UNIVERSITY OF SOUTHAMPTON

Ensemble Algorithms and Feature Selection

by

Jeremy D. Rogers

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Engineering and Applied Science
Department of Electronics and Computer Science

August 2007

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

Ensemble Algorithms and Feature Selection

by Jeremy D. Rogers

A popular technique for modelling data is to construct an ensemble of learners and combine them in to a single hypothesis. This final model can achieve an accuracy that is greater than that of the ensemble members, provided that there is a sufficient level of diversity within these learners. Measuring and promoting this diversity can be achieved in a variety of ways and typically a trade-off exists between the accuracy and diversity of the ensemble members. This thesis investigates and develops ensemble techniques for improving this accuracy and diversity, and compares them to other well-known ensemble methods. These algorithms are shown to successfully promote diversity whilst maintaining the learner accuracy.

An important area of machine learning research is that of feature selection. Choosing an appropriate subset of the available features with which to represent the data can improve the performance of learning algorithms in terms of accuracy, efficiency and interpretability. However, this task is non-trivial and can be complicated further through interactions amongst the features, which can result in features only being relevant within a local area of the space. Through the creation of diverse local models, ensemble methods have the capability to address these issues and identify feature relevance. This work develops new methods that utilise these aspects of ensemble algorithms to identify and exploit feature information. Experiments demonstrate that these methods can surpass existing approaches in terms of classification accuracy, dimensionality reduction and relevance identification.

Contents

Acknowledgements

I would like to thank my supervisor Prof. Steve Gunn for his advice and support during the course of my PhD.

Nomenclature

θ	Hypothesis Space
X	Input Data
Y	Target Labels
N	Sample Size
$\mathcal{X}$	Sample Subset
w	Example Weight
a	Example Ambiguity
h	Base Hypothesis
H	Ensemble Hypothesis
T	Ensemble Size
α	Hypothesis Weight
A	Hypothesis Ambiguity
ϵ	Hypothesis Error
Z	Normalisation Coefficient
F	Data Dimensionality
S	Feature Subset
v	Feature Weight
IG	Information Gain
Ent	Entropy
Imp	Impurity Function
E	Expectation
I	Indicator Function
l	Decision Tree Node
$\mathcal{L}$	Set of Tree Nodes
ϕ	Node Complexity
St	Strength of Hypotheses
ρ	Correlation Measure
C	Correlation Coefficient
γ	Confidence Parameter
D	Distance Measure
β	Angle
σ^2	Sample Variance
μ	True Distribution Mean
$\lvert\cdot\rvert$	Set Cardinality

Chapter 1

Introduction

Many areas of science can benefit from the ability to build models from large amounts of data. Applications possess a need to extract the rules and general principles to improve understanding and enable predictability. This has motivated the development of computer learning systems, which have the ability to construct reliable models in an efficient manner. Typically, the goal of these methods is to create models that capture the essence of the data, whilst not becoming overly complex. Depending on the type of learning problem under consideration and the specific requirement of the application, the choice and complexity of learning algorithm can vary. Research in machine learning deals with the understanding of relationships within data and of developing algorithms to uncover and exploit this information.

1.1 Problem Statement

The type of learning problem considered here is that of supervised learning, where each example in the data is asscoiated with a target value. This target variable, Y, can assume a continuous value in the case of regression or one of a set of labels in the case of classification. Each example is described by a vector of values corresponding to feature variables. This input data, X, can then be viewed as a set of points in an F-dimensional space, where F denotes the number of feature variables. The requirement of the learning algorithm is to identify the relationship between the input data and the target. For clarity this thesis focuses on binary classification, where the target can assume one of two possible labels. However, the potential exists for extending the methods in this thesis, such that they can be applied to any supervised learning task.

There exist many types of learning algorithm to perform binary classification. Ensemble algorithms are one class of learning algorithm, which have achieved much success by combining multiple weak learners to form one strong hypothesis. Two popular approaches to ensemble construction are Bagging (Breiman, 1996) and Boosting (Freund and Schapire, 1999). It is known that a successful ensemble is one that consists of accurate base learners that make their mistakes on different parts of the data. This is because the overall accuracy of an ensemble is dependent upon the amount of data which is predicted correctly by most of the ensemble members. Therefore, ensemble algorithms should attempt to construct learners that are both accurate and diverse. Many algorithms have been developed to achieve this and they do so in a variety of ways. The quality of the data that is presented to each learner is one aspect that can be manipulated to promote accuracy or diversity, and the forementioned Bagging and Boosting methods are examples of this. The feature representation of the data can also be manipulated to improve ensemble performance (Ho, 1998a), whilst other methods explicitly measure the accuracy and diversity of ensemble members and employ a selection strategy to optimise these factors (Opitz, 1999).

Depending upon the objective of the particular learning problem, describing the data in the original F-dimensional space may be sub-optimal. The factors which affect the optimality of a set of features are varied and identification of the best subset is a non-trivial task. Therefore, one of the problems posed by learning tasks is that of selecting an appropriate subset of the original features and much research has been conducted on this topic. The particular motivation for a learning problem can affect the criteria for selecting an optimal set of features. It has been observed that data can contain irrelevant features and that these can be detrimental to the accuracy of the algorithm (Almuallim and Dietterich, 1991). Higher dimensionality in the data can lead to increased computational costs, a reduced potential for interpretability of an induced hypothesis and result in an effect known as *The curse of dimensionality* (Bellman, 1961). Performing a search through the space of possible feature subsets is a combinatorial problem and is infeasible for high dimensional data. However, some algorithms employ a basic search strategy which simplifies the task by limiting the size of the space that is searched. Fast feature selection methods often analyse individual features and rank them according to some relevance measure. These global measures can fail to identify features with strong interaction that may only be relevant within a local area of the input space (Friedman, 1994). Some methods aim to improve learner performance through the identification of this type of local feature relevance (Domingos, 1997; Hong, 1997). The useful information that is carried by a set of features may be subsumed by another set and it can be beneficial if this redundancy within the data is identified and removed. It has also been observed that the particular learning algorithm implemented can affect which features are optimal and therefore, some techniques utilise the learning algorithm to identify the most useful features (Kohavi and John, 1997).

Certain types of ensemble algorithm are useful for feature selection as they can explore different feature representations of the data and produce diverse local models. In particular, this research develops feature relevance identification techniques using the Random Forest algorithm (Breiman, 2001), which constructs an ensemble of decision trees using random feature selection. The ability of Random Forest as a feature selection method is supported by the concept of random subset exploration and the potential of techniques based on decision trees to identify the local relevance of features.

This thesis is concerned with investigating ways of improving ensemble performance through the promotion of accuracy and diversity in the base learners. The work discusses and develops new diversity promotion techniques that employ data and feature manipulation methods. The accuracy of the learners is increased by improving the robustness of these techniques or through feature relevance identification and the development of techniques for employing this obtained knowledge.

1.2 Related Work

The relationship between the accuracy and diversity of the base learners, and the performance of the resulting ensemble has previously been studied. Breiman (2001) motivates the need for ensemble diversity by bounding the generalisation error of an ensemble in terms of this quantity. Their exist different measures for quantifying diversity and the suitability of these is a topic that has been explored by Kuncheva and Whitaker (2001). Explicit attempts have been made to promote diversity through a manipulation of the training data (Melville and Mooney, 2003) or through a manipulation of the feature representation, such as the Random Forest algorithm (Breiman, 2001). This promotes ensemble diversity through random feature selection and, consequently, performs better than Bagging (Breiman, 1996). Further improvements to this have been sought by creating additional diversity (Cutler and Zhao, 2001; Robnik-Sikonja, 2004). Typically, improving the diversity of the ensemble members worsens their accuracy and these methods explore this mechanism. The techniques developed in this thesis are designed to promote diversity whilst being robust to the data and the learning process. In this way, the methods improve the diversity whilst maintaining accuracy.

The employment of ensemble algorithms to perform feature selection has been attempted previously. The Random Forest algorithm lends itself to feature selection because of its random subset exploration and this has recently been explored (Chen and Lin, 2006; Svetnik et al., 2004; Borisov et al., 2006). Other algorithms select a subset of learners based on their accuracy and diversity (Opitz, 1999; Cunningham and Carney, 2000; Oliveira et al., 2003) and it has been suggested that these methods have the capacity to identify diverse feature representations in the data.

The pursuit of ensemble diversity results in algorithms that achieve a good exploration of possible hypotheses, and the ability of these learners may vary in different areas of the input space. Therefore, the potential exists for these type of techniques to uncover local feature relevance. Tsymbal et al. (2006) attempt to exploit this concept through an alteration to the combination strategy of Random Forest. Instead of altering the combination method or employing a selection strategy, the techniques that are developed here examine the ensemble to identify feature information and then use this knowledge to improve ensemble construction. This methodology enables the creation of better ensemble members rather than attempting to combine ones that are sub-optimal. The feature information that is gained from these techniques is also useful for the understanding and interpretation of the data.

1.3 Contributions

This work has made contributions in the following areas:

- Development of novel diversity promotion techniques. This research investigates methods of building accurate and diverse ensembles through the manipulation and exploitation of different aspects of the data and the learning process. Specifically, methods are developed that promote diversity whilst maintaining the accuracy of the base learners. This is achieved through the introduction of new methods for altering the distribution over the data and manipulating the feature representation to exploit the decision tree bias.
- Demonstration and justification of the need for feature selection when performing classification with Random Forest and other diversity promotion techniques. This work discusses and gives theoretical justification for the specific effects of feature selection on Random Forest and demonstrates them through experimental methods.
- Novel approaches to feature weighting/selection using Random Forest. This work explores techniques using the Random Forest algorithm to perform feature relevance identification and ways in which this information can be employed to benefit the algorithm. In particular, methods for improving Random Forest through updating the feature sampling distribution and eliminating irrelevant features are introduced. It is shown that Random Forest can identify more complex relationships between the features and that this knowledge can be utilised by the algorithm to improve performance. This work describes a new technique that improves Random Forest through the identification and employment of local feature information.
- Improved relevance identification through the employment of information theoretic and statistical techniques. Use of statistical methods and techniques from information theory have been adopted to provide more reliability to the techniques, which make them more robust. In particular an information theoretic technique is introduced which improves the measure of feature relevance, and the employment of hypothesis testing is shown to provide more effective feature selection.

This work has contributed to the following publications:

- Rogers J.D. and Gunn S.R. (2005). Ensemble Algorithms for Feature Selection, Sheffield Machine Learning Workshop, LNCS, Vol 3635, Pages 180-198.
- Rogers J.D. and Gunn S.R. (2006). Identifying Feature Relevance Using a Random Forest, Latent Structure and Feature Selection techniques: Statistical and Optimisation Perspectives Workshop. Bohinj, Slovenia 2005, LNCS, Vol 3940, Pages 173-184.
- Rogers J.D. and Gunn S.R. (2007). Ensemble Diversity and Feature Selection, Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence.

1.4 Thesis Outline

Chapter 2 discusses the mechanics of ensemble methods and describes some popular choices of ensemble algorithm, along with the reasons for their success. The chapter also considers the suitability of possible base learning algorithms and what impact these have on the performance of the algorithm as a whole. The chapter concludes with a description of the Random Forest algorithm, which forms a key component of this work.

Chapter 3 reviews some theoretical work concerning the importance of ensemble diversity and examines some currently employed techniques for measuring this quantity. Several algorithms that promote diversity are then compared and categorised according to their subject of manipulation.

Chapter 4 introduces a new diversity promotion method that functions by focussing the algorithm on certain areas of the space. This approach is related to outlier identification and its tolerance to noise is argued. A further new method is also proposed which alters the feature representation of the data to exploit the learning bias of decision tree induction.

Chapter 5 conducts an empirical evaluation of the techniques that were introduced in Chapter 4. The methods are compared to other well known ensemble algorithms and the reasons for the differences in performance are explored and discussed.

Chapter 6 conducts a review of feature selection techniques. A theoretical justification is given for the specific problems that irrelevant features pose to the Random Forest algorithm. The different ways of defining feature relevance are discussed and their limitations are examined. Some currently employed feature selection techniques are described and compared in terms of their efficiency and relevance identification ability.

Chapter 7 conducts a review of feature selection methods that utilise Random Forest and explores the potential of this algorithm to identify feature importance. These measures of feature relevance are then used to derive several feature weighting/selection techniques. Improvements to these techniques are also discussed, which rely on statistical and information theoretic concepts. The chapter also explores the capacity of Random Forest to identify local feature relevance and how this local knowledge can be employed.

Chapter 8 describes experimental results for the methods introduced in Chapter 7 on artificial and real world data sets. The specific effects of feature selection on the Random Forest algorithm are demonstrated and the feature selection techniques introduced are shown to compare favourably to other well known methods. The techniques of Random Forest that identify and exploit local feature knowledge are evaluated and their mechanisms are compared to the other ensemble methods of Chapter 4.

Chapter 9 describes a machine learning application, which involves inferring user preference from eye movements. Details are given on the method of constructing this data set and on the features that are used to represent the eye movements of the user. The methods that are developed in this thesis are applied to this task, and the different aspects of the data that result in a variation in performance of these techniques is discussed.

A discussion concerning the outcome of the work described in this report is given in Chapter 10 along with some possible directions for future work.

Chapter 2

Learning Ensembles

2.1 Introduction

The general scheme for ensemble construction consists of training a set of simple base learners and combining their learned knowledge into a single powerful hypothesis. Ensemble algorithms vary in their choice of base learner, combination strategy and method of training. Broadly, ensemble algorithms can be classed as adaptive or non-adaptive. Adaptive algorithms alter the method of ensemble construction for subsequent learners using knowledge gleaned from the set of hypotheses that have already been trained. In contrast, non-adaptive algorithms construct independent learners, which can be viewed as being drawn from the same distribution. Typically, the combination of these learners is achieved through aggregation, weighted voting or the employment of an ensemble selection strategy.

This chapter describes two popular ensemble techniques, Bagging and Boosting. The choice of base hypothesis is also discussed in the context of some commonly employed learners. The chapter concludes with a description of the Random Forest algorithm.

2.2 Bagging

A popular ensemble method is Bagging (Breiman, 1996), which trains each weak learner on a subset of the training data. These subsets are formed through sampling N examples with replacement from the training data of size N. It can be shown that the probability P of an example not being in the sampled subset of data for hypothesis h_t is,

$$P\left[x \in \mathcal{X}_t^{OOB}\right] = \left(1 - \frac{1}{N}\right)^N , \tag{2.1}$$

where $\mathcal{X}_t^{OOB}$ represents the subset of data that is not sampled for hypothesis h_t and can be referred to as the out-of-bag set. This converges quickly to,

$$\lim_{N\to\infty} P\left[x \in \mathcal{X}_t^{OOB}\right] = \frac{1}{e} \approx \frac{1}{3}. \tag{2.2}$$

As the examples are drawn independently, this method selects around $\frac{2}{3}$ of the training data with some examples selected multiple times. This means that the learners are trained on different parts of the training data, which should result in varying solutions. The prediction of each base learner or hypothesis h_t for example x is then aggregated to form the ensemble prediction,

$$H(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} h_t(x)\right). \tag{2.3}$$

An explanation for the effectiveness of Bagging, given in Breiman (1998), is that of the bias and variance of the classifier. Here, the expected error of any classifier can be decomposed into three terms. The first is the Bayes misclassification rate, which is the theoretical minimum error rate of any classifier. The bias and variance terms are both defined as the difference between the accuracy of the Bayes classifier, $BC(X)$, and the average accuracy of the base hypotheses. However, the bias and variance are defined on two exclusive sets of the total distribution from which the training and testing data is drawn. The bias set consists of all points which give a classification that is contrary to the Bayes classifier more often than not, over all possible hypotheses. The variance set is the complement of this. Example x, y is included in the bias set, $\mathcal{X}_B$, if

$$P_\theta\left[h_\theta(x) = BC(x)\right] < \frac{1}{2} \tag{2.4}$$

and is included in the unbiased set $\mathcal{X}_U$ if

$$P_\theta\left[h_\theta(x) = BC(x)\right] \geq \frac{1}{2}. \tag{2.5}$$

The bias and variance components of a learner that is drawn from a hypothesis distribution, θ, can be written as the difference between the accuracy of the Bayes classifier and the expected accuracy of the learner on these two sets,

$$Bias(h_\theta) = P_{X,Y}\left[BC(X) = Y, X \in \mathcal{X}_B\right] - E_\theta P_{X,Y}\left[h_\theta(X) = Y, X \in \mathcal{X}_B\right] \tag{2.6}$$

$$Variance(h_\theta) = P_{X,Y}\left[BC(X) = Y, X \in \mathcal{X}_U\right] - E_\theta P_{X,Y}\left[h_\theta(X) = Y, X \in \mathcal{X}_U\right]. \tag{2.7}$$

As Bagging works by aggregating hypotheses, the variance term can be eliminated and the overall error reduced. The logical argument is then that bagging works especially well if the base learning algorithm produces classifiers with a large variance and small bias.

One of the advantages of using bagged training sets is that some of the training data is left out in the form of the out-of-bag set. This means that each individual learner automatically has its own independent test set, which can be used to evaluate the performance. The out-of-bag data can also be used to estimate the test error of the ensemble by classifying every training point using all of the hypotheses where it was in the out-of-bag set and aggregating the result.

2.3 Boosting

Boosting is an adaptive type of ensemble algorithm and a well known variant of this is AdaBoost. This algorithm maintains a weighted distribution w_t over the training examples and uses this to train each weak learner. Each of these learners is also weighted by a value α_t, which is calculated according to its accuracy, taken with respect to w_t. The intuition behind AdaBoost is that examples which are misclassified by the algorithm are deemed more important, and AdaBoost then focuses on the more difficult components of the training data in order to construct a more accurate model. At each iteration the weight assigned to example x_i with corresponding label y_i is updated according to,

$$w_{t+1}(x_i) = \frac{w_t(x_i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}, \tag{2.8}$$

where Z_t is a normalising factor to ensure that w_{t+1} is a distribution.

The hypotheses are combined through weighted voting, where hypothesis h_t is assigned weight α_t.

$$H(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{2.9}$$

Freund and Schapire (1999) show that the training error can be bounded by,

$$\frac{1}{N}\sum_{i=1}^{N} I\left(H(x_i) \neq y_i\right) \leq \prod_{t=1}^{T} Z_t, \tag{2.10}$$

where $I(\cdot)$ is the indicator function which equals 1 if the condition within the brackets is true and 0 otherwise.

At each iteration the value of α_t is chosen to minimise Z_t and for binary classification the value chosen is,

$$\alpha_t = \frac{1}{2}\log\frac{(1-\epsilon_t)}{\epsilon_t}, \tag{2.11}$$

where ϵ_t is the training error of hypothesis h_t weighted by w_t.

Breiman (1998) suggests that the success of AdaBoost is due to its re-sampling strategy of focussing on commonly misclassified examples, and is not due to the specific form of the algorithm. Following this intuition, Breiman introduces ARC-X4 which combines learners through aggregation and updates the distribution over the training examples according to,

$$w_{t+1}(x_i) = 1 + \left(\sum_{t'=1}^{t} I\left(h_{t'}\left(x_i\right) \neq y_i\right) \right)^4 . \tag{2.12}$$

This method alters the relative importance of the examples according to their frequency of misclassification and was found to achieve a performance similar to that of ADABOOST.

By updating a distribution over the training data, these methods also alter the distribution over the hypothesis space. Consequently, examples can be moved between the bias and variance sets and it has been observed (Bauer and Kohavi, 1999) that both of these algorithms can reduce the bias component of the error as well as that of the variance.

2.4 Base Learning Algorithms

The choice of which base learning algorithm to implement can have a large effect on the ensemble as a whole. The general requirement of the base learners is that they are accurate enough to give some useful information concerning the target, but are sufficiently different to enable an improvement upon their combination. Also, as many of these hypotheses need to be created, it is important that their construction does not impose a computational overhead that is too great. This section describes three popular choices of base learner, the K-nearest neighbour algorithm, decision trees and the naïve Bayes model.

2.4.1 K-Nearest Neighbour

The nearest neighbour algorithm is a 'lazy' learner that, given a test example, assigns to it the class that is most commonly represented in the set of closest training examples. It is described as a lazy learning algorithm because it has no training stage. All of the training data is kept and no calculations are performed until a test instance needs to be classified. The intuition behind the method is that examples which are in close proximity to each other are more likely to belong to the same class. Although the distance metric can vary, the Euclidian distance is most often used.

$$D\left(x_i, x_j\right) = \|x_i - x_j\|_2 \tag{2.13}$$

If $x_i\left(f\right)$ denotes the value of feature f for example x_i, then this distance can also be written,

$$D^2\left(x_i, x_j\right) = \sum_{f=1}^{F} \left(x_i\left(f\right) - x_j\left(f\right)\right)^2 . \tag{2.14}$$

When classifying new data, the algorithm first uses Equation 2.13 to establish the K nearest neighbours by comparing the test case x_j, with each of the training examples. These nearest examples constitute the set $\mathcal{X}_{nn}$ and x_j is then assigned to the mode class in this set.

$$h_t(x_j) = \arg\max_{y} \sum_{i \in \mathcal{X}_{nn}} I(y_i = y) \tag{2.15}$$

Figure 2.1 shows a simple two dimensional example of a binary classification problem and the decision boundary that is induced by the K-Nearest Neighbour algorithm, with $K = 1$.

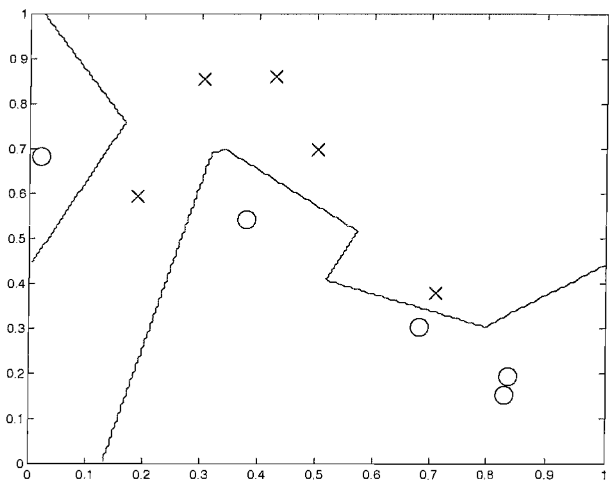

FIGURE 2.1: Example binary classification problem and induced hypothesis boundary through nearest neighbour.

The value of K is a parameter of the method. If it is chosen to be too small, the algorithm can focus too closely on the immediate vicinity of the test cases. In this case the algorithm may overfit the data by being misled by noise or random variations in the data. Conversely, if the value of K is too large, the learner may become too general and not identify the more subtle trends in the data. When considering employing this base learning algorithm in conjunction with an ensemble method, the choice of K will have an effect on the bias and variance components of the error. The stability of the models increases with the value of K and if K is too large, the learners become too similar and do not yield a great benefit from their combination.

2.4.2 Decision Tree Induction

Decision trees attempt to discover relationships in the data by recursive partitioning. At each node a search is performed to find a suitable separation of the data, which results in the formation of child nodes. Here, binary partitions are considered which result in a node being split into two new nodes only. This partitioning continues until a stopping criterion is reached or until no more partitions are available. Test instances can then be run down the tree, following the partitioning conditions of each split, until a terminal node is reached. The output of the hypothesis for the particular test instance is then governed by the state of the data in that terminal node.

The criteria for selecting a particular split usually involves an impurity function, *Imp*, which describes the level of disagreement amongst the values of the target variable, Y, within the data of a node. The objective is to reduce the impurity so that similar data is contained within each terminal node. The gain that is achieved by partitioning the data in node l into a set, $\mathcal{L}_c$, of new child nodes is then the reduction in this impurity.

$$Gain\left(l, \mathcal{L}_c\right) = Imp\left(Y, \mathcal{X}_l\right) - \sum_{i:l_i \in \mathcal{L}_c} \frac{|\mathcal{X}_{l_i}|}{|\mathcal{X}_l|} Imp\left(Y, \mathcal{X}_{l_i}\right), \tag{2.16}$$

where $\mathcal{X}_l$ and $\mathcal{X}_{l_i}$ represent the set of data within node l and child node l_i respectively.

If a tree is constructed for the purpose of regression, the impurity function can take the form of the variance, Var, of the target values.

$$Imp\left(Y, \mathcal{X}_l\right) = Var\left(Y, \mathcal{X}_l\right) = \frac{1}{|\mathcal{X}_l|} \sum_{i:x_i \in \mathcal{X}_l} \left(y_i - \overline{y}\right)^2, \tag{2.17}$$

where $\overline{y}$ denotes the mean of the target values of set $\mathcal{X}_l$. In this case, the output of the hypothesis assumes the mean of the target data in each terminal node. Therefore, the gain represents the reduction in squared error that is achieved by the partition.

For classification problems a different impurity function is required. A popular choice for this is entropy (Quinlan, 1986), *Ent*, which quantifies the amount of information required to describe the data. When using entropy, the gain is referred to as the Information Gain, *IG*.

$$Imp\left(Y, \mathcal{X}_l\right) = Ent\left(Y, \mathcal{X}_l\right) = -\sum_{y \in \mathcal{Y}} \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|} \log_2 \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|}, \tag{2.18}$$

where $\mathcal{Y}$ denotes the set of possible class labels.

Some tree induction procedures consider multiple partitions at each node. In this case, Information Gain will favour features that involve more partitions and bias the algorithm. To overcome this problem, Quinlan (1986) suggests using Gain Ratio,

$$GR\left(l, \mathcal{L}_c\right) = \frac{IG\left(l, \mathcal{L}_c\right)}{EntP\left(\mathcal{L}_c\right)}, \tag{2.19}$$

which penalises the gain with the entropy of the partition,

$$EntP\left(\mathcal{L}_c\right) = -\sum_{c \in \mathcal{L}_c} \frac{|\mathcal{X}_c|}{|\mathcal{X}_l|} \log_2 \frac{|\mathcal{X}_c|}{|\mathcal{X}_l|}. \tag{2.20}$$

A measure implemented by the CART algorithm (Breiman et al., 1984), is the *GINI* index,

$$Imp\,(Y, \mathcal{X}_l) = GINI\,(Y, \mathcal{X}_l) = 1 - \sum_{y \in \mathcal{Y}} \left(\frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|} \right)^2 . \tag{2.21}$$

For binary classification, Heath et al. (1993) employ the sum minority measure SM, to identify a suitable partition, which can be considered as adopting the impurity function,

$$Imp\,(Y, \mathcal{X}_l) = SM\,(Y, \mathcal{X}_l) = \min_{y} \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|}. \tag{2.22}$$

When classifying new data, a test instance descends the tree and finishes in a terminal node. If the output of a classification tree is given by the mode class of the data within this node, the gain for the impurity function of Equation 2.22 represents the reduction in training error.

Figure 2.2 shows how the three impurity measures vary for different proportions of positive and negative examples when performing binary classification. All of the measures are maximum when the node contains equal numbers of each class and are zero when the node is pure.

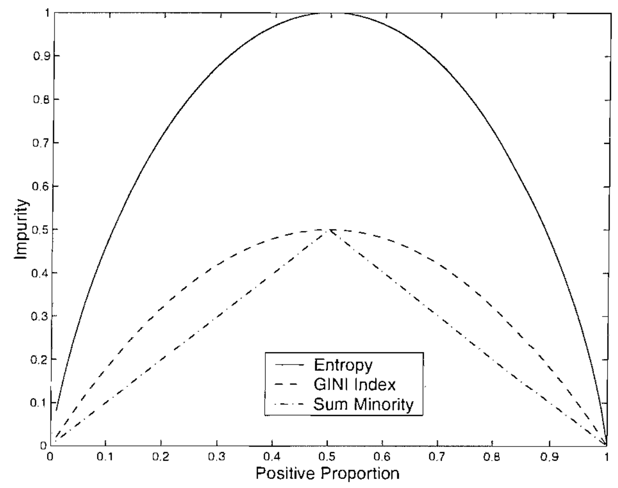

FIGURE 2.2: Node impurity measures for binary classification.

The decision tree algorithm can continue to partition the nodes of a tree until no more splits are possible, but this can lead to the model overfitting the data. To avoid this, a stopping criterion can be used in order to prevent the nodes from becoming overly specific. In the extreme case, each learner consists of a single partition and these are referred to as decision stumps. Rather than halting the construction, the tree can be grown as large as possible and then pruned back to a more suitable size. The criterion for limiting the size of a tree can be based on the performance of the tree, when tested on an independent set of test data. Alternatively, some measure of complexity of the tree, such as the number or size of the terminal nodes can be utilised. One method for controlling the complexity of the tree is to employ the *minimum description length* principle. This principle simply states that the optimal hypothesis should minimise the information required to describe the hypothesis and the data using the hypothesis. Fayyad and Irani (1993) use this method to derive a stopping criterion for their discretisation technique. This uses recursive entropy partitioning to convert continuous features into discrete ones and is therefore equivalent to decision tree induction using a single feature. The algorithm will continue to partition the data until no split can be found that reduces the required information.

When partitioning data, the entropy represents the theoretical minimum average number of bits required to describe each data point in node l. The information gain IG, is then the reduction in this value caused by splitting the data into two new nodes, $l1$ and $l2$. The total amount of information that is saved is $|\mathcal{X}_l|\, IG$ bits. The cost of introducing a partition has several parts. Firstly, if the node contains $|\mathcal{X}_l|$ examples, then there are $|\mathcal{X}_l| - 1$ possible split positions (see Figure 2.3), assuming that they are all realisable. This leads to a cost of $\log_2(|\mathcal{X}_l| - 1)$ bits.

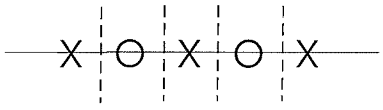

FIGURE 2.3: Illustration of 5 data points and 4 possible split positions. ($|\mathcal{X}_l| = 5$)

Each node requires a key, which assigns a bit vector to each class to enable the description of the data. The size of this key is approximated by $|\mathcal{Y}_l| \cdot Ent\,(l)$, where $|\mathcal{Y}_l|$ is the number of classes represented in l. The keys of nodes $l1$ and $l2$ replace that of node l and it is this change that represents the cost. Furthermore, a description must be given concerning which of the classes are contained within each node. Here, each class can be viewed as having three possible states as a result of the split:

1. Having representative examples in $l1$ only;
2. Having representative examples in $l2$ only;
3. Having representative examples in both $l1$ and $l2$.

Therefore, there are $3^{|\mathcal{Y}_l|}$ possible arrangements for the way in which the classes are distributed between the two new nodes. As the split must separate some data, neither node may be empty. The two arrangements which describe all of the representatives of all of the classes only existing within one of the nodes, must be ignored. The cost for describing this arrangement is then, $\log_2\left(3^{|\mathcal{Y}_l|} - 2\right)$.

The partitioning of the data will only continue if splits can be found which satisfy the inequality,

$$\begin{aligned} |\mathcal{X}_l|\, IG \;\geq\; & \log_2(|\mathcal{X}_l| - 1) + \log_2\left(3^{|\mathcal{Y}_l|} - 2\right) \\ & - |\mathcal{Y}_l|\, Ent\,(l) + |\mathcal{Y}_{l1}|\, Ent\,(l1) + |\mathcal{Y}_2|\, Ent\,(l2)\,, \end{aligned} \tag{2.23}$$

where $|\mathcal{Y}_{l1}|$ and $|\mathcal{Y}_{l2}|$ are the number of classes represented in nodes $l1$ and $l2$ respectively.

This method can be applied directly to decision tree pruning, however, as this method was developed for discretisation, there is no cost associated with describing which features were used and where. Here, the dimensionality of the data is denoted as F and the subset of features that have been used in the construction of the tree is denoted as S. To give the key for the features which were used requires $|S| \cdot \log_2 F$ bits. Each split then requires $\log_2 |S|$ bits to describe which feature was used.

As the tree is constructed, the number of previous splits must be observed and recorded. If the current set of features S, has previously resulted in d splits, then the cost of creating a new split using a feature within S is simply $\log_2 |S|$. However, the cost of introducing a new feature must account for a new addition to the key and the changing of all of the node descriptions. The cost in this case is,

$$\log_2 F + (d+1)\log_2(|S|+1) - d\log_2 |S|. \tag{2.24}$$

This method enables the tree to continue partitioning the nodes, if the gain in class discrimination outweighs the complexity of the hypothesis.

Figure 2.4 shows the same two dimensional example as in Figure 2.1 with a decision boundary formed through decision tree induction. The node impurity of this decision tree was measured using entropy and no pruning or stopping criterion was employed.

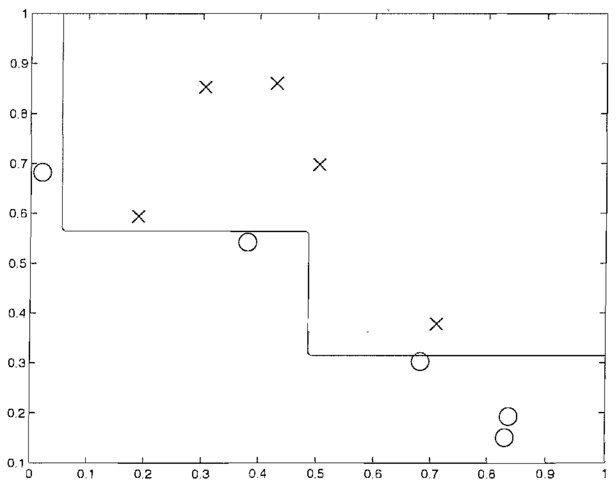

FIGURE 2.4: Example binary classification problem and induced hypothesis boundary through decision tree induction.

When performing classification, the predictions of the base hypotheses can take the form of the class labels. Alternatively, a soft output that reflects the confidence in the predictions can be used. ADABOOST weights all of the outputs of a given hypothesis using Equation 2.8, where the weight reflects the accuracy of the hypothesis on the training data. However, different base learners can provide a readily available measure of confidence in each prediction. Such a measure can be obtained from a decision tree prediction by examining the terminal node that contains the example in question. Frank and Witten (1998) propose the number of training examples contained within the terminal node as a measure of the predictions reliability. This is reasonable as smaller nodes are more sensitive to variations in the data and are more likely to be found in areas of the space that are awkward to classify. Ho (1998b) employs the proportion of examples that belong to the majority class within the node and this was found (Bauer and Kohavi, 1999) to improve the performance of Bagging. This concept of class purity can also be adopted in the nearest neighbour algorithm where the proportion is taken over the set of nearest neighbours, $\mathcal{X}_{nn}$.

2.4.3 Naïve Bayes

The naïve bayes classifier, which is also used to construct ensembles (Bauer and Kohavi, 1999; Tsymbal et al., 2002), attempts to model the probability of the class over the input space. Using Bayes theorem, this can be written,

$$P(Y|X) = P(Y|X_1, ..., X_F) = \frac{P(Y)\,P(X_1, ..., X_F|Y)}{P(X_1, ..., X_F)}, \tag{2.25}$$

where $X_1, ..., X_F$ represent the different features of the data. The output of the classifier for example x_i is then given by,

$$h(x_i) = \arg\max_y \{P(y|x_i)\}. \tag{2.26}$$

When comparing the class probabilities for a particular example, the probability of the data can be considered constant and ignored. The naïve bayes model makes the assumption that the features are independent when conditioned on the class. Two features, X_1 and X_2, can be considered conditionally independent if,

$$P(X_1|Y, X_2) = P(X_1|Y). \tag{2.27}$$

With this assumption and the fact that the denominator of Equation 2.25 can be ignored, the conditional probabilities can be written,

$$P(Y|X_1, ..., X_F) \propto P(Y) \prod_{f=1}^{F} P(X_f|Y). \tag{2.28}$$

This independence assumption enables the probabilities of each feature to be calculated separately and, therefore, more efficiently.

2.4.4 Stability

Breiman (1996) argues that bagging works better for unstable learners, which produce very different hypotheses for a small perturbation in the training data. As bagging works by repeatedly perturbing the training data, a large variety of hypotheses will be created. Decision tree methods fulfil this criteria, as changing small amounts of the training data can result in alternative features or split locations being chosen to partition particular nodes. The descendants of this split will all be affected and an entire branch of the tree can alter dramatically. In contrast, if a stable learner is bagged, all base hypotheses will be similar and no great improvement in accuracy would be achieved. The Nearest Neighbour algorithm is one example of a learner that is stable in this respect. As Bagging typically selects about $\frac{2}{3}$ of the training data, the nearest example in the whole of the training set to any test instance would be expected to be included in $\frac{2}{3}$ of the base hypotheses. Therefore, if the nearest neighbour algorithm is used in conjunction with Bagging and $K = 1$, the majority of the base hypotheses will concur with the result of a single nearest neighbour that uses all of the training data. The resultant ensemble will not generate a significant improvement over the single hypothesis. The output of the nearest neighbour algorithm is given by Equation 2.15 and is based on the set of nearest examples, $\mathcal{X}_{nn}$. Bagging does not create a large improvement with nearest neighbour because its method of sub-sampling the data does not sufficiently alter this set. The naïve bayes learner can also be regarded as a stable learner and has been observed to offer little improvement with Bagging (Bauer and Kohavi, 1999). This is because sub-sampling of the data in this manner does not produce large changes in the estimates of the class distributions and, therefore, creates similar learners.

2.5 Random Forest

It is known that a successful ensemble is one which consists of base learners that make their mistakes on different parts of the data. Therefore, a good ensemble construction process should promote variation in the learners. Breiman (2001) exploits this concept with the Random Forest algorithm. This method uses Bagging to create a set of decision trees, but randomises the split selection process. At each node, instead of performing a search through all of the features for an optimal split, the search is only conducted over a randomly selected subset of the features. To increase the variability of these models further, the trees are grown as large as possible. Figure 2.5 illustrates the diversity of the individual trees with some examples that are produced by Random Forest on a simple binary classification problem in two dimensions. Upon aggregation of 100 trees, a more accurate hypothesis can be formed and is shown in Figure 2.6.

Breiman tried two sizes of subset, one experiment used only a single feature to perform the optimal split and another used subsets of size $\log_2 F + 1$, where F is the total

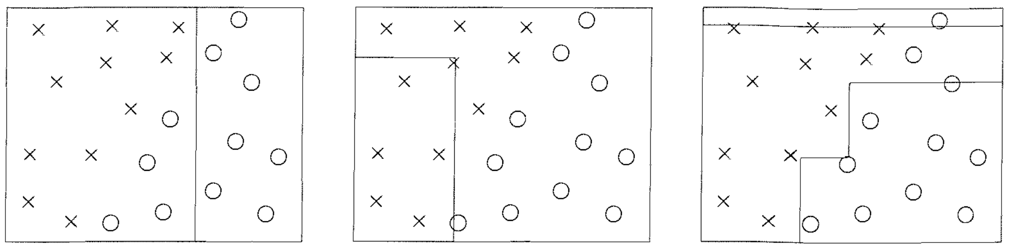

FIGURE 2.5: Examples of individual base hypotheses produced by RF for binary classification.

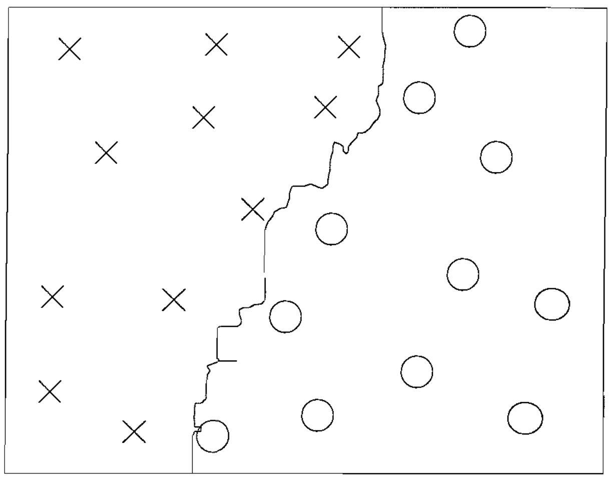

FIGURE 2.6: Aggregated hypothesis over 100 trees.

number of features. It was noted that the procedure was not overly sensitive to the size of subset chosen. Due to this observation and the simplicity and interpretability of using a single feature, the experiments conducted here use subsets of size 1. Breiman also tried random linear combinations of features where at each node the split is performed using a fixed number of features. The features are randomly weighted and summed. This allows different relationships in the data to be explored by not limiting the hypotheses to just creating hyper-rectangles in the input space. Breiman found this method to be marginally better than using a single feature.

2.6 Conclusions

This chapter has described the concept of ensemble algorithms along with three well established methods: Bagging, Boosting and Random Forest. Some base learning methods were examined and their suitability discussed. When generating ensemble members it is important to consider the space of potential hypotheses, as the members that are drawn from this space are required to be accurate and diverse. These quantities of accuracy and diversity are crucial to the success of the ensemble and improvements can be gained through their promotion. The Bagging algorithm depends upon the base learners being unstable such that the perturbation of the data results in learners with a high variance. Random Forest extends this idea further by adding randomisation into the feature selection process and enlarging the hypothesis space. By implementing this idea, the diversity is increased whilst maintaining a degree of accuracy in the members of this space.

Chapter 3

Ensemble Diversity

3.1 Introduction

Ensemble algorithms have achieved much success in machine learning through their ability to combine multiple weak learners to form one strong hypothesis. As discussed in Chapter 2, the methods for construction and combination of these learners vary, but the explanations for how they work centre around the concept of diversity within the base learners. Many algorithms have been developed to exploit this diversity and create more accurate ensembles. Although it is hoped that these base learners will achieve some level of accuracy, it is immediately apparent that there is no benefit in their combination if they are identical.

This chapter reviews and motivates the need for diversity in ensemble combination and discusses the different ways that diversity can be viewed. Several approaches, that are currently employed to measure diversity, are compared and an overview of diversity promotion techniques is also given.

3.2 Motivation

When constructing an ensemble, it is advantageous to build base learners that are accurate and discover useful relationships within the data. However, an ensemble is only useful if the discovered relationships vary in some way. As previously discussed, Breiman (1998) showed that the classification error of an ensemble can be decomposed into a bias and a variance term and that aggregation of the hypotheses can eliminate the variance component. This demonstrates that it is favourable if the base learners make their mistakes on different parts of the data, such that the majority of the ensemble is usually correct. A different way of representing the ensemble error was given by Krogh and Vedelsby (1995) for regression ensembles. For an ensemble that weights each hypothesis,

h_t, with a factor, α_t, the output is given by,

$$H(x_i) = \sum_{t=1}^{T} \alpha_t h_t(x_i), \tag{3.1}$$

where

$$\sum_{t=1}^{T} \alpha_t = 1. \tag{3.2}$$

They define the ambiguity, $a(x_i)$, of an example as the mean squared difference between the hypothesis outputs and the combined ensemble output.

$$a(x_i) = \sum_{t=1}^{T} \alpha_t \left(h_t(x_i) - H(x_i)\right)^2 \tag{3.3}$$

They then show that if $\epsilon_t(x_i)$ is the error of hypothesis h_t on example x_i, the error of a regression ensemble is given by,

$$\left(H(x_i) - y_i\right)^2 = \sum_{t=1}^{T} \alpha_t \epsilon_t(x_i) - a(x_i). \tag{3.4}$$

This shows that the error of a regression ensemble is given by the average error of the hypotheses minus the ambiguity. The ambiguity is a measure of the disagreement amongst the ensemble members and this, therefore, gives evidence advocating the promotion of this type of disagreement.

Analysis of error and diversity can also be performed for classification problems. Tumer and Ghosh (1996) examine the case when an ensemble consists of base learners that produce an output for each class, where each output approximates the conditional class distribution, $P(y|x_i)$.

$$h_t^y(x_i) = P(y|x_i) + \epsilon_t^y(x_i), \tag{3.5}$$

where ϵ_t^y is the error associated with the output for class y of hypothesis h_t.

The Bayes optimum decision is given by classifying an example x_i to class y for which,

$$P(y|x_i) > P(y'|x_i) \;\; \forall \; y' \neq y. \tag{3.6}$$

When an ensemble of these learners are constructed, the final output is given by averaging each class output over the ensemble and predicting class y for example x_i, for which output $H^y(x_i)$ is greatest. They show that the expected additional error of such an ensemble, $Err_{add}(H)$, beyond that of the Bayes optimum decision, is given by,

$$E\left[Err_{add}(H)\right] = E\left[Err_{add}(h)\right]\left(\frac{1+\rho(T-1)}{T}\right). \tag{3.7}$$

The term $E\left[Err_{add}(h)\right]$ is the expected additional error of a single hypothesis and ρ is the correlation between classifier errors, averaged over all pairs and all outputs. If the errors are independent, $\rho = 0$ and the additional error is reduced by $\frac{1}{T}$, the size of the ensemble. If the errors are completely correlated, then $\rho = 1$ and no improvement is gained.

When considering ensembles of classifiers that produce a single class prediction, an important concept is that of the margin. The margin of an example, $marg(x, y)$, is defined as the difference between the probability over the hypothesis space θ, of correct classification and the probability of predicting the next most likely class,

$$marg\left(x,y\right) = P_\theta\left(h_\theta\left(x\right) = y\right) - max_{j \neq y} P_\theta\left(h_\theta\left(x\right) = j\right). \tag{3.8}$$

Breiman (2001) uses the law of large numbers to prove that the misclassification rate of an ensemble converges asymptotically to the probability over the input space of achieving a negative margin.

$$P_{X,Y}\left(H\left(X\right) \neq Y\right) = P_{X,Y}\left(marg\left(X,Y\right) < 0\right) \tag{3.9}$$

Breiman then bounds the generalisation error in terms of the strength and correlation of the base learners. The strength St is defined as the expectation of the margin over the input space,

$$St = E_{X,Y}\left[marg\left(X,Y\right)\right]. \tag{3.10}$$

The correlation is measured between the raw margin functions, $rmarg\left(h_t, x, y\right)$, which can be described as the contribution that hypothesis h_t gives to the margin of example x. The margin of example x can then be defined as the expectation of the raw margin function over the hypothesis space.

$$marg\left(x,y\right) = E_\theta\left[rmarg\left(h_\theta, x, y\right)\right] \tag{3.11}$$

The generalisation error of a majority vote ensemble can then be bounded by,

$$P_{X,Y}\left(H\left(X\right) \neq Y\right) \leq \frac{\overline{\rho}\left(1 - St^2\right)}{St^2}, \tag{3.12}$$

where $\overline{\rho}$ is the average correlation between the raw margin functions. This bound is only valid for $0 \leq St \leq 1$.

Although the bound is loose, it suggests that an accurate ensemble should consist of accurate and diverse base learners.

3.3 Measuring Diversity

It is known that diversity is a useful tool in ensemble construction and some techniques attempt to measure this quantity, although the adopted metrics vary. Sharkey and Sharkey (1997) give four definitions for describing the state of the diversity of an ensemble. Type I diversity exists when all of the training examples are predicted correctly by at least one of the base learners and there are no two classifiers which make the same mistake on any of the data. Type II diversity exists when there are some coincident errors but the majority of the ensemble is always correct. Type III occurs when there is always at least one member that gives the correct prediction but the majority is not always correct and type IV describes the case when some examples exist which none of the ensemble members predict correctly. Types I and II describe the ideal case as an ensemble will converge to a solution that correctly predicts all of the data. These definitions are useful when performing ensemble selection as it may be possible to select a subset of the learners such that a type III ensemble can be converted into a type II and that a type II can be converted into a type I. However, a type IV ensemble cannot be considered upwardly mobile in this sense, as the examples that are misclassified by all hypotheses can not be classified correctly by any subset.

Measuring the level of diversity within an ensemble can generally be achieved in two ways, either by some measure of correlation between the hypotheses or by some measure of ambiguity in the predictions of each example. This section reviews some of these measures that are currently employed.

3.3.1 Correlation Measures

This type of diversity metric analyses the level of agreement between pairs of classifiers and can be averaged over all of the pairs in an ensemble to produce an overall measure. Alternatively, it can be used to select or remove particular learners, based on their similarity to the rest of the ensemble. The measures of pairwise correlation vary, but some employ the standard correlation coefficient. This coefficient $C(V_1, V_2)$, measures the correlation between two variables V_1 and V_2, with respect to a sample of size n, as

$$C(V_1, V_2) = \frac{\sum_{i=1}^{n} \left[\left(V_1(i) - \overline{V_1}\right)\left(V_2(i) - \overline{V_2}\right) \right]}{\sum_{i=1}^{n} \left(V_1(i) - \overline{V_1}\right)^2 \sum_{i=1}^{n} \left(V_2(i) - \overline{V_2}\right)^2}. \tag{3.13}$$

In the analysis of Tumer and Ghosh (1996), this correlation is taken between the errors of the classifier outputs. Their measure of ensemble similarity is this correlation measured over all of the classifier pairs and over all of the classes,

$$\rho = \sum_{y \in \mathcal{Y}} P_{X,Y}(Y = y) \left(\frac{1}{T(T-1)} \sum_{t=1}^{T} \sum_{u \neq t} C\left(\epsilon_t^y(\mathcal{X}), \epsilon_u^y(\mathcal{X})\right) \right), \tag{3.14}$$

where $P_{X,Y}(Y=y)$ is the prior probability of class y and $\epsilon_t^y(\mathcal{X})$ defines the vector of errors for output y of hypothesis h_t, over the set of examples being tested, $\mathcal{X}$.

Breiman's bound on the generalisation error uses the correlation coefficient to analyse the agreement between the raw margin functions of classifier pairs,

$$\rho(h_1, h_2) = C\left(rmarg(h_1, \mathcal{X}), rmarg(h_2, \mathcal{X})\right), \tag{3.15}$$

where $rmarg(h_t, \mathcal{X})$ denotes the vector of raw margin values for hypothesis h_t over the set of examples, $\mathcal{X}$. This is then averaged over all of the pairs in the ensemble.

$$\overline{\rho} = \frac{1}{T(T-1)} \sum_{t=1}^{T} \sum_{u \neq t} \rho(h_t, h_u) \tag{3.16}$$

A simple measure of agreement between classifiers was adopted by Ho (1998b), which defines the correlation between two classifiers as the probability that they concur.

$$\rho(h_1, h_2) = P_{X,Y}\left[h_1(X) = h_2(X)\right] \tag{3.17}$$

An estimation of this probability can be achieved, although it requires an independent set of data with which to test both hypotheses. This estimate can then be written as the proportion of examples in this set, for which the classifiers agree.

$$\rho(h_1, h_2) = prop(h_1, h_2) = \frac{|\mathcal{X}^{00}| + |\mathcal{X}^{11}|}{N} \tag{3.18}$$

Here, $\mathcal{X}^{ab}$ represents the set of examples which satisfy the conditions of a and b, where $a = 1$ if h_1 is correct and $a = 0$ if it is incorrect. Similarly, $b = 1$ if h_2 is correct and $b = 0$ if it is incorrect. Margineantu and Dietterich (1997) employ the Kappa statistic which improves the above measure by compensating for the probability that the agreement occurs by chance. The probability of agreement by chance, P^{ABC}, can be written,

$$P^{ABC}(h_1, h_2) = \frac{\left(|\mathcal{X}^{00}| + |\mathcal{X}^{01}|\right)\left(|\mathcal{X}^{00}| + |\mathcal{X}^{10}|\right)}{N^2} + \frac{\left(|\mathcal{X}^{10}| + |\mathcal{X}^{11}|\right)\left(|\mathcal{X}^{01}| + |\mathcal{X}^{11}|\right)}{N^2}. \tag{3.19}$$

The Kappa statistic, $\kappa(h_1, h_2)$, is then written,

$$\rho(h_1, h_2) = \kappa(h_1, h_2) = \frac{prop(h_1, h_2) - P^{ABC}(h_1, h_2)}{1 - P^{ABC}(h_1, h_2)}. \tag{3.20}$$

A pairwise correlation measure favoured by Kuncheva and Whitaker (2001) is the Q statistic, which measures the correlation between two classifiers h_1 and h_2. This measure assumes a value in the range $[-1, +1]$ and equals 0 if the errors of the hypotheses are independent. This measure is favourable as these values do not depend on the accuracy of the base learners.

$$\rho(h_1,h_2) = Q(h_1,h_2) = \frac{|\mathcal{X}^{11}|\,|\mathcal{X}^{00}| - |\mathcal{X}^{01}|\,|\mathcal{X}^{10}|}{|\mathcal{X}^{11}|\,|\mathcal{X}^{00}| + |\mathcal{X}^{01}|\,|\mathcal{X}^{10}|} \tag{3.21}$$

Another correlation measure is implemented by Goebel and Yan (2004), where the correlation is calculated with respect to the number of coincident errors.

$$\rho(h_1,h_2) = \frac{2\,|\mathcal{X}^{00}|}{|\mathcal{X}^{10}| + |\mathcal{X}^{01}| + 2\,|\mathcal{X}^{00}|} \tag{3.22}$$

Instead of averaging these pairwise quantities, this measure is extended to T classifiers so that the correlation of the entire ensemble can be observed.

$$\rho = \frac{T\,|\mathcal{X}^{0}|}{N - |\mathcal{X}^{0}| - |\mathcal{X}^{1}| + T\,|\mathcal{X}^{0}|}, \tag{3.23}$$

where $|\mathcal{X}^0|$ represents the number of examples that are misclassified by all of the ensemble members and $|\mathcal{X}^1|$ represents the number of examples that are classified correctly by all of the ensemble members.

Correlation can also be measured between a hypothesis and the output of the rest of the ensemble (Zenobi and Cunningham, 2001; Opitz, 1999) and this has been termed the ambiguity of a classifier,

$$A(h_{T+1}) = \frac{1}{N}\sum_{i=1}^{N} I\left(h_{T+1}(x_i) \neq H(x_i)\right). \tag{3.24}$$

This can be useful when employing an ensemble selection strategy, as this measure reflects the impact that hypothesis h_{T+1} has on the diversity of the current ensemble. For binary classification where the possible class labels are $\{+1,-1\}$, this can also be written,

$$A(h_{T+1}) = \frac{1}{2} - \frac{1}{2N}\sum_{i=1}^{N} h_{T+1}(x_i)\,H(x_i). \tag{3.25}$$

3.3.2 Example-Based Measures

Another view of diversity concerns the ambiguity in the predictions of individual data points. By measuring the proportions of the votes for an example, a measure of diversity can be constructed and this can be averaged over all of the data. The diversity of an example is greatest when equal votes are given for each class. For regression ensembles, Krogh and Vedelsby (1995) describe the ambiguity of an example as in Equation 3.3. This quantity measures the variance in the predictions of the example in question and is implemented by Oliveira et al. (2003). This principle is also adopted by Cunningham and Carney (2000) for classification problems. They define the ambiguity of an example as the entropy of the class predictions,

$$a_{Ent\_a}(x) = -\sum_{y_i \in \mathcal{Y}} \left(\frac{\sum_{t=1}^{T} I(h_t(x) = y_i)}{T} \right) \log_2 \left(\frac{\sum_{t=1}^{T} I(h_t(x) = y_i)}{T} \right). \tag{3.26}$$

A different measure of diversity is given by Kuncheva and Whitaker (2001), which they also call a type of entropy. For binary classification, their entropy, a_{Ent_b}, of an example x is given by the number of votes awarded to the minority class in relation to the number that would yield the maximum disagreement, $\lfloor \frac{T}{2} \rfloor$.

$$a_{Ent\_b}(x) = \frac{1}{\lfloor \frac{T}{2} \rfloor} \min \left\{ \sum_{t=1}^{T} I(h_t(x) = y), \sum_{t=1}^{T} I(h_t(x) \neq y) \right\} \tag{3.27}$$

This can also be written,

$$a_{Ent\_b}(x) = \frac{T - \text{abs}\left(\sum_{t=1}^{T} h_t(x)\right)}{2 \lfloor \frac{T}{2} \rfloor}. \tag{3.28}$$

The quantity,

$$\widehat{H}(x) = \text{abs}\left(\sum_{t=1}^{T} h_t(x)\right), \tag{3.29}$$

can be viewed as a measure of confidence in the ensemble prediction and can be seen to be inversely related to $a_{Ent\_b}(x)$. Therefore, it is important to note that an ambiguous or diverse example is one which has not been classified with a large degree of confidence. Care must then be taken with algorithms that explicitly attempt to promote this type of diversity as they can also reduce the confidence in the model. For this reason, it is favourable to employ a correlation-based measure to examine the level of diversity within an ensemble.

3.4 Diversity Promotion

Ensemble diversity can be promoted in a variety of ways and this section overviews some of the techniques that are currently used to achieve this. Here, the methods are categorised by the way that they manipulate the learning technique. They can be classified as manipulating the data, the features or the learning algorithm itself.

3.4.1 Data Manipulation

Some algorithms alter the training examples that are used to construct each learner. Bagging is an example of this, as each learner is trained with a sub-sampled set of the original training data. As previously discussed, Bagging is useful if the base learning algorithm is unstable, such that small perturbations in the training data cause large changes in the induced hypothesis. Bagging will fail to promote the necessary level of diversity if a stable learner is used. It has been observed that Bagging is more useful than Boosting for small sample sizes (Skurichina et al., 2002) and the explanation for this is given by the effect that sample size has on the diversity of the ensemble. For smaller sample sizes the effect of perturbing the training data on the induced hypothesis will be much greater. The improvement created by Bagging is less significant for larger sample sizes as the bagged sets of training data will be similar and result in an ensemble with low diversity. Skurichina et al. (2002) suggests that Boosting methods, such as ADABOOST (Freund and Schapire, 1999) and ARC-X4 (Breiman, 1998), promote diversity by focussing the algorithm on the class boundaries. Unlike Bagging, the manipulation of the data is not performed randomly with Boosting. The distribution over the data is updated in such a way as to award higher weights to the examples which are hard to classify. As discussed in Section 3.3.2, diversity is sometimes measured with respect to example ambiguity and an ambiguous example can be considered hard to classify as the ensemble members disagree about how it should be labelled. Boosting can then be viewed as promoting diversity by focussing the algorithm on the ambiguous examples.

The DECORATE algorithm (Melville and Mooney, 2003) introduces artificial training examples in order to promote diversity within an ensemble. They achieve this by drawing examples from an estimate of the data distribution and labelling them in a way that contradicts the current ensemble. The artificial data deliberately misleads the learning algorithm to create an ensemble with high diversity. The amount of artificial data to create for each learner is a parameter of the algorithm. If the amount of additional data is too small, then an insufficient level of diversity will be created. However, too much additional data will have a detrimental effect on the accuracy of the hypotheses and make them too unreliable. They empirically found that creating as many artificial training examples as there were in the original training data produced good results. Although these artificial examples mislead the base learners through contradicting the general trend of the data, they should not mislead the ensemble, as they are created randomly at each iteration and are not consistently present in the same areas of the input space for all of the learners.

3.4.2 Feature Manipulation

Altering the set of examples that are presented to each learner or their relative measures of importance are not the only methods for creating variation in the base learners. The set of features used to represent the data can also be manipulated. In the same way that ADABOOST produces diverse learners by forcing them to consider difficult examples, the FEATUREBOOST algorithm (O'Sullivan et al., 2000) increases diversity by focussing on less descriptive features. The algorithm maintains a weighted distribution over the features, which represents their relevance to the current learner. At each iteration a set of features is removed, according to this distribution, to force the subsequent learner to explore the discriminative power of features that were not utilised by the previous hypothesis. The method calculates the minimum value, F^{min}, such that if F^{min} features are corrupted, the error of the current learner is expected to be sufficiently increased. F^{min} features are then removed according to the distribution and the reduced feature set is used to construct the next hypothesis.

The Random Subspace Method (Ho, 1998a) trains each base learner on a randomly sampled subset of the features and uses this to introduce a measure of randomness to the ensemble construction. Tsymbal et al. (2002) noted that if this approach is employed with the naïve bayes base learners that were described in Section 2.4, the probabilities for each feature only need to be calculated once. This is because all of the learners are trained on the same set of data and the learner simply selects the probabilities that are relevant for the given feature subset. However, if these learners are used in conjunction with data manipulation algorithms such as Bagging, the probabilities must be re-calculated for each learner. Bay (1999) showed that the random subspace method introduced enough diversity into the base hypotheses to enable Bagging to improve classification accuracy of nearest neighbour algorithms. The diversity that is introduced into a nearest neighbour ensemble comes from the variation that occurs in the distance measure between the examples. This distance measure, given in Equation 2.14, becomes,

$$D(x_i, x_j) = \sum_{f \in S} (x_i(f) - x_j(f))^2, \tag{3.30}$$

where the summation is now taken over the random feature subset S. This has a much greater capacity for variation in the set of nearest neighbours, $\mathcal{X}_{nn}$, than the data subsampling of Bagging alone.

A variant of this technique is implemented by Bryll et al. (2003), where the size of the feature subsets is fixed. The algorithm first conducts trials on the data to establish a suitable subset size. Small subset sizes result in inaccurate but diverse learners, whilst larger subset sizes create learners that are accurate but more correlated. Ho (2002) compared ensembles constructed using Bagging to ones that employed random subspace on several data sets for which different complexity measures are also calculated. It was found that the random subspace method performed more favourably for problems where the training set is large compared to the dimensionality. A measure of the non-linearity of the class boundary was also calculated by training a nearest neighbour classifier on the data set and testing it on artificially generated examples. These examples are created through an approximation of the convex hull of each class. A high error rate on these examples indicates a larger degree of non-linearity of the class boundary. Ho found that the random subspace method was less favourable than Bagging for data sets that exhibited a large degree of this non-linearity. A possible explanation for this is that these data sets have a more complex structure and, as random subspace discards some features for each learner, a suitable hypothesis becomes difficult to generate and the base learners become too unreliable.

The randomisation technique of Dietterich (2000) constructs an ensemble of decision trees where instead of choosing the best possible split at each node, the best twenty splits are found and one is chosen randomly from this set. This simple method reduces the correlation between the trees and was empirically shown to be competitive with Bagging and ADABOOST. The Random Forest algorithm (Breiman, 2001) also introduces diversity by randomising the split selection of decision trees, but uses the random subspace method to select a random subset at each node and finds the best split available from that set.

Randomly reducing the set of features for each learner reduces the similarity and correlation between learners at the cost of also reducing their accuracy. Oza and Tumer (2001) utilise feature subspaces to generate diverse but accurate learners for multi-class classification. Their method constructs a set of learners, where each learner is a specialist for each class. The specialisation is gained through selecting the subset of features that are most correlated to that particular class. The intuition behind this method is that the diversity of the resultant ensemble is gained through training each learner on different feature subsets, but the accuracy is maintained through selecting useful subsets.

3.4.3 Algorithm Manipulation

A variety of hypotheses can be induced from different learning algorithms and the suitability of each is dependent upon the learning problem. Furthermore, different learning biases may be more appropriate in different areas of the input space of a particular problem. Brodley (1995) attempts to uncover this variation with the Model Class Selection algorithm, which selects the best from a set of algorithms to partition the space in a tree structured manner. The method has the capacity to utilise the biases of decision trees, nearest neighbour and linear discriminant functions to construct a composite hypothesis which can achieve a performance that is better than any of the individual algorithms.

When constructing an ensemble, the variation in learning bias can be exploited to produce diverse hypotheses. Caruana et al. (2004) adopts this method by employing many different learning algorithms including K Nearest Neighbour, Decision Trees, Decision Stumps and Support Vector Machines (SVM). These diverse learners are then combined through an ensemble selection strategy. Tsoumakas et al. (2004) also constructs a set of learners using different learning algorithms and uses a statistical test to identify a subset of these learners, which are significantly the most accurate. As an example of the difference between learning biases, Figure 3.1 illustrates the two dimensional example that was given in Chapter 2 as well as the induced hypotheses of a nearest neighbour and a decision tree. Both hypotheses fit the data perfectly but the shaded areas show where the two learners disagree.

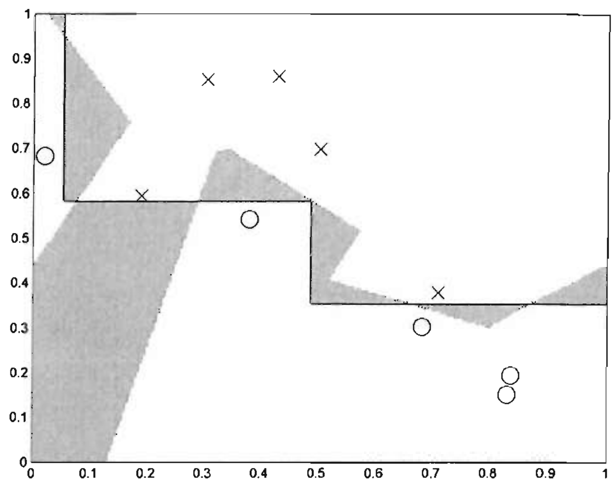

FIGURE 3.1: Binary classification example showing the disagreement between a nearest neighbour classifier and a decision tree.

For some learning algorithms, a set of diverse learners can be created by varying the model parameters. The method of Caruana et al. (2004) constructs several SVM's using various choices of kernel function and cost parameter. Robnik-Sikonja (2004) create further decorrelation between the classifiers of the Random Forest algorithm by varying the choice of node split criterion. The split procedure is randomised further by Cutler and Zhao (2001) in the construction of PERT (PErfect Random Trees). At each node, the split is performed by randomly choosing two examples from different classes and choosing a random point between them along a random feature. Geurts et al. (2006) also randomise the split value at each node, but draw the value randomly from a uniform distribution along the range of the feature. The size of the feature subset, that is randomly selected and searched at each node, is an input parameter of the technique and controls the strength of the trees. In the extreme case of selecting a single feature, the trees are totally randomised and are independent of the target. The large random element involved in these node split selection schemes result in trees which are larger and less accurate than those obtained with random forest. However, due to the high level of diversity between the base learners, the ensembles that are generated by these methods are shown to be competitive with a random forest.

Other algorithms employ some of the above techniques to generate a large number of different hypotheses and then adopt a selection strategy to exploit this variation. The method of Opitz (1999) explicitly measures the accuracy and diversity of the learners and selects a subset which maximises these quantities. Other methods attempt to generate an accurate set of learners and select a diverse group from this set (Cunningham and Carney, 2000; Oliveira et al., 2003). Margineantu and Dietterich (1997) regards each pair of learners as a point in a 2-dimensional space which is defined by their average accuracy and pairwise correlation. This method then selects all of the learner pairs that form the convex hull in this space.

3.5 Conclusions

This chapter has reviewed some theoretical results justifying the importance of diversity in ensemble construction. The definition of this diversity varies, but can be categorised as correlation-based or example-based. Correlation-based methods quantify the similarity between the predictions of pairs of learners, whereas example-based methods utilise a measure of ambiguity in the predictions of individual examples. Correlation-based methods are favourable for examining the level of diversity within an ensemble because example-based measures are inversely related to the ensemble confidence. However, the example ambiguity is a useful concept when considering the mechanics of diversity promotion.

An overview of diversity promotion techniques has been given and these have been grouped according to their subject of manipulation. These methods can be viewed as altering the hypothesis space from which the learners are drawn. Compared to other techniques, Bagging produces accurate but quite correlated learners, therefore, drawing learners from a small hypothesis space. Boosting methods can also employ accurate learners but alter the distribution of the hypothesis space through focussing the learners on awkward problems. The Random Forest algorithm randomises feature selection to produce a much larger and more diverse hypothesis space, which contains learners that are less accurate on average. In order to optimise the accuracy and diversity of the base learners, diversity promotion should be conducted in a way that maintains learner accuracy.

Chapter 4 will introduce new methods for diversity promotion and motivates these approaches in terms of their methods of manipulating and exploiting the data, the features or the base learning algorithm.

Chapter 4

Robust Promotion of Diversity

4.1 Introduction

Chapter 3 reviewed and discussed several approaches for promoting diversity within an ensemble, as well as motivating the concept through previous theoretical results. As previously discussed, Boosting is a type of algorithm that can be viewed as introducing diversity by concentrating the learners on more difficult examples. A method is introduced here, that focusses learners on ambiguous or diverse examples rather than commonly misclassified ones. The advantages of this algorithm are discussed in the context of outlier detection and separate-and-conquer techniques.

Diversity can also be promoted through manipulation of the feature space and Chapter 3 reviewed several examples of this. A new method is given which alters the feature representation of the data to exploit the bias of the base learning algorithm and generate further ensemble diversity.

4.2 Data Manipulation

The ADABOOST algorithm, described in Chapter 2, alters the relative importance of the training examples such that the subsequent base learners concentrate more on the commonly misclassified examples. Skurichina et al. (2002) suggests that Boosting promotes diversity in the ensemble by concentrating on the examples that lie in the regions of class boundaries. This idea also offers an explanation for why boosting algorithms perform well with stable base learners such as decision stumps. Due to the nature of ADABOOST focusing on commonly misclassified examples, it has been observed (Dietterich, 2000; Bauer and Kohavi, 1999) that it can be intolerant to noise.

The objective of ADABOOST is to minimise Z_t, which can be written as,

$$\min \prod_{t=1}^{T} Z_t = \min_{\alpha_T} \frac{1}{N} \sum_{i=1}^{N} \exp\left(-y_i\left(\alpha_T h_T(x_i) + \sum_{t=1}^{T-1} \alpha_t h_t(x_i)\right)\right). \tag{4.1}$$

This loss function increases exponentially for negative values of

$$y_i\left(\alpha_T h_T(x_i) + \sum_{t=1}^{T-1} \alpha_t h_t(x_i)\right), \tag{4.2}$$

and is of key importance when considering the problems faced by ADABOOST in the presence of outliers. An outlier is an example which is expected to be misclassified by a learning algorithm as its presence is contrary to the general trend of the data. Outliers then give large negative values for the above quantity, which can result in a dominating effect on the solution. A variation of ADABOOST called LOGITBOOST (Friedman et al., 1998) implements a cost function which only increases linearly for negative values of the above quantity and is shown to be more robust to the presence of noise. Figure 4.1 shows a two dimensional classification problem, with the region of the class boundary illustrated by the dashed lines. Two outliers are also shown and these examples can be seen to contradict the data and lie away from the class boundary. It is reasonable to expect outliers to be misclassified by a large majority of the base learners and because of this, outliers should not be viewed as ambiguous examples as they do not lie in the regions of class boundaries. This then poses the question of whether these type of algorithms should actively concentrate on commonly misclassified examples or ambiguous ones.

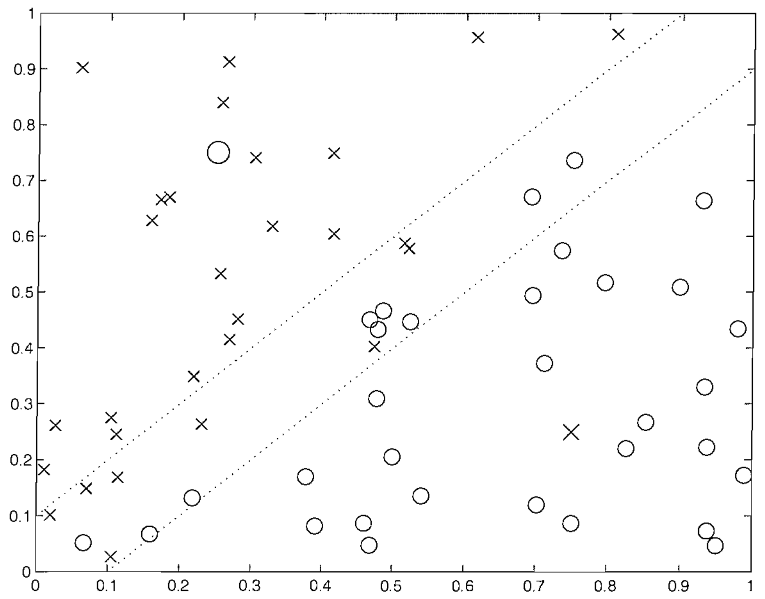

FIGURE 4.1: Two dimensional binary classification problem with dashed lines indicating the approximate class boundary. Two outliers are also shown at (0.25,0.75) and (0.75,0.25).

4.2.1 Outlier Identification

The presence of outliers in data causes several problems for machine learning tasks. They can be defined as examples which contradict the general trend of the data and can mislead learning algorithms. This can result in a reduced generalisation ability or an increase in computational load. Therefore, the elimination of these outliers is beneficial and many methods have been developed to achieve this. It is important to note that due to the different biases imposed by learning algorithms, the effects of outliers vary between them and what may be considered an outlier by one method may not be for another.

One approach to outlier identification is to construct a model that describes the data well and search for the examples which contradict this model. John (1995) uses this approach in the construction of ROBUST-C4.5 decision trees. It is well known that pruning a decision tree can result in a model which gives better generalisation than one that has been expanded fully and classifies the training data perfectly. Consequently, pruned decision trees misclassify part of the training data which is judged to be uninformative. This philosophy can be extended to hypothesising that if these examples can be considered locally uninformative within a leaf of the tree, then they can be considered globally uninformative. In this case, the examples may have misled the algorithm at previous stages, where their presence may have caused a node higher up the tree to be split differently. In order to cope with this, the algorithm of John (1995) repeatedly builds and prunes decision trees and at each iteration removes the examples that are misclassified. The technique continues until all of the examples are classified correctly, and the resultant models are typically more accurate and concise.

Outliers can also be identified through ensemble techniques and two such methods have been introduced by Brodley and Friedl (1999). As with ROBUST-C4.5, a model is created and the examples which are misclassified by this model are labelled as outliers. With an ensemble, an example is misclassified if the majority of the learners predict the wrong class and this form of outlier identification is termed majority filtering. A similar method labels examples as outliers if they have been misclassified by all of the learners and this is termed consensus filtering. The algorithm proposed by Brodley and Friedl (1999) and adopted by Berthelsen and Megyesi (2000) constructs the learners using different learning algorithms in an attempt to identify outliers which are independent of the implemented algorithm. Verbaeten and Assche (2003) adopted consensus and majority filtering but constructed diverse learners by perturbing the training data with Bagging and cross-validation committees.

The choice between consensus and majority filtering is an unresolved issue and appears to be data dependent. Consensus filtering is clearly more conservative at removing data and is therefore prone to leaving outliers in the data. In contrast, majority filtering removes more data but can suffer from removing some good examples and hindering the learning process with a reduced set of training data. It is apparent that these methods can be generalised such that example x_i is designated as an outlier if it is misclassified by at least T_o of the learners.

Where I is the indicator function, example x_i is an outlier if,

$$\sum_{t=1}^{T} I\left(h_t\left(x_i\right) \neq y_i\right) \geq T_o. \tag{4.3}$$

Consensus and majority filtering are then equivalent to Equation 4.3 with $T_o = T$ and $T_o = \frac{T}{2}$ respectively. This can also be written as,

$$y_i \sum_{t=1}^{T} h_t\left(x_i\right) \leq T - 2T_o. \tag{4.4}$$

T_o can then be viewed as a model selection parameter and optimised accordingly. The absolute value of the left hand side of Equation 4.4 is equivalent to the ensemble confidence, $\widehat{H}$, given in Equation 3.29. Therefore, these methods show that outlier detection can be performed by identifying examples that are classified with a high degree of confidence, but also incorrectly. Equation 4.4 indicates that an example is more likely to be an outlier, the lower the value of,

$$y_i \sum_{t=1}^{T} h_t\left(x_i\right). \tag{4.5}$$

From Equation 4.2 it can be seen that ADABOOST assigns larger weights to examples which exhibit low values of,

$$y_i \sum_{t=1}^{T} \alpha_t h_t\left(x_i\right), \tag{4.6}$$

which indicates why ADABOOST can suffer with the presence of outliers.

4.2.2 Separate and Conquer

The separate-and-conquer covering strategy (Furnkranz, 1999) is used to describe a class of algorithms which recursively discover rules to explain part of the training data and then remove the examples that are covered by this rule. Typically, at each stage a rule is chosen that is judged in some way as being the most powerful. By removing the examples that are covered, the technique can be viewed as repeatedly focussing on the more difficult areas of the input space. Frank and Witten (1998) adopt this philosophy in the construction of partial decision trees. This technique effectively chooses the leaf of a pruned decision tree which contains the largest number of examples. The largest leaf is deemed to be the most general rule and the data that it covers is removed from the training data and another tree is created. Although all of the rules are kept for classifying new data, the method contains similarities to the previously discussed outlier detection algorithms. The largest node is chosen as the most general rule and the examples that it covers are removed, including the examples that the rule misclassified. In this sense, the misclassified examples can be viewed as outliers in the methodology of John (1995).

Other methods exist for establishing the most general rule and Ferri et al. (2004) achieves this by introducing the concept of a 'Cautious Classifier'. This type of classifier covers the examples which it classifies with high confidence. In this sense, any soft classifier or classifier which provides an associated measure of confidence in the prediction can be converted into a cautious classifier. This can be achieved by setting a confidence threshold γ, such that an example is covered if it is classified with confidence at least γ. Ferri et al. (2004) selects a value for γ using global absolute percentage, which ensures that each classifier covers at least a fixed proportion p, of the data. This is formulated as,

$$\gamma = \max_{\gamma'} \left\{ \sum_{i=1}^{N} I\left(\widehat{h}\left(x_i\right) \geq \gamma'\right) \geq pN \right\}, \tag{4.7}$$

where $\widehat{h}\left(x_i\right)$ is the associated confidence or real-valued output of classifier h on example x_i.

A variation of this idea was developed by Khoussainov et al. (2005) in the design of TRISKEL, where pairs of classifiers are produced. Each of these classifiers is biased to predicting one of the classes, so that one classifier has high precision on the positive examples and the other has high precision on the negative examples. The examples for which the two hypotheses classify in agreement are deemed to be classified confidently and hence covered. These are removed and the remaining ones are used to train a third, unbiased, classifier. Alternatively, an iterative approach can be used where a further two biased classifiers are constructed and the algorithm repeats. The assumption can be made that when the negatively biased classifier predicts an example as positive, the positively biased classifier does the same and vice versa. In this case, the classifiers are always in agreement when predicting the examples that dictate the positive precision of the negatively biased classifier and the negative precision of the positively biased classifier. As these examples are covered by this agreement, these two precision values give lower bounds on the precision values for the total ensemble.

The separate-and-conquer methodology can then be summarised as removing the examples that are classified confidently and concentrating on the more difficult areas of the space. These difficult areas can be defined as the ambiguous examples within an ensemble, such as the diversity measures described in Section 3.3.2. Some outlier detection techniques take a similar approach by labelling examples as outliers if a hypothesis classifies them confidently and incorrectly. The confidence in the TRISKEL algorithm can be defined as the agreement between two classifiers, whereas the confidence in the consensus and majority filtering approaches can be defined as the level of agreement within an ensemble. Therefore, separate-and-conquer techniques can be viewed as containing an implicit form of outlier detection. The ADABOOST algorithm also concentrates on the difficult areas of the space, but defines these areas as ones which are commonly mis-classified. ADABOOST will then assign large weights to the examples which are mis-classified by the majority of the ensemble. These examples can be seen as being classified confidently but incorrectly and would be removed by the outlier detection techniques and ignored by the separate-and-conquer techniques. Khoussainov et al. (2005) suggests a link between boosting and separate-and-conquer by altering their approach to adjusting the weights of the examples instead of removing them.

4.2.3 Boosting Diversity

The previous sections have illustrated the problems faced by ADABOOST when outliers are present and how an ensemble can be used to detect these outliers. These techniques were also compared to the separate-and-conquer methodology which focusses the algorithm on ambiguous examples. Example ambiguity can be defined as the level of disagreement between the votes of an ensemble, such as the diversity measures described in Section 3.3.2. This section explores the concept of a boosting style algorithm that focuses on ambiguous examples, rather than those that are commonly misclassified.

To minimise the training error of a binary classification problem, the distribution over the data, w_t, should be chosen to alter the hypothesis distribution, θ, such that the number of examples with probability of correct classification greater than 0.5 is maximised. These examples are expected to be classified correctly by a majority vote ensemble and this can be written,

$$\max_{w_t} \left\{ \sum_{i=1}^{N} I\left[P_\theta\left(h_\theta\left(x_i\right) = y_i\right) > 0.5\right] \right\}. \tag{4.8}$$

This maximises the number of examples for which the probability of correct classification is greater than 0.5. This should minimise the training error as examples which fulfill this criteria are expected to be classified correctly by an aggregated ensemble.

The distribution, w_t, can be applied to the base learners within boosting methods through resampling of the data (Drucker and Cortes, 1996). This technique trains each learner on a subset of the data, which is formed through sampling N examples with replacement from the data of size N. However, the sampling probabilities are generated according to the values of w_t. In this case, the probability of correct classification can be decomposed into two terms which are defined by whether or not the example in question was included in the sampled subset. This subset can be referred to as the bagged subset for hypothesis h_t and is denoted as $\mathcal{X}_t^{BAG}$. These probabilities can be written as,

$$P_1(x_i; w_t) = P_\theta\left(h_\theta(x_i) = y_i | x_i \in \mathcal{X}_t^{BAG}\right) \tag{4.9}$$

and

$$P_2(x_i; w_t) = P_\theta\left(h_\theta(x_i) = y_i | x_i \notin \mathcal{X}_t^{BAG}\right). \tag{4.10}$$

The probabilities are dependent upon w_t as this distribution affects the distribution over the hypothesis space, θ. The probability of example x_i being included in the bagged subset is,

$$P_\theta\left(x_i \in \mathcal{X}_t^{BAG}\right) = 1 - \left(1 - \frac{w_t(x_i)}{\sum_{j=1}^N w_t(x_j)}\right)^N \tag{4.11}$$

and the probability of not being included is,

$$P_\theta\left(x_i \notin \mathcal{X}_t^{BAG}\right) = 1 - P_\theta\left(x_i \in \mathcal{X}_t^{BAG}\right). \tag{4.12}$$

The probability of correct classification can then be written,

$$P_\theta(h_\theta(x_i) = y_i) = P_1(x_i; w_t) P_\theta\left(x_i \in \mathcal{X}_t^{BAG}\right) + P_2(x_i; w_t) P_\theta\left(x_i \notin \mathcal{X}_t^{BAG}\right). \tag{4.13}$$

Assuming that $P_1(x_i; w_t) > P_2(x_i; w_t)$, Equation 4.8 can be written,

$$\max_{w_t}\left\{\sum_{i=1}^N I\left[\frac{w_t(x_i)}{\sum_{j=1}^N w_t(x_j)} > 1 - \left(1 - \frac{0.5 - P_2(x_i; w_t)}{P_1(x_i; w_t) - P_2(x_i; w_t)}\right)^{\frac{1}{N}}\right]\right\}. \tag{4.14}$$

The inequality for each example is dependent upon w_t and on the probabilities, $P_1(x_i; w_t)$ and $P_2(x_i; w_t)$, which are determined by the distribution over the hypothesis space, θ. It can be seen that to reduce the required value of $w_t(x_i)$, the values of $P_1(x_i; w_t)$ and $P_2(x_i; w_t)$ should minimise,

$$\frac{0.5 - P_2(x_i; w_t)}{P_1(x_i; w_t) - P_2(x_i; w_t)}. \tag{4.15}$$

As the derivative of this is,

$$\frac{\partial}{\partial\left[P_2(x_i; w_t)\right]}\left\{\frac{0.5 - P_2(x_i; w_t)}{P_1(x_i; w_t) - P_2(x_i; w_t)}\right\} = \frac{0.5 - P_1(x_i; w_t)}{\left(P_1(x_i; w_t) - P_2(x_i; w_t)\right)^2}, \tag{4.16}$$

and assuming that $P_1(x_i; w_t) > 0.5$, the required value of $w_t(x_i)$ is smaller for larger values of $P_1(x_i; w_t)$ and $P_2(x_i; w_t)$. This is intuitively correct as less focus needs to be given to examples which are easier to classify. As $P_2(x_i; w_t)$ is the probability of correct classification when the example is not included in the training set, the value of $P_2(x_i; w_t)$ represents the generalisation ability of the learners on that example. If $P_2(x_i; w_t) > 0.5$, then the example is expected to be classified correctly and a weight of zero for this example is sufficient.

From Equation 4.14 it can be seen that it is advantageous to increase the weights of each example so that the inequality is true for as many examples as possible. However, it can also be seen from the left hand side of the inequality that the sum of all of the weights should be minimised, which indicates that weights should not be larger than they need to be. Therefore, the weights assigned to each example should be chosen according to their values of $P_1(x_i; w_t)$ and $P_2(x_i; w_t)$. Also, examples which are especially awkward to classify may require an exceptionally large value of $w_t(x_i)$. This may not be feasible as the required value may increase the total of the weight vector to such an extent that it prevents the correct classification of other examples. These examples can be considered outliers and it is better to assign them very small values and accept that they will be misclassified. Therefore, a suitable weighting scheme should assign higher weights to examples which are harder to classify and have low values of $P_1(x_i; w_t)$ and $P_2(x_i; w_t)$, but should assign low weights to examples with extremely low values for these probabilities. The method proposed here is to implement a boosting style algorithm, which gives higher weights to more ambiguous examples rather than ones that are commonly misclassified. This should promote the correct type of diversity by improving the performance on awkward examples without being misled by outliers.

The relative weight given to example x_i by the ADABOOST algorithm at iteration $T+1$ is,

$$w_{T+1}(x_i) = \frac{1}{N} \exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)\right). \tag{4.17}$$

If this measure was replaced by one based upon ambiguity rather than common misclassification, one possible candidate would be to remove the influence of the class, y_i, from Equation 4.17,

$$w_{T+1}(x_i) = \frac{1}{N} \exp\left(-\text{abs}\left(\sum_{t=1}^{T} \alpha_t h_t(x_i)\right)\right). \tag{4.18}$$

This would then relate the weighting to the confidence estimate $\widehat{H}(x_i)$, given by Equation 3.29. Other explicit measures of example-based diversity could also be used such as entropy, discussed in Chapter 3. However, the entropy of examples would equal zero if all of the hypotheses were in agreement which would be undesirable as a weighting coefficient. Also, the diversity measure needs to account for the number of predictions that have been made. Measures that are based purely on the proportions of class predictions would be very unreliable in the initial stages of the algorithm. A suitable quantity for weighting the examples should not only measure their degree of ambiguity, but also account for the reliability of this information.

An approach that was introduced by Freund (1995) weights the examples according to,

$$w_{t+1}(x_i) = \begin{pmatrix} T-t-1 \\ \lfloor \frac{T}{2} \rfloor - H_t^+(x_i) \end{pmatrix} \left(\frac{1}{2}+\tau\right)^{\lfloor \frac{T}{2} \rfloor - H_t^+(x_i)} \left(\frac{1}{2}-\tau\right)^{\lceil \frac{T}{2} \rceil - t - 1 + H_t^+(x_i)}, \tag{4.19}$$

where $H_t^+(x_i)$ symbolises the number of correct votes for example x_i in the current ensemble of size t,

$$H_t^+(x_i) = \sum_{t'=1}^{t} I\left(h_{t'}(x_i) = y_i\right). \tag{4.20}$$

The method fixes the ensemble size prior to construction and makes the assumption that the learner errors do not exceed $\frac{1}{2} - \tau$. The weighting scheme then represents the probability that at the final iteration, example x_i requires exactly 1 more correct vote. The weighting for the final iteration is uniform for all such examples and zero otherwise. Examples will receive a weight of zero if the number of correct and incorrect votes is such that it is impossible for the ensemble decision to change.

The following describes a measure that is based on the likelihood of example ambiguity (Rogers and Gunn, 2007). For binary classification, and under the assumption that the base hypotheses are generated independently, the predictions of the hypotheses can be viewed as being generated from a binomial distribution, with probability of correct classification given by,

$$P(H(x) = y) = \sum_{t=\lceil T/2 \rceil}^{T} \left(P_\theta(h_\theta(x) = y)\right)^t \left(1 - P_\theta(h_\theta(x) = y)\right)^{T-t} \begin{pmatrix} T \\ t \end{pmatrix}. \tag{4.21}$$

A measure of confidence in the predictions of a majority vote ensemble is the observed estimate of the margin,

$$marg_{est}(x_i, y_i) = \frac{1}{T} \sum_{t=1}^{T} y_i h_t(x_i). \tag{4.22}$$

The significance of this measure is dependent on the proportions of correct and incorrect predictions and on the size of the ensemble, T. For a more interpretable measure, a sign test can be performed. Given T predictions, the sign of the estimate of the margin represents the ensemble prediction. It is then possible to calculate a measure of likelihood that the true margin contradicts the observed value. The null hypothesis can be set up to represent this case,

$$\operatorname{sgn}\{marg_{est}(x_i, y_i)\} \neq \operatorname{sgn}\{marg(x_i, y_i)\}. \tag{4.23}$$

The maximum likelihood estimate of $P_\theta(h_\theta(x_i) = y_i)$ within the null hypothesis is given by 0.5. Given an example x_i, with a positive observed margin and H_T^+ correct predictions, the likelihood of the null hypothesis is,

$$L\left[P_\theta(h_\theta(x_i) = y_i) < 0.5\right] = \sum_{t=0}^{T-H_T^+} 0.5^T \begin{pmatrix} T \\ t \end{pmatrix}. \tag{4.24}$$

A similar quantity can be expressed for an example with a negative observed margin.

In this second case, the quantity would represent the likelihood that the true margin is positive. An example for which this likelihood is negligible can be considered an outlier.

The general measure adopted here is the likelihood that the true margin is zero and that the observations have occurred by chance. As these weights are normalised to form a distribution, the multiplication by the constant 0.5^T can be ignored.

$$w_{T+1}(x_i) = L\left[P_\theta\left(h_\theta(x_i) = y_i\right) = 0.5\right] = \sum_{t=0}^{\min\{H_T^+, T-H_T^+\}} \frac{\binom{T}{t}}{Z_t} \tag{4.25}$$

Examples for which this quantity is small can be considered as having a significant degree of confidence in their predictions. This can then be used to implement a separate-and-conquer strategy by setting a threshold on this quantity and removing examples accordingly. Alternatively, a boosting style approach can be implemented which weights the examples by this quantity and therefore, emphasises the more ambiguous examples. If the examples are weighted by this measure, then the assumption that the learners are generated independently is violated. The learners are drawn from different hypothesis distributions, according to the distribution w_t. However, this method makes the assumption that there is one single hypothesis distribution from which all of the learners are drawn. As a measure for examining the relative ambiguity in the predictions of the examples, this is a reasonable approximation.

As the algorithm proceeds, the size of the ensemble, T, increases and the difference between the potential maximum weight and potential minimum weight will also become larger. Therefore, it is possible that the weights of the ambiguous examples will continue to increase and dominate the distribution, which could result in the algorithm overfitting. However, this is not expected to occur as the method should control the weights in the following manner. If the weight assigned to example x_i increases, the probability of correct classification of the example also increases. As before, this is assuming that $P_1(x_i) > P_2(x_i)$. Therefore, the expected number of correct predictions will also increase, which will reduce the ambiguity of the example and result in a lower relative value of $w_t(x_i)$. Conversely, as the weight assigned to example x_i decreases, the probability of correct classification also decreases. If example x_i was initially classified confidently and correctly, then the reduction in $w_t(x_i)$ will result in a lower number of correct predictions. This will increase the ambiguity of the example and result in a larger value of $w_t(x_i)$. However, if the example was initially predicted confidently and incorrectly, the lower value of $w_t(x_i)$ will result in even fewer correct predictions. This will further decrease the ambiguity of example x_i and result in an increasingly lower weight. Therefore, the weights assigned to the examples that are considered outliers will continue to decrease as the algorithm progresses, whilst the weights assigned to the remaining examples are not expected to dominate the distribution.

Another consideration is the choice of base learner. In order to effectively measure the ambiguity, it is important that the base learners do not overfit. The ambiguity is measured on the predictions of the training data and, as this is used to train the learners, a bias is introduced. Outliers are expected to be misclassified by the learners most of the time, but if the learners overfit, they may often be classified correctly. Therefore, the requirement that the learners do not overfit is necessary in order to minimise the bias that is imposed on the estimates of the margin for each example in the training data. This requirement can be avoided if an unbiased estimate of the margin can be found. When using Bagging to form the training set for each learner, such an estimate is available through the out-of-bag data. The training examples can be tested on all of the learners for which they were not used to train. However, when comparing the relative ambiguity of examples, this method is unsuitable as each example will be in the out-of-bag set for only a subset of the learners. The size of the subset will vary between examples and introduce a bias on the quantity of Equation 4.25.

The quantity used to weight each of the examples is given by Equation 4.25, but a simpler and more cautious measure would be to use the binomial coefficient,

$$w_{T+1}(x_i) = \frac{\binom{T}{H_T^+}}{Z_t}. \tag{4.26}$$

The methods of updating w according to Equations 4.25 and 4.26 are referred to as DivBoost W1 and DivBoost W2 respectively, and the following pseudo-code describes these methods.

Interestingly, this method can be viewed as performing the opposite operation as that of the Decorate algorithm (Melville and Mooney, 2003), described in Chapter 3. Their technique adds artificial outliers into the data to promote diversity, but these are generated for each learner. Therefore, the artificial data does not result in ensembles that make consistent mistakes. The DivBoost methods promote diversity in a different way and avoid the consistent mistakes that are associated with outliers that exist in the input data.

4.3 Feature Manipulation

Ensemble diversity can be promoted through utilising different learning algorithms and exploiting the variation in their learning biases. This section explores the bias that is imposed by decision tree methods and highlights the possible problems that this bias creates. A new method is then introduced that exploits this bias to promote diversity by manipulating the feature space.

Algorithm 1: Divboost

```
Input        Training Data $X, Y = (x_1, y_1), ..., (x_N, y_N)$
             Test Data $X' = x'_1, ..., x'_M$

Initialise   $w_1(x_1, ..., x_N) = \left(\frac{1}{N}, ..., \frac{1}{N}\right)$
             $H^+(x_1, ..., x_N) = (0, ..., 0)$
             Number of Learners e.g. $T = 100$

Algorithm    For $t = 1$ to $T$
                 $\mathcal{X}^{BAG}$ = Sample $N$ examples from $X, Y$ using $w_t$
                 Train $h_t$ using $\mathcal{X}^{BAG}$
                 Classify $X$ using $h_t$
                 $H^+(x_i) = H^+(x_i) + 1 \ \forall\ i : h_t(x_i) = y_i$
                 If Using Divboost W1
                     $w_{t+1}(x_i) = \sum_{j=0}^{\min\{H^+(x_i), T - H^+(x_i)\}} \binom{t}{j} \ : \ i = 1$ to $N$
                 Else If Using Divboost W2
                     $w_{t+1}(x_i) = \binom{t}{H^+(x_i)} \ : \ i = 1$ to $N$
                 Normalise $w_{t+1}$
                 Classify $X'$ using $h_t$
             end.
             $H(x'_i) = \mathrm{sgn}\left(\sum_{t=1}^{T} h_t(x'_i)\right) \ : \ i = 1$ to $M$

Output       Test Data Predictions $H(x'_1, ..., x'_M)$
```

4.3.1 The Decision Tree Bias

Typically, decision trees are constructed by recursive partitioning using single features. Only considering single features reduces the search space that is traversed by the tree induction process, making their construction simpler and quicker. This is particularly advantageous for ensemble algorithms where multiple trees are required. However, the resultant hypotheses then consist of sets of axis-parallel partitions, which bias the classification of areas of the space. Figure 4.2 illustrates a binary classification problem and two possible decision tree hypotheses. It should be noted that the problem can be solved by simply placing limits on the features such that the hypothesis becomes,

$$\begin{array}{lll} h(x) = +1 & : & x(1) < 0.25 \text{ or } x(1) > 0.75 \\ h(x) = +1 & : & x(2) < 0.25 \text{ or } x(2) > 0.75 \\ h(x) = -1 & : & \text{otherwise} \end{array} \tag{4.27}$$

Due to the rotational symmetry of the problem, these trees are equally likely to be produced. Figure 4.3 shows the areas where the two trees agree (white) and disagree

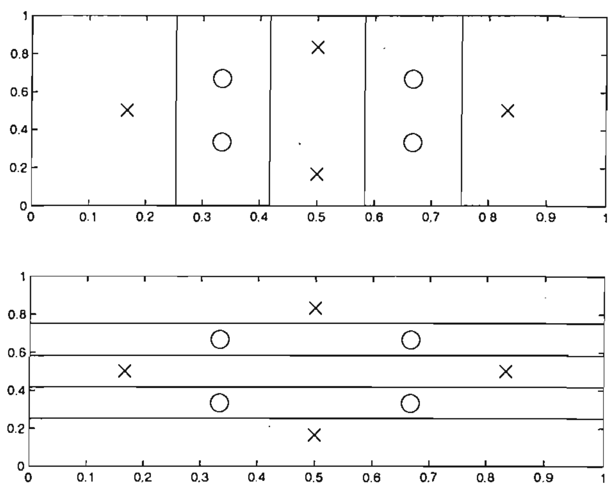

FIGURE 4.2: Binary classification example with two possible decision tree hypotheses.

(shaded). From this figure, the effect of the decision tree bias is clearly visible. Some regions that have been classified in agreement, and therefore confidently, contain no data and are surrounded by ambiguous regions which represent the hypothesised class boundary.

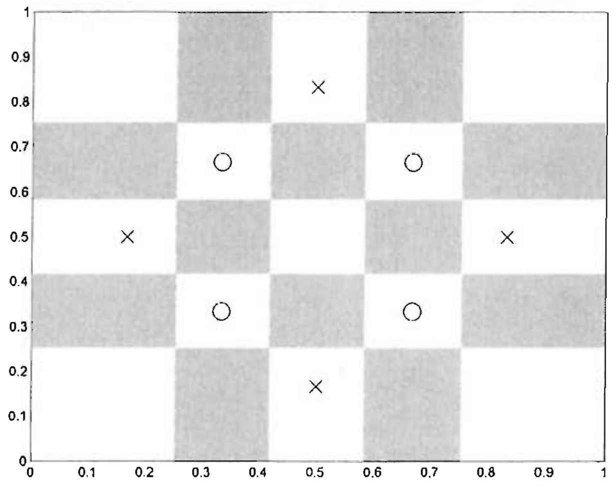

FIGURE 4.3: Binary classification example showing the disagreement between the two decision trees.

Employing Random Forest on this problem results in the hypothesis of Figure 4.4. Despite the randomisation of the feature selection process, the learning bias is still clearly evident. This hypothesis has an overly high complexity, as it describes one class with nine disjoint regions, induced from only four examples. Although this is only a toy example, it highlights the potential problems that can exist with ensembles of decision trees. It is also possible that this type of problem can occur within a local area of the space for some data sets.

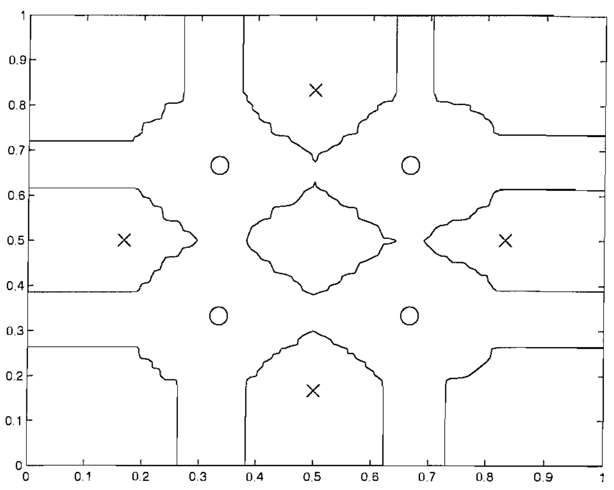

FIGURE 4.4: Binary classification example with hypothesis induced by a Random Forest consisting of 500 trees.

4.3.2 Random Rotation

To avoid the potential complexity that is imposed by the decision tree bias, the tree induction algorithm can be altered so that partitions involving subsets of features are considered. Heath et al. (1993) use simulated annealing to search for such a suitable split and Breiman (2001) tried random feature combinations as part of Random Forest construction. Instead of adapting the algorithm to allow partitions that are not axis-parallel, Rodriguez and Alonso (2004) manipulate the feature space to exploit the decision tree bias and promote diversity within an ensemble. Their Rotation Forest technique performs a principal components analysis (PCA) of feature subsets which are randomly grouped for each learner. The data is then transformed by projection onto the full set of components. By using all of the components, the projection amounts to an axis-rotation of the data, which then alters the bias imposed by decision tree algorithms. As this is performed on random feature groups for each learner, a variety of axis-rotations are produced and the resultant decision trees are quite diverse. Rodriguez et al. (2006) compared the accuracy and diversity of the method to that of simply constructing decision trees in conjunction with the Bagging algorithm. The learners of the Rotation Forest were found to be slightly more accurate and slightly more diverse and the small increase in these quantities yields a large increase in the accuracy of the overall ensemble.

PCA can be used to order the features according to how much of the variance in the data they explain. In some sense, this rotation can be considered optimal as it enables the tree induction process to search for these more potent features. However, the diversity is created from randomly grouping the features and therefore, the overall rotation would not be the same as if the PCA was performed on the full set. Also, PCA is an unsupervised technique which transforms the data based on the directions of maximal variance. This is not necessarily the transformation which enables the optimal partitioning of the classes.

The approach introduced here is a variation of this method, which forms random rotations of the data to exploit the decision tree bias and promote ensemble diversity. The features are randomly paired and a random angle is chosen in the range $[0, \frac{\pi}{2}]$. A rotation of $\frac{\pi}{2}$ simply swaps the axes, so there is no benefit in going beyond this value. A rotation matrix can then be constructed. The following matrix can be used to rotate the data about features 1 and 2 through angle β,

$$R = \begin{pmatrix} \cos(\beta) & \sin(\beta) & 0 & .. & .. & 0 \\ -\sin(\beta) & \cos(\beta) & 0 & .. & .. & 0 \\ 0 & 0 & 1 & 0 & .. & 0 \\ 0 & 0 & 0 & 1 & .. & 0 \\ \vdots & \vdots & \vdots & \vdots & & 0 \\ 0 & 0 & 0 & 0 & .. & 1 \end{pmatrix}. \tag{4.28}$$

The data, X, can then be rotated to a new representation,

$$X^r = XR^T. \tag{4.29}$$

As the rotation is only conducted through feature pairs, the set of possible orientations is limited. However, if the transformed data is kept and rotated in the same manner in the next iteration, the space of possible orientations is enlarged. The rotation of a pair of features results in two new features, which are a combination of the original pair. If one of these features is then rotated with an entirely different feature, the resultant pair is then a combination of the original three. As this continues, the features being created consist of combinations of larger subsets of the original features.

The following pseudo-code describes this algorithm,

Instead of trying to find an optimal rotation of the data, or attempting to develop more potent features through the combination of existing ones, this method simply trains each learner on a random transformation of the data. By not using a single rotation consistently, this method avoids the potential problems of the decision tree bias. To demonstrate this, the previous example of Figure 4.4 is used again to train a Random Forest of 500 trees. However, each learner in this forest is given a different rotation of the data and the final induced hypothesis is shown in Figure 4.5. The hypothesis now appears more reasonable, as it describes only two disjoint regions.

The method proposed here is to perform random rotation with standard decision tree algorithms. Unlike other feature manipulation algorithms, such as Random Subspace or Random Forest, this technique does not discard any feature information. Therefore, the accuracy of the learners created by this method can be expected to be greater. This technique promotes diversity through the exploitation of the decision tree bias

Algorithm 2: Random Rotation

```
Input        Training Data $X, Y = (x_1, y_1), ..., (x_N, y_N)$
             Test Data $X' = x'_1, ..., x'_M$

Initialise   Number of Learners e.g. $T = 100$

Algorithm    For $t = 1$ to $T$
                 $\mathcal{F} = \{X_1, ..., X_F\}$
                 $R = F$ by $F$ identity matrix
                 For $f = 1$ to $\lfloor \frac{F}{2} \rfloor$
                     $X_a, X_b$ =Randomly pick feature pair from $\mathcal{F}$
                     $\mathcal{F} = \mathcal{F} - \{X_a, X_b\}$
                     $\beta$ = Choose random angle in range $[0, \frac{\pi}{2}]$
                     Update corresponding elements of $R$:
                         $R\begin{pmatrix} (a,a) & (b,a) \\ (a,b) & (b,b) \end{pmatrix} = \begin{pmatrix} \cos(\beta) & \sin(\beta) \\ -\sin(\beta) & \cos(\beta) \end{pmatrix}$
                 end.
                 $X = XR^T$
                 Train $h_t$ using $X$
                 $X' = X'R^T$
                 Classify $X'$ using $h_t$
             end.
             $H(x'_i) = \text{sgn}\left(\sum_{t=1}^{T} h_t(x'_i)\right) \; : \; i = 1$ to $M$

Output       Test Data Predictions $H(x'_1, ..., x'_M)$
```

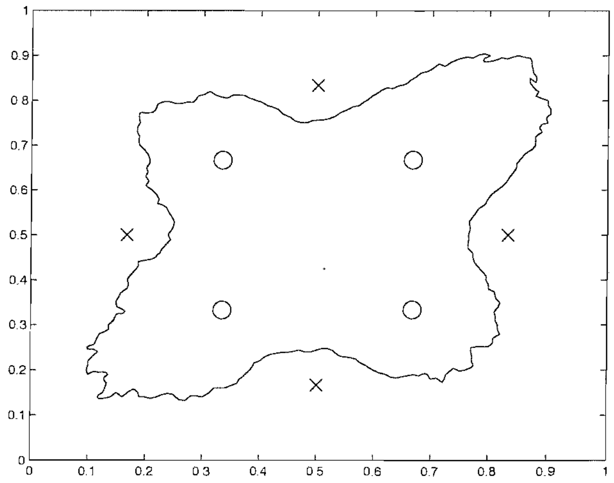

FIGURE 4.5: Binary classification example with hypothesis induced by a Random Forest consisting of 500 trees with data rotation.

and its rotation variant nature. Therefore, the potential for ensemble improvement only exists when this type of method is used in conjunction with rotation variant learners. No improvement would be found with rotation invariant learners such as Nearest Neighbour, unless the distance metric was replaced with one that was rotation variant.

4.4 Conclusions

This chapter has considered boosting in the context of diversity promotion, and conjectured that boosting methods promote diversity through concentrating on the awkward examples. The concept of example ambiguity has been defined and related to the example-based diversity measures of Chapter 3. The effect of outliers on the ADABOOST algorithm has been discussed and this motivates the idea of focussing on ambiguous examples. This new approach was related to separate-and-conquer style algorithms, which were shown to implicitly perform outlier detection. Following this intuition, a new boosting-style algorithm was introduced that characterises example ambiguity in terms of the margin function.

The bias of decision trees was discussed, along with the potential problems that it creates. An ensemble algorithm was proposed that manipulates the feature space through random rotation. This technique has the capacity to exploit the decision tree bias and can avoid some of its associated problems.

Chapter 5 conducts some experiments with these methods and investigates their performance as diversity promotion techniques.

Chapter 5

Empirical Evaluation of Diversity Promotion Techniques

5.1 Datasets

The properties of the data sets used in these experiments are shown in Table 5.1. The Wisconsin Breast Cancer (WBC), Pima Diabetes, Sonar, Ionosphere and Votes are available from the UCI Repository (Blake and Merz, 1998).

TABLE 5.1: Data Set Properties

Data Set	No. Examples	No. Features
WBC	683	9
Pima	768	8
Sonar	208	60
Ionosphere	351	34
Votes	435	16
Friedman	200	10

The Friedman dataset, (Friedman, 1991) is an artificial data set that contains five irrelevant features and additive gaussian target noise. It is generated according to the following formula and has been thresholded in order to convert it into a binary classification problem. A threshold value of 14 was chosen to yield a reasonably balanced data set.

$$Y = 10\sin(\pi X_1 X_2) + 20\left(X_3 - \frac{1}{2}\right)^2 + 10X_4 + 5X_5 + \mathcal{N}(0, 1.0) \tag{5.1}$$

5.2 Diversity Promotion Through Data Manipulation

The base learners used here are decision trees that use Information Gain as the node split criterion. They are pruned with a method that is based on the discretisation technique of Fayyad and Irani (1993), which adopts the minimum description length principle and is described in more detail in Section 2.4.2. After the construction of each tree, it is used to classify the training data and update the weights for each example. These weights are applied to the data in the same manner as Drucker and Cortes (1996), such that each learner is constructed on a sample of the training data where each example is selected randomly, with replacement and with probability,

$$P(x_i) = \frac{w(x_i)}{\sum_{i=1}^{N} w(x_i)} \tag{5.2}$$

Four experiments are conducted. Bagging is used to generate an ensemble of trees, where the sampling distribution of the data is kept uniform throughout and the final ensemble prediction is given by a majority vote. ADABOOST is implemented by updating the sampling distribution according to Equation 2.8 and weighting each vote by the weight given in Equation 2.11. These methods are compared to the two DIVBOOST variants, where the sampling distribution is updated according to Equations 4.25 or 4.26. In both of these cases the learners are combined through a majority vote.

Each data set is randomly partitioned into 90% for training and 10% for testing, and this is repeated over 100 trials. The results are averaged and shown in Table 5.2. The average error of each learner in the ensemble is also recorded, along with the averaged pairwise Q-statistics for each method. These results are shown in Table 5.3 and Table 5.4. The Q-statistics measure the average correlation between the base learners and are calculated according to,

$$Q(h_1, h_2) = \frac{|\mathcal{X}^{11}|\,|\mathcal{X}^{00}| - |\mathcal{X}^{01}|\,|\mathcal{X}^{10}|}{|\mathcal{X}^{11}|\,|\mathcal{X}^{00}| + |\mathcal{X}^{01}|\,|\mathcal{X}^{10}|}. \tag{5.3}$$

Here, $\mathcal{X}^{ab}$ represents the set of examples which satisfy the conditions of a and b, where $a = 1$ if h_1 is correct and $a = 0$ if it is incorrect. Similarly, $b = 1$ if h_2 is correct and $b = 0$ if it is incorrect.

TABLE 5.2: Error rates when employing ensemble algorithms with decision tree base learners. The values in brackets are the corresponding variances of test error over the 100 trials.

Data Set	Bagging	Adaboost	Divboost W1	Divboost W2
WBC	0.0345(0.0005)	0.0284(0.0003)	0.0319(0.0004)	0.0325(0.0005)
Sonar	0.1890(0.0072)	0.1324(0.0045)	0.1295(0.0060)	0.1329(0.0056)
Votes	0.0450(0.0009)	0.0498(0.0010)	0.0416(0.0007)	0.0395(0.0007)
Pima	0.2586(0.0025)	0.2570(0.0017)	0.2383(0.0023)	0.2458(0.0021)
Ionosphere	0.0606(0.0016)	0.0686(0.0015)	0.0617(0.0016)	0.0614(0.0015)
Friedman	0.1885(0.0068)	0.1485(0.0059)	0.1505(0.0063)	0.1445(0.0057)

DIVBOOST can be seen to compare favourably with ADABOOST as it is significantly more accurate on several data sets and is only beaten on two.

TABLE 5.3: Averaged pairwise Q-statistic measured over the test data when employing ensemble algorithms with decision tree base learners.

Data Set	Bagging	Adaboost	Divboost W1	Divboost W2
WBC	0.6858	0.3977	0.3185	0.3381
Sonar	0.3904	0.0723	0.0958	0.0939
Votes	0.7464	0.1748	0.1365	0.1325
Pima	0.7797	0.0217	0.0269	0.0409
Ionosphere	0.6970	0.1595	0.2288	0.2368
Friedman	0.5276	0.0460	0.0310	0.0228

TABLE 5.4: Average base learner error when employing ensemble algorithms with decision tree base learners.

Data Set	Bagging	Adaboost	Divboost W1	Divboost W2
WBC	0.0512	0.1157	0.1775	0.1672
Sonar	0.2816	0.3692	0.3514	0.3552
Votes	0.0543	0.2337	0.2644	0.2618
Pima	0.2817	0.4047	0.4416	0.4336
Ionosphere	0.0981	0.2532	0.2258	0.2165
Friedman	0.2458	0.3839	0.4079	0.4093

From examination of Table 5.3 and Table 5.4, it can be seen that the underlying mechanisms of ADABOOST and DIVBOOST are quite similar. Both algorithms concentrate on the awkward examples and increase the ensemble diversity at the cost of increased error rates in the base learners. In contrast, Bagging generates much more accurate base learners that are less diverse. Therefore, the reduction in error is less significant in Bagging.

Figure 5.1 shows how the average Q-statistic changes as the ensemble is constructed in conjunction with ADABOOST and DIVBOOST W1. Separate plots are given for each data set and they illustrate the diversity measured with respect to the training and test data. Apart from the Pima data set, the training correlation of the learners constructed using ADABOOST begins low and gradually rises with increasing ensemble size. This is due to ADABOOST attempting to correct its previous mistakes on the training data. Therefore, the method produces very different hypotheses in the initial stages of the algorithm. In contrast, the correlation between learners constructed with DIVBOOST tends to gradually decrease as the algorithm proceeds. This tendency is due to DIVBOOST progressively focussing on the ambiguous training examples. The plots illustrating the correlation, measured with respect to the test data, show a pronounced increase in diversity as DIVBOOST proceeds, demonstrating the utility of this algorithm as a diversity promotion technique.

Both of the DIVBOOST methods perform significantly better than ADABOOST on the

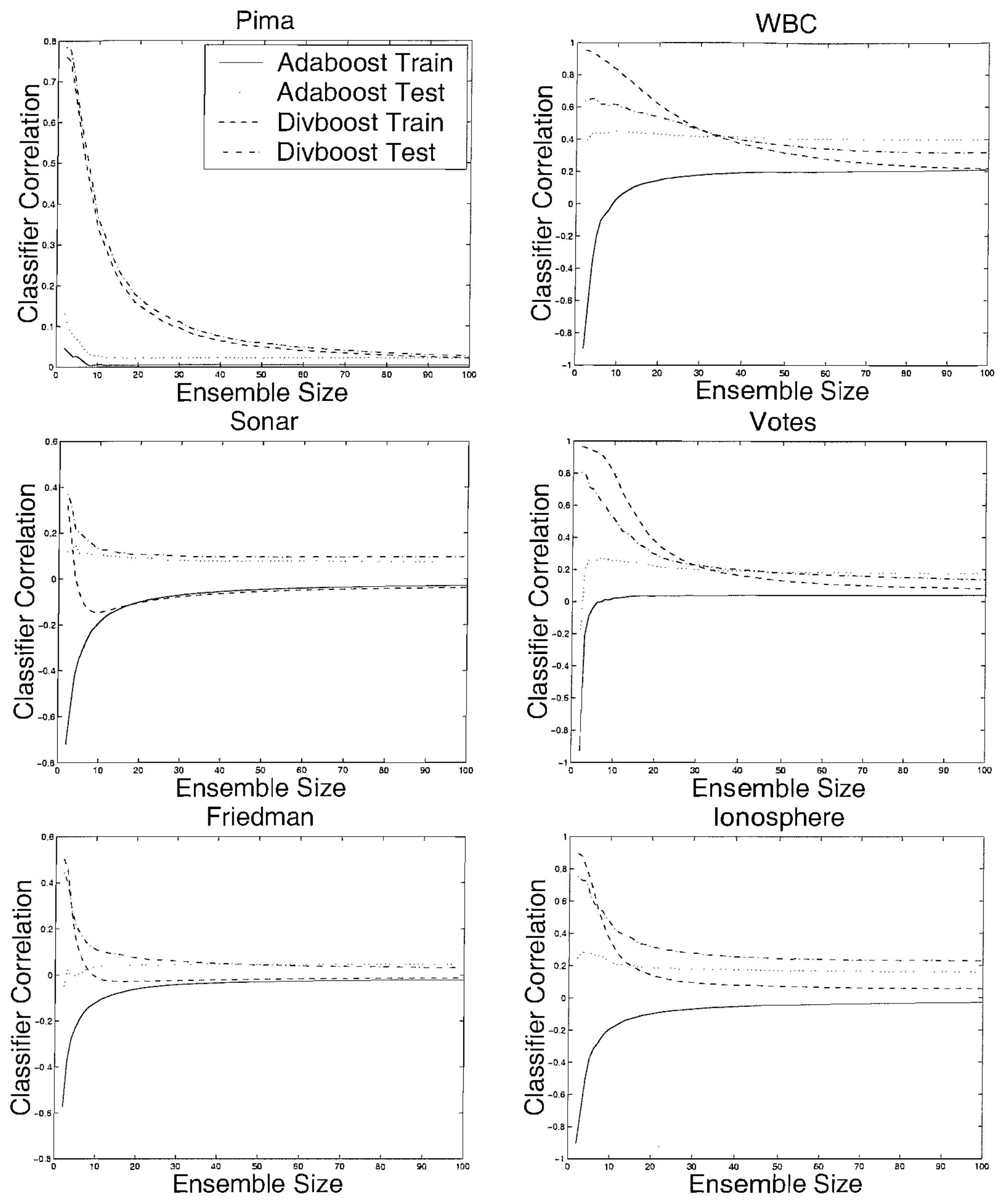

FIGURE 5.1: Plots showing variation of average Q-statistic with ensemble size for ADABOOST and DIVBOOST. Correlation is measured with respect to training and test data.

Pima data set. This data set is noisy and it is known that ADABOOST can be intolerant to noise. Figure 5.2 compares the average final weights given to the data by ADABOOST and DIVBOOST. It is easily seen that ADABOOST gives very large weights to some examples which are given very low weights by DIVBOOST. These are the examples which are misclassified most often and could be considered outliers. DIVBOOST achieves better generalisation on this data set because it does not concentrate on these examples but rather focuses on the ambiguous ones.

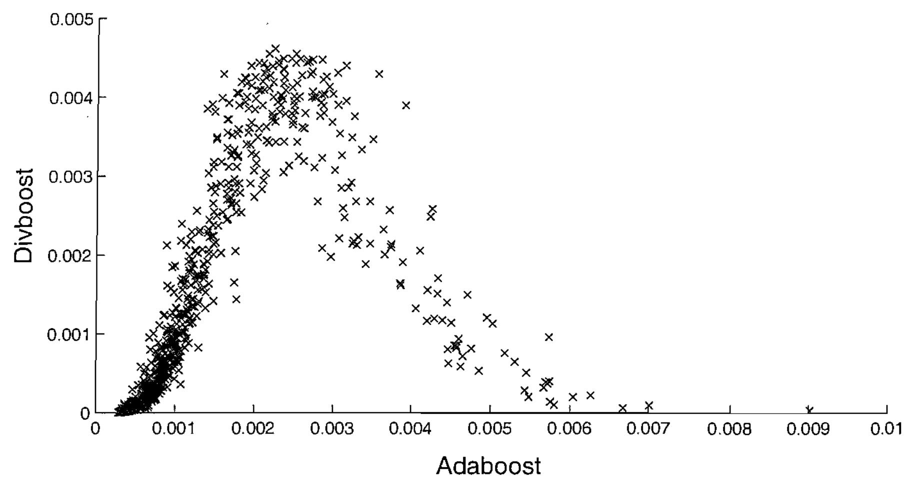

FIGURE 5.2: Comparison of the average final weighting given by ADABOOST and DIVBOOST for the Pima data set.

5.3 Diversity Promotion Through Feature Manipulation

The experiments conducted here examine the potential of Random Rotation as a diversity promotion technique. The method is compared against the other feature manipulation techniques of Random Subspace and Random Forest. As Random Forest incorporates Bagging as part of its algorithm, the same is also done for Random Subspace and Random Rotation. A Bagged sample of data is used to train each base learner. Random Subspace selects a random subset of the features for each of these samples and projects the data onto this set before constructing a decision tree. Random Rotation creates a new representation of the data for each sample by performing a rotation of the data, as described in Chapter 4. The results for Bagging without any feature manipulation are also given as a basis for comparison.

As before, the data is randomly partitioned into 90% for training and 10% for testing and this is repeated over 100 trials. Ensembles of 100 learners are constructed and the results are given in Table 5.5. Random Rotation can be seen to compare favourably to the other techniques as it gives the best performance on 4 of the 6 data sets.

The average error of the base learners within these ensembles is given in Table 5.7 and the average pairwise Q-Statistic is given in Table 5.6. Bagging tends to create the most accurate base learners, but also the most correlated. In particular, Bagging creates accurate learners for the Votes and Friedman data sets. It is known that Votes contains a large number of redundant features and that half of the features in the Friedman data set are irrelevant. By not manipulating the feature space, Bagging can keep these features separate whilst the other algorithms are forced to use them. Random Subspace and Random Forest will often disregard the useful features whilst randomly selecting less potent or completely irrelevent ones. Random Rotation can dilute the potency of a useful feature by randomly combining it with a less useful one.

Table 5.5: Error rates of ensemble algorithms when using Bagging in conjunction with feature manipulation techniques. The values in brackets are the corresponding variances of test error over the 100 trials.

Data Set	Bagging	Random Subspace	Random Forest	Random Rotation
WBC	0.0345(0.0005)	0.0290(0.0003)	0.0293(0.0004)	0.0278(0.0004)
Sonar	0.1890(0.0072)	0.2033(0.0082)	0.1671(0.0066)	0.1705(0.0068)
Votes	0.0450(0.0009)	0.0457(0.0009)	0.0586(0.0012)	0.0486(0.0008)
Pima	0.2586(0.0025)	0.2683(0.0024)	0.2525(0.0022)	0.2414(0.0020)
Ionosphere	0.0606(0.0016)	0.0633(0.0015)	0.0756(0.0016)	0.0486(0.0011)
Friedman	0.1885(0.0068)	0.1895(0.0066)	0.1670(0.0070)	0.1650(0.0069)

Apart from the Votes and Friedman data sets, the average base learner error of random rotation is comparable to that of Bagging. However, Random Rotation tends to create more diversity within the ensemble, which results in a better overall performance.

Table 5.6: Averaged pairwise Q-statistic when using Bagging in conjunction with feature manipulation techniques.

Data Set	Bagging	Random Subspace	Random Forest	Random Rotation
WBC	0.6858	0.6339	0.6051	0.6908
Sonar	0.3904	0.3323	0.1267	0.2653
Votes	0.7464	0.4747	0.4728	0.4993
Pima	0.7797	0.6255	0.4466	0.7471
Ionosphere	0.6970	0.6243	0.4326	0.4110
Friedman	0.5276	0.1586	0.1120	0.2365

Table 5.7: Average base learner error when using Bagging in conjunction with feature manipulation techniques.

Data Set	Bagging	Random Subspace	Random Forest	Random Rotation
WBC	0.0512	0.0632	0.0645	0.0420
Sonar	0.2816	0.2985	0.3518	0.3094
Votes	0.0543	0.0920	0.1226	0.0944
Pima	0.2817	0.3099	0.3405	0.2868
Ionosphere	0.0981	0.1216	0.1708	0.1389
Friedman	0.2458	0.3343	0.3646	0.3082

The method of Rodriguez et al. (2006), described in Chapter 3, introduces diversity into an ensemble of trees by randomly grouping the features and performing a principal component analyis (PCA) on each group. By including all of the components, this method amounts to a rotation of the data. The intuition behind this method is that the different rotations will promote diversity, whilst the employment of PCA should produce more relevant features and, therefore, more accurate hypotheses. Table 5.8 compares this method, denoted PCA rotation, to the random rotation technique and it can be seen that random rotation performs better than PCA rotation on 5 of the 6 data sets tested here. However, it should be noted that PCA rotation out-performs Bagging, Random Subspace and Random Forest on 4 of the 6 data sets. The table also shows the average base learner accuracy and Q-statistic for these methods. These quantities show how the ensembles produced by PCA rotation consist of learners that tend to be slightly more accurate and slightly more correlated than those produced by random rotation. PCA is a dimensionality reduction technique that can remove redundancy in data. This would explain why PCA rotation performs better than random rotation on the Votes data set, as this data contains a large amount of redundancy.

TABLE 5.8: Comparison of PCA and Random Rotation techniques. The table shows the ensemble error (Error), average base learner error (B.E.) and average Q-statistic (Q-Stat) for both methods.

Data Set	**PCA Rotation**			**Random Rotation**		
	Error	**B.E.**	**Q-Stat**	**Error**	**B.E.**	**Q-Stat**
WBC	0.0278(0.0004)	0.0421	0.6842	0.0278(0.0004)	0.0420	0.6908
Sonar	0.1876(0.0086)	0.2910	0.3172	0.1705(0.0068)	0.3094	0.2653
Votes	0.0418(0.0007)	0.0585	0.5875	0.0486(0.0008)	0.0944	0.4993
Pima	0.2458(0.0030)	0.2744	0.7726	0.2414(0.0020)	0.2868	0.7471
Ionosphere	0.0561(0.0013)	0.1247	0.4801	0.0486(0.0011)	0.1389	0.4110
Friedman	0.1905(0.0092)	0.3081	0.2643	0.1650(0.0069)	0.3082	0.2365

As the diversity element of PCA rotation is produced by randomly grouping the features, it is possible that the obtained level of diversity is limited by the dimensionality of the data. PCA rotation groups the features into sets of 3. For a data set of dimensionality F, this method results in $\begin{pmatrix} F \\ 3 \end{pmatrix}$ possible feature subsets, each creating 3 new features. Therefore, PCA rotation creates a possible $3\begin{pmatrix} F \\ 3 \end{pmatrix}$ new features, from which a subset is selected and used to train each learner. The diversity that is introduced by this method may be limited by this number of possible features, which increases quickly with the dimensionality of the data. In contrast, there is no limit to the number of features that can be created by the random rotation technique. Therefore, PCA rotation may perform better with data of higher dimensionality, especially if this data contains a large amount of redundancy.

5.4 Conclusions

This chapter has demonstrated the importance of diversity within a learning ensemble and compared several algorithms for promoting this. The DIVBOOST variants promote diversity through manipulation of the relative importance of each example and compare favourably to Bagging and ADABOOST. The methods focus the algorithm on the regions of the class boundaries and are shown to be more robust to the presence of outliers than

AdaBoost. This outlier tolerance comes from the similarities between this method and the approaches of separate-and-conquer and outlier identification. The technique promotes diversity in a robust way as the algorithm is not misled by outliers, which could have a detrimental effect on the accuracy of the base learners.

The Random Rotation algorithm exploits the decision tree bias to create more diversity than Bagging alone and performs well when compared to Random Forest and Random Subspace. Unlike these techniques, Random Rotation does not discard any feature information and the induced learners are typically more accurate as a result. One of the problems faced by feature manipulation techniques is that of the presence of irrelevant features. By randomising the feature representation of the data, the effect of these features is increased. For Random Forest and Random Subspace, the feature selection process is randomised, which increases the likelihood of relying on the irrelevant features. Random Rotation can reduce the potency of some features by rotating them with irrelevant ones. The data manipulation algorithms do not suffer as badly from the presence of irrelevant features, as they maintain the original feature representation. When using decision tree learners, the implicit feature selection that the algorithm incorporates at each node of construction avoids features with low class discrimination.

Chapter 6

Feature Selection

6.1 Introduction

One of the important areas of machine learning research is that of selecting an optimal set of the available features. The motivation behind this is partly due to the computational load that is imposed upon many learning algorithms by higher dimensional data. The way in which execution time scales with the number of features varies between learning algorithms, and in some cases can be poor. For certain situations, the accuracy of a classification algorithm can be significantly reduced with the presence of irrelevant features. This can be a consequence of the learning algorithm such as in nearest neighbour, which has no implicit feature selection and has shown significant degradation in accuracy with an increased number of irrelevant features (Almuallim and Dietterich, 1991). However, it can also be caused by an effect known as *The curse of dimensionality* (Bellman, 1961). This problem is a result of the increased hyper-volume of the space with increased dimensionality, which results in an increase in the amount of data required to sustain a given spatial density. Therefore, for high-dimensional data there may be an insufficient number of examples to describe the target function.

The interpretability of an induced hypothesis is also a desirable attribute for certain applications, and this benefits greatly from the employment of a reduced set of features. The different motivations behind feature selection can alter the definition of what the optimal subset should be and result in different methods for selection. The criteria may be based on classification accuracy, computational complexity, interpretability or a compromise between some or all of these factors.

6.2 Irrelevant Features and Random Forest

The Random Forest algorithm makes no distinction between the relevance of features during construction of the forest. As the features are selected randomly and with equal probability at each node, the performance can suffer significantly from the presence of irrelevant features (Rogers and Gunn, 2006). Standard decision tree algorithms will select the optimal feature at each split in terms of maximal information gain. As Random Forest lacks this implicit feature selection, the probability of selecting an irrelevant feature increases with the proportion of irrelevant features present. These irrelevant features can then mislead the algorithm and increase the generalisation error. Also, as irrelevant features are not effective at separating the data, they can result in unnecessarily large trees and therefore, an increased computational load.

An explanation for the effect of irrelevant features on the Random Forest algorithm can be found by considering the space of possible hypotheses. As previously discussed, one of the explanations for how ensemble methods work centres around the concept of the margin. The margin for example x, with label y, is the difference between the probability of correct classification and of being classified as belonging to the next most likely class. For binary classification, the margin is simply,

$$marg\left(x,y\right)=2P_{\theta}\left(h_{\theta}\left(x\right)=y\right)-1, \qquad (6.1)$$

where θ represents the environment in which hypothesis h, was constructed. This includes the bagged set of examples that were used to construct h and the features that were chosen.

The probability of classifying a data point correctly converges to either 1 or 0 depending on the sign of the margin and Breiman (2001) uses the law of large numbers to prove that the misclassification rate of an ensemble H, converges asymptotically to the probability over the input space of obtaining an example with a negative margin. However, for a finite number, T, of hypotheses, the probability of correct classification is given by,

$$P\left(H\left(x\right)=y\right)=\sum_{t=\lceil T/2\rceil}^{T}\left(P_{\theta}\left(h_{\theta}\left(x\right)=y\right)\right)^{t}\left(1-P_{\theta}\left(h_{\theta}\left(x\right)=y\right)\right)^{T-t}\begin{pmatrix}T\\t\end{pmatrix}. \qquad (6.2)$$

As more base hypotheses are added to the ensemble, the error rate converges in line with this function. Therefore, the speed of convergence can be increased by increasing the size of the margin. For Random Forest the diversity within the base hypotheses is introduced through the combination of bagging, which trains the hypothesis on a random subset of the training data, and random input selection (Ho, 1998a). The space of possible base hypotheses is affected by the set of features chosen to represent the data. As a consequence of this, feature selection can alter the margin values of the data. Ideally, this should result in fewer data points having a negative margin and allow the algorithm to converge to a smaller error rate. But, it also has the ability to increase the size of the margin values and result in faster convergence. Therefore, fewer trees may be needed, which lowers the computational requirement.

6.3 Feature Relevance

Many algorithms adopt the principle of selecting the subset of features which carry the most information concerning the target, such as Almuallim and Dietterich (1991), with their algorithm FOCUS. Designed to be used for data with binary features, the technique identifies the smallest set of features for which no two examples exist that agree on all the features but not on the class. This enables complete discrimination between the classes, but could easily fail when noise is present as the selected subset can be affected by a single example. By preferring smaller subsets, the method implements what is termed the *min-features bias*, which although useful for some applications, can be inappropriate for certain data.

The FOCUS algorithm deems features to be relevant if they are essential for discriminating between all examples of different classes. For continuous data, the class conditional probabilities can be examined to form different definitions of feature relevance. As described by John et al. (1994) and Kohavi and John (1997), one such definition is that a feature X_i is relevant if there exists values x_i and y assigned to X_i and the target Y respectively, such that,

$$P(Y = y|X_i = x_i) \neq P(Y = y). \tag{6.3}$$

Intuitively, this appears logical as knowledge concerning a relevant feature is expected to alter the distribution of the target. It is equivalent to defining a feature as relevant if there is a significant degree of correlation between the feature and the class. Consequently, there are many algorithms which attempt to identify relevant features by calculating some measure of correlation between the feature and the target. Information theoretic measures such as information gain are ideal for this purpose as they are not limited to linear aspects of correlation and can identify non-linear relationships. Roobaert et al. (2006) use information gain in this way to measure the correlation between the features and the target, and select the features with a correlation value greater than some threshold.

There are problems with using this simple approach. One is that this definition of feature relevance does not account for redundancy within the feature subset. Information concerning the target that is repeated in different features is only required once. A possible extension to this type of algorithm is to approximate the redundancy by measuring the pairwise correlation between the features in the subset. Battiti (1994), Yu and Liu

(2004a), Hall (2000) and Koller and Sahami (1996) adopt this unsupervised approach using various measures of correlation.

Here, we define $\rho(X_i, Y)$ as the measure of correlation between a feature X_i and the class Y, and the corresponding measure of correlation between a pair of features X_i and X_j is given by $\rho(X_i, X_j)$. The algorithm of Yu and Liu (2004a) then attempts to find the largest feature subset that contains no two features, X_i and X_j for which,

$$\rho(X_i, Y) \geq \rho(X_j, Y) \tag{6.4}$$

and

$$\rho(X_i, X_j) \geq \rho(X_j, Y). \tag{6.5}$$

The CFS algorithm of Hall (2000) searches for the feature subset, S, of F features that maximises the merit function,

$$Merit(S) = \frac{\sum_{i:X_i \in S} \rho(X_i, Y)}{\sqrt{F + \sum_{i:X_i \in S} \sum_{j:X_j \in S, i \neq j} \rho(X_i, X_j)}}. \tag{6.6}$$

Lee et al. (2006) form clusters by grouping features with high values of $\rho(X_i, X_j)$. The feature with the greatest value of $\rho(X_i, Y)$ from each of the clusters is selected as a representer. The number of clusters is a parameter that must be chosen by the user.

As already mentioned, the linear correlation coefficient is unable to identify non-linear relationships within the data, but is implemented in Koller and Sahami (1996). Another way of measuring the degree of correlation between pairs of features is the *Maximum Information Compression Index* (Mitra, 2002). The measure is based on the smallest eigenvalue of the covariance matrix of the data, when projected onto the subspace of the feature pair. If this value is zero then the data lies in a one-dimensional subspace and the features are linearly dependent. A benefit of this method is that it is insensitive to rotation. However, it can still only identify linear relationships within the data.

Information theoretic measures can be used to identify non-linear correlation between features, such as information gain, IG. This was described in Chapter 2 as a node split criterion. Here, the notation used for information gain is slightly different to reflect the fact that it is being used to measure the correlation between two variables.

$$IG(X_1, X_2) = Ent(X_1) - \sum_{i:l_i \in \mathcal{L}_2} \frac{|\mathcal{X}_{l_i}|}{N} Ent(X_1, \mathcal{X}_{l_i}) \tag{6.7}$$

where $Ent(X_1, \mathcal{X}_{l_i})$ defines the entropy of variable X_1 over the subset of data $\mathcal{X}_{l_i}$. If no subset is specified, the full set of data is assumed. The set of partitions for feature X_2 is denoted as $\mathcal{L}_2$.

These type of measures require discretisation of continuous features. As simple uniform partitioning may not produce a meaningful representation of the data, different methods exist for identifying suitable intervals (Dougherty et al., 1995). The number of intervals created by these methods can be data dependent and therefore, different features may have a different number of partitions. Information gain is biased in favour of features with more partitions and is not a fair measure if used to identify the relative importance of features which have been discretised into a dynamically chosen number of partitions, as above. For this reason, Gain Ratio, GR, can be used instead of information gain.

$$GR(X_1, X_2) = \frac{IG(X_1, X_2)}{Ent(X_2)} \tag{6.8}$$

This measure divides the information gain by the entropy of the feature, but is not suitable for measuring pairwise feature correlation because it is not symmetric. The properties of symmetry and invariance to the number of feature partitions makes the symmetrical uncertainty, SU, measure ideal for this purpose (Press et al., 1988). This measure divides the information gain by the average of the two variable entropies and is used in Yu and Liu (2004a), Yu and Liu (2004b) and Hall (2000).

$$SU(X_1, X_2) = 2\left(\frac{IG(X_1, X_2)}{Ent(X_1) + Ent(X_2)}\right) \tag{6.9}$$

Although the pairwise correlation between features can reveal redundancy within them, the process does not discriminate between the useful information, which is helpful in predicting the class, and other correlations in the data. Yu and Liu (2004b) make an improvement to their algorithm of Yu and Liu (2004a) by examining the combined feature correlation to the class, instead of pairwise feature correlation. This technique then eliminates a feature, X_j, if there exists another feature, X_i, for which,

$$\rho(X_i, Y) \geq \rho(X_j, Y) \tag{6.10}$$

and

$$\rho(X_i, Y) \geq \rho((X_i, X_j), Y), \tag{6.11}$$

where $\rho((X_i, X_j), Y)$ denotes the correlation between the target Y and the combination of the two features, X_i and X_j. This method deems the feature, X_j, to be relevant with respect to another feature, X_i, if the combination of the two features contains more information about the target than is contained within X_i alone. As well as being able to identify information that is shared between pairs of features, this method can also detect interaction between them. A feature that shows very little correlation to the class may contain a large amount of information when combined with other features. John et al. (1994) and Kohavi and John (1997) illustrate this point with the parity problem.

Example 6.1. *If the target, Y is given by the exclusive OR of the binary features, X_1*

and X_2, then Y is fully described by the features and they are both relevant.

$$Y = X_1 \oplus X_2$$

However, if each feature assumes the values of 1 or 0 with equal probability, then there appears to be no correlation between either of the features individually and the class. This is because knowing the value of one of the features gives no information about the target without knowing the value of the other feature.

Friedman (1994) describes strongly interacting features as being locally predictive. In this sense, measuring the degree of correlation between the feature and the class can be viewed as a global measure. A feature may only be relevant within a particular area of the input space, which has been determined by the values of other features. In Example 6.1, the features appear locally relevant when considered within areas of the input space that have been restricted by the other feature. This idea of local feature relevance is highlighted in the example of the Multiplexor (Apte et al., 1997).

Example 6.2. *A multiplexor has a set of control variables which dictate which of the input channels is produced at the output. For the multiplexor shown in Figure 6.1, the input variables X_1 and X_2 are the control variables and X_3 to X_6 constitute the input channels. A set of data can be created by this model, by randomly assigning values to the input variables and recording the output of the multiplexor as the target, Y. For any*

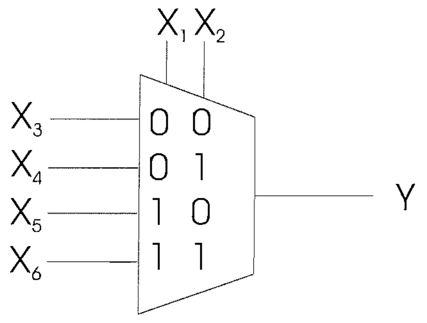

FIGURE 6.1:

given example, the target is given by one of the input channels, which can be determined by the two control variables. Therefore, only three features are relevant within a given area of the space. The input channels are locally relevant features, where the locality is defined by the control variables.

Hong (1997) explores this idea by measuring feature relevance according to its local ability. For each example, the algorithm identifies the K nearest examples of the opposite class. The feature relevance measure is then the sum of the distances between each example and the K counter-examples, along the feature in question. Although this method examines the local relevance of the feature, the K nearest counter examples are chosen with respect to all of the features. Therefore, this concept of locality can be corrupted by the presence of irrelevant features. A similar approach is adopted by Domingos (1997) to improve the nearest neighbour algorithm through local feature selection. For each training example, this technique identifies the nearest neighbour of the same class and searches for any features which differ. In the case of continuous-valued features, examples are considered different if their values differ by more than one standard deviation of the samples of that feature. If any such features are found, they are removed from the considered example. Cross-validation is then performed to determine the effect of the removal on the accuracy of the classifier and, if the classifier performance has been worsened, the change is reversed.

Within the algorithms of Hong (1997) and Domingos (1997) the concept of locality is defined by the neighbourhood of each example. However, Howe and Cardie (1997) measure the relevance of each feature with respect to a given class. Their method uses categorical features and constructs a distribution for each one. This distribution takes the form of a vector containing the relative frequencies of the possible values of that feature. Such a distribution can then be created over the subset of data belonging to each class. The distribution of feature X_f, for class y, is designated as $\Psi\left(X_f, y\right)$. The relevance of a feature to a particular class is measured as the ability of that feature to discrmininate between whether or not examples belong to that class. Their relevance measure is then given as the difference between the distribution taken over the class in question, and of that taken over the rest of the data,

$$Rel\left(X_f, y\right) = \left\|\Psi\left(X_f, y\right) - \Psi\left(X_f, \mathcal{Y} - y\right)\right\|_1, \tag{6.12}$$

where $\Psi\left(X_f, \mathcal{Y} - y\right)$ is the distribution of feature X_f, calculated with respect to examples belonging to all classes other than y.

John et al. (1994) and Kohavi and John (1997) formalise these ideas into the definitions of strong and weak relevance. If S_i is the subset of all of the features apart from X_i and s_i is a value assignment to those features, then X_i is strongly relevant if there exists some x_i, y and s_i such that,

$$P\left(Y = y | X_i = x_i, S_i = s_i\right) \neq P\left(Y = y | S_i = s_i\right). \tag{6.13}$$

Removing a strongly relevant feature from the set will result in a loss of information about the target.

Weak relevance is used to describe features that carry information about the target, but which is repeated in other features. Unlike strongly relevant features, if a weakly relevant feature is removed, no information about the target is lost. X_i is weakly relevant if it is not strongly relevant and there exists some subset, S_i' of S_i and values y, x_i and s_i' such that,

$$P\left(Y = y | X_i = x_i, S_i' = s_i'\right) \neq P\left(Y = y | S_i' = s_i'\right). \tag{6.14}$$

These definitions of feature relevance are now sufficient to identify redundancy and cope with feature interaction. This theory then leads to the concept of identifying Markov Blankets (Koller and Sahami, 1996). A Markov Blanket can be defined in the following way. If S_i' is some subset of the features, such that $X_i \notin S_i'$, and S_i is the subset of remaining features, that excludes S_i' and X_i, then S_i' is a Markov Blanket for X_i if there are no value assignments to the variables such that,

$$P\left(S_i = s_i, Y = y | X_i = x_i, S_i' = s_i'\right) \neq P\left(S_i = s_i, Y = y | S_i' = s_i'\right). \tag{6.15}$$

The optimal feature subset is then the minimal subset that forms a Markov Blanket for the remaining features. Due to the fact that feature interaction can occur amongst any number of the available features, searching for a Markov Blanket is a combinatorial problem and prohibitive for high dimensional data. Yu and Liu (2004a), Yu and Liu (2004b) and Koller and Sahami (1996) attempt to approximate Markov Blankets in their feature selection algorithms, but use correlation based measures to simplify the problem. A Markov Blanket for a feature incorporates all of the information that the feature contains. This information not only concerns the target, but all of the other features as well. Therefore, the attractive property of Markov Blanket identification is that features can be eliminated recursively. If a subset of features, S_i, forms a Markov Blanket for a feature, X_i and another subset, S_j, forms a Markov Blanket for X_j and $X_j \in S_i$, then $S_i \cup S_j - \{X_j\}$ also forms a Markov Blanket for X_i. This means that, provided a feature is removed only when a corresponding Markov Blanket is discovered, a feature that has been removed cannot become relevant again as more features are eliminated.

6.4 Decision Tree Based Methods

Decision tree methods, such as CART (Breiman et al., 1984), select the feature at each node in the tree which yields the optimal partition according to some split criterion and, therefore, contain an implicit form of feature selection. If such a tree is pruned in a way that yields an induced hypothesis with a good generalisation ability, then the features that were used in the tree construction can be considered relevant. Cardie (1993) uses C4.5 trees to identify relevant features in exactly this way. The stopping criterion given in Equation 2.24 uses the minimum description length principle to associate a cost with describing the features that a tree utilises. This cost biases the algorithm to prefer features which have been used previously in the construction of the tree and minimises the number of features chosen. Frey and Fisher (2003) demonstrated that an improvement could be found if instead of simply selecting the chosen features, the selection is based upon the frequency with which each feature is selected. This method was proposed as a way of identifying Markov Blankets and has certain advantages over correlation based methods. Each node in the tree represents a region of the input space that is bounded by values along some of the features. All of the nodes, for which the particular node is a direct descendant, form these boundaries when they are split. Consequently, the feature that is selected to perform a split anywhere in the tree, is optimal only for the local area (Friedman, 1994). This exploitation of local relevance can then use features in the context of other features and therefore, aids in the identification of feature interaction. Also, redundancy can be removed, as features are less likely to be chosen if the information that they contain has already been applied by the subset of features that were used to form the path from the root. Apte et al. (1997) employ a decision tree approach to partition the data and identify regions of the input space for which different features are relevant. They use correlation measures to construct vectors that represent the relative values of feature importance within a region of the input space. The relevance, $Rel\left(\mathcal{X}_a, X_f\right)$, of feature X_f can be calculated for a region of the input space that contains the subset of the data, $\mathcal{X}_a$. The vector, $M\left(\mathcal{X}_a\right)$, then consists of the relevance measures for all of the features within region, $\mathcal{X}_a$,

$$M\left(\mathcal{X}_a\right) = \left[Rel\left(\mathcal{X}_a, X_1\right), Rel\left(\mathcal{X}_a, X_2\right) ... Rel\left(\mathcal{X}_a, X_F\right)\right]. \tag{6.16}$$

They introduce the concept of an Importance Profile Angle IPA, which quantifies the difference in feature relevance between two disjoint regions of the input space.

$$IPA\left(\mathcal{X}_a, \mathcal{X}_b\right) = \cos^{-1}\left(\frac{M\left(\mathcal{X}_a\right)^T M\left(\mathcal{X}_b\right)}{\left\|M\left(\mathcal{X}_a\right)\right\| \left\|M\left(\mathcal{X}_b\right)\right\|}\right) \tag{6.17}$$

The motivation for this technique is that learning problems that contain locally relevant features can be broken down to yield separate sub-problems that contain consistent measures of feature relevance. A decision tree is used to create the separate regions and any learning algorithm can then be applied to the sub-problems.

Due to feature interaction, the actual optimal split at any point of a tree may not result in good class discrimination, but partition the data in such a way that subsequent features can separate the classes. As these methods select a feature on the basis of how accurately it partitions the current node and do not examine subsequent splits, the optimal feature may not be selected. Another limitation with decision tree based methods is that the number of employed features is restricted by the number of examples in the training data.

6.5 Ensemble Feature Selection

When applying feature selection for ensemble algorithms, the definition of feature relevance is not solely motivated by the reduction of dimensionality and the inclusion of target information. As previously discussed, ensemble algorithms require diversity amongst the base learners and therefore, the criteria for selecting the features should incorporate this. The optimal feature subset should be comprised of features, which not only contain information concerning the target with a minimal quantity of redundancy, but must also promote diversity. Typically, feature selection for ensemble learners is embedded within the ensemble construction process. Multiple learners can be created on random feature subsets using the random subspace method (Ho, 1998a). From this set, a subset of accurate and diverse learners can be chosen to represent the ensemble. An example of this was proposed by Opitz (1999), which measured the fitness of each learner in terms of the accuracy and diversity. Other algorithms use a search method on the set of features for each learner to increase the accuracy of each one and then select a subset of these learners to maximise the diversity, (Cunningham and Carney, 2000; Oliveira et al., 2003). These methods suggest that effective feature selection for ensemble learners should not involve the identification of a single global subset, but combine multiple diverse representations of the data. The FEATUREBOOST algorithm (O'Sullivan et al., 2000) attempts to find such diverse models by forcing the algorithm to consider features that have not been relied upon to a great extent. The measurement of diversity can be defined as the ambiguity in the target predictions. These methods then select learners, which explore the areas of the input space that have high ambiguity in the target information. This is similar to the method of boosting, which concentrates the construction of the base learners on the examples which are commonly misclassified and therefore, also explores the ambiguous areas of the space.

As previously discussed, some features may be relevant within a local area of the input space. This local relevance can be identified and exploited by ensembles that create diversity by varying the employed set of features. Puuronen and Tsymbal (2001) adopt this approach by utilising the random subspace method to explore various feature subsets and altering the integration method. Each vote is weighted according to the performance of the hypothesis within the vicinity of the example being classified. Cross-validation is performed to obtain the performance estimate of each model on each example and the locality is defined by the K nearest neighbours. This integration method was applied to an ensemble of naïve bayes learners (Tsymbal et al., 2002). Although this was found to yield an improvement over simple aggregation of the ensemble, the ability of this type of ensemble to identify local feature relevance is limited. This is due to the naïve bayes learner assuming independence between the features when conditioned on the class, as discussed in Section 2.4.3. Strong interactions between features are not expected to be identified when utilising this base learner.

Puuronen and Tsymbal (2001) also select which hypotheses to combine for each test case. This is achieved through the construction of a decision tree and examination of the relevant branch. Section 6.4 discussed the ability of decision trees to uncover feature interaction. For a particular example, the features that are perceived to be relevant are the ones that comprise the path from the root to the appropriate leaf. For each test example, their method only combines the votes of hypotheses that were trained on a feature subset that includes the features along the relevant branch of the tree.

6.6 Wrapper Methods

Feature selection algorithms can broadly be categorised into two types, Filter and Wrapper methods (Kohavi and John, 1997). Filter methods tend to examine the structure of the data, in order to identify the relevant features efficiently. However, Wrapper methods perform a search through the possible feature subsets and utilise the learning algorithm to compare their abilities. As discussed by Kohavi and John (1997), if the aim is to maximise the generalisation ability of the algorithm, then for certain algorithms and certain data sets, the optimal feature subset may not contain all of the relevant features and may contain some irrelevant features. Therefore, the argument for employing Wrapper methods is that unlike Filter methods, the Wrapper incorporates the bias of the learning algorithm for which the selected feature subset is intended. Performing an exhaustive search through the possible feature subsets is computationally expensive and infeasible for many data sets. For this reason a search is conducted through this space, which only requires the testing of a limited number of subsets but hopefully identifies the optimal subset. Forward selection is one example of this, where initially there are no features selected and at each stage, the single feature is added that increases the accuracy of the algorithm the most. Backward elimination begins with the full set of features and removes the single feature at each stage that reduces the accuracy the least. Both of these methods can become stuck in local minima, as the algorithm terminates when no further improvement is possible. To overcome this problem, the algorithms can be modified to include a degree of backtracking. Skalak (1994) use Random Mutation Hill Climbing (RMHC), which randomly adds or removes a feature at each step and keeps the change if it improves the generalisation ability of a nearest neighbour algorithm.

The simplest evaluation of the feature subsets involves the generalisation error rate. For variable cost problems, a more extensive analysis of the performance of the hypothesis is required and typically involves examination of the Receiver Operating Characteristic (ROC) curve. This curve shows the effect on class error rates of imposing different biases on the algorithm. The PARCEL algorithm (Scott et al., 1998a), performs a random exploration of possible feature subsets and uses them to construct classifiers. These classifiers are then deemed to be useful if they extend the convex hull of the ROC space, using the theory of Maximum Realisable ROC (MRROC) (Scott et al., 1998b).

6.7 Feature Weighting

Some algorithms do not attempt to select a subset of the available features, but try to assign a degree of relevance to each one. Each feature can then be relied upon to a varying extent. Nearest Neighbour algorithms can easily be modified to incorporate this varying importance by re-scaling the input space. Assigning weights to each of the features is not as beneficial to the interpretability of the induced hypothesis and does not reduce the dimensionality. It is usually employed in incremental or embedded algorithms where feature selection is desirable but only a small amount of information concerning the features is available.

By enabling varying degrees of relevance to be assigned to the features instead of 0 and 1, as in selection, the size of the search space is dramatically increased and can lead to over-fitting. Kohavi et al. (1997) showed that weighting the features for a nearest neighbour algorithm rarely produces a more accurate hypothesis than selection and can do significantly worse.

6.8 Conclusions

This chapter has explained the need for feature selection when using Random Forest, in terms of accuracy and computational load. The effects that can be created by irrelevant features on Random Forest have been explored and a theoretical justification has been given for the particular effect on error convergence. A review of some current feature selection techniques has been conducted and their ability at relevance identification has been examined. The chapter has explored the different definitions of feature relevance and has found that the identification of an optimal subset can be complicated by interactions within the data. Performing feature selection for ensembles presents an added problem, as there is a requirement to promote diversity amongst the base hypotheses. Chapter 7 will review and introduce some feature relevance identification techniques using Random Forest to enable random subset exploration and local relevance exploitation.

Chapter 7

Random Forest Methods

7.1 Introduction

Due to the complex relationships that can exist between features, the identification of an optimal set is a challenging problem. To best explore these relationships it is necessary to evaluate subsets of features rather than individuals. The Random Forest algorithm builds a large number of simple classifiers using randomly chosen features and therefore, achieves a good exploration of possible feature subsets. Unlike the random subspace method (Ho, 1998a), random forest enables the evaluation of multiple feature subsets within an individual learner. As Bagging is employed as an integral component of the algorithm, the out-of-bag data enables evaluation of the feature subsets without the need for an independent test set. Also, these measures of feature importance incorporate the bias of decision tree induction, which is beneficial if they are applied to decision tree techniques. For these reasons, Random Forest lends itself to feature selection well.

7.2 Relevance Identification with Random Forests

7.2.1 Out-of-Bag Estimation

Breiman (2001) gave a method for identifying feature relevance using Random Forest through examination of the out-of-bag error rate. Random Forest employs Bagging to form the training sample with which to construct each tree. Typically, this selects about $\frac{2}{3}$ of the entire training data, as explained in Chapter 2. Each training example can then be classified by all of the trees in the forest that did not include the example in their training set. The out-of-bag error rate is then the average error of all of the training examples, classified in this way. To measure the amount of information contained within a feature, the values of the feature are randomly permuted among all of the examples.

The out-of-bag error rate is then calculated again and the difference represents the amount of information that was lost by the feature permutation. This method can identify all of the predictive information in the feature, including information that is caused by feature interaction. However, it is important to note that this is not equivalent to a Wrapper method, that removes the feature and re-trains the algorithm. When the algorithm is initially trained, any information that is contained in the features will be used for construction of the hypothesis and relied upon for future classification. After feature permutation, the algorithm is not re-trained and the predictive information that was contained within the feature will continue to be falsely relied upon. Any useful information will cause degradation in accuracy upon permutation, even if it is redundant and therefore, this method is unable to identify redundancy. Svetnik et al. (2004) use this method for feature selection, however, Chen and Lin (2006) suggest that the method is computationally expensive for high dimensional data and apply a pre-processing technique that examines the individual performance of each feature. Again, this pre-processing can eliminate features with strong interaction.

7.2.2 Evaluation of Feature Importance in Tree Construction

At each node in the construction of a Random Forest, a feature is selected randomly and used to split the node and maximise the information gain. As already mentioned, the information gain can be used as a measure of correlation between the feature and the class. Borisov et al. (2006) use these measures of feature importance to increase the performance of the learning algorithm. Although these measures appear to be simpler forms of information gain, there are some benefits to using this method over standard information gain. As discussed in Section 6.4, each terminal node in a decision tree can be viewed as a learner that has been trained on the features that were used in the path from the root. Consequently, the information gain values are not simply measures of the individual feature performance, but measures of the ability of the feature in a variety of possible feature subsets. Figure 7.1 shows how these feature subsets are formed in the construction of the tree. In this example feature X_3 is tested on data that has previously been partitioned on features X_1 and X_2. It is clear from the XOR example that if the data was split using one of the features and then split again using the other feature, that the second split would reveal the relevance of the feature. This is consistent with the idea of local relevance (Friedman, 1994), and therefore, there is some allowance for relationships between the features with this method.

The problem with using the average information gain achieved by each feature, is that some nodes in the tree are easier to split than others. The number of ways a node can be split is determined by its composition, and the measure of information gain is clearly more unreliable when partitioning smaller nodes. The following describes a measure to weight each value of information gain according to its reliability (Rogers and Gunn,

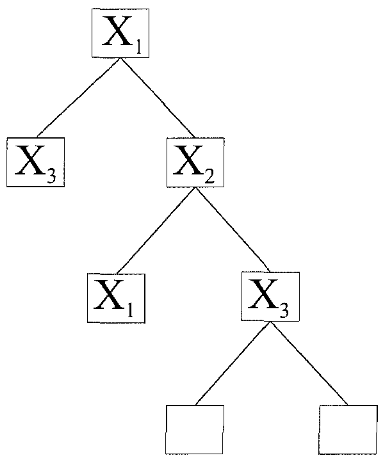

FIGURE 7.1: Example of possible tree showing which features were used at each node.

2005).

7.2.3 A Node Complexity Measure

This measure attempts to assign a value of reliability to a node by examining its composition and calculating the information associated with the splitting of such a node. For every split, the data is projected along a single feature. The assumptions made here are that once the data is projected into the one dimensional space, no data points lie on top of one another and that during the split optimisation procedure, all possible splits are found. Figure 7.2 shows the possible split positions for one node once the data is projected into the one dimensional space.

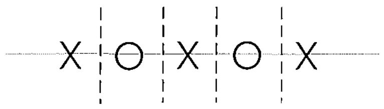

FIGURE 7.2: Illustration of possible splits in one dimensional space of a node consisting of 3 data points from one class and 2 from the other.

The problem is now a matter of considering how many possible arrangements of the data are possible. For binary classification problems, $|\mathcal{X}_l|$ is the number of examples contained in the node and $|\mathcal{X}_{l+}|$ is the number of positive examples. The number of possible arrangements is given by the combinatorial function,

$$\begin{pmatrix} |\mathcal{X}_l| \\ |\mathcal{X}_{l+}| \end{pmatrix} = \frac{|\mathcal{X}_l|!}{|\mathcal{X}_{l+}|!\,(|\mathcal{X}_l| - |\mathcal{X}_{l+}|)!} \tag{7.1}$$

However, some of these arrangements are merely reflections of each other and will, therefore, result in the same optimised information gain. For example, a node containing only two examples, one of each class, will have two possible arrangements. The optimal split value is the same in both cases and this node can yield only one information gain

TABLE 7.1: Number of unique arrangements for node

$\|\mathcal{X}_l\|$	$\|\mathcal{X}_{l+}\|$	A_u
EVEN	**EVEN**	$\binom{\|\mathcal{X}_l\|/2}{\|\mathcal{X}_{l+}\|/2}$
ODD	**ODD**	$\binom{(\|\mathcal{X}_l\|-1)/2}{(\|\mathcal{X}_{l+}\|-1)/2}$
ODD	**EVEN**	$\binom{(\|\mathcal{X}_l\|-1)/2}{\|\mathcal{X}_{l+}\|/2}$
EVEN	**ODD**	0

value. Using the assumptions stated above, this example would be split perfectly by all features and would result in a maximum information gain value. Therefore, this illustrates the need for effective weighting of the nodes.

Not all of the arrangements have a reflected twin because some arrangements are symmetrical about their centre. These symmetrical arrangements shall be referred to as unique and their frequency designated by A_u. Their counterparts shall be referred to as non-unique and their frequency can be written $\binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|} - A_u$. A_u can be calculated by considering the arrangements of half of the data and then taking the reflection to form the other half. This technique is dependent upon whether the values of $|\mathcal{X}_l|$ and $|\mathcal{X}_{l+}|$ are odd or even and the corresponding functions are given in Table 7.1.

The probability of a random occurrence of a particular unique arrangement, U_x is simply the probability of any particular arrangement,

$$P(U_x) = \binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}^{-1}. \tag{7.2}$$

As non-unique arrangements have two possible configurations, their corresponding probability, N_x is,

$$P(N_x) = 2\binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}^{-1}. \tag{7.3}$$

Assuming that the arrangements are random, the node complexity measure, ϕ, which is the information associated with the split of node l is,

$$\phi(l) = -A_u \binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}^{-1} \log_2 P(U_x) - \left(\binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|} - A_u\right)\binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}^{-1} \log_2 P(N_x), \tag{7.4}$$

which can be simplified to,

$$\phi(l) = \log_2 \binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|} - \left(1 - A_u \binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}^{-1}\right). \tag{7.5}$$

This is a suitable weight for calculating the average information gain because it represents the node complexity and therefore, how useful it is in identifying the predictive power of the feature.

7.3 The Feature Sampling Distribution

The measures of feature importance can be used to select the most relevant features, but another application is to include all of the features in the learning process and assign a weight to each one. Random Forest can be adapted quite easily to achieve this. The standard Random Forest method chooses a feature at each split randomly from the set of all possible features. This feature sampling distribution is typically uniform but can be altered to incorporate the learned feature importance (Rogers and Gunn, 2005; Borisov et al., 2006). By applying this technique, features that are deemed to be more important are chosen with a greater probability. Standard decision trees consider all features at each stage of construction and choose the feature that provides the highest information gain. Altering the feature sampling distribution can be viewed as increasing the similarity of the randomly created trees to a tree that incorporates feature selection in its construction. If a feature selection algorithm is applied to Random Forest, then the class of possible trees that can be built is restricted and the diversity is reduced. By altering the feature sampling distribution, a trade off is introduced between increasing the strength of the base learners and maintaining the diversity of the ensemble. The goal is then to maximise the generalisation performance by optimising the feature sampling distribution in terms of these factors. This method alters the distribution from which the base learners are created, so that the more accurate learners are created more frequently. This can be compared to some ensemble feature selection algorithms which actively choose accurate base learners, and is also similar to methods that alter the combination strategy such that the learners are weighted according to their accuracy.

The alteration of the feature sampling distribution can be achieved in two ways. A two-stage method can be adopted, where an evaluation of the feature importance is conducted first and then applied to the construction of a Random Forest. Another approach is to combine the evaluation stage and the construction stage in a parallel scheme. As each tree in the forest is constructed, it can be used to evaluate the features and update the feature sampling distribution accordingly. The two-stage approach has the advantage of developing a reliable and accurate estimate of the ideal feature sampling distribution from which to build the forest. The parallel approach would be faster but has the problem of instability during the initial stages of the algorithm. When the forest is still small, there is very little information about the features from which to update the sampling distribution. Initial overweighting of some features may create a sampling distribution that is far from ideal and the algorithm may not be able to recover from this as more trees are added. An implementation of the parallel method by Borisov et al.

(2006) uses the measure of gain as the feature importance metric. The weights of the features, v, are updated according to,

$$v(X_f, T) = \lambda Imp(X_f, 0) + \sum_{t=1}^{T} Gain(X_f, t), \tag{7.6}$$

where $Gain(X_f, t)$ is the average reduction in impurity that is caused by feature X_f in the construction of the t^{th} tree. $Imp(X_f, 0)$ is taken as the impurity of the whole data and λ is a parameter which is used to control the rate at which the feature sampling distribution changes. By increasing the value of λ the rate is decreased and the problem of initial overweighting is overcome. However, if λ is too high, the sampling distribution will not change significantly and a forest very close to a standard Random Forest will be produced. Therefore, there is a need for tuning of the λ parameter, which can typically be achieved using cross validation on the training data but the advantage of the small computational requirement is lost.

This method is comparable to the FeatureBoost algorithm (O'Sullivan et al., 2000), described in Chapter 3, which also alters the selection of features to improve ensemble performance. However, although these two techniques have apparent similarities, their underlying processes are very different. The method described here employs Random Forest base learners, which have a high level of diversity, and attempt to improve their accuracy by utilising the more relevant features. Conversely, the FeatureBoost algorithm employs accurate base learners and attempts to promote diversity by forcing the algorithm to consider the less relevant features.

7.3.1 A Stable Parallel Method Using Confidence intervals

A method of avoiding the cross validation stage would certainly be beneficial to the performance of the algorithm but a way of estimating the optimal convergence rate of the sampling distribution is required. The method introduced here, is to calculate a confidence interval for the estimate of expected information gain for each feature. Effectively, by observing the information gain values one is sampling from a distribution, which is assumed here to be normal. What is then required is the ability to approximate the probable distance between the mean of this normal distribution and the observed average information gain. Although the mean and variance of the true distribution are unknown, this can be accomplished by using the pivotal quantity method.

Given the sample mean (observed average information gain), $\overline{IG}$, the sample variance, σ^2, the sample size, m and the true mean of the distribution μ. The pivotal quantity is,

$$\frac{\overline{IG} - \mu}{\sigma / \sqrt{m}}, \tag{7.7}$$

and has a Student's t distribution with $m - 1$ degrees of freedom. A confidence interval can then be constructed within the distribution of the pivotal quantity,

$$P\left[q_1 < \frac{\overline{IG} - \mu}{\sigma/\sqrt{m}} < q_2\right] = \gamma, \tag{7.8}$$

which gives the bound,

$$\left[\overline{IG} - \frac{q_2\sigma}{\sqrt{m}}, \overline{IG} - \frac{q_1\sigma}{\sqrt{m}}\right]. \tag{7.9}$$

The process then consists of taking the observed information gains for each feature, calculating the sample mean and sample variance and deciding what level of confidence to use. A value of 0.95 for γ is typical. As the Student's t distribution is symmetrical, the optimal boundary values will occur when $q_1 = -q_2$. These can be calculated from the value of γ by using an inverse Student's t distribution and then used to give the confidence interval around the sample mean.

If the sample mean is calculated using the weighted method then the sample variance must be weighted accordingly. Also, the sample size m must be re-examined, as a definition is required for a unit observation. A sensible value should be close to the information associated with the split of a node, averaged over all nodes in the tree. However, this is not known before the construction of the forest and must remain constant throughout.

These confidence intervals can then be used to update the feature sampling distribution by choosing values for each feature that lie within each confidence interval that yield the most uniform distribution. Here, the average information gain for each feature is viewed as assuming a value within a range of possible values, which are determined by the corresponding confidence interval. These average information gains can then be normalised and applied directly to set the feature sampling distribution, but their values must first be chosen such that they remain similar to each other and within their respective ranges. One simple method for achieving this, is to find the midpoint between the maximum lower bound and minimum upper bound of all of the confidence intervals. The value for each feature is then chosen to be as close to this value as possible without falling outside of the corresponding confidence interval.

This method will only update the feature sampling distribution when it has a confidence equal to γ. As more trees are added to the forest, the confidence in each estimate increases and the confidence intervals become smaller. Consequently, the feature sampling distribution becomes less uniform and closer to the ideal.

The confidence interval construction requires the calculation of an inverse Student's t distribution and the mean and variance of information gain for each feature. The experiments in this thesis update the confidence intervals after the construction of every tree, in order to utilise the information concerning the features as soon as it is available. However, the computational load can be reduced, if desired, by updating the confidence intervals after a larger number of trees have been constructed. The cost of this is that some of the trees will be constructed using a feature sampling distribution that has not

Algorithm 3: Parallel Update of Feature Sampling Distribution

```
Input       Training Data $X, Y = (x_1, y_1), ..., (x_N, y_N)$
            Test Data $X' = x'_1, ..., x'_M$

Initialise  Feature sampling distribution $v(X_1, ..., X_F) = \left(\frac{1}{F}, ..., \frac{1}{F}\right)$
            Confidence Parameter e.g. $\gamma = 0.95$
            Sets of gain samples $IG(X_i) = \text{empty} : i = 1 \text{ to } F$
            Number of Learners e.g. $T = 100$

Algorithm   For $t = 1$ to $T$
                $\mathcal{X}^{BAG}$ = Use Bagging to sample $N$ examples from $X, Y$
                Train $h_t$ using $\mathcal{X}^{BAG}$ and $v_t$
                For $l$ = all non-terminal nodes in $h_t$
                    $X_f$ = Feature used to partition $l$
                    Add information gain achieved by $X_f$,
                    when partitioning $l$, to the set $IG(X_f)$
                end.
                For $i = 1$ to $F$
                    Calculate lower $(low(X_i))$ and upper $(upp(X_i))$
                    confidence bounds using $\gamma$
                end.
                midPoint=$\frac{1}{2}(\max_i\{low(X_i)\} + \min_i\{upp(X_i)\})$
                For $i = 1$ to $F$
                    If midPoint> $upp(X_i)$
                        $v(X_i) = upp(X_i)$
                    Else If midPoint< $low(X_i)$
                        $v(X_i) = low(X_i)$
                    Else
                        $v(X_i)$ =midPoint
                end.
                Normalise $v$
                Classify $X'$ using $h_t$
            end.
            $H(x'_i) = \text{sgn}\left(\sum_{t=1}^{T} h_t(x'_i)\right) : i = 1 \text{ to } M$

Output      Test Data Predictions $H(x'_1, ..., x'_M)$
```

been created from all of the information that is available.

7.4 A Feature Selection Threshold

It is conjectured that the average information gain during the construction of decision trees is a measure of feature relevance. As previously discussed, it tests the feature on different areas of the input space and consequently accounts for the different relationships between features. If these measures of feature importance are applied to learning algorithms which are based on decision trees, it also contains the bias of the learning algorithm. However, the worst performance of any given feature for the splitting of any given node, is that no reduction in entropy is possible and an information gain of zero is achieved. This means that the average information gain for any feature is the mean of a non-negative sample. The problem that arises from this, is that a feature which is completely irrelevant will produce some reductions in entropy purely by chance. Therefore, a non-zero feature importance value will be produced. This problem is particularly detrimental to performance when there are a relatively large number of irrelevant features and these values are used to update the feature sampling distribution. This is because, the probability of sampling any of the irrelevant features is the sum of all of their individual probabilities and although these may be small, the total can easily become significant if there are many. To overcome this problem, a feature selection threshold is introduced here, which approximates the expected information gain that is achieved by an irrelevant feature, given the size of the node being split.

Assuming that the task is binary classification and the data is projected onto a single feature, a node of size $|\mathcal{X}_l|$, containing $|\mathcal{X}_{l+}|$ positive examples has $\binom{|\mathcal{X}_l|}{|\mathcal{X}_{l+}|}$ possible arrangements. If the feature is irrelevant then these arrangements occur with equal probability. By constructing all of the possible arrangements and finding the maximum information gain for each one, the expected value is calculated for various node constitutions and the outcome is shown in Figure 7.3.

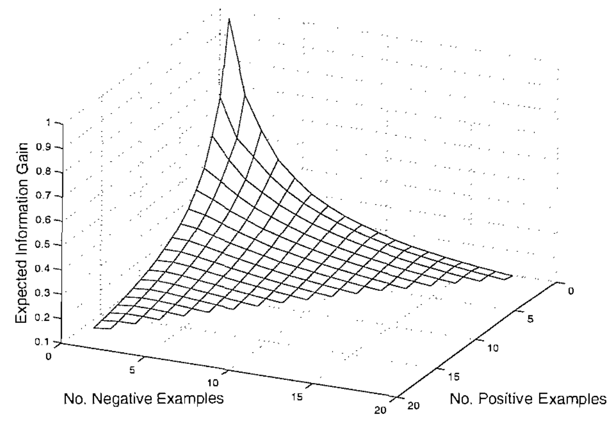

FIGURE 7.3: Expected information gain for nodes containing various numbers of positive and negative examples

Due to the huge computational cost of evaluating the expected information gain in this manner, it is not a feasible method for feature selection, however, it can be approximated. For a fixed node size, the maximum expected information gain appears to be when there are equal numbers of positive and negative examples and the minimum occurs when the ratio is most unbalanced. Therefore, the minimum expected information gain for a node of fixed size $|\mathcal{X}_l|$, occurs when it contains only one example of one class, $|\mathcal{X}_{l+}| = 1$ or

$|\mathcal{X}_{l-}| = 1$. This can be calculated in the following manner.

The information gain is the difference between the parent entropy and the combined child entropy and as the parent entropy for any given composition is fixed, only the combined child entropy needs to be considered. The case used here is that there is only one positive example, $|\mathcal{X}_{l+}| = 1$, and the optimal split leaves this example in the left node of size $|\mathcal{X}_{l1}|$. The right node then contains only negative examples and will have an entropy of zero. The combined child entropy is then,

$$\begin{aligned} Ent\,(l1, l2) &= -\frac{|\mathcal{X}_{l1}|}{|\mathcal{X}_l|}\left(\frac{1}{|\mathcal{X}_{l1}|}\log_2\frac{1}{|\mathcal{X}_{l1}|} + \frac{|\mathcal{X}_{l1}|-1}{|\mathcal{X}_{l1}|}\log_2\frac{|\mathcal{X}_{l1}|-1}{|\mathcal{X}_{l1}|}\right) \\ &= -\frac{1}{|\mathcal{X}_l|}\left((|\mathcal{X}_{l1}|-1)\log_2(|\mathcal{X}_{l1}|-1) - |\mathcal{X}_{l1}|\log_2|\mathcal{X}_{l1}|\right) \qquad (7.10) \end{aligned}$$

Differentiating by $|\mathcal{X}_{l1}|$ then gives,

$$\frac{\partial}{\partial|\mathcal{X}_{l1}|}\left[Ent\,(l1, l2)\right] = \frac{1}{|\mathcal{X}_l|}\left(\log_2|\mathcal{X}_{l1}| - \log_2(|\mathcal{X}_{l1}|-1)\right) \qquad (7.11)$$

For the case when $|\mathcal{X}_{l1}|$ is not equal to 1 and consequently, must be a positive value of a least 2, the entropy is always increasing with $|\mathcal{X}_{l1}|$. Therefore, the optimal split is obtained when $|\mathcal{X}_{l1}|$ is minimal. For a parent node of size $|\mathcal{X}_l|$, the single positive example can assume only one of the possible $|\mathcal{X}_l|$ positions. If $|\mathcal{X}_l|$ is taken to be even, then by symmetry only $\frac{|\mathcal{X}_l|}{2}$ of the arrangements need to be considered. The expected child entropy can then be written,

$$\begin{aligned} E\left[Ent\,(l1, l2)\right] &= -\frac{2}{|\mathcal{X}_l|}\sum_{|\mathcal{X}_{l1}|=1}^{\frac{|\mathcal{X}_l|}{2}}\frac{|\mathcal{X}_{l1}|}{|\mathcal{X}_l|}\left(\frac{1}{|\mathcal{X}_{l1}|}\log_2\frac{1}{|\mathcal{X}_{l1}|} + \frac{|\mathcal{X}_{l1}|-1}{|\mathcal{X}_{l1}|}\log_2\frac{|\mathcal{X}_{l1}|-1}{|\mathcal{X}_{l1}|}\right) \\ &= -\frac{2}{|\mathcal{X}_l|^2}\log_2\left(\prod_{|\mathcal{X}_{l1}|=1}^{|\mathcal{X}_l|/2}\frac{(|\mathcal{X}_{l1}|-1)^{|\mathcal{X}_{l1}|-1}}{|\mathcal{X}_{l1}|^{|\mathcal{X}_{l1}|}}\right) \\ &= \frac{1}{|\mathcal{X}_l|}\log_2|\mathcal{X}_l| - \frac{1}{|\mathcal{X}_l|} \qquad (7.12) \end{aligned}$$

Re-introducing the parent entropy, the expected information gain for a node of size $|\mathcal{X}_l|$ with only one example of one class is a lower bound on the general expected information gain of an irrelevant feature $E\left[IG^{IF}\right]$, and can be written,

$$E\left[IG^{IF}\right] \geq \frac{1}{|\mathcal{X}_l|} - \frac{|\mathcal{X}_l|-1}{|\mathcal{X}_l|}\log_2\frac{|\mathcal{X}_l|-1}{|\mathcal{X}_l|}. \qquad (7.13)$$

The expected information gain for the case when there are equal numbers of each class cannot be calculated as easily. By examining the data that was generated for the construction of Figure 7.3 and plotting the expected information gain for the case when the classes are equal on a logarithmic scale, it is seen that this quantity gives an upper bound on the general expectation and can be approximated by the following expression,

$$E\left[IG^{IF}\right] \leq \left(\frac{|\mathcal{X}_l|}{2}\right)^{-0.82}. \tag{7.14}$$

These two quantities can be viewed as upper and lower bounds on the expected information gain that is achieved by splitting a node of size $|\mathcal{X}_l|$ and are shown in Figure 7.4. The mid-point between these two bounds represents an estimate of the expected information gain of an irrelevant feature and can be applied as a feature selection threshold. The assumption that is made here is that no data points lie on top of one another and all of the possible split positions are realisable. However, the Random Forest algorithm uses Bagging, which samples the data with replacement to form different sets from which to construct the base learners. As a result of this some data points are selected multiple times and will consequently lie on top of one another. This limits the possible arrangements and results in a higher observed information gain. To account for this, Bagging is performed as usual to form a sample of the training data and then the multiple instances are removed.

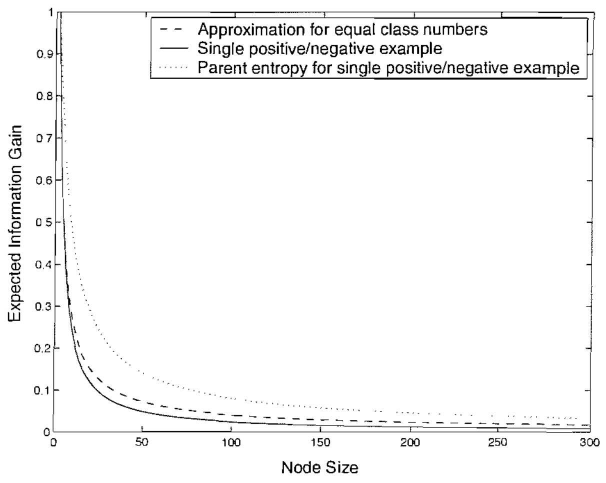

FIGURE 7.4: Bounds on the expected information gain for varying node size. The parent entropy of a node containing a single example of one class is also plotted, as this represents the maximum achievable information gain for this case.

7.4.1 Hypothesis Testing

The approximated value $E\left[IG^{IF}\right]$, is assumed to be the worst case performance of any feature. If a feature is irrelevant then the value of the mean $\overline{IG}$, should be equal to $E\left[IG^{IF}\right]$. However, as the technique uses a sample to estimate this mean, there is still a good chance that it will be greater than this threshold. Consequently, irrelevant features may be chosen. Hypothesis testing can be employed here to discover the degree of confidence there is in a feature being relevant. Each feature has a set of samples of information gain, corresponding to when it was used to split data. On each of these occasions the expected information gain of an irrelevant feature is also approximated. If the feature is irrelevant, then these values should approximate one another. The variable that is used here is the difference between these values and it is assumed to have a normal distribution. The thresholding method is then equivalent to rejecting the feature if the observed mean, $\overline{IG - E\left[IG^{IF}\right]}$, is less than or equal to zero.

The null hypothesis is then set up to represent an irrelevant feature by assuming that the true mean, μ, is less than or equal to zero.

$$\mathcal{H}_0 : \mu \leq 0 \tag{7.15}$$

The alternate hypothesis must be the complement of this.

$$\mathcal{H}_a : \mu > 0 \tag{7.16}$$

The null hypothesis can then be rejected if the corresponding likelihood is less than some confidence value, γ. If $\overline{IG - E\left[IG^{IF}\right]}$ is less than 0 then the null hypothesis cannot be rejected. However, if it is greater than zero then a one tailed t test can be performed, where the null hypothesis can be rejected with confidence $1 - \gamma$ if,

$$\frac{\overline{IG - E\left[IG^{IF}\right]} - \mu}{\frac{\sigma}{\sqrt{m}}} > q. \tag{7.17}$$

Where m is the sample size and σ is the standard deviation of the sample. The left hand side of this inequality is a variable which has a Student's t distribution with $m - 1$ degrees of freedom. The value of q represents a threshold, where the likelihood function at this point is equal to γ. This simplifies to,

$$\frac{\overline{IG - E\left[IG^{IF}\right]}\sqrt{m}}{\sigma} > q. \tag{7.18}$$

It is important to note that while this method will identify features that are relevant with some degree of confidence, the relevance of the remaining features will be unknown. Therefore, as a feature selection technique, this method may discard some relevant features.

The following pseudo-code describes how this algorithm identifies a relevant subset of features,

The reliability of the estimated relevance of a feature is dependent upon the number of samples of information gain, m, that are gathered. This number is dependent upon the dimensionality of the data, the number of ensemble members and the size of the trees. The trees supply the samples of information gain, which are randomly allocated to each of the F features. If $\mathcal{L}_t$ denotes the set of non-terminal nodes in hypothesis h_t, then the average number of samples of information gain for each feature, that are gathered in a

Algorithm 4: Feature Selection Through Random Forest

```
Input       Training Data $X, Y = (x_1, y_1), ..., (x_N, y_N)$

Initialise  Confidence Parameter e.g. $\gamma = 0.05$
            Subset of relevant features $S$ = empty
            Sets of gain samples $IG(X_i)$ = empty : $i = 1$ to $F$
            Number of Learners e.g. $T = 100$

Algorithm   For $t = 1$ to $T$
                $\mathcal{X}^{BAG}$ = Use Bagging to sample $N$ examples from $X, Y$
                Train $h_t$ using $\mathcal{X}^{BAG}$
                For $l$ = all non-terminal nodes in $h_t$
                    $X_f$ = Feature used to partition $l$
                    $gain(l)$ = Information Gain achieved by partition of $l$
                    Add $\left(gain(l) - E\left[IG^{IF}\right]\right)$ to set $IG(X_f)$
                end.
            end.
            For $i = 1$ to $F$
                Compute likelihood of null hypothesis $L^0$ from set $IG(X_i)$
                If $L^0 < \gamma$
                    Add $X_i$ to set $S$
            end.

Output      Subset of relevant features $S$
```

Random Forest of size T, can be written,

$$m = \frac{1}{F} \sum_{t=1}^{T} |\mathcal{L}_t| . \tag{7.19}$$

Therefore, the number of required trees may vary between data sets. Data which yields smaller trees or has higher dimensionality may require a larger number of trees to be constructed.

7.5 Local Feature Relevance

In Chapter 6 it was discussed how some learning problems contain features that are only relevant within a local area of the space. An example of this was the Multiplexor, as in Example 6.2. This kind of feature interaction leads to some features only being useful when they are used in conjunction with another specific subset of features. Some algorithms have been developed to identify and exploit this local relevance, by examining the discriminative capabilities of features within different localities (Hong, 1997;

Domingos, 1997; Howe and Cardie, 1997; Apte et al., 1997).

Ensemble algorithms that promote diversity through manipulation of the feature space, have the potential to uncover feature interaction by exploring different feature subsets. Puuronen and Tsymbal (2001) take advantage of this random subset exploration by employing local weights for each hypothesis. The intuition behind their method is that the combination of random feature selection and the exploitation of local hypothesis performance, implicitly performs local feature selection. This dynamic integration can also be applied to Random Forest (Tsymbal et al., 2006), and there is an extra advantage to employing Random Forest with this technique. As Random Forest utilises the Bagging algorithm to sample the training data, each learner has an independent set of data in the form of the out-of-bag examples. This set of data can be used to assess the performance of the learner, without the need for cross-validation.

Another issue that is considered by Tsymbal et al. (2006), when applying dynamic integration to Random Forest, is that of the distance measure. Previously, their technique defined the locality of an example through the Euclidian distance metric. When analysing data with decision tree induction, this measure may be inappropriate. Figure 7.5 shows an example decision tree and the corresponding split positions in the input space. The two examples are in close proximity in terms of the Euclidian distance, but are partitioned into very different sections of the tree. In this case, the local performance of the tree on one example is not representative of the ability of the tree on the other.

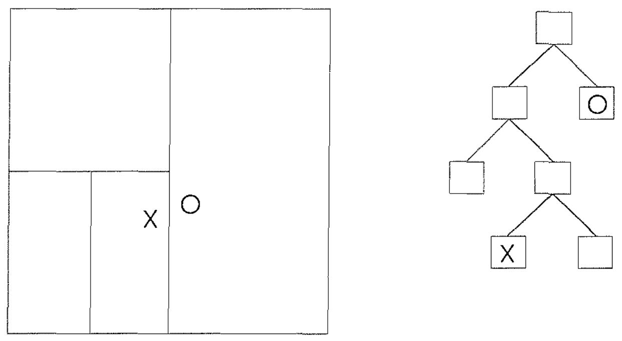

FIGURE 7.5: Example showing possible split locations of a decision tree model and the corresponding tree. The two examples appear close in the Euclidian sense, but are classified by different sections of the tree.

Their method measures the similarity of two examples as the proportion of decision trees in a Random Forest for which the examples lie in the same leaf node.

$$RFSIM\left(x_1, x_2\right) = \frac{\sum_{t=1}^{T} I\left(h_t^l\left(x_1\right) = h_t^l\left(x_2\right)\right)}{T}, \tag{7.20}$$

where $h_t^l(x)$ is the leaf node position of example x in hypothesis h_t. This similarity measure is calculated with respect to the classification model and is, therefore, a more reliable quantity.

Each prediction is weighted according to the performance of the hypothesis on the out-of-bag set and the similarity of these examples to the test example.

$$w_t(x) = \frac{\sum_{i:x_i \in \mathcal{X}_t^{OOB}} RFSIM^3(x, x_i)\, y_i h_t(x_i)}{\sum_{i:x_i \in \mathcal{X}_t^{OOB}} RFSIM^3(x, x_i)} \tag{7.21}$$

where $\mathcal{X}_t^{OOB}$ represents the set of out-of-bag data for hypothesis h_t. They empirically found that cubing the similarity measure gave good performance. The output of the ensemble for test case x is then given by,

$$H(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} w_t(x)\, h_t(x)\right). \tag{7.22}$$

7.5.1 A measure of local relevance

When constructing a Random Forest, statistics can be gathered on the performance of each feature. As previously discussed, Random Forest has the ability to discover feature interaction through its nature of testing features within the context of other features. The statistics that have been presented earlier in this chapter have the capacity to account for feature interaction, but are still formulated as global measures for the relative importance of each feature. However, these measures can be adapted to reflect their local ability. The partitioning of each node provides a measure of information gain for the selected feature, which represents the degree of correlation to the target. This correlation is relevant to the specific area that is being partitioned, and this area can be defined by the set of training data that is within the node. The following describes a method to record the average gain that is achieved by each feature, for each example (Rogers and Gunn, 2007). The feature relevance statistics can then be stored in a matrix that is of equal size to the training data.

An observed value of information gain represents the reduction in entropy of the class labels, which is the average information required to describe the data.

$$Ent(Y, \mathcal{X}_l) = -\sum_{y \in \mathcal{Y}} \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|} \log_2 \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y\}|}{|\mathcal{X}_l|} \tag{7.23}$$

If the measures of feature relevance are to become example-specific, then the quantity should reflect the reduction in information that is required to describe the example in question. The information is dependent on the class of the example and can be written,

$$Info(x_j, l) = -\log_2 \frac{|\{i : x_i \in \mathcal{X}_l, y_i = y_j\}|}{|\mathcal{X}_l|}, \tag{7.24}$$

and the example-based feature relevance statistic, $FR(X_f, x_i)$, is the average observed reduction in this information for example x_i when partitioning on feature X_f.

7.5.2 Local Feature Sampling

The statistic introduced in Section 7.5.1 builds up a picture of how useful each feature is across the training data. If applied correctly, this example-based feature information can be used to improve classifier performance. In Section 7.3 the relative measures of feature importance are used to alter the feature sampling distribution of a Random Forest, but the distribution is fixed and remains constant for all partitions. Here, the motivation is to adapt the Random Forest algorithm so that it utilises the local knowledge, whilst maintaining the diversity of the ensemble. The method described by Rogers and Gunn (2007) constructs a separate feature sampling distribution for each node, accounting for the data that is being partitioned. This can be achieved by calculating the average feature relevance over the data concerned. The relative weight v, for each feature X_f, in node l, can be written,

$$v\left(X_f, l\right) = \frac{\sum_{i:x_i \in \mathcal{X}_l} FR\left(X_f, x_i\right)}{|\mathcal{X}_l|}, \tag{7.25}$$

which is normalised to form a distribution.

It is important to note that the measure of gain can assume a negative value and, therefore, so can the corresponding sampling probability. In this case, the sampling probability is set to zero.

The pseudo-code on the following page describes how each tree is constructed with localised sampling,

7.6 Conclusions

This chapter has discussed the utility of employing Random Forest as a mechanism for identifying feature relevance. The values of information gain that are observed during Random Forest construction are proposed as a measure of feature importance. The reliability of this measure was examined and an improvement was suggested through weighting the observations with a new measure of node complexity.

Several methods for employing these measures of feature relevance to improve Random Forest performance have also been proposed. A feature weighting technique, that updates the feature sampling distribution during Random Forest construction, has been introduced that implements a statistical technique to control the rate of update. A feature selection threshold has been derived through an analysis of the expected performance of an irrelevant feature and the notion of combining this with a hypothesis testing technique has also been proposed.

It has been discussed that, as Random Forest tests features in different areas of the space,

Algorithm 5: Localised Feature Sampling

```
Input        Bagged training data $\mathcal{X}^{BAG}, \mathcal{Y}^{BAG}$
             Feature relevance statistics $FR \begin{bmatrix} (X_1, x_1) & .. & (X_F, x_1) \\ .. & .. & .. \\ (X_1, x_N) & .. & (X_F, x_N) \end{bmatrix}$

Initialise   Tree node $l = 1$
             Tree $h_t$ = Unexpanded root node $l$
             $\mathcal{X}_l = \mathcal{X}^{BAG}$

Algorithm    While(Tree Complete = FALSE)
                 For $j = 1$ to $F$
                     $v(X_j) = \sum_{i:x_i \in \mathcal{X}_l} FR(X_j, x_i)$
                     If($v(X_j) < 0$)
                         $v(X_j) = 0$
                 end.
                 Normalise $v$
                 $X_f$ = Choose random feature using distribution $v$
                 Partition node $l$ using feature $X_f$ to maximise information gain
                 Add child nodes $l1$ and $l2$ to $h_t$, along with corresponding
                 subsets $\mathcal{X}_{l1}$ and $\mathcal{X}_{l2}$
                 $l = l + 1$
             end.

Output       $h_t$
```

it incorporates the ability to identify local feature information. An example-specific measure of feature relevance has been presented to exploit this idea, which measures the individual gain of examples. A method has also been described for altering the feature sampling distribution to accomodate these measures that takes advantage of local information and improves ensemble accuracy.

Chapter 8

Empirical Evaluation of Random Forest Methods

8.1 Datasets

The propeties of the data sets used in these experiments are shown in Table 8.1, including the number of relevant features where known. The Wisconsin Breast Cancer (WBC), Pima Diabetes, Sonar, Ionosphere and Votes are available from the UCI Repository (Blake and Merz, 1998).

TABLE 8.1: Data Set Properties

Data Set	No. Examples	No. Features	No. Relevant Features
WBC	683	9	?
Pima	768	8	?
Sonar	208	60	?
Ionosphere	351	34	?
Votes	435	16	?
Friedman	200	10	5
Simple	300	9	2
Madelon(Train)	2000	500	?
Madelon(Valid)	600	500	?

Simple is an artificial dataset consisting of 9 features and 300 examples. The output is generated according to the function,

$$Y = X_1^2 + 2X_2. \tag{8.1}$$

The remaining seven features are redundant and consequently this data set should benefit significantly from feature selection algorithms. It is important to note that as the input values to the function are drawn from a uniform distribution on $[0, 1]^9$, feature 2 has a larger influence on the target.

The Madelon dataset is synthetic and was used in the NIPS 2003 Feature Selection Challenge (Guyon et al., 2006).

8.2 Irrelevant Features and Random Forest

To demonstrate the effect of irrelevant features on Random Forest, the five real data sets are tested with various numbers of additional random features, generated from a uniform distribution. The number of irrelevant features is varied between 0 and 30. For each experiment, the data sets are randomly partitioned into 90% for training and 10% for testing. 100 trees are consructed to form the forest and classify the test data. This is repeated over 100 trials and the results for the extreme cases of 0 and 30 irrelevant features are shown in Table 8.2. The average tree sizes represent the number of nodes within each tree, averaged over the entire forest.

TABLE 8.2: Error rates and average tree sizes in Random forest for 0 and 30 irrelevant features. Values in brackets for the error rates are the corresponding variances.

Data Set	Error(0)	Av. Tree(0)	Error(30)	Av. Tree(30)
WBC	0.0293(0.0004)	55.7	0.0370(0.0004)	148.7
Pima	0.2525(0.0022)	265.3	0.2999(0.0025)	334.1
Sonar	0.1671(0.0066)	81.9	0.2190(0.0071)	89.4
Ionosphere	0.0756(0.0016)	61.0	0.0975(0.0021)	97.2
Votes	0.0586(0.0012)	51.1	0.0982(0.0021)	123.7

Figure 8.1 shows how the error rates increase steadily, as more irrelevant features are added to the forest and Figure 8.2 demonstrates how the average tree size also increases with the presence of irrelevant features. It is important to note that in some cases the error rate does not increase as rapidly, because there is sufficient data to allow the trees to grow larger and compensate. This increase in tree size is undesirable as it increases the computational load.

8.3 The Node Complexity Measure

It is proposed that our node complexity measure can improve the estimate of average information gain. In order to examine this, the effect of our measure on the information gain distributions is evaluated. The Simple dataset is used to generate 5000 trees and the information gain values for all of the features are recorded. The values are discretised into intervals of size 0.01 so that each feature has a set of bins. As each node is split, the algorithm increments the bin corresponding to the feature being used and the information gain value obtained. As a comparison, one method increments the

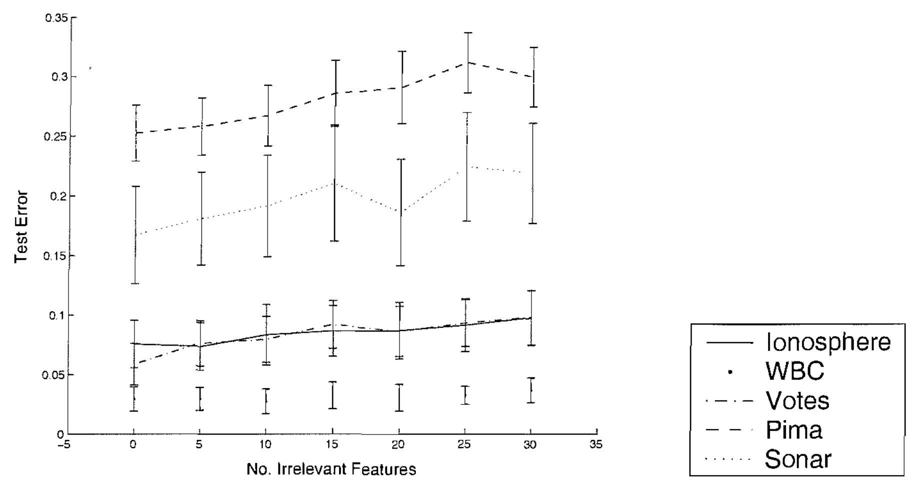

FIGURE 8.1: Error rates of Random Forest on five real data sets with varying numbers of additional irrelevant features. Error bars have a width of one standard deviation recorded over 100 trials.

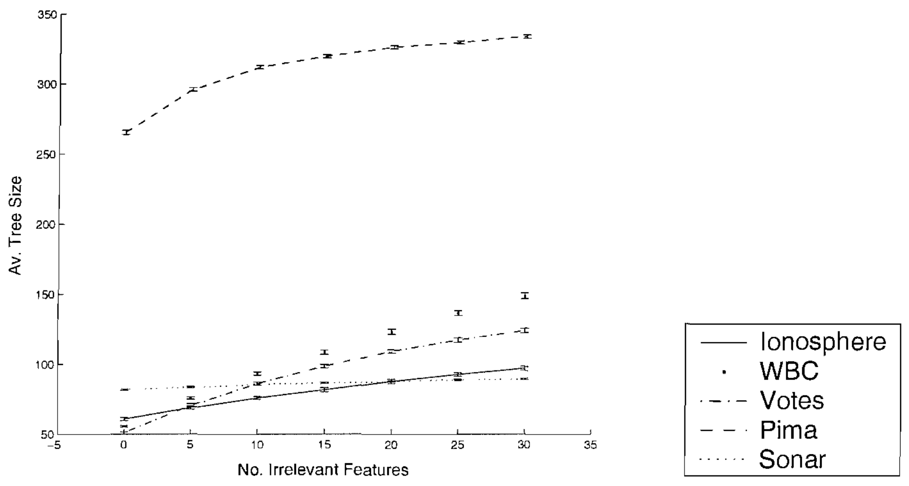

FIGURE 8.2: Average tree sizes created by Random Forest on five real data sets with varying numbers of additional irrelevant features. Error bars have a width of one standard deviation recorded over 100 trials.

bins by a single unit, the other method increments by the measure of node complexity, $\phi(l)$. Incrementing by this value shows the effect of weighting the information gains in this manner. The results for three of the features are shown in Figure 8.3. The middle example is feature two, which carries the most information about the target. The left example is the next most important feature and the example on the right is a redundant feature.

It is particularly interesting to note the spikes that occur when unit weighting is used. At first glance, they appear to simply be noise but closer inspection reveals that they occur in the same places for all three features. The extreme right hand spike is the information gain that is achieved by the perfect split of a node made up from half of each class. This can occur when a node containing two examples, one from each class,

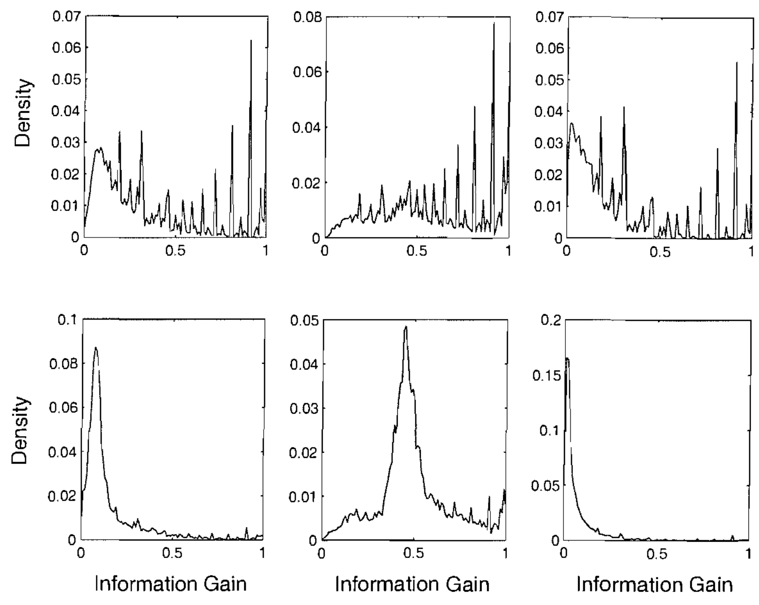

FIGURE 8.3: Observed density functions of information gain for three features from the Simple dataset. Observed density using unit weighting (top) and observed density using node complexity weighting (bottom).

is split. The spike immediately to the left of the maximum one can occur when a node containing two examples of one class and one of the other is split perfectly. These smaller nodes are much easier to split and can be split perfectly by features which carry very little information about the target. This is illustrated by the eradication of the spikes in the lower plots, where the samples of information gain are weighted according to the node complexity. This result clearly demonstrates the ability of the node complexity measure to improve the estimate of feature importance by analysing the reliability of each sample.

8.4 Feature Selection by Decision Tree Induction

The feature weighting/selection techniques introduced here, involve the average information gain achieved during construction of a Random Forest as a measure of feature importance. These methods are compared to a correlation-based technique Hall (2000), and the MDL motivated decision tree method described in Section 2.4.2. Firstly, the decision tree method is tested in two experiments. The first experiment builds a decision tree, optimising the information gain at each split in terms of split position and feature used. The tree is pruned using the MDL constraint and the features that were used in the tree are the ones selected. These features are then applied to constructing a Random Forest with a uniform feature sampling distribution and this forest is used to classify the test data. The second experiment is conducted in exactly the same way, except that instead of only selecting features which were used in the tree, all features that were deemed feasible at any point are selected. The term feasible refers to features that could split a node without violating the MDL pruning condition, even if it was not the feature that maximised the information gain. The first approach has the problem that the number of features that can be selected is limited by the size of the tree. The second approach should not have this problem, but may select some redundant features that do not provide any additional target information.

The data sets are partitioned into 90% for training and 10% for testing. The experiment is repeated over 100 trials and the results are averaged. Table 8.3 compares Random forest without feature selection to the decision tree methods.

TABLE 8.3: Classification error rates of Random Forest without feature selection (RF), and with decision tree feature selection, selecting only features that were used (DT1), and all feasible (DT2). The values in brackets are the corresponding variances of test error over the 100 trials.

Data Set	RF	DT1	DT2
WBC	0.0293(0.0004)	0.0310(0.0005)	0.0251(0.0003)
Sonar	0.1671(0.0066)	0.2567(0.0077)	0.2043(0.0056)
Votes	0.0586(0.0012)	0.0418(0.0008)	0.0625(0.0015)
Pima	0.2525(0.0022)	0.2674(0.0024)	0.2381(0.0018)
Ionosphere	0.0756(0.0016)	0.0892(0.0019)	0.0614(0.0017)
Friedman	0.1670(0.0070)	0.1975(0.0077)	0.1910(0.0068)
Simple	0.0870(0.0028)	0.0147(0.0005)	0.0147(0.0005)

It can be seen that the method of only selecting the features that were used (DT1), performs very badly for certain data sets. This is due to the number of features that can be selected being limited by the size of the tree. However, this method does perform significantly better for the votes data set, as this data contains a large amount of redundancy and the second method is unable to remove redundancy as it selects all features that appear useful. The second method performs well for most data sets but can suffer like all decision tree based methods, as the features chosen at any node may not be optimal in regards to the subsequent splits.

8.5 Updating The Feature Sampling Distribution

Examining the effect of using the average information gain as a feature weighting method is explored here. The following experiments use the relative average information gain of each feature to alter the feature sampling distribution, in an attempt to improve the accuracy. The parallel method is employed to form confidence intervals on the estimates of feature importance and the feature sampling distribution is updated after every tree. The assumption that the information gain values are normally distributed is an approximation as the values are bounded on [0,1]. However, the shape of the distribution approximates normal if the node complexity weighting is used.

The initial feature sampling distribution is uniform and the algorithm keeps the most uniform distribution that is within the confidence interval of every feature. As the measures of information gain are weighted, a value for a unit of weight is required. This value should represent the information of the average split in a tree built on the dataset concerned. This value is approximated using a fraction of the node complexity of the entire data.

100 trials are conducted, using 90% of the data for training and 10% for testing. On each of the trials the rate of decrease of average confidence interval size is recorded and the result averaged over all of the trials. This represents the rate of convergence towards the final feature sampling distribution and the results are shown in Figure 8.4.

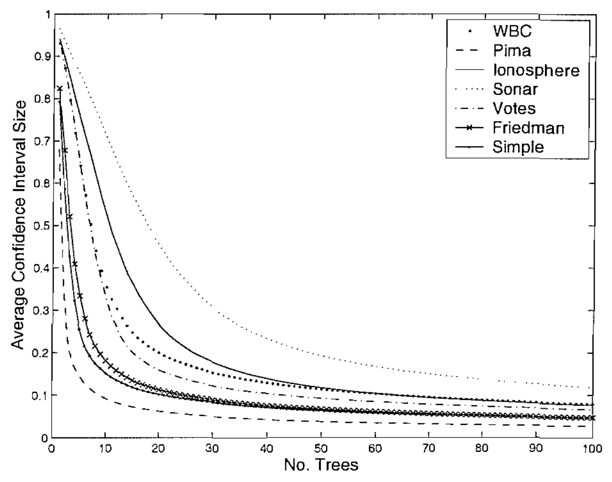

FIGURE 8.4: Convergence rates for the feature sampling distribution. This shows how the average confidence interval size becomes smaller as more trees are added to the forest.

The convergence rates vary with the size of the trees constructed and with the dimensionality of the data. The Pima data set has few features and produces large trees so the features are picked more often within each tree. In contrast, the Sonar data set has 60 features and produces smaller trees.

A two-stage method is also applied where a single decision tree is built on the training data before construction of the forest to produce estimates of the feature importance. The average information gain is calculated using the measure of node complexity as before. However, a standard decision tree performs a search through all of the features for each split. Therefore, an information gain value for every feature is supplied at each node, regardless of whether or not the feature was used. The forest is then constructed using the resultant fixed feature sampling distribution.

These methods are also compared to the CFS algorithm of Hall (2000), which selects a subset of features that have high correlation with the class and low correlation with each other. The selected features are then used to construct a Random Forest using a uniform feature sampling distribution.

Table 8.4 shows the error rates for Random Forest with four feature selection/weighting methods: The best of the two decision tree methods, which selects all of the feasible features (DT), the CFS algorithm (CFS), weighted sampling using confidence interval method (CI WS RF) and the two-stage method using a single tree for evaluation (TREE WS).

TABLE 8.4: Test errors showing the improvement that three feature relevance identification techniques give to Random Forest construction. The values in brackets are the corresponding variances of test error over the 100 trials.

Data Set	DT	CFS	CI WS RF	TREE WS
WBC	0.0251(0.0003)	0.0245(0.0004)	0.0241(0.0003)	0.0288(0.0004)
Sonar	0.2043(0.0056)	0.2329(0.0070)	0.1624(0.0066)	0.2276(0.0068)
Votes	0.0625(0.0015)	0.0452(0.0010)	0.0491(0.0011)	0.0455(0.0008)
Pima	0.2381(0.0018)	0.2560(0.0020)	0.2461(0.0021)	0.2634(0.0019)
Ionosphere	0.0614(0.0017)	0.0625(0.0015)	0.0614(0.0015)	0.0592(0.0015)
Friedman	0.1910(0.0068)	0.1785(0.0056)	0.1630(0.0062)	0.1720(0.0081)
Simple	0.0147(0.0005)	0.1693(0.0037)	0.0387(0.0010)	0.0217(0.0007)

Both methods of updating the feature sampling distribution improve the accuracy for some data sets. This improvement is most noticeable for the Simple data set, which contains a number of irrelevant features. The confidence interval method does not significantly reduce the accuracy for any data set tested here, suggesting that the problem of initial over weighting of the features has been avoided. The two-stage tree method is shown to work well here, although it does significantly reduce the accuracy on the Sonar data set. Both methods compare favourably to the CFS algorithm, as CFS eliminates relevant features for some of the data sets, and consequently degrades the accuracy significantly.

8.6 Feature Selection Thresholding

To view the suitability of the measures of expected information gain as feature selection thresholds, 100 trees are constructed on the Simple dataset and the average information gain for each feature was recorded. The measures of expected information gain for irrelevant features are also calculated. Figure 8.5 shows that the seven irrelevant features are within the bounds that an irrelevant feature is expected to be in and the two relevant features are shown to be more important.

The expected information gain of an irrelevant feature is approximated at each node of forest construction by the mid-point of the two bounds, given by Equations 7.13 and 7.14. This represents the ability of a feature that contains no useful information concerning the target and can be used as a feature selection threshold. Hypothesis testing can also be used as an extension to this by including only the features that were considered relevant with a degree of confidence. In each experiment, 100 trees are constructed to obtain the average information gain for each feature, using the node complexity measure.

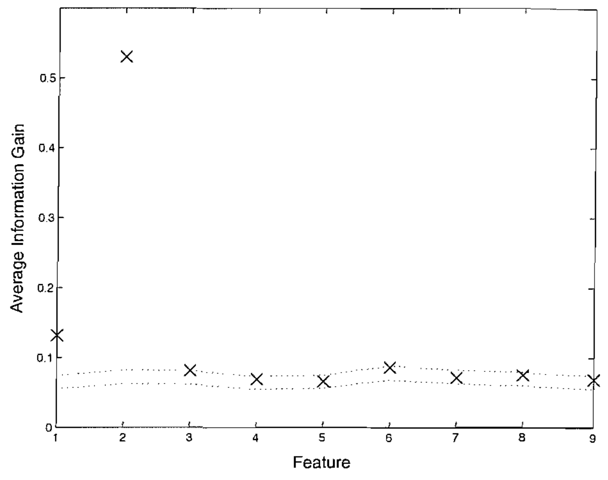

FIGURE 8.5: The measures of feature importance for each feature in the Simple dataset and the approximate bounds for the expected values of irrelevant features. Features 1 and 2 are relevant, the remaining features are irrelevant.

Along with these estimates of feature importance, the corresponding approximations for the performance of an irrelevant feature are also calculated. This information is then used to select a subset of the features and a further 100 trees are constructed based on this subset. Again, the data is partitioned into 90% training and 10% testing and the experiment is repeated over 100 trials. These techniques are compared against RF without feature selection and the CFS algorithm of Hall (2000), which is a correlation-based method. The observed error rates are shown in Table 8.5 and the average number of features selected are shown in Table 8.6.

TABLE 8.5: Error rates for RF when using different feature selection strategies. Without feature selection (standard RF), the CFS algorithm (CFS) and using the expected information gain of an irrelevant feature for thresholding (RF Thr) and with hypothesis testing (RF HT).

Data Set	DT	CFS	RF Thr	RF HT
WBC	0.0251(0.0003)	0.0245(0.0004)	0.0228(0.0003)	0.0228(0.0003)
Pima	0.2381(0.0018)	0.2560(0.0020)	0.2530(0.0024)	0.2530(0.0024)
Sonar	0.2043(0.0056)	0.2329(0.0070)	0.1762(0.0092)	0.1910(0.0089)
Ionosphere	0.0614(0.0017)	0.0625(0.0015)	0.0708(0.0015)	0.0744(0.0015)
Votes	0.0625(0.0015)	0.0452(0.0010)	0.0602(0.0013)	0.0575(0.0011)
Friedman	0.1910(0.0068)	0.1785(0.0056)	0.1610(0.0064)	0.1515(0.0069)
Simple	0.0147(0.0005)	0.1693(0.0037)	0.0750(0.0026)	0.0300(0.0010)

The CFS algorithm performs significantly better on the Votes data set, as this contains a large proportion of redundant features. However, the CFS algorithm significantly degrades in performance on a number of data sets as it can eliminate relevant features. It can be seen that using the expected information gain of an irrelevant feature performs well on most data sets and even those where the accuracy is degraded slightly, the dimensionality is greatly reduced. Hypothesis testing is also shown to be beneficial as it enables further dimensionality reduction.

TABLE 8.6: Number of features used with different feature selection strategies. Without feature selection (standard RF), the CFS algorithm (CFS) and using the expected information gain of an irrelevant feature for thresholding (RF Thr) and with hypothesis testing (RF HT).

Data Set	DT	CFS	RF Thr	RF HT
WBC	9.00(100.0%)	8.62(95.8%)	9.00(100.0%)	9.00(100.0%)
Pima	5.36(67.0%)	3.12(39.0%)	4.00(50.0%)	4.00(50.0%)
Sonar	19.66(32.8%)	14.61(24.4%)	58.39(97.3%)	40.47(67.5%)
Ionosphere	31.29(92.0%)	14.33(42.2%)	32.83(96.6%)	32.60(95.9%)
Votes	14.00(87.5%)	1.00(0.0625)	12.95(80.9%)	12.43(77.7%)
Friedman	3.22(32.2%)	3.06(30.6%)	7.99(79.9%)	5.63(56.3%)
Simple	2.00(22.2%)	1.00(11.1%)	7.06(78.4%)	3.91(43.4%)

The accuracy of the RF HT method is dependent upon the number of trees that were constructed to form the estimates of the average information gain for each feature. Two parameters for this method are the number of trees constructed for this purpose and the level of confidence used. Here the Madelon data set (Guyon et al., 2006), is employed as it is already separated into a training and validation set. 10,000 trees are constructed on the training set to form estimates for the average information gain. The confidence level can then be varied to alter the features that are selected. These features are then used to represent the data and a further 1000 trees are constructed on the training set in order to test the validation set. The results are shown in Table 8.7.

TABLE 8.7: Effect of applying feature selection technique to Madelon data set and varying confidence level. The validation set error and the Out-of-Bag estimate of test error (OOB Est.) is shown.

Confidence	Error	OOB Est.	#Features	Av. Tree Size
0.05	0.3800	0.3825	257	1236
0.025	0.3567	0.3660	232	1232
0.01	0.3783	0.3575	196	1226
0.001	0.3583	0.3255	130	1209
0.0001	0.3167	0.3050	82	1180
10^{-6}	0.2333	0.2445	52	1130
10^{-9}	0.1800	0.1770	30	1025

As the confidence level is lowered, it becomes more difficult to reject the null hypothesis and deem features to be relevant. The Out-of-Bag estimate of test error is gained through testing each training point on the subset of the forest which was not included in the bagged set. It does not use the test data and can therefore, be used to optimise the confidence level.

8.7 Error Convergence

The effect of feature selection on the error convergence rate can be tested by calculating the error as each tree is added to the forest. This is demonstrated here using the Madelon data set. The experiment is performed twice, once without feature selection and once using 100 trees to obtain estimates of the avergage information gain and using these with RF HT with a confidence level of 0.05. The results are averaged over 100 trials and shown in Figure 8.6.

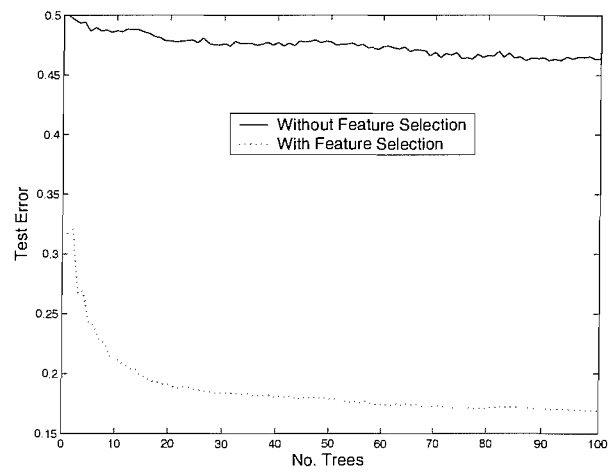

FIGURE 8.6: Comparison of error convergence for Madelon with RF. Top shows convergence without feature selection. Bottom shows convergence with RF HT and confidence level of 0.05.

The improved accuracy that is generated by the feature selection method is immediately apparent, but the effect on the error convergence is also clearly visible. It can clearly be seen that after 100 trees have been added to the forest, the error has not yet converged when all of the features are present. However, when the feature selection scheme is used, the algorithm is close to convergence.

8.8 Local Feature Relevance

This section analyses the potential for local relevance identification within Random Forest. The method that was introduced in Section 7.5 is implemented to construct measures of example-based feature importance. To demonstrate the effect of this method, a simple two-dimensional problem is created and shown in Figure 8.7. The examples are generated by drawing feature values from a uniform distribution on [0,1] and labelling them according to,

$$\begin{aligned} Y &= +1 : \quad X_2 > 2.5X_1 - 0.75 \\ Y &= -1 : \quad \text{otherwise} \end{aligned} \tag{8.2}$$

From Figure 8.7, X_1 appears to be more useful at the global level as it can partition

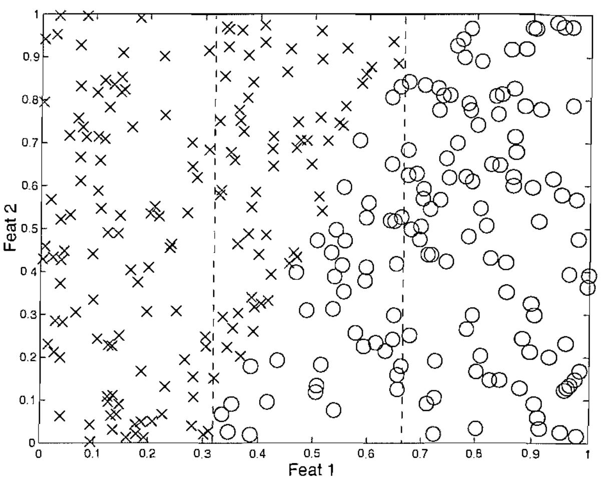

FIGURE 8.7: Artificial Data generated by Equation 8.2 and possible split locations for feature 1, as shown by the dashed lines.

the data as shown by the dashed lines. Feature X_2 does not appear to be as useful at the global level, but can be considered locally useful within the area that is between the dashed lines. This data set is used to construct a Random Forest of 100 trees and record the example-based feature relevance.

Figure 8.8 shows the feature relevance that is observed for the different examples. The top plot shows the original data and the middle and bottom plots show the relevance measures for features X_1 and X_2 respectively. The measures of feature relevance are ordered along feature X_1 so that the bottom two plots represent the importance of the feature as you move across the space from left to right.

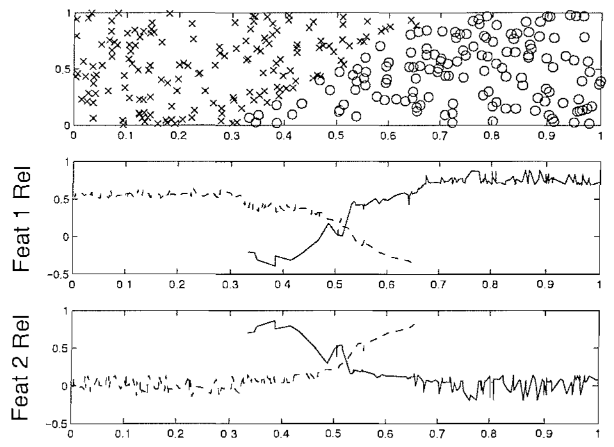

FIGURE 8.8: Observed local relevance for the artificial data. Top plot shows original data, middle plot shows local relevance for feature X_1 (Feat 1 Rel) and bottom plot shows the same for feature X_2 (Feat 2 Rel). The dashed line illustrates the feature importance for the positive examples and the solid line shows the same for the negative examples.

The figure shows the importance of feature X_1 to the outer examples, which are approximately those that are not within the dashed region of Figure 8.7. The opposite effect is observed for feature X_2, which contains very little relevance for the outer examples, but is more relevant to the central area. From the figure, the two features can be considered to have approximately equal relevance within the dashed central area.

The local relevance measures of the features in Figure 8.8 appear to be inversely related to each other and the reason for this is as follows. The local relevance for example x_i and feature X_j is defined as the the average observed reduction in information, required to describe x_i, that is caused by feature X_j.

$$FR(X_j, x_i) = \frac{\sum_{t=1}^{T} \sum_{l \in \mathcal{L}_t} I\left(l^f = X_j\right) I\left(x_i \in \mathcal{X}_l\right) gain\left(l, y_i\right)}{\sum_{t=1}^{T} \sum_{l \in \mathcal{L}_t} I\left(l^f = X_j\right) I\left(x_i \in \mathcal{X}_l\right)}, \tag{8.3}$$

where l^f represents the feature used to partition node l, $\mathcal{X}_l$ represents the subset of data within node l and $\mathcal{L}_t$ is the subset of non-terminal nodes within hypothesis h_t.

The value of $gain\left(l, y_i\right)$ is the reduction in the information length of examples belonging to class y_i, that was yielded from the partitioning of node l. This reduction can sometimes be negative as the information length may increase as an example is partitioned into new nodes. As the trees are grown as large as possible, the leaf nodes of each tree contain examples from only one class and thus produce an information length of zero. Therefore, the total reduction in information length for any example in any tree is given by the information length of the example in the root node of that tree. If the numerator of equation 8.3 is denoted as $FR^{RAW}\left(X_j, x_i\right)$, then the sum of these values for any example across the features can be written,

$$\sum_{j=1}^{F} FR^{RAW}\left(X_j, x_i\right) = \sum_{t=1}^{T} \sum_{l \in \mathcal{L}_t} I\left(x_i \in \mathcal{X}_l\right) gain\left(l, y_i\right). \tag{8.4}$$

This value equals the sum of the information lengths of an example belonging to class y_i in the root nodes of all the trees in the ensemble. Therefore, it is constant for all examples belonging to class y_i.

$$\sum_{j=1}^{F} FR^{RAW}\left(X_j, x_i\right) = \sum_{j=1}^{F} FR^{RAW}\left(X_j, x_{i'}\right) \ \forall \ i, i' : y_i = y_{i'} \tag{8.5}$$

For the example data set, there are only two features and the values for FR^{RAW} for these features must sum to the corresponding constant for the class concerned. If $\mathcal{L}_t(1)$ corresponds to the root node of hypothesis h_t and $Info\left(x_i, \mathcal{L}_t(1)\right)$ denotes the information length of example x_i in this node, as defined in Equation 7.24, then,

$$FR^{RAW}\left(X_1, x_i\right) + FR^{RAW}\left(X_2, x_i\right) = \sum_{t=1}^{T} Info\left(x_i, \mathcal{L}_t(1)\right). \tag{8.6}$$

Therefore, an inverse relationship would be expected between these quantities.

The statistics that are actually employed incorporate the denominator of equation 8.3. However, as the features are selected randomly with equal probability, the value of the denominator is expected to be approximately the same for each feature and simply produces a scaling effect.

$$\sum_{t=1}^{T} \sum_{l \in \mathcal{L}_t} I\left(l^f = X_1\right) I\left(x_i \in \mathcal{X}_l\right) \approx \sum_{t=1}^{T} \sum_{l \in \mathcal{L}_t} I\left(l^f = X_2\right) I\left(x_i \in \mathcal{X}_l\right) \tag{8.7}$$

8.8.1 Localised Feature Sampling

Here, the method of observing and applying this feature relevance is tested on the real and artificial data sets. A Random Forest of 100 trees is constructed on the training data and the example-based feature relevance is recorded. Then, a second forest is constructed on the training data, but this time using the feature information to alter the sampling distribution, as described in Section 7.5. The second forest also consists of 100 trees and is used to classify the test data. The results are averaged over 100 trials and in each trial the data is randomly partitioned into 90% for training and 10% for testing. Table 8.8 compares the results of this method to the standard Random Forest algorithm, Random Forest with the dynamic integration method of Tsymbal et al. (2006) and the results of DIVBOOST W2 from Chapter 5.

TABLE 8.8: Classification error rates of different ensemble algorithms; Random Forest (RF), Random Forest with dynamic integration (RF DYN INT), Random Forest with localised sampling (RF LOCAL) and DIVBOOST W2. The values in brackets are the corresponding variances of test error over the 100 trials.

Data Set	RF	RF DYN INT	RF LOCAL	Divboost W2
WBC	0.0293(0.0004)	0.0301(0.0004)	0.0278(0.0004)	0.0325(0.0005)
Sonar	0.1671(0.0066)	0.1333(0.0060)	0.1552(0.0065)	0.1329(0.0056)
Votes	0.0586(0.0012)	0.0516(0.0011)	0.0477(0.0008)	0.0395(0.0007)
Pima	0.2525(0.0022)	0.2648(0.0024)	0.2283(0.0020)	0.2458(0.0021)
Ionosphere	0.0756(0.0016)	0.0703(0.0015)	0.0683(0.0018)	0.0614(0.0015)
Friedman	0.1670(0.0070)	0.1745(0.0070)	0.1620(0.0066)	0.1445(0.0057)
Simple	0.0870(0.0028)	0.0610(0.0020)	0.0403(0.0015)	0.0240(0.0009)

It can be observed that applying the measures of example-based feature importance to the feature sampling distribution improves the accuracy of Random Forest for all of the data sets. The method also performs better than Random Forest with dynamic integration on all but the Sonar data set. The averaged pairwise Q-statistics and base learner errors are shown in Tables 8.9 and 8.10. These tables show that employing local feature knowledge produces more accurate learners that are slightly less diverse, when compared to standard Random Forest.

The DIVBOOST algorithm performs better for the artificial data sets as these contain irrelevant features, which are mostly removed by the implicit feature selection of the

TABLE 8.9: Averaged pairwise Q-statistic when using the different learning ensembles.

Data Set	RF	LOCAL RF	Divboost W2
WBC	0.6051	0.5882	0.3381
Sonar	0.1267	0.1772	0.0939
Votes	0.4728	0.6475	0.1325
Pima	0.4466	0.5351	0.0409
Ionosphere	0.4326	0.4671	0.2368
Friedman	0.1120	0.2067	0.0228
Simple	0.1928	0.2395	0.1265

TABLE 8.10: Average base learner errors when using the different learning ensembles.

Data Set	RF	LOCAL RF	Divboost W2
WBC	0.0645	0.0529	0.1672
Sonar	0.3518	0.3244	0.3552
Votes	0.1226	0.0762	0.2618
Pima	0.3405	0.3099	0.4336
Ionosphere	0.1708	0.1392	0.2165
Friedman	0.3646	0.3184	0.4093
Simple	0.2669	0.1291	0.1123

DIVBOOST learners. This implicit feature selection may also explain why DIVBOOST performs well for the Votes data set, as this particular problem contains a high level of redundancy.

By focussing on a local area of the space and employing a feature selection scheme within decision tree induction, DIVBOOST can be viewed as performing a degree of local feature selection. To examine the similarities of these methods, the levels of feature importance can be observed as the DIVBOOST method proceeds. The weight, $w_t(x_i)$, that DIVBOOST assigns to example x_i at iteration t, can be combined with the measures of example-based feature relevance to produce a feature relevance measure for hypothesis h_t,

$$divFR_t(X_j) = \sum_{i=1}^{N} w_t(x_i) FR(X_j, x_i). \tag{8.8}$$

This measure represents the importance of feature X_j at iteration t of the DIVBOOST algorithm, that is expected from the local feature knowledge. This value can be compared to the actual change in feature selection that is produced by DIVBOOST. A measure of feature importance that is specific to a decision tree can be formulated as the frequency with which the feature is selected (Frey and Fisher, 2003),

$$divFreq_t(X_j) = \frac{\sum_{l \in \mathcal{L}_t} I(l^f = X_j)}{|\mathcal{L}_t|}, \tag{8.9}$$

where l^f represents the feature used to partition node l and $\mathcal{L}_t$ represents the set of non-terminal nodes within h_t.

Figure 8.9 shows the plots for the different features within the Friedman data set. As the DIVBOOST algorithm progresses, the values of Equations 8.8 and 8.9 are compared. The local feature relevance measures are the averaged observations from the local feature selection experiments involving Random Forest. The plots show how the decision trees alter their feature selection in a manner that is expected from the local feature relevance estimates. For the Friedman data set, features 1 to 5 are relevant and features 6 to 10 are irrelevant. The features 1,2 and 4 are used with a high frequency in the initial stages and are gradually used less frequently as the algorithm progresses. Conversely, the frequency with which the irrelevant features are used gradually increases. However, although there is an apparent degree of correlation between these measures, their relative values between different features are not the same. In the latter stages of the algorithm, DIVBOOST still does not select the irrelevant features as often as the relevant ones. In contrast, the expected feature importance converges to approximately the same level for all features. This illustrates how the local relevance technique with Random Forest can still assign a high degree of perceived relevance to irrelevant features, particularly when examining the ambiguous examples.

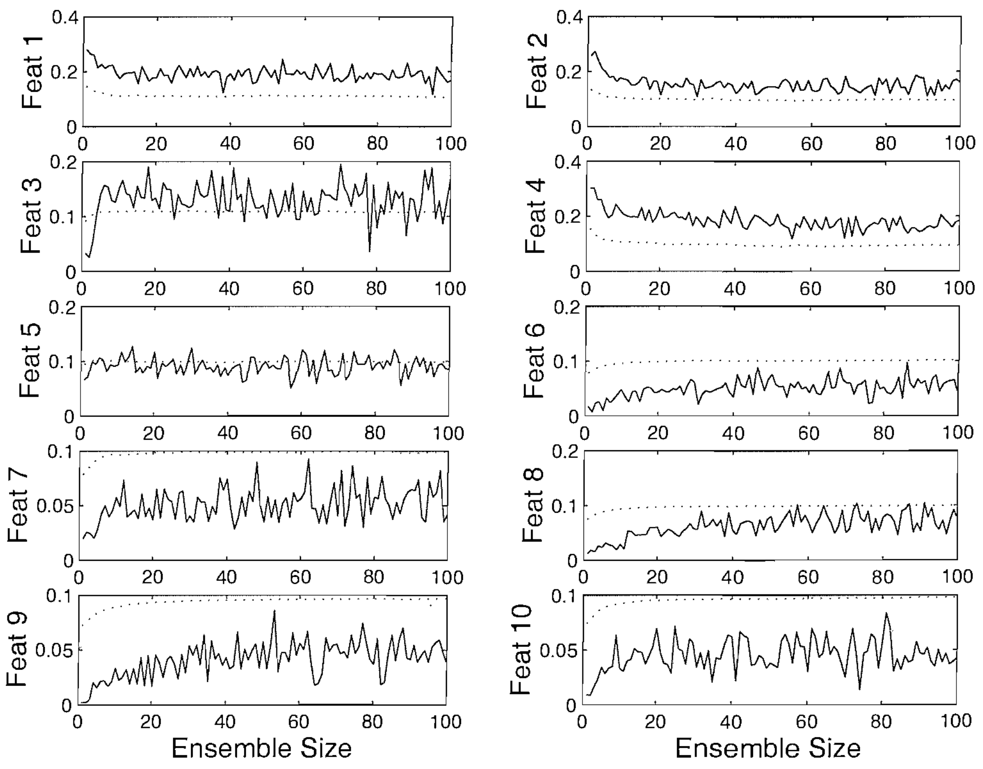

FIGURE 8.9: Measures of feature importance for each of the ten features in the Friedman data set during DIVBOOST operation. Dashed lines indicate expected feature relevance using local feature knowledge and solid lines represent the actual frequency that the feature is used.

To further investigate the potential of localised Random Forest, the method is combined with the feature selection strategy that was previously introduced. A Random Forest of 100 trees is constructed and used to gather the example-based feature information, along with the global measures of information gain for each feature. The hypothesis testing method is conducted with a confidence paramter of 0.025 to remove the features which are perceived to be irrelevant, and then a second forest is constructed using the remaining features. This forest locally alters the feature sampling distribution according to the example-based information that was gathered from the first forest. This second forest is then used to classify the test data and the results are shown in Table 8.11.

TABLE 8.11: Classification error rates when employing Random Forest with different feature selection strategies.

Data Set	RF	RF HT	RF LOCAL	RF HT + RF LOCAL
WBC	0.0293(0.0004)	0.0228(0.0003)	0.0278(0.0004)	0.0264(0.0003)
Sonar	0.1671(0.0066)	0.1910(0.0089)	0.1552(0.0065)	0.1729(0.0072)
Votes	0.0586(0.0012)	0.0575(0.0011)	0.0477(0.0008)	0.0459(0.0010)
Pima	0.2525(0.0022)	0.2530(0.0024)	0.2283(0.0020)	0.2452(0.0023)
Ionosphere	0.0756(0.0016)	0.0744(0.0015)	0.0683(0.0018)	0.0575(0.0013)
Friedman	0.1670(0.0070)	0.1515(0.0069)	0.1620(0.0066)	0.1425(0.0047)
Simple	0.0870(0.0028)	0.0300(0.0010)	0.0403(0.0015)	0.0367(0.0012)

The combination of the two methods compares well to each of the techniques individually. By eliminating irrelevant features, locally weighting the features or combining the two, the Random Forest algorithm can be significantly improved. However, the hypothesis testing method can remove some partially relevant features, which may be useful when employing the localised Random Forest. For example the feature selection scheme removes approximately half of the features of the Pima data set. This reduction in dimensionality has little effect on the classification accuracy of the standard Random Forest, but clearly impedes the performance of localised Random Forest.

8.9 Conclusions

This chapter has shown that the Random Forest algorithm can suffer significantly with the presence of irrelevant features. The lack of implicit feature selection within the algorithm allows irrelevant features to be used which consequently reduce accuracy. Irrelevant features have also been shown to increase the size of the resultant trees and, if there is sufficient data, the algorithm can compensate to some extent by allowing the trees to grow larger. However, larger trees represent an increased computational load and are undesirable. Another reason for the use of feature selection with Random Forest is the effect that irrelevant features have on the error convergence rate. This effect has been demonstrated and an improvement can be achieved with the implementation of a suitable feature selection scheme.

The average information gain achieved during Random Forest construction achieves good identification of feature relevance if treated correctly. The node complexity measure has been shown to improve the reliability of this feature importance by compensating for the node constitution. The algorithms that were introduced in Chapter 7 were implemented to investigate the application of these feature relevance measures. Including all of the features and altering their sampling probabilities allows exploration of the trade-off between improving the accuracy of the base learners and maintaining the diversity within the ensemble. This can be performed by either using a parallel method or a two stage method. One of the problems associated with employing a parallel method is that the algorithm may proceed incorrectly if the sampling distribution is updated too quickly. It is shown that by constructing confidence intervals on the estimates of feature importance, the rate of convergence can be controlled and the stability maintained. Through examination of the expected performance of an irrelevant feature and the employment of hypothesis testing, the feature selection threshold is shown to achieve good performance in terms of improved accuracy and dimensionality reduction.

It has been shown that local feature relevance identification can be achieved with an example-based relevance measure for Random Forest, which was illustrated with a simple example. The measures were also applied to the feature sampling distribution in a manner that was introduced in Chapter 7. This localised feature sampling was shown to capitalise on the local knowledge to improve learner accuracy whilst maintaining ensemble diversity. This technique was also compared to the DIVBOOST method, introduced in Chapter 4. The DIVBOOST algorithm concentrates on ambiguous examples whilst employing decision trees with implicit feature selection. Although the structure of the algorithms are quite different, similarities between their mechanisms have been shown. Whilst focussing on an ambiguous local area of the space, DIVBOOST selects different features in a manner that is consistent with the local feature knowledge that was gained from Random Forest.

Chapter 9

Application to the Relevance of Eye Movements

9.1 Introduction

An important application where machine learning methods have proved useful is that of information retrieval. When users search for particular topics, they rely on the system to identify relevant material from simple text queries. These text queries can often be ambiguous and the search can be complicated further because the target of the search may be unclear to the user. Therefore, it is useful to incorporate some form of feedback concerning the relevance of viewed documents so that the system can formulate a more accurate interpretation of the user's interest. Explicit feedback can be given in the form of the user telling the system which of the search results are relevant, but this is laborious and undesirable. A better approach is to utilise some form of implicit feedback and one way that this can be achieved is through studying the user's eye movements as they examine the documents. Constructing eye movement data and relating this information to the relevance of viewed documents is an increasingly studied machine learning problem.

9.2 Eye Movement Data

Hardoon et al. (2007) analysed the eye movements of subjects as they searched for documents on a given topic. Their data consists of articles from 25 topics, from which they first construct a classifier to identify each topic based on word occurrences. Each document is represented by an example, and the feature values consist of how often each word occurs in the document. These examples can then be viewed as points in a high dimensional space with the dimensionality equal to the size of the dictionary. Using this data, they train a set of linear support vector machines to distinguish each topic from the rest. This produces an 'ideal' weight vector for each topic and enables them to construct a regression model from the eye movement data to the coefficients of these vectors. When a new query is sought, the eye movement information is recorded and the regression model uses this data to construct a new weight vector. Each coefficient represents the perceived relevance of the corresponding word and this weight vector can then be used to predict the relevance of documents based on their word frequencies.

9.2.1 Eye Movement Features

The eye movement information consists of 22 features which describe how a subject examines each word in the document. Salojarvi et al. (2005) explain the different features that are used. A subject's eye movement trajectory can be segmented into fixations and saccades. A fixation occurs when the eye remains fairly motionless and is inspecting a specific area, while saccades are rapid eye movements that occur between fixations. Once the eye movement data is segmented in this way, a set of features can be constructed. The following describes the features that are used with the corresponding number of each feature given in the brackets.

Information concerning the users interest can be contained in their initial reaction to a word and some of the features correspond to the initial processing of the word. Consequently, the duration of the first fixation to the word (6) and this duration as a ratio of the sum of the durations of all fixations (21) are recorded. Other forms of this information are the summed durations of all of the fixations to the word during the first pass (7) and the number of fixations to the word when it is first encountered (2). Salojarvi et al. (2005) also include the fixation duration that preceded the first fixation to the word (5) and the fixation duration that followed the first fixation (8). Two binary features are also used to analyse the differences between initial and late processing. These are whether or not a fixation to the word occurred when the corresponding line was encountered for the first time (3), and when it was encountered for the second time (4).

Three other features were used which cover all fixations to the word. These are the sum of the durations of all of the fixations to the word (14), the number of fixations to the word (1) and the mean duration of fixations to the word (15). These three features contain redundancy through interaction as the mean duration can be calculated from the sum of the durations and their number. Therefore, only two of these features are necessary as they are capable of explaining the third.

The position within the word of each fixation can also provide information concerning the processing of the word. The optimal viewing position when the eye trajectory first lands on a word is approximately a quarter of the way along the word. This position minimises the recognition time of the word and deviation from this optimal position can influence subsequent eye movements. Three position features are included and these are measured as distances from the beginning of the word. They are the distance to the location of the fixation prior to the first fixation on the word (11), the distance to the location of the first fixation on the word (12) and the distance to the last fixation on the word (13). Two other distance features are included that concern the saccades to and from the word. These are the distances between the location of the first fixation on the word to the previous fixation location (9), and the location of the last fixation on the word to the next fixation location (10).

Regressions occur when a fixation is made to a previously read word. Generally, eye movements are controlled by low-level brain functions, but higher level functions can interfere if something needs to be clarified (Salojarvi et al., 2005). Therefore, a regression is indicative of a higher level process and four features are measured to represent this. These are the number of regressions that originated from the word (16), the sum of the durations of these regressions (17) and the sum of the durations of the regressions to the word (19). Sometimes the processing of a word may continue whilst the next word is being read. Therefore, the reaction to the word may cause a regression that initiates from the next word. To incorporate this information, a binary feature is included which states whether or not a regression originated from the following word.

Other features are included that are also indicative of higher level processing. Pupil dilation is one such factor and the mean pupil diameter during fixation to the word (20) is used. Sometimes less important words can be skipped if they are easily predicted and the data records the number of words that are skipped before the fixation onto the word (22).

9.2.2 Identifying Relevant Words

The objective here is to identify when a user regards a word as relevant to their search topic, using the information from their eye movements. To achieve this, it is necessary to first identify the important words for discriminating between the topics. This task is relatively simple as it is just a matter of identifying words that are frequently used in a particular topic and less so in other topics. Therefore, there are no complex interactions expected between the features and a univariate test is suitable. Also, it is not desirable to remove redundancy in the data as all of the words that are indicative of the topic are wanted.

For each topic, the documents are labelled according to whether they are relevant to the topic (+1) or not (-1). The features of this data are the inverse document frequencies for each word.

$$X\left(doc, word\right) = wf\left(word, doc\right) \log\left(\frac{N}{N_{word}}\right) \tag{9.1}$$

where $wf\,(word, doc)$ is the number of times *word* occurs in document *doc*, N is the number of documents and N_{word} is the number of documents which contain *word*.

Each feature/word is tested by employing it to partition the documents. The partition is chosen to maximise the information gain and these measures of gain represent the importance of the word for identifying the topic in question. For each topic, the d most important words are selected and these are provided in Appendix A.

9.2.3 Data Construction

It is now possible to construct a binary classfication problem where the target to be learned is whether or not the user regards a particular word as being relevant to their search topic. Hardoon et al. (2007) collected eye movement data for subjects as they searched for documents on a particular topic. Here, this data is used to form a binary classification task. This is achieved through examination of the examples for which users are inspecting one of the d most discriminative words for their search topic. If the user marks a document as relevant, the eye movement features for all of the examined words that are in the set of most discriminative are kept as an example and allocated to the positive class. If the document is marked as irrelevant, the eye movement features for all of the examined words that are in one of the sets of most discriminative words for any other topic are kept and allocated to the negative class. This data set contains eye movements concerning the inspection of discriminative words, and the learning problem is to use the eye movement information to identify when a user recognises a relevant word.

One limitation with constructing the data in this way is that subjects may not recognise a word as being relevant to their search and, therefore, exhibit similar eye movements to those associated with irrelevant words. Also, the subjects are told to search for a specific topic, but may have interests in other areas and regard words from other topics as relevant to them. Another problem is that the set of words that have been chosen to represent each topic may be good discriminators, but are not necessarily expected to provoke a large human response. For example, the word 'given' is measured as the best discriminator for topic Speeches, but would not be considered synonymous with the topic. Also, the limited number of documents with which to find the best words can result in obscure words being considered useful. The 5th best discriminator for topic Speeches is the word 'quebec', which may be regarded as more relevant to topic Cities. The word 'laws' is measured as the 15th best discriminator for topic Elections, but is possibly more suitable for topic Court Systems. This possible confusion can lead to a user regarding relevant words as irrelevant and vice versa.

The value of d determines the number of examples in the data set. A small value of d will create few examples, but the data will consist of the most important words. Conversely, a large value of d will result in more examples, but will include some less useful words. A value of 15 is chosen for d such that a sufficient number of examples is selected without including too many irrelevant words. The data set consists of 3744 examples, which equates to eye movement information for a total of 3744 viewed words. Of these words, 1594 are relevant to the search topic and labelled positive, and 2150 are relevant to a different topic and labelled negative.

9.3 Experiments

The algorithms that have been described in this work are tested on this data set. The data is randomly partitioned into 90% for training and 10% for testing. Each experiment constructs 100 trees on the training data with which to classify the test data and this is repeated over 100 trials. Each of the algorithms are tested in this way and the results are shown in Table 9.1. A method of always predicting the majority class would result in an error of 0.4257. As these learning techniques achieve accuracies that are significantly better than this, it is evident that the eye movement information can be used as an indicator of user preference.

TABLE 9.1: Ensemble error (Error), average base learner error (B.E.) and average Q statistic (Q-Stat) for different ensemble algorithms.

Method	Error	B.E.	Q-Stat
RF	0.3853(0.0006)	0.4369	0.4174
Local RF	0.3650(0.0005)	0.4424	0.2302
Adaboost	0.3896(0.0006)	0.4712	-0.0493
Divboost	0.3671(0.0005)	0.4743	0.0332
Bagging	0.3860(0.0005)	0.3947	0.9439
Random Rotation	0.4088(0.0006)	0.4161	0.8925
PCA Rotation	0.3823(0.0006)	0.3933	0.9259

Plots showing how the average pairwise Q statistic changes as the ADABOOST and DIVBOOST algorithms progress are shown in Figure 9.1. It can be seen that ADABOOST exhibits a low level of correlation during the initial stages, but this gradually increases as the method proceeds. Conversely, DIVBOOST begins with high correlation and increases the diversity as each learner is added to the ensemble. Table 9.1 shows that ADABOOST and DIVBOOST produce learners with very similar accuracy, but the ADABOOST learners are more diverse and slightly more accurate. However, DIVBOOST produces a significantly more accurate ensemble. This data set contains a large amount of noise and it has already been shown that ADABOOST can be misled by outliers. Figure 9.2 compares the average final weighting of the examples that are produced by ADABOOST and DIVBOOST. ADABOOST gives very high weights for the examples that are most commonly misclassified, whilst DIVBOOST is not misled by these. It can be seen that DIVBOOST allocates lower weights to these outliers, which results in a better ensemble.

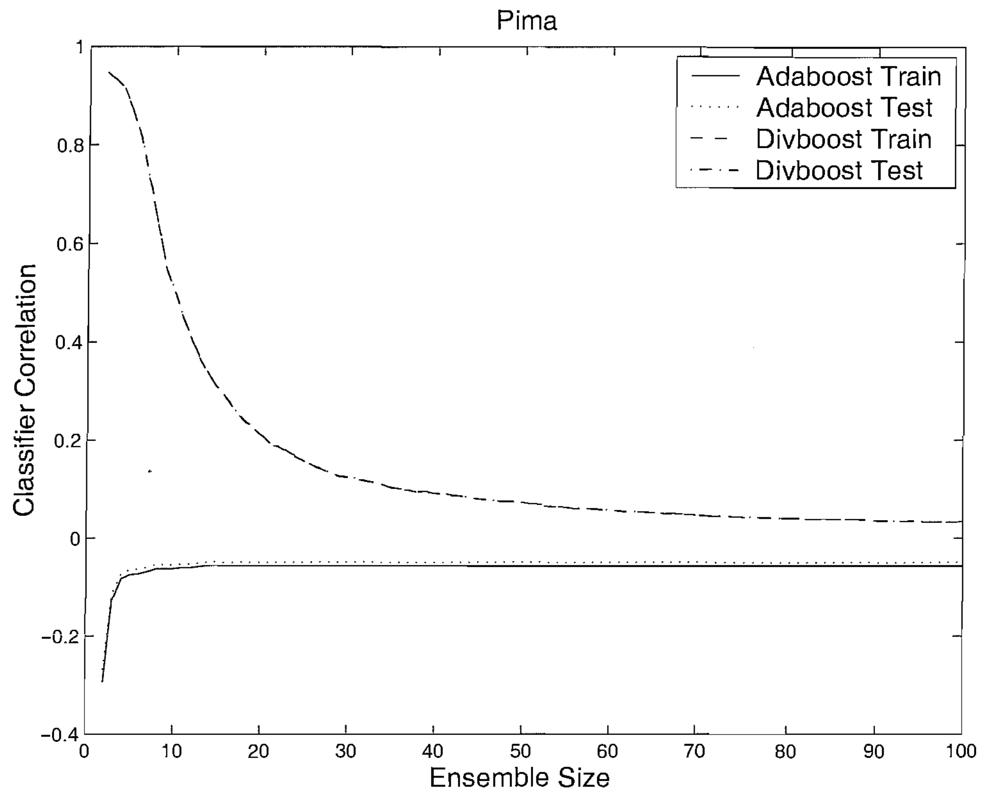

FIGURE 9.1: Plots illustrating how average pairwise Q statistic varies with ensemble size for ADABOOST and DIVBOOST.

Employing localised sampling with random forest results in a significant improvement over the standard algorithm. This method constructs 100 trees using standard Random Forest to gather the local feature information as described in Section 7.5.1. The measures of local feature relevance can be summed across all of the data and are shown in Figure 9.3. Binary and discrete features introduce a bias on the measures of feature importance. These features have a limited set of split positions and consequently achieve lower gain values. Features 1, 2, 3, 4, 16, 18 and 22 are discrete or binary and, except for feature 22, exhibit low measures of relevance. The most useful feature for identifying relevant words to the search topic is feature 21, which is the duration of the first fixation to the word as a ratio of the durations of all fixations. This ratio is more useful than simply using the duration of the first fixation, as with feature 6. The second most discriminative feature is feature 20 which is the mean pupil diameter during fixation to the word. This supports the notion that pupil dilation is indicative of higher level processing.

The localised sampling method produces learners that are less accurate than the standard algorithm, but significantly more diverse. This is unexpected as the feature selection should improve the accuracy of the learners at the cost of reducing some of the diversity. This increase in diversity is possibly due to the local feature selection correctly identifying diverse local models within the data which produces a variety of good

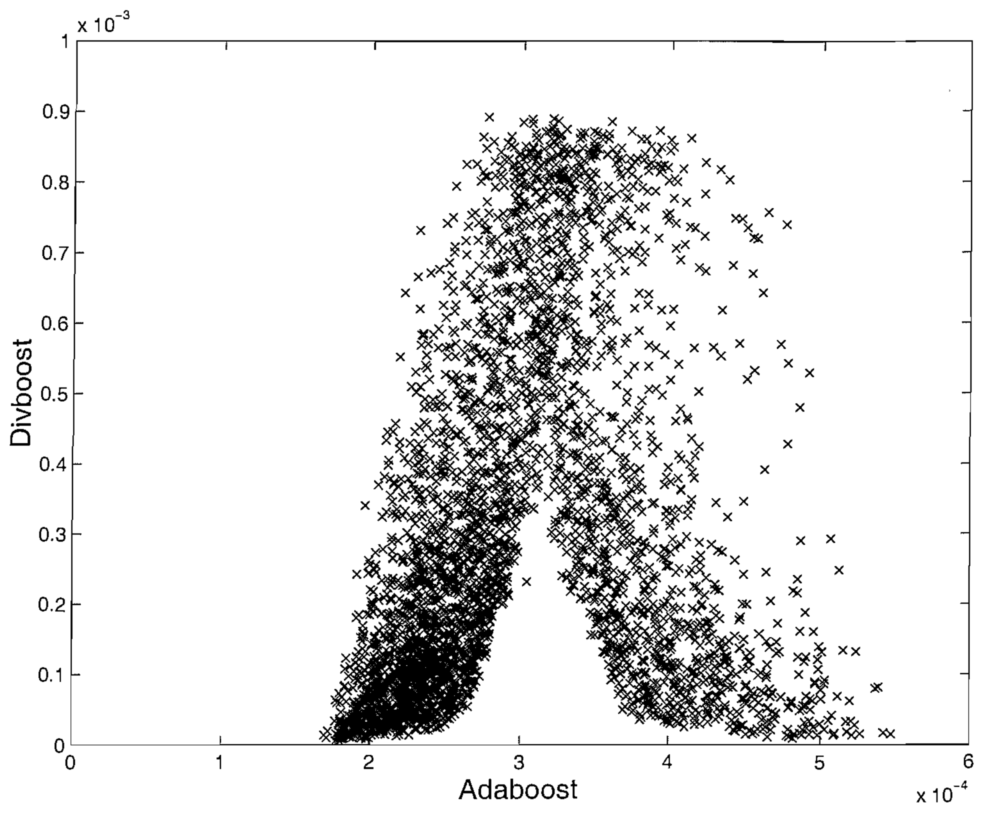

FIGURE 9.2: Comparison of average final weighting given to examples by ADABOOST and DIVBOOST.

representations.

For this data set, PCA rotation produces a significantly more accurate ensemble than random rotation. This is due to a high level of redundancy within the data which results in PCA producing more useful representations. The random grouping of subsets within this algorithm enables a sufficient level of diversity to be created, whilst still gaining benefit from the PCA technique.

Examining all of these methods, localised sampling with Random Forest and DIVBOOST result in the most accurate ensembles. Although they achieve very similar performance, DIVBOOST produces less accurate learners with a much greater level of diversity. Random Forest, Bagging, ADABOOST and PCA rotation are not as accurate as local Random Forest or DIVBOOST, but achieve similar performance to each other. The levels of accuracy and diversity of the base learners of Bagging and PCA rotation are very similar, but the the slight increase in both of these quantities with PCA rotation yields a slightly better ensemble accuracy. The Random Forest learners are less accurate and far more diverse which results in an ensemble that is competitive with these methods. The ADABOOST learners are the most diverse, but due to the problems that the algorithm experiences with outliers, the resultant ensemble is less accurate.

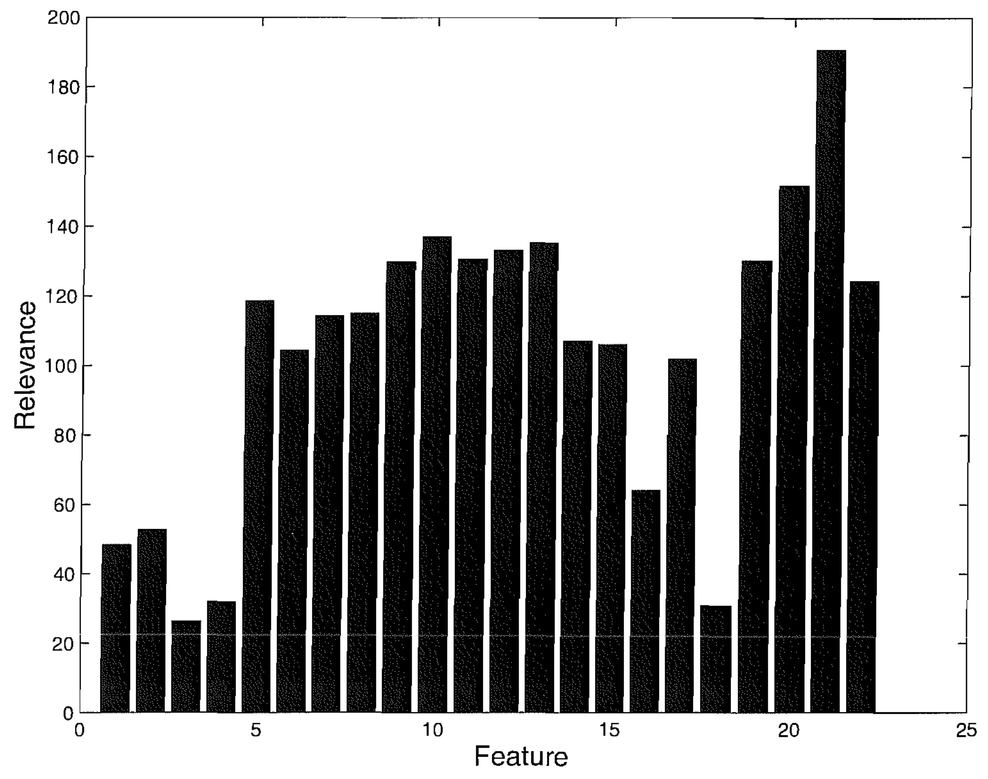

FIGURE 9.3: Measures of global feature importance for eye movement data.

9.4 Conclusions

It can be concluded here that information concerning user preference can be inferred from their eye movement data. The learning techniques employed here utilise eye movements to discriminate between words that are considered relevant and irrelevant by the user. These methods achieve an accuracy that is significantly better than random and advocate the employment of this information. Certain aspects of the data result in some differing levels of performance between these algorithms. It has been previously shown that ADABOOST can perform poorly in the presence of outliers and the high level of noise within this data demonstrates the benefits of employing DIVBOOST. Also, redundancy within this data means that PCA rotation is preferable over random rotation. Random Forest performs well, but can be significantly improved through the use of localised sampling, which employs knowledge of local feature relevance.

The techniques that are demonstrated here can help to determine the relevance of viewed words. To enable this information to improve document retrieval, this can be extended such that the relevance of words is used to rank other documents. A possible way that this can be achieved is through employing a soft-valued output which reflects the degree of relevance to the search topic. Unseen documents can then be ranked according to these relevance measures and their corresponding word frequencies.

Chapter 10

Conclusions

10.1 Summary of Work

A strong relationship exists between the diversity of an ensemble and its accuracy. This thesis has highlighted the need for diversity and has reviewed methods for measuring and promoting this aspect of learning ensembles. Many algorithms have been designed to exploit this concept by manipulating the conditions through which each ensemble member is constructed. These algorithms can be viewed as altering the distribution over the hypothesis space from which each learner is drawn. Through performing this, these methods explore the trade-off between the accuracy and diversity of the base learners. Boosting methods can employ accurate base learners, whilst promoting diversity through progressively focussing on more awkward aspects of the problem. The mechanism of Random Forest improves the performance of the Bagging algorithm by randomising the feature representation to produce a more diverse set of learners. Other methods employ a selection strategy to explicitly measure and optimise the levels of accuracy and diversity.

A new method has been introduced that promotes diversity in an ensemble by focussing the learner on ambiguous examples. This ambiguity was related to the concept of the margin and concentrates the algorithm on the regions of the class boundaries. Therefore, the method produces a diverse set of learners in a similar way to that of other boosting techniques. However, the method possesses the capacity to produce more accurate learners as it was shown to be more tolerant to outliers. Employing this strategy produced significant improvements in accuracy when compared to other well known ensemble methods.

Diversity promotion can be achieved through a manipulation of the feature space, although these methods typically weaken the learners by discarding some feature information. A method has been introduced that retains all of the feature information and exploits the bias of decision tree learners. The technique achieves a high level of diversity through random rotations of the data and has been shown to compare well to other diversity promotion methods.

It is known that the performance of learning algorithms can be improved through the selection of a suitable feature subset. This reduction in dimensionality can result in higher learner accuracy, a reduced computational load and an improvement in the interpretability of an induced hypothesis. Here, it has been observed that, due to their nature, feature manipulation techniques are particularly susceptible to the presence of irrelevant features. The Random Forest algorithm has been shown to suffer significantly with the presence of such features. The lack of implicit feature selection within the algorithm allows irrelevant features to be used and consequently, reduces the accuracy. Irrelevant features have also been shown to increase the size of the resultant trees and if there is sufficient data, the algorithm can compensate by allowing the trees to grow larger. However, larger trees represent an increased computational load and are undesirable. Another motivation for applying feature selection to Random Forest is the effect that irrelevant features can have on the error convergence rate. This has been explained through the concept of the margin and demonstrated with experimental results.

Due to their subset exploration, ensemble algorithms that promote diversity through feature manipulation can be employed to perform feature relevance identification. Random Forest is a particularly useful tool for this purpose, as it provides measures of information gain for each feature. These are more than simply measures of correlation as the features are tested within different feature subsets and, therefore, interactions between the features can be accounted for. These measures of information gain are improved by considering the size and composition of the node being split. A node complexity measure has been introduced to achieve this through considering the information associated with partitioning such a node, and is shown to perform well.

A feature weighting method can be adopted by altering the feature sampling distribution to reflect the learned feature importance. The relative average information gain achieved by each feature provides useful values from which the sampling probabilities can be weighted. A parallel method has been introduced, that employs confidence intervals to control the rate at which the sampling distribution is updated. This method initially produces diverse learners and, through weighting the features, gradually alters the hypothesis distribution to favour more accurate hypotheses. Therefore, the mechanism of this algorithm is in contrast to that of boosting approaches, which employ accurate learners and then alter the hypothesis distribution to favour more diverse hypotheses. Although weighting the features in this manner is shown to benefit Random Forest, the nature of the technique yields non-zero weights for irrelevant features, which can be detrimental to performance. Therefore, further improvement is possible through the identification and removal of these irrelevant features.

The identification of irrelevant features can be achieved by comparing the performance of each feature with an approximation of what is expected from an irrelevant feature. A feature selection method has been introduced, where hypothesis testing is used to identify features that achieve an information gain which is significantly greater than the expected value of an irrelevant feature. Experiments demonstrated that this method achieves good results in terms of classification accuracy and dimensionality reduction.

One aspect of data, which can make feature selection a challenging task, is that groups of features can interact or that certain features may only be relevant within a local area of the space. Consequently, the target information that is provided by a single feature is not a sufficient definition of feature relevance. It has been noted here that employing Random Forest for feature relevance identification can account for this type of feature interaction by testing features in different areas of the space. This knowledge can be exploited further by examining the local performance of the base learners. A new measure has been proposed for recording the feature relevance with respect to particular examples and has been shown to identify variations within different regions of the space. This knowledge can be employed to improve the performance of Random Forest and a technique to achieve this has been introduced. This method updates the feature sampling distribution depending on the area of the space concerned and is shown to improve the accuracy of Random Forest. A relationship between this method and the DivBoost algorithm has been demonstrated. Random Forest with localised feature sampling constructs a diverse set of hypotheses, but employs local feature knowledge to improve learner accuracy. In comparison, the DivBoost algorithm promotes diversity through focussing on a local region of the space and employs implicit feature selection within each learner to identify the local feature knowledge.

The methods that have been developed in this thesis were applied to a data set, where the objective is to infer user preference from eye movements. Due to the high level of noise in this data, the DivBoost algorithm was shown to perform better than AdaBoost. A significant improvement to the performance of Random Forest was demonstrated on this data set, through the employment of the new localised sampling technique. This method and DivBoost produced the best performance for this task, although all of the techniques achieved an accuracy that is significantly better than random. Therefore, these methods have demonstrated the potential for inferring information concerning user preference from eye movements.

10.2 Further Work

The feature selection techniques, that have been introduced here, have been developed for binary classification problems. However, they can be adapted for other supervised learning scenarios and a useful extension to this work would be to alter these techniques so that they can be applied to regression tasks. Instead of employing information gain as the measure of feature importance, a suitable substitute would be the reduction in variance that is caused by a decision tree partition. This would then directly relate the measures of feature importance to the mean-squared error. A suitable measure of node complexity would need to be derived, based on its size and composition, and a feature selection threshold could also be derived through an analysis of the expected performance of an irrelevant feature.

The random rotation technique is a feature manipulation method that promotes diversity whilst not discarding feature information. It would be interesting to investigate possible extensions to this method that form better rotations through an analysis of the feature relevance. Rodriguez et al. (2006) use PCA to form the rotations, while better results may be possible through the employment of supervised feature relevance detection techniques. Diversity promotion methods that function through feature manipulation possess the capacity for feature relevance identification, as they can build up statistics concerning each feature. The random rotation technique is not so suitable for this type of method because the features are constantly changing. However, as this method utilises decision trees, which search for the best feature at each node, the potential exists to guide the rotations using this implicit feature selection. It has been discussed here that each terminal node in a tree can be regarded as an individual learner that is trained on the features that form the path from the root. It is possible that levels of interaction can exist amongst this feature subset and it may be beneficial to perform rotations within these subsets.

It has been observed that algorithms can benefit from the employment of local feature knowledge and that decision tree methods enable the identification of this relevance. A comparison has also been made here between locally sampling the features and the implicit feature selection of decision tree induction when used in conjunction with the DIVBOOST methods. As DIVBOOST progressively focusses on a more localised area of the space, the frequency with which each feature is selected changes. It would be interesting to investigate the utility of employing the knowledge of local feature relevance to boosting algorithms. Instead of using the weighted distribution over the data to alter the sampling probabilities of the examples, the distribution can be combined with the local feature information to adapt a distribution over the features. Any method that updates a distribution over the data can implement this method, provided that the adopted base learner can incorporate some form of feature weighting.

The methods developed here extract local measures of feature relevance, and this information takes the form of a matrix containing relevance values for each feature and for each example. This has proved to be useful in terms of improved classifier performance, although interpretation of this data is not trivial in its current form. This type of method could benefit from the development of algorithms that model this feature information, such that the variation in feature relevance can be understood and the areas of local relevance can be better defined. This would improve the interpretability of the feature information and enable a better understanding of the data that is being learned.

Appendix A

Relevant Words for Eye Movement Data

These tables show the words that were found to be the most useful at identifying each topic.

TABLE A.1: Sets of most discriminative words for each topic.

Topic	**Most Discriminative Words**
Ball Games	game balls tennis players throw side thrown propel playground Square Wall squash indoor four-wall hockey
Dinosaurs	dinosaur species Cretaceous fossil lizard weighing theropod bone teeth carnivorous predators metre plant-eating Jurassic vertebrae
Space exploration	Space NASA probes Pioneer Apollo astronauts Voyager rocket Exploration spaceflight Sputnik solar orbits travels interplanetary
Literature	literature worked stories literary novel epic poet movement novella reader poetry intellectual prose cycles modern
Government	government parliamentary legislature oligarchy ruler dictatorships junta appropriate unicameralism motion might occur described forms highest
Court Systems	court judges jurisdictions jurors juries criminal justices magistrate appellate equivalent judgment staffed indictable inferior Supreme
Cities	city Euphrates city-states sovereign Tigris suburban density still that when does independent area Opis Akkadian
Film	film movie filmmakers often slang Hollywood B-movie dating being started like score synthetic simply style
Sculpture	sculptures marble depicting three-dimensional sculpting statue stone installed durable legs korean jade difficulties Greek need
Natural Disasters	Hurricane storm depression cyclone strength earthquake result drowned westward September strengthening reaching Auguste category wind

Table A.2: Sets of most discriminative words for each topic.

Topic	**Most Discriminative Words**
Education	education students grade School board Kindergarten course States public academic assessment institutions significantly classroom funded
Printing	Printing inks colorized pressing sheet printer master papers black duplicator offset copier Dots darkening adhered
Writing Systems	symbols syllables stenography methods writing represents script grapheme abugida operating translation vowel linear Syllabary text
Optical Devices	lens optical microscopes focal glasses mirror magnify microscopy magnification binoculars polarization convex interference eyepiece minimize
Internet	Internet online pages server protocol websites search contents site audio webpage file producer availability packet
Family	sibling father mother marries household parent half-sister half-brother family role biological sociological fatherhood husband half-siblings
Television	television channel cable serials satellite number featuring factors receiving series action displayed conducted commonplace Journal
Speeches	given speech politicians freedom Quebec farewell retirement troops Destiny Reagan value distant Canada according dollars
Postal System	postal mail postage stamps post deliveries recipient self-addressed courtesy senders sent reply payment forwarding medium
Languages	dialect Indo-European Baltic Latin grammar Northeastern Meroitic ancestral Anatolia variety language spoken proto-language liturgical borrowed
Music	music song improvised term bands group piece thrash popularised Forbidden teutonic crossover instrument composition acoustic
Olympics	Olympic athletes flag medal held Winter Olympiad Committee Pierre Coubertin oath Athens game competitive organised
Astronomy	astronomical celestial star planets galaxy observed astrology physics earth mechanical Jupiter gravitational catalogue horizon Milky
Transportation	vehicle airports bicycles transit wheels mode transport ship intermodal automobile subway railroad contract cargo driver
Elections	elections voter ballot polling electoral voting party candidate suffrage from appear process dimensions attended Laws

Bibliography

H. Almuallim and T.G. Dietterich. Learning with many irrelevant features. In *9th National Conference on Artificial Intelligence*, pages 547–552, 1991.

C. Apte, S.J. Hong, J. Hosking, J. Lepre, E. Pednault, and B. Rosen. Decomposition of heterogeneous classification problems. *Intelligent Data Analysis*, 1280:17–28, 1997.

R. Battiti. Using mutual information for selecting features in supervised neural net learning. In *IEEE Transactions on Neural Networks*, pages 537–550, 1994.

E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, 36:105–142, 1999.

S.D. Bay. Nearest neighbour classification from multiple feature subsets. *Intelligent Data Analysis*, 3(3):191–209, 1999.

R. Bellman. *Adaptive Control Processes: A Guided Tour.* Princeton University Press, 1961.

H. Berthelsen and B. Megyesi. Ensemble of classifiers for noise detection in pos tagged corpora. In *3rd International Workshop on Text, Speech and Dialogue*, volume 1902 of *Lecture Notes in Computer Science*, pages 27–32, 2000.

C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

A. Borisov, V. Eruhimov, and Eugene Tuv. Tree-based ensembles with dynamic soft feature selection. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications.* Springer, 2006.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification And Regression Trees.* Wadsworth, 1984.

C.E. Brodley. Recursive automatic bias selection for classifier construction. *Machine Learning*, 20:63–94, 1995.

C.E. Brodley and M.A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, pages 131–167, 1999.

R. Bryll, R. Gutierrez-Osuna, and F. Quek. Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets. *Pattern Recognition*, 36:1291–1302, 2003.

C. Cardie. Using decision trees to improve case-based learning. In *10th International Conference on Machine Learning*, pages 25–32, 1993.

R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *21st International Conference on Machine Learning*, 2004.

Y. Chen and C. Lin. Combining svms with various feature selection strategies. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, 2006.

P. Cunningham and J. Carney. Diversity versus quality in classification ensembles based on feature selection. In *11th European Conference on Machine Learning*, volume 1810, pages 109–116. Springer, 2000.

A. Cutler and G. Zhao. Pert - perfect random tree ensembles. In *Computing Science and Statistics*, volume 33, 2001.

T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40 (2):139–157, 2000. ISSN 0885-6125.

P. Domingos. Context-sensitive feature selection for lazy learners. *Artificial Intelligence Review*, 11:227–253, 1997.

J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *12th International Conference on Machine Learning*, pages 194–202, 1995.

H. Drucker and C. Cortes. Boosting decision trees. *Advances in Neural Information Processing Systems*, 8:479–485, 1996.

U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *International Joint Conference on Artificial Intelligence*, pages 1022–1029, 1993.

C. Ferri, P. Flach, and J. Hernandez-Orallo. Delegating classifiers. In *21st International Conference on Machine Learning*, 2004.

E. Frank and I.H. Witten. Generating accurate rule sets without global optimization. In *15th International Conference on Machine Learning*, pages 144–151, 1998.

Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

Y. Freund and R. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, 1999.

L. Frey and D. Fisher. Identifying markov blankets with decision tree induction. In *3rd International Conference on Data Mining*, pages 59–66, 2003.

J. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19: 1–141, 1991.

J. Friedman. Flexible metric nearest neighbour classification. Technical report, Dept. of Statistics, Stanford University, 1994.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical report, Dept. of Statistics, Stanford University, 1998.

J. Furnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, pages 3–54, 1999.

P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63:3–42, 2006.

K. Goebel and W. Yan. Choosing classifiers for decision fusion. In *7th International Conference on Information Fusion*, pages 563–568, 2004.

I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors. *Feature Extraction, Foundations and Applications*. Springer, 2006.

M.A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *17th International Conference on Machine Learning*, pages 359–366, 2000.

D.R. Hardoon, J. Shawe-Taylor, A. Ajanki, K. Puolamaki, and S. Kaski. Information retrieval by inferring implicit queries from eye movements. In *11th International Conference on Artificial Intelligence and Statistics*, 2007.

D. Heath, S. Kasif, and S. Salzberg. Induction of oblique decision trees. In *International Joint Conference on Artificial Intelligence*, pages 1002–1007, 1993.

T. K. Ho. A data complexity analysis of comparative advantages of decision forest constructors. *Pattern Analysis and Applications*, 5:102–112, 2002.

T.K. Ho. Nearest neighbours in random subspaces. In *Advances in Pattern Recognition*, volume 1451 of *Lecture Notes in Computer Science*, pages 640–648. Springer, 1998a.

T.K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998b.

S.J. Hong. Use of contextual information for feature ranking and discretization. *IEEE Transactions on Knowledge and Data Engineering*, 9:718–730, 1997.

N. Howe and C. Cardie. Examining locally varying weights for nearest neighbor algorithms. In *2nd International Conference on Case-Based Reasoning*, Lecture Notes in Artificial Intelligence, pages 455–466, 1997.

G.H. John. Robust decision trees: Removing outliers from databases. In *1st International Conference on Knowledge Discovery and Data Mining*, pages 174–179, 1995.

G.H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In W.W. Cohen and H. Hirsh, editors, *Machine Learning*, pages 121–129. Morgan Kaufmann, 1994.

R. Khoussainov, A. Hess, and N. Kushmerick. Ensembles of biased classifiers. In *22nd International Conference on Machine Learning*, pages 425–432, 2005.

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

R. Kohavi, P. Langley, and Y. Yun. The utility of feature weighting in nearest-neighbor algorithms. In *European Conference on Machine Learning*, 1997.

D. Koller and M. Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292, 1996.

A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems*, 7:231–238, 1995.

L.I. Kuncheva and C.J. Whitaker. Ten measures of diversity in classifier ensembles: Limits for two classifiers. In *IEE Workshop on Intelligent Sensor Processing, Birmingham*, 2001.

S. Lee, S. Yi, and B. Zhang. Combining information-based supervised and unsupervised feature selection. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, 2006.

D.D. Margineantu and T.G. Dietterich. Pruning adaptive boosting. In *14th International Conference on Machine Learning*, pages 211–218, 1997.

P. Melville and R.J. Mooney. Constructing diverse classifier ensembles using artificial training examples. In *International Joint Conference on Artificial Intelligence*, pages 505–510, 2003.

P. Mitra. Unsupervised feature selection using feature similarity. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 301–312, 2002.

L.S. Oliveira, R. Sabourin, F. Bortolozzi, and C.Y. Suen. Feature selection for ensembles: A hierarchical multi-objective genetic algorithm approach. In *7th International Conference on Document Analysis and Recognition*, pages 676–680, 2003.

D. Opitz. Feature selection for ensembles. In *16th National Conference on Artificial Intelligence*, pages 379–384. AAAI, 1999.

J. O'Sullivan, J. Langford, R. Caruana, and A. Blum. Featureboost: A meta-learning algorithm that improves model robustness. In *17th International Conference on Machine Learning*, pages 703–710, 2000.

N.C. Oza and K. Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. In *2nd International Workshop on Multiple Classifier Systems*, pages 238–247, 2001.

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

S. Puuronen and A. Tsymbal. Local feature selection with dynamic integration of classifiers. *Fundamenta Informaticae*, 47:91–117, 2001.

J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

M. Robnik-Sikonja. Improving random forests. In *15th European Conference on Machine Learning*, pages 359–370, 2004.

J.J. Rodriguez and C.J. Alonso. Rotation-based ensembles. In *10th Conference of the Spanish Association for Artificial Intelligence*, volume 3040 of *Lecture Notes in Computer Science*, pages 498–506. Springer, 2004.

J.J. Rodriguez, L.I. Kuncheva, and C.J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.

J.D. Rogers and S.R. Gunn. Ensemble algorithms for feature selection. In *Sheffield Machine Learning Workshop*, volume 3635 of *Lecture Notes in Computer Science*, pages 180–198. Springer, 2005.

J.D. Rogers and S.R. Gunn. Identifying feature relevance using a random forest. In *Latent Structure and Feature Selection techniques: Statistical and Optimisation Perspectives Workshop. Bohinj, Slovenia.*, volume 3940 of *Lecture Notes in Computer Science*, pages 173–184. Springer, 2006.

J.D. Rogers and S.R. Gunn. Ensemble diversity and feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007. Submitted.

D. Roobaert, G. Karakoulas, and N.V. Chawla. Information gain, correlation and support vector machines. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, 2006.

J. Salojarvi, K. Puolamaki, J. Simola, L. Kovanen, I. Kojo, and S. Kaski. Inferring relevance from eye movements: Feature extraction. Technical report, 2005.

M.J.J. Scott, M. Niranjan, and R.W. Prager. Parcel: feature subset selection in variable cost domains. Technical report, Cambridge University Engineering Department, 1998a.

M.J.J. Scott, M. Niranjan, and R.W. Prager. Realisable classifiers: Improving operating performance on variable cost problems. In *9th British Machine Vision Conference*, 1998b.

A.J.C. Sharkey and N.E. Sharkey. Diversity, selection, and ensembles of artificial neural nets. In *3rd International Conference on Neural Networks and their Applications*, pages 205–212, 1997.

D.B. Skalak. Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *International Conference on Machine Learning*, pages 293–301, 1994.

M. Skurichina, L.I. Kuncheva, and R. Duin. Bagging and boosting for the nearest mean classifier: Effects of sample size on diversity and accuracy. In *3rd International Workshop on Multiple Classifier Systems*, volume 2364 of *Lecture Notes in Computer Science*, pages 62–71, 2002.

V. Svetnik, A. Liaw, C. Tong, and T. Wang. Application of breiman's random forest to modeling structure-activity relationships of pharmaceutical molecules. In *5th International Workshop on Multiple Classifier Systems*, volume 3077 of *Lecture Notes in Computer Science*, pages 334–343. Springer, 2004.

G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective voting of heterogeneous classifiers. In *European Conference on Machine Learning*, pages 465–476, 2004.

A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Dynamic integration with random forests. In *17th European Conference on Machine Learning*, 2006.

A. Tsymbal, S. Puuronen, and D. Patterson. Feature selection for ensembles of simple bayesian classifiers. In *13th International Symposium on Foundations of Intelligent Systems*, volume 2366 of *Lecture Notes in Computer Science*, pages 592-600, 2002.

K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3):385–404, 1996.

S. Verbaeten and A.V. Assche. Ensemble methods for noise elimination in classification problems. In *4th International Workshop on Multiple Classifier Systems*, pages 317–325, 2003.

L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004a.

L. Yu and H. Liu. Redundancy based feature selection for microarray data. In *10th International Conference on Knowledge Discovery and Data Mining*, pages 737–742, 2004b.

G. Zenobi and P. Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In *12th European Conference on Machine Learning*, volume 2167 of *Lecture Notes in Computer Science*, pages 576–587, 2001.