Bayesian Information Criterion (BIC) is useful when estimating the number of components required in a mixture model, although regularity conditions do not hold

[16]. These estimates are also included to supplement the estimates of the Gap Statistic. The Plaid Models program incorporates a regularisation technique at each stage, to determine if addition of a further layer to the model is warranted [8]. This essentially allows an estimate of the optimal number of layers for the model.

A second challenge surrounding cluster analyses entails gauging the quality of the derived clusters. Here we incorporate External and Internal Criteria and Figure of Merits [17] to assess the quality of the derived clusters. Known class memberships allow the incorporation of an External Criterion. An External Criterion provides an estimate of the quality of the derived clusters by using information obtained outside the clustering process to ascertain the degree of correct class recovery [9]. The External Criterion used in this study was the Adjusted Rand Index [18].

An Internal Criterion requires no prior knowledge of class membership and is a "measure of the goodness of fit between the data and the obtained solution" [9]. Because an Internal Criterion only uses information obtained from the clustering process, it complements an External Criterion [9]. The Internal Criterion used was a Point Biserial Correlation.

Figure of Merits (FOMs) are a means of authenticating clusters by assessing the "predictive power" of a clustering technique. They are a balance between an External and Internal Criterion: FOMs require no a priori knowledge, nor rely on information obtained entirely from the clustering process.

2. Theory

2.1. *Multivariate Regression Trees*

Multivariate Regression Trees were originally designed to simultaneously predict several continuous response variables using explanatory variables [1]. Initially, all observational units are contained in a single node. Regression Trees recursively partition the dataset. At each stage, a node is divided into two using a binary decision function that makes the two new subsets as homogenous as possible with respect to the response variables [19].

Regression Trees partition a node, t, into two subsets, t_L and t_R, on the basis of the value of an explanatory variable. All possible splits of each explanatory variable are considered. The optimal split is saved for each node. The node with the split that maximises the decrease in $R(T)$ is partitioned. $R(T)$ is given by:

$$R(T) = \frac{1}{n} \sum_{t \in \tilde{T}} \sum_{\boldsymbol{x}_i \in t} (\boldsymbol{y}_i - \bar{\boldsymbol{y}}(t))^2$$

where $\boldsymbol{x}_i$ is the vector of measurements of p explanatory variables for the i^{th} observational unit; $\boldsymbol{y}_i$ is the vector of measurements of the response variables for the i^{th} observational unit; $\tilde{T}$ is the set of all nodes and; $\bar{\boldsymbol{y}}(t)$ is the average response vector of node t.

We replicate the explanatory variables as the response variables. This allows a Multivariate Regression Tree, which requires response variables, to be grown using a dataset containing only explanatory variables. The observational units in each of the terminal nodes constitute the clusters of the dataset. By using identical response and explanatory variables the clusters will be as homogenous as possible with respect to all the variables. The variables that determine the split at the nodes will be the "important variables": that is, the variables that best define clusters of the dataset.

2.2. *K-means*

K-means is a partitioning clustering method. The method requires both the number of clusters, k, and initial estimates of cluster centres to be specified a priori. Each observational unit, $\boldsymbol{x}_i$, is visited in turn and assigned to the cluster whose centroid is closest, using a Euclidean distance metric. Cluster centroids are then updated. The process repeats until no more reassignments of the observational units occur.

2.3. *Mixture Models of Factor Analysers*

Mixture Models of Factor Analysers are an extension of Normal Mixture Models, a probability based clustering method. Mixture Models of Factor Analysers decrease the number of parameters that are estimated in a Normal Mixture Model, by performing dimension reduction (Factor Analysis) in each component.

The probability density function of an observational unit arising from a Mixture Model of Factor Analysers is taken to be a mixture of k multivariate normal component density functions in unknown proportions. The probability density function of $\boldsymbol{x}$ is given by

$$f(\boldsymbol{x},\psi)=\sum_{j=1}^{k}\pi_j\phi(\boldsymbol{x},\boldsymbol{\mu}_j,\Sigma_j) \qquad 0\le\pi_j\le 1 \quad \sum_{j=1}^{k}\pi_j=1$$

where π_j is the mixing proportion of the j^{th} component; $\phi(\boldsymbol{x},\boldsymbol{\mu}_j,\Sigma_j)$ is the multivariate normal density function of the j^{th} component; $\boldsymbol{\mu}_j$ is the j^{th} component mean; Σ_j is the

j^{th} component covariance matrix; ψ is the vector of unknown parameters $\pi_j, \boldsymbol{\mu}_j, \Sigma_j \quad j = 1,....,k$.

Factor Analysis is performed within each component. This allows each component covariance matrix to be modelled by $\Sigma_j = \Lambda_j \Lambda_j^T + \boldsymbol{D}_j$ where q is the number of factors (taken to be the same for each component); Λ_j is a $p \times q$ matrix of factor loadings for the j^{th} component; $\boldsymbol{D}_j$ is a diagonal matrix.

The parameters are estimated using the Expectation-Maximisation (EM) algorithm [20]. Each component describes one of the clusters. An outright clustering of the data into k groups is achieved by assigning each observational unit to the component for which it has the highest posterior probability of belonging [16].

The number of factors, q , here was varied to alter the degree of dimension reduction for the model. Using each chosen number of factor analysers, the model was fit twice to allow for different solutions obtained because of random starts.

2.4. *Plaid Models*

Plaid Models, a form of two-way, overlapping Cluster Analysis were first introduced by Lazzeroni and Owen [8]. Plaid Models represent the dataset as a sum of k terms, called "layers". Each layer represents a cluster, and observational units and variables can exist in more than one cluster or none at all. The $(i, j)^{th}$ element of the data matrix, $\boldsymbol{X}$, is represented algebraically as:

$$X_{ij} = \sum_{g=0}^{k} \rho_{ig} \kappa_{jg} \theta_{ijg}$$

where θ_{ijg} represents an ANOVA fit to the rows and columns in a layer - we used the fourth ANOVA type suggested by Lazzeroni and Owen [8]; $\rho_{ig}, \kappa_{jg} \in \{0,1\}$ $\rho_{ig} = 1$ if observational unit i is in layer g $\kappa_{jg} = 1$ if variable j is in layer g; $\rho_{ig}, \kappa_{jg} = 0$ otherwise.

A layer is sought at each stage. The Plaid Algorithm searches for a layer, G, in the current residual matrix, $\boldsymbol{Z}$, where $\boldsymbol{Z}$ is defined by:

$$Z_{ij} = X_{ij} - \theta_{ij0} - \sum_{g=1}^{G-1} \rho_{ig} \kappa_{jg} \theta_{ijg} \ .$$

The layer found minimizes the sum of squared errors, $\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{p}\left(Z_{ij} - \rho_{iG}\kappa_{jG}\theta_{ijG}\right)^{2}$. The parameters of the layer can be estimated iteratively using Lagrange Multipliers.

As Plaid Models extract layers that may be overlapping and/or not cover the entire dataset, the division of the dataset into k clusters had to be defined. Because this definition could be made arbitrarily, it was decided to make use of the known group memberships via the Adjusted Rand Index. It is hoped that this ensured Plaid Models were depicted fairly amongst the other techniques. Various definitions were trialled, and the cluster assignment rule was chosen as the rule that maximised the Adjusted Rand Index for each dataset. This resulted in different cluster assignment rules for the datasets. For the Vietnam dataset, observational units were assigned to the first layer they appeared in. For the Thyroid dataset, observational units were assigned to the last layer they appeared in.

There are various “preferences” available when finding a layer using the Plaid Models program [8]. We used the “unisign options” when clustering the Vietnam dataset, and the default parameters for the Thyroid dataset. Again, these preferences were chosen to maximise the External Statistic for each dataset (data not shown).

2.5. *Gap Statistic*

The Gap Statistic “compares the change in within cluster dispersion to that expected under an appropriate reference null distribution” [12]. The Gap Statistic chooses the optimal number of clusters as the value for which the within cluster dispersion curve falls farthest below a reference curve. The algorithm is discussed in more detail in Tibshirani et al. [12].

2.6. *External Criterion: Adjusted Rand Index*

The Adjusted Rand Index is an extension of the Rand Index [21] such that its expected value when comparing two random partitions is zero [18]. The maximum value of the Adjusted Rand Index is one, indicating perfect agreement between the classes and the clusters [18]. The Adjusted Rand Index is given by:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i\bullet}}{2} + \sum_j \binom{n_{\bullet j}}{2}\right] - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}$$

where n_{ij} = number of observational units in class i and cluster j; $n_{i\bullet}$ = number of observational units in class i; $n_{\bullet j}$ = number of observational units in cluster j; n = number of observational units.

2.7. *Internal Criterion: Point Biserial Correlation*

The Point Biserial Correlation is the Pearson Correlation between two variables with one of the variables being binary [9]. The dichotomous variable is the concatenated columns of the matrix that has its $(i,j)^{th}$ element set to zero if the observational units $\boldsymbol{x}_i, \boldsymbol{x}_j$ are clustered together by the algorithm. The non-dichotomous variable contains the values of the dissimilarity matrix. Large positive values close to one indicate a good fit between the data and the obtained clusters.

2.8. *Figure of Merits*

Figure of Merits (FOMs) assess the "predictive power" of a clustering algorithm by leaving out a variable, j, clustering the data (into k clusters), then calculating the Root Mean Square Error (RMSE) of j relative to the cluster means:

$$RMSE(j,k) = \sqrt{\frac{1}{n}\sum_{r=1}^{k}\sum_{\boldsymbol{x}_i \in C_r}\left(x_{ij} - \mu_{C_r}(j)\right)^2}$$

where x_{ij} is the measurement of the j^{th} variable on the i^{th} observational unit; n is the number of observational units; C_r is the set of observational units in the r^{th} cluster; $\mu_{C_r}(j)$ is the mean of variable j for the observational units in the r^{th} cluster.

Each variable is omitted and its RMSE calculated. These RMSE are summed over all variables to give an aggregate FOM (AFOM), $AFOM(k) = \sum_{j=1}^{p} RMSE(j,k)$. The AFOM is calculated for each k, and adjusted for cluster size. Here the AFOM was also divided by the number of variables "left out":

$$AFOM_{adj}(k) = \frac{1}{p\sqrt{\frac{n-k}{n}}} AFOM(k)$$

Low values of a clustering algorithm's AFOM indicate that the algorithm has high predictive power [17]. In this way, different clustering algorithms can be compared, by comparing their AFOM values at each k. However, Yeung et al. [17] comment that clustering algorithms should only be compared to one another using AFOMs if the algorithms' similarity metrics are identical. Therefore, we use AFOMs to compare K-means and AAMRTs to each other, and to illustrate how all four techniques are performing as the number of clusters increases.

3. Datasets and Analysis

3.1. *Vietnam dataset*

This dataset [22] consists of measurements of chemical elements in hair samples of six different groups of Vietnamese. There are 17 variables that have been log transformed and z-standardised: Ti, V, Cr, Mn, Ni, Cu, As, Se, Sr, Mo, Cd, Sn, Ba, Pb, Th, U, Hg. The groups differ in their amount of exposure to coal:

(1) Control Adults (C-A): 31 males with low exposure to coal

(2) Control Children (C-C): 31 children with low exposure to coal

(3) Miner Adults (M-A): 56 males employed at a coal mine

(4) Miner Children (M-C): 47 children of male coal miners

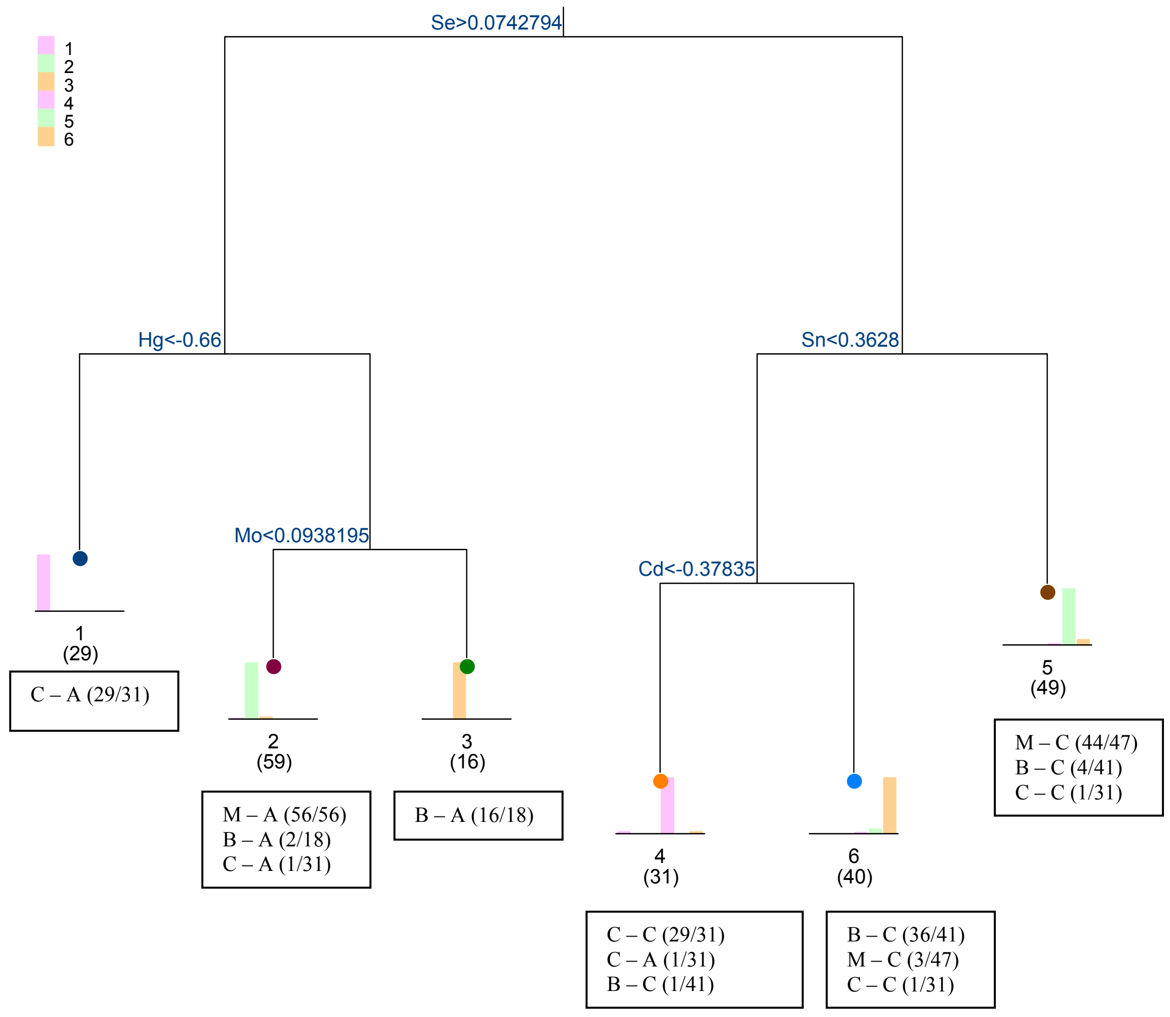
(5) Burner Adults (B-A): 18 females using coal for cooking

(6) Burner Children (B-C): 41 children with exposure to coal through its use for cooking

This group membership is used in the calculation of the Adjusted Rand Index. The six classes are well classified using a Classification Tree (Figure 1).

Figure 1.

Classification Tree of the Vietnam dataset



3.2. *Thyroid dataset*

This dataset [23] consists of measurements of five hormones (z-standardised) in patients with differing thyroid functions. The five hormones are T4, T3, RT3U, TSH, DTSH. The three classes are:

(1) Normal thyroid function (eu): 150 patients

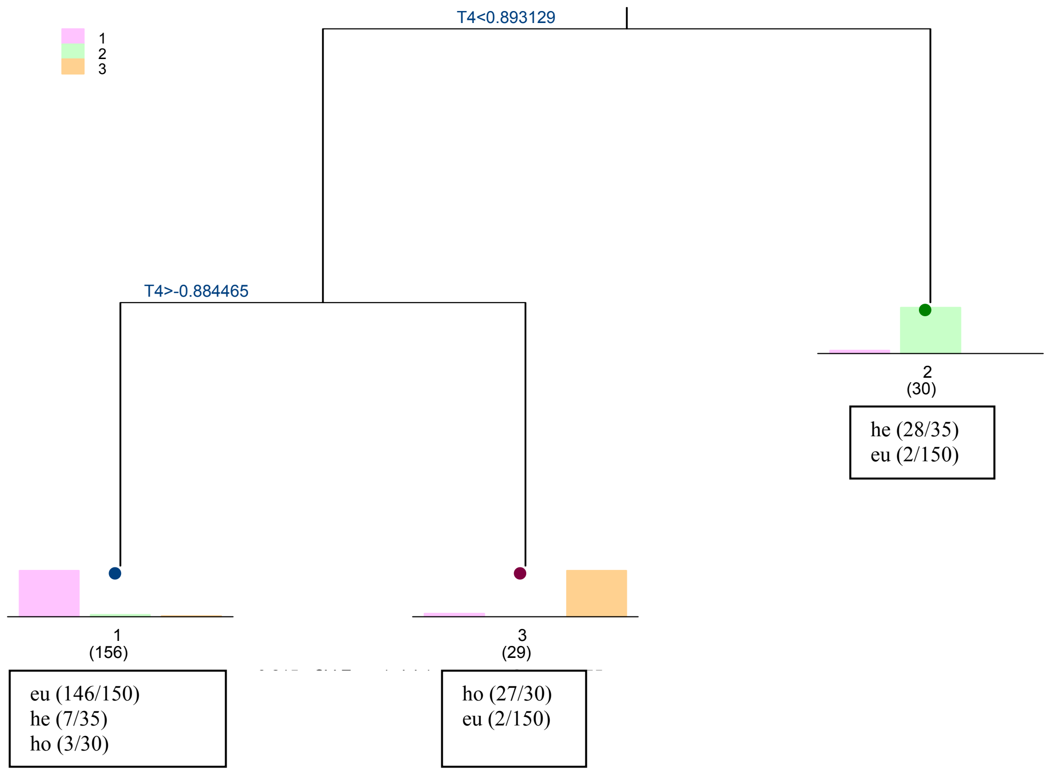
(2) Hyperthyroid function (he): 35 patients

(3) Hypothyroid function (ho): 30 patients

These classes are shown in a Classification Tree (Figure 2). In contrast to the Vietnam dataset, the classes are overlapping in the two-dimensional Fisher's linear discriminant space (data not shown). The eu cluster is dense and spherical, the other two clusters are elliptically shaped.

Figure 2.

Classification Tree of the Thyroid dataset



3.3. *Procedure*

For the Vietnam dataset, two to twenty clusters were extracted using the four techniques of interest, AAMRTs, K-means, Mixture Models of Factor Analysers and Plaid Models. The clusters obtained using each of the techniques were compared to each other using the comparative criteria. Because the Thyroid dataset is known to contain only three classes, the number of clusters extracted using each technique ranged from two up to ten.

4. Results

4.1. *Vietnam dataset*

The results of the criteria are shown in Table 1. When using the AAMRT, the Gap Statistic selected seven clusters as optimal. Interestingly, the “elbow” of the relative error curve occurred at $k = 5$ (Figure 3). The Gap Statistic selected six clusters when used in conjunction with K-means, but not using the Mixture Models of Factor Analysers or the Plaid Model. The BIC selected two or three clusters as optimal on the multiple runs of Mixture Models of Factor Analysers. The regularization procedure of Plaid Models indicated that fourteen layers were needed to describe the data.

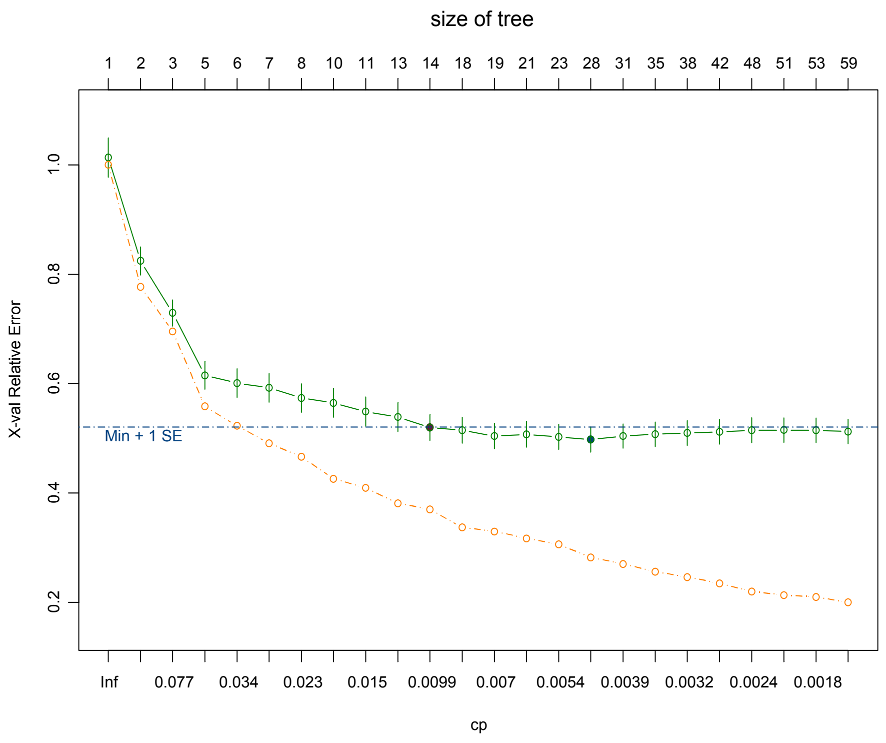
Table 1.

Results using the Vietnam dataset

	Gap	Max ARI	ARI $_{(k=6)}$	Max PBC	PBC $_{(k=6)}$	Other Criteria
AAMRT	7	0.6523 (7)	0.5751	0.4393 (5)	0.4119	Elbow -5
						Min+1SE -14
K-means	6	0.8159 (6)	0.8159	0.5430 (3)	0.5137	
MMFA $_{(q=2)}$	2	0.5991 (7)	0.4578	0.4639 (5)	0.4188	BIC – 3
MMFA $_{(q=2)}$	4	0.6075 (5)	0.5806	0.4476 (5)	0.4113	BIC – 3
MMFA $_{(q=3)}$	1	0.4877 (7)	0.4877	0.4594 (6)	0.4594	BIC – 3
MMFA $_{(q=3)}$	2	0.6921 (6)	0.6921	0.4735 (2)	0.4631	BIC – 2
MMFA $_{(q=4)}$	2	0.6322 (9)	0.4823	0.4457 (9)	0.4128	BIC – 2
MMFA $_{(q=4)}$	1	0.6025 (8)	0.4108	0.4736 (8)	0.4105	BIC – 3
PLAID	3	0.5431 (5)	0.5309	0.5294 (3)	0.4168	REG - 14

Figure 3.

Relative error curve for the Auto-Associative Multivariate Regression Tree grown on the Vietnam dataset[1]



The maximum value of the Adjusted Rand Index is shown in the third column of the table. The number of clusters associated with this value is in parentheses. The result of using a AAMRT was comparable to those obtained using Mixture Models of Factor Analysers. The values of the Adjusted Rand Index at $k = 6$ clusters (the known number of classes) are shown in the following column of the table. The values associated with Mixture Models of Factor Analysers, the Plaid Model and the AAMRT were similar to each other but less than that of the K-means technique.

The maximum value of the Internal Criterion is shown in the fifth column of Table 1; the number of clusters associated with this value is in parentheses. The maximum value

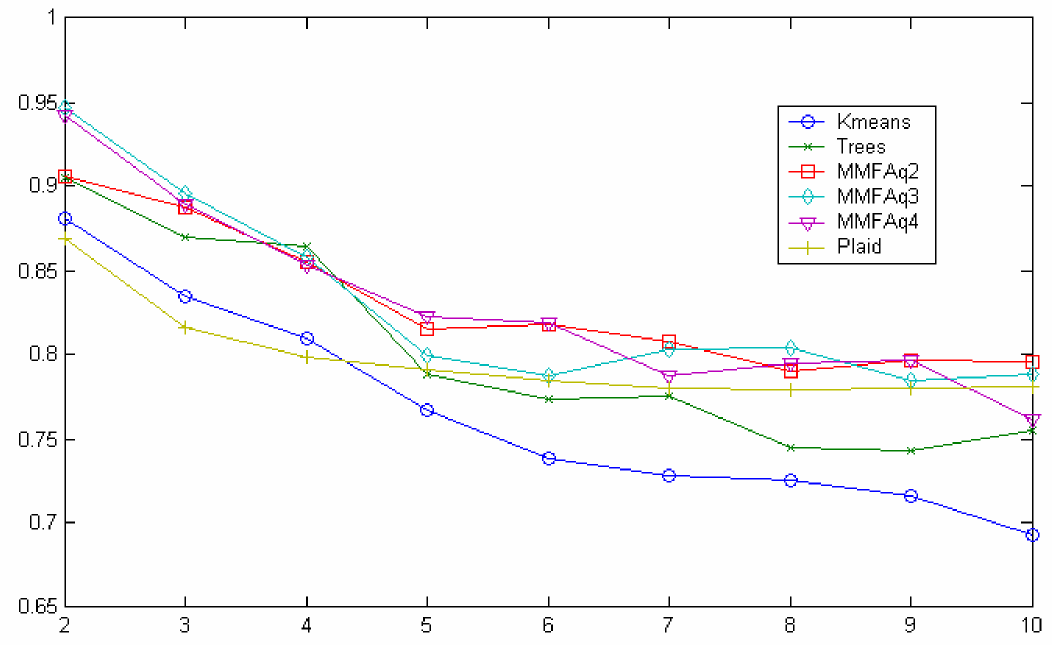
[1] See Ref [2] for an explanation of the complexity parameter (cp). The size of tree axis represents the number of terminal nodes.

of the Internal Criterion using a tree was comparable to the values obtained using Mixture Models of Factor Analysers. However, the Plaid Model and K-means values were higher than the Tree. The Internal Criterion values at $k = 6$ clusters are shown in the sixth column. Again, the value obtained using a AAMRT was comparable to those obtained using Mixture Models of Factor Analysers, but these values were not as high as the K-means Point Biserial Correlation.

Figure 4 shows the Figure of Merits for each of the four techniques. For clarity, repeated runs of the Mixture Models are not shown, nor are the results for more than ten clusters. Most lines plateau at 5 or 6 clusters. K-means achieved lower AFOM values (and therefore higher quality clusters) than the AAMRT. This complemented the results shown in Table 1.

Figure 4.

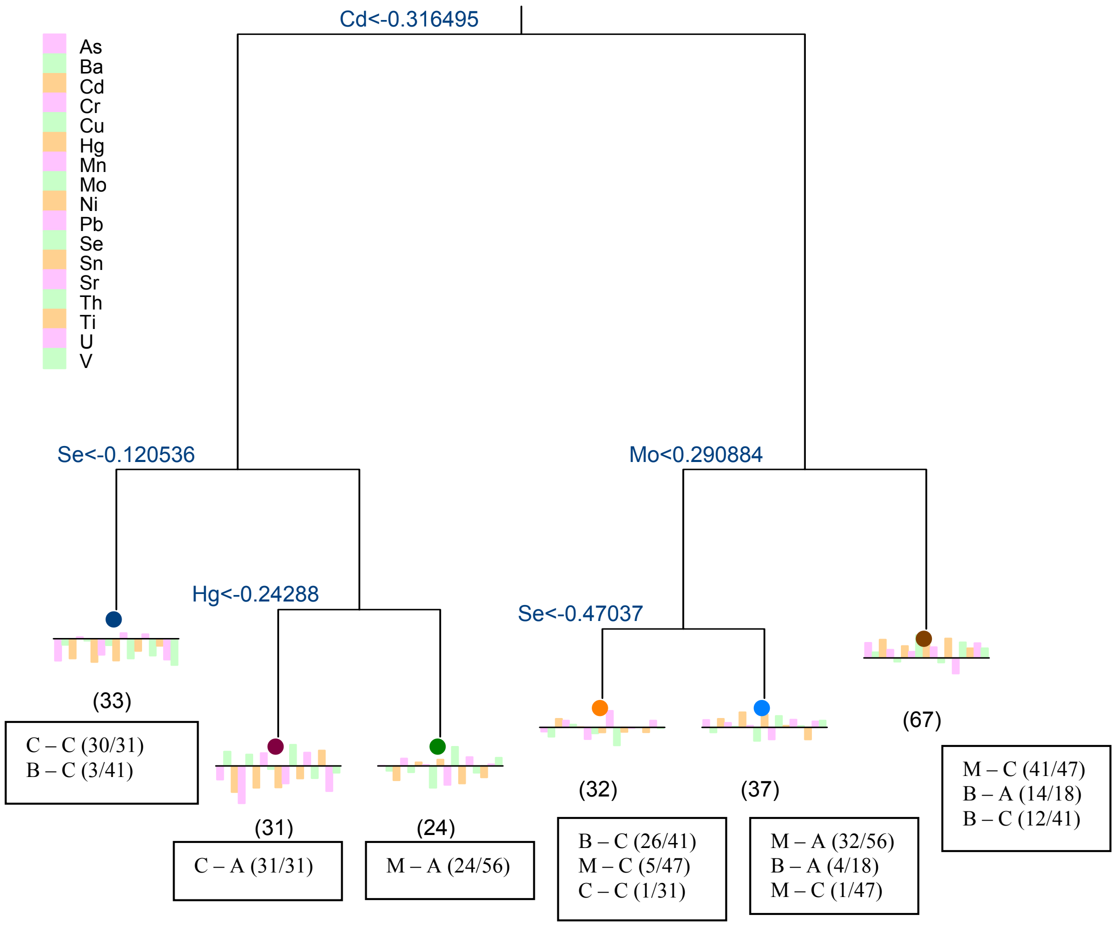
AFOM graphs for the Vietnam dataset



The variables used in the AAMRT were Cd, Se, Hg and Mo (Figure 5). These variables and Sn were also used in the Classification Tree (Figure 1). The first split used Cd and separated the control groups and some of the miner (adults) group from the other groups. This is in contrast to the Classification Tree (Figure 1), which initially used Se to separate the adults and children. The numbers of each group are shown at each of the terminal nodes of the tree.

Figure 5.

Auto-Associative Multivariate Regression Tree of the Vietnam dataset grown to six terminal nodes



4.2. *Thyroid dataset*

The Gap Statistic only selected one cluster using K-means, the AAMRT and the Plaid Model (Table 2). However, the Gap Statistic selected three clusters when using Mixture Models of Factor Analysers with one factor in each component. Although the cross-validated estimate of tree size was high, the relative error curve's "elbow" occurred at $k = 3$ (Figure 6). The BIC's estimates were varied.

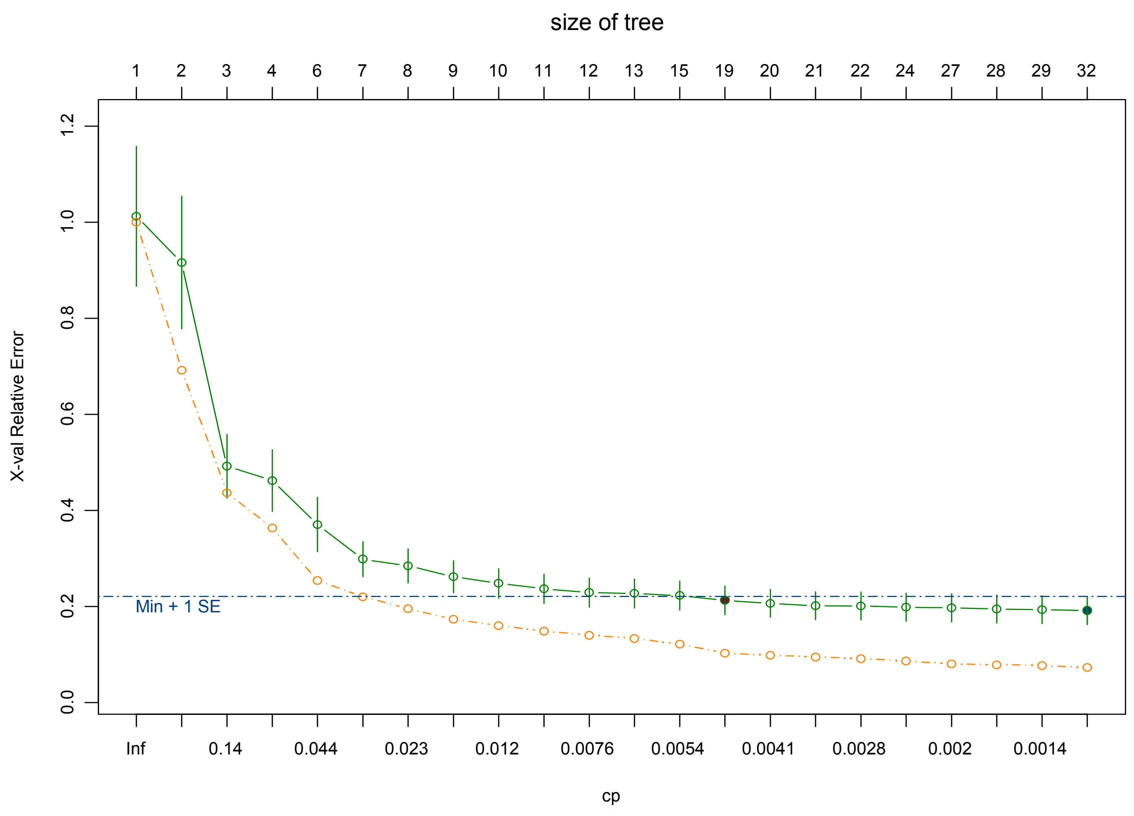
Table 2.

Results using the Thyroid dataset

	Gap	Max ARI	ARI $_{(k=3)}$	Max PBC	PBC $_{(k=3)}$	Other Criteria
AAMRT	1	0.5832 (3)	0.5832	0.7950 (5)	0.7829	Elbow - 3
						Min+1SE - 19
K-means	1	0.5832 (3)	0.5832	0.7877 (3)	0.7877	
MMFA $_{(q=1)}$	3	0.8149 (5)	0.7496	0.7175 (4)	0.7093	BIC – 7
MMFA $_{(q=1)}$	3	0.8231 (5)	0.7823	0.7079 (3)	0.7079	BIC – 10
MMFA $_{(q=2)}$	1	0.8763 (3)	0.8763	0.6759 (2)	0.6629	BIC – 3
MMFA $_{(q=2)}$	1	0.8578 (4)	0.5936	0.7138 (3)	0.7138	BIC – 4
PLAID	1	0.8373 (3)	0.8373	0.6522 (3)	0.6522	REG – 2

Figure 6.

Relative error curve for the Auto-Associative Multivariate Regression Tree grown on the Thyroid dataset

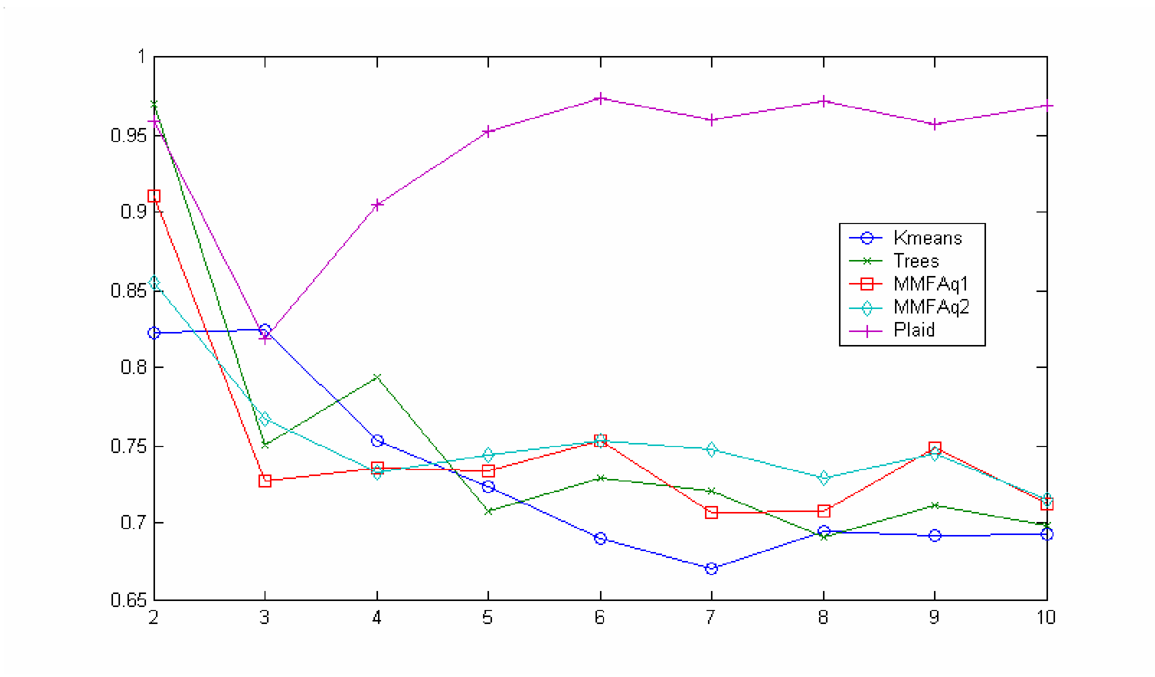


The values of the Adjusted Rand Index at $k = 3$ clusters (the known number of classes) obtained with the AAMRT and K-means were the lowest of the techniques (Table 2). The values obtained using the Plaid Model and Mixture Models of Factor Analysers were very high, indicating that these techniques found clusters very similar to the known groups of the data. The highest values of the Internal Criterion for $k = 3$ clusters occurred in conjunction with K-means and the AAMRT.

Figure 7 shows the Figure of Merits for each of the techniques. Four of the five lines decrease noticeably until three clusters. For $k > 3$ clusters, the Plaid Model's AFOM curve increases dramatically. This increase can be explained by the choice of cluster definition employed for the Plaid Model with the Thyroid dataset. After four or more layers were extracted, the predictive capability of the model decreased.

Figure 7.

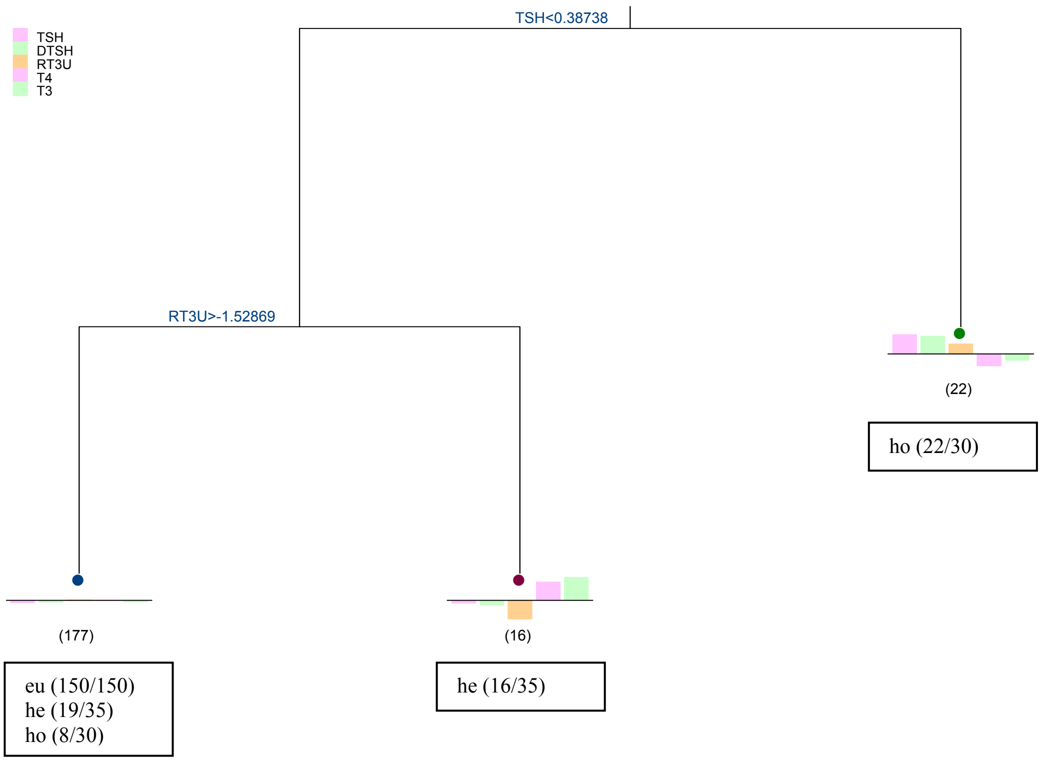
AFOM graphs for the Thyroid dataset



The variables of the Thyroid dataset used as splitting criteria in the corresponding AAMRT (with three terminal nodes – Figure 8) were TSH and RT3U. The root node was initially split on the variable TSH. Conversely, the Classification Tree (Figure 2) was split using only the variable T4. Although the AFOM graphs indicated that five clusters might exist in the data, the tree (not shown) became no more homogenous with respect to its terminal nodes. This result supports the Adjusted Rand Index which peaked at three clusters for the AAMRT.

Figure 8.

Auto-Associative Multivariate Regression Tree of the Thyroid dataset grown to three terminal nodes



5. Discussion

This study presented an opportunity to illustrate various descriptive and comparative criteria employed with cluster analyses whilst simultaneously investigating the capabilities of Auto-Associative Multivariate Regression Trees. The results were mixed. It is known that the Thyroid dataset contains overlapping clusters. The results of the Gap Statistic applied with the Thyroid dataset may indicate that the Gap Statistic has underestimated the true number of clusters in the dataset because of their overlapping nature, see for more information Tibshirani et al. [12].

The BIC and regularisation procedure also gave estimates of cluster number that differed from the "gold standard". However, for the Vietnam dataset with six groups, the elbow of the AAMRT’s relative error curve occurred at five clusters; for the Thyroid dataset the elbow coincided with the known number of groups.

AFOM graphs generally indicated the “correct number” of clusters for both datasets. However, the AFOM graph for the AAMRT indicated there may have been five clusters within the Thyroid dataset. Again, closer inspection of the tree showed its terminal nodes to be no more homogenous than those of the tree with three terminal nodes.

The External and Internal Criteria tended to peak at the known number of clusters for the Vietnam and Thyroid datasets. Because the External Criterion relies on known class membership, and the Internal Criterion uses the same information that forms the clusters to authenticate the clusters [17], these criteria should not be used individually as estimates of cluster quality.

Therefore, this study highlighted the necessity for a collection of criteria in order to estimate the number of clusters and cluster quality. No criterion could be used individually with these datasets and ensure a comprehensive explanation of the derived clusters. Using these criteria, we note that K-means outperformed the other techniques with regard to the Vietnam dataset. However, this was not the case with the Thyroid dataset. Here, the AAMRT performed similarly to K-means, but the Plaid Model and Mixture Models of Factor Analysers were the “winners” when considering the Adjusted Rand Index. The conclusion to be drawn from these results is that there was neither an identifiably superior nor inferior method, but that AAMRTs presented a viable clustering solution.

6. Conclusions

This was a preliminary investigation assessing the suitability of AAMRTs as a clustering method using datasets containing only explanatory variables. The "cluster trees" were compared to other successful clustering techniques, K-means, Mixture Models of Factor Analysers and Plaid Models. The results showed the Trees produced clusters of similar quality to the other techniques. The advantages of AAMRTs as a clustering technique include their capabilities to present a visually interpretable and intuitive cluster analysis. This study has shown that AAMRTs produce a sensible cluster analysis in the low dimensional setting. Investigations of the suitability of AAMRTs as a clustering technique in the high dimensional setting are in progress. The study also highlighted the need for a "panel" of criteria in order to estimate the number of clusters and cluster quality.

7. Software

R (available from http://www.R-project.org.) was used to implement the K-means algorithm. MATLAB 6.1 (The MathWorks, Inc.) code created by Ghahramani and Hinton [6] (available at ftp://ftp.cs.toronto.edu/pub/zoubin/mfa.tar.gz) was used to fit the Mixture Model of Factor Analysers. This algorithm fits a common diagonal matrix $\boldsymbol{D}$ (that is, $\boldsymbol{D}_j = \boldsymbol{D}$ for $j = 1, ..., k$). Plaid™, available from http://www-stat.stanford.edu/~owen/clickwrap/plaid.html, was used to fit the Plaid Models. TreesPlus [5], an add-in module for S-PLUS 2000, was used to obtain the Multivariate Regression Trees. The module is a wrapper of the R-PART [24] code for Trees. MATLAB 6.1 (The MathWorks, Inc.) and R were used to generate the Gap Statistic and other comparative criteria.

References

[1] M. R. Segal, Tree-Structured Methods for Longitudinal Data, Journal of the American Statistical Association 87 (1992) 407-418.
[2] L. Breiman, J. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.
[3] F. Questier, R. Put, D. Coomans, B. Walczak, Y. V. Heyden, The use of CART and multivariate regression trees for supervised and unsupervised feature selection, Chemometrics and Intelligent Laboratory Systems 76 (2005) 45-54.
[4] D. R. Larsen, P. L. Speckman, Multivariate regression trees for analysis of abundance data, Biometrics 60 (2004) 543-549.
[5] G. De'Ath, Multivariate regression trees: a new technique for modeling species-environment relationships, Ecology 83 (2002) 1105-1117.
[6] Z. Ghahramani, G. Hinton, The EM Algorithm for Mixtures of Factor Analyzers.CRG-TR-96-1.University of Toronto. (1996).
[7] G. J. McLachlan, D. Peel, Mixtures of Factor Analyzers, the 17th International Conference on Machine Learning, San Francisco, USA, 2000.
[8] L. Lazzeroni, A. Owen, Plaid models for gene expression data, Statistica Sinica 12 (2002) 61-86.
[9] G. W. Milligan, An examination of the effect of six types of error perturbation on fifteen clustering algorithms, Psychometrica 45 (1980) 325-342.
[10] S. Monti, P. Tamayo, J. Mesirov, T. Golub, Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data, Machine Learning 52 (2003) 91-118.
[11] L. Kaufman, P. Rousseeuw, Finding groups in data: an introduction to cluster analysis, Wiley, New York, 1990.
[12] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the Gap statistic, Journal of the Royal Statistical Society Series B-Statistical Methodology 63 (2001) 411-423.
[13] J. Hartigan, Clustering algorithms, Wiley, New York, 1975.
[14] A. Ben-Hur, A. Elisseeff, I. Guyon, A stability based method for discovering structure in clustered data, the Pacific Symposium on Biocomputing, Lihue, Hawaii, USA, 2002.
[15] S. Dudoit, J. Fridlyand, A prediction-based resampling method for estimating the number of clusters in a dataset, Genome Biology 3 (2002) 0036.0031-0036.0021.
[16] G. J. McLachlan, R. W. Bean, D. Peel, A mixture model-based approach to the clustering of microarray expression data, Bioinformatics 18 (2002) 413-422.
[17] K. Y. Yeung, D. R. Haynor, W. L. Ruzzo, Validating clustering for gene expression data, Bioinformatics 17 (2001) 309-318.
[18] L. Hubert, P. Arabie, Comparing Partitions, Journal of Classification 2 (1985) 193-218.
[19] A. Ploner, C. Brandenburg, Modelling visitor attendance levels subject to day of the week and weather: a comparison between linear regression models and regression trees, Journal of Nature Conservation 11 (2004) 297-309.
[20] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm, Journal of the Royal Statistical Society Series B-Statistical Methodology 39 (1977) 1-38.
[21] W. M. Rand, Objective Criteria for the Evaluation of Clustering Methods, Journal of the American Statistical Association 66 (1971) 846-850.

[22] H. T. Nguyen, D. Coomans, M. Leermakers, J. Boman, Multivariate statistical analysis of human exposure to trace elements from coal in Vietnam, SPRUCE IV International Conference on Statistical Aspects of Health and the Environment, Enschede, the Netherlands 1997.
[23] D. Coomans, M. Jonckheer, D. L. Massart, I. Broeckaert, P. Block, The application of linear discriminant analysis in the diagnosis of thyroid diseases, Analytica Chimica Acta 103 (1978) 409-415.
[24] Therneau, Atkinson, RPART software, (1998).

Clustering Noisy Data in a Reduced Dimension Space via Multivariate Regression Trees

Christine Smyth[*], Danny Coomans, and Yvette Everingham

Statistics and Intelligent Data Analysis Group,

School of Mathematical and Physical Sciences,

James Cook University,

Australia

Pattern Recognition

Volume 39

Pages 424-431

2006

ABSTRACT

Cluster Analysis is sensitive to noise variables intrinsically contained within high dimensional datasets. As the size of datasets increases, clustering techniques robust to noise variables must be identified. This investigation gauges the capabilities of recent clustering algorithms applied to two real datasets increasingly perturbed by superfluous noise variables. The recent techniques include Mixture Models of Factor Analysers and Auto-Associative Multivariate Regression Trees. Statistical techniques are integrated to create two approaches useful for clustering noisy data: Multivariate Regression Trees with Principal Component Scores and Multivariate Regression Trees with Factor Scores. The tree techniques generate the superior clustering results.

Keywords: Cluster Analysis, Noise Variables, Multivariate Regression Trees, Dimension Reduction

1. Introduction

A characteristic common to many datasets is “noise variables”. Noise variables contain no relevant information and mask the underlying structure of the dataset. The prevalence of noise variables in datasets is increasing: an unavoidable consequence as the size of datasets increases.

It is known that care must be taken when clustering large noisy datasets because including superfluous variables may induce spurious clusters or blur existing cluster boundaries. The researcher must use a clustering algorithm suitable for noisy high dimensional datasets. These clustering algorithms will intrinsically incorporate dimension reduction.

Here we trial auto-associative multivariate regression trees [1] using noisy data. We further extend multivariate regression trees as a clustering technique by incorporating dimension reduction, and demonstrate their capabilities when clustering noisy data. We reduce the dimension of the dataset globally using either factor or principal components analysis, and subsequently cluster in the reduced factor or principal components space via the regression tree. The capabilities of mixture models of factor analysers [2,3], a clustering technique featuring local dimensionality reduction, are also investigated.

We assess the potential of these algorithms when clustering noisy datasets, by perturbing two datasets via the introduction of superfluous variables. Clustering techniques are applied to the perturbed dataset and their capabilities to recover the known clusters are gauged. This error perturbation experiment has been used previously by Milligan [4]. We extend Milligan’s experiment to include real-life datasets, whilst benchmarking to the classical K-means technique. Our results show the superiority of multivariate regression trees with principal component scores and/or factor scores when clustering noisy data.

2. Theory

2.1. Principal Components Analysis and Factor Analysis

Both principal components analysis and factor analysis are dimension reduction techniques. Principal components analysis attempts to model the total variance of the original dataset, via new uncorrelated variables called principal components [5]. The principal components are linear combinations of the original variables:

$$\boldsymbol{y} = A^T \boldsymbol{x}$$

where $\boldsymbol{x}$ is a vector of the original variables; $\boldsymbol{y}$ is a p element vector of principal component scores; and A is obtained from the spectral decomposition of Σ.

There are p principal components. However, the first q principal components usually account for most of the variation within the dataset. Dimension reduction is achieved by discarding the latter $p-q$ principal components. Then Σ is approximated by its first q eigenvectors. Observational units are represented in the q dimensional subspace via the first q principal component scores.

Factor Analysis attempts to explain the variables by assuming that they can be generated as a linear combination of q unobservable common factors (usually $q << p$) plus a unique factor [5]. The factor analysis model is given by

$$\boldsymbol{x} = \boldsymbol{\mu} + F\boldsymbol{z} + \boldsymbol{\varepsilon} \tag{1.1}$$

where μ is a mean vector; F is a $p \times q$ matrix of factor loadings; z is a q dimensional vector of hypothetical common factors; and ε is a unique factor. Because the z are hypothetical, imposing assumptions $z \sim N(0, I_q)$ and $\varepsilon \sim N(0, D)$ allows the estimation of F.

We see that unlike principal components analysis, factor analysis distinguishes between common and unique variance. The factor analysis model implies that $\Sigma = FF^T + D$. The $p \times q$ matrix F contains the factor loadings. The factor loadings are the correlations of a variable with a common factor z [6]. D, a diagonal matrix, contains the specific variances of each variable: the unique variance of each variable that is not associated with the other variables. Therefore, the $p \times p$ covariance matrix Σ is modelled by a $p \times q$ matrix F and a diagonal matrix D, implying a substantial amount of dimension reduction if $q << p$.

Unlike principal components analysis, equation (1.1) shows that factor analysis does not provide a unique transformation from factors to variables. In fact, the solution can be rotated to make it more interpretable. Observational units can be represented in the q dimensional factor space by the estimated values of the hypothetical common factors, called “factor scores”.

As a dataset increases in size, the factor analysis solution is likely to approach the principal components solution [5]. Despite this fact, we use both factor scores and principal component scores as response variables in the multivariate regression trees and investigate any differences.

2.2. Multivariate Regression Trees and Auto-Associative Multivariate Regression Trees

Regression Trees begin with all the data contained within a single node. The root node is then split in two on the basis of the value of an explanatory variable so as to make the two new nodes more homogenous with respect to the response variables. The splitting process is continued until the terminal nodes (nodes not split in two) are sufficiently homogenous.

Mathematically, the binary decision function that spits a node is chosen such that it maximises the decrease in $R(T)$ [7]. $R(T)$ is given by:

$$R(T) = \frac{1}{n}\sum_{t\in\tilde{T}}\sum_{\boldsymbol{x}_i\in t}(\boldsymbol{y}_i - \overline{\boldsymbol{y}}(t))^2 \qquad (1.2)$$

where $\boldsymbol{x}_i$ is the vector of measurements of p explanatory variables for the i^{th} observational unit; $\boldsymbol{y}_i$ is the vector of measurements of the response variables for the i^{th} observational unit; $\tilde{T}$ is the set of all nodes and; $\overline{\boldsymbol{y}}(t)$ is the average response vector of node t.

Observational units within a terminal node are similar to each other with respect to the response variables. By replicating the explanatory variables as the response variables (that is, using identical response and explanatory variables) an auto-associative multivariate regression tree can be used as a divisive clustering technique.

We suggest a relaxed criterion for selecting the natural number of clusters found by an auto-associative multivariate regression tree: the “elbow” of the tree’s relative error curve. This tree will not attain optimal predictive performance, but any further splitting will result in only a small decrease in the heterogeneity of the terminal nodes. Therefore, at the number of nodes indicated by the location of the elbow, the clusters are sufficiently homogenous.

The location of the elbow is questionable. We deemed the elbow as the point, $\hat{k}$, where the gradient of the relative error curve changed from being steep to gentle. Specifically, we chose $\hat{k}$ as the k that minimised

$$abs\left(\frac{RE(k+1)-RE(k)}{RE(k)-RE(k-1)}\right)$$

where $RE(k)$ is the value of the relative error curve at k.

2.2.1. Multivariate Regression Trees with Principal Component Scores and Factor Scores

Clustering via an auto-associative multivariate regression tree by replicating the explanatory variables as response variables is computationally intensive if the dataset is large. Moreover, including redundant variables as response variables may induce suboptimal results. Reducing the dimension of the response variables may produce more stable results. We propose two techniques for reducing the dimension of the response space:

(1) Using principal component scores for q principal components as response variables. The reader will remember that these are essentially a linear transformation of the original data.

(2) Using factor scores for q factors as response variables. Factor scores are estimated after obtaining a factor loadings matrix, such that they "explain" the observational units.

A variety of q (see procedure section) are trialled to investigate the effect of amount of dimension reduction on the results.

The algorithm splits nodes so that the new nodes are homogenous with respect to the scores. Therefore, if q is taken to be smaller than p, the clustering is achieved in the reduced dimension factor/principal component space. It must be noted that unlike mixture models of factor analysers (see section 2.3.1), where the dimension reduction occurs locally within a cluster, here the dimension reduction occurs globally (that is, over the entire dataset).

Finally, the factor scores were generated using maximum likelihood factor analysis. Maximum likelihood factor analysis assumes that the data are multivariate normally distributed. The superfluous noise variables used to perturb the dataset are not all multivariate normal. Therefore, to avoid violating an assumption of maximum likelihood factor analysis, the multivariate regression tree with factor scores technique is tested only with the noise variables (type II) specifically generated to be multivariate normal (see procedure section).

2.3. Mixture Models

The probability density function of an observational unit arising from a mixture model is

$$f(\boldsymbol{x},\Psi)=\sum_{j=1}^{k}\Pi_j\Phi_j(\boldsymbol{x},\Psi) \tag{1.3}$$

Equation (1.3) shows that the overall density function $f(\boldsymbol{x},\Psi)$ is the sum of density functions $\Phi_j(\boldsymbol{x},\Psi)$ in proportions Π_j. Each $\Phi_j(\boldsymbol{x},\Psi)$ is commonly referred to as a component density function. Mixture models assume that each component density function describes a cluster. Observational units are assigned to the component for which they have the highest posterior probability of belonging to achieve an outright clustering of the data [3]. Therefore, mixture models are a probability based clustering method.

Parameters of the model, Π_j and Ψ, are estimated using the Expectation Maximisation (EM) algorithm [8]. Component density functions can be of various forms, but most commonly they are assumed to be multivariate normal. Therefore, the elements of Ψ include $\boldsymbol{\mu}_j$, the component (cluster) means and Σ_j, the component covariance matrices.

2.3.1. Mixture Models of Factor Analysers

Mixture models of factors analysers [2,3] incorporate factor analysis into each component density function, $\Phi_j(\boldsymbol{x}, \Psi)$, of a mixture model. Each component (cluster) covariance matrix is represented by $\Sigma_j = F_j F_j^T + D_j$ where F_j is a $p \times q$ factor loadings matrix for that component. Therefore, dimensionality reduction is performed locally within a cluster. In a normal mixture model there are $p(p+1)/2$ parameters to be estimated for each component covariance matrix, whereas a mixture model of factor analysers requires only $pq + p$ parameters for each component covariance matrix. This reduces the overfitting problem associated with normal mixture models. A variety of q (see procedure section) were trialled to investigate the effect of amount of dimension reduction on the results.

The Bayesian Information Criterion (BIC) is useful when estimating the number of components (clusters) required in a mixture model, although regularity conditions do not hold [3]. These estimates are also included to supplement the estimates of the silhouette statistic (section 2.5).

Mixture models of factor analysers require a random start. It is very possible that an optimal model will not be found using a single random start. We used 50 random starts. The adjusted rand index values (see section 2.6) reported are the average of the top ten adjusted rand index values attained by the 50 random starts. The estimates of the BIC and the silhouette statistic (see section 2.5) reported are those most frequently occurring over the 50 random starts.

2.4. K-means

K-means is a partitioning clustering method. The method requires both the number of clusters, k, and initial estimates of cluster centres to be specified a priori. Each observational unit, $\boldsymbol{x}_i$, is visited in turn and assigned to the cluster whose centroid is closest, using a Euclidean distance metric. Cluster centroids are then updated. The process repeats until no more reassignments of the observational units occur.

2.5. Silhouette Statistic

The silhouette statistic [9] can be used to assess the optimal number of clusters within a dataset. The silhouette statistic for observational unit $\boldsymbol{x}_i$ is given by

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\left(a(x_i), b(x_i)\right)}$$

where $a(x_i)$ is the mean distance between $\boldsymbol{x}_i$ and all the observational units within the same cluster; $b(x_i)$ is the mean distance between $\boldsymbol{x}_i$ and all observational units in the next nearest cluster (the nearest cluster is the cluster with the minimum $b(x_i)$). The optimal number of clusters is the number of clusters that maximises the average $s(x_i)$ for the entire dataset.

2.6. External Criterion: Adjusted Rand Index

An external criterion uses known class membership to provide an estimate of the quality of the obtained clusters [4]. The external criterion used in this study was the adjusted rand index [10]. The adjusted rand index is an extension of the Rand [11] Index such that its expected value when comparing two random partitions is zero. The maximum value of the adjusted rand index is one, indicating perfect agreement between the classes and the clusters [10]. The adjusted rand index is given by:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}{1/2\left[\sum_i \binom{n_{i\bullet}}{2} + \sum_j \binom{n_{\bullet j}}{2}\right] - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}$$

where n_{ij} = number of observational units in class i and cluster j; $n_{i\bullet}$ = number of observational units in class i; $n_{\bullet j}$ = number of observational units in cluster j; n = number of observational units.

3. Procedure

Two real datasets (the "Vietnam dataset" and the "Thyroid dataset") underpinned by biochemical phenomena were used in this analysis. The datasets are described in more detail in sections 3.1 and 3.2. There were 12 additions of noise variables in total for each dataset. Four types of noise variables were appended to these datasets in three increments. The three increments were: p, 50, and 100 (where p is the number of "real" variables in the dataset) variables. The four types of noise variables were:

(I) The variable was uniformly distributed between the minimum and maximum of the true dataset.

(II) The variable was normally distributed with mean zero and standard deviation one.

(III) The variable was uniformly distributed between the minimum and maximum of the dataset. However, there existed an interval $[a,b]$ between which the observational units were not allowed to lie. The parameters, a,b were chosen randomly with the conditions $b-a \geq 2$, and $\min(dataset) < a,b < \max(dataset)$.

(IV) Combination set of types I, II, and III.

After each addition of noise variables to the datasets, clusters were extracted using the techniques. The quality of the clusters was assessed using the criteria outlined in the previous section.

3.1. Vietnam dataset

This dataset [12] consists of measurements of chemical elements in hair samples of six different groups of Vietnamese. There are 17 variables that have been log transformed and z-standardized: Ti, V, Cr, Mn, Ni, Cu, As, Se, Sr, Mo, Cd, Sn, Ba, Pb, Th, U, Hg. The groups differ in their amount of exposure to coal:

(1) Control Adults (C-A): 31 males with low exposure to coal

(2) Control Children (C-C): 31 children with low exposure to coal

(3) Miner Adults (M-A): 56 males employed at a coal mine

(4) Miner Children (M-C): 47 children of male coal miners

(5) Burner Adults (B-A): 18 females using coal for cooking

(6) Burner Children (B-C): 41 children with exposure to coal through its use for cooking

Using this dataset, the values of the parameter q (required for mixture models of factor analysers, multivariate regression trees with principal component scores, and multivariate regression trees with factor scores) are $2, 3, 4, 5, 6, 7, 9, 12$.

3.2. Thyroid dataset

This dataset [13] consists of measurements of five hormones (z-standardized) in patients with differing thyroid functions. The five hormones are T4, T3, RT3U, TSH, DTSH. The three classes are:

(1) Normal thyroid function (eu): 150 patients

(2) Hyperthyroid function (he): 35 patients

(3) Hypothyroid function (ho): 30 patients

Using this dataset, the values of the parameter q are $1, 2, 3, 4, 5, 6$.

4. Results

4.1. Vietnam Dataset

The algorithms' adjusted rand index values at $k = 6$ clusters (the known number of classes) after the three additions of type I noise are illustrated in Fig. 1. Results are reported for the dimension (q=2) resulting in the highest adjusted rand index values for the mixture models of factor analysers (MMFA). The multivariate regression trees with principal component scores (MRTPCS) tended to produce similar adjusted rand index values across all dimensions of the principal component space. We show the most frequently occurring adjusted rand index value for these trees.

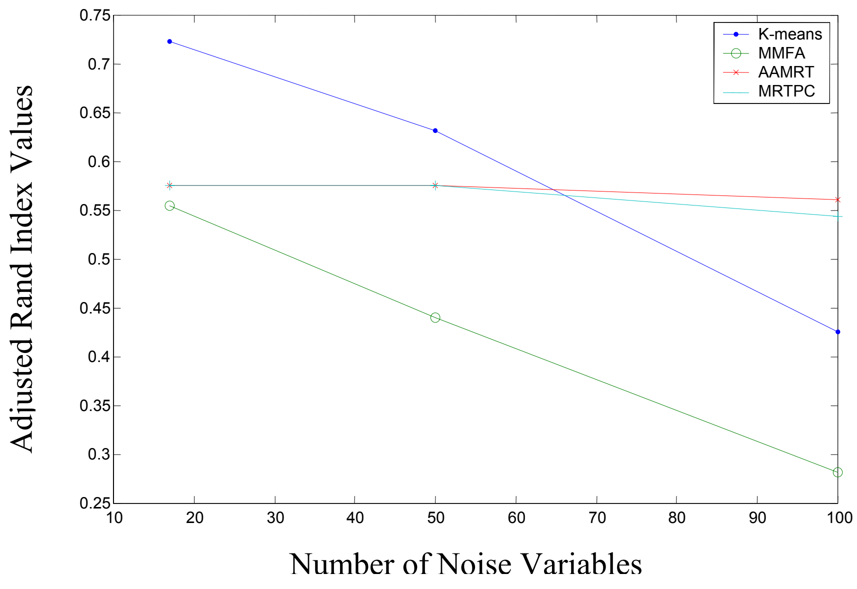


Fig. 1. Adjusted rand index values at k=6 clusters for the Vietnam dataset perturbed by type I noise variables.

The figure indicates that as the amount of noise increases, K-means and mixture models of factor analysers are adversely affected. Both tree techniques (AAMRT and MRTPCS) produce more stable results than K-means and the mixture models of factor analysers. The tree techniques maintain their adjusted rand index values, despite the increase in the number of superfluous noise variables.

Fig. 2 shows the algorithms' adjusted rand index values at $k = 6$ clusters when the dataset was perturbed by type II noise variables. Results are reported for the dimension resulting in the highest adjusted rand index values for the mixture models of factor analysers (q=2) and multivariate regression trees with factor scores (q=3) (MRTFS). The adjusted rand index values plotted for the multivariate regression trees with principal component scores are those most frequently occurring. The figure again indicates the ability of the tree techniques to recover the known clusters despite large error perturbation of the dataset. All tree techniques produce stable results. Multivariate regression trees grown using factor scores produce the highest quality clusters of the data. Conversely, K-means and mixture models of factor analysers are again adversely affected by the larger numbers of noise variables.

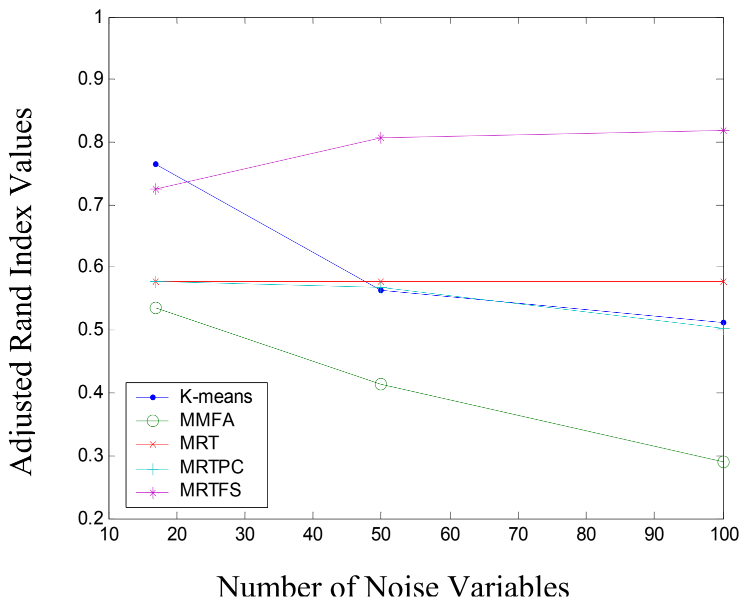


Fig. 2. Adjusted rand index values at k=6 clusters for the Vietnam dataset perturbed by type II noise variables.

Table 1 augments Fig. 2. Here, only the adjusted rand index values of the techniques using dimension reduction are reported. The effect of the amount of dimension reduction on an algorithm's capabilities to recover the known clusters is different for each technique. Mixture models of factor analysers attain optimal results by using two factors within each component. The multivariate regression trees with factor scores attain optimal results by clustering in the three-dimensional factor space. The multivariate regression trees with principal component scores are impervious to the amount of dimension reduction, attaining stable results across the range of numbers of principal components. The table highlights the superiority of the multivariate regression tree with factor scores technique. Their adjusted rand index values are the highest and show that these trees produce consistent results regardless of the number of extraneous noise variables.

Table 1. Adjusted rand index values at k=6 clusters using the Vietnam Dataset

Number of Type II Noise Variables	*Number of Factors*	*MMFA*[a]	*MRTPCS*	*MRTFS*
17	2	.5346	.5719	.6848
	3	.5020	.5751	.7242
	4	.3962	.5768	.6285
	5	.4212	.5768	.5151
	6	.4042	.5768	.6775
	7	.3890	.5694	.6606
	9	.3104	.5694	.6661
	12	.3049	.5694	.5660
50	2	.4150	.6103	.6711
	3	.3360	.5690	.8078
	4	.2523	.5690	.6725
	5	.2141	.5690	.6791
	6	.2113	.5690	.6629
	7	.1884	.5690	.6661
	9	.1864	.5690	.6661
	12	.1615	.5690	.4709
100	2	.2891	.5035	.6711
	3	.2410	.5656	.8185
	4	.1662	.5656	.6255
	5	.1432	.5705	.6796
	6	.1250	.5705	.6595
	7	.1291	.5035	.6646
	9	.0998	.5035	.5079
	12	.1071	.5002	.5050

The estimates of the silhouette statistic are shown in Table 2. The estimates are stable across all techniques. The multivariate regression trees with factor scores produce clustering results that the silhouette statistic deems closest to the natural number of clusters. The BIC indicates that only one cluster is necessary to describe the data (Table 2). The elbows of the tree techniques mostly occur in the same position as the number of clusters indicated by the silhouette statistic.

[a] Average of top ten adjusted rand index values from 50 random starts

The results using type III noise variables and the combination sets were similar to those already presented. For clarity, we have not reported these results.

Table 2. Estimates of cluster number for the Vietnam dataset.

		Silhouette Statistic			*BIC/ELBOW*[b]		
Number of Type II Noise Variables		17	50	100	17	50	100
K-means		2	2	2	/	/	/
AAMRT		2	2	2	2	2	2
MMFA[c]	2	2	2	2	2	1	1
	3	2	2	2	1	1	1
	4	2	2	2	1	1	1
	5	2	2	2	1	1	1
	6	2	2	2	1	1	1
	7	2	2	2	1	1	1
	9	2	2	2	1	1	1
	12	2	2	2	1	1	1
MRTPCS	2	2	2	2	2	2	2
	3	2	2	2	2	2	2
	4	2	2	2	2	2	2
	5	2	2	2	2	2	2
	6	2	2	2	2	2	2
	7	2	2	2	2	2	2
	9	2	2	2	2	4	2
	12	2	2	2	2	4	2
MRTFS	2	4	4	4	3	3	3
	3	4	4	4	4	4	4
	4	4	3	4	4	4	4
	5	4	4	4	5	4	4
	6	4	4	4	4	4	4
	7	3	4	4	4	4	4
	9	4	4	2	4	4	4
	12	5	3	4	4	4	4

[b] BIC estimates for the mixture models of factor analysers. Elbow estimates are for the multivariate regression trees with principal component scores and factor scores

[c] Most frequently occurring estimate from 50 random starts

4.2. Thyroid dataset

Fig. 3 shows the algorithms' adjusted rand index values at $k = 3$ clusters (the known number of classes) after the Thyroid dataset was increasingly perturbed by type I noise. The results for the mixture models of factor analysers are reported for the number of factors (q=1) that produced the highest adjusted rand index values. The multivariate regression trees with principal component scores tended to produce similar adjusted rand index values across all dimensions of the principal component space. We show the most frequently occurring adjusted rand index value for these trees. The figure shows that the mixture models of factor analysers initially recover the known clusters better than the other techniques. However, as the amount of noise introduced into the dataset increases, the mixture models of factor analysers are unable to maintain their superiority. K-means is also adversely affected by the perturbation of the dataset. The tree techniques are stable: the adjusted rand index values do not decrease for either technique.

The results using Type II noise variables are shown in Fig. 4. The multivariate regression tree with factor scores is also shown. The results are similar to those in Fig. 2. Clearly, the multivariate regression tree with factor scores is the superior technique.

Table 3 shows trends similar to Table 1. The adjusted rand index values for the mixture models of factor analysers decrease as the amount of dimension reduction in a component decreases. The multivariate regression trees with principal component scores produce stable adjusted rand index values irrespective of the dimension of the response space. The multivariate regression trees with factor scores produce optimal results using only a single dimension response space. As the dimension of the factor space increases, the adjusted rand index values decrease.

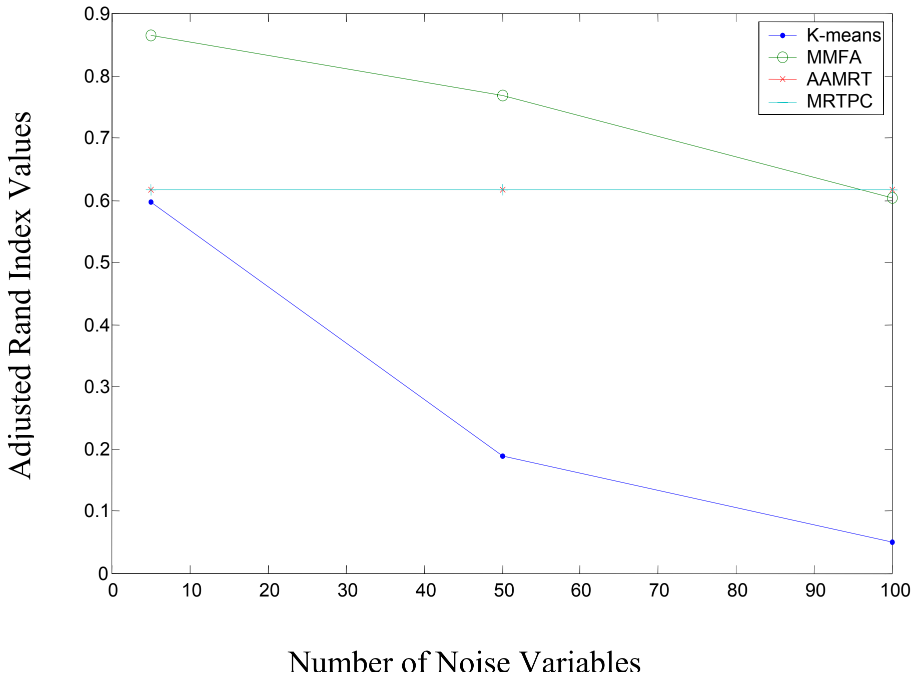


Fig. 3. Adjusted rand index values at k=3 clusters for the Thyroid dataset perturbed by type I noise variables.

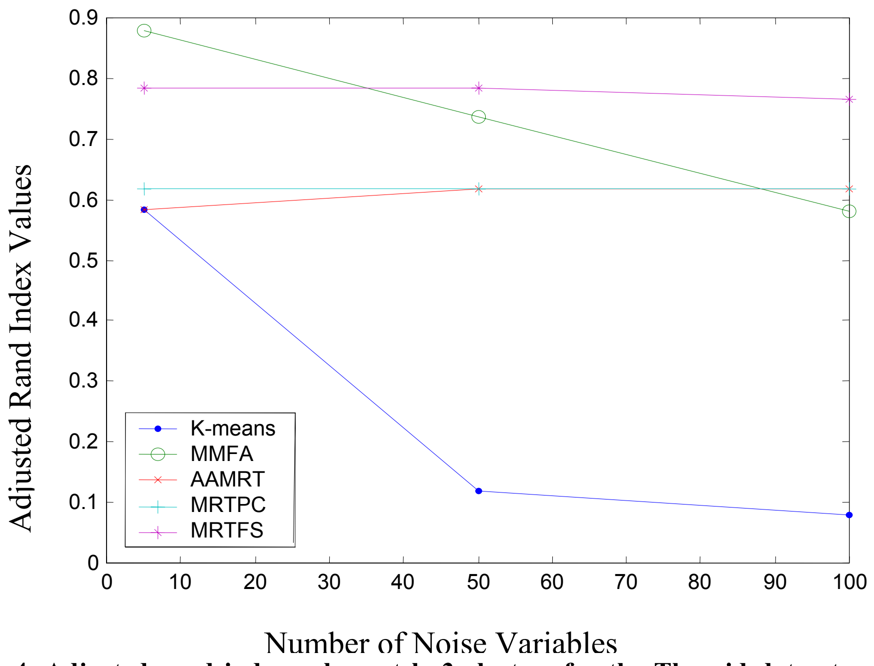


Fig. 4. Adjusted rand index values at k=3 clusters for the Thyroid dataset perturbed by type II noise variables.

Table 3. Adjusted rand index values at k=3 clusters using the Thyroid dataset

Number of Type II Noise Variables	*Number of Factors*	*MMFA*[d]	*MRTPCS*	*MRTFS*
5	1	.8635	.6167	.7848
	2	.8782	.6167	.5832
	3	.8781	.6167	.6118
	4	.8988	.6167	.4240
	5	.8819	.6167	.0817
	6	.8513	.6167	.0815
50	1	.7003	.6167	.7848
	2	.7353	.6167	.7663
	3	.5939	.6167	.4240
	4	.5619	.6167	.4240
	5	.4136	.6167	.4240
	6	.3320	.6167	.4240
100	1	.5804	.6167	.7663
	2	.5815	.6167	.7663
	3	.4212	.6167	.6573
	4	.3446	.6167	.5999
	5	.2275	.6167	.5999
	6	.2131	.6167	.5999

The BIC estimates are initially in agreement with the known number of clusters in the dataset (Table 4). However, as the amount of error perturbation increases, the BIC indicates that the mixture models of factor analysers only require one cluster to describe the data. The elbow estimates increase for the auto-associative multivariate regression tree as the amount of noise increases. However, the elbow estimates are fairly stable and usually reflect the known number of clusters for the trees grown using either principal component scores or factor scores. The silhouette statistic estimates were stable for all techniques, excluding K-means (Table 4).

[d] Average of top ten adjusted rand index values from 50 random starts

Table 4. Estimates of cluster number for the Thyroid dataset.

		Silhouette Statistic			*BIC/ELBOW*[e]		
Number of Type II Noise Variables		5	50	100	5	50	100
K-means		2	10	10	/	/	/
AAMRT		2	2	2	3	3	10
MMFA[f]	1	2	2	2	>4	2	1
	2	2	2	2	3	2	1
	3	2	2	2	3	2	1
	4	2	2	2	3	2	1
	5	2	2	2	3	1	1
	6	2	2	2	3	1	1
MRTPCS	1	2	2	2	3	3	3
	2	2	2	2	3	3	4
	3	2	2	2	3	3	3
	4	2	2	2	3	3	3
	5	2	2	2	3	3	3
	6	2	2	2	3	3	3
MRTFS	1	2	2	2	2	5	5
	2	2	2	2	3	10	2
	3	3	2	2	7	3	3
	4	2	2	2	7	3	3
	5	2	2	2	6	3	3
	6	2	2	2	3	3	3

5. Discussion

K-means was substantially affected by the error perturbation of these datasets, consistent with the results of Milligan [14]. The results of the Vietnam dataset were marginally better than the Thyroid dataset. This can be attributed to the natural clusters of the Vietnam dataset being spherical; a shape more easily found by K-means than the Thyroid dataset's elliptical clusters. We used a large number of iterations to counteract the random start employed by the K-means algorithm, however it is possible that the algorithm did not iterate to convergence.

[e] BIC estimates are for the mixture models of factor analysers. Elbow estimates are for the multivariate regression trees with principal component scores and factor scores
[f] Most frequently occurring estimate from 50 random starts

Mixture models of factor analysers were also affected by the large numbers of superfluous variables. Despite performing local dimensionality reduction, which offers several advantages over global dimensionality reduction [2], mixture models of factor analysers did not perform as successfully as the tree techniques using these datasets. Additionally, the results using these datasets show the sensitivity of the algorithm to the amount of dimension reduction chosen within a component. Previously this parameter was indicated to be fairly insensitive [3].

Normal mixture models can also be used to remove variables prior to fitting a mixture model of factor analysers. A normal mixture model is fit to a variable, and the variable is retained if the log likelihood ratio statistic for testing one versus two components is large enough. The test lacks the power of a multivariate selection criterion, because each variable is assessed individually. However, incorporating this variable selection step may improve the results of the mixture models of factor analysers. We did not incorporate variable selection, because the aim of the investigation was to ascertain the capabilities of the algorithms when explicitly clustering noisy data.

Using these two datasets, the results indicated that the tree techniques were the most stable amongst those studied. The auto-associative multivariate regression trees and multivariate regression trees with principal component scores performed almost identically. Because the principal component scores are essentially a re-representation of the dataset in a different basis system, it is not surprising that these techniques produced similar results.

The multivariate regression tree with factor scores was the superior technique amongst those studied. However, it has associated limitations. Firstly, we used maximum likelihood factor analysis, which assumes multivariate normality. If the dataset cannot be assumed to be multivariate normal, either the technique of factor analysis must change or the maximum likelihood results must be interpreted with caution. Secondly, these results show that the dimension of the factor space must be carefully chosen. A technique for assessing the optimal dimension of the factor space for clustering via a multivariate regression tree is not known. Maximum likelihood factor analysis provides a test for the correct number of factors. However, this test did not suggest the number of factors that provided the optimal adjusted rand index results for either dataset.

The tree techniques have additional advantageous characteristics. Firstly, trees are visually interpretable. Secondly, and possibly most importantly, the tree techniques have an automatic, variable importance measure [1]. This variable importance measure is multivariate. The splitting variables of the tree are the important variables, and their positions in the tree denote their relative performance. It is possible to use this variable importance measure as a variable selection step, and re-grow the tree using only the variables initially deemed important. Finally, the trees offer an intrinsic estimator of cluster size: the elbow of the relative error graph. However, we realise that this estimator is relatively ad-hoc, and requires fine-tuning to be considered as a viable, accurate estimator of cluster size.

Future work may involve determining a method to assess the optimal dimension of the factor (response) space for the multivariate regression trees with factor scores; and deriving a more complicated method to locate the elbow of the relative error curve.

However, overall the results show that the tree techniques provide stable, high-quality results when clustering noisy data.

6. Conclusion

We used an error perturbation study to demonstrate the capabilities of recent clustering algorithms. The results have shown that not all techniques are robust to superfluous noise. K-means and mixture models of factor analysers failed to find the natural clusters as the datasets were increasingly perturbed by noise.

We also introduced two integrated techniques of cluster analysis: multivariate regression trees with principal component and factor scores. The tree techniques offer an automatic, multivariate variable importance measure. All tree techniques produced stable, high-quality clusters after the datasets were perturbed by different types of extraneous variables. The multivariate regression tree with factor scores generated superior results amongst the tree techniques. These results indicate the potential of the tree techniques to cluster datasets with superfluous noise, for example high dimensional datasets.

7. Implementation

MATLAB 6.1 (The MathWorks, Inc.) code created by Ghahramani and Hinton [2] (available at ftp://ftp.cs.toronto.edu/pub/zoubin/mfa.tar.gz) was used to fit the mixture models of factor analysers. R (available from http://www.R-project.org.) was used to implement the K-means algorithm, auto-associative multivariate regression trees, multivariate regression trees with principal component and factor scores, and the silhouette statistic. MATLAB 6.1 (The MathWorks, Inc.) was used to calculate the adjusted rand index.

References

[1] F. Questier, R. Put, D. Coomans, B. Walczak, Y.V. Heyden, The use of CART and multivariate regression trees for supervised and unsupervised feature selection, Chemometrics and Intelligent Laboratory Systems. 76 (2005) 45-54.

[2] Z. Ghahramani, G. Hinton, The EM Algorithm for Mixtures of Factor Analyzers, University of Toronto, Canada 1996.

[3] G.J. McLachlan, R.W. Bean, D. Peel, A mixture model-based approach to the clustering of microarray expression data, Bioinformatics. 18 (2002) 413-422.

[4] G.W. Milligan, An examination of the effect of six types of error perturbation on fifteen clustering algorithms, Psychometrica. 45 (1980) 325-342.

[5] G.H. Dunteman, Principal Components Analysis, Sage, Newbury Park, CA, 1989.

[6] P. Kline, An Easy Guide to Factor Analysis, Routledge, New York, 1994.

[7] L. Breiman, J. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.

[8] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm, Journal of the Royal Statistical Society Series B-Statistical Methodology. 39 (1977) 1-38.

[9] L. Kaufman, P. Rousseeuw, Finding groups in data: an introduction to cluster analysis, Wiley, New York, 1990.

[10] L. Hubert, P. Arabie, Comparing Partitions, Journal of Classification. 2 (1985) 193-218.

[11] W.M. Rand, Objective Criteria for the Evaluation of Clustering Methods, Journal of the American Statistical Association. 66 (1971) 846-850.

[12] H.T. Nguyen, D. Coomans, M. Leermakers, J. Boman, Multivariate statistical analysis of human exposure to trace elements from coal in Vietnam, SPRUCE IV International Conference on Statistical Aspects of Health and the Environment, Enschede, the Netherlands 1997.

[13] D. Coomans, M. Jonckheer, D.L. Massart, I. Broeckaert, P. Block, The application of linear discriminant analysis in the diagnosis of thyroid diseases, Analytica Chimica Acta. 103 (1978) 409-415.

[14] G.W. Milligan, M.C. Cooper, A Study of the Comparability of External Criteria for Hierarchical Cluster Analysis, Multivariate Behavioral Research. 21 (1986) 441-458.

SYNOPSIS

This section initially showed that AAMRTs could successfully double as a clustering algorithm in the low dimensional setting. Furthermore, multivariate regression trees integrated with dimension reduction techniques (principal components analysis and factor analysis) produced accurate clusters with large, noisy datasets.

By growing many trees, many clustering solutions can be obtained and each solution can be assessed in terms of "prediction error". The predictive gold standard allows regression post processing techniques to calculate the weights of an ensemble of regression trees, implying that predictable clustering solutions are assigned high weights. The next section investigates post processing techniques for regression ensembles (combining regression trees and simple linear regression models) with the objective of finding a superior weighting strategy (applicable within the tree-based cluster ensemble framework).

POST PROCESSING REGRESSION ENSEMBLES

OVERVIEW

Ensembles combine many models to give an overall solution. Some models within the ensemble will naturally be better than others. An ensemble's accuracy can be increased, if the poor models are assigned low (or zero) weights by a post processing technique. The manuscripts of this section are dedicated to comparing regression post processing techniques with the objective of finding a superior weighting strategy. The first manuscript trials the lasso, Bayesian linear regression, and Bayesian linear regression with genetic algorithms. The second manuscript assesses the lasso, quadratic programming, quadratic programming with genetic algorithms and evolution strategies. Although each manuscript contains the necessary theory, some theory is elaborated in the "Supporting Theory" appendix.

Parsimonious Ensembles for Regression

Christine Smyth and Danny Coomans

Statistics and Intelligent Data Analysis Group,

School of Mathematical and Physical Sciences,

James Cook University,

Australia

Computing Science and Statistics

Volume 38

2006

1. Abstract

An ensemble of regression models predicts by taking a weighted average of the predictions made by individual models. Predictions based on ensembles have been shown to be very effective on large datasets. Calculating the weights such that they reflect the accuracy of individual models (post processing the ensemble) has been shown to increase an ensemble's accuracy. The success of previous research motivates the study of other strategies as potential post processing techniques. This paper introduces post processing techniques and demonstrates the improvements attained by using more parsimonious ensembles of linear regression models and regression trees.

2. Introduction

An ensemble learner combines the predictions from many regression models to give on average a more stable and accurate prediction for an observational unit. Mathematically, the ensemble is given by:

$$F(\underset{\sim}{x}) = \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}) \tag{1.1}$$

where $f_i(\underset{\sim}{x})$ is the prediction of an observational unit $\underset{\sim}{x}$ by the i^{th} model; ω_i is the weight assigned to $f_i(\underset{\sim}{x})$; and M is the number of models.

The $f_i(\underset{\sim}{x})$ are usually of the same family of models but this is not mandatory. They are individualized by using randomized operators within the model (Friedman and Popescu 2003). The randomness intrinsic to each $f_i(\underset{\sim}{x})$ implies that some models are better than others. However, this is not often reflected in the weights of an ensemble, which can be a simple average of the models, $1/M$. Post processing is a process that suggests choices of ω_i that reflect the relevance of each $f_i(\underset{\sim}{x})$ (Friedman and Popescu 2003).

The most obvious intelligent choice for $\underset{\sim}{\omega}$, the vector of all weights, is to choose $\underset{\sim}{\omega}$ to minimize a generic loss criterion:

$$\hat{\underset{\sim}{\omega}} = \min_{\underset{\sim}{\omega}} \sum_{j=1}^{n} L\left(y_j, \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) \right) \tag{1.2}$$

where n is the number of observational units. If $L\left(y_j, \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) \right)$ is the simple squared error loss given by:

$$L\left(y_j, \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) \right) = \sum_{j=1}^{n} \left(y_j - \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) \right)^2 \tag{1.3}$$

post processing the weights is identical to an ordinary least squares regression problem. The $\hat{\underset{\sim}{\omega}}$ are given by:

$$\hat{\underset{\sim}{\omega}} = (X^T X)^{-1} X^T \underset{\sim}{y} \tag{1.4}$$

where X is an $n \times M$ matrix where each column represents the predictions for n observational units for a single model; and $\underset{\sim}{y}$ are the observed data.

However, it is well known that ordinary least squares coefficients are prone to overfitting, and Friedman and Popescu (2003) suggested a regularized regression approach:

$$\hat{\underset{\sim}{\omega}} = \arg\min_{\omega} \sum_{j=1}^{n} \left(y_j - \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) \right)^2 + \lambda \sum_{i=1}^{M} \omega_i \tag{1.5}$$

The minimization criterion (1.5) is known as the lasso criterion. Because the parameter λ ranges from 0 (giving the least squares estimates of $\underset{\sim}{\omega}$ (1.4)) to ∞ (giving the weights as equivalently zero) the solution space to (1.5) is massive and its solution cumbersome. Friedman and Popescu (2004) suggest fast forward search algorithms for solving (1.5), for the squared error loss and other loss functions.

It can be shown that the coefficients given by the lasso criterion can be viewed as those given by Bayesian linear regression with independent double exponential priors on the regression coefficients (Hastie, Tibshirani and Friedman 2001). Therefore, implementing different priors on the coefficients will induce different minimization criteria, and thereby different sets of weights. The aim of this paper is to introduce "Bayesian-orientated" post processing techniques and compare them to Hastie et al.'s (2001) forward stagewise algorithm over several real datasets.

This paper illustrates four different priors: multivariate normal, multivariate t, Weibull and the double exponential. Genetic algorithms, intelligent random search techniques, are used in parallel with the Bayesian linear regression models to reduce the number of models within the ensemble. It is expected that this enforced parsimony will invoke more accurate ensembles.

3. Theory

3.1 Lasso Criterion

The solution to (1.5) is approximated with a forward stagewise algorithm (Hastie et al. 2001) which is henceforth referred to as the "lasso heuristic". The algorithm is as follows:

```
1. Set all weights to zero. Choose $\varepsilon$ as a small number greater than zero, and
   choose the number of iterations, $its$, to be quite large.
2. for $l = 1 : its$
   2.1. $(\beta^*, k^*) = \arg\min_{\beta,k} \sum_{j=1}^{n} \left( y_j - \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j) - \beta \times f_k(\underset{\sim}{x}_j) \right)^2$
   2.2. $\hat{\omega}_{k^*} = \hat{\omega}_{k^*} + \varepsilon \times sign(\beta^*)$
3. $F(\underset{\sim}{x}) = \sum_{i=1}^{M} \hat{\omega}_i f_i(\underset{\sim}{x})$
```

In step one all weights are zero, and this is analogous to $\lambda = \infty$ in (1.5). The parameter its is inversely related to λ in (1.5). After the set number of iterations, many weights will still remain zero.

3.2 Genetic Algorithm Theory

A genetic algorithm can be used to calculate which models are the most relevant in the ensemble. Each potential solution is encoded via a chromosome and the set of all chromosomes at any iteration is referred to as the population (Davis 1991). Each chromosome represents important models by a $'1'$ in the corresponding bit position and superfluous models with a $'0'$ in the corresponding bit position.

The initial generation is randomly generated by setting bits to zero with high probability and bits to one with low probability. This reflects the notion that it is likely most models contain no relevant information.

To calculate the fitness values of each chromosome, the set of models deemed to have a non-zero weight by the chromosome ($'1'$ in corresponding bit position) are parsed to a Bayesian linear regression model. The Bayesian linear regression model calculates the weights for these models only, and parses this information back to the fitness evaluation module of the genetic algorithm. The fitness of each chromosome is given by the inverse of the root mean square error:

$$RMSE_i = \sqrt{\frac{1}{n}\left(\underset{\sim}{y} - X_i \underset{\sim}{\hat{\omega}}_i\right)^2} \tag{1.6}$$

where $\underset{\sim}{\hat{\omega}}_i$ is the vector of weights calculated by the Bayesian linear regression model for chromosome i; X_i is the predictions matrix of the set of models chromosome i indicates to have non-zero coefficients.

The genetic algorithm used here implements roulette wheel selection with one point crossover (Davis 1991). The selection and reproduction process is repeated until enough offspring are produced to replace the entire generation.

The offspring are mutated by setting bits with value '1' to '0' if too many bits in any chromosome are '1'. This ensures that the expectation that only a few models are relevant is met.

The current generation is then replaced by the offspring generation and the process iterates until the chromosomes converge. The genetic algorithm only allows a small number of models to have non-zero weight coefficients. Also, the genetic algorithm only allows sets of all positive weights. These conditions are enforced within the mutation and fitness evaluation modules of the algorithm. The genetic algorithm will eventually converge to the set of models that give the lowest root mean square error.

The genetic algorithm should improve upon results obtained using only Bayesian linear regression on the basis of parsimony. The genetic algorithm is run five times for each proposed prior and the best and worst results reported. The algorithm is programmed in R (R Development Core Team 2004).

3.3 Bayesian Linear Regression

Bayesian theory incorporates exogenous knowledge ("prior distribution") to estimate the (posterior) distribution of population parameters $\utilde{\theta}$. Using Bayes' Rule, we obtain the posterior density $p(\utilde{\theta} \mid \utilde{y})$:

$$p(\utilde{\theta} \mid \utilde{y}) = \frac{p(\utilde{\theta}, \utilde{y})}{p(\utilde{y})} = \frac{p(\utilde{\theta})p(\utilde{y} \mid \utilde{\theta})}{p(\utilde{y})} \tag{1.7}$$

where $p(\utilde{y}) = \sum_{\utilde{\theta}} p(\utilde{\theta})p(\utilde{y} \mid \utilde{\theta})$ for discrete $\utilde{\theta}$; $p(\utilde{y}) = \int_{\utilde{\theta}} p(\utilde{\theta})p(\utilde{y} \mid \utilde{\theta})d\utilde{\theta}$ for continuous $\utilde{\theta}$; $p(\utilde{\theta})$ is the prior probability distribution of the parameters; $p(\utilde{y} \mid \utilde{\theta})$ is the sampling distribution (likelihood function) dependent on the parameters $\underset{\sim}{\theta}$; and $p(\underset{\sim}{\theta} \mid \underset{\sim}{y})$ is the posterior probability distribution of the parameters given the observed data.

Because $p(\underset{\sim}{y})$ is independent of $\underset{\sim}{\theta}$ and can therefore be considered constant for fixed $\underset{\sim}{y}$, we arrive at the unnormalized posterior density:

$$p(\underset{\sim}{\theta} \mid \underset{\sim}{y}) \propto p(\underset{\sim}{y} \mid \underset{\sim}{\theta}) p(\underset{\sim}{\theta}) \tag{1.8}$$

Bayesian Theory can be applied to ordinary linear regression by imposing prior distributions on the parameters of the regression model, the regression coefficients, $\underset{\sim}{\omega}$ and the variance of the error term, σ^2 (Gelman, Carlin, Stern and Rubin 1995). The likelihood function $l(\underset{\sim}{\theta} \mid \underset{\sim}{y})$ where $l(\underset{\sim}{\theta} \mid \underset{\sim}{y}) = p(\underset{\sim}{y} \mid \underset{\sim}{\theta})$ is given by:

$$l(\underset{\sim}{\omega}, \sigma^2 \mid \underset{\sim}{y}) = (2\pi)^{-n/2} (\sigma^2)^{-n/2} \exp\left(-(2\sigma^2)^{-1} (\underset{\sim}{y} - X\underset{\sim}{\omega})^T (\underset{\sim}{y} - X\underset{\sim}{\omega})\right) \tag{1.9}$$

Here, we use Bayesian linear regression in a manner such that the regression coefficients can be considered as the vector of weight coefficients for our ensemble.

3.3.1 Double Exponential Prior

The prior for each regression coefficient is assumed to be an independent double exponential distribution with hyperparameters τ_i, μ_i:

$$p(\omega_i) \sim dexp(\tau_i, \mu_i) \tag{1.10}$$

Each hyperparameter μ_i is specified a priori as zero. The prior probability distribution of a single regression coefficient is then peaked at zero. The hyperparameter τ_i reflects the certainty that a weight ω_i is equal to zero. Here, all τ_i are identical.

The prior distribution of the error variance, σ^2, is taken to be inverse gamma for conjugacy reasons discussed later. The inverse gamma distribution for the error variance is used in conjunction with all the different priors on the regression coefficients. The parameter, σ^2, is not of interest, and therefore the inverse-gamma prior is used throughout:

$$p(\sigma^2) \sim Inv - gamma(\alpha, \beta) \tag{1.11}$$

The joint prior probability density function is given by:

$$p(\underset{\sim}{\omega}, \sigma^2) \propto (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta \big/ \sigma^2 - \tau \sum_{i=1}^{M} | \omega_i | \right) \tag{1.12}$$

The joint unnormalized posterior distribution is given by:

$$p(\underset{\sim}{\omega}, \sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2 - \tau \sum_{i=1}^{M} | \omega_i | \right) \exp\left(-(2\sigma^2)^{-1} (\underset{\sim}{y} - X\underset{\sim}{\omega})^T (\underset{\sim}{y} - X\underset{\sim}{\omega}) \right) \tag{1.13}$$

The conditional distribution of the vector of regression coefficients given σ^2 is:

$$p(\underset{\sim}{\omega} \mid \sigma^2, \underset{\sim}{y}) \propto \exp\left(-(2\sigma^2)^{-1} (\underset{\sim}{y} - X\underset{\sim}{\omega})^T (\underset{\sim}{y} - X\underset{\sim}{\omega}) - \tau \sum_{i=1}^{M} | \omega_i | \right) \tag{1.14}$$

By consideration of (1.14) it can easily be seen that using independent double exponential prior distributions for each of the regression coefficients is analogous to the regularized regression (1.5) where $\lambda = 2\sigma^2\tau$ (Hastie et al. 2001).

3.3.2 Weibull Prior

The prior of each regression coefficient is assumed to be an independent Weibull distribution:

$$p(\omega_i) \sim Weib(\alpha_i, \beta_i) \tag{1.15}$$

The Weibull prior ensures all coefficients are greater than zero. Here, for computational convenience, α_i is always two. The hyperparameter β_i is varied according to the prior belief that the regression coefficient is equal to zero. Here, all β_i are identical.

The joint prior is then given by:

$$p(\underset{\sim}{\omega},\sigma^2) \propto (\sigma^2)^{-(\alpha+1)} \exp\left[-\beta/\sigma^2 - \sum_{i=1}^{M}\beta_i\omega_i^2\right]\prod_{i=1}^{M}\omega_i \quad (1.16)$$

The joint posterior is given by:

$$\begin{aligned} p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right)\prod_{i=1}^{M}\omega_i \times \\ \exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}-X\underset{\sim}{\omega})^T(\underset{\sim}{y}-X\underset{\sim}{\omega}) - \sum_{i=1}^{M}\beta_i\omega_i^2\right) \end{aligned} \quad (1.17)$$

Conditioning on σ^2 we can easily see that this is another regularized regression where the hyperparameters β_i have a direct influence on the set of weights selected:

$$p(\underset{\sim}{\omega} \mid \sigma^2, \underset{\sim}{y}) \propto \exp\left(-(2\sigma^2)^{-1}\left(\underset{\sim}{y}-X\underset{\sim}{\omega}\right)^T\left(\underset{\sim}{y}-X\underset{\sim}{\omega}\right) - \sum_{i=1}^{M}\left(\beta_i\omega_i^2 - \ln\omega_i\right)\right) \quad (1.18)$$

3.3.3 Multivariate Normal Prior

The prior of the coefficients is taken to be multivariate normal:

$$p(\underset{\sim}{\omega}) \sim \mathrm{N}_M(\underset{\sim}{\mu}_0, \Sigma_0) \quad (1.19)$$

The hyperparameters, $\underset{\sim}{\mu}_0$ and Σ_0, are specified a priori. The vector, $\underset{\sim}{\mu}_0$ is specified as a vector of zeros, reflecting that prior to analysis all regression coefficients (models) are expected to be non-informative. The matrix Σ_0, varies according to the certainty that the weights are all equal to zero. The matrix is a diagonal matrix where all elements are identical. The combination of the multivariate normal prior on the $\underset{\sim}{\omega}$ and the inverse gamma for the error variance is considered the conjugate prior for regression analysis. The joint prior is:

$$p(\underset{\sim}{\omega},\sigma^2) \propto |\Sigma_0|^{-1/2} \exp\left(-1/2(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)^T\Sigma_0^{-1}(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)\right)\beta^\alpha/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)}\exp\left(-\beta/\sigma^2\right) \quad (1.20)$$

The posterior of the parameters is:

$$p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right)\exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}^T \underset{\sim}{y} - 2\underset{\sim}{\omega}^T X^T \underset{\sim}{y} + \underset{\sim}{\omega}^T X^T X \underset{\sim}{\omega})\right)\exp\left(-1/2\,(\underset{\sim}{\omega}^T \Sigma_0^{-1} \underset{\sim}{\omega}_0)\right)$$
$$p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right)\exp\left(-1/2\left(\underset{\sim}{\omega}^T \left((\sigma^2)^{-1} X^T X + \Sigma_0^{-1}\right)\underset{\sim}{\omega} - 2\underset{\sim}{\omega}^T (\sigma^2)^{-1} X^T \underset{\sim}{y}\right)\right) \tag{1.21}$$

Conditioning (1.21) on σ^2, shows the conditional distribution of the weights is normal:

$$p(\underset{\sim}{\omega} \mid \sigma^2, \underset{\sim}{y}) \sim \mathrm{N}_M\left(\left((\sigma^2)^{-1} X^T X + \Sigma_0^{-1}\right)^{-1} (\sigma^2)^{-1} X^T \underset{\sim}{y}, \left((\sigma^2)^{-1} X^T X + \Sigma_0^{-1}\right)^{-1}\right) \tag{1.22}$$

The multivariate normal prior with $\underset{\sim}{\mu}_0 = \underset{\sim}{0}$ will induce estimates of the regression coefficients similar to the ordinary least squares estimates for large prior variance. However, as we decrease the prior variance, greater emphasis is placed on prior beliefs (that is $\underset{\sim}{\omega} = \underset{\sim}{0}$) than the data itself.

3.3.4 Multivariate T Prior

The final prior on the weights is taken to be multivariate t:

$$p(\underset{\sim}{\omega}) \sim t_\nu(\underset{\sim}{\mu}_0, \Sigma_0) \tag{1.23}$$

The hyperparameters $\underset{\sim}{\mu}_0, \Sigma_0$ are specified to reflect prior belief regarding the regression coefficients in a manner identical to that of the multivariate normal prior. The multivariate t prior with these hyperparameters anticipates that most of the weights are zero. However, the longer tails of a t distribution prior allow larger weights than the multivariate normal prior.

The joint prior is given by:

$$p(\underset{\sim}{\omega},\sigma^2) = \frac{\Gamma((\nu+M)/2)}{\Gamma(\nu/2)\nu^{M/2}\pi^{M/2}} |\Sigma_0|^{-1/2} \left(1+\frac{1}{\nu}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)^T \Sigma_0^{-1}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)\right)^{-(\nu+M)/2} \beta^\alpha/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/(\sigma^2)\right) \tag{1.24}$$

The joint posterior is given by:

$$p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/(\sigma^2)\right)\exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y} - X\underset{\sim}{\omega})^T(\underset{\sim}{y} - X\underset{\sim}{\omega})\right)\times$$
$$\left(1+\frac{1}{\nu}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)^T \Sigma_0^{-1}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)\right)^{-(\nu+M)/2} \tag{1.25}$$

3.3.5 Sampling from the posterior distribution

To obtain an estimate for the parameter of interest, a draw is taken from the posterior distribution. Often, when the parameter vector $\underset{\sim}{\theta}$ is very large, it is necessary to approximate or simulate the posterior distribution. Here, the posterior distributions are simulated using a Gibb's sampler in Winbugs 1.4 (Spiegelhalter, Thomas, Best and Lunn 2003). The parameter vector $\underset{\sim}{\theta}$, is subdivided into $\underset{\sim}{\theta} = (\underset{\sim}{\theta}_1, \underset{\sim}{\theta}_2, ..., \underset{\sim}{\theta}_k)$ such that the conditional distributions $p(\underset{\sim}{\theta}_i \mid \underset{\sim}{\theta}_1, ..., \underset{\sim}{\theta}_{i-1}, \underset{\sim}{\theta}_{i+1}, ..., \underset{\sim}{\theta}_k, \underset{\sim}{y})$ are easily recognized or approximated for all $\underset{\sim}{\theta}_i$. Initial values are assigned to each $\underset{\sim}{\theta}_i, i = 1, ..., k$, to give $\underset{\sim}{\theta}^{(0)} = (\underset{\sim}{\theta}_1^{(0)}, ..., \underset{\sim}{\theta}_k^{(0)})$. The algorithm then iterates through each $\underset{\sim}{\theta}_i$ updating it with a draw from its conditional distribution given the current estimates of $\underset{\sim}{\theta}_j$, for all $j \neq i$. That is, $\underset{\sim}{\theta}_i^{(t)}$ is a draw from $p(\underset{\sim}{\theta}_i \mid \underset{\sim}{\theta}_1^{(t)}, \underset{\sim}{\theta}_2^{(t)}, ..., \underset{\sim}{\theta}_{i-1}^{(t)}, \underset{\sim}{\theta}_{i+1}^{(t-1)}, ..., \underset{\sim}{\theta}_k^{(t-1)})$. Each full iteration gives values for all parameters, $\underset{\sim}{\theta}^{(t)}$, where $\underset{\sim}{\theta}^{(t)} = (\underset{\sim}{\theta}_1^{(t)}, ..., \underset{\sim}{\theta}_k^{(t)})$. After a large number of iterations called a "burn-in", $\underset{\sim}{\theta}^{(t)}$ converges to a draw from the joint posterior distribution.

4. Datasets and Procedure

4.1 Procedure

Each dataset was partitioned into three subsets (Table I) to allow for an estimate of the true accuracy of the ensemble via a test set. The first training set was used to grow the models. The weights for the ensembles were derived using the second training subset. The third subset, the "test set", was used to obtain the R^2 of the ensembles.

M linear regression models (each using a few randomly selected variables) and M regression trees (where the M trees were the individual trees of a single random forest (Breiman 2001)) were obtained for each dataset using R (R Development Core Team 2004). The linear regression models and regression tree models were combined using various weighting strategies. The weighting strategies were: the lasso heuristic (referred to in the results as **LASSO**) and Bayesian linear regression with four different priors: independent double exponential (**DE**), independent Weibull (**W**), multivariate normal (**N**), and multivariate t (**T**).

Furthermore, for each prior type (**DE**, **W**, **N**, **T**) the Bayesian linear regression model was run in four different ways:

(**1**) Allowing all models to have non-zero weights. The numeral '**1**' follows the prior type in the results. For example, **DE1** refers to weights obtained using Bayesian linear regression with independent double exponential priors.

(**2&3**) Reducing the number of models using two different iterative schemes.

The first iterative scheme progressively shaved off the models with the smallest weights (setting these weights to zero) and recalculated the weights of the remaining models. This shaving process was repeated three times until only five percent of the models had non-zero weights. Weights obtained using this iterative scheme have a '**2**' following the respective prior type in the results section. For example, **W2** refers to weights obtained using Bayesian linear regression with independent Weibull priors and the first iterative scheme.

The second iterative scheme selected the models with the largest weights over five simultaneous runs of the Bayesian linear regression model, and re-estimated the weights using only these models (all other weights were set to zero). Weights obtained using this iterative scheme have a '**3**' following the respective prior type in the results section. For example, **N3** refers to weights obtained using Bayesian linear regression with a multivariate normal prior and the second iterative scheme.

In both iterative schemes as the number of models with non-zero weights decreased, the hyperparameters changed to allow the non-zero weights to vary more significantly from zero.

(4) Reducing the number of models by using a genetic algorithm. Weights obtained using a Bayesian linear regression model with a genetic algorithm have a '**4**' following the appropriate prior type in the results. For example, **T4** refers to weights obtained using a genetic algorithm plus Bayesian linear regression with a multivariate t prior.

As a benchmarking tool, the linear regression and regression tree models were also combined using identical weights, $1/2M$. The R^2 of these simple, non-informative ensembles using the test subset of each dataset is given in Table II under the "**SA**" column. The R^2 of both the single best linear regression model (**REG**) and single best tree (**TREE**), chosen on the basis of the second training set and applied to the test set, are also reported.

4.2 Datasets

The weighting schemes were tested using five datasets. The datasets are summarized in Table I and the reader is directed to the references for more detailed explanations. The partition of each dataset into independent subsets and the choice of M were chosen to best mimic previous analyses. The table shows the number of terminal nodes of a tree within the forest, TN: chosen to either mimic previous analyses or diversify the size of the trees across the datasets.

Table I. Description of the Datasets.

Dataset Name and Response	Dimension ($N*p$[1])	Train/Train/Test	M	TN
Boston Housing (available from http://www.ics.uci.edu/~mlearn/MLRepository.html). *Median house values.*	506*13	350/100/56	200	4
Fat (available from http://stat.cmu.edu/datasets/). *Fat content of food measured by Tecator Infratec Food and Feed Analyzer.*	215*22	129/43/43	100	19
Body Fat (available from http://stat.cmu.edu/datasets/). *Percentage of body fat of males.*	252*13	143/59/50	100	21
Prostate (available from http://www-stat.stanford.edu/ElemStatLearn). *Prostate Specific Antigen levels of prostate cancer patients.*	97*8	67/67[2]/30	100	10
Friedman (Friedman 1991). *Generated.*	500*10	200/200/100	100	29

5. Results and Discussion

The results using each weighting strategy on each dataset are displayed in Table II. The results are separated into two rows for each dataset. The first row shows the R^2 obtained using each weighting strategy. Any negative R^2 values were set to zero, for ease of interpretation. The second row shows the number of models with non-zero weights found by each weighting technique.

The table shows that the single best linear regression model or tree (or both) is better than the **SA** for all five datasets. This reinforces the notion that many models are redundant and their inclusion in an ensemble decreases its accuracy. This is obvious when considering the body fat dataset. The body fat dataset is a simple dataset and can

[1] Where p is the number of predictor variables
[2] Training set one=training set two

be predicted remarkably well using a single linear regression model. We notice that by combining too many predictors the accuracy of the model is decreased.

The overwhelming trend of the table is that the single best models and simple average ensembles can be improved significantly by intelligently combining models. This is particularly evident with the Friedman, fat and Boston housing datasets. The improvements are most noticeable using the lasso heuristic and genetic algorithm approaches.

As a general rule, using the lasso heuristic tends to improve significantly upon the results of the single best models and the simple average ensembles. Instead of searching the entire solution space given by (1.5), using the lasso heuristic produces results that are very good in a very short period of time. Also, the double exponential prior results validate the lasso criterion as an excellent form of penalized regression. There is some degree of overlap between the models selected by the lasso heuristic and the double exponential prior. The models are not identical, but this is expected.

The genetic algorithms applied with the different priors produce results that are equally as good as those of the lasso heuristic. The genetic algorithm approaches create more parsimonious ensembles than the lasso heuristic. These results are reflected in Figure I and Figure II. Figure I shows the R^2 of the **REG**, **TREE**, and **SA** models well below the R^2 of the standout techniques: **LASSO**, **DE4**, **W4**, **N4**, and **T4**. Figure II shows that the genetic algorithm approaches combine far fewer models than the lasso heuristic. However, the genetic algorithm approaches are more computationally intensive than the lasso heuristic.

The iterative schemes **DE2**, **DE3,** **W2**, **W3**, **N2**, **N3**, **T2**, and **T3** tend to produce mediocre models. These strategies could effectively be viewed as a form of stepwise model elimination, and it is possible that the important models are removed too early in the elimination process. The genetic algorithm approaches avoid this eventuality because they allow models to reenter the ensemble via crossover and mutation. Analysis of the models selected by the genetic algorithm and the iterative techniques (data not shown) indicates that this may have been the case: there is little overlap between the sets of models.

The **W1** strategy does not perform as well as the **DE1**, **N1**, and **T1** strategies. Because the Weibull weights can never be zero, it is possible that the inclusion of many models with very small positive weights obscure the better models. The **DE1**, **N1**, and **T1** strategies allow negative weights and may outperform the **W1** strategy for this reason. The small negative weights may "cancel" with small positive weights, allowing the better models to dominate.

Figure I. R^2 values of the different weighting strategies applied to each dataset. Figures generated using MATLAB 7.0.4 (The MathWorks, Inc. 2005).

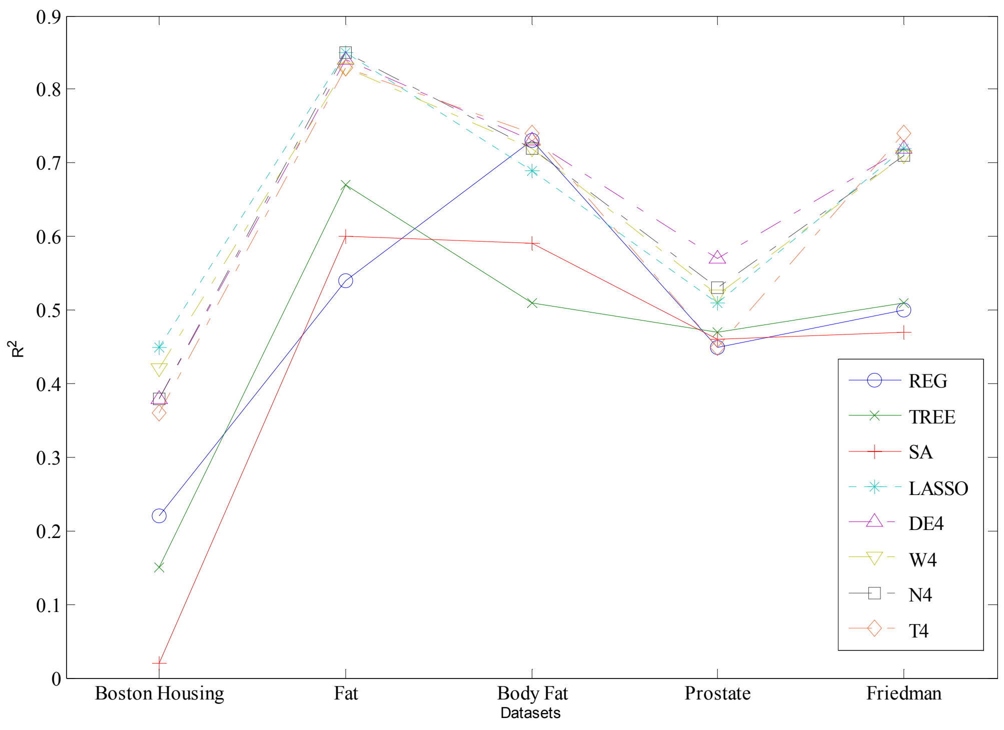


Figure II. Number of models required by each weighting strategy.

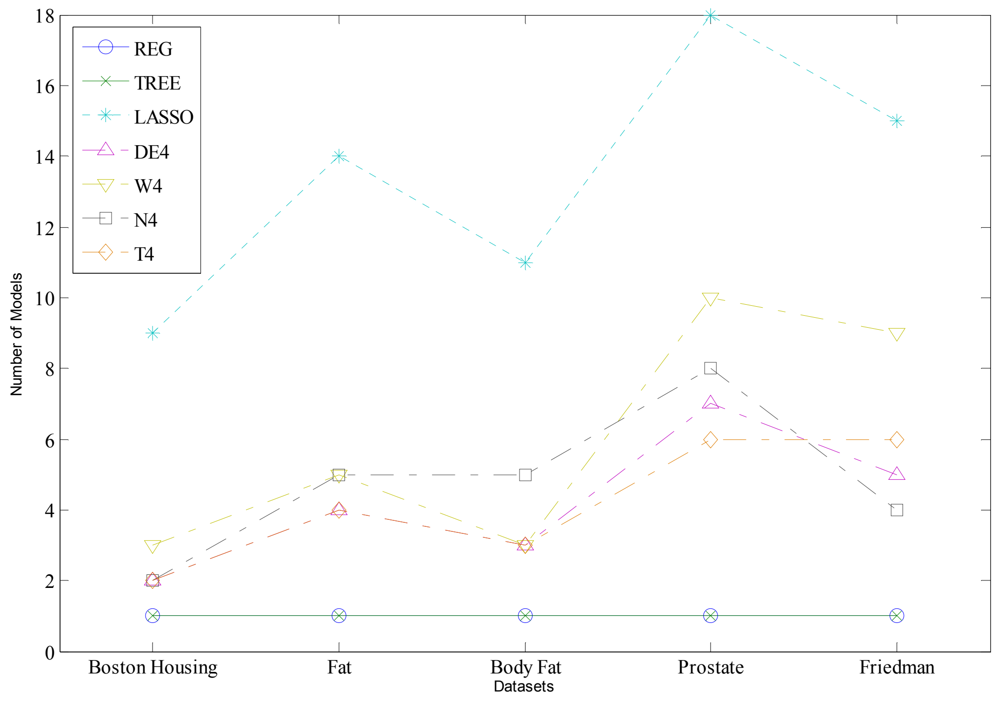


Table II. Results of the various weighting schemes over the five datasets. Both the best and worst genetic algorithm results are presented.

Dataset		**REG**	**TREE**	**SA**	**LASSO**	**DE1**	**DE2**	**DE3**	**DE4**	**DE4**	**W1**	**W2**	**W3**	**W4**	**W4**	**N1**	**N2**	**N3**	**N4**	**N4**	**T1**	**T2**	**T3**	**T4**	**T4**
Housing	R^2	.22	.15	0.02	.45	.21	.17	.16	.38	0	.04	.09	.22	.42	.16	0	0	0	.38	.15	0	0	0	.36	.14
	Models	1	1	2M	9	2M	20	34	2	2	2M	20	62	3	6	2M	20	25	2	1	2M	20	24	2	2
Fat	R^2	.54	.67	.60	.85	.80	.82	.83	.84	.62	.60	.76	.54	.83	.81	.89	.78	.62	.85	.67	.90	.80	.69	.83	.76
	Models	1	1	2M	14	2M	10	17	4	4	2M	10	46	5	9	2M	10	15	5	4	2M	10	13	4	4
Body Fat	R^2	.73	.51	.59	.69	.65	.68	.69	.73	.62	.59	.69	.58	.72	.65	.68	.65	.67	.72	.67	.72	.66	.67	.74	.67
	Models	1	1	2M	11	2M	10	28	3	6	2M	10	46	3	8	2M	10	15	5	3	2M	10	13	3	3
Prostate	R^2	.45	.47	.46	.51	.51	.56	.49	.57	.40	.46	.39	.46	.52	.28	.45	.49	.50	.53	.07	.46	.48	.57	.45	.32
	Models	1	1	2M	18	2M	10	28	7	6	2M	10	47	10	7	2M	10	16	8	5	2M	10	16	6	4
Friedman	R^2	.50	.51	.47	.72	.81	.73	.72	.72	.63	.50	.73	.72	.71	.64	.83	.71	.71	.71	.57	.85	.71	.70	.74	.60
	Models	1	1	2M	15	2M	10	25	5	3	2M	10	30	9	5	2M	10	13	4	3	2M	10	12	6	3

6. Conclusion

This paper highlighted that parsimonious ensembles perform more accurately than ensembles which include all available models. Additionally, a parsimonious ensemble is more easily interpreted than a large ensemble. The genetic algorithm combined with Bayesian linear regression approaches and the lasso heuristic outperformed other tested weighting strategies. The genetic algorithm approaches are computationally intensive but produce more parsimonious models than the lasso heuristic.

7. References

Breiman, L. (2001). "Random Forests." Machine Learning **45**(1): 5-32.

Davis, L. (1991). Handbook of genetic algorithms. New York, Van Nostrand Reinhold.

Friedman, J. (1991). "Multivariate Adaptive Regression Splines." The Annals of Statistics **19**(1): 1-67.

Friedman, J. and Popescu, B. (2003). Importance Sampled Learning Ensembles, Stanford University, Department of Statistics.

Friedman, J. and Popescu, B. (2004). Gradient directed regularization for linear regression and classification, Stanford University, Department of Statistics.

Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (1995). Bayesian Data Analysis. London, Chapman & Hall.

Hastie, T., Tibshirani, R. and Friedman, J. (2001). The Elements of Statistical Learning. New York, Springer-Verlag.

MATLAB 7.0.4.(2005). The MathWorks, Inc.

R A language and environment for statistical computing.(2004). R Foundation for Statistical Computing.

Spiegelhalter, D., Thomas, A., Best, N. and Lunn, D. WinBUGS User Manual Version 1.4.(2003).

Post processing regression ensembles: imposing parsimony to improve predictions

Christine Smyth[*] and Danny Coomans

Statistics and Intelligent Data Analysis Group,

School of Mathematics, Physics and Information Technology,

James Cook University,

Australia

Submitted

Journal of Applied Statistics

ABSTRACT

The inclusion of inaccurate models within a regression ensemble undoubtedly detracts from its performance. Post processing a regression ensemble involves finding a parsimonious subset of “good” models that give rise to a higher level of accuracy. This paper reviews well known post processing techniques: the lasso, quadratic programming and evolution strategies. Genetic algorithms are also employed in conjunction with the quadratic programs in an aggressive bid to further reduce the size of the ensemble and ensure solutions to the quadratic program are global. Four benchmark datasets are analysed and the results are reported using independent test sets. The results indicate that finding a parsimonious ensemble using post processing techniques usually increases the accuracy of a large simple average ensemble.

KEYWORDS: regression ensembles, quadratic programs, lasso, evolutionary algorithms, post processing

1 INTRODUCTION

An ensemble of regression models is mathematically given by:

$$F(\underset{\sim}{x}) = \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}) \tag{1.1}$$

where $f_i(\underset{\sim}{x})$ is the prediction of an observational unit $\underset{\sim}{x}$ (vector) by the i^{th} model - the $f_i(\underset{\sim}{x})$ are usually of the same family of models but this is not mandatory; ω_i is the weight assigned to $f_i(\underset{\sim}{x})$; and M is the number of models.

The merits of ensembles of regression models are particularly evident when considering large datasets. Here, a single regression model may fail to capture all information inherent in the dataset or be particularly unstable, and as such regression ensembles hold a distinct appeal. Each predictive model within an ensemble is individualized by using randomized operators (Friedman and Popescu, 2003). The use of these randomized operators implies that some models within the ensemble are better than others and some may have no predictive capability at all. Post processing is a procedure which suggests choices of ω_i that reflect the relevance of each $f_i(\underset{\sim}{x})$ (Friedman and Popescu, 2003). Generally, post processing removes redundant models from a large ensemble via penalization or creates a small ensemble via stagewise addition of models. In short, post processing achieves a higher predictive accuracy by enforcing parsimony.

Penalization post processing methods create parsimonious ensembles by seeking weights that minimize some generic loss function, $L\left(y_j, \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j)\right)$ plus a penalty function, $\lambda \cdot P\left(\underset{\sim}{\omega}\right)$:

$$\hat{\underset{\sim}{\omega}} = \arg\min_{\underset{\sim}{\omega}} \sum_{j=1}^{n} L\left(y_j, \sum_{i=1}^{M} \omega_i f_i(\underset{\sim}{x}_j)\right) + \lambda \cdot P(\underset{\sim}{\omega}). \tag{1.2}$$

where y_j is the response of the j^{th} observational unit and $\underset{\sim}{x}_j$ is the vector of explanatory variables for the j^{th} observational unit.

Obviously, different weights will be produced by imposing different penalty functions. Two well known penalty functions are 'ridge regression' (Hoerl and Kennard, 1970):

$$P\left(\underset{\sim}{\omega}\right) = \sum_{i=1}^{M} \omega_i^2 \tag{1.3}$$

and the 'lasso' (Tibshirani, 1996):

$$P(\underset{\sim}{\omega}) = \sum_{i=1}^{M} \left|\omega_i\right|. \tag{1.4}$$

Ridge regression tends to produce similar estimates for all the non-zero weights whereas the lasso tends to produce more diverse estimates between non-zero weights. Obviously, the choice of penalty can be made on the basis of prior belief regarding the diversity of the weights. The parameter λ in (1.2) 'tunes' the estimates. The parameter ranges from zero, giving the least stable estimates of the weights, to ∞, giving the most stable estimates of the weights by setting them all as equivalently zero. As a result the solution space to (1.2) is massive. The lasso solution is approximated here by a forward stagewise algorithm detailed in section 2.

The lasso (and indeed equation (1.2)) can be formulated as a quadratic program:

$$\underset{\sim}{\omega}^{lasso} = \arg\min_{\underset{\sim}{\omega}} \left(\underset{\sim}{y} - X\underset{\sim}{\omega}\right)^T \left(\underset{\sim}{y} - X\underset{\sim}{\omega}\right) \tag{1.5}$$

subject to

$$\sum_{i=1}^{M} \left|\omega_i\right| \leq s \tag{1.6}$$

where X is an $n*M$ matrix with each column containing the predictions of the n observational units for a single model; and the loss function is the squared error loss.

It is then easy to see that changing the objective function and constraints of the above quadratic program will suggest different weights for the ensemble. Quadratic programming (and also linear programming) has previously been used to calculate "optimal" weights of ensembles with commendable results (Krogh and Vedelsby, 1995; Heskes, 1997; Heskes, 1998). Section 2 describes further how post processing an ensemble can easily be formulated as a quadratic program, and the different objective function and constraints that are used in this study.

Furthermore, we employ genetic algorithms (section 2) in conjunction with the quadratic programs. Previously, it has been shown that integrating genetic algorithms with penalized Bayesian post processing techniques (such as the Bayes estimate of the lasso) has improved predictions (Smyth and Coomans, 2006). Here, the genetic algorithms induce greater parsimony and thereby ensure the solutions to the quadratic programs are global.

To contrast with the forward stagewise algorithm and quadratic programs which create a path in the solution space close to the true solution in an intelligent fashion, we also use a purely random technique, evolution strategies, to create a random walk in the solution space. Evolution strategies have previously been used as a post processing technique and were shown to outperform other post processing techniques (Zhou, Wu, Jiang and Chen, 2001). Evolution strategies are detailed in section 2.

2 THEORY

2.1 Lasso Heuristic

The solution to (1.2) and (1.4) is approximated by a stagewise algorithm (Hastie, Tibshirani and Friedman, 2001; Friedman and Popescu, 2004). Because of its similarity to the lasso, we refer to this stagewise algorithm as the "lasso heuristic" and in the results as "LASSO". The algorithm is as follows (Hastie, Tibshirani and Friedman, 2001):

1. Set all weights to zero. Choose ε as a small number greater than zero, and choose the number of iterations, its.

```
2. for $l = 1 : its$
    2.1. $\left(\beta^*, k^*\right) = \arg\min_{\beta,k} \sum_{j=1}^{n} \left( y_j - \sum_{i=1}^{M} \hat{\omega}_i f_i(\underset{\sim}{x}_j) - \beta \times f_k(\underset{\sim}{x}_j) \right)^2$
    2.2. $\hat{\omega}_{k^*} = \hat{\omega}_{k^*} + \varepsilon \times sign(\beta^*)$
3. $F(\underset{\sim}{x}) = \sum_{i=1}^{M} \hat{\omega}_i f_i(\underset{\sim}{x})$
```

In step one all weights are zero, and this is analogous to $\lambda = \infty$ in (1.2). The parameter its is inversely proportional to λ in (1.2). Only one weight is updated on any iteration and after the set number of iterations many weights will remain zero.

2.2 Quadratic Programs

A quadratic program refers to a constrained optimization problem, where the function to be maximized/minimized is quadratic and the constraints are linear. The function to be optimized is referred to as the objective function, $q(\underset{\sim}{\omega})$, and the solution as $\underset{\sim}{\omega}^*$. The set of constraints can contain both equality and inequality constraints. A quadratic program can be written as

$$\min_{\underset{\sim}{\omega}} \; q(\underset{\sim}{\omega}) = \frac{1}{2} \underset{\sim}{\omega}^T G \underset{\sim}{\omega} + \underset{\sim}{\omega}^T \underset{\sim}{d} \tag{1.7}$$

subject to

$$\begin{aligned} \underset{\sim}{a}_i^T \underset{\sim}{\omega} &= b_i \quad i \in \mathrm{E} \\ \underset{\sim}{a}_i^T \underset{\sim}{\omega} &\geq b_i \quad i \in \mathrm{I} \end{aligned} \tag{1.8}$$

where E and I are finite sets of indices that reference the equality and inequality constraints respectively; G is a $M \times M$ symmetric Hessian matrix; $\underset{\sim}{d}, \underset{\sim}{\omega}$ and $\{\underset{\sim}{a}_i\}$, $i \in \mathrm{E} \bigcup \mathrm{I}$ are $M \times 1$ dimensional vectors.

The quadratic program may be infeasible (no feasible point) or unbounded ($q(\underset{\sim}{\omega}) \rightarrow -\infty$ for $\underset{\sim}{\omega}$ in the feasible region) and has no solution. However these two situations are easily exposed. If the quadratic program can be solved, the uniqueness of the solution is dependent on the nature of the Hessian matrix, G . If G is indefinite, the solution is a local solution and the quadratic program is a "non-convex problem". If G is positive semi-definite, the solution is a global solution, and if G is positive definite the solution is global and unique. In these instances, the quadratic program is called a convex quadratic program.

As mentioned previously, the lasso can be formulated as a quadratic program (1.5) and (1.6). Obviously changing the objective function and constraints will suggest different sets of weights for the ensemble. Specifically, here we wish to minimize the loss of using the weighted sum of models to predict the response. If the loss is the squared error loss function, the problem becomes

$$\min_{\underset{\sim}{\omega}} \quad \text{q}(\underset{\sim}{\omega})=(\underset{\sim}{y} - X\underset{\sim}{\omega})^T(\underset{\sim}{y} - X\underset{\sim}{\omega}) \tag{1.9}$$

where X is the $n*M$ predictions matrix, subject to

$$\begin{aligned} &\omega_i \geq 0 \text{ for all } i \\ &\sum_{i=1}^{M} \omega_i = 1. \end{aligned} \tag{1.10}$$

This is easily converted to a quadratic program

$$\min_{\underset{\sim}{\omega}} \quad q(\underset{\sim}{\omega}) = \underset{\sim}{\omega}^T G \underset{\sim}{\omega} + \underset{\sim}{\omega}^T \underset{\sim}{d} \tag{1.11}$$

subject to (1.10); where $G = X^T X$ and $\underset{\sim}{d} = -2X^T \underset{\sim}{y}$.

The solution to this quadratic program is referred to as QP1. A number of other quadratic programs were formulated to calculate weights of an ensemble and are detailed below:

Quadratic Program 2 (QP2)

The second quadratic program is given by:

$$\min_{\underset{\sim}{\omega},\varepsilon} \quad q(\underset{\sim}{\omega},\varepsilon) = -\underset{\sim}{\omega}^T \underset{\sim}{\omega} + \varepsilon \tag{1.12}$$

subject to

$$\begin{aligned} X\underset{\sim}{\omega} &\geq \underset{\sim}{y} - \underset{\sim}{\varepsilon} \\ X\underset{\sim}{\omega} &\leq \underset{\sim}{y} + \underset{\sim}{\varepsilon} \\ \sum_{i=1}^{M} \omega_i &= 1 \end{aligned} \tag{1.13}$$

where $\underset{\sim}{\varepsilon}$ is a vector of $1's$ multiplied by ε.

The objective function of the quadratic program selects simultaneously

1. weights that are not a simple average of the models (that is, weights giving a dominant model)
2. the smallest tolerance rectangle around the observed predictions that the weighted predictions must fall within.

The reader will notice some similarities with support vector regression, see for example (Hastie, Tibshirani and Friedman, 2001) .

Quadratic Program 3 (QP3)

The third quadratic program is given by:

$$\min_{\underset{\sim}{\omega},\varepsilon} \quad q(\underset{\sim}{\omega},\varepsilon) = -\underset{\sim}{\omega}^T \underset{\sim}{\omega} + \omega_1\omega_2 + \omega_1\omega_3 + ... + \varepsilon^2 \tag{1.14}$$

subject to (1.13). This quadratic program is designed to select simultaneously

1. a few models with large weights. Quadratic Program 3 encourages this diversity more than Quadratic Program 2, by virtue of the addition of the "cross products" of the weights
2. smaller values of ε (if possible) than Quadratic Program 2, by minimizing the square of ε.

Quadratic Program 4 (QP4)

The final quadratic program is given by:

$$\min_{\underset{\sim}{\omega},\varepsilon} \quad q(\underset{\sim}{\omega},\varepsilon) = \omega_1\omega_2 + \omega_1\omega_3 + ... + \varepsilon \tag{1.15}$$

subject to (1.13). This objective function is designed to favor only a few models with non-zero weights. The rectangular constraint boundary is broader than Quadratic Program 3, because the objective function no longer minimizes the square of ε.

Many algorithms exist to solve various types of quadratic programs. We used the Optimization Toolbox in MATLAB (2005) which employs the active set method. The active set method can be used with both convex and non-convex quadratic programs. In this context, if the number of models is greater than the number of observational units, G is indefinite and the active set method for non-convex problems is employed. The readers are directed to (Fletcher, 1987; Nocedal and Wright, 1999) for further information regarding the active set method.

2.3 Genetic Algorithms

A genetic algorithm can be used to calculate which models are the most relevant in the ensemble. Each potential solution is encoded via a chromosome and the set of all chromosomes at any iteration is referred to as the population (Davis, 1991). Each chromosome represents important models in the ensemble by a '1' in the corresponding bit position and superfluous models with a '0' in the corresponding bit position.

The initial generation is randomly generated by setting bits to zero with high probability and bits to one with low probability. This reflects the notion that it is likely most models contain no relevant information.

To calculate the fitness values of each chromosome, the set of models deemed to have a non-zero weight by the chromosome ('1' in corresponding bit position) are parsed through a quadratic program (QP1:QP4). The solution to the quadratic program gives the weights for these models only, and this information is parsed back to the fitness evaluation module of the genetic algorithm. The fitness of each chromosome is given by the inverse of the root mean square error:

$$RMSE_k = \sqrt{\frac{1}{n}\left(\underset{\sim}{y} - X_k \underset{\sim}{\hat{\omega}}_k\right)^T \left(\underset{\sim}{y} - X_k \underset{\sim}{\hat{\omega}}_k\right)} \tag{1.16}$$

where $\underset{\sim}{\hat{\omega}}_k$ is the vector of weights given by the solution of the quadratic program for chromosome k; X_k is the predictions matrix of the set of models chromosome k indicates to have non-zero coefficients.

Offspring are then created via roulette wheel selection with one point crossover (Davis, 1991). The selection and reproduction process is repeated until enough offspring are produced to replace the entire generation. The offspring are mutated by setting bits with value '1' to '0' if more than a pre-specified number of bits in any chromosome are '1'. This ensures that the expectation that only a few models are relevant is met.

The current generation is then replaced by the offspring generation and the process iterates until the chromosomes converge to the set of models that give the lowest root mean square error - all chromosomes become virtually identical.

The genetic algorithm should improve upon results obtained using only quadratic programs on the basis of parsimony. By restricting the quadratic program to only a few models the Hessian matrix is no longer potentially indefinite, the quadratic program becomes convex and therefore its solution is global. Owing to its stochastic nature the genetic algorithm is run 100 times using each objective function. The results of these 100 runs are averaged and presented as “GAQP1:GAQP4”. The best result over all the 100 runs for each objective function is also presented as “GAQP1*:GAQP4*”. The algorithm is programmed in MATLAB (2005).

2.4 Evolution Strategies

The main conceptual difference between genetic algorithms and evolution strategies is that the chromosomes of genetic algorithms contain $0's$ and $1's$, and the chromosomes of evolution strategies can contain any real number. Thus, an evolution strategy can be used to calculate the weights of an ensemble directly.

Each chromosome is a possible set of weights. Explicitly, each element of a chromosome contains the weight assigned to the corresponding model. The initial generation is created by setting a few bits of each chromosome as a draw from a random uniform distribution on [0,1]. Each chromosome is then standardized so that the sum of its elements equals one.

The fitness of the k^{th} chromosome is the inverse of the root mean square error, $RMSE_k$:

$$RMSE_k = \sqrt{\frac{1}{n}\left(\underset{\sim}{y} - X\underset{\sim}{\hat{\omega}}_k\right)^T \left(\underset{\sim}{y} - X\underset{\sim}{\hat{\omega}}_k\right)} \tag{1.17}$$

where $\underset{\sim}{\hat{\omega}}_k$ is the k^{th} chromosome (weights are given directly by the chromosome).

To prevent premature convergence during the early iterations and to engender convergence to optimal solutions during the later iterations, the fitness values of the evolution strategies are scaled using the simple rank scaling function (Davis, 1991).

Parent chromosomes are selected via roulette wheel selection and produce offspring via line recombination (Mühlenbein and Schlierkamp-Voosen, 1993). Line recombination is attractive in this instance as it indirectly enforces the condition that the elements of the offspring chromosomes sum to one.

Over time, the evolution strategy may assign non-zero weights to all models, and would not be as appropriate as other post processing techniques if the goal is to find the best, smallest set of non-zero weights. Therefore, we "mutate" the offspring by randomly setting weights to zero if too many weights (more than the number found by the lasso heuristic) are non-zero. The chromosomes are then renormalized such that the elements of each chromosome sum to one.

The offspring replace the current generation (generational replacement) and the process is iterated until the chromosomes converge to the set of weights that minimize the root mean square error. Again owing to the stochastic nature of evolution strategies, the algorithm is run 100 times on each dataset. The results are averaged and are presented as “ES”. The best result over all the runs is presented as “ES*”. The algorithm is programmed in R (2004).

3 PROCEDURE AND DATASETS

3.1 Procedure

Each dataset was partitioned into three subsets (table 1) to allow for an estimate of the true accuracy of the ensemble via a test set. The first training set was used to grow $2M$ individual models. The models included both M linear regression models and M regression trees. Both linear regression and regression tree models were selected because of their simplicity and complementary nature. The M trees were the individual trees of a single random forest (Breiman, 2001). The models were obtained for each dataset using R (2004).

Each regression model used randomly selected variables as predictors. For the datasets with only a small number of predictor variables (Boston Housing and Friedman datasets) there was no restriction placed upon the number of randomly selected predictors. For the dataset with a larger number of predictor variables (Fat dataset) each regression model was restricted to only a small number of randomly selected variables. For all datasets (excluding the Ozone dataset) each variable was included on its own as a potential linear regression model. The Ozone dataset contained categorical variables and hence the linear regression models were not estimated using this dataset.

In a simple ensemble, the predictions of these $2M$ models would be averaged. However, here the post processing techniques (LASSO, QP1-QP4, GAQP1-GAQP4, ES) were employed to find more parsimonious ensembles. The post processing techniques were applied to the second training subset. For the smaller datasets, the first and second training subsets were identical. The third subset, the “test set”, was used to obtain the R^2 of the post processed ensembles.

As a benchmarking tool, the linear regression models were combined using identical weights, $1/M$ and similarly for the regression tree models. The R^2 of these two simple, non-informative ensembles using the test subset of each dataset are given in table 2 in the “SAREG” and “SARF” rows respectively. The linear regression and regression tree models were also combined into a single simple ensemble using identical weights, $1/2M$ (denoted by SAREGRF). The R^2 of both the single best linear regression model (REG) and single best tree (TREE) in the ensembles, chosen on the basis of the second training set errors and applied to the test set, are also reported.

3.2 Datasets

The post processing schemes were tested using four datasets. The datasets are summarized in table 1 and the reader is directed to the references for more detailed explanations. The datasets varied considerably in size and composition structure. The Ozone dataset originally contained missing values; these were removed prior to analysis.

The partition of each dataset into independent subsets was chosen to best mimic previous analyses. In the situation where similar previous studies were either irreproducible or non existent, the partition was chosen such that approximately twenty percent of the data was set aside as an independent test set. If, after the removal of a test set, the dataset was still considerably large, two independent training subsets were employed. Otherwise the two training subsets were identical (Ozone, Fat and Boston Housing datasets).

The choice of M was also chosen on the basis of previous analyses. In the situation where similar previous analyses were unavailable, the point where the error convergence plot of the random forest leveled off was taken as a rough estimate for M. Finally, the table also shows the number of terminal nodes of a tree within the forest, TN. If possible, this parameter was also chosen to mimic previous analyses, otherwise it was chosen such that terminal nodes remained relatively large, creating a stable tree.

Table 1. Description of the Datasets.

Dataset Name and Response	**Dimension (N*p[a])**	**Train/Train/Test**	**M**	**TN**
Ozone (Breiman and Friedman, 1985) available from (Leisch and Dimitriadou, 2005). *Ozone concentration.*	203*12	163/163/40	300	6
Fat (Borggaard and Thodberg, 1992) available from (http://stat.cmu.edu/datasets/). *Fat content of food measured by Tecator Infratec Food and Feed Analyzer.*	215*22	172/172/43	300	6
Boston Housing (Harrison and Rubinfeld, 1978) available from (Leisch and Dimitriadou, 2005). *Median house values.*	506*13[b]	406/406/100	500	4
Friedman (Friedman, 1991) available from (Leisch and Dimitriadou, 2005). *Generated.*	1000*10	400/400/200	500	10

4 RESULTS AND DISCUSSION

The results of applying each post processing technique to each dataset are summarised in tables 2 and 3, and figs. 1 to 4. Table 2 shows the R^2 values obtained using each post processing technique (rows) for the four datasets (columns). Table 3 contains the

[a] Where p is the number of predictor variables.
[b] Note that the Charles river dummy variable was included in the analyses.

number of models selected as non-zero by each post processing technique (rows) for each dataset (columns). The more interesting information from table 3 is expanded upon in the figures. Each figure depicts the models selected as non-zero by the best performing post processors for a single dataset. The post processing techniques are printed on the left hand axis, and their corresponding R^2 are printed on the right hand axis (table 2). The bottom axis shows the number of models, and the top axis indicates the partition of the models into linear regression and tree models (table 1). If a model is deemed to have a non-zero coefficient by a post processing technique, a vertical line is drawn at the coordinates of (model number, post processing technique). The colour of the line indicates the size of the coefficient. Blue lines represent coefficients of one or almost one, green lines represent smaller coefficients, and yellow lines represent the smallest coefficients.

The Ozone dataset is analysed using only tree models. The SARF has a higher R^2 than the TREE (table 2). The LASSO and QP1:QP4 give R^2 similar to the SARF despite removing some of the redundant tree models. The advantages of removing poor models are emphasized with the GAQP1:GAQP4 post processors. These post processing techniques remove the largest number of models from the ensemble (table 3). The averages (over 100 runs) are on par with the LASSO and QP1:QP4 techniques, with far fewer models. However, the best performing GAQP1:GAQP4 and ES produce R^2 values noticeably higher than the other techniques (table 2).

Table 2. R^2 values for the four datasets using each post processing technique.

Dataset / Post Proc.	Ozone	Fat	Boston Housing	Friedman
REG	-	.92	.73	.73
TREE	.66	.66	.76	.52
SAREG	-	.29	.69	.60
SARF	.70	.66	.75	.62
SAREGRF	-	.51	.74	.63
LASSO	.67	.92	.84	.72
QP1	.69	.92	.84	.77
QP2	.68	.90	.77	.73
QP3	.68	.88	.78	.73
QP4	.68	.88	.78	.73
GAQP1	.69	.92	.83	.77
GAQP1*	.76	.93	.85	.78
GAQP2	.67	.92	.84	.73
GAQP2*	.78	.93	.85	.77
GAQP3	.67	.92	.84	.73
GAQP3*	.75	.93	.85	.77
GAQP4	.68	.91	.84	.73
GAQP4*	.80	.93	.86	.77
ES	.69	.90	.82	.77
ES*	.79	.94	.85	.79

Table 3. Number of models selected as non-zero by each post processing technique.

Dataset / Post Proc.	Ozone	Fat	Boston Housing	Friedman
REG	-	1	1	1
TREE	1	1	1	1
SAREG	-	300	500	500
SARF	300	300	500	500
SAREGRF	-	600	1000	1000
LASSO	18	13	23	19
QP1	20	14	18	19
QP2	14	13	16	11
QP3	14	19	18	11
QP4	14	18	16	11
GAQP1	6.66	4.44	5.69	5.57
GAQP1*	6	5	6	6
GAQP2	5.9	3.11	4.07	3.79
GAQP2*	4	3	4	4
GAQP3	4.94	3.27	4.24	3.68
GAQP3*	4	3	4	3
GAQP4	4.31	3.2	4.18	4.06
GAQP4*	5	3	5	3
ES	17.25	12.16	21.27	16.26
ES*	20(7)	15(2)	19(7)	20(6)

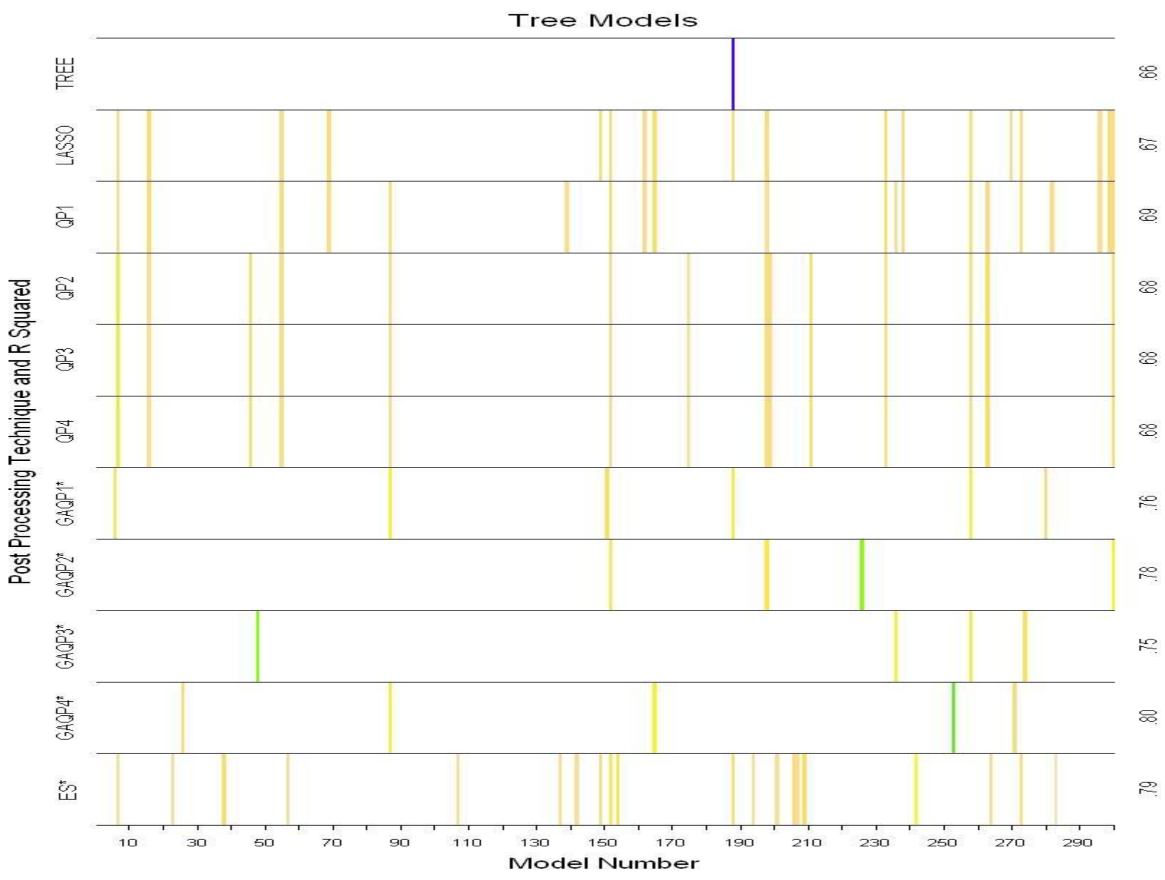


Fig.1. Models selected by the post processing techniques for the Ozone dataset.

The best TREE model is picked by the LASSO and GAQP1* (fig. 1). GAQP1* combines the TREE model with five other models to improve upon the TREE R^2. There is a large degree of overlap between the LASSO and QP1 models. The quadratic programs QP2:QP4 tend to pick very similar models irrespective of the choice of objective function. The best performing genetic algorithms for each objective function tend to contain a subset of the corresponding quadratic program. The ES* picked a large number of models to be non-zero. However closer analysis revealed that seven of these were substantially larger than the others.

The Fat dataset can be predicted better by linear regression models than tree models, because the R^2 of the TREE model is much smaller than the R^2 of the REG model (table 2). However, the reader will also note that not all regression models are predictive because R^2 of the SAREG is particularly small. Surprisingly the TREE R^2 is not improved upon by creating a SARF. The LASSO and QP1:QP4 post processors produce results similar to REG. Again, by reducing the size of the ensembles via genetic algorithms we attain on average results similar to the LASSO and QP1:QP4 techniques, and fractionally better for the best performing runs.

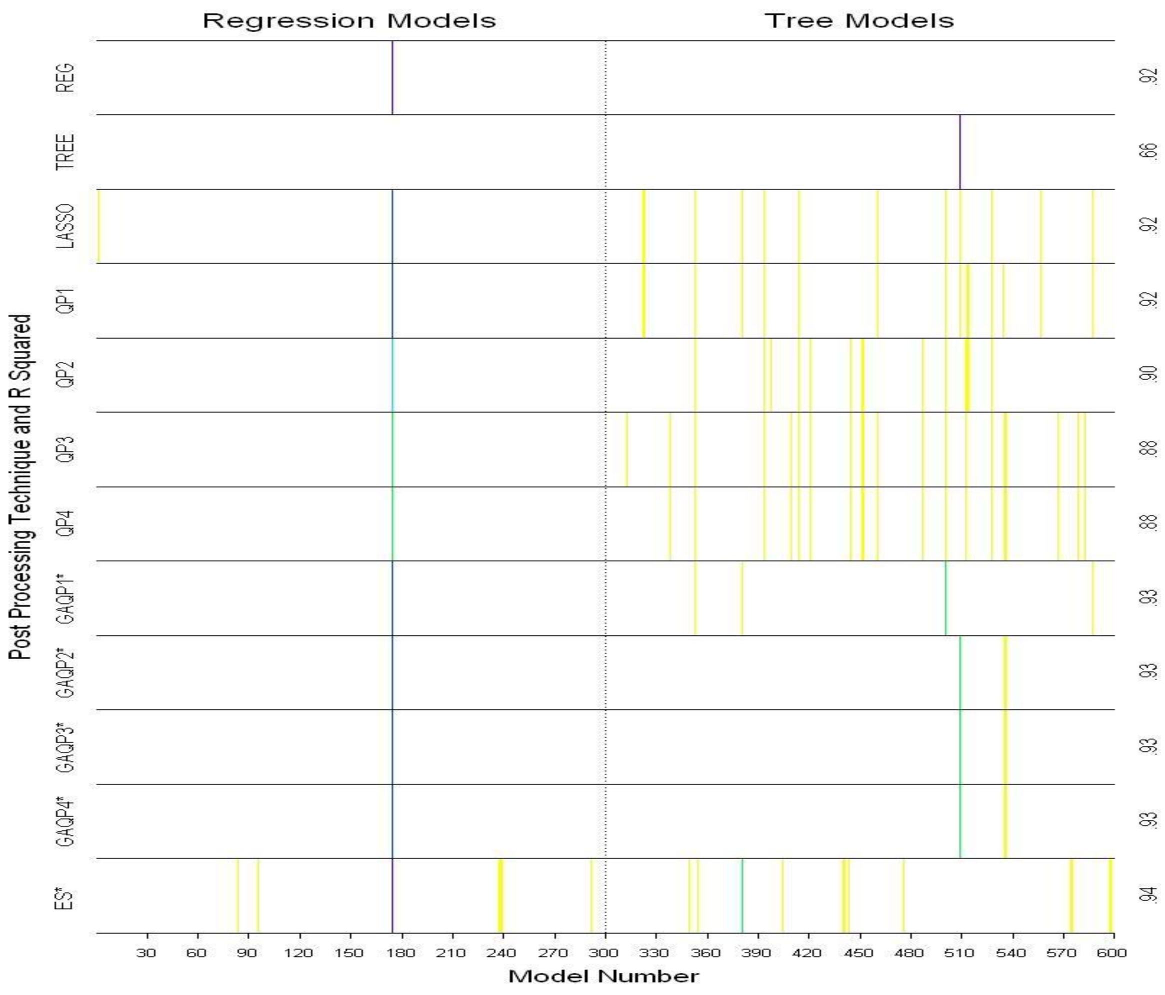


Fig.2. Models selected by the post processing techniques for the Fat dataset.

Not surprisingly the post processors all pick the best REG model (fig. 2). However, the techniques usually do not pick any other linear regression models. The LASSO and QP1:QP4 post processing techniques combine similar regression tree models. Again, the best performing quadratic programs contain a subset of the corresponding quadratic program. The evolution strategy picked a large number of models; however, on closer inspection of the best performing chromosome, only two models were particularly large.

The results for both the Boston Housing and Friedman datasets are included in the tables and figures (figs. 3-4). The results follow the same general trends as those previously discussed for the Ozone and Fat datasets.

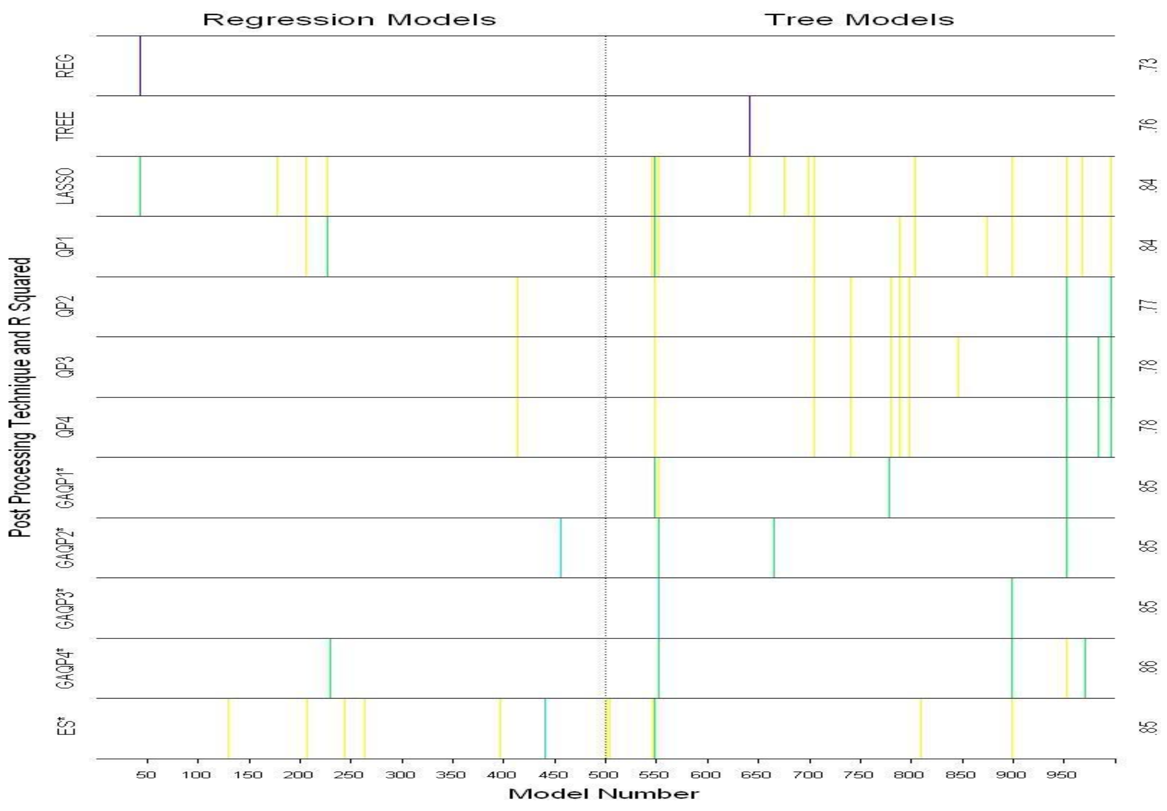


Fig.3. Models selected by the post processing techniques for the Boston Housing dataset.

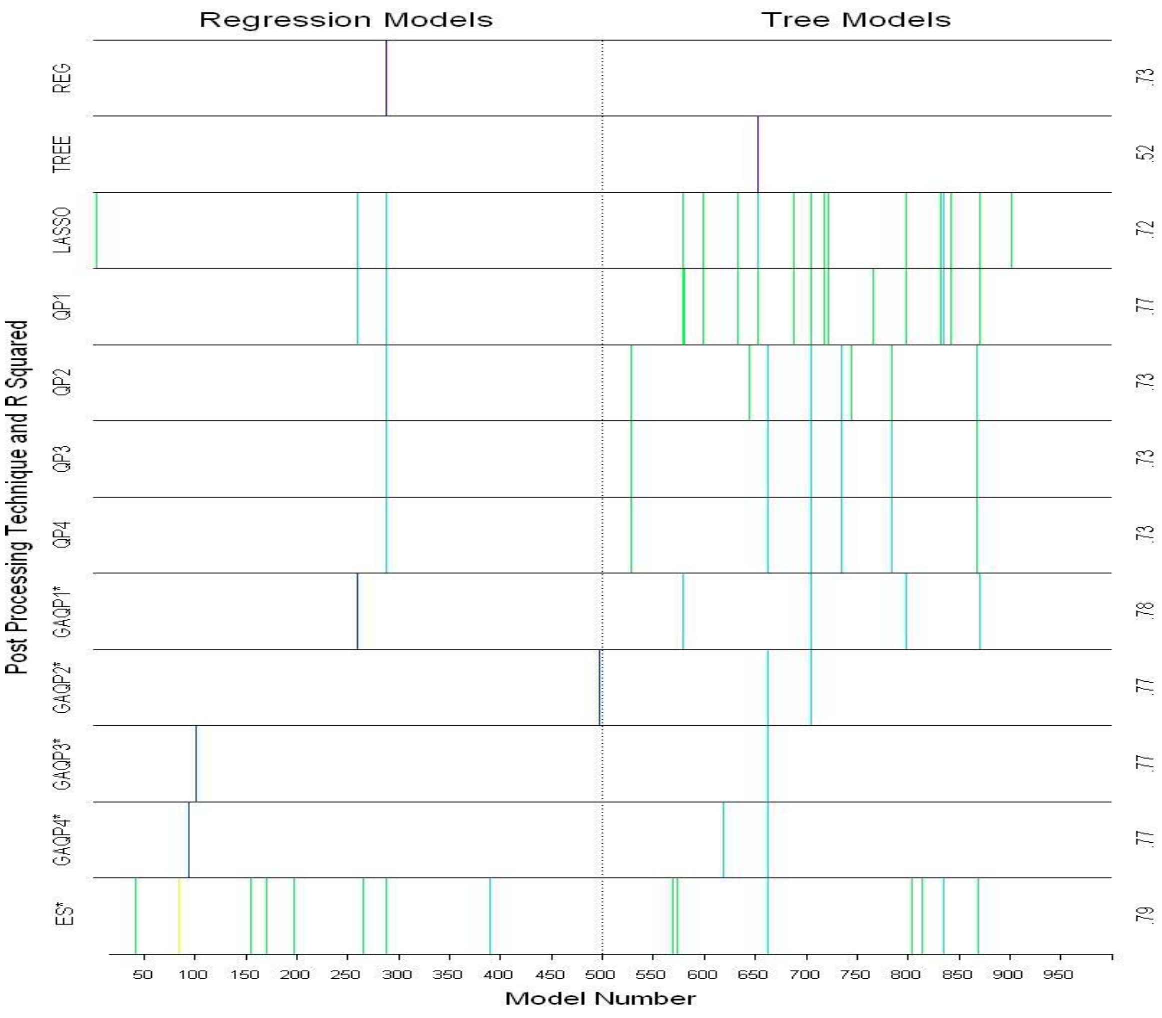


Fig.4. Models selected by the post processing techniques for the Friedman dataset.

The reader will have noticed that while large, simple average ensembles can improve upon the predictive performance of a single model this is not always the case. Obviously, by including “bad” models, the ensemble’s performance can remain on par with the single best model or even decrease dramatically. However, post processing improves the predictive performance of the large ensembles as evidenced by the presented results. Overall, QP1 always gave (with one exception) an R^2 value equal to or higher than the best single model or simple ensemble. QP1 fractionally outperformed the LASSO post processing technique, which in turn fractionally outperformed the other quadratic programming techniques. Furthermore, enforcing very small ensembles via genetic algorithms and evolution strategies marginally improved upon the lasso and quadratic programming results. However, it must be noted that the genetic algorithms and evolution strategies are computer intensive, and the increase in predictive performance may not be worthwhile if speed is important.

Other evidence supporting post processed ensembles over simple average ensembles includes the colours on the figures. The colours are not uniform: some models are given higher weights within the reduced ensembles. It is quite possible that even using a simple average of entirely "good" models will not attain optimal predictive performance.

Further investigation of the figures show there was an amount of overlap between the models selected by each post processing technique. The reader will have noticed the similarity of the quadratic programs QP2, QP3, and QP4 to each other, and therefore will not be surprised by the results. QP1 has a different objective function to the other quadratic programs and hence its selected models did not agree entirely with the other quadratic programs' selected models. The QP1 results agreed quite closely with the LASSO results. This is explained by the fact that both the QP1 and the LASSO techniques were endeavouring to minimize the same function, with slightly different constraints. The evolution strategies are a random technique guided by minimizing the same function as the QP1 and the LASSO. However, their random nature induced different subsets to the LASSO and QP1 techniques.

The ensembles tended to combine more trees than linear regression models. It is reasonable to assume the tree models (weak learners) would not have been as predictive as the linear regression models. It is possible that the greater proportion of trees selected indicates that a few linear regression models can predict the data well, but in places of the dataset the predictions need to be slightly “jiggled” up or down. Further investigation would determine if this were indeed the case.

5 CONCLUSION

Realistically, not all models in a regression ensemble are predictive. Post processing can be employed to create a parsimonious ensemble where a subset of accurate models remains and the poor models are removed. Parsimonious regression ensembles can be considered a balance between predictive accuracy and stability. Parsimonious ensembles facilitate the discovery of underlying trends within a dataset, especially if the ensemble combines different modelling methods, and also increase the interpretability of the ensemble. The results of this research supported the notion that parsimonious regression ensembles can achieve higher predictive accuracy than the single best models or simple average ensembles measured using the R^2 of an independent test set. Such findings are in agreement with other studies. Here, in terms of speed, the lasso heuristic was much faster than the other techniques. However, the best post processing techniques in terms of predictive accuracy were quadratic programming combined with genetic algorithms and evolution strategies. Future work could incorporate other base models such as support vector machines and neural networks into the ensemble. Also, common feature selection techniques could be trialled as potential post processors.

ACKNOWLEDGMENT

Christine Smyth is a recipient of a James Cook University Prestige Postgraduate Research Scholarship.

REFERENCES

Borggaard, C. & Thodberg, H. H. (1992) Optimal Minimal Neural Interpretation of Spectra, *Analytical Chemistry*, 64,545-551.

Breiman, L. (2001) Random Forests, *Machine Learning*, 45(1),5-32.

Breiman, L. & Friedman, J. H. (1985) Estimating Optimal Transformations for Multiple Regression and Correlation, *Journal of the American Statistical Association*, 80(391),580-598.

Davis, L. (1991) *Handbook of genetic algorithms* (New York: Van Nostrand Reinhold).

Fletcher, R. (1987) *Practical Methods of Optimization* (Chichester: John Wiley & Sons).

Friedman, J. (1991) Multivariate Adaptive Regression Splines, *The Annals of Statistics*, 19(1),1-67.

Friedman, J. & Popescu, B. (2003) Importance Sampled Learning Ensembles, Stanford University, Department of Statistics.

Friedman, J. & Popescu, B. (2004) Gradient directed regularization for linear regression and classification, Stanford University, Department of Statistics.

Harrison, D. & Rubinfeld, D. (1978) Hedonic housing prices and the demand for clean air, *Journal of Environmental Economics and Management*, 5(1),81-102.

Hastie, T., Tibshirani, R. & Friedman, J. (2001) *The Elements of Statistical Learning* (New York: Springer-Verlag).

Heskes, T. (1997). Balancing between bagging and bumping, in M. Mozer, M. Jordan and T. Petsche (Ed.) *Advances in Neural Information Processing Systems*, 466-472 (Cambridge: The MIT Press).

Heskes, T. (1998). Selecting weighting factors in logarithmic opinion pools, in M. Jordan, M. Kearns and S. Solla (Ed.) *Advances in Neural Information Processing Systems*, 266-272 (Cambridge: The MIT Press).

Hoerl, A. E. & Kennard, R. (1970) Ridge Regression: Biased estimation for nonorthogonal problems, *Technometrics*, 12,55-67.

Krogh, A. & Vedelsby, J. (1995). Neural Network Ensembles, Cross Validation, and Active Learning, in G. Tesauro, D. S. Touretzky and T. K. Leen (Ed.) *Advances in Neural Information Processing Systems*, 231-238 (Cambridge: The MIT Press).

Leisch, F. & Dimitriadou, E. The mlbench Package for R - Machine Learning Benchmark Problems. 2005.

MATLAB. 7.0.4. 2005. The MathWorks, Inc.

Mühlenbein, H. & Schlierkamp-Voosen, D. (1993) Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization, *Evolutionary Computation*, 1(1),25-49.

Nocedal, J. & Wright, S. J. (1999) *Numerical Optimization* (New York: Springer-Verlag).

R. A language and environment for statistical computing. 2004. R Foundation for Statistical Computing.

Smyth, C. & Coomans, D. (2006). Creating Parsimonious Ensembles. 38th Symposium on the interface of statistics, computing science, and applications, Los Angeles.

Tibshirani, R. (1996) Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 58(1),267-288.

Zhou, Z.-H., Wu, J.-X., Jiang, Y. & Chen, S.-F. (2001). Genetic Algorithm based Selective Neural Network Ensemble. 17th International Joint Conference on Artificial Intelligence, Seattle, 797-802.

SYNOPSIS

These manuscripts combined different regression models using various post processing techniques. The results of both manuscripts show that parsimonious weighted ensembles are more accurate than simple average ensembles or single best models. The results were measured using the R^2 of an independent test set. Some further results that were not included in the manuscripts are explicated below.

It is known that the lasso coefficients are equivalent to the modes of the conditional posterior distributions if a double exponential prior for each regression coefficient is used. In “Parsimonious Ensembles for Regression”, the reported estimates are the expectations of the posterior distributions, not the modes. The difference between the expectations and the modes was minimal. If the mode had been used the post processed weights would have been much sparser, because the estimates would have been absolutely zero. This explains why the first iterative scheme gives results that are almost identical to the lasso heuristic. The iterative scheme uncovers the largest weights using a double exponential prior: these correspond to the weights that would have been non-zero if the posterior mode had been employed. These results also illustrate the similarity of the lasso heuristic and the true lasso (Bayesian linear regression with double exponential priors).

The manuscript, “Parsimonious Ensembles for Regression” used a hybrid post processing strategy, where genetic algorithms were run in conjunction with Bayesian linear regression. Figure A shows that there is only a small difference between the models selected using the four prior types (double exponential, Weibull, multivariate normal, and multivariate t) when combined with a genetic algorithm. The left hand y-axis shows the prior type, and the right hand y-axis shows the test set R^2 value. A line is drawn if a weight coefficient is non-zero. The color of the line corresponds to the size of a coefficient: large coefficients are blue, small coefficients are yellow. The priors do induce slightly different models, indicating that they may be performing some selection. However, it is more likely that the large degree of overlap amongst priors is indicative that there is only minimal model selection being performed by the priors and the brunt of selection is being performed by the genetic algorithm. This suggested that a purely stochastic strategy may produce accurate results without the need for Bayesian linear regression. Therefore, in “Post processing regression ensembles: imposing parsimony to improve predictions”, evolution strategies were trialed as a potential post processing technique.

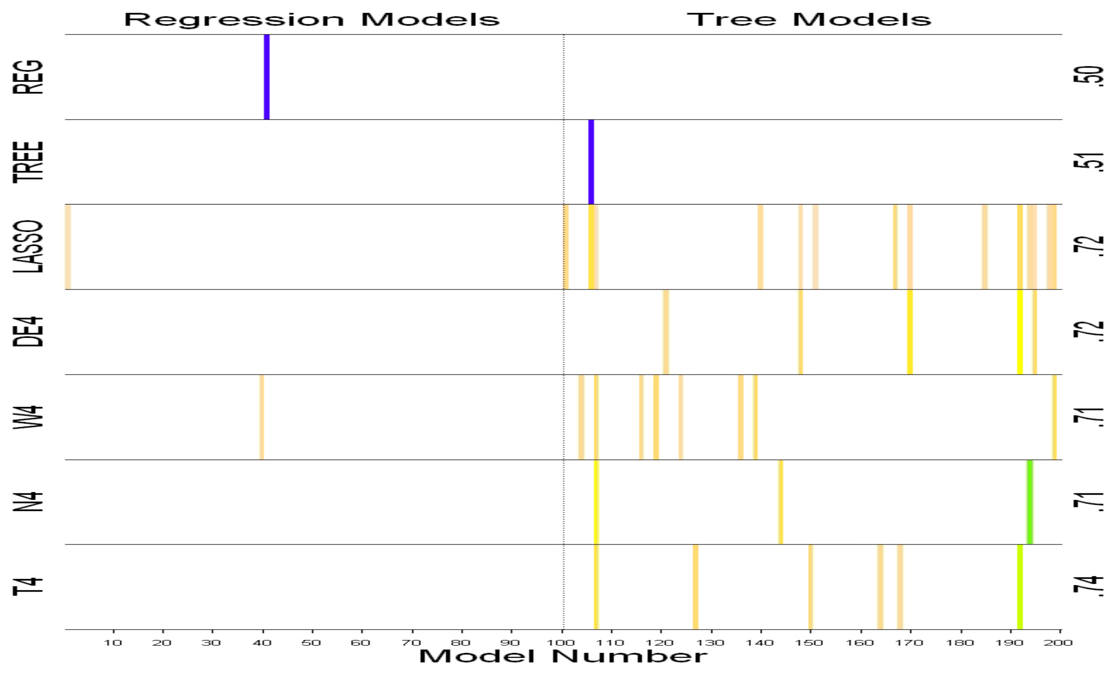


Figure A. Models selected as non-zero for the Friedman dataset using Bayesian linear regression with genetic algorithms.

The manuscript, “Post processing regression ensembles: imposing parsimony to improve predictions” presented the “best” regression models as those with the highest predictive accuracies within the ensembles. The models were generated with random inputs so there was no guarantee that the "best" regression model of the ensembles was indeed the optimal predictive model. To calculate a more predictive model, stepwise linear regression was performed on the datasets. It was found that the test set R^2's using stepwise linear regression did not increase noticeably as shown in Table A.

Table A. Comparison of the best regression models with the stepwise linear regression models.

Dataset	R^2 of Best Regression Model of "Post processing regression ensembles: imposing parsimony to improve predictions"	R^2 of Stepwise Linear Regression Model
Fat	0.92	0.93
Friedman	0.73	0.73
Boston Housing	0.73	0.73

In "Post processing regression ensembles: imposing parsimony to improve predictions" the evolution strategies produced excellent results. Similarly, the quadratic programs with genetic algorithms also produced commendable results. However, the evolutionary algorithms are particularly computer intensive. The lasso heuristic was much faster than any of the other techniques. In both manuscripts, the lasso produced weighted ensembles with accuracies approaching the best solutions. Additionally, the lasso is well known to produce sparse estimates of the weight coefficients, thereby enforcing the desired parsimony over the ensemble. Therefore, it was decided to henceforth post process the regression trees using the forward stagewise approximation to the lasso.

By weighting regression trees, the implicit cluster solutions are also weighted according to their predictability. Weighting cluster solutions should improve the cluster ensemble's accuracy on the basis of parsimony: by removing redundant or inaccurate models, the overall ensemble quality is increased. In the next section, the regression trees' cluster solutions are post processed using the lasso heuristic (extended for the multivariate case) and the overall result is a weighted co-occurrence matrix which is partitioned using a novel technique, SBK. The objective is to ascertain if the predictive weighting of cluster ensembles improves their accuracy.

PREDICTIVE WEIGHTING FOR TREE-BASED CLUSTER ENSEMBLES

OVERVIEW

Recently, improved accuracy and stability have been obtained by combining the results of many clustering solutions. However, weighting individual solutions within the ensemble has remained relatively unexplored because it is difficult to assess the accuracy of an individual solution. This section shows how the clustering solutions within a tree-based ensemble can be weighted according to their predictive accuracies using the research of the previous two sections.

It has been shown in this thesis that AAMRTs, MRTPCs, and MRTFSs find stable group structure within large datasets. Obviously, many trees could be grown to obtain an ensemble of clustering solutions. A post processing technique can be applied to the ensemble by momentarily disregarding the grouping structure and focusing on the ensemble as a pure regression ensemble, where each tree is trying to predict a multivariate response: either the explanatory variables (AAMRT); the principal component scores (MRTPC); or the factor scores (MRTFS). The superior technique of the previous section, the lasso, can be extended such that it finds the trees that best predict the multivariate response, and assigns these trees a high weight. This process will unearth an ensemble that describes predictive group structure. The focus is returned to the cluster ensemble rather than the tree-based regression ensemble via the creation of a weighted co-occurrence matrix.

Each tree's co-occurrence matrix is multiplied by the tree's assigned weight and aggregated. The weighted co-occurrence matrix represents the predictive group structure over the entire ensemble in a manageable form. This section also illustrates an innovative technique of splitting the weighted co-occurrence matrix, similarity-based k-means.

Furthermore, the modified figures of merit are also demonstrated; and the ensemble's weighted variable importance list is also established. The first manuscript, "Predictive Weighting for Cluster Ensembles" uses ensembles of AAMRTs on three small datasets and establishes the entire methodology.

The second manuscript, "Clustering Microarrays with Predictive Weighted Ensembles", applies the developed methodology to two DNA microarray datasets. Ensembles of all tree types (AAMRTs, MRTPCs, MRTFSs) are illustrated and potential biomarkers are presented. Although each manuscript contains the necessary theory, some theory is elaborated in the "Supporting Theory" appendix.

Predictive Weighting for Cluster Ensembles

Christine Smyth[*] and Danny Coomans

Statistics and Intelligent Data Analysis Group,

School of Mathematics, Physics and Information Technology,

James Cook University,

Australia

Accepted

Journal of Chemometrics

SUMMARY

An ensemble of regression models predicts by taking a weighted average of the predictions made by individual models. Calculating the weights such that they reflect the accuracy of individual models (post processing the ensemble) has been shown to increase the ensemble's accuracy. However, post processing cluster ensembles has not received as much attention because of the inherent difficulty in assessing the accuracy of an individual cluster model. By enforcing the notion that clusters must be "predictable", this paper suggests a means of implicitly post processing cluster ensembles by drawing analogies with regression post processing techniques. The product of the post processing procedure is an intelligently weighted co-occurrence matrix. A new technique, similarity-based k-means, is developed to split this matrix into clusters. The results using three real life datasets underpinned by chemical and biological phenomena show that splitting an intelligently weighted co-occurrence matrix gives accuracy that approaches supervised classification methods.

KEYWORDS: post processing, cluster ensembles

1. INTRODUCTION

Ensembles are becoming accepted within the field of Chemometrics, see for example [1-3]. Ensembles of regression models usually average the predictions from individual models grown on bootstrap datasets to give an ensemble prediction [4]. Ensembles of classification models usually classify an observational unit using the majority vote of the individual models grown on bootstrap datasets [4]. However, within the ensemble there are some "good" models and some "bad" models. Post processing, the process of suggesting a weight for each model which reflects its relative accuracy, has been shown to give improvements over simple average regression ensembles and majority vote classification ensembles [5].

Cluster models can also be combined to create a cluster ensemble [6, 7]. Often, this involves the creation of a "co-occurrence matrix". The co-occurrence matrix indicates how often observational units have been clustered together by the individual cluster models. The matrix is then split to give an ensemble clustering solution, see for example [6, 8]. Other techniques of creating cluster ensembles involve applying optimization techniques over the cluster labels, see for example [9].

Fern et al. [10] state that "There is no doubt that some of the base clusterings may be better than others.". By removing redundant models and using a more parsimonious solution, the accuracy of the ensemble should ultimately be increased. However, post processing cluster ensembles is relatively unexplored because it is impossible to assess the accuracy of an individual model. This is a well known problem of cluster analysis: there is no "gold standard".

This paper uses a regression technique (auto-associative multivariate regression trees [11]) that doubles as a clustering technique to create individual clustering solutions. The individual models are combined using regression post processing methodology (lasso [12]). Therefore, the post processed ensemble does not rely on any definition of a cluster to calculate the weights. It focuses only on improving the predictability of the ensemble. "Predictability" in cluster analysis is gaining respect. Grotkjaer et al. [13] states that "a good clustering has predictive power". Yeung et al. [14] and Tibshirani et al. [15] both assert that by assessing the predictability of a clustering solution, the quality of the clusters can be determined.

Auto-associative multivariate regression trees (AAMRTs) are an extension to multivariate regression trees. Multivariate regression trees are a common method of predicting a multivariate response. At each stage, the dataset is recursively split in two on the basis of an explanatory variable, such that the two new nodes are more homogeneous with respect to the response variables. Nodes that are deemed sufficiently homogenous are "terminal nodes". It has been shown that by duplicating the explanatory variables as the response variables (referred to as an AAMRT), a regression tree can be grown on a dataset consisting of only explanatory variables, and the observational units in the terminal nodes can be regarded as the clusters of the dataset [11, 16]. Obviously, each regression tree can give an individual co-occurrence matrix. The $(i, j)^{th}$ element of the matrix is one if the two observational units fall into the same terminal node and zero otherwise. These co-occurrence matrices are an integral part of this research.

It is possible to assess the accuracy of an AAMRT by considering it entirely in the regression context. Each node contains a predicted value: the mean of the observational units in the training sample that fall within that node. By using the squared error loss between the observational units and their predicted values, the accuracy of the tree can be assessed. After growing an ensemble of trees, regression post processing methodology can be applied, and trees with the highest predictive accuracy are given the highest weights. The post processing strategy used here is the lasso technique [12]. The ensemble co-occurrence matrix is obtained by multiplying each tree's co-occurrence matrix with its corresponding lasso weight and summing the weighted matrices together. One major difference between this work and others is that the similarity matrix is a weighted sum of individual co-occurrence matrices.

The weighted co-occurrence matrix could be split by converting it to a dissimilarity matrix and then using any clustering technique that takes a dissimilarity matrix as input (e.g. partitioning around medoids). However, we propose a new technique of clustering the co-occurrence matrix: similarity-based k-means. Similarity-based k-means (SBK) enforces the predictability of the solution by explicitly predicting the group structure found within the entire similarity matrix (including the covariance submatrices) shown to be important in [17].

To gain an estimate of the natural number of clusters in the dataset, Figure of Merits (FOMs) [14] are modified such that they can be used in conjunction with SBK. A variable importance measure is also produced using the individual models' variable importance lists.

We compare across three datasets, benchmarking to a single cluster solution and a "simple average" cluster ensemble. The results show that post processed ensembles give a marked improvement over simple average ensembles, with accuracy approaching a classification tree.

2. THEORY

2.1. Auto-Associative Multivariate Regression Trees

Regression trees [18, 19] begin with all the data in one node. At each stage, the regression tree partitions a non-split node in two. Regression trees partition a node, t, into two subsets, t_L and t_R, on the basis of the value of an explanatory variable. At each node all possible splits of each explanatory variable are considered. The optimal split is saved for each node. The node with the split that maximizes the decrease in $R(T)$ is partitioned at each stage. $R(T)$ is given by:

$$R(T) = \frac{1}{n}\sum_{t\in\tilde{T}}\sum_{\mathbf{x}_i\in t}\left(\mathbf{y}_i - \overline{\mathbf{y}}(t)\right)^T\left(\mathbf{y}_i - \overline{\mathbf{y}}(t)\right) \tag{1.1}$$

where $\mathbf{x}_i$ is the vector of measurements of P explanatory variables for the i^{th} observational unit; $\mathbf{y}_i$ is the vector of measurements of the response variables for the i^{th} observational unit; $\tilde{T}$ is the set of all terminal nodes and; $\overline{\mathbf{y}}(t)$ is the mean response vector of terminal node t.

After growing the tree, the non-split nodes are deemed "terminal". The predicted value for a terminal node, t_{term}, is:

$$\hat{\mathbf{y}}(t_{term}) = \frac{1}{n_{t_{term}}}\sum_{\mathbf{x}_i\in t_{term}}\mathbf{y}_i \tag{1.2}$$

where the sum is over all $\mathbf{y}_i$ such that $\mathbf{x}_i \in t_{term}$ and $n_{t_{term}}$ is the total number of cases in the terminal node.

The observational units in each of the terminal nodes are the clusters of the dataset: the terminal nodes are as homogeneous as possible reflecting the intuitive definition of a cluster. The clusters are found in the response space and the explanatory variables that form the tree are deemed to be important in determining the clusters. To allow multivariate regression trees to be applied in the traditional clustering framework where there are no response variables, AAMRTs were suggested [11, 16]. AAMRTs replicate the explanatory variables as response variables and grow the tree using identical response and explanatory datasets. This implies that the clusters will be as homogenous as possible with respect to all the explanatory variables.

2.1.1. Algorithm for creating ensembles of AAMRTs

Algorithm 1 shows the process used to grow an ensemble of regression trees such that they can be used to create a cluster ensemble.

Algorithm 1: Growing an ensemble of trees

1. Choose the number of individual trees in the ensemble, M.
2. Replicate the original dataset as the response dataset, $\mathbf{Y}$.
3. Create M explanatory datasets by randomly sampling variables with percentage p_v from the original dataset. Here we use $p_v = 0.33$. We stress that although the variables may be sampled to create the explanatory datasets, the response for each tree is always the entire dataset.
4. Grow an AAMRT using each explanatory dataset to predict $\mathbf{Y}$ to k terminal nodes (clusters). Create a co-occurrence matrix for each AAMRT $\mathbf{C}(m)$. Create a variable importance list for each tree using Algorithm 5.

2.2. Lasso Heuristic

A regression ensemble can be represented by:

$$F(\mathbf{x}) = \sum_{m=1}^{M} \omega_m f_m(\mathbf{x}) \tag{1.3}$$

where $f_m(\mathbf{x})$ is the prediction of an observational unit $\mathbf{x}$ by the m^{th} model - the $f_m(\mathbf{x})$ are usually of the same family of models but this is not mandatory; ω_m is the weight assigned to $f_m(\mathbf{x})$; and M is the number of models.

The individual regression solutions are combined to form an ensemble by taking a weighted sum of the individual solutions. Usually, the weights are an average of the number of models, $1/M$, see for example [4]. Post processing is a procedure which suggests choices of ω_m that reflect the relevance of each $f_m(\mathbf{x})$ [5]. Post processing usually achieves greater accuracy by enforcing parsimony. The lasso [12] post processing procedure finds the weights that minimize

$$\hat{\boldsymbol{\omega}} = \arg\min_{\boldsymbol{\omega}} \sum_{i=1}^{n} \left(\mathbf{y}_i - \sum_{m=1}^{M} \omega_m f_m(\mathbf{x}_i) \right)^T \left(\mathbf{y}_i - \sum_{m=1}^{M} \omega_m f_m(\mathbf{x}_i) \right) + \lambda \sum_{m=1}^{M} |\omega_m| \tag{1.4}$$

Here, the solution to the lasso is approximated with a forward stagewise algorithm [4] which is henceforth referred to as the “lasso heuristic”. The algorithm is as follows:

Algorithm 2: The lasso heuristic

1. Set all weights to zero. Choose ε as a small number greater than zero, and choose the number of iterations, its, to be quite large.
2. for $l = 1:its$

    2.1. $\left(\beta^*, h^*\right) = \arg\min_{\beta,h} \sum_{i=1}^{n} \left(\mathbf{y}_i - \sum_{m=1}^{M} \omega_m f_m(\mathbf{x}_i) - \beta \times f_h(\mathbf{x}_i) \right)^T \left(\mathbf{y}_i - \sum_{m=1}^{M} \omega_m f_m(\mathbf{x}_i) - \beta \times f_h(\mathbf{x}_i) \right)$

    2.2. $\hat{\omega}_{h^*} = \hat{\omega}_{h^*} + \varepsilon \times sign(\beta^*)$

3. $F(\mathbf{x}) = \sum_{m=1}^{M} \hat{\omega}_m f_m(\mathbf{x})$

In the first step all weights are zero, and this is analogous to $\lambda = \infty$ in equation (1.4). The parameter *its* is inversely related to λ in (1.4). After the set number of iterations, many weights will still remain zero.

2.2.1. Algorithm for creating a weighted co-occurrence matrix

Algorithm 3 shows the process used to create a weighted co-occurrence matrix.

Algorithm 3: Producing a weighted co-occurrence matrix

1. Create an ensemble of trees using Algorithm 1.
2. Post process the ensemble of regression trees to find the weights using Algorithm 2.
5. Create an overall co-occurrence matrix, $\mathbf{C}$ by taking a weighted sum of the individual co-occurrence matrices: $\mathbf{C} = \sum_{m=1}^{M} \hat{\omega}_m \mathbf{C}(m)$

Taking a weighted sum of dissimilarity matrices created from different sources (where the weights were chosen in a "subjective way") was suggested previously by Kaufman et al. [20].

2.3. Similarity-based k-means

Similarity-based k-means is a divisive clustering algorithm that takes a co-occurrence matrix, $\mathbf{C}$, (similarity matrix) as input. Formally, SBK seeks clusters to minimize either of the objective functions:

$$\min \sum_{r=1}^{k} \sum_{i,j \in S_r} \left(C_{i,j} - \bar{C}_r \right)^2 + \sum_{r=1}^{k} \sum_{\substack{r' \neq r \\ r'=1}}^{k} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left(C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right)^2 \tag{1.5}$$

or

$$\min \sum_{r=1}^{k} \sum_{i,j \in S_r} \left| C_{i,j} - \bar{C}_r \right| + \sum_{r=1}^{k} \sum_{\substack{r' \neq r \\ r'=1}}^{k} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left| C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right| \tag{1.6}$$

where k is the number of clusters; i, j index observational units i and j; S_r is the set of observational units in the r^{th} cluster; $C_{i,j}$ is the $(i, j)^{th}$ element of the co-occurrence matrix; $\overline{C}_r$ is the mean similarity of the r^{th} cluster; and $\overline{COV}_{(S_r, S_{r'})}$ is the mean similarity of the (covariance) matrix where the rows are given by the observational units in cluster r and the columns are dictated by the observational units in the r'^{th} cluster. The covariance submatrices should be considered the number of times that observational units in one cluster are grouped with observational units in another cluster during the ensemble creation.

Because of the mean squared and absolute error terms in the objective functions (1.5) and (1.6), SBK can be viewed almost entirely in the prediction sense. The algorithm seeks to predict the entire co-occurrence matrix using the cluster and covariance means. In doing so, observational units with high similarity are grouped together. A validity criterion [17] is imposed to ensure that the clustering ideology prevails over the prediction ideology. The validity criterion dictates that clusters must have higher mean similarities than their covariance matrices. The SBK algorithm is given by Algorithm 4.

Algorithm 4: SBK

1. Choose the number of clusters and an initial partition of the data. Here, we use initial partitions given by both hierarchical clustering of the co-occurrence matrix and entirely random partitions. Choose the objective function; either the mean squared error (1.5) or absolute error (1.6).
2. Visit each observational unit and assign it to the cluster which will result in the largest decrease of the objective function. Before moving the observational unit ensure that the validity criterion is upheld.
3. Update the mean similarity of:

   a. the cluster the observational unit has left
   b. the cluster the observational unit has joined
   c. and all appropriate covariance means.

4. Repeat steps two and three until no more reassignments of the observational units take place.

2.4. Cluster number estimation

SBK allows for an estimate of the number of clusters in the dataset by considering the average predictive capability of the algorithm, for any number of clusters, k. The estimate closely resembles the Figure of Merit (FOM) method proposed by Yeung et al. [14]. FOMs are a method of authenticating clusters by assessing the "predictive power" of a clustering technique. FOMs require no a priori knowledge. FOMs have been shown to provide an accurate estimate of the natural number of clusters [14, 16]. FOMs assess the "predictive power" of a clustering algorithm by leaving out a variable j, clustering the data (into k clusters), then calculating the Root Mean Square Error (RMSE) of j relative to the cluster means:

$$RMSE(j,k)=\sqrt{\frac{1}{n}\sum_{r=1}^{k}\sum_{\mathbf{x}_i\in S_r}\left(x_{ij}-\overline{x}_r(j)\right)^2} \tag{1.7}$$

where x_{ij} is the measurement of the j^{th} variable on the i^{th} observational unit; n is the number of observational units; S_r is the set of observational units in the r^{th} cluster; $\overline{x}_r(j)$ is the mean of variable j for the observational units in the r^{th} cluster.

Each variable is omitted and its RMSE calculated. These RMSE are summed over all variables to give an aggregate FOM (AFOM):

$$AFOM(k)=\sum_{j=1}^{P}RMSE(j,k)\,. \tag{1.8}$$

The AFOM is calculated for each k, and adjusted for cluster size to give $AFOM_{adj}(k)$.

It is simple to expand the above AFOM theory to the results obtained by SBK. Here, no variables are removed from the dataset; the random nature of SBK introduces enough variability. Simply, if the dataset is clustered into k clusters and this process is repeated P times, then the $AFOM(k)$ is defined as

$$AFOM(k) = \sum_{p=1}^{P} \sqrt{\frac{1}{n^2} \sum_{r=1}^{k} \sum_{i,j \in S_r(p)} \left(C_{i,j} - \bar{C}_r(p) \right)^2} \qquad (1.9)$$

where $S_r(p)$ is the set of observational units in cluster r on the p^{th} run; $\bar{C}_r(p)$ is the mean similarity of the observational units in cluster r on the p^{th} run; $C_{i,j}$ is the $(i,j)^{th}$ element of the co-occurrence matrix; and n^2 is the dimension of the similarity matrix. Here, the adjusted figure of merit is given by

$$AFOM_{adj}(k) = \frac{AFOM(k)}{P\sqrt{\frac{n^2 - k}{n^2}}} \qquad (1.10)$$

The $AFOM_{adj}$ is obtained for varying levels of k and the "elbow" of the graph is selected as the optimal number of clusters.

2.5. Variable Importance

One advantage of using multivariate regression trees as the ensemble clustering technique is that they allow for an easy calculation of a variable importance list. Most other clustering techniques do not have this capability. When considering multivariate regression trees, many definitions of variable importance exist, such as those that only consider surrogate splits. We apply a very simple (but naive) definition of variable importance. Our definition of variable importance, if applied to only one tree grown on the entire dataset would over-inflate the importance of some variables and underestimate the importance of others. However, our reasoning is that the random sampling of variables to build each tree will give some stability to our variable importance list that would otherwise not exist. We calculate a variable importance list for each tree in the ensemble using Algorithm 5. The variable importance list for the entire ensemble is then the weighted sum of the variable importance lists for each tree, using the weights given by the lasso heuristic of Algorithm 2.

Algorithm 5: Variable importance list for a single tree

1. For each variable, sum the $\Delta R(t)$ for all splits made by that variable within the tree to obtain the variable importance of $x(j)$, $VI_{x(j)}$. Mathematically, $VI_{x(j)}$ is given by:

   $$VI_{x(j)} = \sum_{\substack{t\in\tilde{T}\text{ where}\\ \text{the node is}\\ \text{split by } x(j)}} \Delta R(t)$$

   where $R(t) = \sum_{\mathbf{x}_i \in t} \left(\mathbf{y}_i - \overline{\mathbf{y}}(t)\right)^T \left(\mathbf{y}_i - \overline{\mathbf{y}}(t)\right)$

   and $\Delta R(t) = R(t) - (R(t_L) + R(t_R))$;

   and t designates the parent node; and t_L and t_R designate the left and right nodes respectively.

   If a variable is not included in the predictor set of a particular tree, its corresponding variable importance for the tree is zero.
2. Standardize the variable importance for the tree such that the individual importances sum to one.

2.6. Cluster Evaluation

Assessing the validity and accuracy of clustering algorithms is not a straightforward task. Various algorithms have been suggested in the recent literature, see for example [14] and [21]. However, in this paper we use datasets with known classifications and we assume these to be the gold standard. As such, we report only the number of "misclassifications" (similar to [13] and [6]), but recognize that in real world settings this is not possible.

3. PROCEDURE AND DATASETS

3.1. Datasets

Three datasets are analysed: the well known Iris [22] dataset and two datasets, Vietnam [23] and Thyroid [24], underpinned by chemical and biological phenomena. The Vietnam dataset consists of measurements of chemical elements in hair samples of six different groups of Vietnamese. There are 17 variables that have been log transformed and z-standardised: Ti, V, Cr, Mn, Ni, Cu, As, Se, Sr, Mo, Cd, Sn, Ba, Pb, Th, U, Hg. The groups differ in their amount of exposure to coal. The Thyroid dataset consists of measurements of five hormones (z-standardized) in patients with differing thyroid functions. The five hormones are: T4, T3, RT3U, TSH, DTSH. Further information surrounding the datasets is included in Table I. All of the datasets have known groupings which were used as "gold standards" to compare the obtained clusterings against. The parameters used in the ensemble creation are also described in Table I.

Table I. Explanation of the three datasets used in the analyses.

Dataset name	Number of observational units * number of variables	Group sizes	Number of trees, M
Iris[22]	150*4	Setosa: 50 Versicolor: 50 Virginica: 50	200
Thyroid[24]	215*5	Normal thyroid function (eu): 150 patients Hyperthyroid function (he): 35 patients Hypothyroid function (ho): 30 patients	200
Vietnam[23]	224*17	Control Adults: 31 males with low exposure to coal Control Children: 31 children with low exposure to coal Miner Adults: 56 males employed at a coal mine Miner Children: 47 children of male coal miners Burner Adults: 18 females using coal for cooking Burner Children: 41 children with exposure to coal through its use for cooking	200

3.2. Procedure

To illustrate the advantages of cluster ensembles, we benchmark them to the following techniques:

1. K-means (average misclassification rate over 100 random starts is reported)
2. Classification Trees [18]. Classification Trees are a supervised classification technique and the reader is directed to the reference for more information.
3. AAMRT
4. Average co-occurrence matrices split with SBK.

An ensemble is constructed by employing Algorithm 1 to create M co-occurrence matrices. The individual AAMRT models require specification of the number of terminal nodes and the minimum terminal node size. To assess the sensitivity of the results to varying values of these parameters, we ran Algorithm 1 with either (5,10,15) minimum terminal node size and either (3,6,10) terminal nodes. There were 3*3=9 choices of parameters and an ensemble of trees was grown for each choice. The M co-occurrence matrices for a set of parameters were then averaged to give a simple average co-occurrence matrix, or weighted using Algorithms 2 and 3. The average and weighted co-occurrence matrices were then split using SBK.

When splitting co-occurrence matrices the minimization criteria (1.5) and (1.6) of SBK were used and both results are shown. The results reported were the most frequently occurring using 15 different starting points. The datasets were split to a maximum of 10 clusters so that the AFOM graphs could be obtained. However, the reported results are those when the co-occurrence matrix is split to the known number of groups in the data. A sample variable importance list is also included. The AFOM results are also reported for the post processed cluster ensembles.

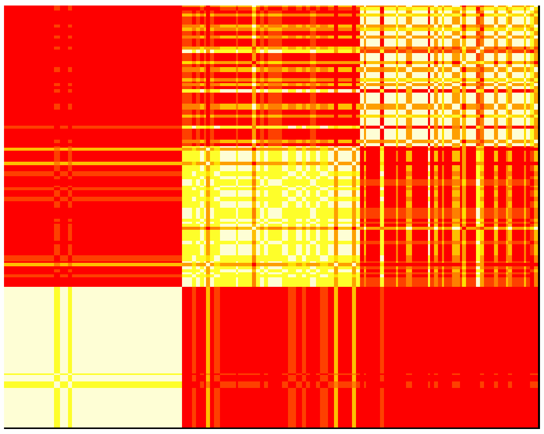
3.3. Software

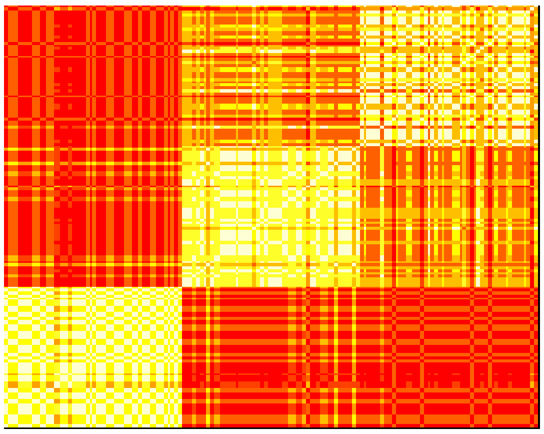
R [25] was used to implement the lasso heuristic, SBK and AFOM algorithms. The AAMRTs and classification trees were created using the mvpart [26] package of R.

4. RESULTS AND DISCUSSION

Firstly a weighted co-occurrence matrix for the Iris dataset is presented on the left hand side of Figure 1. Areas of high similarity are yellow, areas of low similarity are red. The matrix has been reordered according to its groups. There are clearly three groups in the dataset. The average co-occurrence matrix is presented on the right hand side of Figure 1. There is clearly less definition in the average co-occurrence matrix.

Figure 1. Left to right: weighted co-occurrence matrix for the Iris dataset using an ensemble of AAMRTs with three terminal nodes and minimum terminal node size of five; average co-occurrence matrix for the Iris dataset using an ensemble of AAMRTs with three terminal nodes and minimum terminal node size of five.





The results are presented for the three datasets in the following tables. The Iris dataset is considered first. Table II shows the mean number of misclassifications using k-means over 100 random starts. Table II also shows the number of misclassifications of both the AAMRT and classification tree when grown to three terminal nodes (the known number of clusters). Varying the minimum terminal node size (columns) does not have an affect on the number of these misclassifications. The co-occurrence matrices were produced using a variety of minimum terminal node sizes (columns) and number of terminal nodes (rows) of the individual AAMRT models. The co-occurrence matrices are split using SBK to three clusters using both criteria. The top number shows the number of misclassifications using (1.5). The bottom number shows the number of misclassifications using (1.6).

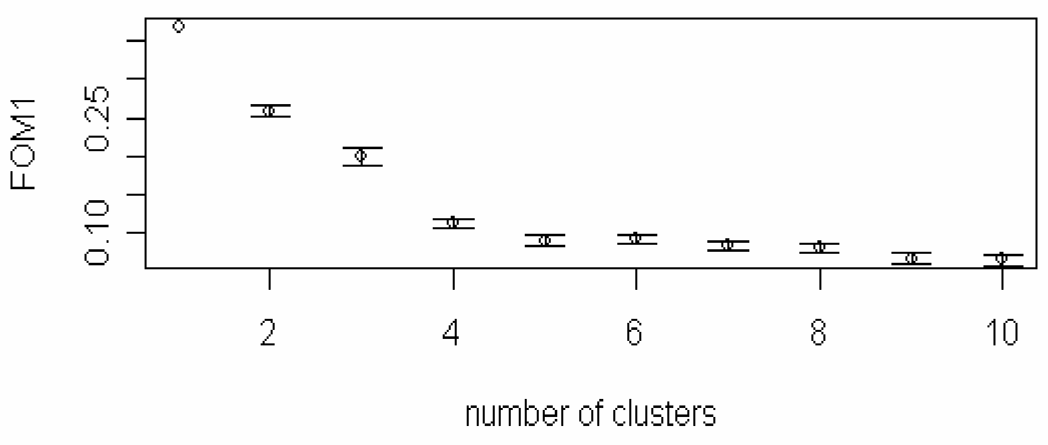
The results show that the simple average co-occurrence matrices created by AAMRTs with three terminal nodes and split with SBK produce results similar to the AAMRT and k-means. The weighted co-occurrence matrices created by AAMRTs with three terminal nodes induce superior results to the simple average co-occurrence matrices, and approach the performance of the classification tree. By allowing the number of terminal nodes to be too large, the co-occurrence matrices and SBK tend to suffer unless the minimum terminal node size is quite high, where the weighted co-occurrence matrix again performs better than the average and approaches the classification tree result.

The AFOM graphs indicate that there are actually four groups in the data. (A sample AFOM graph is given in Figure 2). We split the AAMRT and classification tree to four clusters and report the results in Table III. The classification tree was unable to split to four nodes if the minimum terminal node size was set too high, shown by a 'NP' in Table III. The average and weighted co-occurrence matrices (for all combinations of minimum terminal node size and number of terminal nodes) are also split to four clusters using SBK. The k-means results are also reported. The results show an improvement; again the weighted co-occurrence matrices produce superior results to the average co-occurrence matrices and approach the classification tree performance. There is still a noticeable difference as the number of terminal nodes increases: unless the minimum terminal node size is large enough, the results of the ensemble techniques tend to suffer.

Table II. Results of the Iris dataset split to three clusters.

			Misclassifications		
K-means			24.5		
Tree-based techniques			Minimum terminal node size		
			5	**10**	**15**
AAMRT			19	19	19
Simple average co-occurrence matrix split with SBK	Number of terminal nodes	**3**	17	17	17
			20	20	20
		6	24	26	18
			42	41	24
		10	24	26	18
			42	41	24
Post processed co-occurrence matrix split with SBK	Number of terminal nodes	**3**	9	9	9
			5	5	5
		6	16	39	20
			41	30	9
		10	16	39	20
			41	30	9
Classification Tree			6	6	6

Figure 2. AFOM graphs obtained by splitting the Iris dataset's weighted co-occurrence matrix using SBK to various numbers of clusters. The individual AAMRTs were grown to 10 terminal nodes with a minimum terminal node size of 15. The splitting criterion for the top graph is given by (1.5) and the splitting criterion for the bottom graph is given by (1.6).



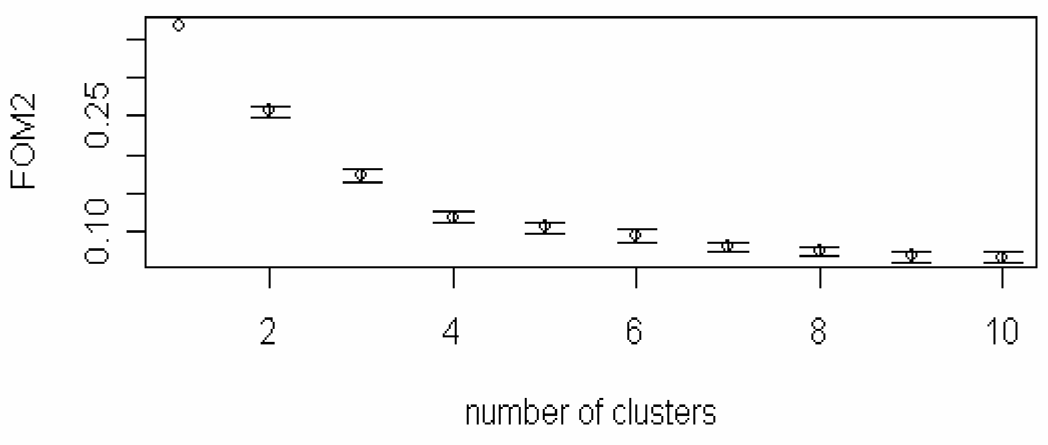
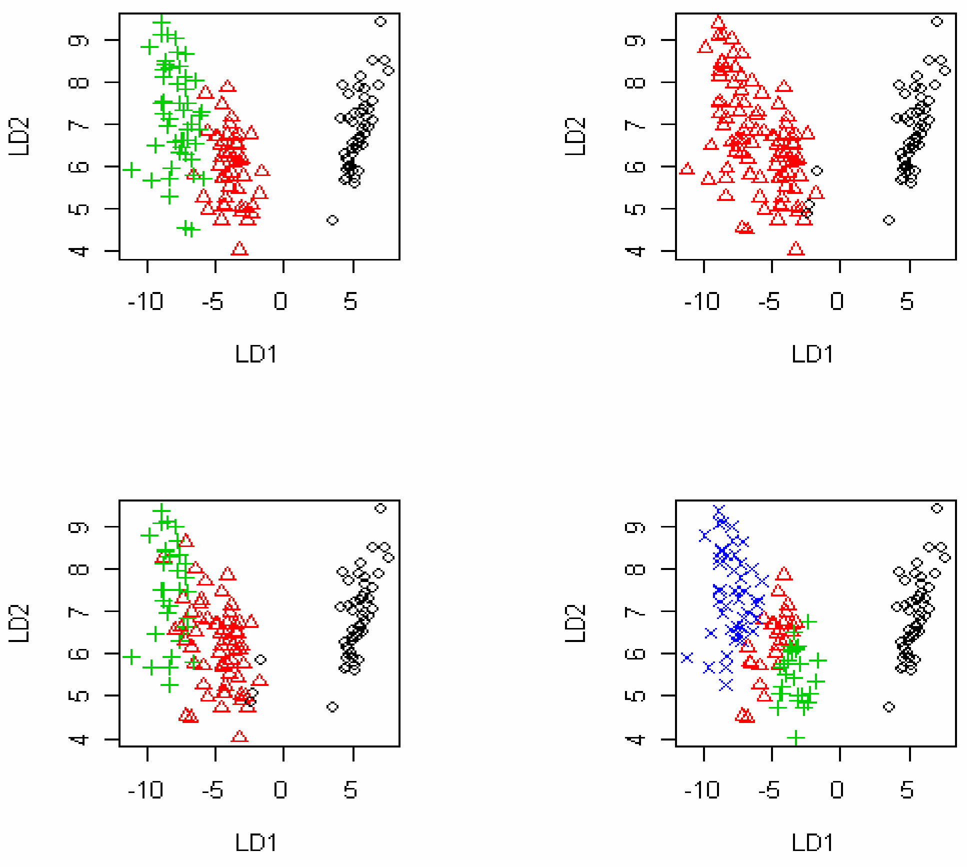


Table III. Results of the Iris dataset split to four clusters.

			Misclassifications		
K-means			18.76		
Tree-based techniques			Minimum terminal node size		
			5	**10**	**15**
AAMRT			19	19	19
Simple average co-occurrence matrix split with SBK	Number of terminal nodes	**3**	17	17	17
			20	20	20
		6	17	17	20
			32	34	20
		10	17	17	20
			32	34	20
Post processed co-occurrence matrix split with SBK	Number of terminal nodes	**3**	7	7	7
			4	4	4
		6	18	19	6
			18	19	9
		10	18	19	6
			18	19	9
Classification Tree			4	NP	NP

Figure 3 shows how SBK splits the weighted co-occurrence matrix into two (top right), three (bottom left) and four (bottom right) clusters. The top left graph shows the three known groups of the Iris dataset. All of the graphs have been projected onto the two-dimensional discriminant plane.

Figure 3. Clockwise from top left: three known groups of the Iris dataset projected onto the discriminant plane; two clusters found by SBK using criterion (1.5) on the weighted co-occurrence matrix; four clusters found by SBK using criterion (1.5) on the weighted co-occurrence matrix; three clusters found by SBK using criterion (1.5) on the weighted co-occurrence matrix. The individual AAMRTs were grown to 10 terminal nodes with a minimum terminal node size of 15.



Finally, to conclude with the Iris dataset analysis we show the splits generated by the SBK algorithm (using the absolute deviations criterion (1.6)) applied to the co-occurrence matrices shown in Figure 1. The matrices are split to three clusters and the resulting splits are shown in Figure 4.

Figure 4. Left to right: weighted co-occurrence matrix for the Iris dataset using an ensemble of AAMRTs with three terminal nodes and minimum terminal node size of five split to three groups using the SBK algorithm with absolute deviation criterion; average co-occurrence matrix for the Iris dataset using an ensemble of AAMRTs with three terminal nodes and minimum terminal node size of five split to three groups using the SBK algorithm with absolute deviation criterion.

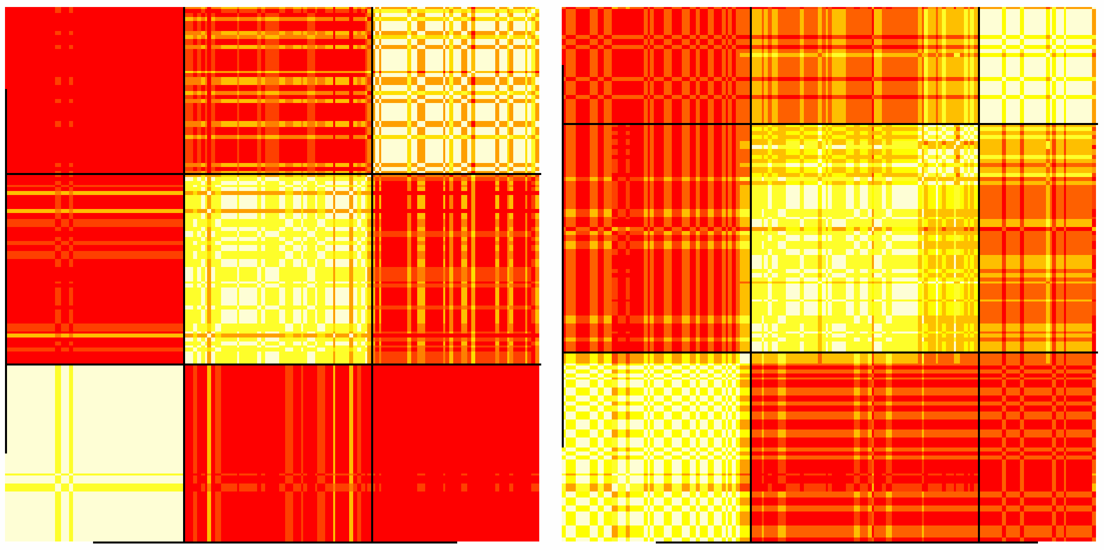


Table IV shows the results when applying the clustering algorithms to the Thyroid dataset. The AAMRT and the classification tree are grown to three terminal nodes (the number of groups) and the results do not depend on the minimum terminal node size. Splitting the average co-occurrence matrices using SBK to three clusters gives less misclassifications than an AAMRT and k-means, however if the trees of the ensemble are grown too large, then the performance is adversely affected. Surprisingly, the weighted co-occurrence matrices do not give superior results to the average co-occurrence matrices. However, on inspection of the AFOM graphs, the majority indicate that the SBK algorithm should split the matrix into four groups. A sample AFOM graph is given in Figure 5.

Therefore, the average and weighted co-occurrence matrices are split to four clusters using SBK and the results are shown in Table V. The results of growing an AAMRT and classification tree to four terminal nodes are also shown. Again, the classification tree could not split to four terminal nodes if the minimum terminal node size was too high, shown by a ‘NP’ in the table. The weighted co-occurrence matrices produce more accurate results than before and a more stable misclassification rate than the average co-occurrence matrix. Also the weighted co-occurrence matrix produces fewer misclassifications than the AAMRT and k-means. There is no longer the effect of individual tree size with the weighted-co-occurrence matrix producing stable results as long as the individual trees are grown as large or larger than the number of groups in the data.

Table IV. Results of the Thyroid dataset split to three clusters.

			Misclassifications		
K-means			27.59		
Tree-based techniques			Minimum terminal node size		
			5	**10**	**15**
AAMRT			27	27	27
Simple average co-occurrence matrix split with SBK	Number of terminal nodes	**3**	14	12	12
			16	16	15
		6	34	30	30
			32	29	29
		10	36	30	30
			34	29	29
Post processed co-occurrence matrix split with SBK	Number of terminal nodes	**3**	23	38	38
			24	37	38
		6	24	41	18
			22	45	45
		10	31	41	18
			13	45	45
Classification Tree			14	14	14

Figure 5. AFOM graphs obtained by splitting the Thyroid dataset's weighted co-occurrence matrix using SBK to various numbers of clusters. The individual AAMRTs were grown to six terminal nodes with a minimum terminal node size of 10. The splitting criterion for the top graph is given by (1.5) and the splitting criterion for the bottom graph is given by (1.6).

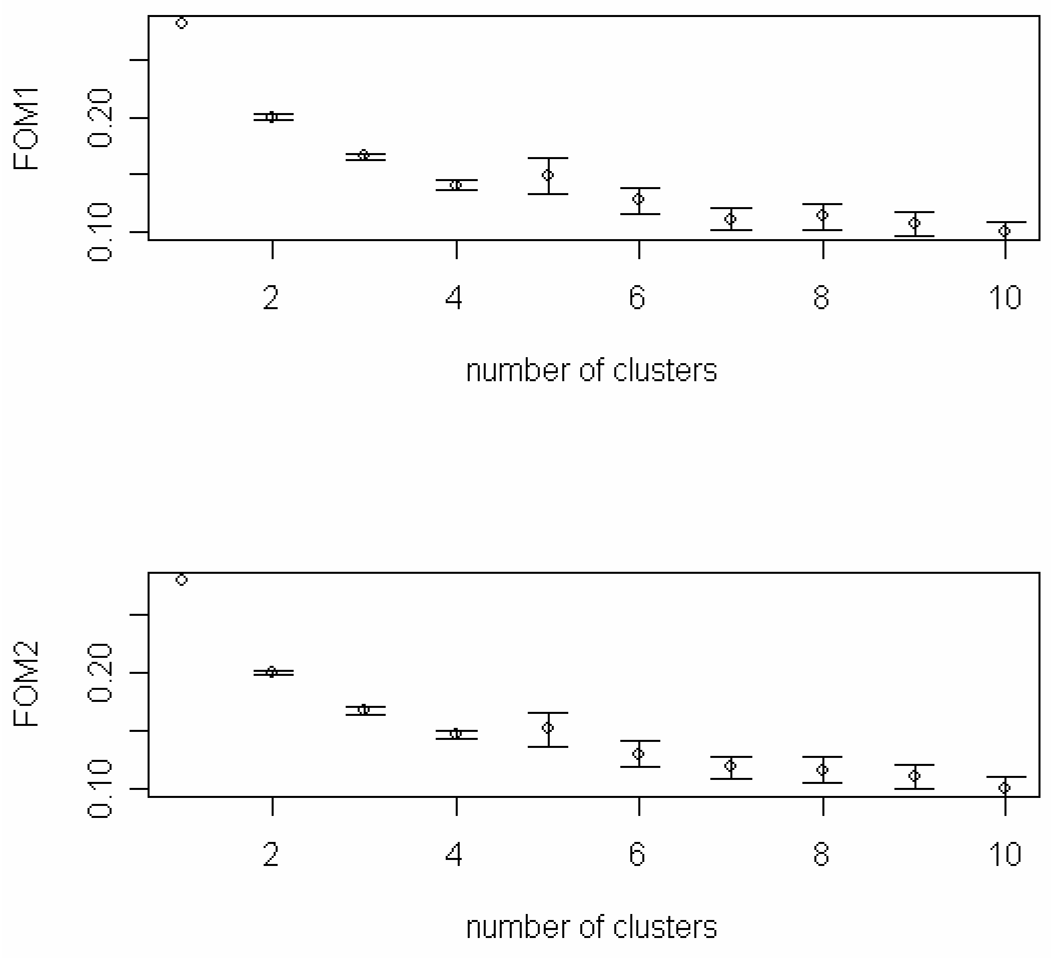
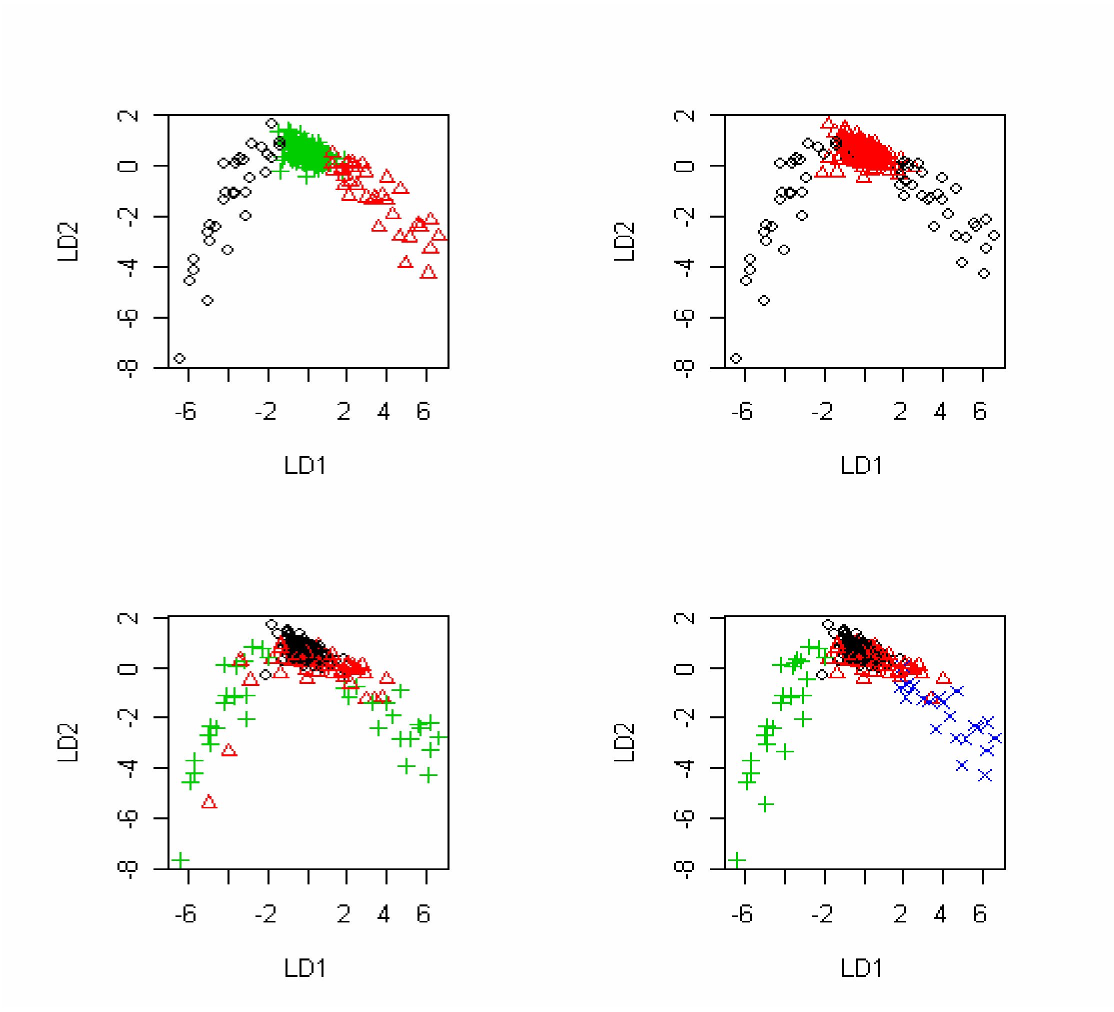


Table V. Results of the Thyroid dataset split to four clusters.

			Misclassifications		
K-means			26.61		
Tree-based techniques			Minimum terminal node size		
			5	**10**	**15**
AAMRT			27	27	27
Simple average co-occurrence matrix split with SBK	Number of terminal nodes	**3**	20	21	17
			18	31	29
		6	28	16	15
			41	16	16
		10	33	16	15
			22	16	16
Post processed co-occurrence matrix split with SBK	Number of terminal nodes	**3**	26	20	23
			16	20	20
		6	23	20	20
			20	19	20
		10	18	20	20
			19	19	20
Classification Tree			11	NP	NP

Figure 6 shows how SBK splits the weighted co-occurrence matrix into two (top right), three (bottom left) and four (bottom right) clusters. The top left graph shows the three known groups of the Thyroid dataset. All of the graphs have been projected onto the two-dimensional discriminant plane.

Figure 6. Clockwise from top left: three known groups of the Thyroid dataset projected onto the discriminant plane; two clusters found by SBK using criterion (1.6) on the weighted co-occurrence matrix; four clusters found by SBK using criterion (1.6) on the weighted co-occurrence matrix; three clusters found by SBK using criterion (1.6) on the weighted co-occurrence matrix. The individual AAMRTs were grown to six terminal nodes with a minimum terminal node size of 10.

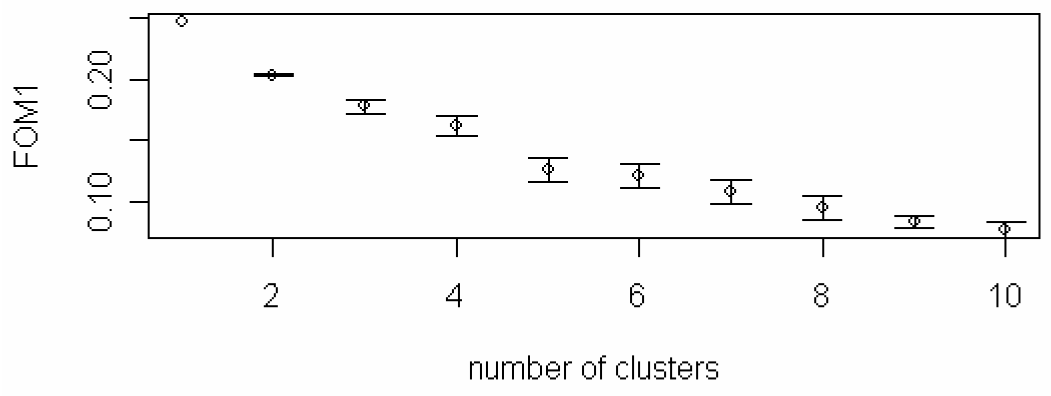


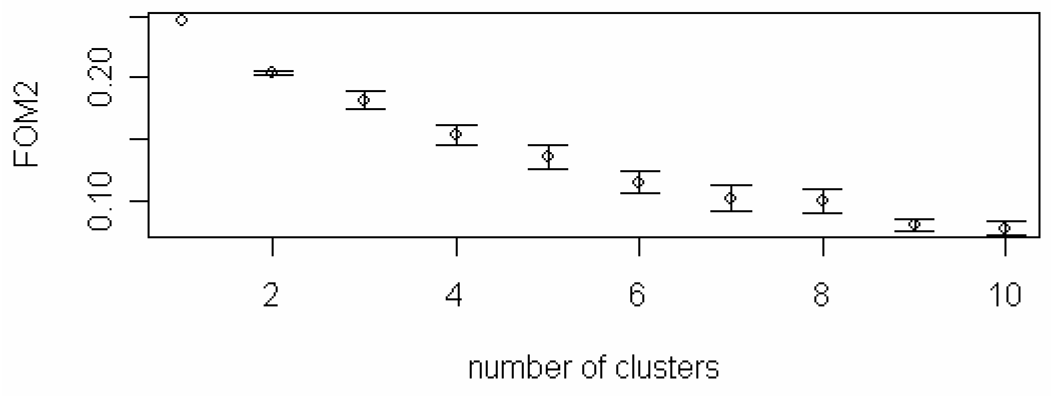
The results using the Vietnam dataset are reported in Table VI. The AAMRT and classification tree are grown to 6 nodes (the known number of groups in the data). The average and weighted co-occurrence matrices are split to six groups using SBK. The average co-occurrence matrices give superior results to the AAMRT and k-means, as long as the individual ensemble trees are grown as large, or larger than the known number of clusters. The weighted co-occurrence matrices show even better results with the misclassification rate on par with the classification tree. A sample AFOM graph is given in Figure 7. The reader will note that the AFOM graphs suggest splitting the weighted co-occurrence matrix to five or six clusters.

Table VI. Results of the Vietnam dataset split to six clusters.

			Misclassifications		
K-means			28.87		
Tree-based techniques			Minimum terminal node size		
			5	**10**	**15**
AAMRT			40	40	40
Simple average co-occurrence matrix split with SBK	Number of terminal nodes	**3**	54	54	54
			59	59	59
		6	17	14	19
			19	20	20
		10	12	16	15
			16	16	17
Post processed co-occurrence matrix split with SBK	Number of terminal nodes	**3**	19	19	19
			24	24	24
		6	13	13	13
			12	13	18
		10	19	7	13
			10	7	16
Classification Tree			14	14	14

Figure 7. AFOM graphs obtained by splitting the Vietnam dataset's weighted co-occurrence matrix using SBK to various numbers of clusters. The individual AAMRTs were grown to six terminal nodes with a minimum terminal node size of five. The splitting criterion for the top graph is given by (1.5) and the splitting criterion for the bottom graph is given by (1.6).





Overall, the weighted co-occurrence matrices split with SBK gave results that approached the misclassification rate of a classification tree. The simple average co-occurrence matrices performed better than the AAMRT and k-means, but not as well as the weighted matrices. This mirrors the results found in the regression settings where single models can be improved upon using simple average ensembles, and simple average ensembles can be further improved by post processing.

The weighted co-occurrence matrices produced excellent results as long as the size of the AAMRTs in the ensemble were as large, or larger than the known number of groups. The only exception occurred with the Iris dataset. Here, as the size of the individual trees became too large, the performance decreased. The decrease could be mitigated by choosing a large minimum terminal node size. Such results are understandable in the context of the datasets. The Iris dataset contains three clusters, all with 50 observational units. As the number of the terminal nodes increased with a small minimum terminal node size, the individual trees were shaving off subgroups slowly. If the minimum terminal node size was increased, this "shaving" was prevented and the true groups were found even with far more terminal nodes than groups. This dependence on minimum terminal node size was not obvious with the Thyroid and Vietnam datasets, because the groups are smaller and therefore setting the minimum terminal node size small encouraged the discovery of all the groups even with far more terminal nodes than groups.

As is often the case with cluster analysis, if the researcher was unsure about the structure of the groups within the dataset it would be advisable to create co-occurrence matrices for different values of the tree parameters. The AFOM graphs could then be viewed to determine the approximate number of clusters in the dataset, k_{approx}. The ensemble co-occurrence matrix could be recreated by growing the trees to k_{approx}, and then resplit using SBK. The above results show that splitting a weighted co-occurrence matrix created by trees grown to the "right" number of clusters produces excellent results irrespective of the minimum terminal node size.

The AFOM graphs provided a fairly accurate means of assessing the optimal number of clusters in the co-occurrence matrices. These optimal numbers tended to agree with known number of groups in the data, however if they differed, growing to the number suggested by the AFOM graphs gave superior results. The AFOM graphs have the advantage that they do not require a "gold standard" to assess the groups: the graphs show purely predictive performance.

Another means of visualizing the clustering solution to determine the number of clusters involves re-ordering the co-occurrence matrix according to areas of high similarity. This would give the researcher an overview of the structure of the co-occurrence matrix and how best to split the matrix.

The SBK algorithm gives a simple way of dividing a similarity matrix. The results here show that there was little difference between the two criteria (1.5) and (1.6). The sums of squares criterion may have slightly outperformed the absolute value criterion. As the ultimate aim of SBK is prediction, it may be wise to employ the more commonly used squared error loss. However, the numbers of a co-occurrence matrix are small and therefore there is no real difference between the criteria. Also, initial results (data not shown) indicate that SBK is relatively stable to the starting point. However, Figure 6 suggests that SBK may be biased towards finding "large", similar-sized clusters. When splitting the weighted co-occurrence matrix into two clusters, the algorithm uncovers the large dense group in the middle of the discriminant plot, but puts the other two smaller groups together. It is not until the data is divided into four clusters, that the two smaller groups are uncovered. This bias may explain the relatively poor performance of

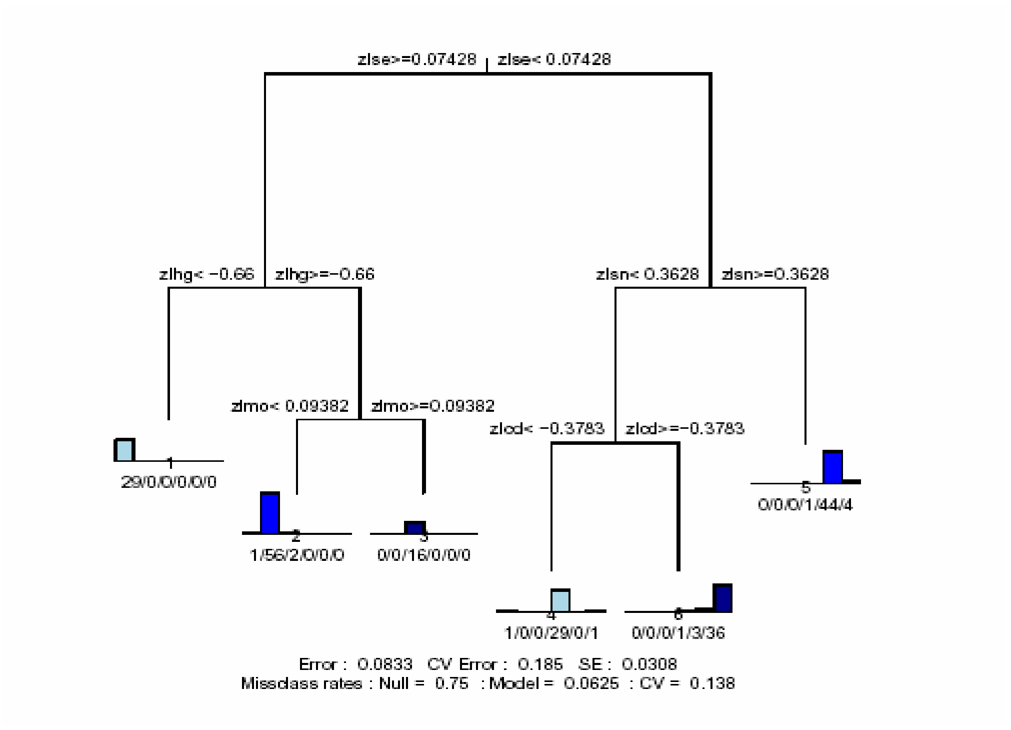
SBK on the Thyroid dataset. The bias would not have affected the Vietnam or Iris datasets because their group sizes are relatively similar.

To illustrate the variable importance lists produced, Table VII shows the weighted variable importance list obtained by the trees (with six terminal nodes, minimum terminal node size of 10) grown on the Vietnam dataset. Figure 8 shows the classification tree grown to six nodes using a minimum terminal node size of 10. There is a very high agreement between the variables with large importance and their position in the classification tree.

Table VII. Sample variable importance list for the Vietnam dataset.

VARIABLE	IMPORTANCE
zlse	0.163
zlhg	0.106
zlcd	0.097
zlsn	0.091
zlti	0.083
zlth	0.066
zlu	0.065
zlmo	0.054
zlpb	0.046
zlni	0.044
zlsr	0.040
zlcr	0.035
zlv	0.033
zlba	0.028
zlmn	0.022
zlas	0.017
zlcu	0.008

Figure 8. Classification tree of the Vietnam dataset with minimum terminal node size of 10.



5. CONCLUSIONS

This paper presented a strategy for post processing cluster ensembles. The strategy involved growing regression models (AAMRT) which could double as cluster models.

Trees were the ideal ensemble members because they produce associated variable importance lists and the predictive accuracy of a tree is well defined. The regression models were post processed and the most accurate models were retained. This process identified the most “predictable” clusters and implicitly post processed the cluster ensembles. The result was a weighted co-occurrence matrix. A new technique, SBK was suggested to divide the weighted co-occurrence matrix. Again, the emphasis of SBK was on predictability of the groups. Figure of merits were further extended to give an estimate of the natural number of groups in the data, and “post processed” variable importance lists were created. Overall, the results presented here agreed with results from the regression ensemble methodology: average ensembles produced more accurate results than a single model, and post processed ensembles produced more accurate models still.

The dependence on the ensemble’s parameters (minimum terminal node size and number of terminal nodes) was not extreme if the base trees were grown to at least the number of groups in the data. Growing individual trees past the number of groups in the data did not tend to adversely affect the performance of the ensemble, as long as the minimum terminal node size was set reasonably.

To make the procedure more amenable for use with large datasets, a partition of the datasets could be considered in the manner of Smyth et al. [27]. For example, the first subset would be used to train the individual trees, the second subset would be used to calculate the lasso weights, and the third subset could be used as a test subset. This would avoid overfitting the cluster ensemble. Alternatively, the individual trees could be cross validated from the outset, and only the cross validated predictions used. Also, future work could incorporate recent advances in the AAMRT methodology. This could involve using factor scores as response variables [28]. Factor scores are considered to be more robust in the large dataset setting.

REFERENCES

1. Satten, G. A.; Datta, S.; Moura, H.; Woolfitt, A. R.; Carvalho, M. d. G.; Carlone, G. M.; De, B. K.; Pavlopoulos, A.; Barr, J. R., Standardization and denoising algorithms for mass spectra to classify whole-organism bacterial specimens. *Bioinformatics* 2004; **20**(17):3128-3136.
2. Qu, Y.; Adam, B.-L.; Yasui, Y.; Ward, M. D.; Cazares, L. H.; Schellhammer, P. F.; Feng, Z.; Semmes, O. J.; Wright, G. L., Boosted Decision Tree Analysis of Surface-enhanced Laser Desorption/Ionization Mass Spectral Serum Profiles Discriminates Prostate Cancer from Noncancer Patients. *Clinical Chemistry* 2002; **48**(10):1835-1843.
3. Hancock, T.; Put, R.; Coomans, D.; Vander-Heyden, Y.; Everingham, Y., A performance comparison of modern statistical techniques for molecular descriptor selection and retention prediction in chromatographic QSRR studies. *Chemometrics and Intelligent Laboratory Systems* 2005; **76**(2):185-196.
4. Hastie, T.; Tibshirani, R.; Friedman, J., *The Elements of Statistical Learning*. Springer-Verlag: New York, 2001.
5. Friedman, J.; Popescu, B. *Importance Sampled Learning Ensembles*; Stanford University, Department of Statistics: 2003.
6. Dudoit, S.; Fridlyand, J., Bagging to improve the accuracy of a clustering procedure. *Bioinformatics* 2003; **19**(9):1090-1099.
7. Strehl, A.; Ghosh, J., Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 2002; **3**:583-617.
8. Greene, D.; Tsymbal, A.; Bolshakova, N.; Cunningham, P. In *Ensemble Clustering in Medical Diagnostics*, 17th IEEE Symposium on Computer-Based Medical Systems, Texas, 2004.
9. Ghosh, J.; Strehl, A.; Merugu, S. In *A consensus framework for integrating distributed clusterings under limited knowledge sharing*, NSF Workshop on Next Generation Data Mining, Baltimore, 2002.
10. Fern, X. Z.; Brodley, C. E. *Cluster ensembles for high dimensional data clustering: An empirical study*; CS06-30-02; Oregon State University, Department of Computer Science: 2006.
11. Questier, F.; Put, R.; Coomans, D.; Walczak, B.; Heyden, Y. V., The use of CART and multivariate regression trees for supervised and unsupervised feature selection. *Chemometrics and Intelligent Laboratory Systems* 2005; **76**(1):45-54.
12. Tibshirani, R., Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B-Statistical Methodology* 1996; **58**(1):267-288.
13. Grotkjaer, T.; Winther, O.; Regenberg, B.; Nielsen, J., Robust multi-scale clustering of large DNA microarray datasets with the consensus algorithm. *Bioinformatics* 2006; **22**(1):58-67.
14. Yeung, K. Y.; Haynor, D. R.; Ruzzo, W. L., Validating clustering for gene expression data. *Bioinformatics* 2001; **17**(4):309-318.

15. Tibshirani, R.; Walther, G.; Botstein, D.; Brown, P., Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics* 2005; **14**(3):511-528.
16. Smyth, C.; Coomans, D.; Everingham, Y.; Hancock, T., Auto-Associative Multivariate Regression Trees for Cluster Analysis. *Chemometrics and Intelligent Laboratory Systems* 2006; **80**(1):120-129.
17. Hancock, T. Multivariate Consensus Trees: Tree-based clustering and profiling for mixed data types. PhD, James Cook University, Townsville, 2006.
18. Breiman, L.; Friedman, J.; Olshen, R. A.; Stone, C. J., *Classification and Regression Trees*. Wadsworth: Belmont, CA, 1984.
19. Segal, M. R., Tree-Structured Methods for Longitudinal Data. *Journal of the American Statistical Association* 1992; **87**(418):407-418.
20. Kaufman, L.; Rousseeuw, P., *Finding groups in data: an introduction to cluster analysis*. Wiley: New York, 1990.
21. Monti, S.; Tamayo, P.; Mesirov, J.; Golub, T., Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning* 2003; **52**(1-2):91-118.
22. Fisher, R. A., The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 1936; **7**:179-188.
23. Nguyen, H. T.; Coomans, D.; Leermakers, M.; Boman, J. In *Multivariate statistical analysis of human exposure to trace elements from coal in Vietnam*, SPRUCE IV International Conference on Statistical Aspects of Health and the Environment, Enschede, the Netherlands, 1997.
24. Coomans, D.; Jonckheer, M.; Massart, D. L.; Broeckaert, I.; Block, P., The application of linear discriminant analysis in the diagnosis of thyroid diseases. *Analytica Chimica Acta* 1978; **103**(4):409-415.
25. R *A language and environment for statistical computing*, R Foundation for Statistical Computing: Vienna, Austria., 2004.
26. Therneau, T.; Atkinson, B. *mvpart: Multivariate partitioning*, 2004.
27. Smyth, C.; Coomans, D. In *Creating Parsimonious Ensembles*, 38th Symposium on the interface of statistics, computing science, and applications, Los Angeles, 2006.
28. Smyth, C.; Coomans, D.; Everingham, Y., Clustering noisy data in a reduced dimension space via multivariate regression trees. *Pattern Recognition* 2006; **39**(3):424-431.

Clustering Microarrays with Predictive Weighted Ensembles

Christine Smyth and Danny Coomans

Statistics and Intelligent Data Analysis Group,

School of Mathematics, Physics and Information Technology,

James Cook University,

Australia

Accepted

IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology

2007

Abstract-**Cluster ensembles seek a consensus across many individual partitions and the resulting solution is usually stable. Cluster ensembles are well suited to the analysis of DNA microarrays, where the tremendous size of the dataset can thwart the discovery of stable groups. Post processing cluster ensembles, where each individual partition is weighted according to its relative accuracy improves the performance of the ensemble whilst maintaining its stability. However, weighted cluster ensembles remain relatively unexplored, primarily because there are no common means of assessing the accuracy of individual clustering solutions. This paper describes a technique of creating weighted cluster ensembles suitable for use with microarray datasets. A regression technique is used to obtain individual cluster solutions. Each solution is then weighted according to its predictive accuracy. The consensus partition is obtained using a novel modification to the traditional k-means algorithm which further enforces the predictability of the solution. An estimate of the natural number of clusters can also be obtained using the modified k-means algorithm. Furthermore, a valuable byproduct of this weighted ensemble approach is a variable importance list. The methodology is applied on two well-known microarray datasets with promising results.**

I. INTRODUCTION

Cluster analysis plays a vital role in the understanding of large DNA microarrays. However, the large number of variables in these datasets can cloud the underlying groups, and traditional clustering algorithms may produce inaccurate or unstable results. This motivates the application of cluster ensembles to DNA microarrays. Cluster ensembles seek a consensus across many individual clustering solutions, often grown on smaller subsets of the data, with the aim of finding a stable partition.

Cluster ensembles combine individual solutions in various ways. A common approach involves the creation of a co-occurrence matrix for each clustering solution. Basically, the $(i,j)^{th}$ element of the co-occurrence matrix equals one if observational units i and j are clustered together by the algorithm and zero otherwise. The co-occurrence matrices of each model within the ensemble are aggregated to give an overall co-occurrence matrix, where the $(i,j)^{th}$ element represents the percentage of times observational units i and j are clustered together. The overall matrix is a similarity matrix and can be split using a variety of clustering techniques, such as hierarchical clustering or partitioning around medoids [1].

However, within a cluster ensemble there will be both "good" and "bad" partitions [2]. Assigning low weights to inaccurate co-occurrence matrices, and then taking a weighted aggregation of the individual co-occurrence matrices should improve the performance of the cluster ensemble. However, weighting (post processing) individual clustering solutions within an ensemble remains relatively unexplored. Unlike regression and classification ensembles, where the accuracy of individual models can easily be gauged using a loss criterion between the predicted values and the observed response, there are no criteria suitable for assessing individual clustering solutions within an ensemble.

Previously, we suggested a technique of weighting cluster ensembles for small datasets [3]. The accuracy of each clustering solution was assessed on the basis of its predictive error. The weighted cluster ensembles outperformed simple average ensembles and individual clustering models. Here, we apply the technique with some modifications for large datasets to DNA microarrays.

We propose that by using a regression technique, multivariate regression trees, as a clustering algorithm, each solution can be assessed according to its predictive accuracy.

Previous literature has shown that multivariate regression trees double effectively as a clustering technique [4],[5]. If clustering in a low dimensional setting, the explanatory variables are replicated as the response variables (auto-associative multivariate regression tree), and the clusters are found in the entire variable space. If clustering in a high dimensional setting, the dimension is first reduced using principal components analysis or factor analysis, and the resulting scores are used as the response variables [6]. The response set can be made as small as desired by taking the first q principal components or factors. Searching in the reduced dimension space for clusters is particularly appealing when analyzing DNA microarrays where some variables serve only to distort the underlying grouping structure.

By sampling explanatory variables, many trees can be grown. Trees with high predictive accuracy are then given large weights. The weighting procedure can easily be performed using any well-known regression post processing technique. Here we use the forward stagewise approximation [7] to the lasso [8].

By taking the co-occurrence matrix given by each tree and multiplying it by the tree's weight, and then summing the weighted co-occurrence matrices together, an overall "weighted co-occurrence matrix" is obtained. This co-occurrence matrix can be considered the output of a weighted cluster ensemble approach. The approach assumes that trees with high predictive accuracy produce "good" clusters. Using predictive accuracy to assess cluster quality has previously been suggested [9],[10],[11].

To partition the weighted co-occurrence matrix we introduce similarity-based k-means (SBK) [3]. SBK enforces the predictability of the solution by explicitly predicting the group structure found within the entire similarity matrix (including the covariance submatrices) shown to be important in [12]. An approximation to the natural number of clusters in the dataset can also be obtained with SBK using a technique modeled on [10].

Furthermore, the underpinning weighted ensemble produces a list of variables (genes) that are important in differentiating the clusters. A variable importance list gives experimentalists an idea of genes that may warrant further investigation as potential biomarkers particularly if the genes are differentiating between two groups (say cancer versus non-cancer).

We illustrate the weighted cluster ensemble approach on two well-known DNA microarray datasets. The clustering results are consistent with others in the literature. Some genes in the derived variable importance lists are known to be important in classifying the groups within the datasets. The estimates of the natural number of clusters tend to agree with the known number of classes in the data.

II. THEORY

A. Multivariate Regression Trees

Regression trees [13],[14] begin with all the data in one node. At each stage, the regression tree partitions a non-split node in two. Regression trees partition a node, t, into two subsets, t_L and t_R, on the basis of the value of an explanatory variable. At each node all possible splits of each explanatory variable are considered. The optimal split is saved for each node. The node with the split that maximizes the decrease in $R(T)$ is partitioned at each stage. $R(T)$ is given by:

$$R(T) = \frac{1}{n} \sum_{t \in \tilde{T}} \sum_{\underset{\sim}{x}_i \in t} \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t) \right)^T \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t) \right) \tag{1}$$

where $\underset{\sim}{x}_i$ is the vector of measurements of P explanatory variables for the i^{th} observational unit; $\underset{\sim}{y}_i$ is the vector of measurements of the response variables for the i^{th}

observational unit; $\tilde{T}$ is the set of all terminal nodes and; $\underset{\sim}{\bar{y}}(t)$ is the mean response vector of terminal node t.

After growing the tree, the non-split nodes are deemed "terminal". The predicted value for a terminal node, t_{term}, is:

$$\underset{\sim}{\hat{y}}(t_{term}) = \frac{1}{n_{t_{term}}} \sum_{\underset{\sim}{x}_i \in t_{term}} \underset{\sim}{y}_i \tag{2}$$

where the sum is over all $\underset{\sim}{y}_i$ such that $\underset{\sim}{x}_i \in t_{term}$ and $n_{t_{term}}$ is the total number of cases in the terminal node.

The observational units in each of the terminal nodes are the clusters of the dataset: the terminal nodes are as homogeneous as possible reflecting an intuitive definition of a cluster. The clusters are found in the response space and the explanatory variables that form the tree are deemed to be important in determining the clusters. To allow multivariate regression trees to be applied in the traditional clustering framework where there are no response variables, auto-associative multivariate regression trees (AAMRTs) were suggested [4],[5]. AAMRTs replicate the explanatory variables as response variables and grow the tree using identical response and explanatory datasets.

If the number of the variables is too large, AAMRTs may be confused by the redundant or 'noise' variables and may produce inaccurate results. To overcome this flaw, the dimension of the response space can be reduced using either principal components or factor analysis. Principal components analysis attempts to model the total variance of the original dataset, via new uncorrelated variables called principal components. Factor analysis attempts to explain the variables by assuming that they can be generated as a linear combination of q unobservable common factors (usually $q << P$) plus a unique factor [15]. We use either the principal component scores from the first q principal components or the factor scores from the q factors as the response variables of the tree [6]. The clustering is obtained in the reduced dimension space as q is less than P. Trees grown using principal component scores are referred to as MRTPCs, and similarly, trees grown using factor scores are referred to as MRTFSs.

B. Algorithm for creating ensembles of AAMRTs, MRTPCs, or MRTFSs

Algorithm 1 shows the process used to grow an ensemble of regression trees such that they can be used to create a cluster ensemble.

Algorithm 1: Growing an ensemble of trees

1) Choose the number of individual trees in the ensemble, M.

2) If the trees in the ensemble are AAMRTs, replicate the original dataset as the response dataset, Y. If the trees in the ensemble are MRTPCs calculate the first q principal component scores as the response dataset, Y. Or, if the trees in the ensemble are MRTFSs, calculate the first q factor scores as the response dataset, Y. The choice of q is left to the investigator.

3) Create M explanatory datasets by randomly sampling variables with percentage p_v from the original dataset. Here we use $p_v = 0.05$. We stress that although the variables may be sampled to create different explanatory datasets, the response for each tree remains constant.

4) Grow a tree using each explanatory dataset to k terminal nodes (clusters). Create a co-occurrence matrix for each tree, $C(m)$, where the $(i, j)^{th}$ element of the matrix is one, if observational units i and j are in the same cluster (terminal node). Create a variable importance list for each tree using Algorithm 5.

C. Lasso Heuristic

A regression ensemble can be represented by:

$$F(\underset{\sim}{x}) = \sum_{m=1}^{M} \omega_m f_m(\underset{\sim}{x}) \tag{3}$$

where $f_m(\underset{\sim}{x})$ is the prediction of an observational unit $\underset{\sim}{x}$ by the m^{th} model - the $f_m(\underset{\sim}{x})$ are usually of the same family of models but this is not mandatory; ω_m is the weight assigned to $f_m(\underset{\sim}{x})$; and M is the number of models.

The individual regression solutions are combined to form an ensemble by taking a weighted sum of the individual solutions. Usually, the weights are an average of the number of models, $1/M$, see for example [7]. Post processing is a procedure which suggests choices of ω_m that reflect the relevance of each $f_m(\underset{\sim}{x})$ [16]. Post processing usually achieves greater accuracy by enforcing parsimony. The lasso [8] post processing procedure finds the weights that minimize

$$\hat{\underset{\sim}{\omega}} = \arg\min_{\underset{\sim}{\omega}} \sum_{i=1}^{n} \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m f_m(\underset{\sim}{x}_i) \right)^T \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m f_m(\underset{\sim}{x}_i) \right) + \lambda \sum_{m=1}^{M} |\omega_m|. \tag{4}$$

Here, the solution to the lasso is approximated with a forward stagewise algorithm [7] which is henceforth referred to as the "lasso heuristic". The algorithm is as follows:

Algorithm 2: The lasso heuristic

1) Set all weights to zero. Choose ε as a small number greater than zero, and choose the number of iterations, its, to be quite large.

2) For $l = 1 : its$

$$\left(\beta^*, h^*\right) = \arg\min_{\beta,h} \sum_{i=1}^{n} \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m f_m(\underset{\sim}{x}_i) - \beta \times f_h(\underset{\sim}{x}_i) \right)^T \left(\underset{\sim}{y}_i - \sum_{m=1}^{M} \omega_m f_m(\underset{\sim}{x}_i) - \beta \times f_h(\underset{\sim}{x}_i) \right). \tag{5}$$

$$\hat{\omega}_{h^*} = \hat{\omega}_{h^*} + \varepsilon \times sign(\beta^*). \tag{6}$$

3) Finally,

$$F(\underset{\sim}{x}) = \sum_{m=1}^{M} \hat{\omega}_m f_m(\underset{\sim}{x}). \tag{7}$$

In the first step all weights are zero, and this is analogous to $\lambda = \infty$ in (4). The parameter *its* is inversely related to λ in (4). After the set number of iterations, many weights will still remain zero.

D. Algorithm for producing a weighted co-occurrence matrix

Algorithm 3 shows the process used to create a weighted co-occurrence matrix.

Algorithm 3: Producing a weighted co-occurrence matrix

1) Create an ensemble of trees using Algorithm 1.

2) Post process the ensemble of regression trees to find the weights using Algorithm 2. Here, $f_m(\underset{\sim}{x}_i)$ is the prediction of observational unit i using the m^{th} regression tree. The response vector, $\underset{\sim}{y}_i$ is given by: $\underset{\sim}{x}_i$ if using AAMRTs or; the associated vector of q principal component scores if using MRTPCs or; the associated vector of q factor scores if using MRTFSs.

3) Create an overall co-occurrence matrix, C by taking a weighted sum of the individual co-occurrence matrices:

$$C = \sum_{m=1}^{M} \hat{\omega}_m C(m). \tag{8}$$

Taking a weighted sum of dissimilarity matrices created from different sources (where the weights were chosen in a “subjective way”) was suggested previously by [17].

E. Similarity-based k-means

Similarity-based k-means is a divisive clustering algorithm that takes a co-occurrence matrix, C, (similarity matrix) as input. Formally, SBK seeks clusters to minimize either of the objective functions:

$$\min \sum_{r=1}^{k} \sum_{i,j \in S_r} \left(C_{i,j} - \bar{C}_r \right)^2 + \sum_{r=1}^{k} \sum_{\substack{r' \neq r \\ r'=1}}^{k} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left(C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right)^2 \tag{9}$$

or

$$\min \sum_{r=1}^{k} \sum_{i,j \in S_r} \left| C_{i,j} - \bar{C}_r \right| + \sum_{r=1}^{k} \sum_{\substack{r' \neq r \\ r'=1}}^{k} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left| C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right| \tag{10}$$

where k is the number of clusters; i, j index observational units i and j; S_r is the set of observational units in the r^{th} cluster; $C_{i,j}$ is the $(i, j)^{th}$ element of the co-occurrence matrix; $\bar{C}_r$ is the mean similarity of the r^{th} cluster; and $\overline{COV}_{(S_r, S_{r'})}$ is the mean similarity of the (covariance) matrix where the rows are given by the observational units in cluster r and the columns are dictated by the observational units in the r'^{th} cluster. The covariance submatrices should be considered the number of times that observational units in one cluster are grouped with observational units in another cluster during the ensemble creation.

Because of the mean squared and absolute error terms in the objective functions (9) and (10), SBK can be viewed almost entirely in the prediction sense. The algorithm seeks to predict the entire co-occurrence matrix using the cluster and covariance means. In doing so, observational units with high similarity are grouped together. A validity criterion [12] is imposed to ensure that the clustering ideology prevails over the prediction ideology. The validity criterion dictates that clusters must have higher mean similarities than their covariance matrices. The SBK algorithm is given by Algorithm 4.

Algorithm 4: SBK

1) Choose the number of clusters and an initial partition of the data. Here, we use initial partitions given by both hierarchical clustering of the co-occurrence matrix and entirely random partitions. Choose the objective function; either the mean squared error (9) or absolute error (10).

2) Visit each observational unit and assign it to the cluster which will result in the largest decrease of the objective function. Before moving the observational unit ensure that the validity criterion is upheld.

3) Update the mean similarity of: the cluster the observational unit has left; the cluster the observational unit has joined; and all appropriate covariance means.

4) Repeat steps two and three until no more reassignments of the observational units take place.

F. Cluster number estimation

An approximation to the natural number of clusters in the dataset can also be obtained with SBK by considering the average predictive capability of the algorithm, for any number of clusters, k. The estimate closely resembles the figure of merit (FOM) method proposed by [10]. FOMs are a method of authenticating clusters by assessing the "predictive power" of a clustering technique. FOMs require no a priori knowledge of group membership. FOMs have been shown to provide an accurate estimate of the natural number of clusters [5],[10]. FOMs assess the "predictive power" of a clustering algorithm by leaving out a variable p, clustering the data (into k clusters), then calculating the root mean square error (RMSE) of p relative to the cluster means:

$$RMSE(p,k) = \sqrt{\frac{1}{n}\sum_{r=1}^{k}\sum_{\underset{\sim}{x}_i \in S_r}\left(x_{ip} - \overline{x}_r(p)\right)^2} \tag{11}$$

where x_{ip} is the measurement of the p^{th} variable on the i^{th} observational unit; n is the number of observational units; S_r is the set of observational units in the r^{th} cluster; $\bar{x}_r(p)$ is the mean of variable p for the observational units in the r^{th} cluster.

Each variable is omitted and its RMSE calculated. These RMSEs are summed over all variables to give an aggregate FOM (AFOM):

$$AFOM(k) = \sum_{p=1}^{P} RMSE(p,k). \tag{12}$$

The AFOM is calculated for each k, and adjusted for cluster size to give $AFOM_{adj}(k)$.

It is simple to expand the above AFOM theory to the results obtained by SBK. Here, no variables are removed from the dataset; the random nature of SBK introduces enough variability. Simply, if the dataset is clustered into k clusters and this process is repeated P times, then the $AFOM(k)$ is defined as

$$AFOM(k) = \sum_{p=1}^{P} \sqrt{\frac{1}{n^2} \sum_{r=1}^{k} \sum_{i,j \in S_r(p)} \left(C_{i,j} - \bar{C}_r(p) \right)^2} \tag{13}$$

where $S_r(p)$ is the set of observational units in cluster r on the p^{th} run; $\bar{C}_r(p)$ is the mean similarity of the observational units in cluster r on the p^{th} run; $C_{i,j}$ is the $(i,j)^{th}$ element of the co-occurrence matrix; and n^2 is the dimension of the similarity matrix. Here, the adjusted figure of merit is given by:

$$AFOM_{adj}(k) = \frac{AFOM(k)}{P\sqrt{\frac{n^2-k}{n^2}}}. \tag{14}$$

The $AFOM_{adj}$ is obtained for varying levels of k and the “elbow” of the graph is selected as the optimal number of clusters.

G. Variable Importance

Multivariate regression trees allow for an easy calculation of a variable importance list. Although many definitions of variable importance exist, such as those that consider surrogate splits [13], we apply a very simple (but naive) definition of variable importance. Our definition of variable importance, if applied to only one tree grown on the entire dataset would over-inflate the importance of some variables and underestimate the importance of others. However, our reasoning is that the random sampling of variables to build each tree will give some stability to our variable importance list that would otherwise not exist. We calculate a variable importance list for each tree in the ensemble using Algorithm 5. The variable importance list for the entire ensemble is then the weighted sum of the variable importance lists for each tree, using the weights given by the lasso heuristic of Algorithm 2.

Algorithm 5: Variable importance list for a single tree

1) For each variable, p sum the $\Delta R(t)$ for all splits made by that variable within the tree to obtain the variable importance of p ,VI_p. Mathematically, VI_p is given by:

$$VI_p = \sum_{\substack{t \in \tilde{T} \text{ where} \\ \text{the node is} \\ \text{split by } p}} \Delta R(t) \tag{15}$$

where

$$R(t) = \sum_{x_i \in t} \left(\underset{\sim}{y}_i - \underset{\sim}{\bar{y}}(t) \right)^T \left(\underset{\sim}{y}_i - \underset{\sim}{\bar{y}}(t) \right) \tag{16}$$

and

$$\Delta R(t) = R(t) - (R(t_L) + R(t_R)) \tag{17}$$

and t designates the parent node; and t_L and t_R designate the left and right nodes respectively.

If a variable is not included in the predictor set of a particular tree, its corresponding variable importance for the tree is zero.

2) Standardize the variable importance for the tree such that the individual importances sum to one.

H. Cluster Evaluation

Assessing the validity and accuracy of clustering algorithms is not a straightforward task. Various algorithms have been suggested in the recent literature, see for example [10] and [18]. However, in this paper we use datasets with known classifications and we assume these to be the gold standard. As such, we report only the number of "misclassifications" (similar to [9] and [19]), but recognize that in real world settings this is not possible.

III. DATA

Two well known microarray datasets were analyzed. The reader is directed to the references for detailed information regarding these datasets. The first dataset, 'Alon' [20], contains 62 samples measured on 2000 genes. There are 22 samples of normal colon tissue, and 40 samples of tumor tissue. The 100 variables with the largest variance were used in this analysis. The dataset, available from the R package 'dprep' [21], was preprocessed by taking the logarithm (base 10), and standardizing the tissues and genes to have zero mean and unit standard deviation.

The second dataset, 'Golub' [22], contains 72 samples measured on 6817 genes. The number of genes was decreased to 3571 using the steps of [23]. There are 47 samples of Acute Lymphoblastic Leukemia (ALL) and 25 samples of Acute Myeloid Leukemia (AML). The ALL class can be further divided into two subgroups consisting of 38 B-cell ALL and 9 T-cell ALL. The 100 variables with the largest variance were used in this analysis in the same manner as [23]. The dataset, (which has already been log transformed and row standardized) available from the R package ‘dprep’, was preprocessed by standardizing the genes to have unit standard deviation.

IV. PROCEDURE

The individual tree models in the ensemble require the specification of the number of terminal nodes and the minimum terminal node size. To assess the sensitivity of the results to varying values of these parameters, we ran Algorithm 1 with either (1,5,10) minimum terminal node size and either (2,4,6) terminal nodes. There were 3*3=9 choices of parameters and an ensemble of trees was grown for each choice. We also grew an ensemble with random inputs to the parameters. Each tree within the ensemble was randomly assigned a minimum terminal node size and number of terminal nodes from the above sets. This resulted in a total of 10 ensembles being grown for each of AAMRTs, MRTFSs, and MRTPCs. There were therefore 30 ensembles created for each dataset. All ensembles were grown to 500 trees. The parameter q was taken to be 10.

The M co-occurrence matrices for a set of parameters and response type were weighted using Algorithms 2 and 3. The weighted co-occurrence matrices were then split using SBK. When splitting co-occurrence matrices the minimization criteria (9) and (10) of SBK were used and both results are shown. The results reported were the most frequently occurring using 15 different starting points. The datasets were split to a maximum of 10 clusters so that the AFOM graphs could be obtained. However, the reported results are those when the co-occurrence matrix was split to the known number of groups in the data. Variable importance lists were also obtained. All analysis was conducted using [24].

V. RESULTS

A. Alon Dataset

The results of applying SBK to the weighted co-occurrence matrices created by each of the ensemble types are reported in Table I. The first row shows the number of terminal nodes of the trees in the ensemble. The second row shows the minimum terminal node size of the trees in the ensemble. The 'R' in both the first and second rows corresponds to the ensembles of trees grown on random parameter (minimum terminal node size and number of terminal nodes) values. The types of trees in the ensemble are shown in the final three rows. The reader will see that there are ten ensembles grown for each response type. The number of misclassifications using SBK with (9) is shown as the top number of the cell, and the number of misclassifications using (10) is shown as the bottom number of the cell.

The results of applying SBK to the co-occurrence matrices created by the ensembles of AAMRTs and MRTPCs are fairly consistent across minimum terminal node size and number of terminal nodes. The misclassification rates of SBK applied to the co-occurrence matrices created by the AAMRT and MRTPC ensembles grown with random parameters are a fair compromise of the misclassifications using set parameters. The misclassification rates of applying SBK to the co-occurrence matrices created by ensembles of MRTFSs are less stable than the other two ensemble types.

The AFOM graphs tend to indicate that the weighted co-occurence matrices should be split to three clusters. A sample AFOM graph (with error bars) is shown in Fig. 1. It was obtained by applying SBK with (9) to a weighted co-occurrence matrix constructed by MRTPCs with random parameters. Growing to three clusters improves the results considerably as shown in Table II. There is a high degree of similarity between the misclassification rates of applying SBK to the co-occurrence matrices of the ensembles of AAMRTs and MRTPCs. Growing these ensembles with random parameters gives a compromise of the misclassifications using the set parameters. On the other hand, using random parameters with ensembles of MRTFSs does not give solutions that are representative of ensembles with set parameters.

The top five important variables using each response type are presented in Table III. The variables are presented in decreasing order of importance. There is a degree of overlap between the response types, particularly using ensembles of AAMRTs and MRTPCs.

TABLE I

NUMBER OF MISCLASSIFICATIONS FOR THE ALON DATASET – TWO CLUSTERS

Number of terminal nodes	2			4			6			R
Minimum terminal node size	1	5	10	1	5	10	1	5	10	R
AAMRT	9	9	13	15	15	7	14	15	7	13
	14	14	14	15	15	10	15	15	10	13
MRTPC	14	14	9	15	16	7	16	6	7	13
	13	13	13	15	15	10	15	13	10	12
MRTFS	22	22	22	7	22	10	9	6	10	12
	22	22	22	8	13	9	13	6	9	10

TABLE II

NUMBER OF MISCLASSIFICATIONS FOR THE ALON DATASET – THREE CLUSTERS

Number of terminal nodes	2			4			6			R
Minimum terminal node size	1	5	10	1	5	10	1	5	10	R
AAMRT	10	10	10	9	9	6	7	7	6	8
	10	10	10	10	10	10	8	11	10	9
MRTPC	10	10	10	9	9	7	7	6	7	8
	10	10	10	9	9	10	8	8	10	8
MRTFS	8	7	10	7	6	7	6	6	7	13
	8	7	8	9	9	10	6	6	10	12

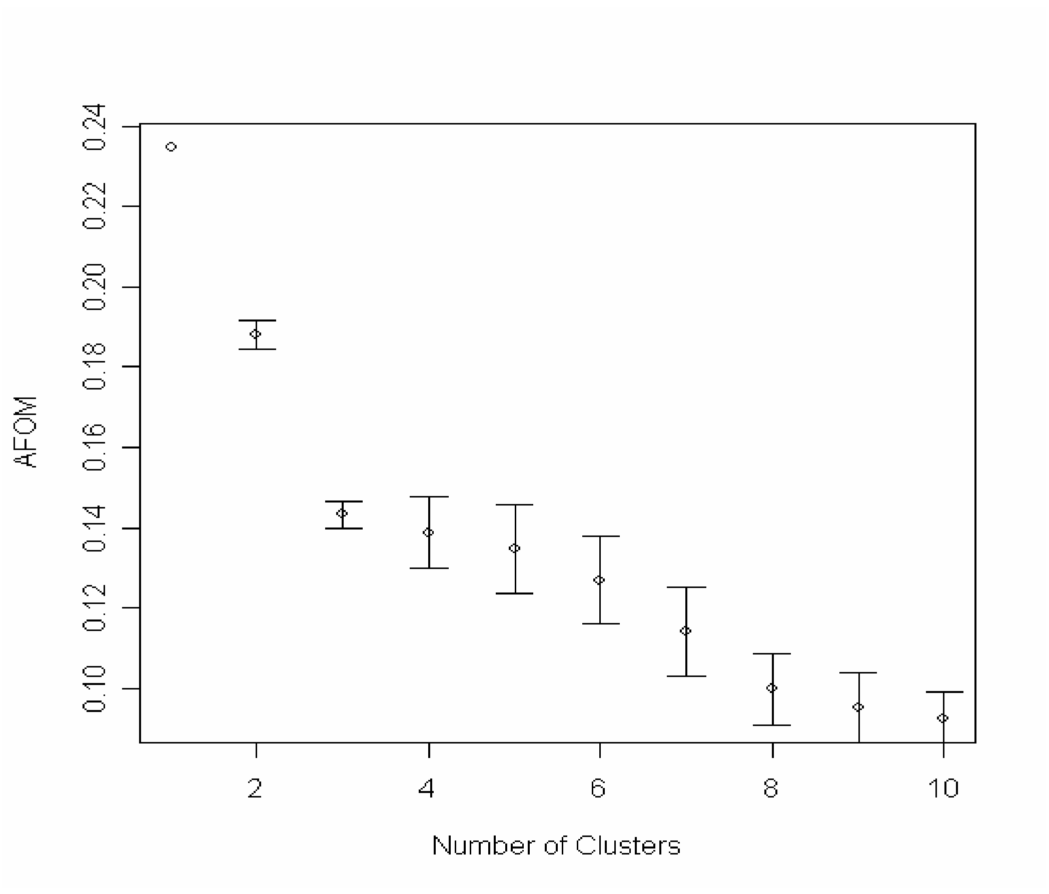


Fig. 1. AFOM graph for the Alon dataset.

TABLE III

IMPORTANT VARIABLES FOR THE ALON DATASET

Ensemble	Gene description
AAMRT	Human 11 beta-hydroxysteroid dehydrogenase type II mRNA, complete cds
	ACTIN, AORTIC SMOOTH MUSCLE (HUMAN)
	H. sapiens mRNA for hevin like protein
	P24050 40S RIBOSOMAL PROTEIN
	Human mRNA for fibronectin (FN precursor)
MRTPC	Human 11 beta-hydroxysteroid dehydrogenase type II mRNA, complete cds
	ACTIN, AORTIC SMOOTH MUSCLE (HUMAN)
	H. sapiens mRNA for hevin like protein
	PUTATIVE 126.9 KD TRANSCRIPTIONAL REGULATORY PROTEIN IN YSW1-RIB7 INTERGENIC REGION (Saccharomyces cerevisiae)
	TRANSLATIONAL INITIATION FACTOR 2 BETA SUBUNIT (HUMAN)
MRTFS	P24050 40S RIBOSOMAL PROTEIN
	Human CO-029
	Human 11 beta-hydroxysteroid dehydrogenase type II mRNA, complete cds
	H. sapiens mRNA for novel DNA binding protein
	SELENIUM-BINDING PROTEIN (Mus musculus)

B. Golub Dataset

The results of splitting the weighted co-occurrence matrices created by each of the tree types are shown in Table IV. The misclassification rates using SBK on co-occurrence matrices created by ensembles of AAMRTs and MRTPCs are similar. Using these two types of trees with random parameters also gives misclassification rates that are representative of the set parameters. Again, SBK applied to the co-occurrence matrices created by MRTFSs does not produce as stable results as with the other two types of trees.

The AFOM graphs indicate splitting to three clusters will produce the optimal results. A sample AFOM graph is shown in Fig. 2. The graph was obtained by applying SBK with (9) to the weighted co-occurrence matrix constructed by AAMRTs with a minimum terminal node size of five and two terminal nodes.

Splitting the weighted co-occurrence matrices uncovers the three known subgroups in the data. The misclassification rates are shown in Table V. The table may indicate that if the minimum terminal node size of the trees is too large, the misclassification rates of SBK suffer. Again, splitting the co-occurrence matrices created by ensembles of AAMRTs and MRTPCs produces similar, stable results. However, splitting the co-occurrence matrices of ensembles of MRTFSs using SBK produces unstable results across set tree parameters. Also, the results are not indicative of the set parameters when the trees use random parameters.

The top five variables using each tree type are shown in Table VI. Again, there is a large degree of overlap amongst the ensembles of AAMRTs and MRTPCs.

TABLE IV

NUMBER OF MISCLASSIFICATIONS FOR THE GOLUB DATASET – TWO CLUSTERS

Number of terminal nodes	2			4			6			R
Minimum terminal node size	1	5	10	1	5	10	1	5	10	R
AAMRT	9	9	9	10	10	2	10	10	4	9
	9	9	9	10	10	9	10	9	9	9
MRTPC	9	9	9	10	10	10	10	10	4	9
	9	9	9	10	9	10	10	10	10	9
MRTFS	13	13	17	7	7	11	10	12	12	10
	13	13	17	19	19	9	10	11	5	10

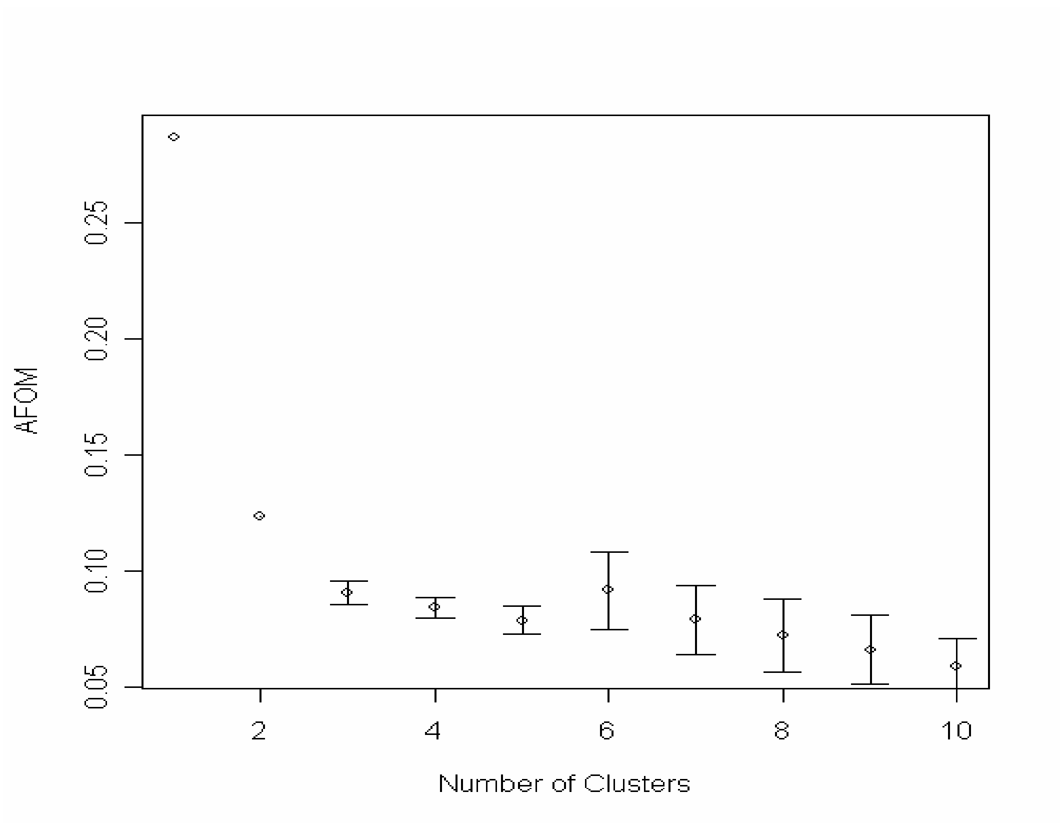


Fig. 2. AFOM graph for the Golub dataset.

TABLE V

NUMBER OF MISCLASSIFICATIONS FOR THE GOLUB DATASET – THREE CLUSTERS

Number of terminal nodes	2			4			6			R
Minimum terminal node size	1	5	10	1	5	10	1	5	10	R
AAMRT	7	7	6	3	3	3	7	9	8	7
	3	3	6	5	3	3	8	8	8	6
MRTPC	3	3	6	3	3	3	3	3	7	8
	6	6	6	5	3	5	3	3	9	8
MRTFS	7	7	18	5	5	17	3	5	16	3
	7	7	18	5	13	4	16	4	13	2

TABLE VI

IMPORTANT VARIABLES FOR THE GOLUB DATASET

Ensemble	Gene description
AAMRT	MB-1 gene
	LGALS1 Ubiquinol-cytochrome c reductase core protein II.
	PROBABLE PROTEIN DISULFIDE ISOMERASE ER-60 PRECURSOR
	DFD component of complement (adipsin)
	Zyxin
MRTPC	MB-1 gene
	Zyxin
	PROBABLE PROTEIN DISULFIDE ISOMERASE ER-60 PRECURSOR
	GLUTATHIONE S-TRANSFERASE, MICROSOMAL
	DFD component of complement (adipsin)
MRTFS	Growth factor receptor tyrosine kinase (STK-1) mRNA
	AFFX-M27830_5_at (endogenous control)
	GLYCOPHORIN B PRECURSOR
	CAPG Capping Protein (actin filament) gelsolin-like
	MDK Midkine (neurite growth-promoting factor 2)

VI. DISCUSSION

Firstly, we noticed no major trends with minimum terminal node size and number of terminal nodes. There may have been a very small effect of minimum terminal node size on the misclassification rate of SBK (three clusters) applied to the co-occurrence matrices of the Golub dataset. Because the smallest subgroup contains only nine observational units, if the minimum terminal node size was set too high (i.e. 10), this group could not be recovered perfectly.

The two criteria of SBK did not produce remarkably different results: criterion (9) could be performing slightly better than criterion (10). As the ultimate aim of SBK is prediction, it may be wise to employ the more commonly used squared error loss criterion.

Splitting the co-occurrence matrices of ensembles of AAMRTs and MRTPCs with SBK produced similar misclassification rates. The variable importance lists of these two ensembles were also alike. This indicates that the actual AAMRTs and MRTPCs were similar. The similarity between the results of AAMRTs and MRTPCs has been noted elsewhere [6]. The misclassifications using these two tree types were fairly stable. Furthermore, using random parameters gave a compromise misclassification rate of the ensembles grown using set parameters.

On the other hand, ensembles of MRTFSs, although capable of creating optimal solutions, tended to be fairly unstable and without any discernable pattern across minimum terminal node size and number of terminal nodes. A representative solution was not found by using random parameters.

The poor results obtained using MRTFSs were surprising. In a previous study these trees have been shown to outperform AAMRTs and MRTPCs on datasets perturbed by noise variables [6].

In the previous study the results of MRTPCs were generally stable to the number of factors. Here, we see that the stability of MRTPCs also extends to other parameters: the number of terminal nodes and minimum terminal node size.

The AFOM graphs indicated that there were three clusters within each dataset. The results were improved when the co-occurrence matrices of the Alon dataset were split to three clusters. Splitting the co-occurrence matrices of the Golub dataset unearthed the subgroups of the dataset. Generally, the misclassification rates agreed with other studies (see [25] and [19]). However, it is difficult to make a direct comparison because of different standardization (amongst other things).

The variable importance measures indicated similar genes across tree type. This was particularly evident with the important genes of ensembles of AAMRTs and MRTPCs. The genes deemed to be important by the algorithm agreed with those found in supervised classification studies. For example, the Zyxin gene of the Golub dataset is commonly selected in classification rules in [26]. In [22] Zyxin, MB-1, and adipsin are illustrated as genes useful in distinguishing AML from ALL. To highlight the power of the variable importance lists, we took the top five variables found by the ensemble of AAMRTs in Table VI and grew a single AAMRT using only these variables. For the two group case, the number of misclassifications decreased to four; and for the three group case, the number of misclassifications decreased to five. The variable importance lists here are derived without external knowledge of the grouping structure. Therefore, these important variables may determine not only known groups but also smaller subgroups. The important variables warrant further investigation as biomarkers.

Finally, because of the stability and dimension reduction associated with the ensembles of MRTPCs, we suggest using these trees to create the weighted co-occurrence matrices. If suitable parameters of the ensemble were unknown prior to analysis, it is advisable to use randomly selected values. With further research, weighted ensembles of MRTFSs could also give accurate clustering solutions. The optimal dimension of the response space deserves further investigation.

VII. CONCLUSION

Cluster analysis is an essential exploratory technique, often applied as a first step in the analysis of a large microarray dataset [27]. Cluster ensembles have been shown to give improved accuracy and stability over individual clustering solutions, in many fields [28],[29],[30] including Bioinformatics [19]. The improvements afforded by cluster ensembles on large datasets parallel results obtained with regression and classification ensembles. It is mooted that greater accuracy is attainable if the researcher is willing to take a weighted aggregation of the individual clustering models to give the ensemble.

This research suggested a technique of creating a weighted cluster ensemble suitable for large datasets. Each cluster model, a multivariate regression tree, was weighted according to its predictive strength. The clusters were found in the response space of each tree; either the entire dataset, or the reduced dimension space constructed with the factor scores or the principal component scores of the dataset.

The resulting weighted co-occurrence matrix was split using SBK and the clusters agreed with the known groupings in the data. Interestingly, the technique uncovered two known subgroups in one dataset. Weighted co-occurrence matrices created with MRTPCs produced the most stable results across the datasets. Because of their stability and dimension reduction we recommend MRTPCs as the preferred tree type.

A valuable byproduct of the ensemble technique was an indication of the variables that were important in determining the clusters. Growing a single AAMRT on the variables selected as important, decreased the number of misclassifications. The important variables could warrant further investigation; some variables (genes) could be biomarkers of a disease.

REFERENCES

[1] L. Kaufman and P. Rousseeuw, "Clustering by means of medoids," in *Statistical Data Analysis based on the L_1 Norm*, Y. Dodge, Ed. Amsterdam: Elsevier, 1987, pp. 405-416.

[2] X. Z. Fern and C. E. Brodley, "Cluster ensembles for high dimensional data clustering: An empirical study," Oregon State University, Department of Computer Science, Technical Report CS06-30-02, 2006.

[3] C. Smyth and D. Coomans, "Predictive Weighting for Cluster Ensembles," *unpublished.*

[4] F. Questier, R. Put, D. Coomans, B. Walczak, and Y. V. Heyden, "The use of CART and multivariate regression trees for supervised and unsupervised feature selection," *Chemometrics and Intelligent Laboratory Systems*, vol. 76, pp. 45-54, 2005.

[5] C. Smyth, D. Coomans, Y. Everingham, and T. Hancock, "Auto-Associative Multivariate Regression Trees for Cluster Analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 80, pp. 120-129, 2006.

[6] C. Smyth, D. Coomans, and Y. Everingham, "Clustering noisy data in a reduced dimension space via multivariate regression trees," *Pattern Recognition*, vol. 39, pp. 424-431, 2006.

[7] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001.

[8] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B-Statistical Methodology*, vol. 58, pp. 267-288, 1996.

[9] T. Grotkjaer, O. Winther, B. Regenberg, and J. Nielsen, "Robust multi-scale clustering of large DNA microarray datasets with the consensus algorithm," *Bioinformatics*, vol. 22, pp. 58-67, 2006.

[10] K. Y. Yeung, D. R. Haynor, and W. L. Ruzzo, "Validating clustering for gene expression data," *Bioinformatics*, vol. 17, pp. 309-318, 2001.

[11] R. Tibshirani, G. Walther, D. Botstein, and P. Brown, "Cluster validation by prediction strength," *Journal of Computational and Graphical Statistics*, vol. 14, pp. 511-528, 2005.

[12] T. Hancock, "Multivariate Consensus Trees: Tree-based clustering and profiling for mixed data types," James Cook University, School of Mathematical and Physical Sciences, PhD Thesis 2006.

[13] L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Belmont, CA: Wadsworth, 1984.

[14] M. R. Segal, "Tree-Structured Methods for Longitudinal Data," *Journal of the American Statistical Association*, vol. 87, pp. 407-418, 1992.

[15] G. H. Dunteman, *Principal Components Analysis*. Newbury Park, CA: Sage, 1989.

[16] J. Friedman and B. Popescu, "Importance Sampled Learning Ensembles," Stanford University, Department of Statistics, Technical Report 2003.

[17] L. Kaufman and P. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. New York: Wiley, 1990.

[18] S. Monti, P. Tamayo, J. Mesirov, and T. Golub, "Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data," *Machine Learning*, vol. 52, pp. 91-118, 2003.

[19] S. Dudoit and J. Fridlyand, "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics*, vol. 19, pp. 1090-1099, 2003.

[20] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, et al., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, pp. 6745-6750, 1999.

[21] E. Acuna and C. Rodriguez, "dprep: Data preprocessing and visualization functions for classification. R package version 1.0"

[22] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, et al., "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science*, vol. 286, pp. 531-537, 1999.

[23] S. Dudoit and J. Fridlyand, "A prediction-based resampling method for estimating the number of clusters in a dataset," *Genome Biology*, vol. 3, pp. 0036.1-0036.21, 2002.

[24] R Development Core Team, "R: A language and environment for statistical computing" Vienna, Austria: R Foundation for Statistical Computing, 2006.

[25] G. J. McLachlan, R. W. Bean, and D. Peel, "A mixture model-based approach to the clustering of microarray expression data," *Bioinformatics*, vol. 18, pp. 413-422, 2002.

[26] S. G. Baker and B. S. Kramer, "Identifying genes that contribute most to good classification in microarrays," *BMC Bioinformatics*, vol. 7, 2006.

[27] S. Datta and S. Datta, "Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes," *BMC Bioinformatics*, vol. 7, 2006.

[28] D. Greene, A. Tsymbal, N. Bolshakova, and P. Cunningham, "Ensemble Clustering in Medical Diagnostics," presented at 17th IEEE Symposium on Computer-Based Medical Systems, Texas, 2004.

[29] A. Weingessel, E. Dimitriadou, and K. Hornik, "An Ensemble Method for Clustering," presented at Distributed Statistical Computing, Vienna, Austria, 2003.

[30] A. Strehl and J. Ghosh, "Cluster ensembles - a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583-617, 2002.

SYNOPSIS

The manuscript, “Predictive Weighting for Cluster Ensembles” illustrated the superiority of weighted cluster ensembles over simple average ensembles. SBK, an extension of the traditional k-means algorithm, accurately partitioned the weighted co-occurrence matrix. The modified FOMs were demonstrated to accurately approximate the natural number of clusters in the datasets. The manuscript also established the ensemble’s weighted variable importance lists. Some results that were not included in the manuscripts are explicated below.

In “Predictive Weighting for Cluster Ensembles”, the percentage of variables randomly chosen to grow each individual tree was 33%. The results were also obtained for both 50% and 67%. Table A shows the number of misclassifications using SBK on the co-occurrence matrices created by ensembles of AAMRTs for the Vietnam dataset. The results of the other two datasets followed similar trends. The columns show the number of nodes and the minimum terminal node size of the individual trees. The rows show the percentage of variables randomly selected to grow the tree. The top number of each cell shows the number of misclassifications using the squared error loss function, and the bottom number of each cell shows the number of misclassifications using the absolute loss function of SBK. The smaller percentage of variables gave less misclassification. This was probably because the smaller percentage of variables created a larger variety of individual tree models for potential inclusion within the ensemble.

Table A. Number of misclassifications using SBK on the co-occurrence matrices created by ensembles of AAMRTs with varying numbers of variables.

Number of Nodes		**3**			**6**			**10**		
Minimum Terminal Node Size		**5**	**10**	**15**	**5**	**10**	**15**	**5**	**10**	**15**
Percentage of Variables	**33%**	19	19	19	13	13	13	19	7	13
		24	24	24	12	13	18	10	7	16
	50%	29	29	29	19	20	20	21	21	21
		29	29	29	20	20	19	15	22	21
	67%	40	40	40	17	17	20	17	17	22
		34	34	34	21	21	18	21	54	21

"Predictive Weighting for Cluster Ensembles" suggested that "cross-validated" trees could be used in the ensemble. This implies that each tree within the ensemble is cross-validated: a proportion of the observational units are removed, the tree is grown with the remaining observational units and randomly selected variables, and the left-out observational units are predicted by the tree. The process is repeated until all observational units have been predicted. These "cross-validated" predictions are then used to assess the predictive accuracy of each tree. The $(i,j)^{th}$ element of the co-occurrence matrix of each tree gives the proportion of times observational unit i and j were clustered into the same terminal node when neither was in the left-out subset.

Growing five-fold (20% of observational units removed) cross-validated trees on the Vietnam dataset, post processing the trees with the lasso heuristic, and then partitioning the weighted co-occurrence matrices using SBK gives misclassification rates shown in Table B. The columns show the number of nodes and the minimum terminal node size of each individual tree. The rows show the percentage of variables randomly selected to grow the tree. The top number of each cell shows the number of misclassifications using the squared error loss function, and the bottom number of each cell shows the number of misclassifications using the absolute loss function of SBK. The results show almost an opposite trend to those shown in Table A. As the number of variables available to the AAMRT algorithm increases, the number of misclassifications decreases in Table B. Here, if the percentage of variables available to the algorithm is too low, coupled with the randomness being introduced by cross-validation, there is too much instability in the individual trees.

Table B. Number of misclassifications using SBK on the co-occurrence matrices created by ensembles of cross-validated AAMRTs with varying numbers of variables.

Number of Nodes		**3**			**6**			**10**		
Minimum Terminal Node Size		**5**	**10**	**15**	**5**	**10**	**15**	**5**	**10**	**15**
Percentage of Variables	**33%**	56	56	28	21	21	21	20	18	19
		30	30	30	21	21	20	20	19	18
	50%	27	27	27	13	15	15	20	19	21
		27	27	27	14	15	16	21	19	22
	67%	29	29	29	15	13	11	18	21	22
		29	29	29	15	15	15	17	20	21

"Post processing regression ensembles: imposing parsimony to improve predictions" showed that the stochastic procedure, evolution strategies, produced excellent results when creating parsimonious regression ensembles. Table C shows the number of misclassifications (Vietnam dataset) when the lasso procedure is replaced by evolution strategies. Cross-validated trees as in Table B are used. The columns show the number of nodes and the minimum terminal node size of each individual tree. The rows show the percentage of variables randomly selected to grow the tree. The top number of each cell shows the number of misclassifications using the squared error loss function, and the bottom number of each cell shows the number of misclassifications using the absolute loss function of SBK. The results of Table B and Table C are fairly similar particularly when the trees were grown as large, or larger than the number of clusters in the dataset. However, as previously mentioned the lasso heuristic is much faster than the evolution strategies, so it remains the recommended post processing technique.

Table C. Number of misclassifications using SBK on the co-occurrence matrices created by ensembles of cross-validated AAMRTs weighted with evolution strategies.

Number of Nodes		**3**			**6**			**10**		
Minimum Terminal Node Size		**5**	**10**	**15**	**5**	**10**	**15**	**5**	**10**	**15**
Percentage of Variables	**33%**	56	56	56	15	19	21	13	21	19
		57	57	37	21	25	24	16	19	20
	50%	26	26	26	15	15	13	20	17	20
		27	27	27	15	16	16	15	18	20
	67%	69	69	69	16	12	14	17	19	13
		62	62	62	16	15	15	9	20	14

The results of "Predictive Weighting for Cluster Ensembles" showed that the parameters of the individual trees could affect the misclassification rates of SBK. A potential solution to this problem is to grow the individual trees with parameters that are selected randomly from a suitable set. This should provide results that are a compromise solution of the set parameters. Random parameters were trialed in the second manuscript "Clustering Microarrays with Predictive Weighted Ensembles".

In "Clustering Microarrays with Predictive Weighted Ensembles" the developed methodology was applied successfully to two DNA microarray datasets. Splitting the co-occurrence matrices created with the ensembles of MRTPCs gave the best results.

"Clustering Microarrays with Predictive Weighted Ensembles" commented that the presented misclassification rates agreed with other studies. Whilst it is impossible to compare directly to the other studies because of slightly different standardization steps used and the sheer number of other studies available, here we present a brief comparison to two other analyses. The other analyses were chosen because they are both well known; one uses factor analysis for dimension reduction, and the other uses an ensemble co-occurrence matrix.

The first analysis, conducted by McLachlan, et al. (2002), applied mixture models of factor analyzers to the same two datasets. Mixture models of factor analyzers also rely on factor analysis to reduce the dimensionality of the data. The dimension of the factor space controls the number of parameters to be estimated in the mixture model (McLachlan and Peel 2000). Mixture models of factor analyzers are expected to be useful when clustering a small number of observational units on the basis of a large number of variables (for example, when clustering genetic data) (McLachlan and Peel 2000). McLachlan, et al. (2002) have realized the benefits of this technique and have developed "EMMIX-GENE": a computer program that fits a mixture model of factor analyzers to a dataset.

In McLachlan, et al. (2002), the authors followed the same standardization procedures for both datasets as outlined in "Clustering Microarrays with Predictive Weighted Ensembles", with the exception of the base 10 logarithm transformation. Instead, the authors use the natural logarithm transformation. When clustering the Alon dataset, EMMIX-GENE selected 446 genes as relevant. Using $q = 6$ factors, the authors found two clusters that had 22 misclassifications when considering the tumor/non-tumor partition. However, there was a change in protocol during the experiment and if the clusters were compared to an old/new protocol partition, rather than the tumor/non-tumor partition, there were four misclassifications. When clustering on the basis of the top 50 genes selected by EMMIX-GENE, there were 21 misclassifications when considering the tumor/non-tumor partition, and 21 misclassifications when considering the old/new protocol partition. The dataset was also clustered using smaller clusters of genes, with misclassification rates dependent on the groups.

When considering the Golub dataset, the EMMIX-GENE approach reduced the number of variables to 2015 genes. If the top 50 genes were used to cluster the dataset using $q = 8$ factors into two clusters, there were ten misclassifications. Also, when started from random and k-means based starts, the EMMIX-GENE approach could cluster the dataset into the three groups corresponding to the three subgroups with 15 misclassifications. The EMMIX-GENE approach can produce slightly superior results when it is initialized using the known classifications of the data. However, as this information is usually unknown to the investigator, the results are not reported.

The second analysis performed by Dudoit and Fridlyand (2003) used ensemble clustering techniques to group the Golub dataset. The dataset was standardized in exactly the same manner as "Clustering Microarrays with Predictive Weighted Ensembles", including the base 10 logarithm transformation, with the exception of standardizing the genes. The 100 most variable genes were used in the analysis. There were two techniques of finding an ensemble partition. The first used plurality voting: taking the majority cluster label for each observational unit. The second used a co-occurrence matrix where the $(i,j)^{th}$ element denoted the proportion of times observational units i and j had been clustered together. The co-occurrence matrix was converted to a dissimilarity matrix and partitioned. Both techniques misclassified three observational units when the dataset was divided into the three known subgroups.

In summary, the results showed that weighting the cluster solutions according to their predictability improved the ensemble's accuracy. In this fashion, regression trees were shown to be excellent ensemble members for not only regression and classification, but cluster analysis as well. The technique, SBK, which further enforces the predictability of the solution, was shown to find accurate partitions of the co-occurrence matrix. The weighted variable importance list suggested variables which have been identified before as capable of differentiating clusters. The modified FOMs produced estimates of the natural number of clusters that were very close to the known number of classes.

CONCLUDING SUMMARY

"Super-sized" is not just an adjective used in takeaway stores daily, but also in statistics to describe the size of datasets that are occurring with increasing frequency. Analyzing data of this dimension presents unique statistical challenges and is a fairly new endeavour for statisticians. It is well known that single models will not represent all the data available within a large dataset. This has motivated the application of ensembles to large datasets. Ensembles combine many models and produce stable and accurate representations of the data.

Typically, ensembles average the outputs of the individual models. However, weighting (post processing) the individual models in the ensemble according to their relative accuracy gives improved performance whilst maintaining the ensemble's stability. Post processing ensembles has been investigated in the regression and classification contexts; however it has remained relatively unexplored in the cluster ensemble framework. This is because of the lack of a "gold standard" associated with cluster analyses. Whilst it is relatively easy to assess the performance of an individual classification or regression model by comparing its predictions to the observed values, there is no such gold standard associated with cluster models.

This thesis described a technique of creating a weighted cluster ensemble, where the weight of each individual clustering solution was based upon its predictability. A regression technique, multivariate regression trees, was used to obtain individual cluster solutions. Multivariate regression trees were extended to auto-associative multivariate regression trees so that they could be applied to datasets consisting entirely of explanatory variables. The auto-associative multivariate regression trees were further extended for use with high dimensional data. The dataset was reduced using principal components analysis or factor analysis and the resulting principal component scores or factor scores constituted the response set. The regression trees were shown to produce accurate clusters in both the low dimensional setting, and the noisy, high dimensional setting.

Each solution was then weighted according to its predictive accuracy. Each tree within the ensemble was aiming to predict its response: either the explanatory variables (AAMRT); the principal component scores (MRTPC); or the factor scores (MRTFS). Each tree within the ensemble could be weighted in terms of how well it predicted the response, that is, its predictive accuracy. Many different weighting strategies were trialed in this thesis. The best weighting strategy in terms of speed and predictive accuracy was the lasso heuristic. The lasso heuristic found sparse weights, enforcing a parsimonious ensemble with high predictive accuracy. The results supported those already presented in the literature: weighting models in the ensemble according to their accuracy improves the ensemble's performance.

The co-occurrence matrix of each tree was multiplied by the tree's assigned weight and these matrices were then aggregated together. The overall result was a weighted co-occurrence matrix. The weighted co-occurrence matrix represented the grouping structure over the ensemble in a manageable form. The structure in a weighted co-occurrence matrix was clearer than the structure within a simple average co-occurrence matrix.

The weighted co-occurrence matrix was partitioned using a novel modification to the traditional k-means algorithm. The modified k-means algorithm, similarity-based k-means (SBK), further enforced the predictability of the solution. The results of partitioning weighted co-occurrence matrices using the new algorithm were excellent.

An estimate of the natural number of clusters could also be obtained using the modified k-means algorithm by considering the average predictive capability of the algorithm, for any number of clusters, k. The technique is based on figure of merit theory (Yeung, et al. 2001). The estimates tended to agree with the known number of clusters in the dataset.

A valuable by-product of this weighted ensemble approach is a variable importance list. Each individual tree in the ensemble produced a variable importance list. Variables that were chosen to partition the observational units were given an associated importance based on their position within the tree. The importance list of each tree was multiplied by the tree's weight and the lists were then aggregated. The overall list indicated which variables were the most important in determining the clusters. The variables in the overall weighted list had high agreement with variables known to contain grouping structure.

This thesis created an integrated approach for clustering large datasets. Not only were accurate clusters obtained, but also an estimate of the natural number of clusters in the dataset, and a variable importance measure. Therefore, the technique could be applied to any continuous dataset and a holistic cluster solution obtained.

The developed techniques were applied in the later stages of this thesis to two DNA microarray datasets. The results were comparable to others in the literature. Known subgroups were found accurately in one dataset. The variables selected as important have been identified in other studies as potential biomarkers. The estimates of the optimal number of clusters in the dataset agreed with the known number of groups.

In conclusion, the technique of creating a weighted cluster ensemble suggested by this thesis involved:

- **Obtaining an ensemble of clustering solutions using multivariate regression trees.** Trees are the ideal ensemble member because the accuracy of trees can easily be assessed unlike other clustering techniques.
- **Weighting each tree within the ensemble according to its predictive accuracy and viewing the weighted regression ensemble as a weighted cluster ensemble via the construction of an overall weighted co-occurrence matrix.**
- **Splitting the weighted co-occurrence matrix using SBK.** SBK accurately partitions the dataset by enforcing the predictability of the entire weighted co-occurrence matrix.
- **Exploiting the cluster ensemble to assess the natural number of clusters and obtain variable importance lists.** These estimates are accurate because the redundant models have been removed from the ensemble.

Overall, the results illustrated the power of weighting models within a cluster ensemble.

FUTURE WORK

This section details some potential avenues of future research that have arisen as a result of this thesis.

ASSESSING MODEL COMPLEXITY VIA THE ELBOW OF THE RELATIVE ERROR CURVE

The manuscript “Auto-Associative Multivariate Regression Trees for Cluster Analysis” proposed the use of the elbow of the relative error curve to determine model complexity (similar to using the elbow of a principal components scree plot). In “Clustering Noisy Data in a Reduced Dimension Space via Multivariate Regression Trees”, the elbow was further examined. AAMRTs may offer an “inbuilt” estimator of cluster number: the elbow of the relative error curve. This procedure warrants further investigation.

SEARCHING FOR DIFFERENT SHAPED CLUSTERS

In this thesis, multivariate regression trees split a node to find two new nodes that are more homogeneous using the “sums of squares” criterion. It is likely that this splitting criterion facilitates finding clusters that are spherical. Using different splitting criteria, such as entropy or kurtosis, may engender the discovery of differently shaped clusters when clustering with regression trees. Whilst not investigated in this thesis, this notion would be a very interesting avenue to pursue.

LOCAL DIMENSION REDUCTION

Local dimension reduction (within each cluster) offers several advantages over global dimension reduction (Ghahramani and Hinton 1996). MRTPCs and MRTFSs use global dimension reduction: the entire dataset is initially reduced and the tree is grown on the reduced subset. It may be beneficial to perform local dimensionality reduction by updating scores within each node. However, if the nodes became too small the updates would not be possible. This is indeed a potential avenue of future exploration.

OPTIMAL RESPONSE FOR THE FACTOR SPACE

As stated in “Clustering Noisy Data in a Reduced Dimension Space via Multivariate Regression Trees” the degree of dimension reduction directly impacts upon the tree’s ability to find the natural clusters. An avenue of investigation may develop a method to determine the optimal degree of dimension reduction. The technique could be based upon Ben-Hur and Guyon’s (2003) research.

VARIABLE SELECTION APPROACH

This paper treated the post processing of the regression ensemble as a standard regression problem. However, because one objective is finding a small set of models with non-zero weights, post processing may be treated as a variable selection technique. Applying variable selection techniques to find the optimal models of the ensemble could form the basis of future work.

ASSESSING THE IMPORTANCE OF THE SIMPLE AVERAGE ENSEMBLE

Some members of the statistical community may believe that parsimony of ensembles is irrelevant because the ensemble is essentially a black box, and reducing its size will not increase its interprebility. Furthermore, some may believe that by removing models from the ensemble, the ensemble’s stability and accuracy are destroyed. Whilst the results of “Parsimonious Ensembles for Regression” and “Post processing regression ensembles: imposing parsimony to improve predictions” show the opposite of the latter claim, more research regarding the large, simple average ensemble is warranted. One way of assessing the importance of the simple average ensemble is including its predictions as a potential model for the post processing technique. If the post processing technique gave the simple average predictions a large weight, then removing too many models from the ensemble would possibly induce inferior results.

MODEL SELECTION

The manuscript “Post processing regression ensembles: imposing parsimony to improve predictions” shows that post processing an ensemble can not only produce more accurate ensembles, but also indicate which models (either simple linear regression or regression trees) are best suited to analyzing a particular dataset. With the addition of more model types such as neural networks to the ensemble, careful analysis of the post processed weights could indicate complicated trends within the data.

RANDOM RESPONSE VARIABLES

To introduce further diversity into the ensemble of trees for cluster analysis, each tree could be grown using random response variables. Each tree will be modelling different regions and this may induce the discovery of clusters that would otherwise be masked in the entire variable space. However, this idea would raise complexity issues and the post processing procedure would also need to be suitably modified.

CLUSTERING VARIABLES

The weighted cluster ensemble technique could easily be used to cluster together variables as well as the observational units. By transposing the matrix, the variables would be clustered together. However, if the number of variables is large, the final weighted co-occurrence matrix will be massive. Therefore, future work could involve "approximate" co-occurrence matrices, which only consider nearest neighbors (Fred and Jain 2003).

SBK CONVERGENCE

SBK may benefit from an integrated genetic algorithm to induce convergence to a global optimum (Lu, Lu, Fotouhi, Deng and Brown 2004). This will avoid different solutions created by different starting partitions.

DIFFERENT STANDARDIZATION

The preprocessing steps used in "Clustering Microarrays with Predictive Weighted Ensembles" have been performed by other researchers analyzing the same datasets. However, it must be noted that superior results may be attainable if different standardization routines were incorporated. Discovering optimal standardization routines could be an avenue of future investigation.

USING A REDUCED SUBSET OF VARIABLES AS INPUT TO A FURTHER ANALYSIS

Traditional statistical procedures such as multiple linear regression may produce erroneous and unstable results when applied to very large datasets. These techniques may suffer from the collinearity between the variables or some variables may serve only to distort the true structure of the dataset by creating unnecessary and uninterpretable noise. Therefore, large datasets require a variable selection step before supervised and/or unsupervised statistical analyses are performed.

The variables designated as important by the weighted cluster ensemble could constitute the ‘selected variables’. Such a variable selection step would contrast to those more commonly used, which consider known classifications or simple statistics like variance. Rather, the selected variables of the weighted ensemble would define groups in the data. Future work could assess if using these selected variables as input to other statistical analyses results in increased accuracy.

SUPPORTING THEORY

This section describes in more detail the theory behind the weighted cluster ensemble technique. The nomenclature may not agree perfectly with the published manuscripts, however nomenclature across this entire section is consistent. As a general guide, vectors are lowercase italics with tildes, and matrices are uppercase italics and emboldened.

1 REGRESSION TREES

Regression trees were used extensively throughout this thesis. This section describes the multivariate regression tree methodology in more detail.

1.1 Regression Tree Splitting

Regression trees, a well known prediction method, recursively partition a dataset, so that a “tree” is obtained. In the beginning, all observational units are contained within one parent node. The node is partitioned in two, by the value of an explanatory variable, such that the two new nodes are as homogeneous as possible with respect to the response variables. This process continues, previously non-partitioned nodes are split in two, until the non-partitioned nodes are deemed sufficiently homogeneous.

A regression tree partitions a node, t, into two on the basis of the value of an explanatory variable. For each independent explanatory variable, X_{p}, p=1,...,P all possible splits of the node, t, into $\mathrm{t_L}$ and $\mathrm{t_R}$ are considered. If the variable, X_{p}, is continuous or ordinal, all splits of the following form are considered:

$$\mathrm{t_L} = \left\{ \underset{\sim}{x}_{\mathrm{i}} \in \mathrm{t}, X_{\mathrm{p}} \le \mathrm{c} \right\} \tag{1.1}$$

$$\mathrm{t_R} = \left\{ \underset{\sim}{x}_{\mathrm{i}} \in \mathrm{t}, X_{\mathrm{p}} > \mathrm{c} \right\} \tag{1.2}$$

where c is an element of all the values X_p can take. If the variable, X_p, is categorical with F categories, all $2^{F-1}-1$ subsets must be considered. For each variable, the split that maximizes $\Delta R(t)$ is calculated and saved. The value $\Delta R(t)$ is given by:

$$\Delta R(t) = R(t) - \left(R(t_L) + R(t_R) \right) \tag{1.3}$$

$$R(t) = \frac{1}{n} \sum_{\underset{\sim}{x}_i \in t} \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t) \right)^T \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t) \right) \tag{1.4}$$

$$R(t_L) = \frac{1}{n} \sum_{\underset{\sim}{x}_i \in t_L} \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t_L) \right)^T \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t_L) \right) \tag{1.5}$$

$$R(t_R) = \frac{1}{n} \sum_{\underset{\sim}{x}_i \in t_R} \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t_R) \right)^T \left(\underset{\sim}{y}_i - \overline{\underset{\sim}{y}}(t_R) \right) \tag{1.6}$$

where $\overline{\underset{\sim}{y}}(t)$ is the mean of the response of observational units in node t; $\underset{\sim}{x}_i \in t$ indexes those observational units in node t; and $\underset{\sim}{y}_i$ is the response vector of the ith observational unit. Then the variable with the maximum $\Delta R(t)$ is selected as the node's optimal split, s_t^*.

At each stage, it must be decided which node to split. The optimal split, s^*, is calculated for each node. Then, the best optimal split over the set of all nodes is chosen and the corresponding node is partitioned.

At the point where non-partitioned nodes are deemed either small or homogeneous enough, the splitting stops and a node that has not been split in two is referred to as "terminal".

1.2 Predicting an Observational Unit Using a Regression Tree

To predict an observational unit, it is dropped down the tree, following the binary decision functions at each node, until it comes to rest in a terminal node. Each terminal node contains a predicted value for all observational units falling into that node. The predicted value for a terminal node, $\mathrm{t_{term}}$ is:

$$\hat{\underset{\sim}{y}}(\mathrm{t_{term}}) = \bar{\underset{\sim}{y}}(\mathrm{t_{term}}) = \frac{1}{\mathrm{n_{t_{term}}}} \sum_{\underset{\sim}{x}_\mathrm{i} \in \mathrm{t_{term}}} \underset{\sim}{y}_\mathrm{i} \qquad (1.7)$$

where the sum is over $\underset{\sim}{y}_\mathrm{i}$ such that $\underset{\sim}{x}_\mathrm{i} \in \mathrm{t_{term}}$; and $\mathrm{n_{t_{term}}}$ is the total number of cases in the terminal node. By choosing the predicted value as the average response for all cases of $(\underset{\sim}{x}_\mathrm{i}, \underset{\sim}{y}_\mathrm{i})$ falling into $\mathrm{t_{term}}$ the resubstitution estimate for the mean squared error of the predictor is minimized (Breiman, et al. 1984).

1.3 Auto-Associative Multivariate Regression Trees

The observational units in each of the terminal nodes of a regression tree are intuitively the clusters of the dataset. The clusters are found in the response space and the explanatory variables that form the tree are deemed to be important in determining the clusters. The position of an explanatory variable in the tree denotes its importance in determining these homogeneous subsets. The predicted value of a terminal node can represent the cluster.

However multivariate regression trees can not be applied in the traditional clustering sense, because the trees require response variables and traditional unconstrained cluster analysis is performed on datasets consisting entirely of explanatory variables. To solve this conundrum, auto-associative multivariate regression trees (Questier, et al. 2005) were suggested. Auto-associative multivariate regression trees replicate the explanatory variables as response variables. The regression tree is grown using identical response and explanatory variables, and in this way the clusters will be as homogeneous as possible with respect to all the explanatory variables.

1.4 Multivariate Regression Trees with Principal Component Scores and Multivariate Regression Trees with Factor Scores

To allow regression trees to be applied to large, noisy datasets, the dimension of the dataset must be reduced. This thesis uses principal components analysis and factor analysis as the methods of dimension reduction. The resulting principal component scores or factor scores are used as the response dataset. The original dataset is retained as the explanatory dataset.

2 DIMENSION REDUCTION TECHNIQUES

Principal components analysis and factor analysis were used in this thesis as dimension reduction techniques. The theory surrounding both techniques is expanded upon here.

2.1 Principal Components Analysis

Principal components analysis (Pearson 1901) linearly transforms the variables of a dataset to a set of uncorrelated variables called principal components and is a commonly used data reduction technique (Dunteman 1989). The full principal components analysis model is given by:

$$\underset{\sim}{\varphi} = \boldsymbol{\Gamma}^{\mathrm{T}} \underset{\sim}{x} \qquad (2.1)$$

where $\underset{\sim}{x}$ is a observational unit measured on P variables; $\underset{\sim}{\varphi}$ is a vector of principal component scores; and here $\boldsymbol{\Gamma}$ is obtained from the spectral decomposition of $\boldsymbol{R}$, the correlation matrix.

Principal components analysis attempts to model the total variance of the original dataset via the uncorrelated principal components (Dunteman 1989). If the variables are standardized, principal components analysis finds the first principal component by maximizing the variance of the linear combination $\underset{\sim}{\gamma}_1^{\mathrm{T}}\underset{\sim}{x}$ given that $\underset{\sim}{\gamma}_1^{\mathrm{T}}\underset{\sim}{\gamma}_1 = 1$ (Dunteman 1989). The second principal component $\underset{\sim}{\gamma}_2$ maximizes the variance of $\underset{\sim}{\gamma}_2^{\mathrm{T}}\underset{\sim}{x}$ given that $\underset{\sim}{\gamma}_2^{\mathrm{T}}\underset{\sim}{\gamma}_2 = 1$ and $\underset{\sim}{\gamma}_1^{\mathrm{T}}\underset{\sim}{\gamma}_2 = 0$. This reasoning can be continued for $\underset{\sim}{\gamma}_{\mathrm{P}}$ principal components. Using these definitions, $\underset{\sim}{\gamma}_{\mathrm{p}}$ is the p^{th} eigenvector of $\boldsymbol{R}$:

$$\boldsymbol{R}\underset{\sim}{\gamma}_{\mathrm{p}} = \lambda_{\mathrm{p}}\underset{\sim}{\gamma}_{\mathrm{p}} \tag{2.2}$$

where λ_{p} is the p^{th} eigenvalue (and also the variance of the p^{th} principal component). Obviously if $\boldsymbol{\mathit{\Lambda}}$ is a diagonal matrix with the eigenvalues ordered along the diagonal from largest to smallest then $\boldsymbol{R\mathit{\Gamma}} = \boldsymbol{\mathit{\Gamma\Lambda}}$ where the matrix $\boldsymbol{\mathit{\Gamma}}$ is given by:

$$\boldsymbol{\mathit{\Gamma}} = \begin{bmatrix} \underset{\sim}{\gamma}_1 & \underset{\sim}{\gamma}_2 & \cdots & \underset{\sim}{\gamma}_{\mathrm{P}} \end{bmatrix}. \tag{2.3}$$

The value $\lambda_{\mathrm{p}}/\mathrm{P}$ gives the proportion of variation explained by the p^{th} principal component (Dunteman 1989). The loadings vector $\sqrt{\lambda_{\mathrm{p}}}\underset{\sim}{\gamma}_{\mathrm{p}}$ gives the correlations of the variables with the principal components (Dunteman 1989).

There are as many principal components as variables. However, often only the first q principal components (that is the largest q eigenvalues and corresponding eigenvectors) are used to approximate the correlation matrix $\boldsymbol{R}$ and indeed the entire dataset. The value q can be deduced using a number of heuristics but most often a scree plot (Cattell 1966) of the eigenvalues is obtained and the location of the "elbow" is considered to be the optimal q. Observational units are represented in the q dimensional subspace via the first q principal component scores. Dimension reduction is achieved by discarding the latter $P-q$ principal components.

2.2 Factor Analysis

Factor analysis assumes that the observed variables can be generated as a linear combination of q unobservable common factors plus a unique factor (Dunteman 1989). The factor analysis model is given by:

$$\underset{\sim}{x}_i = \underset{\sim}{\mu} + \boldsymbol{F}\underset{\sim}{v}_i + \underset{\sim}{\varepsilon}_i \tag{2.4}$$

where $\underset{\sim}{\mu}$ is a mean vector; $\boldsymbol{F}$ is a $P \times q$ matrix of factor loadings; $\underset{\sim}{v}_i$ is a q dimensional vector of hypothetical common factors; and $\underset{\sim}{\varepsilon}_i$ is a unique factor.

The difference between principal components analysis and factor analysis is obvious after comparing equations (2.1) and (2.4). Principal components analysis creates new variables as linear sums of the original variables and factor analysis models the original variables as linear sums of "hypothetical factors".

Because the factors are artificially created it is necessary to impose some assumptions so that the factor loadings can be estimated. Firstly, the $\underset{\sim}{v}_i$ are assumed to be independent and identically distributed as $N(\underset{\sim}{0}, \boldsymbol{I}_q)$, (where $\boldsymbol{I}_q$ is a $q \times q$ identity matrix), independently of the errors $\underset{\sim}{\varepsilon}_i$. Secondly, the errors are assumed to be independent and identically distributed as $N(\underset{\sim}{0}, \boldsymbol{\Psi})$ where $\boldsymbol{\Psi}$ is a diagonal matrix (McLachlan, et al. 2002). The diagonality of $\boldsymbol{\Psi}$ implies that the variables are independent given the factors (Ghahramani and Hinton 1996). In other words, the factors attempt to explain the correlations of the observed variables (McLachlan and Peel 2000). Conditional on the $\underset{\sim}{y}_i$'s, the $\underset{\sim}{x}_i$'s are independently and identically distributed as $N(\underset{\sim}{\mu} + \boldsymbol{F}\underset{\sim}{y}_i, \boldsymbol{\Psi})$ (McLachlan and Peel 2000). Unconditionally, the standardized $\underset{\sim}{x}_i$'s are independently and identically distributed as $N\left(\underset{\sim}{\mu}, \boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{\Psi}\right)$ (McLachlan and Peel 2000) and therefore factor analysis models the correlation matrix as $\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{\Psi}$.

The decomposition of the correlation matrix shows how factor analysis partitions the total variance into "common" and "unique" variance. The matrix, $\boldsymbol{F}$, is a matrix of factor loadings: the correlations of a variable with a common factor (Kline 1994). The diagonal matrix $\boldsymbol{\Psi}$ contains the specific variance of each variable: the "unique" variance of each variable that is not associated with the other variables.

This decomposition of the correlation matrix also finally elucidates the estimation of the factor loadings. If the matrix $\boldsymbol{R}_c$ is defined as: $\boldsymbol{R}_c = \boldsymbol{R} - \boldsymbol{\Psi} = \boldsymbol{F}\boldsymbol{F}^T$, then the matrix $\boldsymbol{F}$ can be obtained via methods such as maximum likelihood estimation or principal components analysis after the estimation of $\boldsymbol{\Psi}$ (Dunteman 1989).

Substantial dimension reduction can be obtained using factor analysis if $\mathrm{q} \ll \mathrm{P}$. Observational units can be represented in the q dimensional factor space by the estimated values of the hypothetical common factors, called factor scores. The factor scores can be obtained easily using regression analysis.

3 CREATING INDIVIDUAL MODELS FOR ENSEMBLES

Models were individualized throughout this thesis in a variety of ways. This section details the sampling strategies that were employed to create these models.

3.1 Bootstrapping and Random Features

Assume n observational units are sampled with replacement from a dataset. The obtained dataset is called a 'bootstrap dataset'. Usually the sampling is repeated B times to obtain B bootstrap datasets. The bootstrap datasets can be used to assess the accuracy of a parameter estimate or prediction. Bootstrapping also allows many different models to be grown on the same dataset, and these models can be combined into an ensemble.

Another sampling strategy that can be used to create models suitable for an ensemble is "random features". Here, random subsets of the variables are selected B times. These B datasets induce B individualized models that are combined into an ensemble.

3.2 Bagging

Combining models into an ensemble given by B bootstrap datasets is commonly referred to as 'bagging' (**b**ootstrap **agg**regat**ing**). Bagging is known to improve parameter estimates and predictions, and reduce overall variance (Hastie, et al. 2001). Bagging regression models commonly involves taking the average prediction of an observational unit given by a set of models grown on separate bootstrap datasets. Mathematically, this is given by (Hastie, et al. 2001):

$$\hat{f}_{\text{bag}}(\underset{\sim}{x}) = 1/\text{B}\sum_{\text{b}=1}^{\text{B}} \hat{f}^{*\text{b}}(\underset{\sim}{x}) \qquad (3.1)$$

where B is the number of bootstrap datasets; $\hat{f}^{*b}(\underset{\sim}{x})$ is the prediction of observational unit $\underset{\sim}{x}$ made by the model grown on the b^{th} bootstrap dataset. Bagging classification models commonly involves taking the majority vote over the classifications given by the models grown on B bootstrap datasets.

3.3 Random Forests

A well known bagging algorithm is random forests (Breiman 2001). Random forests combine bagging with randomization. A bootstrap sample is taken, and a tree is grown on the bootstrap sample. The number of variables, m, used to determine the split at a node is pre-specified to be much smaller than the number available. At each node, m variables are randomly selected and the best split is found amongst these.

4 BAYESIAN LINEAR REGRESSION

Bayesian linear regression was used in this thesis to calculate the coefficients for a weighted ensemble of regression models. In this section Bayesian theory is briefly introduced and then Bayesian linear regression is further explored. Gelman, Carlin, Stern and Rubin (1995) was used extensively as a reference source throughout this section.

Bayesian theory incorporates exogenous knowledge (a “prior” distribution) to estimate the posterior distribution of a parameter. The prior distribution may incorporate known or assumed information about the parameter before consideration of the data. The prior distribution of the parameter is transformed to the posterior distribution by the data via the model likelihood. The extent to which the posterior distribution resembles the prior distribution is inversely proportional to the amount of data available. From the posterior distribution an estimate of the parameter can be obtained.

Mathematically, the posterior distribution is given by Bayes' Theorem:

$$p(\underset{\sim}{\theta} \mid \underset{\sim}{y}) \propto p(\underset{\sim}{y} \mid \underset{\sim}{\theta})\, p(\underset{\sim}{\theta}) \tag{4.1}$$

where $p(\underset{\sim}{\theta})$ is the prior probability distribution of the parameter; $p(\underset{\sim}{y} \mid \underset{\sim}{\theta})$ is the sampling distribution (likelihood function) dependent on the parameter $\underset{\sim}{\theta}$; and $p(\underset{\sim}{\theta} \mid \underset{\sim}{y})$ is the posterior probability distribution of the parameter given the observed data. The '|' notation indicates the dependence of the posterior distribution on the data.

Thus, the problem of estimating a parameter reduces to

1) Determining a suitable prior probability distribution of the parameter (section 4.1)
2) Calculating the likelihood function of the data (section 4.2)
3) Determining the posterior distribution of the parameter (section 4.3)
4) Sampling from the (approximated) posterior distribution (section 4.4)

4.1 Determining a Suitable Prior Distribution

The prior distribution is either an "informative" or "non-informative" distribution dependent on the information available to the analyst. If there is no prior knowledge or information pertaining to the parameter on hand, a "non-informative" prior distribution is recommended. A "non-informative" prior is usually uniform across the parameter space. If information pertaining to the parameter is available, it can easily be incorporated into an "informative prior" such as a normal prior centered at the assumed value of the parameter and a variance inversely proportional to the certainty of the prior estimate of the parameter.

A prior distribution will also be either “conjugate” or “non-conjugate”. A “conjugate prior” implies the posterior distribution of the parameter is of the same class as the prior distribution. It is often advantageous to choose a conjugate prior, for it engenders computational convenience and facilitates interpretation of posterior distributions. However, there are situations which necessitate the use of a non-conjugate prior.

4.2 Calculating the Likelihood Function of the Data

The sampling distribution, $p(\underset{\sim}{y} \mid \underset{\sim}{\theta})$ in (4.1) is given by:

$$p(\underset{\sim}{y} \mid \underset{\sim}{\theta}) = \prod_{i=1}^{n} f(y_i \mid \underset{\sim}{\theta}) \tag{4.2}$$

where $f(y_i \mid \underset{\sim}{\theta})$ is the probability density function of y_i given the parameter vector $\underset{\sim}{\theta}$.

If we regard the sampling distribution $p(\underset{\sim}{y} \mid \underset{\sim}{\theta})$ in (4.2) as a function of $\underset{\sim}{\theta}$ for a fixed $\underset{\sim}{y}$ rather than a function of $\underset{\sim}{y}$ for fixed $\underset{\sim}{\theta}$ we refer to it as a “likelihood function”, $l(\underset{\sim}{\theta} \mid \underset{\sim}{y})$:

$$l(\underset{\sim}{\theta} \mid \underset{\sim}{y}) = p(\underset{\sim}{y} \mid \underset{\sim}{\theta}) = \prod_{i=1}^{n} f(y_i \mid \underset{\sim}{\theta}). \tag{4.3}$$

Because of the constant of proportionality in (4.1), the resultant posterior distribution is not changed by multiplication of the likelihood by

(a) a constant

(b) any function of $\underset{\sim}{y}$ only.

This detail is often utilised to simplify the calculation of the posterior distribution.

4.3 Determining the Posterior Distribution of the Parameter

The joint probability distribution of $\underset{\sim}{\theta}$ and $\underset{\sim}{y}$ is given by:

$$p(\underset{\sim}{\theta},\underset{\sim}{y}) = p(\underset{\sim}{\theta})p(\underset{\sim}{y}\,|\,\underset{\sim}{\theta}) \tag{4.4}$$

where $p(\underset{\sim}{\theta})$ is the prior distribution of $\underset{\sim}{\theta}$; and $p(\underset{\sim}{y}\,|\,\underset{\sim}{\theta})$ is the sampling distribution of $\underset{\sim}{\theta}$. Using Bayes' Rule, we obtain the posterior density $p(\underset{\sim}{\theta}\,|\,\underset{\sim}{y})$:

$$p(\underset{\sim}{\theta}\,|\,\underset{\sim}{y}) = \frac{p(\underset{\sim}{\theta},\underset{\sim}{y})}{p(\underset{\sim}{y})} = \frac{p(\underset{\sim}{\theta})p(\underset{\sim}{y}\,|\,\underset{\sim}{\theta})}{p(\underset{\sim}{y})} \tag{4.5}$$

where $p(\underset{\sim}{y}) = \sum_{\underset{\sim}{\theta}} p(\underset{\sim}{\theta})p(\underset{\sim}{y}\,|\,\underset{\sim}{\theta})$ for discrete $\underset{\sim}{\theta}$; and $p(\underset{\sim}{y}) = \int_{\underset{\sim}{\theta}} p(\underset{\sim}{\theta})p(\underset{\sim}{y}\,|\,\underset{\sim}{\theta})d\underset{\sim}{\theta}$ for continuous $\underset{\sim}{\theta}$. Because $p(\underset{\sim}{y})$ is independent of $\underset{\sim}{\theta}$ and can therefore be considered constant for fixed $\underset{\sim}{y}$, we arrive at (4.1). Substituting (4.3), we arrive at the expression for the posterior distribution:

$$p(\underset{\sim}{\theta}\,|\,\underset{\sim}{y}) \propto p(\underset{\sim}{\theta})l(\underset{\sim}{\theta}\,|\,\underset{\sim}{y}) = p(\underset{\sim}{\theta})\prod_{\mathrm{i}=1}^{\mathrm{n}} f(y_{\mathrm{i}}\,|\,\underset{\sim}{\theta}). \tag{4.6}$$

4.4 Sampling from the (Approximated) Posterior Distribution

An estimate of the parameters can be obtained from the posterior distribution via sampling. If the posterior distribution is of a known family, drawing a sample is straightforward. This occurs when a conjugate prior has been used. More complicated posterior densities arise in multiparameter problems where the density may involve a product of two families. In this instance separating the posterior probability distribution using basic probability theory facilitates sampling. The joint posterior distribution is separated into the product of marginal and conditional densities:

$$p(\underset{\sim}{\theta}\,|\,\underset{\sim}{y}) = p(\underset{\sim}{\theta}_{\kappa}\,|\,\underset{\sim}{\theta}_{-\kappa},\underset{\sim}{y})p(\underset{\sim}{\theta}_{-\kappa}\,|\,\underset{\sim}{y}) \tag{4.7}$$

where $p(\underset{\sim}{\theta}_\kappa \mid \underset{\sim}{\theta}_{-\kappa}, \underset{\sim}{y})$ is the conditional distribution of $\underset{\sim}{\theta}_\kappa$ given $\underset{\sim}{\theta}_{-\kappa}$; $p(\underset{\sim}{\theta}_{-\kappa} \mid \underset{\sim}{y})$ is the marginal density of $\underset{\sim}{\theta}_{-\kappa}$, all parameters not considered in $\underset{\sim}{\theta}_\kappa$, calculated by averaging over $\underset{\sim}{\theta}_\kappa$:

$$p(\underset{\sim}{\theta}_{-\kappa} \mid \underset{\sim}{y}) = \int_{\underset{\sim}{\theta}_\kappa} p(\underset{\sim}{\theta} \mid \underset{\sim}{y}) d\underset{\sim}{\theta}_\kappa. \tag{4.8}$$

To sample from the posterior, a draw is first taken from $p(\underset{\sim}{\theta}_{-\kappa} \mid \underset{\sim}{y})$, $\hat{\underset{\sim}{\theta}}_{-\kappa}$. The conditional distribution $p(\underset{\sim}{\theta}_\kappa \mid \underset{\sim}{\theta}_{-\kappa}, \underset{\sim}{y})$ is then given by $p(\underset{\sim}{\theta}_\kappa \mid \hat{\underset{\sim}{\theta}}_{-\kappa}, \underset{\sim}{y})$, and a draw is taken from this conditional distribution to give $\hat{\underset{\sim}{\theta}}_\kappa$. This technique of separating the posterior is suitable when $p(\underset{\sim}{\theta}_\kappa \mid \underset{\sim}{\theta}_{-\kappa}, \underset{\sim}{y})$ and $p(\underset{\sim}{\theta}_{-\kappa} \mid \underset{\sim}{y})$ are easily recognized. If they are not easily identified, techniques exist to approximate $p(\underset{\sim}{\theta}_\kappa \mid \underset{\sim}{\theta}_{-\kappa}, \underset{\sim}{y})$ and $p(\underset{\sim}{\theta}_{-\kappa} \mid \underset{\sim}{y})$ and indeed the entire joint posterior distribution.

When the parameter vector $\underset{\sim}{\theta}$ is very large, simulation techniques are also available. Markov Chain simulations are suited for high dimensional problems. The two main types of Markov Chain simulations are the generalized Metropolis algorithm and the Gibbs sampler. The Gibbs sampler approach was used extensively in this thesis and it is described in more detail below.

4.4.1 Gibbs Sampler

The parameter vector $\underset{\sim}{\theta}$, is subdivided into $\underset{\sim}{\theta} = (\underset{\sim}{\theta}_1, \underset{\sim}{\theta}_2, ..., \underset{\sim}{\theta}_K)$ such that the conditional distributions $p(\underset{\sim}{\theta}_\kappa \mid \underset{\sim}{\theta}_1, ..., \underset{\sim}{\theta}_{\kappa-1}, \underset{\sim}{\theta}_{\kappa+1}, ..., \underset{\sim}{\theta}_K, \underset{\sim}{y})$ are easily recognized or approximated for all $\underset{\sim}{\theta}_\kappa$. Initial values are assigned to each $\underset{\sim}{\theta}_\kappa, \kappa = 1, ..., K$, to give $\underset{\sim}{\theta}^{(0)} = (\underset{\sim}{\theta}_1^{(0)}, ..., \underset{\sim}{\theta}_K^{(0)})$. The algorithm then iterates through each $\underset{\sim}{\theta}_\kappa$ updating it with a draw from its conditional distribution given the current estimates of $\underset{\sim}{\theta}_{-\kappa}$. That is, $\underset{\sim}{\theta}_{\kappa}^{(\zeta)}$ is a draw from $p(\underset{\sim}{\theta}_{\kappa} \mid \underset{\sim}{\theta}_{1}^{(\zeta)}, \underset{\sim}{\theta}_{2}^{(\zeta)}, ..., \underset{\sim}{\theta}_{\kappa-1}^{(\zeta)}, \underset{\sim}{\theta}_{\kappa+1}^{(\zeta-1)}, ..., \underset{\sim}{\theta}_{K}^{(\zeta-1)})$. Each full iteration gives values for all parameters, $\underset{\sim}{\theta}^{(\zeta)}$, where $\underset{\sim}{\theta}^{(\zeta)} = (\underset{\sim}{\theta}_{1}^{(\zeta)}, ..., \underset{\sim}{\theta}_{K}^{(\zeta)})$. After a large number of iterations called a "burn-in", $\underset{\sim}{\theta}^{(\zeta)}$ converges to a draw from the joint posterior distribution. It is possible to measure convergence for the chain, see for example Gelman, et al. (1995).

4.4.2 Using Samples to Summarize the Posterior Density Function

Once samples of the posterior distribution have been obtained, they may be used to summarize the posterior distribution. For example, the posterior mean is easily calculated once sufficient samples have been drawn.

4.5 Using Bayesian Linear Regression to Calculate Weight Coefficients

Bayesian theory can be extended to ordinary linear regression by imposing prior distributions on the parameters of the regression model, namely the regression coefficients, $\underset{\sim}{\beta}$ and the variance of the error terms, σ^2. The choice of prior distribution influences the final regression coefficients selected. For example, a non-informative prior will induce a multivariate normal conditional posterior for the regression coefficients, with mean equal to the classical least squares estimates. If different priors are imposed, Bayesian linear regression will suggest different regression coefficients.

In this thesis, Bayesian linear regression theory was used to obtain the weight coefficients for the ensemble: the traditional data matrix, $\boldsymbol{X}$, was replaced by the predictions matrix $\hat{\boldsymbol{Y}}$[1]; and the traditional regression coefficients were replaced by the weight coefficients, $\underset{\sim}{\omega}$. The vector $\underset{\sim}{y}$ is the observed univariate response vector. Four

[1] The predictions matrix is an $n*M$ matrix with each column containing the predictions of the n observational units for a single model.

different prior distributions were used to obtain the weight coefficients. All necessary distributions are described below and the resulting posterior distributions are also derived.

4.5.1 Distributions

4.5.1.1 Multivariate Normal Distribution

The density function of a K dimensional parameter vector $\underset{\sim}{\theta}$ arising from a multivariate normal distribution is given by:

$$\begin{aligned}&\underset{\sim}{\theta} \sim N(\underset{\sim}{\mu},\Sigma)\\ &p(\underset{\sim}{\theta}) = (2\pi)^{-\mathrm{K}/2}\,|\,\Sigma\,|^{-1/2}\exp\left(-1/2\,(\underset{\sim}{\theta}-\underset{\sim}{\mu})^T\Sigma^{-1}(\underset{\sim}{\theta}-\underset{\sim}{\mu})\right)\\ &-\infty < \underset{\sim}{\theta} < \infty\end{aligned} \tag{4.9}$$

where Σ is a symmetric, positive definite $\mathrm{K}\times\mathrm{K}$ covariance matrix; and $\underset{\sim}{\mu}$ is a $\mathrm{K}\times 1$ vector.

4.5.1.2 Inverse Gamma Distribution

The density function of a parameter θ arising from an inverse gamma distribution is given by:

$$\begin{aligned}&\theta \sim Inv-gamma(\alpha,\beta)\\ &p(\theta) = \beta^{\alpha}/\Gamma(\alpha)\,\theta^{-(\alpha+1)}\exp\left(-\beta/\theta\right)\\ &\theta > 0.\end{aligned} \tag{4.10}$$

4.5.1.3 Multivariate T Distribution

The probability density function of a K dimensional parameter vector $\underset{\sim}{\theta}$, arising from a multivariate t distribution is given by:

$$\begin{aligned}&\underset{\sim}{\theta} \sim t_{\nu}(\underset{\sim}{\mu},\Sigma)\\ &p(\underset{\sim}{\theta}) = \frac{\Gamma((\nu+\mathrm{K})/2)}{\Gamma(\nu/2)\nu^{\mathrm{K}/2}\pi^{\mathrm{K}/2}}\,|\,\Sigma\,|^{-1/2}\left(1+\frac{1}{\nu}\left(\underset{\sim}{\theta}-\underset{\sim}{\mu}\right)^T\Sigma^{-1}\left(\underset{\sim}{\theta}-\underset{\sim}{\mu}\right)\right)^{-(\nu+\mathrm{K})/2}\\ &-\infty < \underset{\sim}{\theta} < \infty\end{aligned} \tag{4.11}$$

where ν is the degrees of freedom, $\nu > 0$; $\underset{\sim}{\mu}$ is a $K \times 1$ location vector; and Σ is a symmetric $K \times K$ positive definite scale matrix. As ν increases, the multivariate t distribution approaches the normal distribution.

4.5.1.4 Weibull Distribution

The density function of a parameter θ, arising from a Weibull distribution is given by

$$\begin{aligned} &\theta \sim Weib(\delta, \upsilon) \\ &p(\theta) = \delta\upsilon\theta^{\delta-1}\exp\left(-\upsilon\theta^{\delta}\right) \\ &\theta > 0. \end{aligned} \tag{4.12}$$

4.5.1.5 Double Exponential Distribution

The probability density function of a parameter θ, arising from a double exponential distribution is given by:

$$\begin{aligned} &\theta \sim dexp(\mu, \tau) \\ &p(\theta) = \frac{\tau}{2}\exp(-\tau\,|\,\theta - \mu\,|) \\ &-\infty < \theta < \infty. \end{aligned} \tag{4.13}$$

4.5.2 Posterior Distributions

4.5.2.1 Posterior Distribution - Multivariate Normal Prior

The prior of the weight coefficients of the ensemble is taken to be multivariate normal:

$$p(\underset{\sim}{\omega}) \sim N(\underset{\sim}{\mu}_0, \Sigma_0). \tag{4.14}$$

The hyperparameters, $\underset{\sim}{\mu}_0$ and Σ_0, are specified a priori. The vector, $\underset{\sim}{\mu}_0$ is specified as a vector of zeros, reflecting that prior to analysis all regression coefficients (models) are expected to be non-informative. The matrix Σ_0, varies according to the certainty that the weight coefficients are all equal to zero.

The prior distribution of the error variance, σ^2, is taken to be inverse gamma. The inverse gamma distribution for the error variance is used in conjunction with all the different priors on the weight coefficients. The parameter, σ^2, is not of interest, and therefore the inverse-gamma prior is used throughout:

$$p(\sigma^2) \sim Inv-gamma(\alpha,\beta). \tag{4.15}$$

Values of the hyperparameters α,β, are dictated by the data.

The combination of the multivariate normal prior on the $\underset{\sim}{\omega}$ and the inverse gamma for the error variance is considered the conjugate prior for regression analysis. The joint prior is given by:

$$\begin{aligned} p(\underset{\sim}{\omega},\sigma^2) &= p(\underset{\sim}{\omega} \mid \sigma^2)p(\sigma^2) \\ &= p(\underset{\sim}{\omega})p(\sigma^2). \end{aligned} \tag{4.16}$$

Substituting (4.14),(4.15),(4.9) and (4.10), the prior becomes:

$$p(\underset{\sim}{\omega},\sigma^2) = (2\pi)^{-K/2} \mid \boldsymbol{\Sigma}_0 \mid^{-1/2} \exp\left(-1/2(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)^T \boldsymbol{\Sigma}_0^{-1}(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)\right)\beta^{\alpha}/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)}\exp\left(-\beta/\sigma^2\right). \tag{4.17}$$

The posterior of the parameters is:

$$p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) = p(\underset{\sim}{y} \mid \underset{\sim}{\omega},\sigma^2)p(\underset{\sim}{\omega} \mid \sigma^2)p(\sigma^2). \tag{4.18}$$

Substituting (4.17) and the model likelihood:

$$l(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) = (2\pi)^{-n/2}(\sigma^2)^{-n/2}\exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})^T(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})\right) \tag{4.19}$$

(4.18) becomes:

$$\begin{aligned} p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) = (2\pi)^{-n/2}(\sigma^2)^{-n/2}\exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})^T(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})\right)\times \\ (2\pi)^{-K/2} \mid \boldsymbol{\Sigma}_0 \mid^{-1/2}\exp\left(-1/2\,(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)^T \boldsymbol{\Sigma}_0^{-1}(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0)\right)\times \\ \beta^{\alpha}/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)}\exp\left(-\beta/\sigma^2\right). \end{aligned} \tag{4.20}$$

Rearranging gives:

$$
\begin{aligned}
&p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right)\times \\
&\qquad \exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}^T\underset{\sim}{y}-2\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\underset{\sim}{y}+\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\underset{\sim}{\omega})\right)\exp\left(-1/2(\underset{\sim}{\omega}^T\boldsymbol{\Sigma}_0^{-1}\underset{\sim}{\omega})\right) \\
&p(\underset{\sim}{\omega},\sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right)\times \\
&\qquad \exp\left(-1/2\left(\underset{\sim}{\omega}^T\left((\sigma^2)^{-1}\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}+\boldsymbol{\Sigma}_0^{-1}\right)\underset{\sim}{\omega}-2\underset{\sim}{\omega}^T(\sigma^2)^{-1}\hat{\boldsymbol{Y}}^T\underset{\sim}{y}\right)\right)\exp\left(-(2\sigma^2)^{-1}\underset{\sim}{y}^T\underset{\sim}{y}\right).
\end{aligned}
\tag{4.21}
$$

Conditioning on σ^2, (4.21) shows the conditional distribution of the weight coefficients is normal:

$$
p(\underset{\sim}{\omega} \mid \sigma^2, \underset{\sim}{y}) \sim N\left(\left((\sigma^2)^{-1}\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}+\boldsymbol{\Sigma}_0^{-1}\right)^{-1}(\sigma^2)^{-1}\hat{\boldsymbol{Y}}^T\underset{\sim}{y}, \left((\sigma^2)^{-1}\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}+\boldsymbol{\Sigma}_0^{-1}\right)^{-1}\right). \tag{4.22}
$$

The prior distribution, $p(\underset{\sim}{\omega} \mid \sigma^2)$ is independent of σ^2, however in the conditional posterior distribution $p(\underset{\sim}{\omega} \mid \sigma^2 \underset{\sim}{y})$ is dependent on σ^2. This implies that the prior is semi-conjugate.

The multivariate normal prior with $\underset{\sim}{\mu}_0 = \underset{\sim}{0}$ will induce estimates of the regression coefficients similar to the ordinary least squares estimates for large prior variance. However, as we decrease the prior variance, greater emphasis is placed on the prior belief than the data.

4.5.2.2 Posterior Distribution - Multivariate T Prior

The prior on the weight coefficients is taken to be multivariate t:

$$
p(\underset{\sim}{\omega}) \sim t_v(\underset{\sim}{\mu}_0, \boldsymbol{\Sigma}_0). \tag{4.23}
$$

The hyperparameters $\underset{\sim}{\mu}_0, \boldsymbol{\Sigma}_0$ are specified to reflect prior belief regarding the weight coefficients. The vector, $\underset{\sim}{\mu}_0$ is specified as a vector of zeros, reflecting that prior to analysis all regression coefficients (models) are expected to be non-informative. The matrix $\boldsymbol{\Sigma}_0$, varies according to the certainty that the weight coefficients are all equal to zero. The wider tails of a t distribution allow larger weight coefficients than the normal distribution.

The prior on the error variance σ^2, is taken to be inverse gamma, with hyperparameters dictated by the data. This combination of priors is not conjugate. The joint prior is given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega},\sigma^2)=p(\underset{\sim}{\omega})p(\sigma^2)\\
&p(\underset{\sim}{\omega},\sigma^2)=\frac{\Gamma((\nu+K)/2)}{\Gamma(\nu/2)\nu^{K/2}\pi^{K/2}}|\boldsymbol{\Sigma}_0|^{-1/2}\left(1+\frac{1}{\nu}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)^T\boldsymbol{\Sigma}_0^{-1}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)\right)^{-(\nu+K)/2}\beta^\alpha/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)}\exp\left(-\beta/(\sigma^2)\right).
\end{aligned}\tag{4.24}$$

The posterior of the regression coefficients (substituting (4.19) for the model likelihood) is given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega},\sigma^2\,|\,\underset{\sim}{y})=l(\underset{\sim}{\omega},\sigma^2\,|\,\underset{\sim}{y})p(\underset{\sim}{\omega},\sigma^2)\\
&p(\underset{\sim}{\omega},\sigma^2\,|\,\underset{\sim}{y})=(2\pi)^{-n/2}(\sigma^2)^{-n/2}\exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})^T(\underset{\sim}{y}-\hat{\boldsymbol{Y}}\underset{\sim}{\omega})\right)\times\\
&\qquad\frac{\Gamma((\nu+K)/2)}{\Gamma(\nu/2)\nu^{K/2}\pi^{K/2}}|\boldsymbol{\Sigma}_0|^{-1/2}\left(1+\frac{1}{\nu}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)^T\boldsymbol{\Sigma}_0^{-1}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)\right)^{-(\nu+K)/2}\times\\
&\qquad\beta^\alpha/\Gamma(\alpha)(\sigma^2)^{-(\alpha+1)}\exp\left(-\beta/(\sigma^2)\right)\\
&p(\underset{\sim}{\omega},\sigma^2\,|\,\underset{\sim}{y})\propto(\sigma^2)^{-n/2-(\alpha+1)}\exp\left(-\beta/(\sigma^2)\right)\exp\left(-(2\sigma^2)^{-1}\left(\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\underset{\sim}{\omega}-2\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\underset{\sim}{y}+\underset{\sim}{y}^T\underset{\sim}{y}\right)\right)\times\\
&\qquad\left(1+\frac{1}{\nu}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)^T\boldsymbol{\Sigma}_0^{-1}\left(\underset{\sim}{\omega}-\underset{\sim}{\mu}_0\right)\right)^{-(\nu+K)/2}.
\end{aligned}\tag{4.25}$$

The joint posterior probability density function is not of known form, but integrates to a finite value. The conditional distribution of the vector of weight coefficients is:

$$p(\underset{\sim}{\omega}\,|\,\sigma^2,\underset{\sim}{y})\propto\exp\left(-(2\sigma^2)^{-1}\left(\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\underset{\sim}{\omega}-2\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\underset{\sim}{y}\right)\right)\left(1+\frac{\underset{\sim}{\omega}^T\boldsymbol{\Sigma}_0^{-1}\underset{\sim}{\omega}}{\nu}\right)^{-(\nu+K)/2}.\tag{4.26}$$

4.5.2.3 Posterior Distribution - Weibull Prior

The prior of each weight coefficient is assumed to be a Weibull distribution

$$p(\omega_\kappa) \sim Weib(\delta_\kappa, \upsilon_\kappa). \tag{4.27}$$

The Weibull prior ensures all regression coefficients are greater than zero. The value of δ_κ is always 2. The hyperparameter υ_κ is varied according to the prior belief that the regression coefficient is equal to zero.

The prior distribution for the error variance is again an inverse gamma distribution, with hyperparameters dictated by the data. The joint prior is then given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega}, \sigma^2) = p(\underset{\sim}{\omega}) p(\sigma^2) \\
&p(\underset{\sim}{\omega}, \sigma^2) = p(\sigma^2) \prod_{\kappa=1}^{K} p(\omega_\kappa) \\
&p(\underset{\sim}{\omega}, \sigma^2) = \beta^\alpha / \Gamma(\alpha) (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \prod_{\kappa=1}^{K} \delta_\kappa \upsilon_\kappa \omega_\kappa^{\delta_\kappa - 1} \exp\left(-\upsilon_\kappa \omega_\kappa^{\delta_\kappa}\right) \\
&p(\underset{\sim}{\omega}, \sigma^2) \propto (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \exp\left(-\sum_{\kappa=1}^{K} \upsilon_\kappa \omega_\kappa^2\right) \prod_{\kappa=1}^{K} \omega_\kappa .
\end{aligned} \tag{4.28}$$

The joint posterior distribution is given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega}, \sigma^2 \mid \underset{\sim}{y}) = l(\underset{\sim}{\omega}, \sigma^2 \mid \underset{\sim}{y}) p(\underset{\sim}{\omega}, \sigma^2) \\
&p(\underset{\sim}{\omega}, \sigma^2 \mid \underset{\sim}{y}) \propto (\sigma^2)^{-n/2-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \prod_{\kappa=1}^{K} \omega_\kappa \times \\
&\qquad \exp\left(-(2\sigma^2)^{-1}\left(\underset{\sim}{\omega}^T \hat{\boldsymbol{Y}}^T \hat{\boldsymbol{Y}} \underset{\sim}{\omega} - 2\underset{\sim}{\omega}^T \hat{\boldsymbol{Y}}^T \underset{\sim}{y} + \underset{\sim}{y}^T \underset{\sim}{y}\right) - \sum_{\kappa=1}^{K} \upsilon_\kappa \omega_\kappa^2\right).
\end{aligned} \tag{4.29}$$

The posterior is not recognizable as coming from a known distribution family. However, the posterior distribution integrates to a finite positive value.

The conditional posterior distribution for a single regression coefficient (used in the Gibb's sampler) is:

$$p(\omega_\kappa \mid \underset{\sim}{\omega}_{-\kappa}, \sigma^2, \underset{\sim}{y}) \propto \omega_\kappa \exp\left(-(2\sigma^2)^{-1}\left(\omega_\kappa \left[\hat{\boldsymbol{Y}}^T \hat{\boldsymbol{Y}}\right]_{\kappa,\kappa} \omega_\kappa + 2\omega_\kappa \left[\hat{\boldsymbol{Y}}^T \hat{\boldsymbol{Y}}\right]_{\kappa,-\kappa} \underset{\sim}{\omega}_{-\kappa} - 2\omega_\kappa \left[\hat{\boldsymbol{Y}}^T \underset{\sim}{y}\right]_\kappa\right) - \upsilon_\kappa \omega_\kappa^2\right). \tag{4.30}$$

4.5.2.4 Posterior Distribution - Double Exponential Prior (Lasso)

The prior for each regression coefficient is assumed to be an independent double exponential distribution with hyperparameters τ_κ, μ_κ:

$$p(\omega_\kappa) \sim dexp(\tau_\kappa, \mu_\kappa). \tag{4.31}$$

The hyperparameters μ_κ are specified a priori as zero. The prior probability distribution of a single regression coefficient is then peaked at zero. With this prior, the lasso regularized regression coefficients are easily derived. The hyperparameter τ_κ reflects the certainty that a weight coefficient is equal to zero. All τ_κ are identical.

The error variance prior is inverse gamma with hyperparameters determined by the data. The joint prior probability density function is not conjugate and is given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega}, \sigma^2) = p(\underset{\sim}{\omega}) p(\sigma^2) \\
&p(\underset{\sim}{\omega}, \sigma^2) = p(\sigma^2) \prod_{\kappa=1}^{K} p(\omega_\kappa) \\
&p(\underset{\sim}{\omega}, \sigma^2) = \beta^\alpha / \Gamma(\alpha) (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \prod_{\kappa=1}^{K} \frac{\tau_\kappa}{2} \exp(-\tau_\kappa \,|\, \omega_\kappa - \mu_\kappa \,|) \\
&p(\underset{\sim}{\omega}, \sigma^2) \propto (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \exp\left(-\tau \sum_{\kappa=1}^{K} |\, \omega_\kappa \,|\right).
\end{aligned} \tag{4.32}$$

The joint posterior distribution is given by:

$$\begin{aligned}
&p(\underset{\sim}{\omega}, \sigma^2 \,|\, \underset{\sim}{y}) = p(\underset{\sim}{\omega}, \sigma^2) l(\underset{\sim}{\omega}, \sigma^2 \,|\, \underset{\sim}{y}) \\
&p(\underset{\sim}{\omega}, \sigma^2 \,|\, \underset{\sim}{y}) = l(\underset{\sim}{\omega}, \sigma^2 \,|\, \underset{\sim}{y}) p(\sigma^2) \prod_{\kappa=1}^{K} p(\omega_\kappa) \\
&p(\underset{\sim}{\omega}, \sigma^2 \,|\, \underset{\sim}{y}) \propto (\sigma^2)^{-(\alpha+1)} \exp\left(-\beta/\sigma^2\right) \exp\left(-\tau \sum_{\kappa=1}^{K} \omega_\kappa \,|\right) (2\pi)^{-n/2} (\sigma^2)^{-n/2} \exp\left(-(2\sigma^2)^{-1} (\underset{\sim}{y} - \hat{\boldsymbol{Y}} \underset{\sim}{\omega})^T (\underset{\sim}{y} - \hat{\boldsymbol{Y}} \underset{\sim}{\omega})\right).
\end{aligned} \tag{4.33}$$

The joint posterior distribution, although not of a known type integrates to a finite value.

The conditional distribution of the vector of regression coefficients given σ^2 is:

$$\begin{aligned} p(\underset{\sim}{\omega} \mid \sigma^2, \underset{\sim}{y}) &\propto \exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega})^T(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}) - \tau\sum_{\kappa=1}^{K} |\omega_\kappa| \right) \\ &\propto \exp\left(-(2\sigma^2)^{-1}(\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\underset{\sim}{\omega} - 2\underset{\sim}{\omega}^T\hat{\boldsymbol{Y}}^T\underset{\sim}{y}) - \tau\sum_{\kappa=1}^{K} |\omega_\kappa| \right). \end{aligned} \tag{4.34}$$

The conditional distribution of a single regression coefficient (used by the Gibb's sampler) is given by:

$$p(\omega_\kappa \mid \underset{\sim}{\omega}_{-\kappa}, \sigma^2, \underset{\sim}{y}) \propto \exp\left(-(2\sigma^2)^{-1}\left(\omega_\kappa \left[\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\right]_{\kappa,\kappa} \omega_\kappa + 2\omega_\kappa \left[\hat{\boldsymbol{Y}}^T\hat{\boldsymbol{Y}}\right]_{\kappa,-\kappa} \underset{\sim}{\omega}_{-\kappa} - 2\omega_\kappa \left[\hat{\boldsymbol{Y}}^T\underset{\sim}{y}\right]_\kappa \right) - \tau|\omega_\kappa| \right). \tag{4.35}$$

By consideration of (4.34) it can easily be seen that using independent double exponential prior distributions for each of the weight coefficients is analogous to the lasso (Tibshirani 1996) regularized regression:

$$\hat{\underset{\sim}{\omega}} = \arg\min \sum_{i=1}^{n} \left(y_i - \sum_{m=1}^{M} \omega_m \hat{Y}_{i,m} \right)^2 + \lambda \sum_{m=1}^{M} |\omega_m|. \tag{4.36}$$

5 EVOLUTIONARY ALGORITHMS

Evolutionary algorithms are optimization techniques that mimic the Darwin theory of natural selection. This thesis employs two types of evolutionary algorithms: genetic algorithms and evolution strategies. The main conceptual difference between genetic algorithms and evolution strategies is that the chromosomes of genetic algorithms contain $0's$ and $1's$, and the chromosomes of evolution strategies can contain any real number.

Initially, these algorithms randomly suggest solutions to an optimization problem. Each solution is encoded via a chromosome and the set of all chromosomes at any iteration is referred to as the population. Subject to some assessment of fitness, the best chromosomes of the ζ^{th} generation reproduce to create "offspring". The offspring are mutated with some probability and replace the parents to give the $\zeta+1^{th}$ generation. Over time (iterations), all chromosomes approach the best (local) solution and will become virtually identical. At this time, the algorithm is said to have converged, and the optimal solution to the optimization problem (encoded via a chromosome) is obtained.

From an optimization viewpoint, these algorithms can be considered an intelligent random search of the solution space. The randomised operators within genetic algorithms and evolution strategies imply these algorithms are more robust than other optimization techniques. The algorithms eventually converge to a solution, sometimes local, that minimizes/maximizes a loss function.

The stepwise procedure for these algorithms is:

1) Randomly create a population of chromosomes to give the initial generation
2) Then iterate:
    a) Calculate the fitness of each chromosome in the current generation.
    b) Breed pairs of chromosomes according to fitness to give offspring chromosomes.
    c) Mutate offspring with some probability.
    d) Replace chromosomes of current generation with offspring with some probability.

This iterative process is described in more detail below. Davis (1991) was used extensively as a reference source throughout this section.

5.1 Calculating Fitness Values

The fitness function is derived to ascertain which chromosomes provide the best solution to the optimization problem. The higher the fitness value, the better the chromosome solves the optimization problem.

5.1.1 Fitness Scaling

Fitness scaling is used to prevent premature convergence during the early iterations and to engender convergence to optimal solutions during the later iterations. In the early stages of iteration there may be one outstanding individual. If the fitness values are unscaled, the outstanding individual will contribute many offspring to the next generation, thereby increasing convergence to a suboptimal solution. In the later iterations, there may be little difference between the fitness values of the best and worst individuals in the generation. If the fitness values are left unscaled, the probability of a bad chromosome producing offspring for the next generation is only fractionally less than that of the probability of the best chromosome, and it is unlikely that the algorithm will converge. There are various methods of scaling fitness values, the evolution strategy of this thesis simply ranked the fitness values.

5.2 Breeding Chromosomes According to Fitness

Two chromosomes from the current population are randomly selected with the probability of any chromosome being selected equal to its (scaled) fitness value divided by the sum of all (scaled) fitness values. This is known as roulette wheel selection however other selection techniques exist. These two chromosomes (parents) reproduce to create two offspring via a process referred to as "crossover" (genetic algorithms) and "recombination" (evolution strategies). The selection and reproduction process is repeated until enough offspring are produced to replace the entire generation.

5.2.1 Crossover - Genetic Algorithm

The two chromosomes (parents) selected via roulette wheel selection reproduce to create two offspring via a process referred to as "crossover". Crossover entails a point between $1:\mathrm{P}$ (where P is the length of the chromosome) being randomly selected. After this point, the elements of the parents are interchanged to give the two offspring. This is known as one point crossover: other crossover definitions exist. The crossover operator occurs with some probability, η_{CROSS}. If crossover does not occur, the offspring are identical to the parents.

5.2.2 Recombination - Evolution Strategy

The two parent chromosomes in an evolution strategy reproduce to create two offspring chromosomes via a process called "recombination". The elements of the offspring chromosome, $\underset{\sim}{\omega}_{\mathrm{O}}$ are given as a mixture of the two parent chromosomes, $\underset{\sim}{\omega}_{\mathrm{P1}}$ and $\underset{\sim}{\omega}_{\mathrm{P2}}$:

$$\underset{\sim}{\omega}_{\mathrm{O}} = \alpha\underset{\sim}{\omega}_{\mathrm{P1}} + (1-\alpha)\underset{\sim}{\omega}_{\mathrm{P2}} \tag{5.1}$$

where $\underset{\sim}{\omega}_{\mathrm{O}}$ is the offspring chromosome vector; $\underset{\sim}{\omega}_{\mathrm{P1}}$ is the first parent chromosome vector; $\underset{\sim}{\omega}_{\mathrm{P2}}$ is the second parent chromosome vector; and $\alpha \in [-0.25, 1.25]$ drawn randomly. For each pair of parent chromosomes, two values of α are randomly drawn from the set to give two offspring chromosomes. This process is called line recombination. Other recombination techniques exist.

5.3 Mutation

The mutation operator reintroduces diversity into the population and can be fairly arbitrary.

5.3.1 Mutation - Genetic Algorithm

Generally, the mutation operator of a genetic algorithm permutes each element (bit) of any offspring chromosome with small probability η_{MUT}. If the operator mutates the bit, its value is flipped to the opposite of its current value.

5.3.2 Mutation - Evolution Strategy

The mutation operator of an evolution strategy is more complicated than that of a genetic algorithm. For each chromosome one bit, bit_ρ is randomly chosen to be mutated. The bit is then replaced by a mutated value, bit_ρ^{MUT}:

$$bit_\rho^{MUT} = bit_\rho + h_\rho \times g_\rho \times e_\rho \tag{5.2}$$

where $h_\rho \in \{-1,1\}$ chosen randomly; $g_\rho = 0.1 \times domain_\rho$; $e_\rho = 2^{-u_\rho \times f_\rho}$; $u_\rho \in [0,1]$ chosen randomly; and $f \in \{4,5,...,20\}$ chosen randomly.

The parameter f indirectly influences the size of the mutation. This mutation operator produces small mutations with a high probability and large mutations with a low probability.

5.4 Creation of a New Generation

After a complete set of offspring have been produced, it must be determined which individuals (from the current generation and the set of offspring) will form the new generation. This decision may be made on the basis of fitness values (which must also be calculated for the offspring). Another possibility is to replace the entire current generation by the set of offspring. This is known as generational replacement. Generational replacement is advantageous because it is simple and it avoids premature convergence of the chromosomes.

5.5 Using Genetic Algorithms and Evolution Strategies to Calculate Weight Coefficients

This section will explain genetic algorithms and evolution strategies in the context of calculating optimal weight coefficients.

5.5.1 Genetic Algorithm Approach

A genetic algorithm can be used to calculate which models are the most relevant in the ensemble. Each chromosome represents important models by a '1' in the corresponding bit position and superfluous models with a '0' in the corresponding bit position.

The initial generation is randomly generated by setting bits to zero with high probability and bits to one with low probability. This reflects the notion that it is likely most models contain no relevant information.

To calculate the fitness values of each chromosome, the set of models deemed to have a non-zero weight coefficient by the chromosome ('1' in corresponding bit position) are parsed to another method (for example Bayesian linear regression). The "shadow method" calculates the weights for these models only, and parses this information back to the fitness evaluation module of the genetic algorithm. The fitness of each chromosome is given by the inverse of the residual sums of squares, $\mathrm{RSS_j}$:

$$\mathrm{RSS_j} = \sqrt{\frac{1}{\mathrm{n}}\left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}_{\mathrm{j}}^{*}\underset{\sim}{\omega}_{\mathrm{j}}\right)^{T}\left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}_{\mathrm{j}}^{*}\underset{\sim}{\omega}_{\mathrm{j}}\right)} \tag{5.3}$$

where $\underset{\sim}{\omega}_{\mathrm{j}}$ is the vector of weights calculated by the shadow method for chromosome j; and $\hat{\boldsymbol{Y}}_{\mathrm{j}}^{*}$ is the set of models chromosome j indicates to have non-zero coefficients.

Therefore a chromosome with $1's$ in the bit positions of the most important models will have a high fitness value and a chromosome with $1's$ in the bit positions of models with small predictive capabilities will have a low fitness value.

Based on the fitness values, chromosomes are selected as parents and reproduce via crossover to produce offspring. Chromosomes which indicate the truly important models have a high chance of producing offspring. The offspring are mutated by setting bits with a corresponding weight of less than 0.05 to $'0'$. This enforces the expectation that only models with high weights are relevant.

The current generation is then replaced by the offspring generation (generational replacement) and the process iterates until the chromosomes converge. The genetic algorithm only allows a small number of models to have non-zero weight coefficients and will eventually converge to the set of models that give the lowest residual sums of squares with weights calculated using the shadow method (for example Bayesian regression).

5.5.2 Evolution Strategy Approach

An evolution strategy can be used to calculate the weight coefficients directly, without the "shadow method" a genetic algorithm requires. However, the evolution strategy will assign non-zero weights to all models, and therefore may not be as appropriate if the goal is to find the best, smallest set of non-zero weight coefficients.

Each chromosome is a possible set of weight coefficients. Explicitly, each element of a chromosome contains the weight assigned to the corresponding model. The initial generation is created by setting a few bits of each chromosome as a draw from a uniform distribution on $[0,1]$. Each chromosome is then standardized so that the sum of its elements equals one.

The fitness of the j^{th} chromosome is the inverse of the residual sums of squares, $\mathrm{RSS}_{\mathrm{j}}$:

$$\mathrm{RSS}_{\mathrm{j}} = \sqrt{\frac{1}{\mathrm{n}}\left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}_{\mathrm{j}}\right)^{T}\left(\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}_{\mathrm{j}}\right)} \tag{5.4}$$

where $\underset{\sim}{\omega}_{\mathrm{j}}$ is the j^{th} chromosome (weights are given directly by the chromosome). To prevent premature convergence, the fitness values are scaled.

Chromosomes with high fitness values (weights that give low residual sums of squares) have a high chance of being selected as parents. Parent chromosomes are selected via roulette wheel selection and produce offspring via line recombination. Line recombination is attractive in this instance as it indirectly enforces the condition that the elements of the offspring chromosomes sum to one.

The offspring chromosomes are mutated as previously described. Furthermore, if too many weights are non-zero, some weights are randomly set to zero. The offspring chromosomes are then renormalized such that the elements of each chromosome sum to one. The offspring replace the current generation (generational replacement) and the process is iterated until the chromosomes converge to the set of weights that minimize the residual sums of squares.

6 QUADRATIC PROGRAMMING

Quadratic programming was used in this thesis to calculate the weights for regression ensembles. Quadratic programs, the active set method, and the extension to calculating weight coefficients are explained in this section. Nocedal and Wright (1999) and Fletcher (1987) were used extensively as reference sources throughout this section.

A quadratic program refers to a constrained optimization problem, where the function to be maximized/minimized is quadratic and the constraints are linear. The function to be optimized is referred to as the objective function, $q(\underset{\sim}{w})$, and the solution as $\underset{\sim}{w}^*$. The constraints, can be both equality and inequality constraints, and create the feasible region. A quadratic program can be written as:

$$\min_{\underset{\sim}{w}} \; q(\underset{\sim}{w}) = \frac{1}{2}\underset{\sim}{w}^T \boldsymbol{G}\underset{\sim}{w} + \underset{\sim}{w}^T \underset{\sim}{d} \tag{6.1}$$

subject to:

$$\begin{aligned} \underset{\sim}{a}_j^T \underset{\sim}{w} &= b_j \quad j \in \mathrm{E} \\ \underset{\sim}{a}_j^T \underset{\sim}{w} &\geq b_j \quad j \in \mathrm{I} \end{aligned} \tag{6.2}$$

where E and I are finite sets of indices that reference the equality and inequality constraints respectively; $\boldsymbol{G}$ is a $\mathrm{K} \times \mathrm{K}$ symmetric Hessian matrix; $\underset{\sim}{d}, \underset{\sim}{w}$ and $\left\{\underset{\sim}{a}_j \mid j \in \mathrm{E} \cup \mathrm{I}\right\}$ are $\mathrm{K} \times 1$ dimensional vectors.

The quadratic program may be infeasible or unbounded and have no solution. However these two situations are easily exposed. If the quadratic program can be solved, the uniqueness of the solution is dependent on the nature of the Hessian matrix, $\boldsymbol{G}$. If $\boldsymbol{G}$ is indefinite, the solution is a local solution and the quadratic program is a "non-convex problem". If $\boldsymbol{G}$ is positive semi-definite, the solution is a global solution, and if $\boldsymbol{G}$ is positive definite the solution is global and unique. In these instances, the quadratic program is called a convex quadratic program.

There are a set of conditions satisfied at a local solution, $\underset{\sim}{w}^*$, of a quadratic program. These are called necessary conditions for a local solution and involve the Lagrange multipliers corresponding to $\underset{\sim}{w}^*$. For completeness, the Lagrangian function of the general constrained optimization problem:

$$\min_{\underset{\sim}{w}\in\mathbb{R}^{\mathrm{K}}} f(\underset{\sim}{w}) \tag{6.3}$$

subject to:

$$\begin{aligned} c_{\mathrm{j}}(\underset{\sim}{w}) &= 0 \quad \mathrm{j}\in\mathrm{E} \\ c_{\mathrm{j}}(\underset{\sim}{w}) &\geq 0 \quad \mathrm{j}\in\mathrm{I} \end{aligned} \tag{6.4}$$

where E and I are two finite sets of indices is given by:

$$\ell(\underset{\sim}{w},\underset{\sim}{\lambda}) = f(\underset{\sim}{w}) - \sum_{\mathrm{j}\in\mathrm{E}\bigcup\mathrm{I}} \lambda_{\mathrm{j}} c_{\mathrm{j}}(\underset{\sim}{w}). \tag{6.5}$$

The set of necessary conditions for the general constrained optimization problem are:

$$\begin{aligned} &\nabla_{\underset{\sim}{w}}\ell(\underset{\sim}{w}^*,\underset{\sim}{\lambda}^*) = 0 \\ &c_{\mathrm{j}}(\underset{\sim}{w}^*) = 0 \quad \text{for all } \mathrm{j}\in\mathrm{E} \\ &c_{\mathrm{j}}(\underset{\sim}{w}^*) \geq 0 \quad \text{for all } \mathrm{j}\in\mathrm{I} \\ &\lambda_{\mathrm{j}}^* \geq 0 \quad \text{for all } \mathrm{j}\in\mathrm{I} \\ &\lambda_{\mathrm{j}}^* c_{\mathrm{j}}(\underset{\sim}{w}^*) = 0 \quad \text{for all } \mathrm{j}\in\mathrm{E}\bigcup\mathrm{I}. \end{aligned} \tag{6.6}$$

These conditions are referred to as the Karush-Kuhn-Tucker (KKT) conditions. Any point that satisfies the KKT conditions is known as a KKT point. The KKT conditions are used as a basis to derive many quadratic programming algorithms.

When considering a quadratic program, (6.1) subject to (6.2), the KKT conditions are given by:

$$\begin{aligned}
&\nabla_{\underset{\sim}{w}}\ell(\underset{\sim}{w}^*,\underset{\sim}{\lambda}^*) = \nabla_{\underset{\sim}{w}}\left(\frac{1}{2}\underset{\sim}{w}^{*T}\boldsymbol{G}\underset{\sim}{w}^* + \underset{\sim}{w}^{*T}\underset{\sim}{d} - \sum_{\mathrm{j}\in\mathrm{I}\cup\mathrm{E}}\lambda_{\mathrm{j}}^*\left(\underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w}^* - b_{\mathrm{j}}\right)\right)\\
&\nabla_{\underset{\sim}{w}}\ell(\underset{\sim}{w}^*,\underset{\sim}{\lambda}^*) = \boldsymbol{G}\underset{\sim}{w}^* + \underset{\sim}{d} - \sum_{\mathrm{j}\in\mathbb{A}(\underset{\sim}{w}^*)}\lambda_{\mathrm{j}}^*\underset{\sim}{a}_{\mathrm{j}} = 0\\
&\underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w}^* = b_{\mathrm{j}} \quad \text{for all } \mathrm{j}\in\mathbb{A}(\underset{\sim}{w}^*)\\
&\underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w}^* \geq b_{\mathrm{j}} \quad \text{for all } \mathrm{j}\in\mathrm{I}\setminus\mathbb{A}(\underset{\sim}{w}^*)\\
&\lambda_{\mathrm{j}}^* \geq 0 \quad \text{for all } \mathrm{j}\in\mathrm{I}\cap\mathbb{A}(\underset{\sim}{w}^*)
\end{aligned} \tag{6.7}$$

where $\mathbb{A}(\underset{\sim}{w}^*) = \{\mathrm{j}\in\mathrm{E}\cup\mathrm{I} : \underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w}^* = b_{\mathrm{j}}\}$. $\mathbb{A}(\underset{\sim}{w}^*)$ is called the optimal active set.

When considering a quadratic program with only equality constraints (an EP):

$$\min_{\underset{\sim}{w}}\ q(\underset{\sim}{w}) = \frac{1}{2}\underset{\sim}{w}^T\boldsymbol{G}\underset{\sim}{w} + \underset{\sim}{w}^T\underset{\sim}{d} \tag{6.8}$$

subject to:

$$\underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w} = b_{\mathrm{j}} \quad \mathrm{j}\in\mathrm{E} \tag{6.9}$$

the KKT conditions are given by:

$$\begin{aligned}
&\nabla_{\underset{\sim}{w}}\ell(\underset{\sim}{w}^*,\underset{\sim}{\lambda}^*) = \boldsymbol{G}\underset{\sim}{w}^* + \underset{\sim}{d} - \sum_{\mathrm{j}\in\mathrm{E}}\lambda_{\mathrm{j}}^*\underset{\sim}{a}_{\mathrm{j}} = 0\\
&\underset{\sim}{a}_{\mathrm{j}}^T\underset{\sim}{w}^* = b_{\mathrm{j}} \quad \text{for all } \mathrm{j}\in\mathrm{E}.
\end{aligned} \tag{6.10}$$

These conditions are more compactly expressed in matrix format:

$$\begin{bmatrix}\boldsymbol{G} & -\boldsymbol{A}^T\\ \boldsymbol{A} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\underset{\sim}{w}^*\\ \underset{\sim}{\lambda}^*\end{bmatrix} = \begin{bmatrix}-\underset{\sim}{d}\\ \underset{\sim}{b}\end{bmatrix} \tag{6.11}$$

where $\boldsymbol{A} = [\underset{\sim}{a}_{\mathrm{j}}]_{\mathrm{j}\in\mathrm{E}}^T$. The matrix in (6.11) is called the KKT matrix, and if it is non-singular, (6.11) is solved directly to give a local solution to the EP, $\underset{\sim}{w}^*, \underset{\sim}{\lambda}^*$. The solution is given by:

$$\begin{bmatrix} \underset{\sim}{w}^* \\ \underset{\sim}{\lambda}^* \end{bmatrix} = \begin{bmatrix} \boldsymbol{H} & \boldsymbol{T} \\ -\boldsymbol{T}^T & \boldsymbol{U} \end{bmatrix} \begin{bmatrix} -\underset{\sim}{d} \\ \underset{\sim}{b} \end{bmatrix}$$
$$\boldsymbol{H} = \boldsymbol{G}^{-1} - \boldsymbol{G}^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{G}^{-1}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{G}^{-1} \tag{6.12}$$
$$\boldsymbol{T} = \boldsymbol{G}^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{G}^{-1}\boldsymbol{A}^T)^{-1}$$
$$\boldsymbol{U} = (\boldsymbol{A}\boldsymbol{G}^{-1}\boldsymbol{A}^T)^{-1}.$$

Many algorithms exist to solve various types of quadratic programs. Some algorithms can only be used with convex problems, however some (for example active set methods) can be used with both convex and non-convex quadratic programs. The active set method is described in detail below.

6.1 Active Set Methods

Active set methods are powerful algorithms used to solve both convex and non-convex quadratic programs. An optimal solution to any quadratic program, $\underset{\sim}{w}^*$, will satisfy some (or none) of the inequality constraints as strict equalities. The set of inequality constraints satisfied as strict equalities at $\underset{\sim}{w}^*$ (plus the set of equality constraints) is referred to as the optimal active set. If the optimal active set is known or found, the quadratic program becomes an EP and is easily solved. Solving a quadratic program can therefore be treated as the problem of finding the optimal active set. Active set methods attempt to find the optimal active set via expanding or contracting an active set at each iteration. These methods are guaranteed to converge to a solution if the quadratic program is convex. Firstly, the method as applied to convex quadratic programs is described, and then the extension that makes the method applicable with non-convex quadratic programs is detailed (in section 6.1.2).

6.1.1 Active Set Methods for Convex Quadratic Programs

Let us assume that we wish to solve (6.1) subject to (6.2) where $\boldsymbol{G}$ is positive (semi-) definite. Assume that there is an initial feasible point $\underset{\sim}{w}^{(0)}$. The set of inequality constraints satisfied as strict equalities by $\underset{\sim}{w}^{(0)}$ and the set of equality constraints form the initial active set, $\mathcal{A}^{(0)}$.

Let us assume that $\underset{\sim}{w}^{(0)}$ does not minimize $q(\underset{\sim}{w})$ over $\mathcal{A}^{(0)}$ and that the minimizer over $\mathcal{A}^{(0)}$ is given by $\underset{\sim}{w}^{+}$:

$$\underset{\sim}{w}^{+} = \underset{\sim}{w}^{(0)} + \underset{\sim}{p}^{(0)} \tag{6.13}$$

where $\underset{\sim}{p}^{(0)}$ is a step from $\underset{\sim}{w}^{(0)}$. So we must find the minimizer $\underset{\sim}{w}^{+}$ by solving the EP:

$$\min_{\underset{\sim}{w}} \; q(\underset{\sim}{w}) = \frac{1}{2}\underset{\sim}{w}^{T}\boldsymbol{G}\underset{\sim}{w} + \underset{\sim}{w}^{T}\underset{\sim}{d} \tag{6.14}$$

subject to:

$$\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w} = b_{\mathrm{j}} \quad \text{for all } \mathrm{j} \in \mathcal{A}^{0}. \tag{6.15}$$

Using the KKT matrix representation on the EP we write:

$$\begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}^{T} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \underset{\sim}{w}^{+} \\ \underset{\sim}{\lambda}^{+} \end{bmatrix} = \begin{bmatrix} -\underset{\sim}{d} \\ \underset{\sim}{b} \end{bmatrix}. \tag{6.16}$$

If we consider $\underset{\sim}{w}^{+} = \underset{\sim}{w}^{(0)} + \underset{\sim}{p}^{(0)}$, (6.16) can be written as:

$$\begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}^{T} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \underset{\sim}{w}^{(0)} + \underset{\sim}{p}^{(0)} \\ \underset{\sim}{\lambda}^{+} \end{bmatrix} = \begin{bmatrix} -\underset{\sim}{d} \\ \underset{\sim}{b} \end{bmatrix} \tag{6.17}$$

and rearranging we obtain:

$$\begin{gathered} \begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}^{T} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \underset{\sim}{p}^{(0)} \\ \underset{\sim}{\lambda}^{+} \end{bmatrix} = \begin{bmatrix} \underset{\sim}{g}^{(0)} \\ \underset{\sim}{c}^{(0)} \end{bmatrix} \\ \underset{\sim}{g}^{(0)} = -\underset{\sim}{d} - \boldsymbol{G}\underset{\sim}{w}^{(0)} \\ \underset{\sim}{c}^{(0)} = \underset{\sim}{b} - \boldsymbol{A}\underset{\sim}{w}^{(0)} = \underset{\sim}{0}. \end{gathered} \tag{6.18}$$

The solution is easily found by (6.12). (Obviously, if $\underset{\sim}{p}^{(0)}$ is equal to the zero vector, $\underset{\sim}{w}^{(0)}$ minimizes $q(\underset{\sim}{w})$ over $\mathtt{A}^{(0)}$. The Lagrange multipliers, $\underset{\sim}{\lambda}^{+}$, indicate if $\underset{\sim}{w}^{(0)}$ is a KKT point.)

Now, $\underset{\sim}{w}^{+} = \underset{\sim}{w}^{(0)} + \underset{\sim}{p}^{(0)}$ will satisfy all the constraints within the active set $\mathtt{A}^{(0)}$ because of the constraint (6.15) enforced by the EP. If $\underset{\sim}{w}^{+}$ satisfies all the constraints not in $\mathtt{A}^{(0)}$ then we update $\underset{\sim}{w}^{(0)}$ to $\underset{\sim}{w}^{(1)} = \underset{\sim}{w}^{+}$. If $\underset{\sim}{w}^{+}$ does not satisfy all the constraints not in $\mathtt{A}^{(0)}$ then we update $\underset{\sim}{w}^{(0)}$ to $\underset{\sim}{w}^{(1)} = \underset{\sim}{w}^{(0)} + \alpha^{(0)}\underset{\sim}{p}^{(0)}$ where $\alpha^{(0)}$ is the maximum possible step length in the direction $\underset{\sim}{p}^{(0)}$ for which all constraints remain satisfied. The step length $\alpha^{(0)}$ can be calculated by considering the inequality constraints not satisfied by $\underset{\sim}{w}^{+}$. If a constraint j is not satisfied by $\underset{\sim}{w}^{+}$ then:

$$\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w}^{+} \le b_{\mathrm{j}} \Rightarrow \underset{\sim}{a}_{\mathrm{j}}^{T}(\underset{\sim}{w}^{(0)} + \underset{\sim}{p}^{(0)}) \le b_{\mathrm{j}} \Rightarrow \underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{p}^{(0)} \le b_{\mathrm{j}} - \underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w}^{(0)} \Rightarrow \underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{p}^{(0)} < 0. \tag{6.19}$$

To ensure that these constraints remain satisfied we must choose a step length to enforce:

$$\underset{\sim}{a}_{\mathrm{j}}^{T}(\underset{\sim}{w}^{(0)} + \alpha^{(0)}\underset{\sim}{p}^{(0)}) \ge b_{\mathrm{j}}. \tag{6.20}$$

So:

$$\begin{aligned} &\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w}^{(0)} + \alpha^{(0)}\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{p}^{(0)} \ge b_{\mathrm{j}} \\ &\alpha^{(0)}\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{p}^{(0)} \ge b_{\mathrm{j}} - \underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w}^{(0)} \\ &\alpha^{(0)} \le \frac{b_{\mathrm{j}} - \underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{w}^{(0)}}{\underset{\sim}{a}_{\mathrm{j}}^{T}\underset{\sim}{p}^{(0)}}. \end{aligned} \tag{6.21}$$

The value $\alpha^{(0)}$ must be calculated for all constraints not satisfied by $\underset{\sim}{w}^{+}$, to find the minimum $\alpha^{(0)}$. This becomes the step length. Explicitly, $\alpha^{(0)}$ is given by:

$$\alpha^{(0)} = \min\left(1, \min_{j \notin \mathcal{A}^{(0)}, \underset{\sim}{a}_j^T \underset{\sim}{p}^{(0)} < 0} \frac{b_j - \underset{\sim}{a}_j^T \underset{\sim}{w}^{(0)}}{\underset{\sim}{a}_j^T \underset{\sim}{p}^{(0)}}\right). \tag{6.22}$$

(If $\alpha^{(0)}$ is 1, then all constraints were satisfied by $\underset{\sim}{w}^+$.) If $\alpha^{(0)}$ is less than 1, the constraint corresponding to the minimum $\alpha^{(0)}$ is added to the active set to give $\mathcal{A}^{(1)}$.

The active set method now iterates:

For $\zeta = 1, 2, ...$

1) Find $\underset{\sim}{p}^{(\zeta)}$ by solving:

$$\begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \underset{\sim}{p}^{(\zeta)} \\ \underset{\sim}{\lambda}^{(\zeta)} \end{bmatrix} = \begin{bmatrix} \underset{\sim}{g}^{(\zeta)} \\ \underset{\sim}{0} \end{bmatrix} \tag{6.23}$$

where $\underset{\sim}{g}^{(\zeta)} = -\underset{\sim}{d} - \boldsymbol{G}\underset{\sim}{w}^{(\zeta)}$. If $\underset{\sim}{p}^{(\zeta)} \neq \underset{\sim}{0}$, check if $\underset{\sim}{w}^{(\zeta)} + \underset{\sim}{p}^{(\zeta)}$ satisfies all constraints not in $\mathcal{A}^{(\zeta)}$. If so, go to 2a. If not, go to 2b. If $\underset{\sim}{p}^{(\zeta)} = \underset{\sim}{0}$, go to 2c.

2)

a) Set $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)} + \underset{\sim}{p}^{(\zeta)}$ and $\mathcal{A}^{(\zeta+1)} = \mathcal{A}^{(\zeta)}$. Return to 1.

b) Then $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)} + \alpha^{(\zeta)}\underset{\sim}{p}^{(\zeta)}$ where $\alpha^{(\zeta)}$ is given by (6.22) with $\mathcal{A}^{(0)}$ replaced by $\mathcal{A}^{(\zeta)}$, $\underset{\sim}{w}^{(0)}$ replaced by $\underset{\sim}{w}^{(\zeta)}$; and $\underset{\sim}{p}^{(0)}$ replaced by $\underset{\sim}{p}^{(\zeta)}$. Add constraint corresponding to $\alpha^{(\zeta)}$ to $\mathcal{A}^{(\zeta)}$ to give $\mathcal{A}^{(\zeta+1)}$. Return to 1.

The active set method continues to iterate until $\underset{\sim}{w}^{(\zeta)}$ minimizes $q(\underset{\sim}{w})$ over $\mathcal{A}^{(\zeta)}$. At this point $\underset{\sim}{p}^{(\zeta)}$ is the zero vector. The KKT conditions (6.7) are considered to ascertain if $\underset{\sim}{w}^{(\zeta)}$ is a KKT point of the original problem. Consideration of (6.18) shows the first KKT condition to be satisfied. The second and third conditions are obviously satisfied. If the fourth KKT condition is satisfied then we conclude $\underset{\sim}{w}^{(\zeta)}$ is a KKT point and must be a global minimum of the quadratic program because $\boldsymbol{G}$ is positive (semi-) definite. The optimal active set is $\mathbb{A}^{(\zeta)}$. If however, the $\lambda_{\mathrm{j}}^{(\zeta)}$ are not all greater than or equal to zero, a local minimum has not yet been reached. A $\lambda_{\mathrm{j}}^{(\zeta)}$ with a negative value, implies that $q(\underset{\sim}{w})$ can be decreased by removing the corresponding constraint from $\mathbb{A}^{(\zeta)}$. In the situation of multiple Lagrange multipliers with negative values, the constraint with the most negative Lagrange multiplier is removed from $\mathbb{A}^{(\zeta)}$ to give $\mathbb{A}^{(\zeta+1)}$. The active set method then checks if $\underset{\sim}{w}^{(\zeta+1)}$ (where $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)}$) minimizes $q(\underset{\sim}{w})$ over $\mathbb{A}^{(\zeta+1)}$ and the iterations continue.

The algorithm for the active set method now reads in its entirety for step 2:

c) Calculate $\underset{\sim}{\lambda}^{(\zeta)}$ via:

$$\begin{bmatrix} \underset{\sim}{p}^{(\zeta)} \\ \underset{\sim}{\lambda}^{(\zeta)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}^{T} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix}^{-1} \begin{bmatrix} \underset{\sim}{g}^{(\zeta)} \\ \underset{\sim}{0} \end{bmatrix} \tag{6.24}$$

where the inverse of the KKT matrix is given by (6.12). If all $\underset{\sim}{\lambda}^{(\zeta)} \geq 0$ STOP. If $\lambda_{\mathrm{j}}^{(\zeta)} < 0$ for some j, remove the constraint corresponding to the most negative $\lambda_{\mathrm{j}}^{(\zeta)}$ from $\mathbb{A}^{(\zeta)}$ to give $\mathbb{A}^{(\zeta+1)}$, set $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)}$. Return to 1.

6.1.2 Active Set Methods for Non-Convex Quadratic Programs

It is known that if the reduced Hessian matrix, $\boldsymbol{Z}^{T}\boldsymbol{G}\boldsymbol{Z}$ (where $\boldsymbol{Z}$ is a matrix whose columns form the basis of the null space of $\boldsymbol{A}$ such that $\boldsymbol{A}\boldsymbol{Z} = \boldsymbol{0}$) is positive definite, then the step direction $\underset{\sim}{p}^{(\zeta)}$ at any iteration will point in the direction of the minimum. Therefore, in the case of convex quadratic programs, the positive definiteness of $\boldsymbol{G}$ enforces the positive definiteness of $\boldsymbol{Z}^{T}\boldsymbol{G}\boldsymbol{Z}$, and $\underset{\sim}{p}^{(\zeta)}$ is guaranteed to point towards the minimum. However, when considering a non-convex quadratic program, $\boldsymbol{Z}^T\boldsymbol{GZ}$ is not necessarily positive definite. If the reduced Hessian is not positive definite, the $\underset{\sim}{p}^{(\zeta)}$ points toward a saddle point. In this situation, an amendment of the active set method which determines a direction of negative curvature, $\underset{\sim}{s}^{(\zeta)}$ is required.

The indefiniteness of the reduced Hessian is easily detected using a $\boldsymbol{LDL}^T$ factorization. If any of the elements of the diagonal matrix $\boldsymbol{D}$ are negative, then $\boldsymbol{Z}^T\boldsymbol{GZ}$ is not positive definite. The $\boldsymbol{LDL}^T$ factorization is then further utilized to calculate a direction of negative curvature.

Let us assume that the quadratic program to be solved is given by (6.1) subject to (6.2) where $\boldsymbol{G}$ is indefinite. Let us also assume that we have a current solution to the quadratic program, $\underset{\sim}{w}^{(\zeta)}$ which minimizes $q(\underset{\sim}{w})$ over $\mathrm{A}^{(\zeta)}$ and that $\boldsymbol{Z}^{(\zeta)^T}\boldsymbol{GZ}^{(\zeta)}$ is positive definite. The Lagrange multipliers, $\underset{\sim}{\lambda}^{(\zeta)}$, are then calculated. If all $\underset{\sim}{\lambda}^{(\zeta)} \geq 0$, the algorithm will terminate. If however, $\lambda_{\mathrm{j}}^{(\zeta)} < 0$ for various j, let δ be the index of the multiplier with the most negative value. Constraint δ should be removed from the current active set. Hypothetical removal of constraint δ from $\mathrm{A}^{(\zeta)}$ will increase the dimension of $\boldsymbol{Z}^{(\zeta)}$ (where $\boldsymbol{A}^{(\zeta)}\boldsymbol{Z}^{(\zeta)} = \boldsymbol{0}$) by one column, to give $\boldsymbol{Z}^{(\zeta^*)}, \boldsymbol{Z}^{(\zeta^*)} = [\boldsymbol{Z}^{(\zeta)} \mid \underset{\sim}{z}]$.
$\boldsymbol{Z}^{(\zeta^*)^T}\boldsymbol{GZ}^{(\zeta^*)}$ is factored into $\boldsymbol{L}^{(\zeta^*)}\boldsymbol{D}^{(\zeta^*)}\boldsymbol{L}^{(\zeta^*)^T}$ where $\boldsymbol{L}^{(\zeta^*)} = \begin{bmatrix} \boldsymbol{L}^{(\zeta)} & \underset{\sim}{0} \\ \underset{\sim}{l}^T & 1 \end{bmatrix}$ and

$\boldsymbol{D}^{(\zeta^*)} = \begin{bmatrix} \boldsymbol{D}^{(\zeta)} & \underset{\sim}{0} \\ \underset{\sim}{0} & \mathrm{d}_{\mathrm{K-t+1}} \end{bmatrix}$ for some vector $\underset{\sim}{l}$ and element $\mathrm{d}_{\mathrm{K-t+1}}$ (where t is the number of constraints in $\mathrm{A}^{(\zeta)}$). If $d_{\mathrm{K-t+1}}$ is positive, then $\boldsymbol{Z}^{(\zeta^*)^T}\boldsymbol{GZ}^{(\zeta^*)}$ remains positive definite, and we can remove constraint δ from $\mathrm{A}^{(\zeta)}$ to give $\mathrm{A}^{(\zeta+1)}$. If $\mathrm{d}_{\mathrm{K-t+1}}$ is negative, $\boldsymbol{Z}^{(\zeta^*)^T}\boldsymbol{GZ}^{(\zeta^*)}$

is not positive definite and we cannot use the direction given by $\underset{\sim}{p}^{(\zeta+1)}$ because it does not point in the direction of the minimum. Instead a direction of negative curvature is calculated via:

$$\underset{\sim}{s}^{(\zeta)} = \boldsymbol{Z}^{(\zeta*)} \underset{\sim}{s}_z^{(\zeta)} \tag{6.25}$$

where:

$$\boldsymbol{L}^{(\zeta*)^T} \underset{\sim}{s}_z^{(\zeta)} = \underset{\sim}{e}_{\mathrm{K-t+1}}. \tag{6.26}$$

It is easily verified that $\underset{\sim}{s}^{(\zeta)}$ points in a direction of negative curvature because:

$$\underset{\sim}{s}^{(\zeta)^T} \nabla^2 q \underset{\sim}{s}^{(\zeta)} = \underset{\sim}{s}^{(\zeta)^T} \boldsymbol{G} \underset{\sim}{s}^{(\zeta)} = \underset{\sim}{s}_z^{(\zeta)^T} \boldsymbol{Z}^{(\zeta*)^T} \boldsymbol{G} \boldsymbol{Z}^{(\zeta*)^T} \underset{\sim}{s}_z^{(\zeta)} = \underset{\sim}{s}_z^{(\zeta)^T} \boldsymbol{L}^{(\zeta*)} \boldsymbol{D}^{(\zeta*)} \boldsymbol{L}^{(\zeta*)^T} \underset{\sim}{s}_z^{(\zeta)} = \underset{\sim}{e}_{\mathrm{K-t+1}}^T \boldsymbol{D}^{(\zeta*)} \underset{\sim}{e}_{\mathrm{K-t+1}} = \mathrm{d}_{\mathrm{K-t+1}} < 0. \tag{6.27}$$

The next iterate $\underset{\sim}{w}^{(\zeta+1)}$ is given by $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)} + \alpha^{(\zeta)} \underset{\sim}{s}^{(\zeta)}$ where $\alpha^{(\zeta)}$ is calculated by:

$$\alpha^{(\zeta)} = \min_{\mathrm{j} \notin \mathrm{A}^{(\zeta)}, \underset{\sim}{a}_{\mathrm{j}}^T \underset{\sim}{s}^{(\zeta)} < 0} \frac{b_{\mathrm{j}} - \underset{\sim}{a}_{\mathrm{j}}^T \underset{\sim}{w}^{(\zeta)}}{\underset{\sim}{a}_{\mathrm{j}}^T \underset{\sim}{s}^{(\zeta)}}. \tag{6.28}$$

The step length $\alpha^{(\zeta)}$ indicates how far we move along $\underset{\sim}{s}^{(\zeta)}$ before we encounter a constraint, j. This constraint is added to the active set to give $\mathrm{A}^{(\zeta+1)}$. Note at this point constraint δ still remains in $\mathrm{A}^{(\zeta)}$ as a psuedoconstraint. Its hypothetical removal induced $\boldsymbol{Z}^{(\zeta*)}$ and $\underset{\sim}{s}^{(\zeta)}$.

After the addition of constraint j to $\mathrm{A}^{(\zeta)}$ (to give $\mathrm{A}^{(\zeta+1)}$) the (hypothetical) removal of constraint δ is considered. If $\boldsymbol{Z}^{(\zeta+1*)^T} \boldsymbol{G} \boldsymbol{Z}^{(\zeta+1*)}$ remains indefinite, then constraint δ is not removed, however if $\boldsymbol{Z}^{(\zeta+1*)^T} \boldsymbol{G} \boldsymbol{Z}^{(\zeta+1*)}$ becomes positive definite, then constraint δ is actually removed from $\mathrm{A}^{(\zeta+1)}$. The search for a minimizing direction $\underset{\sim}{p}^{(\zeta+1)}$ is resumed. If the hypothetical removal of constraint δ implies $\boldsymbol{Z}^{(\zeta+1*)^T} \boldsymbol{G} \boldsymbol{Z}^{(\zeta+1*)}$ is indefinite, then δ remains in $\mathrm{A}^{(\zeta+1)}$ as a psuedoconstraint. That is, we use constraint δ's hypothetical removal to calculate $\boldsymbol{Z}^{(\zeta+1*)}, \boldsymbol{D}^{(\zeta+1*)}, \boldsymbol{L}^{(\zeta+1*)}$ and a subsequent $\underset{\sim}{s}^{(\zeta+1)}$, but do not actually remove it from the active set $\mathcal{A}^{(\zeta+1)}$. We continue to iterate in the fashion, until eventually, the addition of a constraint j will allow the removal of δ from the current active set. After δ has been removed, the search for a minimizing direction $\underset{\sim}{p}$ is resumed.

The algorithm for non-convex quadratic programs is therefore very similar to the algorithm for convex quadratic programs (assuming that $\boldsymbol{Z}^{(0)^T}\boldsymbol{G}\boldsymbol{Z}^{(0)}$ is positive definite). However step 2 now reads:

c) Calculate $\underset{\sim}{\lambda}^{(\zeta)}$ via (6.24) where the inverse of the KKT matrix is given by (6.12). If all $\underset{\sim}{\lambda}^{(\zeta)} \geq 0$ STOP. If $\lambda_{\mathrm{j}}^{(\zeta)} < 0$ for some j, find the constraint with the most negative multiplier, δ. Calculate the matrix $\boldsymbol{Z}^{(\zeta*)}$ where $\boldsymbol{A}^{(\zeta*)}\boldsymbol{Z}^{(\zeta*)} = \boldsymbol{0}$ and $\boldsymbol{A}^{(\zeta*)}$ is the matrix of constraints from $\mathcal{A}^{(\zeta)}$ but not including δ. If $\boldsymbol{Z}^{(\zeta*)^T}\boldsymbol{G}\boldsymbol{Z}^{(\zeta*)}$ is positive definite (checked easily via the diagonal elements of $\boldsymbol{D}^{(\zeta*)}$ where $\boldsymbol{Z}^{(\zeta*)^T}\boldsymbol{G}\boldsymbol{Z}^{(\zeta*)} = \boldsymbol{L}^{(\zeta*)}\boldsymbol{D}^{(\zeta*)}\boldsymbol{L}^{(\zeta*)^T}$) remove δ from $\mathcal{A}^{(\zeta)}$ to give $\mathcal{A}^{(\zeta+1)}$, set $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)}$, and return to 1. Otherwise go to step 2d.

d) Hypothetically remove constraint δ from $\mathcal{A}^{(\zeta)}$ to give $\mathcal{A}^{(\zeta*)}$. Calculate $\underset{\sim}{s}^{(\zeta)}, \alpha^{(\zeta)}$ via equations (6.25), (6.26), and (6.28). Calculate $\underset{\sim}{w}^{(\zeta+1)} = \underset{\sim}{w}^{(\zeta)} + \alpha^{(\zeta)}\underset{\sim}{s}^{(\zeta)}$. Add constraint j to $\mathcal{A}^{(\zeta)}$ to give $\mathcal{A}^{(\zeta+1)}$. Hypothetically remove δ from $\mathcal{A}^{(\zeta+1)}$ to give $\mathcal{A}^{(\zeta+1*)}$. If $\boldsymbol{Z}^{(\zeta+1*)^T}\boldsymbol{G}\boldsymbol{Z}^{(\zeta+1*)}$ is positive definite, actually remove δ from $\mathcal{A}^{(\zeta+1)}$ and return to 1. If not, repeat step 2d, until δ can be removed.

6.2 Using Quadratic Programming to Calculate Weight Coefficients

The problem of finding weight coefficients can easily be formulated as a quadratic program. We wish to minimize the loss of using the weighted sum of models to predict the response. If the loss is the squared error loss function, the problem becomes:

$$\min_{\underset{\sim}{\omega}} \quad \mathrm{q}(\underset{\sim}{\omega}) = (\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega})^T (\underset{\sim}{y} - \hat{\boldsymbol{Y}}\underset{\sim}{\omega}) \tag{6.29}$$

subject to:

$$\begin{aligned} &\omega_{\mathrm{m}} \geq 0 \quad \text{for all m} \\ &\sum_{\mathrm{m}=1}^{\mathrm{M}} \omega_{\mathrm{m}} = 1. \end{aligned} \tag{6.30}$$

This is easily converted to a quadratic program:

$$\min_{\underset{\sim}{\omega}} \quad q(\underset{\sim}{\omega}) = \underset{\sim}{\omega}^T \boldsymbol{G} \underset{\sim}{\omega} + \underset{\sim}{\omega}^T \underset{\sim}{d} \tag{6.31}$$

subject to (6.30); where $\boldsymbol{G} = \hat{\boldsymbol{Y}}^T \hat{\boldsymbol{Y}}$ and $\underset{\sim}{d} = -2\hat{\boldsymbol{Y}}^T \underset{\sim}{y}$.

The active set method is employed to solve the quadratic program. By formulating the problem as a quadratic program, the loss function is not restricted to the squared error loss. A number of other quadratic programs were formulated to calculate the weight coefficients and are detailed in the manuscript “Post processing regression ensembles: imposing parsimony to improve predictions”.

7 K-MEANS

K-means was used as a benchmark method throughout this thesis, and was later modified to engender the creation of similarity-based k-means. K-means clustering is a divisive clustering technique. K-means seeks clusters that minimize their within group sums of squares. K-means attempts to minimize the objective function:

$$\sum_{r=1}^{K} \sum_{\underset{\sim}{x}_i \in S_r} (\underset{\sim}{x}_i - \overline{\underset{\sim}{x}}_{S_r})^T (\underset{\sim}{x}_i - \overline{\underset{\sim}{x}}_{S_r}) \tag{7.1}$$

where S_r includes all the observational units in cluster r and $\overline{\underset{\sim}{x}}_{S_r}$ is the mean of observational units in cluster r.

The optimization of the objective function can be performed by a "hill climbing" algorithm (Everitt 1993). A hill climbing algorithm moves observational units between clusters, and keeps solutions if and only if they improve the value of the objective function. Applied to k-means, 'hill climbing' results in the following algorithm:

1) Choose the number of clusters and initial estimates of the cluster centers.
2) Visit each observational unit and assign it to the cluster whose centroid is closest using the squared Euclidean distance $dist^2(\underset{\sim}{x}_i, \overline{\underset{\sim}{x}}_{S_r})$:

   $$dist^2(\underset{\sim}{x}_i, \overline{\underset{\sim}{x}}_{S_r}) = (\underset{\sim}{x}_i - \overline{\underset{\sim}{x}}_{S_r})^T (\underset{\sim}{x}_i - \overline{\underset{\sim}{x}}_{S_r}). \tag{7.2}$$

   Recalculate the centroids of the clusters the observational unit has left and joined.
3) Repeat step 2 until no more reassignments of the observational units take place.

Step 1 requires not only the number of clusters but also initial estimates of the cluster centers to be specified in advance. These estimates can be obtained in various ways. The observational units can (randomly) be divided into groups and the mean vector of each group calculated. Alternatively, the centers can be random seed points. The initial estimates can dramatically affect the final clustering solution, with some starts steering the solution to only a local optimum.

8 SIMILARITY-BASED K-MEANS

The traditional k-means algorithm was modified in this thesis to create similarity-based k-means (SBK). SBK is designed to cluster a similarity matrix. Formally, SBK seeks clusters to minimize either of the objective functions:

$$\min \sum_{r=1}^{K} \sum_{i,j \in S_r} \left(C_{i,j} - \overline{C}_r \right)^2 + \sum_{r=1}^{K} \sum_{\substack{r' \neq r \\ r'=1}}^{K} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left(C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right)^2 \tag{8.1}$$

or:

$$\min \sum_{r=1}^{K} \sum_{i,j \in S_r} \left| C_{i,j} - \overline{C}_r \right| + \sum_{r=1}^{K} \sum_{\substack{r' \neq r \\ r'=1}}^{K} \sum_{\substack{i \in S_r \\ j \in S_{r'}}} \left| C_{i,j} - \overline{COV}_{(S_r, S_{r'})} \right| \tag{8.2}$$

where S_r is the set of observational units in the r^{th} cluster; $C_{i,j}$ is the $(i,j)^{th}$ element of the co-occurrence matrix; $\overline{C}_r$ is the mean similarity of the r^{th} cluster; and $\overline{COV}_{(S_r, S_{r'})}$ is the mean similarity of the covariance matrix where the rows are given by the observational units in cluster r and the columns are given by the observational units in cluster $r^{'}$.

Because of the mean squared error and absolute error terms in the objective functions, SBK can be viewed almost entirely in the prediction sense. However, a validity criterion (Hancock 2006) is imposed to ensure that the clustering ideology prevails over the prediction ideology. The validity criterion is simply:

$$\overline{C}_r \,\&\, \overline{C}_{r'} > \overline{COV}_{(S_r, S_{r'})}. \tag{8.3}$$

This validity criterion ensures that an observational unit is only placed in a cluster if it will increase the mean similarity of the cluster whilst simultaneously decreasing the mean similarity of the corresponding covariance submatrix. Thus, clusters must have higher mean similarities than their covariance.

Identical to traditional k-means, the algorithm is implemented via a “hill climbing” approach. The SBK algorithm is given by:

1) Choose the number of clusters and an initial partition of the data. Here, we use initial partitions given by both hierarchical clustering of the co-occurrence matrix and entirely random partitions. Choose the objective function: either the mean squared error (8.1) or absolute error (8.2).
2) Visit each observational unit and assign it to the cluster which will result in the largest decrease of the objective function. Before moving the observational unit ensure that the validity criterion is upheld.
3) Update the mean similarity of:
    a) the cluster the observational unit has left
    b) the cluster the observational unit has joined
    c) and all appropriate covariance means.
4) Repeat steps two and three until no more reassignments of the observational units take place.

9 DETERMINING THE NATURAL NUMBER OF CLUSTERS

Similarity-based k-means allows for an estimate of the number of clusters in the dataset. This estimate considers the average predictive capability of the algorithm, for any number of clusters, k. In theory, the estimate closely resembles the figure of merit method proposed by Yeung, et al. (2001). This section describes the figure of merit methodology. The similarity between the figure of merit methodology and the technique used in conjunction with SBK is also described.

9.1 Figures of Merit

A figure of merit (FOM) assesses the "predictive power" of a clustering algorithm by leaving out a variable, p, clustering the data (into k clusters), then calculating the root mean square error (RMSE) of p relative to the cluster means, $\mathrm{RMSE}(p,k)$:

$$\mathrm{RMSE}(p,k) = \sqrt{\frac{1}{n}\sum_{r=1}^{k}\sum_{\underset{\sim}{x}_i \in S_r}\left(x_{ip} - \bar{x}_r(p)\right)^2} \tag{9.1}$$

where x_{ip} is the measurement of the p^{th} variable on the i^{th} observational unit; n is the number of observational units; S_r is the set of observational units in the r^{th} cluster; and $\bar{x}_r(p)$ is the mean of variable p for the observational units in the r^{th} cluster. Each variable is omitted and its RMSE calculated. These RMSE are summed over all variables to give an aggregate FOM (AFOM):

$$\mathrm{AFOM}(k) = \sum_{p=1}^{P}\mathrm{RMSE}(p,k). \tag{9.2}$$

Obviously, low values of a clustering algorithm's AFOM indicate that the algorithm has high predictive power (Yeung, et al. 2001).

The AFOM is calculated for each k, and adjusted for cluster size. The reasoning behind adjusting the aggregate FOM for cluster size is simple. As we increase the number of clusters, the AFOM will artificially decrease because the clusters are smaller and naturally will not be as spread out. The adjusted AFOM mitigates the artificial decrease effect, by dividing the aggregate figure of merit by $\sqrt{\frac{n-k}{n}}$.

Yeung, et al. (2001) detail the reasoning behind the factor $\sqrt{\frac{n-k}{n}}$. An idealized model is assumed where the n observational units fall into χ real classes, and the variables of each class are normally distributed with mean $\bar{\mu}_r(p)$ and variance $\sigma^2_{r,p}$ $r=1,...,\chi$. Assume all of one class is clustered together. The expected value of $\sum_{i=1}^{\xi_r n}\left(x_{ip} - \sum_{i=1}^{\xi_r n} x_{ip} \Big/ \xi_r n \right)^2$ is $(\xi_r n - 1)\sigma^2_{r,p}$ where ξ_r is the proportion of observational units in class r. If the class is split up into $\xi_r k$ smaller clusters then the expected value of $\sum_{i=1}^{\xi_r n}\left(x_{ip} - \sum_{i=1}^{\xi_r n} x_{ip} \Big/ \xi_r n \right)^2$ reduces to $(\xi_r n - \xi_r k)\sigma^2_{r,p}$. Therefore:

$$\begin{aligned} \mathrm{AFOM}(k) &= \sum_{p=1}^{P}\sqrt{\frac{1}{n}\sum_{r=1}^{\chi}\left(\xi_r n - \xi_r k\right)\sigma^2_{r,p}} \\ &= \sum_{p=1}^{P}\sqrt{\frac{1}{n}\sum_{r=1}^{\chi}(n-k)\xi_r\sigma^2_{r,p}} \\ &= \sqrt{\frac{n-k}{n}}\sum_{p=1}^{P}\sqrt{\sum_{r=1}^{\chi}\xi_r\sigma^2_{r,p}}. \end{aligned} \tag{9.3}$$

Thus, the adjusted figure of merit is given by: $\mathrm{AFOM}_{adj}(k) = \dfrac{\mathrm{AFOM}(k)}{\sqrt{\frac{n-k}{n}}}$.

9.2 Extending Figures of Merit

It is simple to expand the above adjusted FOM theory to the results obtained by the SBK algorithm. If the dataset is clustered to k clusters, and this process is repeated P times, then the AFOM is defined as:

$$\mathrm{AFOM}(k) = \sum_{p=1}^{P}\sqrt{\frac{1}{n^2}\sum_{r=1}^{k}\sum_{i,j\in S_r(p)}\left(C_{i,j} - \bar{C}_r(p)\right)^2} \tag{9.4}$$

where $S_r(p)$ is the set of observational units in cluster r on the p^{th} run; $\bar{C}_r(p)$ is the mean similarity of observational units in cluster r on the p^{th} run; $C_{i,j}$ is $(i,j)^{th}$ element of the co-occurrence matrix; and n^2 is the dimension of the similarity matrix.

Following the same theory described above, here the adjusted figure of merit is given by:

$$\mathrm{AFOM}_{adj}(k) = \frac{\mathrm{AFOM}(k)}{P\sqrt{\frac{n^2 - k}{n^2}}}. \tag{9.5}$$

We also incorporate P into the adjustment factor to find the mean adjusted figure of merit. The AFOM_{adj} is obtained for varying levels of k, and the smallest AFOM_{adj} indicates the number of clusters. For the sake of parsimony, the elbow of the AFOM_{adj} curve is selected as the optimal number of clusters.

REFERENCES

Acuna, E., and Rodriguez, C., "*dprep: Data preprocessing and visualization functions for classification. R package version 1.0*," R Manual.

Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. G., Sabet, H., Tran, T., Yu, X., Powell, J. I., Yang, L. M., Marti, G. E., Moore, T., Hudson, J., Lu, L. S., Lewis, D. B., Tibshirani, R., Sherlock, G., Chan, W. C., Greiner, T. C., Weisenburger, D. D., Armitage, J. O., Warnke, R., Levy, R., Wilson, W., Grever, M. R., Byrd, J. C., Botstein, D., Brown, P. O., and Staudt, L. M. (2000), "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature*, 403, 503-511.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. (1999), "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences of the United States of America*, 96, 6745-6750.

Al-Razgan, M., and Domeniconi, C. (2006), "Weighted Clustering Ensembles," in *Siam Conference on Data Mining*.

Baker, S. G., and Kramer, B. S. (2006), "Identifying genes that contribute most to good classification in microarrays," *BMC Bioinformatics*, 7.

Ben-Hur, A., Elisseeff, A., and Guyon, I. (2002), "A stability based method for discovering structure in clustered data," in *the Pacific Symposium on Biocomputing*, pp. 6-17.

Ben-Hur, A., and Guyon, I. (2003), "Detecting stable clusters using principal component analysis," in *Methods in Molecular Biology*, eds. M. J. Brownstein and A. Khodursky, New Jersey: Humana press, pp. 159-182.

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999), "When Is "Nearest Neighbor" Meaningful," *Lecture Notes in Computer Science*, 1540, 217-235.

Borggaard, C., and Thodberg, H. H. (1992), "Optimal Minimal Neural Interpretation of Spectra," *Analytical Chemistry*, 64, 545-551.

Breiman, L. (2001), "Random Forests," *Machine Learning*, 45, 5-32.

Breiman, L., Friedman, J., Olshen, R. A., and Stone, C. J. (1984), *Classification and Regression Trees*, Belmont, CA: Wadsworth.

Breiman, L., and Friedman, J. H. (1985), "Estimating Optimal Transformations for Multiple Regression and Correlation," *Journal of the American Statistical Association*, 80, 580-598.

Cattell, R. B. (1966), "The scree test for the number of factors," *Multivariate Behavioral Research*, 1, 245-276.

Coomans, D., Jonckheer, M., Massart, D. L., Broeckaert, I., and Block, P. (1978), "The application of linear discriminant analysis in the diagnosis of thyroid diseases," *Analytica Chimica Acta*, 103, 409-415.

Datta, S., and Datta, S. (2006), "Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes," *BMC Bioinformatics*, 7.

Davis, L. (1991), *Handbook of genetic algorithms*, New York: Van Nostrand Reinhold.

De'Ath, G. (1999), "*New Statistical Methods for Modelling Species-Environment Relationships*," PhD, James Cook University, School of Mathematical and Physical Sciences.

De'Ath, G. (2002), "Multivariate regression trees: a new technique for modeling species-environment relationships," *Ecology*, 83, 1105-1117.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 39, 1-38.

Do, K.-A., McLachlan, G. J., Bean, R. W., and Wen, S. (2002), "A comparison of two methods of clustering gene microarray data," Technical Report, University of Queensland, Centre for Statistics.

Dudoit, S., and Fridlyand, J. (2002), "A prediction-based resampling method for estimating the number of clusters in a dataset," *Genome Biology*, 3, 0036.0031-0036.0021.

Dudoit, S., and Fridlyand, J. (2003), "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics*, 19, 1090-1099.

Duggan, B. J., McKnight, J. J., Williamson, K. E., Loughrey, M., O'Rourke, D., Hamilton, P. W., Johnston, S. R., Schulman, C. C., and Zlotta, A. R. (2003), "The Need to Embrace Molecular Profiling of Tumor Cells in Prostate and Bladder Cancer," *Clinical Cancer Research*, 9, 1240-1247.

Dunteman, G. H. (1989), *Principal Components Analysis*, ed. M. S. Lewis-Beck, Newbury Park, CA: Sage.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004), "Least angle regression," *Annals of Statistics*, 32, 407-499.

Everitt, B. (1993), *Cluster Analysis* (3rd ed.), London: Edward Arnold.

Fern, X. Z., and Brodley, C. E. (2006), "Cluster ensembles for high dimensional data clustering: An empirical study," Technical Report CS06-30-02, Oregon State University, Department of Computer Science.

Fisher, R. A. (1936), "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, 7, 179-188.

Fletcher, R. (1987), *Practical Methods of Optimization* (2nd ed.), Chichester: John Wiley & Sons.

Fred, A. L. N., and Jain, A. K. (2002), "Data Clustering Using Evidence Accumulation," in *16th International Conference on Pattern Recognition*.

Fred, A. L. N., and Jain, A. K. (2003), "Robust Data Clustering," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Friedman, J. (1991), "Multivariate Adaptive Regression Splines," *The Annals of Statistics*, 19, 1-67.

Friedman, J., and Popescu, B. (2003), "Importance Sampled Learning Ensembles," Technical Report, Stanford University, Department of Statistics.

Friedman, J., and Popescu, B. (2004), "Gradient directed regularization for linear regression and classification," Technical Report, Stanford University, Department of Statistics.

Frossyniotis, D. S., Likas, A., and Stafylopatis, A. (2004), "A clustering method based on boosting," *Pattern Recognition Letters*, 25, 641-654.

Frossyniotis, D. S., Pertselakis, M., and Stafylopatis, A. (2002), "A Multi-clustering Fusion Algorithm," in *2nd Hellenic Conference on Artificial Intelligence*, pp. 225-236.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (1995), *Bayesian Data Analysis* (1st ed.), London: Chapman & Hall.

Ghahramani, Z., and Hinton, G. (1996), "The EM Algorithm for Mixtures of Factor Analyzers," Technical Report CRG-TR-96-1, University of Toronto.

Ghosh, J., Strehl, A., and Merugu, S. (2002), "A consensus framework for integrating distributed clusterings under limited knowledge sharing," in *NSF Workshop on Next Generation Data Mining*.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999), "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science*, 286, 531-537.

Greene, D., Tsymbal, A., Bolshakova, N., and Cunningham, P. (2004), "Ensemble Clustering in Medical Diagnostics," in *17th IEEE Symposium on Computer-Based Medical Systems*.

Grotkjaer, T., Winther, O., Regenberg, B., and Nielsen, J. (2006), "Robust multi-scale clustering of large DNA microarray datasets with the consensus algorithm," *Bioinformatics*, 22, 58-67.

Hadjitodorov, S. T., Kuncheva, L. I., and Todorova, L. P. (2006), "Moderate diversity for better cluster ensembles," *Information Fusion*, 7, 264-275.

Hancock, T. (2006), "*Multivariate Consensus Trees: Tree-based clustering and profiling for mixed data types*," PhD, James Cook University, School of Mathematical and Physical Sciences.

Hancock, T., Put, R., Coomans, D., Vander-Heyden, Y., and Everingham, Y. (2005), "A performance comparison of modern statistical techniques for molecular descriptor selection and retention prediction in chromatographic QSRR studies," *Chemometrics and Intelligent Laboratory Systems*, 76, 185-196.

Harrison, D., and Rubinfeld, D. (1978), "Hedonic housing prices and the demand for clean air," *Journal of Environmental Economics and Management*, 5, 81-102.

Hartigan, J. (1975), *Clustering algorithms*, New York: Wiley.

Hastie, T., Tibshirani, R., and Friedman, J. (2001), *The Elements of Statistical Learning*, New York: Springer-Verlag.

He, Z., Xu, X., and Deng, S. (2002), "Clustering Mixed Numeric and Categorical Data: A Cluster Ensemble Approach," Technical Report, Harbin Institute of Technology, Department of Computer Science and Engineering.

Heskes, T. (1997), "Balancing between bagging and bumping," in *Advances in Neural Information Processing Systems* (Vol. 9), eds. M. Mozer, M. Jordan and T. Petsche, Cambridge: The MIT Press, pp. 466-472.

Heskes, T. (1998), "Selecting weighting factors in logarithmic opinion pools," in *Advances in Neural Information Processing Systems* (Vol. 10), eds. M. Jordan, M. Kearns and S. Solla, Cambridge: The MIT Press, pp. 266-272.

Hoerl, A. E., and Kennard, R. (1970), "Ridge Regression: Biased estimation for nonorthogonal problems," *Technometrics*, 12, 55-67.

Hu, X., and Yoo, I. (2004), "Cluster Ensemble and Its Applications in Gene Expression Analysis," in *2nd conference on Asia-Pacific bioinformatics*, pp. 297-302.

Hubert, L., and Arabie, P. (1985), "Comparing Partitions," *Journal of Classification*, 2, 193-218.

Jacobs, A. (2006), "Using Self-similarity Matrices for Structure Mining on News Video," in *4th Hellenic Conference on Artificial Intelligence*, pp. 87-94.

Jinang, D., and Zhang, A. (2004), "Cluster Analysis for Gene Expression Data: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, 16, 1370-1386.

Jonasson, L., Hagmann, P., Thiran, J.-P., and Wedeen, V. J. (2005), "Fiber tracts of high angular resolution diffusion MRI are easily segmented with spectral clustering," Technical Report, Signal Processing Institute, EPFL.

Kaufman, L., and Rousseeuw, P. (1987), "Clustering by means of medoids," in *Statistical Data Analysis based on the L_1 Norm*, ed. Y. Dodge, Amsterdam: Elsevier, pp. 405-416.

Kaufman, L., and Rousseeuw, P. (1990), *Finding groups in data: an introduction to cluster analysis*, New York: Wiley.

Kline, P. (1994), *An Easy Guide to Factor Analysis*, New York: Routledge.

Krogh, A., and Vedelsby, J. (1995), "Neural Network Ensembles, Cross Validation, and Active Learning," in *Advances in Neural Information Processing Systems* (Vol. 7), eds. G. Tesauro, D. S. Touretzky and T. K. Leen, Cambridge: The MIT Press, pp. 231-238.

Lai, W., Huang, X., Wibowo, R., and Tanaka, J. (2005), "An On-Line Web Visualization System with Filtering and Clustering Graph Layout," *IEEE Intelligent Informatics Bulletin*, 5, 11-17.

Larsen, D. R., and Speckman, P. L. (2004), "Multivariate regression trees for analysis of abundance data," *Biometrics*, 60, 543-549.

Lazzeroni, L., and Owen, A. (2002), "Plaid models for gene expression data," *Statistica Sinica*, 12, 61-86.

Leisch, F. (1999), "Bagged Clustering," Technical Report 51, Vienna University of Economics and Business Administration, Adaptive Information Systems and Modeling in Economics and Management Science.

Leisch, F., and Dimitriadou, E. (2005), "*The mlbench Package for R - Machine Learning Benchmark Problems*," R Manual.

Levine, E., and Domany, E. (2001), "Unsupervised estimation of cluster validity using resampling," *Neural Computation*, 13, 2573-2593.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. J. (2004), "Incremental genetic K-means algorithm and its application in gene expression data analysis," *BMC Bioinformatics*, 5.

McLachlan, G. J., Bean, R. W., and Peel, D. (2002), "A mixture model-based approach to the clustering of microarray expression data," *Bioinformatics*, 18, 413-422.

McLachlan, G. J., and Peel, D. (2000), "Mixtures of Factor Analyzers," in *the 17th International Conference on Machine Learning*.

Milligan, G. W. (1980), "An examination of the effect of six types of error perturbation on fifteen clustering algorithms," *Psychometrica*, 45, 325-342.

Milligan, G. W., and Cooper, M. C. (1986), "A Study of the Comparability of External Criteria for Hierarchical Cluster Analysis," *Multivariate Behavioral Research*, 21, 441-458.

Modha, D. S., and Spangler, W. S. (2000), "Clustering Hypertext with Applications to Web Searching," in *ACM Hypertext Conference*.

Monti, S., Tamayo, P., Mesirov, J., and Golub, T. (2003), "Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data," *Machine Learning*, 52, 91-118.

Mühlenbein, H., and Schlierkamp-Voosen, D. (1993), "Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization," *Evolutionary Computation*, 1, 25-49.

Nguyen, G. P., and Worring, M. (2004), "Optimizing similarity based visualization in content based image retrieval," in *IEEE International Conference on Multimedia and Expo*, pp. 759-762.

Nguyen, H. T., Coomans, D., Leermakers, M., and Boman, J. (1997), "Multivariate statistical analysis of human exposure to trace elements from coal in Vietnam," in *SPRUCE IV International Conference on Statistical Aspects of Health and the Environment*.

Nocedal, J., and Wright, S. J. (1999), *Numerical Optimization*, eds. P. Glynn and S. M. Robinson, New York: Springer-Verlag.

Pearson, K. (1901), "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, 6, 559-572.

Ploner, A., and Brandenburg, C. (2004), "Modelling visitor attendance levels subject to day of the week and weather: a comparison between linear regression models and regression trees," *Journal of Nature Conservation*, 11, 297-309.

Pollard, K. S., and van der Laan, M. J. (2002), "New methods for identifying significant clusters in gene expression data," in *Proceedings of the American Statistical Association, Biometrics Section*.

Qu, Y., Adam, B.-L., Yasui, Y., Ward, M. D., Cazares, L. H., Schellhammer, P. F., Feng, Z., Semmes, O. J., and Wright, G. L. (2002), "Boosted Decision Tree Analysis of Surface-enhanced Laser Desorption/Ionization Mass Spectral Serum Profiles Discriminates Prostate Cancer from Noncancer Patients," *Clinical Chemistry*, 48, 1835-1843.

Questier, F., Put, R., Coomans, D., Walczak, B., and Vander Heyden, Y. (2005), "The use of CART and multivariate regression trees for supervised and unsupervised feature selection," *Chemometrics and Intelligent Laboratory Systems*, 76, 45-54.

Rand, W. M. (1971), "Objective Criteria for the Evaluation of Clustering Methods," *Journal of the American Statistical Association*, 66, 846-850.

Ribarsky, B. (2003), "*Fast Clustering for Exploratory Visualization of Large Data*," Electronic Source.

Satten, G. A., Datta, S., Moura, H., Woolfitt, A. R., Carvalho, M. d. G., Carlone, G. M., De, B. K., Pavlopoulos, A., and Barr, J. R. (2004), "Standardization and denoising algorithms for mass spectra to classify whole-organism bacterial specimens," *Bioinformatics*, 20, 3128-3136.

Segal, E., Taskar, B., Gasch, A., Friedman, N., and Koller, D. (2001), "Rich Probabilistic Models for Gene Expression," *Bioinformatics*, 17, S243-252.

Segal, M. R. (1992), "Tree-Structured Methods for Longitudinal Data," *Journal of the American Statistical Association*, 87, 407-418.

Smolkin, M., and Ghosh, D. (2003), "Cluster stability scores for microarray data in cancer studies," *BMC Bioinformatics*, 4.

Smyth, C., and Coomans, D., "Predictive Weighting for Cluster Ensembles," *accepted by Journal of Chemometrics*.

Smyth, C., and Coomans, D. (2006), "Creating Parsimonious Ensembles," in *38th Symposium on the interface of statistics, computing science, and applications*.

Smyth, C., Coomans, D., and Everingham, Y. (2006), "Clustering noisy data in a reduced dimension space via multivariate regression trees," *Pattern Recognition*, 39, 424-431.

Smyth, C., Coomans, D., Everingham, Y., and Hancock, T. (2006), "Auto-Associative Multivariate Regression Trees for Cluster Analysis," *Chemometrics and Intelligent Laboratory Systems*, 80, 120-129.

Spiegelhalter, D., Thomas, A., Best, N., and Lunn, D. (2003), "*WinBUGS User Manual Version 1.4*," WinBUGS Manual.

Strehl, A., and Ghosh, J. (2002), "Cluster ensembles - a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, 3, 583-617.

Therneau, T., and Atkinson, B. (1998), "*RPART*," R Manual.

Therneau, T., and Atkinson, B. (2004), "*mvpart: Multivariate partitioning*," R Manual.

Tibshirani, R. (1996), "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 58, 267-288.

Tibshirani, R., Walther, G., Botstein, D., and Brown, P. (2005), "Cluster validation by prediction strength," *Journal of Computational and Graphical Statistics*, 14, 511-528.

Tibshirani, R., Walther, G., and Hastie, T. (2001), "Estimating the number of clusters in a data set via the Gap statistic," *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 63, 411-423.

Topchy, A. P., Jain, A. K., and Punch, W. (2005), "Clustering Ensembles: Models of Consensus and Weak Partitions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, 1866-1881.

Topchy, A. P., Jain, A. K., and Punch, W. F. (2003), "Combining Multiple Weak Clusterings," in *3rd IEEE International Conference on Data Mining*, pp. 331-338.

Topchy, A. P., Jain, A. K., and Punch, W. F. (2004), "A Mixture Model for Clustering Ensembles," in *SIAM International Conference on Data Mining*, pp. 379-390.

Wadsworth, J. T., Somers, K. D., Stack, B. C., Cazares, L., Malik, G., Adam, B. L., Wright, G. L., and Semmes, O. J. (2004), "Identification of patients with head and neck cancer using serum protein profiles," *Archives of Otolaryngology-Head & Neck Surgery*, 130, 98-104.

Weingessel, A., Dimitriadou, E., and Hornik, K. (2003), "An Ensemble Method for Clustering," in *Distributed Statistical Computing*.

Yeung, K. Y., Haynor, D. R., and Ruzzo, W. L. (2001), "Validating clustering for gene expression data," *Bioinformatics*, 17, 309-318.

Yohannes, Y., and Hoddinott, J. (1999), "Classification and Regression Trees: An Introduction," Technical Report 3, International Food Policy Research Institute.

Zhou, Z.-H., Wu, J.-X., Jiang, Y., and Chen, S.-F. (2001), "Genetic Algorithm based Selective Neural Network Ensemble," in *17th International Joint Conference on Artificial Intelligence*, pp. 797-802.