The Utility of Randomness in Decision Tree Ensembles

by

Fei Tony Liu

Thesis
Submitted by Fei Tony Liu
for fulfillment of the Requirements for the Degree of
Master of Information Technology (Research) (1895)

Supervisor: Dr. Kai Ming Ting
Associate Supervisor: Dr. Wei Fan

Faculty of Information Technology
Monash University
March, 2006
© Copyright

by

Fei Tony Liu

2006
To Jackie, Enoch and Philip
# Contents

List of Tables ........................................................................................................ vi

List of Figures ....................................................................................................... vii

Abstract ................................................................................................................ ix

Acknowledgments ................................................................................................ xi

1 Introduction ......................................................................................................... 1
   1.1 Random decision tree ensemble ................................................................. 1
   1.2 Motivations .................................................................................................. 3
       1.2.1 Absence of deterministic test-selection criteria ................................. 3
       1.2.2 The largely unknown effect of variable randomness ......................... 3
   1.3 Goals ............................................................................................................ 4
   1.4 Approach ..................................................................................................... 4
   1.5 Structure of this thesis ................................................................................ 5

2 Literature Review .............................................................................................. 7
   2.1 Decision tree basic ..................................................................................... 7
   2.2 Decision Tree Ensemble ............................................................................. 10
   2.3 Three aspects of Random Tree Ensembles .............................................. 11
       2.3.1 Statistical aspect ............................................................................... 11
       2.3.2 Computational aspect ....................................................................... 12
       2.3.3 Representational aspect ................................................................... 13
   2.4 Taxonomy of Randomisation Techniques for decision tree ensembles .... 14
       2.4.1 Randomisation before model induction ............................................. 14
       2.4.2 Randomisation during model induction ............................................. 15
   2.5 Related Works ............................................................................................ 16
   2.6 Summary ..................................................................................................... 17

3 Randomness: Benefits and Problems .............................................................. 19
   3.1 Benefits ...................................................................................................... 20
       3.1.1 Mapping non-axis-paralleled boundaries ......................................... 20
       3.1.2 Modelling Hard Boolean Functions .................................................. 23
   3.2 Problems .................................................................................................... 25
       3.2.1 Small disjuncts ............................................................................... 25
       3.2.2 Information Reduction ..................................................................... 25
       3.2.3 Insensible splitting .......................................................................... 26
       3.2.4 Premature stopping ........................................................................ 26
   3.3 Summary .................................................................................................... 27
List of Tables

3.1 Error rate (%) comparison of some hard Boolean functions . . . . . . . . 24
5.1 Experiment 1 results: Complete-random tree ensembles’ average errors (%) 39
5.2 Experiment 1 summary: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 40
5.3 Experiment 3 results: Max-diverse. α with αe estimation average errors (%) 47
5.4 Experiment 3 summary: . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47
A.1 Data attributes for data used in the experiments . . . . . . . . . . . . 54
List of Figures

2.1 An illustration of training set $S$ ................................. 7
2.2 A simple decision tree structure ..................................... 8
2.3 A decision tree structure in disjunctive normal form (DNF) ............ 9
2.4 Ensemble learning, combining multiple models into a final model ........ 10
2.5 Buttrey and Kobayashi's Conjecture on Strength and Correlation ........ 12
2.6 Taxonomy of decision tree randomisation techniques .................... 14

3.1 A rectangular partition along with a sample of positive and negative examples 20
3.2 Multiple decision partitions presented in the instance space ............. 21
3.3 Decision tree boundaries in an instance space ........................... 21
3.4 Curve boundary can be represented by Probability Imbrication in an instance space .......................................................... 21
3.5 Gaussian mixture training samples and optimal boundary .................. 22
3.6 Classification of Gaussian mixture data set using different classifiers .... 23
3.7 Decision segments created by the intersection of partitions ............. 24
3.8 Complete-random tree Ensembles with different leaf sizes and their effects on small disjuncts ........................................ 26

5.1 Compare the best variant PFO with Bagging, Random Forests ............. 38
5.2 Compare the best variant PFO with related variants VFO, PHO, PFB ....... 40
5.3 An example of error rate convergence ................................... 41
5.4 Classify Gaussian mixture using Max-diverse.$\alpha$ with different $\alpha$ values. 43
5.5 Classifying Gaussian mixture data set ..................................... 43
5.6 An example of 'U' or convex shape error rates on $\alpha$ values ............. 44
5.7 An example of 'slide' shape error rates on $\alpha$ values .................... 44
5.8 The formation of 'U' shape error rates on $\alpha$ .......................... 45
5.9 An example of strength and correlation plot ................................ 45

B.1 The data sets in which Max-diverse Ensemble distinctively performs better than Bagging and Random Forests (Part 1) ......................... 55
B.2 The data sets in which Max-diverse Ensemble distinctively performs better than Bagging and Random Forests (Part 2) ..................... 56
B.3 The data sets in which Bagging and Random Forests perform well with ........ 57
B.4 The data sets in which all methods have a similar performance (Part 1) .... 58
B.5 The data sets in which all methods have a similar performance (Part 2) .... 59

C.1 Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 1) ............................................................. 61
C.2 Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 2) ............................................................. 62
C.3 Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 3) ............................................................. 63
C.4 Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 4). ................................................................. 64
C.5 Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 5). ................................................................. 65
C.6 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 1). ................................................................. 66
C.7 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 2). ................................................................. 67
C.8 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 3). ................................................................. 68
C.9 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 4). ................................................................. 69
C.10 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 5). ................................................................. 70
C.11 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 6). ................................................................. 71
C.12 Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 7). ................................................................. 72
C.13 Error rates of different $\alpha$ values and three dimensional error surfaces (Irregular shape). ................................................................. 72

D.1 Strength and correlation plots of data sets that conform to the conjecture (part 1). ................................................................. 74
D.2 Strength and correlation plots of data sets that conform to the conjecture (part 2). ................................................................. 75
D.3 Strength and correlation plots of data sets that conform to the conjecture (part 3). ................................................................. 76
D.4 Strength and correlation plots of data sets that conform to the conjecture (part 4). ................................................................. 77
D.5 Strength and correlation plots of data sets that DO NOT conform to the conjecture (part 1). ................................................................. 78
D.6 Strength and correlation plots of data sets that DO NOT conform to the conjecture (part 2). ................................................................. 79
The Utility of Randomness in Decision Tree Ensembles

Fei Tony Liu
Tony.Liu@infotech.monash.edu.au
Monash University, 2006

Supervisor: Dr. Kai Ming Ting
Kaiming.Ting@infotech.monash.edu.au
Associate Supervisor: Dr. Wei Fan
weifan@us.ibm.com

Abstract

The use of randomness in constructing decision tree ensembles has drawn much attention in the machine learning community. In general, ensembles introduce randomness to generate diverse trees and in turn they enhance ensembles’ predictive accuracy. Examples of such ensembles are Bagging, Random Forests and Random Decision Tree. In the past, most of the random tree ensembles inject various kinds of randomness into deterministic models. Very few of these ensembles considered variable randomness or found it insensitive to performance. In contrast, this thesis uses complete-random tree ensembles as a starting point to investigate the utility of randomness and finishes with a variable random model, capable of finding the appropriate settings of randomness for individual data sets in order to improve predictive accuracy.

Firstly, we construct a taxonomy of tree randomisations to categorise existing randomisation techniques. Then, we analyse the benefits and problems of different randomisation techniques to gain a better understanding of their effects.

Secondly, we find that the key component of random tree ensembles is simply the probability averaging ensemble method. Based on the results of vigorous experimentations, probability averaging brings out the best of complete-randomness in decision tree ensembles. Using this key component alone permits the highest degree of diversity. We name this complete-random tree algorithm Max-diverse Ensemble as it achieves exceptional accuracy by maximising diversity. Interestingly, without the presence of any feature selection criterion, Max-diverse Ensemble’s accuracy is comparable to Random Forests, a popular implementation of random tree ensemble.

Furthermore, visual evidence shows that complete-randomness provides a distinctive representational power to model target concepts. Taking the advantages of this representational power, we propose a decision tree algorithm with variable randomness. It is called Max-diverse.\(\alpha\). Max-diverse.\(\alpha\) forms a smooth convex error-rate contour using different degrees of randomness. In many cases, individual data sets with appropriate settings of randomness achieve better accuracy than the complete-random settings.

Finally, we propose a simple estimation technique for estimating an effective settings of randomness generated entirely from the progressive training errors. Applying the estimation technique, Max-diverse.\(\alpha\) improves significantly from Max-diverse Ensemble. The experimental results show that Max-diverse.\(\alpha\) performs significantly better than Random Forests and comparably with the state-of-the-art C5 Boosting.
The Utility of Randomness in Decision Tree Ensembles

Declaration

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

Fei Tony Liu
March 7, 2006
Acknowledgments

I would like to express my deepest gratitude to Dr. Kai Ming Ting, my supervisor. As a teacher, he tirelessly instructed me over the whole course. As a friend, he unreservedly shared his passion for research with me. During this endeavour to study this course, I thank him that he has provided in keeping this research viable. Especially, I would like to thank him for his critical comments and timely response in providing in-depth insights that result in the success of this research.

I would also like to thank Dr. Wei Fan, my associate supervisor, who has inducted me into the research of random tree ensembles and enlightened me with ideas which flourished into a conference paper.

I thank my wife Jackie for her endless love and support. Apart from giving birth to Philip and looking after Enoch everyday, she continuously assists in reviewing my research including this thesis. For this, I would like to express my love and appreciation.

Special thanks to Anne Wilson and James Tan, who provide helpful advice which makes this thesis more readable.

Above all, I thank God that he has provided all that I need and may his name be glorified.

Fei Tony Liu

Monash University
March 2006
Chapter 1

Introduction

Constructing a decision tree is a natural way to hand down a thought process or a concept to posterity. Decision trees are widely used in our everyday life. We can see them in many different forms where knowledge or concepts are retained and passed on. The research of using decision trees to model human concepts goes back to late 1950s. Hunt et al. (1966) presented Concept Learning System (CLS), an algorithm which is capable of ‘learning’ human concepts from examples. In the late 1970s, based on CLS, Quinlan (1986) developed ID3 with the information-gain heuristic to learn concepts from examples. Simultaneously, from a statistical perspective, Breiman et al. (1984) developed CART. It turns out to be very similar to ID3. In the 1980s, vital improvements were developed including noise handling, continuous attributes, missing values and enhanced splitting criteria. Quinlan (1993) updated the decision-tree induction algorithm and released C4.5, which is now commonly used in the machine learning community.

During the 1990s, the movement of multiple models started. Multiple models learn alternative hypotheses of the concepts, then combine their hypotheses to form the final decisions. To learn alternative hypotheses, Bagging (Breiman, 1996) creates multiple models by training the same learner on different samples of the training data, where samples are randomly selected from original training data. This method is generally applicable to any learning algorithm. More commonly, Bagging is applied on decision trees to form decision tree ensembles. Breiman also found that Bagging generally performs better than a single model. The key to Bagging’s success is a reduction of generalisation error by reducing variance in the results of unstable learners; unstable learners are algorithms whose results can change significantly when their training data is changed slightly.

Starting from Bagging, random elements are added to decision tree construction to create variations that generate alternative hypotheses. In general, we call them ‘random tree ensembles’ and they are the focus of this thesis. Random tree ensembles are capable of handling both classification and regression problems. This thesis addresses mainly the classification problems, which are studied more frequently in the machine learning literature.

1.1 Random decision tree ensemble

A random tree ensemble consists of randomly constructed single decision trees. Decision trees in such ensembles are randomly and independently built. Their results are combined using a function, typically by voting. Random tree ensembles are used to predict the class label of a given case, where class label is a discrete value of interest or importance. For example, given a record of a patient, containing information such as chest pain type, smoking history and various medical measures, the task could be to predict whether heart-disease is present. Possible class labels can be ‘absence’ or ‘presence’ to denote the status
CHAPTER 1. INTRODUCTION

of illness. In this case, a random tree ensemble learns from historical medical records and uses the ‘knowledge’ learned to predict any unseen cases.

In recent years, random decision ensemble implementation has flourished. Examples of such are ‘Randomized Trees’ (Amit and Geman, 1997), ‘Random Subspace’ (Ho, 1998), ‘Tree Randomization’ (Dietterich, 2000b), ‘Random Forests’ (Breiman, 2001) and the completely random ‘Random Decision Trees’ (Fan et al., 2003). Different implementations utilise different randomisation techniques to create diverse trees. Ironically, before the advent of these randomisation techniques, decision tree randomisations are considered as imbecilic ways to induce decision trees. A classical example can be found in (Mingers, 1998), where random construction of a decision tree was seen as a control condition for baseline accuracy. Mingers’ experiment concluded that constructing single random decision trees could not be any better than any deterministic methods. Similarly, the early development of random tree ensembles was not aimed at accuracy enhancement, but rather, it attempted to resolve the dimensionality issue with high dimensional data. An example of such could be found in Amit and Geman (1997).

Amit and Geman (1997) were the pioneers in using randomised trees to resolve high dimensional shape recognition problems. When the number of features increases, the tree size increases exponentially. One way to resolve this dimensionality issue was to prepare multiple small training sets. Each set is constructed by randomly selecting a small number of features. Multiple trees are induced using these training sets. Note that the tree construction process was not altered; a deterministic single decision tree algorithm was used, for example C4.5 (Quinlan, 1993) or CART (Breiman et al., 1984). Here, the preparation of randomly selected multiple training sets is the only random element in the entire tree construction process.

Hansen and Salamon (1990) were probably the first to suggest that accurate and diverse individual classifiers are the necessary condition for a performing ensemble. Amit and Geman (1997) were the first to associate the performance bounds with differences among trees. Breiman (2001) extended their analysis to show that the final accuracy of an ensemble depends on the strength of the individual classifiers and the correlation between them. Correlation in this context can be seen as the opposite term to diversity. Breiman’s analysis is generic and applicable to most classifier base ensembles. Furthermore, the relationship between diversity and ensemble accuracy was discussed by Kuncheva and Whitaker (2003). However, they doubted if measuring diversity could improve ensemble accuracy.

Ho (1998) made her version of random tree ensemble, ‘Random Subspace’ based on ‘Stochastic Discrimination’ (Kleinberg, 1990). Rather than choosing an arbitrarily small subset of features as seen in (Amit and Geman, 1997), her idea was to randomly and independently select half the number of features as subspaces; multiple ordinary decision trees are then grown on these subspaces to form an ensemble. Kleinberg (1990) proved that by combining many different unintelligent classifiers, in much less than the number of all possible classifiers, the ensemble was able to come up with a solution which he suggested as “highly probably perfect”.

Dietterich (2000b) experimented with decision tree randomisation, where the internal feature selection of tree construction was unprecedentedly randomised. As Bagging generates diverse classifiers through the instability of classifier’s algorithm (Breiman, 1996), Dietterich intended to find an alternative method for constructing good ensembles that did not rely on instability. He compared C4.5, Bagging (Breiman, 1996), Adaboost.M1 (Freund and Schapire, 1996) and Dietterich’s ‘Randomized C4.5’, a decision tree ensemble algorithm. When constructing Randomized C4.5, each internal tree node randomly picks a test from the best twenty splits available. He described this method as crude but it performed significantly better than C4.5 and better than Bagging, though insignificantly
better. Breiman (2001) proposed Random Forests, which was improved from Dietterich (2000b) and was inspired by Amit and Geman (1997). In Random Forests, when selecting a test for each node, a smaller set of features are randomly pre-selected before the deterministic feature selection. The number of features pre-selected is depend upon the total number of features in the data set. Arguably, Breiman claimed that Random Forests performs comparably to Adaboost (Freund and Schapire, 1996).

Note that all the above mentioned methods rely on the strength of a deterministic feature selection as the major backbone of the tree construction. For most researchers, deviation from the deterministic feature selection was unpopular because the method had proven successes in single decision tree paradigm. However, Fan et al. (2003)’s experiment shows otherwise.

Fan et al. (2003) proposed a completely random tree model. Their model completely ignored the deterministic feature selection. It went against common belief and show that complete-random models generally performed well. Though individual complete-random trees are weak, building an entire ensemble of complete-random trees produces a strong model. This counter-intuitive behaviour has led to many open questions, which form some of our research questions and motivations in the following section.

1.2 Motivations

Individual trees in a random decision tree ensemble are weak, but combining weak trees creates a stronger ensemble. In contrast, the development of single decision trees has left us with mechanisms for building strong single decision trees. Interestingly, some developments below showed that a number of mechanisms are questionable in the context of tree ensembles.

1.2.1 Absence of deterministic test-selection criteria

A deterministic test-selection criterion is virtually used in every existing decision tree algorithm. Deterministic test-selection criterion is designed to direct single decision trees to grow in a way so that resulting trees are both small and accurate, in another words, the simplest models. Examples of deterministic test-selection are \textit{gain ratio} (Quinlan, 1993) and \textit{gini index} (Breiman et al., 1984). They are essentially heuristics in producing the simplest models.

Fan et al. (2003) were motivated by (1) the high computational cost associated with heuristics that produce the simplest models and (2) the doubt about whether the simplest models can truly approximate the optimal models. Optimal models in this case are decision tree ensembles that can best represent the learning concepts at hand. Fan et al. (2003) proposed complete-random tree ensembles that ignore any deterministic test-selection criterion. However, these ensembles are still performing reasonably well. This finding shows that deterministic test-selection criterion is inessential for random tree ensembles. If it is the case, then one may ask, \textit{what is the appropriate place for heuristics such as deterministic test-selection criterion in random tree ensembles?}

1.2.2 The largely unknown effect of variable randomness

Random Forests (Breiman, 2001) is a popular randomised decision tree ensemble. Examples of Random Forest’s applications are (Furlanello et al., 2003) and (Oh et al., 2003). At the time of writing, Random Forests is the most popular implementation which allows variation of randomness through the use of deterministic feature selection criterion. The mechanism to adjust its randomness is somewhat ad hoc. Breiman (2001) concluded that accuracy was insensitive to that mechanism. If accuracy is really insensitive to different
levels of randomness, it would be logical to always adopt the most random setting that avoids the computational intensive test-selection criterion. However, subsequent development of Random Forests such as Improved Random Forests (Robnik-Šikonja, 2004), Perfect Random Tree Ensembles (Cutler and Zhao, 2001) and Probabilistic Random Forests (Breitenbach et al., 2003) do not reflect the logic of adopting the most random setting and none of them promote variable randomness. At this stage, it is still uncertain how the different degrees of randomness affect the final accuracy.

1.3 Goals

This thesis is an analytical and empirical exploration of the utility of randomness in decision tree ensembles. We aim towards the best way to employ randomness in the construction of decision tree ensembles. Our goals are listed as follows:

1. **Investigate the utility of randomness** - We shall investigate the random decision ensembles, which some refer to as the “black boxes”, to better understand the effect of different randomisation techniques on the ensembles. We shall do so by:
   
   (a) reviewing relevant theories on decision tree ensembles,
   
   (b) understanding the different randomisation techniques and their effects on various aspects of decision tree performance, and,
   
   (c) identifying gaps in our understanding of random decision tree ensembles.

2. **Design a better algorithm** - Some of the contemporary random tree algorithms are somewhat complicated and ineffective. Different randomisations are piled together to generate arbitrary amount of diversity where the effects of different randomisations are difficult to measure. In this thesis, we aim to design a simple and effective random tree algorithm through a more thorough understanding of randomness in decision tree ensembles.

1.4 Approach

Most of the previous attempts in random tree ensembles started by using strong individual trees to construct ensembles, then mounting additional randomisation techniques to further randomise the tree construction. When putting together different randomisation techniques, it is difficult to understand or measure the individual effects as they cannot be separately evaluated. As mentioned in Section 1.2.2 regarding Random Forests, it is very hard to analyse why accuracy is insensitive to the randomness adjusting mechanism as two randomisation techniques are applied at the same time. They are bootstrap sampling and randomised test-selection. This kind of approach results in many obstacles in decision tree ensemble development.

With the motivations and goals set out above, our plan to tackle the issues of randomness is rather different from most of the previous attempts. We first extend Fan et al.’s (2003) approach by starting with complete-random trees. At this stage, our base algorithm will be freed from any randomness restricting mechanisms. By doing so, this base algorithm returns to a primitive form of tree construction similar to CLS. Then, rather than aiming at making the individual tree stronger, we work out a way to make the ensemble stronger instead. This way, we avoid the legacy of single decision trees and include the necessary key components only; it also keeps our algorithm simple, effective and comprehensible.
1.5 Structure of this thesis

This chapter has introduced the context of this thesis and established our position of research in relation to the machine learning community. It also clarifies the motivations, goals and approach of this thesis.

Chapter 2 surveys and reviews literature relevant to the development of random decision tree ensembles. In the first part, we will briefly describe the basic construction of decision trees, followed by multiple models or ensemble learning. The second part focuses on the development of random tree ensembles and several important theories that govern their performance.

Chapter 3 analyses the benefits and problems of utilising randomness in decision tree ensembles. This chapter helps us to better appreciate random tree ensembles and understand the negative impacts of different forms of randomness used in decision tree ensembles.

Chapter 4 provides comprehensive implementation details of proposed methods Max-diverse Ensemble and Max-diverse.α.

Chapter 5 reports on three important experiments, which forge our final proposed methods. Experiments in this section are designed to validate and advance the current stage of random tree ensembles that follow the approach stated in the previous section.

Chapter 6 concludes this thesis and suggests on future works.
Chapter 2

Literature Review

“For lack of guidance a nation falls, but many advisers make victory sure.”

Proverbs 11:14

This chapter surveys and reviews contemporary random tree ensembles that are relevant to the novel ensemble learning methods proposed and studied in this thesis. This chapter provides a more in-depth and detailed view on the contemporary random tree ensembles. It supports our motives in proposing the novel ensemble learning methods. This chapter is divided into five sections. Section 2.1 deals with basics of decision tree construction. Section 2.2 progresses to discuss ensemble learning, particularly ensemble learning with decision trees. Section 2.3 discusses the issues surrounding ensemble learning from three different perspectives to provide a comprehensive view. Section 2.4 surveys the types of randomisations employed by different decision tree ensembles. Brief descriptions and discussions on different categories of randomisations are also provided.

2.1 Decision tree basic

Let us start with a typical classification framework. Consider a training set $S = (X, Y)$, which consists of instances $(x_i, y_i), i = 1, ..., n$, where $x_i$ is known as an input vector in $X$ and $y_i$ is a class variable or output in $Y$. $G$ is the set of possible class labels and $y \in G = \{g_1, ..., g_m\}$. $x_i$ is made up of a number of features $f_1 \ldots f_m \in M$. Each feature describes the characteristics of an instance. Using database terminologies, the training set $S$ can be illustrated in the form of a table, an instance $(x_i, y_i)$ is a record, $x_i$ is the whole record except $y_i$ is the class label field. Figure 2.1 illustrates a typical training set $S$ from (Quinlan, 1993).

Figure 2.1: An illustration of training set $S$
Using the above definitions, we are now able to describe the basic formation of decision trees. Decision trees are made up of instances represented as feature vectors. Feature vectors are connected by nodes. Each node consists of a feature test, and each subsequence branch represents a possible value of the test feature that is being selected. A possible tree structure is presented in Figure 2.2 for illustration. Circles denote nodes, arrows denote feature vectors. The top node is called the root node. Nodes at the bottom ends of trees are leaf nodes.

Given training set $S$, a decision tree is built using a simple recursive routine. Let $DT(S, M)$ be a learning routine that outputs a tree structure. Algorithm 1 is the pseudo code for decision tree construction.

**Algorithm 1** A basic decision tree building routine; $DT(S, M)$

<table>
<thead>
<tr>
<th>INPUT</th>
<th>$S$: Training set, $M$: feature set</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>$T$: tree node</td>
</tr>
</tbody>
</table>

if all $y \in S$ are the same then
    return a leaf labelled with $y$

else if $M$ is empty then
    return a leaf labelled with $\text{class\_statistic}(S)$

else
    select an $f_i \in M$
    construct a node with test label $f_i$
    let $v_1...v_m$ be possible values of $f_i$
    for $i = 1$ to $m$ do
        $S_i = \text{filter}(S, f_i == v_i)$
        $node.branch(i) = DT(S_i, M - f_i)$
    end for
    return node
end if

Using a tree structure, an unseen instance $z$ can be classified from the root node to a leaf node. It travels through the tree structure following the vectors and arrives at a leaf
that determine the predicted class label. In this case, the predicted class label is either \textit{Play} or \textit{Don’t Play}. Note that the tree structure can present arbitrary disjunctions and conjunctions. The above tree structure can also be rewritten as rules in disjunctive normal form (DNF) as in Figure 2.3. Therefore, it can represent any discrete function on discrete features. Discrete features are those features that only allow a set of possible values. For continuous features, it can have a range of possible numeric values. Continuous features can be treated as discrete features after a discretisation process which divides numeric values into discrete bins of interval. Alternatively, for labelling a tree node, a test could be set by using a split point on a continuous feature according to some criteria to divide continuous values.

Though tree structure has good expressive power, inducing the ‘right’ structure or finding the ‘best’ node to split is the real challenge. CLS first divides the instances by their feature values, and then uses a primitive frequency measure to determine the best feature to split. The aim of this primitive frequency measure is to find a feature that is the most correlated to the class distribution (Hunt and Hovland, 1963). ID3 uses an entropy based \textit{info gain} function to determine the best feature to split, which is based on the information theory (Quinlan, 1986). Since info gain biases towards features with more possible values, C4.5 introduced \textit{gain ratio} which is targeted to compensate this bias (Quinlan, 1993). CART, from a statistical prospective, uses \textit{gini index} as the selection criterion (Breiman et al., 1984). A comparison from (Hastie et al., 2001) suggests that info gain and gini index are quite similar. Hastie et al. suggested that there is no real advantage in choosing one over the other.

Both C4.5 and CART are the state-of-the-art single decision algorithms. Their accuracies are among the best in the single decision tree arena. Although single decision trees have a good representational power, they do suffer from two major known problems: \textit{lack of smoothness} in modelling non-axis-parallel boundary boundaries and \textit{difficulty in capturing small structures}. They are the side effects of single decision tree deterministic feature selection, which tends to produce small and accurate single trees. For the first problem, \textit{lack of smoothness}, decision tree feature selection determines that different sections of decision boundaries must align to a feature axis in the instance space. Decision boundaries inescapably have to be rectangular shaped, which is the main cause of this problem. For the second problem, \textit{difficulty in capturing small structures}, single decision tree must always select the best node to split at each incremental learning. Significant
structures in the instance space are always being picked up first in single tree construction. Combining with the accumulative nature of the decision tree, small structures are inevitably ignored, which causes higher misclassification as small structures are not being captured. In summary, it is important to understand these two problems in order to gain insights into how ensemble learning helps to break through the plateau of single decision tree problems and limitations.

2.2 Decision Tree Ensemble

In ensemble learning, individual classifiers work as a team to enhance one another. It is assumed that individual classifiers make different mistakes and the majority of the classifiers produce correct predictions. When combining the results of individual classifiers, their results correct each other and usually give more accurate results, compared to single classifiers. Figure 2.4 illustrates the process of ensemble learning. This idea applies to any type of classifiers. When applying decision trees as base classifiers, a closely knitted relationship of trees and ensembles can be described as follows:

In order to predict a class given a test case \( z \), the predicted class \( g_p \), is obtained by:

- Probability Averaging / Bayesian Voting:

  \[
  g_p = \arg \max_g \left( \sum_{i=1}^{N} \frac{n_{h_i,g}}{n_{h_i}} \right), g \in G
  \]  

  \[ (2.1) \]

- Voting:

  \[
  g_p = \arg \max_g \left( \sum_{i=1}^{N} I\left( \frac{n_{h_i,g}}{n_{h_i}} \right) \right), g \in G
  \]

  \[ (2.2) \]

where \( N \) is the number of trees, \( I(.) \in \{0,1\} \) is an indicator function. Relevant to the test case \( z \), \( n_{h_i,g} \) is the class \( g \) count for tree \( h_i \) and \( n_{h_i} \) is the leaf size for \( h_i \). The indicator function \( I \) returns 1 if \( \frac{n_{h_i,g}}{n_{h_i}} \) is maximum, otherwise returns 0. Since there are many possible \( h \), we define the space containing all \( h \) as \( h \in H \). One can imagine \( H \) is

![Figure 2.4: Ensemble learning, combining multiple models into a final model](image-url)
extremely large. Therefore, to construct experiment on ensembles, especially on testing
the infinite limits on certain theories is extremely difficult.

As technology advances, memory expansion has allowed more to be done on ensemble
learning. Building multiple classifiers is becoming more and more practical with the latest
technologies. It is important to realise that the popularity of ensemble learning is not a
continuation of single classifier development. Rather, single classifier development and en-
semble development has been developed in parallel. The development of ensemble learning
is largely enabled by the increases of memory in computers that make ensemble learning
feasible and practical. When dealing with these two concepts, we should not assume what
works for single decision trees must work for decision tree ensembles and vice versa.

In some cases, techniques for ensemble learning require a weakening of single classifiers
in order to be effective. Let us considers two fundamental ensemble techniques which are
Bagging (Breiman, 1996) and Boosting (Freund and Schapire, 1996). Bagging provides
multiple bootstrap samples for individual learners before combining their results by vot-
ing, and boosting utilises residual errors to adjust instance weights for subsequent training.
Boosting also combines results of individual learners by voting. These techniques both pot-
tentially make individual learners weaker against the entire population of training samples.
Bagging reduces available samples to each learner to two thirds of the original training set.
Boosting modifies training samples distribution that could increase error rates of subse-
quent learners. Ensemble learning techniques make individual learners weaker. However,
ensembles become more accurate through these techniques (Schapire, 1990). This shows
the separation and independence of single classifier development and ensemble learning
development.

To further explore the characteristics of ensemble learning with decision trees, the next
section provides a survey of different theories that interact with decision tree ensembles.

2.3 Three aspects of Random Tree Ensembles

In (Dietterich, 2000a), Dietterich explains why ensembles work better than single models
in three respects. They are statistical, computational and representational. These aspects
provide a starting point for further elaboration.

2.3.1 Statistical aspect

In order to construct more accurate ensembles, a necessary condition is that individual
classifiers are diverse and accurate (Hansen and Salamon, 1990). Following Dietterich’s
(2000a) description of a statistical problem, when a learner constructs a hypothesis, it
 can be thought of a search in the entire hypothesis space $H$ to find the best hypothesis.
The statistical problem becomes obvious when training data size is small as compared
to $H$. When data are too small to fill the hypothesis space, the learner may find many
different hypotheses in $H$ that have similar accuracies on training set. In order to reducte
the risk of choosing the incorrect hypothesis, results are averaged. This idea is formalised
by Breiman (2001), who comes up with a statistical analysis to explain the relationship
between the ensemble's accuracy correlation among individual classifiers and the strength
of individual classifiers. Breiman gives

$$PE^* \geq \bar{\rho}(1 - s^2)/s^2$$  \hspace{1cm} (2.3)

where $PE^*$ is the generalisation error for an ensemble, $\bar{\rho}$ is the mean correlation among
trees, and $s$ is the average strength of individual trees in an ensemble. Strength $s$ corre-
sponds to the accuracy of individual trees and correlation $\bar{\rho}$ corresponds to the variability
of tree structures in an ensemble. Correlation is the opposite of diversity.
In essence, Equation (2.3) also suggests that diversified and accurate individual trees create accurate ensembles. In addition, this equation provides a mathematical mechanism to predict ensemble accuracy where strength and diversity vary. Strength $s$ and correlation $\bar{\rho}$ could be measured using suggested formula in (Breiman, 2001; Kobayashi, 2002).

Furthermore, Buttrey and Kobayashis (2003) conjectured that there is a non-linear relationship between average strength $s$ and the correlation $\bar{\rho}$. In their experiment, they used:

- Adaboost
- Bagging
- Random Forests
- perturbation
- perturbation with sub-sampling
- random splitting with permutation stopping
- random splitting with permutation stopping combined with sub-sampling.

to generate models with different strengths and correlations. Plotting their strengths and correlations, Buttrey and Kobayashis find a non-linear relationship between their strengths and correlations. The essence of this conjecture is that the model with the lowest error rate can be found on a boundary form by these models. Figure 2.5 shows the conjecture.

Note that not all models fall on the boundary in the experiments. Most of them fall in the 'Higher error rates' area of the figure leaving only a few models to support the boundary.

![Figure 2.5: Buttrey and Kobayashis' Conjecture on Strength and Correlation](image)

Putting the puzzles together, if one can come up with a universal algorithm that can vary the strength and correlation on demand, one could use Breiman’s equation as a sliding bar to find the most beneficial setting that provides the lowest $PE^*$. It would help further improve the accuracy of ensembles.

### 2.3.2 Computational aspect

Computationally, ensemble learning could be viewed as many individual classifiers employing a gradient search to minimise an error function over the training set. While Boosting minimises residual errors from training samples, Random Tree Ensembles work on a rather
different function: that is, to minimise the difference between the ensemble’s probability estimation and the true probability of a class label.

In general, Boosting tries to fit the model to the training samples. Therefore, it occasionally suffers from overfitting. Random Tree Ensembles, however, mainly provides an average of probability estimations without trying to fit to the training samples. By the Law of Large Numbers, the averaged probability estimation approaches the true probability as the ensemble size becomes large. This way, we are almost sure that the ensemble accuracy increases monotonically and converges to a limit without the effect of overfitting (Breiman, 2001).

Bayesian averaging is another way to explain why ensembles find a good solution by averaging (Buntine, 1992). One suggestion is that when combining probabilities estimated by many different trees, the final approximation approaches the true probability of predicting class labels. However, no experiment was done to confirm the theory due to practical limitations at that time.

On the other hand, Stochastic Discrimination (Kleinberg, 1990) suggests that (in a time less than surveying all possible tree structures) combining independent ‘unintelligent’ decision trees would result in an approximation that come close to the ‘prefect’ solution. An important assumption to clarify is that some assume that one of the hypotheses in an ensemble is the correct one. However, Stochastic Discrimination assumes that none of the hypotheses is prefect or close to correct, only the combined solution is. Knowing this allows us to break free from the requirement of formulating a single best model in an ensemble. In fact, we should take advantage of this property that allows weak and diverse learners to exist in an ensemble, knowing that the Bayesian averaging or Stochastic Discrimination will make sure the ensemble will find a good solution. Also, that enables us to try different models that were not experimented with before, for further exploration.

2.3.3 Representational aspect

Ensemble learning is regarded as a way to increase the representational power of single models (Dietterich, 2000a). In relation to the two shortcomings of single decision trees described in Section 2.1, this increased representational power may help single decision trees to become better classifiers; hence it improves their performance.

In the past, although some researchers have proved asymptotic representation theorems for ensembles (Hornik et al., 1990), and the proof is generic enough to apply on decision tree ensembles, no practical decision tree algorithm has taken up this advantage. Similarly, Dietterich in his paper considered only AdaBoost has something to do with the representational issue, due to the way it constructs new decision trees to overcome residual errors. He consider other methods like Bagging and Tree Randomisation are acting in a similar way to Bayesian Voting, which primarily addresses only the statistical aspect and has little or nothing to contribute to the representational issue. Thus, the representational aspect of random tree ensembles is largely unexplored. How much do random tree ensembles contribute towards representational issue? Can we find any tree randomising technique that helps to increase the representational power of the decision tree ensemble?

In Section 3, we explore this representational issue in terms of benefits and problems of introducing randomness in tree construction.

But first, to get ourselves acquainted with the techniques in randomising decision trees, the following section provide a comprehensive taxonomy on decision tree randomisation techniques.
2.4 Taxonomy of Randomisation Techniques for decision tree ensembles

The categorisation of randomisation techniques is based on existing methods that are known to us. The taxonomy of tree randomisations is summarised in Figure 2.6.

Figure 2.6: Taxonomy of decision tree randomisation techniques

1. **Randomisation before model induction**
   
   (a) Sample randomisation  
   e.g. *Bootstrap sampling* (Breiman, 1996) and *Wagging* (Bauer and Kohavi, 1999; Webb, 2000)  
   
   (b) Feature randomisation  
   e.g. *Randomized Trees* (Amit and Geman, 1997) and *Random Subspace* (Ho, 1998)  
   
   (c) Data perturbation  
   e.g. *Output Flipping* (Breiman, 2000)

2. **Randomisation during model induction**
   
   (a) Partial-random test-selection  
   e.g. *Tree Randomisation* (Dietterich, 2000b) and *Random Forests* (Breiman, 2001)  
   
   (b) Complete-random test-selection  
   e.g. *Random Decision Trees* (Fan et al., 2003)

2.4.1 Randomisation before model induction

Randomisation before model induction can be applied to samples, or features, or injected as noise. These are generally called sample randomisation, feature randomisation and data perturbation.

In sample randomisation, each training set is formed by random selection from an original data set. Bootstrap sampling used in bagging (Breiman, 1996) is a typical example of this subcategory. Bagging follows a scheme to perform random selections on training samples with replacement. The resulting selection is called bootstrap samples. Each decision tree is given a set of independently generated bootstrap samples, so resulting trees can be different from each other. One way to implemented Bagging is using instance weights. When a bootstrap sample set is formed, weight of an instance represents the number of times the instance appear in the bootstrap sample set. In this case, only an integer is assigned to an instance. In additions, Wagging is a modified version of Bagging using the instance weights. Wagging randomises instance weights using Gaussian noise (Bauer and Kohavi, 1999) or Poisson distribution (Webb, 2000) which allows fractions or non-integer values to be assigned to instance weights. Bagging only allows two thirds of training samples to any classifiers. An advantage of Wagging using Poisson distribution is that the full training set is presented to all of individual classifiers reducing the information loss. Overall, all the available features are used in each training set.

The second subcategory is feature randomisation where a subset of features is randomly selected from all available features. In this case, the original data set with a reduced number of features is usually used as the training set. Examples of feature randomisation
include Randomized Trees (Amit and Geman, 1997) and Random Subspace (Ho, 1998). Randomized Trees randomly selects a predefined number of features to form a sub-training set for each tree. The predefined number is usually much smaller than the total number of features the original training set has. Then again, Random Subspace randomly selects half the number of features each time to form the sub-training set. Note that an ordinary learning algorithm is used to produce a model in this category, which is deterministic in the case of decision trees.

The third subcategory is data perturbation. Breiman (2000) proposed output flipping. For output flipping, the outputs are flipped randomly according to a ratio. Then, the flipped outputs are used as normal in tree construction and ensemble generation. Breiman found that output flipping performs generally better than Bagging.

For all existing methods of randomisation-before-model-induction, the key idea is to generate diverse trees by presenting multiple perturbed training sets to the tree induction process. During tree induction, test-selection criteria react differently to each perturbed training set and hence create diverse trees. However, this should have no effect on complete-random test-selection since deterministic test-selection criterion is generally ignored. One undesirable side-effect of randomisation before model induction is that it reduces or distorts information available to each tree. For example, bootstrap sampling only selects about two thirds of the training samples and Wagging distorts data distribution. As tree growth is supported by the variety of training samples, information reduction may impair tree growth and eventually reduce tree diversity since variation is limited. Also, it is not known whether distorting data distribution has any negative side effect on ensembles. To the best of our knowledge, no credible report has covered these issues.

2.4.2 Randomisation during model induction

Randomisation during model induction is focused on the test-selection in decision tree induction. For random test-selection, there are two subcategories depending on the degree of randomness it introduces.

The first subcategory is partial-random test-selection where a random selection is combined with the test-selection based on certain criteria. Some implementations perform random selection before the deterministic test-selection; others do it after. The examples of partially random test-selection are Tree Randomization (Dietterich, 2000b) and Random Forests (Breiman, 2001). In Tree Randomization, deterministic feature selection is replaced with a migrated one. First, features are ranked using normal feature selection criterion. Then, a chosen feature is randomly selected from among the ranked top twenty features. In this case, a random selection happens after a deterministic feature selection. In contrast, Random Forest first randomly selects $F$ number of features. Then, a deterministic feature selection is applied to the selected features to find the chosen feature. In this case, a random selection happens before a deterministic feature selection. Varying $F$ can be seen as a way to vary the randomness in Random Forests. However, Breiman concluded that accuracy is insensitive to $F$ and he also recommended that $F$ should be set to the first integer that is less than $\log_2 m + 1$. $m$ is the total number of features in a training set. One advantage of Random Forests is that the number of feature evaluations is reduced as $F$ is introduced. It helps to reduce time-complexity.

The second subcategory is complete-random test-selection. Since the test-selection is completely random, in its optimal form there is no need to examine the training data in the tree growing process so trees can be built in the shortest time span. The training data are used only to determine when to stop growing a tree and to provide posterior probability estimates or class labels. An example of complete-random test-selection is Random Decision Trees (Fan et al., 2003).
CHAPTER 2. LITERATURE REVIEW

The usual practice in randomisation-during-model-induction is to “weaken” the test-selection criterion to make it partially random. The partial-random test-selection combines random selection with an ordinary test-selection criterion based on certain schemes. The main reason behind this is to maintain strength for individual trees. In a sense, partial-random test-selection becomes less “greedy”, but still contains a deterministic element which restricts tree diversity. On the other hand, complete-random test-selection goes against the usual practice as the strength of individual trees is not maintained at all. As a result, it generates “weak” but diverse trees. Interestingly, Fan et al. found that complete-random tree ensembles are able to produce reasonable results despite their weakness, from which provides a motivation for further investigation.

2.5 Related Works

Some of the random tree ensemble implementations combine randomisation techniques with other mechanisms. Briefly, we describe them as follows:

Robnik-Šikonja delivered an Improved Random Forests using five feature selection criteria and weighted voting, where each feature selection criterion is applied to a certain number of trees and predictions from trees are weighted by the accuracy of each tree (Robnik-Šikonja, 2004).

Probabilistic Random Forests (PRF) (Breitenbach et al., 2003) is based on Minmax Probability Machine Classification and Random Forests. It outputs probability estimation for correct classification and misclassification at each prediction. The experiment was done using six datasets. The main contribution in this work was not the improvement of accuracy instead it was the ability to access risk without the assumption of data distribution or density.

Kobayashi (2002) proposed two random tree ensemble algorithms in his thesis. They are:

1. Randomized splitting with Permutation stopping
2. Perturbation

With the combination of Randomized Splitting and Permutation Stopping, the Randomized Splitting’s part is similar to the partial-random feature selection in Dietterich’s (2000b) Tree Randomization. Furthermore, the Permutation Stopping permutes outputs when constructing tree nodes to decide when to terminate tree growth. Combining these forms the first proposed method. The second proposed method is Perturbation, it is similar to output flipping. However, the flipping probability is decided by the training data. An output is flipped to values with the probability proportional to the fraction of values in the training data.

Kobayashi used thirteen data sets in his experiments. He also compared his two proposed methods (Randomized splitting with Permutation stopping and Perturbation) with Random Forests, Adaboost and Bagging. Altogether, there were five classifiers in his experiments. According to a Friedman test, the results rejected the null hypothesis that all five methods performs equally well on all thirteen data sets. However, the same test also concluded that all five methods are not statistically significantly different from each other at the level of 0.05 (Kobayashi, 2002). In a sense, there is no advantage in selecting one method over the other in these five methods.

It is interesting to observe that: although Random Forests has added more randomisations than Bagging, Random Forests is not statistically different from Bagging in Kobayashi’s experiments. It casts doubts on the approach of injecting various randomisations in decision tree ensembles.
2.6. **SUMMARY**

Perfect Random Tree Ensembles (PERT) (Cutler and Zhao, 2001) is based on CART. PERT randomises the continuous split point selection and combines trees by Bagging. Split points are selected from two random points in the given samples set. In contrast with a more relaxed Max-diverse Ensemble\(^1\), the subtle differences between PERT’s and Max-diverse Ensemble’s split point selections are that:

- PERT requires two points of different class labels and different values for a continuous feature to form a split point,
- Max-diverse Ensemble requires only two points of different values for a continuous feature to form a split point.

The experiment in Cutler and Zhao (2001) uses fifteen data sets. The major contribution in PERT was a reduction in processing time. PERT’s accuracy is comparable to Adaboost and Random Forests.

**2.6 Summary**

In this chapter, we give a brief account of the construction of decision trees and decision tree ensembles. This provides a common ground for further elaboration on the three aspects of random tree ensembles. Briefly, the statistical aspect concerns the risk reduction in choosing the incorrect hypothesis. The computational aspect sees an ensemble as a searching mechanism which minimises errors. Interestingly, the representational aspect of random tree ensembles is largely unknown due to the fact that decision tree ensembles are difficult to interpret. It becomes one of our goals in this thesis to explore the representational aspect of random tree ensembles. In doing so, we intend to capitalise this unexploited power to improve the performance of random tree ensembles. The rests of the chapter provides a taxonomy of decision tree randomisation techniques and related works. These provide an update in the current field of research and help us to position our research in this area.

The following chapter is devoted to the discussion of the benefits and problems of injecting randomness in decision tree ensembles. We shall attempt to thoroughly explore complete-randomness and its utility to look for clues in further improvements for the current models.

\(^1\)Max-diverse Ensemble is one of the proposed implementations in this thesis
Chapter 3

Randomness: Benefits and Problems

“Carefully compare the opposing army with your own, so that you may know where strength is superabundant and where it is deficient.”

Art of War - Sun Tzu

The purpose of injecting randomness into decision tree ensembles is to create diversity in tree construction. Diverse individual decision trees make it possible for ensembles to survey different possibilities and achieve better accuracy. In the past, injection of randomness into decision tree ensembles appeared to be ad hoc and arbitrary (Ho, 2002). For example, the implementation of tree randomisation (Dietterich, 2000b) randomly selects a feature from the best twenty splits and there is no justification for selecting the number twenty. Another example is Random Forests’ F parameter for randomising feature selection: no justification is provided for its default value, log₂ M + 1, where M is the number of features in a data set. At this stage, the development of decision tree randomisation techniques appears to be fragmented. It is because there are heterogeneous implementations and multiple theories surrounding random decision tree ensembles. In addition, most of the analyses on random decision tree ensembles are too general. They are applicable to most classifier-based ensembles and do not address the intricate idea of being random. This leaves two open questions: (1) why random decision tree ensembles work and (2) what are the key differences between these different random tree implementations.

In search for a better explanation as to why random decision tree ensembles work, Dietterich (2000a) offers three main reasons why classifier ensembles are better than single models; we have briefly discussed them in Section 2.3 of Chapter 2, and they are statistical, computational and representational.

Single decision trees are quite interpretable on their own. However, when combining them in a tree ensemble, they lose the unified tree structure and their comprehensibility is greatly reduced. Some even regard a tree ensemble as a ‘black box’, as it is very hard to interpret or comprehend the model represented inside. Because of this, most papers in random decision tree ensemble only provide analysis for the former two, i.e., statistical and computational. In this chapter, we present a representational view of the benefits in using randomised decision tree ensembles. Also, we associate these benefits with the degree of randomness in tree construction, so that we can fully exploit the representational power of random tree ensembles.

This chapter is organised as follows: Section 3.1 focuses on the benefits of introducing randomness to decision tree ensembles, including the ability to model non-axis-parallel boundaries and capture small structures in the feature space. Using a simple two-feature
CHAPTER 3. RANDOMNESS: BENEFITS AND PROBLEMS

Figure 3.1: A rectangular partition along with a sample of positive and negative examples

data set for illustration, we are able to visualise the mapping to non-axis-parallel decision boundaries in the feature space using random decision trees. In Section 3.2, the problems of introducing randomisation techniques are discussed. Problems such as small disjuncts, information reduction, insensible splitting and premature stopping are presented. In the last section we discuss strategies to maximise the benefits and overcome the problems of introducing randomness, which in turn creates better algorithms for random tree ensembles.

3.1 Benefits

The representational power of a classifier is the ability to express or model a concept. One of the weaknesses of a single decision tree is the lack of smoothness and the inability to model non-axis-parallel boundaries. During our research, we observe that decision trees generate better non-axis-parallel boundaries when we construct them randomly and independently. In most cases, individual trees in a random decision tree ensemble trade off accuracy for the sake of diversity. In return, the ensemble gains an ability to model or express difficult concepts, which is very much needed in decision tree model given the two weaknesses discussed in Section 7 of Chapter 7. They are the lack of smoothness and the difficulty in capturing small structure. In the coming section, we will explore the mechanism of modelling non-axis-parallel boundary using a complete-random decision tree ensemble.

3.1.1 Mapping non-axis-parallel boundaries

We first consider an instance space filled with positive and negative samples. For a decision tree, a rectangular box in the plane that separates the positive examples from the negative examples as seen in figure 3.1 is a partition of a leaf node. It is possible to have several leaf nodes which represent several partitions in an instance space. However, due to the way partitions are constructed using a logical test at each node, they do not intersect with each other. Figure 3.2 illustrates decision tree partitions tiled with each others in the instance space. The common outer boundary of these partitions is known as the decision boundary. This decision boundary separates positive samples from negative samples in the instance space. In a random tree ensemble, typically, trees produce different individual decision boundaries. Figure 3.3 shows that each of the decision boundaries is an approximation of the optimal boundary, but none of them is an exact representation of the optimal boundary. Using combining mechanisms like probability averaging, the ensemble boundary is much smoother than any of the individual decision tree boundaries contributed in building the ensemble boundary. The mechanism behind this is similar to the traditional analog-to-digital conversion, in which the quality of conversion is determined by the sampling size,
i.e. the larger the sampling size, the more accurate the estimation. As the diversity and the number of the individual boundaries increase, the quality of the ensemble boundary increases also. We call this phenomenon *Probability imbrication*. Probability imbrication describes the intricate relationship between partitions from different trees and how they combine to form a more delicate ensemble boundary. Figure 3.4 depicts an ensemble boundary that well approximates the curved optimal boundary. This phenomenon agrees with Breiman’s analysis on strength and correlation in Equation (2.3). As $\bar{\rho}$ decreases, diverse partitions reduce the differences between the ensemble decision boundary and the optimal boundary, hence lowering the $PE^*$. We shall see Probability Imbrication in more details in Section 3.1.2.

To handle non-axis-parallel boundaries, oblique tree (Murthy et al., 1994) is another way to approximate non-axis-parallel boundaries. Different from random tree ensembles, oblique tree constructs oblique hyperplanes rather than axis-parallel partitions. It is known that finding an optimal oblique split is NP-complete. When the boundary is curved, an oblique tree may require many leaves to model the boundary. This thesis focuses on
random tree ensembles which utilise only rectangular partitions. We shall see the extent of representational power achievable merely by rectangular partitions.

Although randomised tree ensemble is not considered to have any capacity in overcoming the representational problem (Dietterich, 2000a), complete-random tree ensembles are able to demonstrate their ability to model a non-axis-parallel boundary using a Gaussian mixture data set found in Hastie et al. (2001). The training data (200 samples) and the optimal boundary for the Gaussian mixture data are given in Figure 3.5.

We compare four different decision tree models to illustrate the capacity to model non-axis-paralleled boundaries. We also relate this ability to the degree of randomness a classifier employs. We select these classifiers in a logical progression, starting with a deterministically built single decision tree: C4.5 is completely deterministic and does not carry a random element. Next we have a slightly randomised Bagging, it utilises bootstrap sampling to randomized tree construction. For more randomisation, Random Forests adds partial-random test-selection upon Bagging to increase randomness. Finally, we employ from Fan et al. (2003) a complete-random tree model to illustrate the capacity of a complete-random tree ensemble. All four models utilise unpruned trees and a simple logical test for splitting. Ensemble classifiers use one thousand trees for each ensemble.

Figure 3.6(a) shows the classification of a single decision tree (C4.5). As expected, the decision boundary appears to be axis-paralleled. Bagging produces a slightly smoother boundary as shown in Figure 3.6(b). The decision boundary is being smoothed slightly, but it is still far from the optimal boundary. As observed from the same figure, the problem of small disjuncts arises from using this randomisation method. It is due to the outliers and the random fluctuation of the data distribution.

In Figure 3.6(c), the decision boundary for Random Forests is insignificantly improved over Bagging. This is probably due to the fact that there are only two features in this data set, and random feature selection is not effective under this condition. The notable improvement as a result of the combination of bootstrap sampling and random feature selection is that it eliminates some of the small disjuncts. However, the decision boundary still resembles the rectangular shape from a single decision tree. On the other hand, in Figure 3.6(d), Complete-Random tree ensemble’s decision boundary appears to resemble the optimal boundary. It captures small details presented in the training data, though it also suffers from small disjuncts.

Browsing from C4.5 in Figure 3.6(a) to Complete-Random tree ensemble in Figure 3.6(d), we can see a development on the capacity to model non-axis-parallel boundary. As C4.5 is completely deterministic and Complete-Random tree ensemble is completely random, this shows a relationship between the ability to model non-axis-parallel and the degree of randomness models have. The higher degree of randomness generally provides better ability to model non-axis-boundaries.
3.1. BENEFITS

Figure 3.6: Classification of Gaussian mixture data set using different classifiers.

(a) unpruned C4.5  
(b) Bagging

(c) Random Forests  
(d) Complete-random tree ensemble

3.1.2 Modelling Hard Boolean Functions

Hard Boolean functions are difficult for single decision tree classifiers to model. Examples of such functions are parity or parity-like functions, for example, chess board functions including the classical Xor or Xor-based functions. With decision trees’ greedy learning strategies, they are known to suffer from myopia. Myopia in this context refers to the fact that choices made locally may not be the best globally. Hard Boolean function presents a data distribution which, even with the complete set of training data, misleads single decision tree classifiers to select the incorrect feature to split. The reason for this is that every relevant and irrelevant feature presents similar results when applying the feature selection function, making it very hard to determine the correct feature to split. In other words, the probability of a deterministic single decision tree to find the incorrect feature to split is 1 or close to 1 (Page and Ray, 2003). Since single decision trees cannot determine the correct split, they usually stop the tree construction early, resulting in smaller trees and fewer leaves than are required to represent a hard Boolean function.

Random tree ensembles, however, relax the ‘best split only’ requirement in tree construction, and therefore, many possible tree structures are surveyed in an ensemble. This results in a higher probability of selecting the correct feature. As long as the correct feature or close enough split point is selected in one of the trees, the ensemble can make use of it to improve accuracy.

From another point of view, when different decision partitions overlap each other, many smaller decision segments are formed. This process of forming small decision segments through decision partition of different decision trees is known as probability imbrication as discussed briefly in Section 3.1.1. Random tree ensembles divide the instance space into arbitrarily small segments, so that they can ‘shatter’ the target concept of a hard Boolean function. Figure 3.7 illustrates the creation of decision segments by overlapping decision partitions of different trees.
A limitation of decision partitions is that it must cover at least one training sample limiting the minimum size of a partition. However, since decision segments are formed by the intersection of decision partitions, decision segments can be much smaller and some of them may not even cover any training sample. Because of this, they can represent the details required in some of the complex functions.

An analogy for describing the role of small decision segments is the tesserae in mosaics. Tesserae are pieces of small objects, usually with different colours, that are fitted together with hundreds or thousands of other tesserae, creating a picture or a mosaic. The reason why smaller decision segments are required is similar to using tesserae in mosaics. Smaller segments can be easily fit into the irregular shape of the target concept despite their different shapes and sizes. Through the arrangement of different coloured tesserae, a bigger and more meaningful picture is presented. In the same way, through the arrangement of small decision segments, the complete target concept is presented. The different color objects can be seen as the different probabilities of small decision segments that render through multiple decision partitions using the combinatorial function. This way, the hard Boolean functions are being modelled using small decision segments. This is what we mean by probability imbrication in which the hard Boolean functions are being modelled by the organisation of the decision segments that are intricately formed by decision partitions from different decision trees.

![Figure 3.7: Decision segments created by the intersection of partitions.](image)

We present some of the early findings in our experiment regarding hard Boolean functions in Table 3.1. Most of the error rates in the table are taken from Liu et al. (2005). All error rates are obtained using 10-fold cross validation, with ensemble size ten thousand trees for ensemble methods. In the columns, three randomisation tree ensemble methods are listed, including Complete-Random, Random Forests and Bagging. They are listed in the order of randomness, from the most random to the least. ‘C4.5’ column is the result of a pruned single decision tree; it is listed for comparison.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Complete-random</th>
<th>Random Forests</th>
<th>Bagging</th>
<th>C4.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>threeOf9</td>
<td>0.2</td>
<td>2.2</td>
<td>3.3</td>
<td>3.1</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>9.4</td>
<td>26.4</td>
<td>29.4</td>
<td>16.1</td>
</tr>
</tbody>
</table>

In Table 3.1, the first data set threeOf9 consists of nine binary inputs of 0 and 1 and an output of true and false. threeOf9 outputs true if consecutively three of nine inputs are 1. The second data set tic-tac-toe consists of the complete set of possible board end games configurations of tic-tac-toe games. It has nine inputs representing the nine squares of the tic-tac-toe board. A square can have a value of ‘x’, ‘o’ or ‘b’. ‘x’ indicates player x has taken the square, ‘o’ indicates player o has taken the square, and ‘b’ indicates a blank square.
Both data sets have the lowest errors with complete-random model and the higher errors with Random Forests and Bagging. The results suggest that the degree of randomness is directly related to the performance of modelling hard Boolean functions. The more random the tree ensemble is, the better the ability in modelling hard Boolean functions.

This section has described two major benefits of randomising decision tree ensembles. They are the abilities of mapping non-axis-parallel boundaries and modelling hard Boolean functions. Evidence shows that these two abilities are directly related to the representational aspect of decision tree ensembles. And both abilities show that increased randomness enhances the performance of random decision tree ensembles. Thus, the key to enhance these representational abilities is to maximise randomness.

3.2 Problems

The implementation of randomisation in tree ensembles is not without its problems. Most literature on Random Decision Ensembles discusses only the positive side of generating randomness in tree construction. Very little of them mentions any of the problems associated with introducing randomness. Without careful analysis, it is difficult to formulate a randomisation scheme that maximises benefits and minimises problems when using randomisation techniques. In this section, we will discuss four problems in decision tree randomisation techniques. They are small disjuncts, information reduction, insensible splitting and premature stopping.

3.2.1 Small disjuncts

One of the side effects of modelling non-axis-parallel boundaries is the problem of small disjuncts. With outliers in the instance space, randomisation produces a grainy boundary surrounding outliers. Such a boundary causes misclassification and is known as the problem of small disjuncts (Holte et al., 1989). The effect of small disjuncts is characterized by a higher convergent limit on error rates which is due to the sensitivity of the entire ensemble toward small details. Unlike other overfitting problems, small disjuncts do not cause any increase of complexity when the size of an ensemble become large; error rate remains constant when the size of the ensemble become large. Using the Gaussian mixture data set, error rates at one hundred trees and five hundred trees are very similar, they are 11.4% and 11.2% respectively.

Restricting the leaf size as seen in (Zadrozny and Elkan, 2001) is a way to reduce this problem. Figure 3.8 illustrates the effect of leaf size restriction in reducing the problem of small disjuncts. Comparing Figure 3.8(a) and 3.8(b), leaf size restriction clearly eliminates some of the small disjunct problems in Gaussian mixture data set. However, an automatic way to decide the optimal leaf size is not available and each data set may require a different leaf size to avoid the problem of small disjuncts. Intuitively, the problem of small disjuncts stems from the increased representational power of random tree ensembles. As representational power increased, outliers are over represented which causes misclassification in predictive tasks. In this light, we may have to find a balance of representational power in order to reduce this problem.

3.2.2 Information Reduction

As discussed in Section 2.4.1 of Chapter 2, three randomisation categories 1(a) sample randomisation, 1(b) feature randomisation and 1(c) data perturbation have one common property, in which the training data are reduced or altered before being presented to the learning algorithm. Sample randomisation methods such as bootstrap sampling on average only provide two thirds of the original training samples to each classifier. An example of
Figure 3.8: Complete-random tree Ensembles with different leaf sizes and their effects on small disjuncts

![Figure 3.8](image)

(a) *leaf size* ≥ 1  
(b) *leaf size* ≥ 3

feature randomisation, Random Subspace (Ho, 1998) reduces the number of features to half before feeding the data to a classifier. Data perturbation such as output flipping distorts information (Breiman, 2000). Wagging is a variant of bootstrap sampling which may not reduce training size depending on the implementations (Webb, 2000; Bauer and Kohavi, 1999), but it also distorts the sample distribution of training data. We conjecture that information reduction prevents ensembles from achieving higher accuracy.

### 3.2.3 Insensible splitting

When a deterministic feature selection finds that there is no further sensible split that can be selected, it stops splitting. Any further splitting is deemed insensible. Complete-random trees have no means to avoid insensible splitting. The problem with insensible splitting is that further splitting inevitably introduces irrelevant features into a target concept description. This would reduce the representational power as each split reduces the training samples available for further splitting. Thus, relevant feature would have a smaller chance to participate in decision making and it reduces the tree’s accuracy.

One way to avoid irrelevant features is to filter them before data are presented to the ensemble learning algorithm. While it can be done quite easily, filtering out features may discriminate sub-concepts that rely on a different set of features for prediction. Thus, it is still worthwhile to investigate ways to distinguish relevant and irrelevant feature during model induction.

### 3.2.4 Premature stopping

The randomisation techniques of Random Forests and Random Subspace always blind the test-selection criterion from some features. If all relevant features are blinded from the test-selection criterion, the tree construction stops. This is known as premature stopping. To calculate the probability of premature stopping in Random Forests, we give:

\[
P(\text{premature stopping}) = \prod_{i=1}^{F}(1 - \frac{f_r}{f_t - (i - 1)})
\]

where *F* is the randomness parameter, *f_r* is the number of relevant features, *f_t* is the total number of features in the data set and *i* is inserted for enumeration.

Using *dna* data set from UCI (Blake and Merz, 1998) as an example, a deterministic single decision tree on average uses only four features to classify a test case, which means
only four features are relevant to test a case as far as a deterministic decision tree is concerned. There are in total \( f_t = 60 \) features, which means on average \( f_t = 56 \) features are irrelevant to a test case. \( F \) parameter used in Random Forests is \( F = 5 \) using the default calculation. So, Random Forests will pre-select 5 features for test-selection. Also, there are only 4 features that produce a positive information gain among all the features at the current node, \( f_r = 4 \). Therefore, using Equation 3.1, the probability \( P(\text{premature stopping}) \) for Random Forests on \( dna \) is 0.699. As the probability of premature stopping is greater than 0.5 in the case of \( dna \) data set. The analysis show that there is a very high chance that leaves are formed too general and not specific enough for the problem at hand, hence reduces the predictive power.

### 3.3 Summary

In this chapter, benefits and problems of random decision ensembles are briefly discussed. The benefits of randomness are the ability to map non-axis-parallel boundaries and model hard Boolean functions. These benefits are attributed to the superiority of random decision tree’s representational power. As identified, the key to maximise these benefits is to maximise randomness in decision tree construction.

The problems of introducing randomness are small disjuncts, information reduction, insensible splitting and premature stopping. They contribute to misclassification in the above mentioned ways. To minimise their effects, one way is to find the most useful randomisation technique that provides adequate randomness, and leave the others behind.

To advance the notion of random tree ensembles, finding a way to preserve their benefits and avoid the problems, as staged, is necessary to move this idea forward. We shall see two implementations in the next chapter that attempts to do just that.
Chapter 4

Implementations

“The creation of a thousand forests lies in one acorn.”

- Ralph Waldo Emerson

This thesis proposes two original implementations of building random tree ensembles, they are:

- Max-diverse Ensemble - a complete-random tree ensemble model.
- Max-diverse.$\alpha$ - a variable-random tree ensemble model which uses Max-diverse Ensemble as the base model. An estimation procedure $\alpha_e$ for selecting an appropriate $\alpha$ value is also included.

We group these implementations in this chapter to provide a comprehensive and coherent documentation, so that interested readers may replicate the proposed implementation easily. For the physical implementations, we use a single tree implementation C4.5 as the foundation. However, the algorithmic description in this chapter is not limited to any particular single tree implementation. Thus, interested readers may replicate them using any preferred implementation. In this chapter, Section 4.1 details the implementation of Max-diverse Ensemble. Section 4.2 presents the full picture of Max-diverse.$\alpha$.

4.1 Max-diverse Ensemble

Our first implementation is Max-diverse Ensemble. It is aimed at implementing the complete-randomness in decision tree ensembles. Therefore, deterministic elements in decision tree construction are deliberately taken out. We define it using three algorithms, they are presented in Algorithm 2,3 and 4.

Our implementation is inspired by the Random model in (Fan et al., 2003), but the distinct differences in this implementation are as follows:

- Max-diverse Ensemble promotes unrestricted tree growth,
- Random split point selection for continuous feature selection is formalised, and
- Missing values are handled differently.

In Algorithm 2, $MDE()$ mainly deals with the formation of tree collections. It is given a simple task to iterate according to the number of trees required to produce. Since trees are grown independently, they can be grown in parallel to take advantage of parallel computing.
Algorithm 2 $MDE(S, M, N)$ - building complete-random tree collection

**INPUT** $S$: Training set, $M$: Feature set, $N$: Number of trees

**OUTPUT** $E$: a collection of trees

for $i = 1$ to $N$

\[ E = E \cup RDT(S, M, 1) \]

end for

return $E$

Algorithm 3 $RDT(S, M, W)$ - building a complete-random decision tree

**INPUT** $S$: Training set, $M$: Feature set, $W$: Decision weight

**OUTPUT** $T$: tree node

if all $y \in S$ are the same

return a leaf labelled with $y$

else if $M$ is empty

return a leaf labelled with $\text{class}_\text{statistic}(S) \times W$ \{Preparing leaf node\}

else

randomly select an $f_i \in M$, which is possible to split in $S$

construct a node with test label $f_i$.

if $f_i$ is a continuous feature then

\{Continuous feature handling\}

$\text{node.splitpoint} = \text{randomsplit}(f_i, S)$

$S_1 = \text{filter}(S, f_i > \text{node.splitpoint})$

$S_2 = \text{filter}(S, f_i \leq \text{node.splitpoint})$

$\text{node.branch}(1) = RDT(S_1, M - f_i, W)$

$\text{node.branch}(2) = RDT(S_2, M - f_i, W)$

else

\{Discrete feature handling\}

let \{\(v_1...v_m\)\} be possible values of $f_i$

for $i = 1$ to $m$

\[ S_i = \text{filter}(S, f_i == v_i) \]

\[ \text{node.branch}(i) = RDT(S_i, M - f_i, W) \]

end for

end if

if $f_i$ contains missing values in $S$ then

\{Missing value handling\}

$S_{\text{missing}} = \text{filter}(S, f_i == \text{missing})$

$\text{node.branch}(\text{missing}) = RDT(S_{\text{missing}}, M - f_i, W \times \frac{\text{count}(S_{\text{missing}})}{\text{count}(S)})$

end if

return $\text{node}$

end if

Algorithm 4 $\text{RandomSplit}(f_c, S)$ - Random split point selection

**INPUT** $f_c$: a continuous feature, $S$: training data

**OUTPUT** a split point

$r_1 = \text{randomly select a value of } f_c \text{ in } S$

$r_2 = \text{randomly select a value of } f_c \text{ in } S$

while $r_1 == r_2$

\[ r_2 = \text{randomly select a value of } f_c \]

end while

return $(r_1 + r_2) / 2$
In Algorithm 3, \( RDT() \) details the core implementation of Max-diverse Ensemble; procedures for handling continuous features, discrete features and missing values are provided for easy implementation. \( RDT() \) extends from \( DT() \) in Section 2.1 of Chapter 2 in constructing probability estimating random trees. When preparing a leaf node, the \( \text{class}\_\text{statistic}(S) \) function in \( RDT() \) basically returns the class counts for the given data set \( S \). Class counts are recorded in the form of probability estimates, \( \frac{\text{count}(g_i, S)}{\text{count}(S)} \), \( g_j \in G \).

When selecting a feature \( f_i \) for branching, an important precondition is that \( f_i \) is possible to split in \( S \). It means that \( f_i \) must be able to create at least two branches in \( S \) or \( f_i \) must have different values in \( S \). Note that a continuous feature is allowed only to appear once in a single path for tree height control required in the experiment. This condition is lifted in Max-diverse in the next section.

Although the conventional weight disseminating routine in C4.5 helps individual trees to be better single tree classifiers. Through some initial experiments, we find that the combination of weight disseminating routine, probability averaging and bootstrap sampling has a negative impact on the final accuracy. When applying bootstrap sampling, some of the training cases are over-sampled. Combing with the possible ‘wrong guesses’ that made by the weight disseminating routine, wrong guesses can be magnified multiple times making it very hard for other classifiers in the ensemble to correct. In some cases, error rates increase progressively as the size of the ensemble increases. In this implementation, we propose an alternative method to avoid using weight disseminating routine in complete-random tree ensembles, where Missing values are handled by:

1. treating ‘missing values’ as another possible value
2. introducing a decision weight reduction scheme for missing values.

This avoids the ‘guess work’ arising from the usual missing value weight disseminating routine, and at the same time, provides equitable decision combinations of different trees.

The decision weight \( W \) is inserted to handle missing values; it is designed especially to handle missing values in an ensemble environment. Missing values carry ambiguities in decision making. To identify these ambiguities, decisions involved with missing values are discounted with reduced weights. Since \( RDT() \) is a recursive implementation, decisions involved with more missing values are discounted further. When combining weighted decisions from individual trees, reduced decision weights reflect the certainty of decisions, which make the decision combination sensible.

Through observations, we find that individual trees that are built with missing value branches are weaker than those built with a weight disseminating routine. However, combining these weak trees produces a stronger ensemble.

Furthermore, we formalise the random continuous split point selection which is not described in previous literature (Fan et al., 2003). We propose that random split point is determined as the mid point of two randomly selected distinct sample values. The random continuous split point selection is illustrated in Algorithm 4. We find that trees built with random split point selection may be weaker than those built with conventional split point selection. However, random split point selection is able to produce trees with higher diversity and hence further lower generalisation errors of ensembles.

To understand why random split point selection is able to provide more diversity, a comparison between random split point selection and conventional split point is necessary. In constructing decision trees, possible split points are considered before selecting a continuous feature as the feature test for a node. With deterministic decision trees, only the split point with the highest gain will be selected. Let there be \( l \) number of distinct values for a continuous feature at a certain node, then possible split points are mid points between two neighbouring distinct values. In total, there are \((l - 1)\) possible split points for ordinary continuous split point selection. For complete-random tree ensemble,
the number of possible split points is \( sp, C^l_2 \geq sp \geq (l - 1) \) which creates more diversity. Again, possible split points have equal chances of being picked, which also increases tree diversity.

When evaluating the ensemble \( E \) in Algorithm 2, probability averaging as seen in Equation 2.1 from Section 2.2 Chapter 2 is applied.

In summary, Max-diverse Ensemble provides a realisation of complete-randomness in decision tree ensembles. It is to serve as a foundation and benchmark for further research. In the next section, we present Max-diverse.\( \alpha \), a variable-random decision tree ensemble, which is an implementation that is based on Max-diverse Ensemble.

4.2 Max-diverse.\( \alpha \)

The existing methods randomise decision trees by mounting different types of randomisations on top of deterministic models, making it rigid and inflexible. Randomisations are either on or off, hence there is no middle ground.

Max-diverse.\( \alpha \) is a novel decision tree ensemble algorithm of variable randomness which is based upon Max-diverse Ensemble. The implementation of Max-diverse.\( \alpha \) is described in Algorithm 5 and 6.

The design principle is to control precisely the amount of deterministic feature selection used in complete-random tree ensembles. This way, we limit ourselves to only two well understood techniques, deterministic feature selection and complete-random feature selection, so avoiding the side effect of other randomisation techniques.

\[
\text{Algorithm 5 } MDE.\alpha(S, M, N) - \text{ A variable-random tree ensemble routine}
\]

\[
\begin{aligned}
\text{INPUT } & S: \text{ Training set, } M: \text{ Feature set, } N: \text{ Number of trees} \\
\text{OUTPUT } & E: \text{ a collection of trees} \\
\text{initialise } & \text{ bestTrainErr and } E \\
\{\alpha_e \text{ estimation} \} \\
\text{for } & \alpha_{\text{temp}} = 0 \text{ to } 0.5 \text{ in step of } 0.05 \text{ do} \\
& \text{ let } TrainErr = 0 \\
& \text{ clear } E_{\text{temp}} \\
& \text{ for } i = 1 \text{ to } N \text{ do} \\
& \quad E_{\text{temp}} = E_{\text{temp}} \cup RDT.\alpha(S, M, 1, \alpha_{\text{temp}}) \\
& \quad TrainErr = \text{TrainErr} + \text{evaluate}(E_{\text{temp}}, S) \\
& \text{ end for} \\
& \text{ if } \text{bestTrainErr} > \text{TrainErr} \text{ then} \\
& \quad \text{bestTrainErr} = \text{TrainErr} \\
& \quad E = E_{\text{temp}} \\
& \text{ end if} \\
\text{end for} \\
\text{return } E
\end{aligned}
\]

Max-diverse.\( \alpha \)'s closest related algorithm is Random Forests. The major differences between Random Forests and Max-diverse.\( \alpha \) are:

- Random Forests uses both deterministic and random feature selections in each node and Max-diverse.\( \alpha \) only uses either one of them in each node.

- Max-diverse.\( \alpha \) provides a performance sensitive parameter for setting different degrees of randomness.

- The \( \alpha_e \) estimation procedure turns Max-diverse.\( \alpha \) into an adaptive model that changes its randomness parameter according to the data set.
Algorithm 6 $RDT.\alpha(S, M, W, \alpha)$ - A variable-random decision tree building routine

**INPUT** $S$: Training set, $M$: Feature set, $W$: Decision weight, $\alpha$: $0 \leq \alpha \leq 1$

**OUTPUT** $T$: tree node

if all $y \in S$ are in one $g_j \in G$ then

return a leaf labelled with $g_j$

else if $M$ is empty then

{Preparing leaf node}

return a leaf labelled with $class_{statistic}(S) \times W$

else

let $r$ be a random value, $0 < r \leq 1$

if $r \leq \alpha$ then

{Split point is selected in deterministic feature selection}

select an $f_i \in M$ deterministically

else

randomly select an $f_i \in M$, which is possible to split in $S$

if $f_i$ is a continuous feature then

$node.splitpoint = randomsplit(f_i, S)$

end if

end if

construct a node with test label $f_i$.

if $f_i$ is a continuous feature then

{Continuous feature handling}

$S_1 = filter(S, f_i > node.splitpoint)$

$S_2 = filter(S, f_i \leq node.splitpoint)$

$node.branch(1) = RDT.\alpha(S_1, M, \alpha)$

$node.branch(2) = RDT.\alpha(S_2, M, \alpha)$

else

{Discrete feature handling}

let $\{v_1...v_m\}$ be possible values of $f_i$

for $i = 1$ to $m$ do

$S_i = filter(S, f_i == v_i)$

$node.branch(i) = RDT.\alpha(S_i, M - f_i, W, \alpha)$

end for

end if

if $f_i$ contains missing values in $S$ then

{Missing value handling}

$S_{missing} = filter(S, f_i == missing)$

$node.branch(missing) = RDT.\alpha(S_{missing}, M - f_i, W \times \frac{count(S_{missing})}{count(S)})$

end if

return node

end if
When training the model, progressive training errors are collected for $\alpha$ estimation. Progressive training error is the accumulation of training errors for the whole ensemble growing from one tree to its full size. There are two difficulties in using training errors to estimate or predict the testing errors. First, it is possible for all ensemble training errors of different $\alpha$ to reach zero, making them very hard to compare. Second, when $\alpha > 0.5$, the greediness of deterministic test-selection fits tree structures to the training samples favouring less random ensembles, creating exceptionally low training errors which bias the selection. To overcome these difficulties, we estimate an effective $\alpha$ that is based on the average of progressive training errors. When constructing an ensemble, progressive training errors can be obtained by evaluating training data after adding each decision tree into the ensemble. The average progressive training error reflects the rate of training error convergence from first tree to the last tree. An effective $\alpha_e$ for each data set is estimated as follows:

$$\alpha_e = \arg \min_{0 \leq \alpha \leq 0.5} \frac{1}{t} \sum_{i=1}^{t} err(\alpha, i, S)$$  \hspace{1cm} (4.1)

where $t$ is the total number of trees in an ensemble, $err()$ returns the training error rate of an ensemble of size $i$, set at $\alpha$ and the training samples $S$. After obtaining $\alpha_e$, Max-diverse.$\alpha$ employs the model with $\alpha_e$ for actual predictive tasks.

In Algorithm 5, the $\alpha_e$ estimating procedure samples eleven ensembles with different $\alpha$ values and obtains their progressive training errors for comparison. However, at any one time, only two ensembles are kept in the memory which reserves memory use. In $MDE.\alpha()$, function $evaluate(E_{temp}, S)$ returns an error rate based on ensemble $E_{temp}$ and training data $S$. Progressive training errors are accumulated by a $TrainErr$ variable for comparison. The mechanism is to seek out the $\alpha$ that incurs the lowest progressive training error. The rationale for using progressive training errors in choosing $\alpha$ is that it represents the overall performance of an ensemble instead of a snapshot of the ensemble at a particular size. Observation shows that progressive training error is a more sensitive indicator on the ensemble’s performance.

In Algorithm 6, the deterministic feature selection and the deterministic split point selection as used in $RDT.\alpha()$ are generic. Examples of deterministic feature selection and deterministic split point selection can be found in C4.5 (Quinlan, 1993) or CART (Breiman et al., 1984). Since Max-diverse.$\alpha$ combines the use of deterministic and random feature selections, a subtle difference between $RDT.\alpha()$ and $RDT()$ is that continuous features are allowed to split more than once in a tree structure to cater for the normal operation of deterministic feature selection in $RDT.\alpha()$.

### 4.3 Summary

This chapter has illustrated two random tree ensemble algorithms. They are Max-diverse Ensemble and Max-diverse.$\alpha$. Max-diverse Ensemble produces complete-random trees while Max-diverse.$\alpha$ produces variable-random trees depending on the internally estimation of $\alpha$ variable. For these two algorithms, algorithmic details are given for practical implementation.

The next chapter reports the experimentations of these implementations. It provides in-depth insights into the implementation of Max-diverse.$\alpha$ which are found to be highly accurate and competitive to existing benchmarking decision tree ensembles.
Chapter 5

Experiments

“The soft overcomes the hard, The yielding overcomes the strong; Every person knows this, But no one can practice it.”

Dao De Jing - Lao Tze

The experimentation in this chapter are structured into three sets of experiments. The first set aims to determine the key components of making accurate complete-random tree ensembles. The second set aims to explore the effect of variable randomness, where complete-random trees are used as the base model and control the deterministic elements to provide variable randomness. The third set targets making the proposed algorithm practical and applicable in real life situation. For these experiments, some of the state-of-the-art decision tree ensembles are included for benchmarking. They are C5 boosting, Random Forests and Bagging.

5.1 Experimental Methodology

All experiments are conducted using 10-fold cross validation. Error rate is the basic measurement of performance. For the three sets of experiments, the number of trees in each ensemble is chosen for specific purposes. In the first set, ten thousand trees are adopted to observe the overfitting behaviour and the convergence limit of tree ensembles. It helps us to verify the claims on anti-overfitting capacity made by Breiman (2001) and to see if this capacity extends to complete-random tree ensembles. In the second set of experiments, a thousand trees are used for mid-range performance comparison. In the third set of experiments, each ensemble builds one hundred trees for a realistic comparison with other benchmark ensembles. A special case may apply to C5 Boosting for it has a special routine in stopping further iteration. So, C5 Boosting produces up to a hundred trees in an ensemble.

Data sets are obtained from UCI repository (Blake and Merz, 1998). For the first experiment set, forty data sets are used so that experiments can be completed within a reasonable time frame. For the second and third experiment sets, forty five data sets are used. The number of data sets exceeds any previous comparison on related ensembles which provides better generalisation on performance comparison. Data attributes can be found in Appendix A.

5.2 Experiment 1: The Key Components

It is sometimes difficult to determine the best combination of components unless one performs an exhaustive search with all possible combinations. In building complete-random
tree ensembles, there are a lot of different options one could take. Without thorough experimentation, we cannot assume what has been used previously must be the best setting for complete-random tree ensembles. Random Forests is one of the popular random tree ensemble implementations (Breiman, 2001). When comparing Random Forests with Random Decision Trees (a complete-random trees model) (Fan et al., 2003), one could distinguish three differences between the twos. They are different in terms of (i) ensemble methods, (ii) tree height restriction and (iii) training samples preparation. Using these differences, we are able to come up with eight variants for our first experiment to find the key components for complete-random tree ensembles.

For ensemble methods, Bauer and Kohavi (1999) have shown that Bagging using probabilistic estimates without pruning is better than using conventional votes. In theory, probability should be more superior to voting as it protects against overfitting (Freund et al., 2001). However, (Domingos, 2000) raised a concern over the Bayesian averaging trees. He suggested that overfitting occurs due to random fluctuations in data distribution using the probability averaging trees weighted based on their individual performance. The choice of using conventional voting in Random Forests also reflects a hesitation in using probability averaging. While none of the evaluations are done in the context of complete-random tree, it is worthwhile to empirically investigate the difference between conventional voting and probability averaging in the context of complete-random trees.

Regarding the similarity between conventional voting and probability averaging, Ho (1998) raised an important point regarding unpruned probability averaging trees. She found that normally built decision trees promote homogeneous growth, which means samples in leaf nodes are very likely to have the same class labels. Under this condition, probability averaging becomes conventional voting. Complete-random trees, however, do not promote homogeneous growth, so resulting leaf nodes would have a wider variety than normally built decision trees. We shall see if there is a difference between conventional voting and probability averaging in the context of complete-random tree ensembles.

The tree height restriction originated from Fan et al.’s (2003) implementation. Let \( k \) be the total number of features, setting tree height to \( \frac{k}{2} \) is called half height tree. Alternatively, unrestricted tree growth is called full height tree.

Fan et al.’s (2003) argument in choosing half height trees is based on tree diversity. Tree diversity can be estimated by the number of feature combinations. Consider a rule (which is equivalent to a branch in a tree), when selecting \( i \) features from \( k \) features, there are \( C_k^i = \binom{k}{i} \) unique feature combinations. To use only a single value of \( i \), \( i = \frac{k}{2} \) produces the largest number of combinations. Fan et al. (2003) uses this argument as the basis to choose the tree height limit of \( \frac{k}{2} \). In Fan et al.’s argument, it is implied that maximum tree height is a constant \( k \), so \( i \) is also a constant. However, in real life data where data distribution is uneven or in the cases where data size falls short of the hypothesis space \( H \), the maximum tree height varies. As higher diversity is preferred, allowing any value of \( i \) is more desirable as it gives the maximum choice of \( i \) or higher diversity. Since the total number of possible unique combinations to include any value of \( i \) is \( T(i) = \sum C_k^i \) and \( T(k) > T(\frac{k}{2}) > C_k^{\frac{k}{2}} \). We shall set the tree height limit to \( k \) which means building full height trees instead.

To sum up, when we take into the account that maximum tree height can vary in an ensemble, growing full height trees creates a much higher diversity as the analysis shown above. In contrast, Fan et al. promoted half height trees, as it generates the highest diversity assuming constant maximum tree height. We shall verify the results of this experiment to see if the choice of tree height has any impacts on the ensemble’s performance.

In addition to the choice of tree height, note that the deterministically built decision tree stops splitting at insensible split, so a normally built single decision tree does not equal
to a full height tree. Full height complete-random trees only stop splitting when training
samples run out. Therefore, a complete-random tree ensemble is not equal to Random
Forests at any setting. Also, consider ‘Random Subspace’ in Ho (1998) and ‘Random
Decision Trees’ in Fan et al. (2003), both methods similarly provide half the number of
features to a base learner. The significant difference between them is the choice of base
learner. Random Subspace used an unmodified C4.5 and Random Decision Trees used
complete-random trees. Again, since the context of this experiment is different from the
previous ones, we shall see whether tree height restriction contributes to the accuracy of
complete-random tree ensemble.

For training samples preparation, preparing bootstrap samples or Bagging is a
conventional practice in building random tree ensembles. Bagging relies on the stability of
the base classifiers to generate diversity. Since the tree construction in complete-random
tree ensembles is completely random, complete-random trees are inherently ‘unstable’ and
it generates different trees even there is no change in the data. It raises a question as to
whether bootstrap sampling is really needed on top of readily randomised algorithms. In
our experiment, we will compare algorithms built with bootstrap sampling against those
that are not, to see if bootstrap sampling is really needed in constructing complete-random
tree ensembles.

We have briefly discussed the three possible options of building complete-random tree
ensembles. They are ensemble methods, tree height restriction and training samples. The
next sub-section lays out the different variants of complete-random tree ensembles used in
the experiments.

5.2.1 Eight possible variants

There are three options in the experiments. The followings are the abbreviations used in
the experiments:

1. Ensemble methods
   (a) Voting
   (b) Probability averaging

2. Tree height restriction
   (a) Full height
   (b) Half height

3. Sample randomisation
   (a) Original training samples
   (b) Bootstrap training samples

In total, there are eight possible variants from these three options. Each variant is repre-
sented by three letters, for example “VFO” refers to a random trees ensemble with options
Voting, Full height tree and Original training samples.

5.2.2 Results

In Table 5.1, error rates of the eight variants and two benchmark classifiers are listed.
Variants are named using the convention introduced in the previous sub-section. Data sets
are sorted in alphabetic order. Bold faced error figures indicate best error rates among
classifiers listed. Mean error rates are listed at the bottom of the table for comparison
between classifiers.
CHAPTER 5. EXPERIMENTS

Figure 5.1: Compare the best variant PFO with Bagging, Random Forests.

Table 5.2 summarises comparative results in the terms of wins, losses and draws. Sign tests are performed to show the significant wins or losses in comparisons. According to a sign test table, the critical value at 5% level of significance for forty trails is twenty six provided there is zero draws. It means that the number of wins or losses have to be at least twenty six or above to be deemed as significance. Bold faced figures in the table show significant wins or losses based on sign tests at 5% level of significance.

In this sub-section, we shall first compare the eight variants with the two benchmark classifiers to see how complete-random trees perform against conventional random tree ensembles. Then, we compare the results among the three options in constructing complete-random tree ensembles. This comparison should answer some of the questions posed in the beginning of this section. After that, we highlight the differences between the best variant and the rest of the classifiers to gain a better understanding in why it performs better than others. Finally, we will look at two interesting data sets in reaching their theoretical limits.

Comparing variants with the benchmark classifiers, we summarise the results as follows:

- PFO, PFB and PHO perform comparable to Bagging,
- PFO and PFB perform comparable to Random Forests,
- PFO has the most wins against the two benchmark classifiers having 23 and 21 out of 40.
- PFO has thirteen data sets with the best error rates boldfaced in table 5.1, Random Forests has eleven and Bagging has eight.

For each of the three options, we summarise the results as follows:

- All probability averaging variants are significantly better than their voting counterparts according to the sign test, e.g. PHO vs. VHO, PHO has 30 wins, 4 losses and 6 draws.
- Full height trees perform significantly better than half height trees, e.g. PFO vs. PHO, PFO has 25 wins, 6 losses and 9 draws,
- Bootstrap sampling does significantly impair accuracy as suggested, e.g. PFO vs. PFB, PFO has 25 wins, 11 losses and 4 draws.

The results above suggest that the most accurate variant PFO is comparable to the benchmark classifiers and the probability averaging alone is the key component to generate accurate complete-random tree ensembles.

Figure 5.1 provides an overall comparison of relative performance between PFO and each of the two benchmark classifiers, Bagging and Random Forests. Each point on the figure represents one data sets in the experiment. If points are located close to the
## EXPERIMENT 1: THE KEY COMPONENTS

Table 5.1: Experiment 1 results: Complete-random tree ensembles’ average errors (%) Experiments are conducted using ten thousand trees for each ensemble. Abbreviations are referred to code names in Section 5.2.1. Best error rates are bold faced.

<table>
<thead>
<tr>
<th>Data set</th>
<th>P</th>
<th>P</th>
<th>F</th>
<th>F</th>
<th>H</th>
<th>H</th>
<th>O</th>
<th>O</th>
<th>Bagging</th>
<th>Random Forests</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>29.8</td>
<td>29.9</td>
<td>31.7</td>
<td>31.7</td>
<td>29.9</td>
<td>29.7</td>
<td>31.7</td>
<td>31.7</td>
<td><strong>29.1</strong></td>
<td><strong>29.1</strong></td>
</tr>
<tr>
<td>anneal</td>
<td><strong>0.9</strong></td>
<td>1.7</td>
<td>2.6</td>
<td>2.9</td>
<td>1.0</td>
<td>1.8</td>
<td>3.7</td>
<td>3.9</td>
<td>3.2</td>
<td>14.8</td>
</tr>
<tr>
<td>audiology</td>
<td>19.5</td>
<td>18.5</td>
<td>21.7</td>
<td>19.0</td>
<td>22.6</td>
<td>22.1</td>
<td>26.2</td>
<td>23.8</td>
<td>20.8</td>
<td>37.5</td>
</tr>
<tr>
<td>autos</td>
<td>24.3</td>
<td>22.9</td>
<td>24.4</td>
<td>22.4</td>
<td>26.7</td>
<td>22.0</td>
<td>25.3</td>
<td>22.9</td>
<td><strong>16.2</strong></td>
<td><strong>20.5</strong></td>
</tr>
<tr>
<td>balance</td>
<td>13.8</td>
<td>13.6</td>
<td>13.9</td>
<td>13.8</td>
<td>13.6</td>
<td><strong>13.4</strong></td>
<td>18.2</td>
<td>14.6</td>
<td>15.5</td>
<td>15.5</td>
</tr>
<tr>
<td>breast-w</td>
<td>2.4</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
<td>3.0</td>
<td>3.2</td>
<td>2.9</td>
<td>3.2</td>
<td>3.4</td>
<td>3.1</td>
</tr>
<tr>
<td>breast-y</td>
<td>25.5</td>
<td>25.5</td>
<td><strong>23.4</strong></td>
<td>25.5</td>
<td>24.8</td>
<td>27.6</td>
<td>25.1</td>
<td>27.0</td>
<td>26.6</td>
<td>26.9</td>
</tr>
<tr>
<td>chess</td>
<td>1.6</td>
<td>1.9</td>
<td>2.5</td>
<td>2.8</td>
<td>2.7</td>
<td>4.1</td>
<td>4.9</td>
<td>5.1</td>
<td>4.8</td>
<td>4.9</td>
</tr>
<tr>
<td>cleveland</td>
<td>41.3</td>
<td>41.9</td>
<td>43.9</td>
<td>44.3</td>
<td>42.6</td>
<td>43.6</td>
<td>43.6</td>
<td>44.2</td>
<td>43.6</td>
<td>42.2</td>
</tr>
<tr>
<td>coding</td>
<td><strong>16.3</strong></td>
<td><strong>16.3</strong></td>
<td><strong>16.3</strong></td>
<td><strong>16.3</strong></td>
<td><strong>16.3</strong></td>
<td>18.7</td>
<td>23.9</td>
<td>19.2</td>
<td>23.8</td>
<td>33.5</td>
</tr>
<tr>
<td>credit-a</td>
<td>12.3</td>
<td>12.6</td>
<td>12.9</td>
<td>13.5</td>
<td>13.3</td>
<td>15.1</td>
<td>14.1</td>
<td>14.2</td>
<td>13.2</td>
<td>13.6</td>
</tr>
<tr>
<td>credit-q</td>
<td>25.3</td>
<td>25.7</td>
<td>27.1</td>
<td>27.3</td>
<td>27.3</td>
<td>29.2</td>
<td>29.0</td>
<td>29.3</td>
<td><strong>24.7</strong></td>
<td>26.9</td>
</tr>
<tr>
<td>dna</td>
<td>28.8</td>
<td>28.9</td>
<td>28.8</td>
<td>28.8</td>
<td>16.1</td>
<td>14.5</td>
<td>16.1</td>
<td>14.4</td>
<td>7.1</td>
<td>12.9</td>
</tr>
<tr>
<td>echo</td>
<td>32.9</td>
<td>34.3</td>
<td><strong>32.1</strong></td>
<td>32.8</td>
<td>33.7</td>
<td>36.7</td>
<td>34.5</td>
<td>35.1</td>
<td>35.2</td>
<td>35.9</td>
</tr>
<tr>
<td>flare</td>
<td>18.5</td>
<td>19.0</td>
<td>17.5</td>
<td>17.4</td>
<td>18.2</td>
<td>18.3</td>
<td>17.5</td>
<td><strong>17.1</strong></td>
<td>17.5</td>
<td>17.6</td>
</tr>
<tr>
<td>glass</td>
<td>26.2</td>
<td>26.2</td>
<td>34.6</td>
<td>33.1</td>
<td>28.0</td>
<td>30.0</td>
<td>37.9</td>
<td>36.0</td>
<td>35.2</td>
<td><strong>21.9</strong></td>
</tr>
<tr>
<td>hayes</td>
<td>44.4</td>
<td>40.0</td>
<td>48.1</td>
<td>47.5</td>
<td>53.8</td>
<td>46.9</td>
<td>58.1</td>
<td>47.5</td>
<td>17.5</td>
<td><strong>17.5</strong></td>
</tr>
<tr>
<td>hepatitis</td>
<td>15.3</td>
<td>15.7</td>
<td>16.0</td>
<td>15.7</td>
<td>17.3</td>
<td><strong>15.0</strong></td>
<td>16.0</td>
<td>15.7</td>
<td>24.7</td>
<td>16.3</td>
</tr>
<tr>
<td>horse</td>
<td>18.8</td>
<td>16.9</td>
<td>19.1</td>
<td>17.2</td>
<td>19.9</td>
<td>17.2</td>
<td>20.4</td>
<td>17.4</td>
<td><strong>14.2</strong></td>
<td>16.3</td>
</tr>
<tr>
<td>hypo</td>
<td>2.2</td>
<td>2.4</td>
<td>4.7</td>
<td>4.7</td>
<td>2.3</td>
<td>2.4</td>
<td>4.7</td>
<td>4.7</td>
<td><strong>0.9</strong></td>
<td>1.3</td>
</tr>
<tr>
<td>ionosphere</td>
<td>9.4</td>
<td>9.7</td>
<td>9.4</td>
<td>9.7</td>
<td>11.7</td>
<td>15.9</td>
<td>11.7</td>
<td>15.3</td>
<td>6.8</td>
<td><strong>5.7</strong></td>
</tr>
<tr>
<td>iris</td>
<td>4.7</td>
<td>6.0</td>
<td>7.3</td>
<td>7.3</td>
<td>4.0</td>
<td>5.3</td>
<td>11.3</td>
<td>11.3</td>
<td>6.7</td>
<td>6.0</td>
</tr>
<tr>
<td>led24</td>
<td><strong>28.8</strong></td>
<td><strong>28.8</strong></td>
<td><strong>28.4</strong></td>
<td><strong>29.0</strong></td>
<td>36.8</td>
<td>37.4</td>
<td>36.5</td>
<td>36.5</td>
<td>28.5</td>
<td>29.8</td>
</tr>
<tr>
<td>liver</td>
<td>30.7</td>
<td>31.8</td>
<td>38.3</td>
<td>37.1</td>
<td>29.5</td>
<td>31.6</td>
<td>38.5</td>
<td>37.7</td>
<td><strong>28.1</strong></td>
<td>29.2</td>
</tr>
<tr>
<td>lymph</td>
<td>15.5</td>
<td>15.4</td>
<td>15.5</td>
<td><strong>14.7</strong></td>
<td>15.2</td>
<td>15.4</td>
<td>18.9</td>
<td>15.4</td>
<td>21.6</td>
<td>17.4</td>
</tr>
<tr>
<td>nursery</td>
<td><strong>2.0</strong></td>
<td>2.3</td>
<td>5.4</td>
<td>5.5</td>
<td>2.2</td>
<td>1.9</td>
<td>7.1</td>
<td>5.2</td>
<td>6.4</td>
<td>4.7</td>
</tr>
<tr>
<td>pima</td>
<td>24.7</td>
<td>23.6</td>
<td>28.6</td>
<td>28.1</td>
<td>24.3</td>
<td>25.4</td>
<td>29.4</td>
<td>29.5</td>
<td>24.7</td>
<td><strong>23.4</strong></td>
</tr>
<tr>
<td>primary</td>
<td>56.1</td>
<td>55.2</td>
<td><strong>53.1</strong></td>
<td><strong>53.1</strong></td>
<td>54.6</td>
<td>55.8</td>
<td>55.5</td>
<td>55.2</td>
<td>55.5</td>
<td><strong>53.1</strong></td>
</tr>
<tr>
<td>segment</td>
<td>2.9</td>
<td>3.1</td>
<td>4.9</td>
<td>3.3</td>
<td>3.7</td>
<td>6.6</td>
<td>6.4</td>
<td>2.4</td>
<td><strong>2.3</strong></td>
<td></td>
</tr>
<tr>
<td>sick</td>
<td>7.6</td>
<td>7.8</td>
<td>9.3</td>
<td>9.3</td>
<td>8.1</td>
<td>8.0</td>
<td>9.3</td>
<td>9.3</td>
<td>2.1</td>
<td>3.7</td>
</tr>
<tr>
<td>solar</td>
<td>30.0</td>
<td>29.7</td>
<td>27.2</td>
<td>27.5</td>
<td>29.6</td>
<td>30.6</td>
<td>29.3</td>
<td>28.8</td>
<td>27.2</td>
<td><strong>24.7</strong></td>
</tr>
<tr>
<td>sonar</td>
<td><strong>13.4</strong></td>
<td><strong>16.8</strong></td>
<td><strong>13.4</strong></td>
<td><strong>16.8</strong></td>
<td>23.5</td>
<td>26.9</td>
<td>23.5</td>
<td>26.9</td>
<td>20.1</td>
<td>19.6</td>
</tr>
<tr>
<td>soybean</td>
<td>6.0</td>
<td>5.7</td>
<td>5.7</td>
<td>5.6</td>
<td>5.6</td>
<td>6.0</td>
<td>6.0</td>
<td>5.7</td>
<td>6.2</td>
<td><strong>5.4</strong></td>
</tr>
<tr>
<td>threeOf9</td>
<td>0.2</td>
<td>0.8</td>
<td>11.3</td>
<td>11.1</td>
<td>8.2</td>
<td>2.7</td>
<td>13.1</td>
<td>12.9</td>
<td>3.3</td>
<td>2.2</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>9.4</td>
<td>10.4</td>
<td>24.6</td>
<td>23.8</td>
<td>18.8</td>
<td>27.5</td>
<td>27.5</td>
<td>26.2</td>
<td>29.4</td>
<td>26.4</td>
</tr>
<tr>
<td>vehicle</td>
<td>27.1</td>
<td>29.2</td>
<td>27.4</td>
<td>29.3</td>
<td>27.9</td>
<td>29.1</td>
<td>29.3</td>
<td>28.4</td>
<td>24.9</td>
<td><strong>25.2</strong></td>
</tr>
<tr>
<td>vote</td>
<td>5.3</td>
<td>5.3</td>
<td>5.3</td>
<td>5.3</td>
<td>5.1</td>
<td>6.2</td>
<td>4.8</td>
<td>6.0</td>
<td><strong>4.6</strong></td>
<td>4.8</td>
</tr>
<tr>
<td>wine</td>
<td>2.3</td>
<td>1.7</td>
<td>2.3</td>
<td>1.7</td>
<td>2.3</td>
<td>2.8</td>
<td>2.3</td>
<td>2.3</td>
<td>5.6</td>
<td><strong>1.1</strong></td>
</tr>
<tr>
<td>zoo</td>
<td><strong>2.0</strong></td>
<td><strong>4.0</strong></td>
<td><strong>2.0</strong></td>
<td><strong>3.0</strong></td>
<td><strong>3.0</strong></td>
<td><strong>3.0</strong></td>
<td><strong>3.0</strong></td>
<td><strong>3.0</strong></td>
<td><strong>6.9</strong></td>
<td><strong>7.9</strong></td>
</tr>
</tbody>
</table>

| mean          | **17.3** | **17.5** | **19.0** | **18.9** | **16.7** | **19.1** | **20.7** | **20.2** | **17.4** | **17.8** |
CHAPTER 5. EXPERIMENTS

Table 5.2: Experiment 1 summary:
Summary of pairwise comparison (wins, losses, draws) reading from top
to left. The number of significant wins and losses is bold faced, based on sign tests at 5% level of significance.

<table>
<thead>
<tr>
<th>Max-diverse Ensemble</th>
<th>PFO</th>
<th>PFB</th>
<th>PHO</th>
<th>PHB</th>
<th>VFO</th>
<th>VFB</th>
<th>VHO</th>
<th>VHB</th>
<th>Bagging</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Random Forests</strong></td>
<td>21,19,0</td>
<td>20,19,1</td>
<td>17,23,0</td>
<td>17,23,0</td>
<td>14,26,0</td>
<td>11,29,0</td>
<td>11,29,0</td>
<td>11,29,0</td>
<td>19,20,1</td>
</tr>
<tr>
<td><strong>Bagging</strong></td>
<td>23,17,0</td>
<td>22,18,0</td>
<td>20,20,0</td>
<td>18,22,0</td>
<td>18,22,0</td>
<td>15,25,0</td>
<td>12,28,0</td>
<td>13,27,0</td>
<td></td>
</tr>
<tr>
<td>VHB</td>
<td>30,8,2</td>
<td>29,6,5</td>
<td>25,10,5</td>
<td>29,6,5</td>
<td>29,10,1</td>
<td>22,15,3</td>
<td>14,20,6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VHO</td>
<td>32,6,2</td>
<td>33,6,1</td>
<td>30,4,6</td>
<td>30,14,5</td>
<td>27,8,5</td>
<td>21,15,4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VFB</td>
<td>28,10,2</td>
<td>27,11,2</td>
<td>19,21,0</td>
<td>19,19,2</td>
<td>26,13,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VFO</td>
<td>28,11,1</td>
<td>27,15,2</td>
<td>21,18,1</td>
<td>16,21,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHB</td>
<td>25,10,5</td>
<td>23,9,8</td>
<td>15,16,9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHO</td>
<td>25,6,9</td>
<td>25,11,4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PFB</td>
<td>25,11,4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.2: Compare the best variant PFO with related variants VFO, PHO, PFB.

diagonal line in the figure, then the classifiers are having similar performance across the board. The scatter pattern in the figure indicates that PFO performs variably according to the different data sets as compared with Bagging and Random Forests. Data sets that are located far from the diagonal lines are anneal, chess, coding, dna, hayes, tic-tac-toe and zoo.

Figure 5.2 provides a comparison of PFO with the three related variants VFO, PHO and PFB. When comparing VFO and PFO, points are located mostly on top of the diagonal line, which means that probability averaging performs generally better than voting. When comparing PHO and PFO, a similar pattern is shown which means full height trees are better than half height trees. The comparison between PFB and PFO shows that bootstrap sampling does not improve the accuracy of complete-random trees which answers a previous assumption.

We call PFO Max-diverse Ensemble as it relies on maximising diversity to lower the $PE^*$. In terms of probability averaging, resulting probability estimates are more diversified than votes. In terms of tree height, our analysis early on shows that unrestricted tree growth or building full height trees allows for more diversity. In terms of training samples, original training sets provide more samples than bootstrap sample sets, and hence allow for maximum tree growth and higher diversity. It is also interesting to note that Max-diverse Ensemble is comparable to Random Forests in terms of accuracy. This empirical results confirm that there is an alternative way to reduce generalisation error $PE^*$ through maximising diversity.

In additions, It is noteworthy that Max-diverse Ensemble performs very well with threeOf9 and waveform data. It has error rates of 0.2% and 14.1%, which are very close to the theoretical limits suggested in Blake and Merz (1998). The nearest competitor from the two benchmark classifiers achieves error rate of 2.2% and 14.7%.
5.3. EXPERIMENT 2: THE POTENTIALS OF VARIABLE RANDOMNESS

5.2.3 Overfitting and convergences

In order to investigate the problem of overfitting, Appendix B lists the error rates of Max-diverse Ensemble, Bagging and Random Forests on tested data sets. For better illustration of their performance, the figures are grouped by comparing Max-diverse Ensemble with the two benchmark classifiers. In general, none of them suffer from overfitting which dispels the concern that averaging complete-random trees causes overfitting. An example of error rate convergence can be found in Figure 5.3.

Figure 5.3: An example of error rate convergence.

Regarding error rate convergences of Max-diverse Ensemble, they generally begin with a much higher error rate. However, it catches up with the benchmark classifiers at about ten trees, for examples chess, nursery, lymph and waveform data sets. Error rates usually converge at about hundred trees and most of the data sets in Appendix B exhibit this.

5.2.4 Lesson learned

In this experiment, the most unexpected finding is that complete-random tree ensembles, without the presence of deterministic feature selection, perform comparable to Random Forests. This led us to reconsider the utility of deterministic feature selection in random tree ensembles.

When we recapitulate what we have discussed in previous chapters, we learn that deterministically generated single trees have two problems, they are the lack of smoothness and difficulty in capturing small structures as discussed in Section 2.1 of Chapter 2. On the other hand, in Section 3.2 of Chapter 3, we learn that implementing complete-randomness is not without its problems. Problems like small disjuncts and insensible splitting are aggravated in complete-random feature selection. When we examine the extremes of determinism and complete-randomness of tree construction, each extreme has its advantages and propensities. Take into account their propensities as stated, the question then presents itself of how best to harness the advantages from both ends and manages their propensities at the same time. In the next section, we shall see one possible approach that adjusts the randomness of tree construction and makes it possible to balance the effect of their advantages and propensities.

5.3 Experiment 2: The Potentials of Variable Randomness

Variable randomness provides the flexibility to better cater for individual data sets. It serves as a mechanism for adjusting the balance of strength and diversity of individual trees in decision tree construction. There are three motivations behind the following experiment:
1. To preserve the representational power from complete-randomness and yet manage it propensities.

2. Buttrey and Kobayashi (2003) conjectured that the best performing ensemble would be found on a boundary that is formed by ensembles of different strength and correlation.

3. Random Forests’ $F$ parameter as a variable random mechanism.

In Section 3.1, we discussed the representational power of complete-random tree ensembles. If giving up some of their representational power is possible to manage problems like small disjuncts and insensible splitting, one may possess a very powerful algorithm which performs well with different data sets. The challenge now is to actualise this trade off in an effective way.

Consider the conjecture of strength and correlation mentioned in Section 2.3.1 of Chapter 2, Buttrey and Kobayashi (2003) generate the boundary using algorithms of different randomness. They are ranging from highly deterministic methods (e.g. Adaboost) to highly random methods (e.g. random splitting with permutation stopping, combined with sub-sampling). One would naturally attempt to replace these algorithms with a universal one. It is not hard to see that variable randomness can vary the diversity of ensembles. The only missing puzzle remained is whether strength would correspond to the changes of diversity and follows the conjecture as stated. We shall see if this experiment would provide an answer.

Random Forests provides an $F$ parameter to control the level of randomness, but Breiman (2001) concluded that accuracy is insensitive to it. It is then a good idea to investigate with alternative variable-random mechanism to verify the claim.

### 5.3.1 Objectives

There are several objectives in these experiments. They are listed as follows:

1. Verify whether accuracy is sensitive to randomness.

2. Investigate Max-diverse.$\alpha$’s ability to generate models with different strength and diversity.

3. Examine the potential advantages and problems in Max-diverse.$\alpha$

For the first objective, we will generate error rates as a performance measure. Differing from the previous experiment, we generate an error rate for each $\alpha$ value for each data sets. We sample twenty one $\alpha$ values from 0 to 1 in steps of 0.05, that is $\alpha = 0, 0.05, \ldots , 1$. Furthermore, visualisation of error rates against $\alpha$ values would help to determine the relationship between them.

For the second objective, we implemented the strength and correlation measures as described in (Breiman, 2001). The corrected version of these measures can be found in (Kobayashi, 2002). These strength and correlation measures will be plotted against each other and we can visually verify whether they conform to the conjecture.

For the third objective, we analyse the results established by the first two to see if they provide any insights.

### 5.3.2 Results

The results show that accuracy is sensitive to $\alpha$. Using the Gaussian mixture for decision boundary visualisation, we find that Max-diverse.$\alpha$ does trade off the representational power to manage the problem of small disjuncts.
5.3. EXPERIMENT 2: THE POTENTIALS OF VARIABLE RANDOMNESS

Figure 5.4: Classify Gaussian mixture using Max-diverse.α with different α values. Changes in decision boundary rectangularity can be observed from highly random (α = 0.1) to highly deterministic (α = 0.9).

![Classify Gaussian mixture using Max-diverse.α with different α values.](image)

To illustrate the effect of α, Figure 5.4 demonstrates changes in decision boundary rectangularity and small disjuncts when α changes. When α is between 0.3 and 0.5 in this example, error rates drop below 10%.

Figure 5.5: Classifying Gaussian mixture data set.

When plotting the error rates against different α values, they form a surface as shown in Figure 5.5. Interestingly, we find a range of α values that achieve lower error rates than those generated for C4.5, Random Forests and Max-diverse Ensemble.

When surveying the error rates using different α values among the forty five data sets, eighteen of them form a convex ‘U’ shape, twenty five form a ‘slide’ shape and two form an unusual irregular shape. All data sets in ‘slide’ shape slide from α = 1 to α = 0, which means complete-random setting usually provides better accuracy. Examples for ‘U’ shape and ‘slide’ shape error rates can be found in Figure 5.6 and 5.7. Left diagrams are
When examine datasets with irregular shape error rate, we find that errors variate within two percentage points which is trivial. At this stage, no generalisable characteristic can be found with this kind of data sets. In this experiment, they are echo and flare data sets.

To investigate the formation of ‘U’ shape error rate, we turn to our early investigation of problems with randomisation techniques. In Section 3.2, insensible splitting and small disjuncts are problems of complete-randomness. Logically, the problem of insensible splitting diminishes with the increase of $\alpha$ as deterministic feature selection restricts insensible growth. Similarly, the problem of small disjuncts exhibits the same property in Figure 5.4 and 5.5, where the increase of $\alpha$ lessens the effect of small disjuncts. We conjecture that the reason why the error rate reduces in the initial increase of $\alpha$ is that the effects of insensible split and small disjuncts are being shrunk. That explains half of the observations of ‘U’ shape error rates.

Another half of the ‘U’ shape error rates can be explained by the slide shape observations. The slide shape error rates represent most of the data set behaviour with Max-diverse.$\alpha$. As $\alpha$ approaches to 0, the representational power is maximised and lowers the error rate. Logically, ideal data sets that do not contain any irrelevant attributes or
5.3. EXPERIMENT 2: THE POTENTIALS OF VARIABLE RANDOMNESS

Figure 5.8: The formation of ‘U’ shape error rates on $\alpha$.

outliers should be directly benefited from $\alpha$ being 0. When $\alpha = 0$, error rate should be minimised. That explain the second half of the ‘U’ shape error rates.

When we overlay the error caused by small disjuncts and insensible split together with the ‘slide’ shape error rate of varying $\alpha$, Figure 5.8 illustrates the formation of ‘U’ shape error rates as the result of these effects. We can see that the lowest point of error is where the effects are balanced.

Potentially, if we can pick the best $\alpha$ prior to training a model, this algorithm would be very strong. In a hypothetical comparison, if we can use the best error rates of Max-diverse.\(\alpha\) as shown in Appendix C, Max-diverse.\(\alpha\) would be significantly better than Max-diverse Ensemble, Random Forests and Bagging. In this light, Max-diverse.\(\alpha\) could be a very strong algorithm as long as the estimation of $\alpha$ is limited to using training samples.

For this experiment, we also plot strength of individual trees $s$ against correlation $\bar{\rho}$ for each data sets. They are listed in Appendix D. To sum up, thirty out of forty five data sets conform to Buttrey and Kobayashis’ conjecture, only fifteen of the data sets do not. Figure 5.9 shows one of the data set that conforms to the conjecture. As the majority of the data sets used conform to the conjecture, we are confident that Max-diverse.\(\alpha\) is an effective tool to survey the boundary of strength and correlation. But we are careful not to rule out any possible future improvement on Max-diverse.\(\alpha\) in this capacity.

Figure 5.9: An example of strength and correlation plot.
5.3.3 Lesson learned

$\alpha$ values form a predictable surface on error rates. Visual evidence shows that small disjuncts are mitigated as $\alpha$ increases. In a potential comparison using the best of Max-diverse.$\alpha$ with benchmarking classifiers, Max-diverse.$\alpha$ could be very strong provided an appropriate estimation of $\alpha$ values is in place. In contrast to Breiman’s (2001) conclusion, the results indicate that accuracy is sensitive to the varying degree of randomness. The remaining task is to estimate $\alpha$ values that can exploit this accuracy sensitivity prior to the evaluation of the model. Next experiment reports one of the ways to estimate an effective $\alpha$ values based purely on training errors.

5.4 Experiment 3: An Estimation of Effective Randomness

Picking an effective $\alpha$ value prior to applying the model is essential for practical applications. The effective range of $\alpha$ values is data dependent. We observe from the last experiment that most of the optimal $\alpha$ values were found in the range of $0 \leq \alpha \leq 0.5$. At this stage, there is no optimal way to estimate this $\alpha$ value using any data characteristics from training data. However, training errors provide an avenue to forecast the predictive performance. The pseudo code for estimating an effective alpha using progressive training errors is proposed as in Algorithm 5 of Chapter 4.

Our aim in this final experiment is to compare realistically how well Max-diverse.$\alpha$ performs in constrained resources, that is, with only hundred trees in an ensemble. Also, how well does it compare with industrial strength applications, for example, the state-of-the-art C5 Boosting. The comparison is done with the experimental methodology as stated in Section 5.1.

5.4.1 Results

Table 5.3 lists resulting error rates of four random tree ensemble algorithms including Max-diverse.$\alpha$, Max-diverse Ensemble, Random Forests and C5 boosting. Table 5.4 compares the four methods in terms of wins, losses and draws. The result shows the followings:

- Compared to Max-diverse Ensemble, Max-diverse.$\alpha$ wins in thirty two data sets, loses in twelve data sets and draws in one data set. This is significant in a sign test at 5% level of significance.

- Compared to Random Forests, Max-diverse.$\alpha$ wins in twenty six data sets, loses in fourteen data sets and draws in five data sets. This is also significant in a sign test at 5% level of significance.

- Compared to C5 Boosting, Max-diverse.$\alpha$ wins in twenty one data sets, loses in twenty four data sets and draws in none. C5 Boosting and Max-diverse.$\alpha$ are comparable to each other.

Furthermore, of all the data sets, Max-diverse.$\alpha$ has the lowest errors in twelve, C5 Boosting has seventeen, Max-diverse Ensemble has seven and Random Forests has six. Max-diverse.$\alpha$ also has the lowest mean error rate of 15.6% which is very similar to C5 boosting of 15.9%. Note that the Max-diverse.$\alpha$ improves over Max-diverse Ensemble specifically in led24 and waveform40 in which there are many irrelevant attributes. Max-diverse.$\alpha$ brings down both errors by roughly two percentage points to a level similar to other contemporary ensembles.

Ideally, Max-diverse.$\alpha$ should do no worse than Max-diverse Ensemble since Max-diverse Ensemble is equivalent to Max-diverse.$\alpha$ when $\alpha = 0$. In the experiment, we found
Table 5.3: Experiment 3 results: Max-diverse.α with αe estimation average errors (%)
Experiments are conducted using one hundred trees for each ensemble. Best error rates are bold faced.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Max-diverse.α</th>
<th>C5 Boosting</th>
<th>Max-diverse Ensemble</th>
<th>Random Forests</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>30.5</td>
<td>31.1</td>
<td>30.2</td>
<td>29.5</td>
</tr>
<tr>
<td>anneal</td>
<td>1.1</td>
<td>5.0</td>
<td>1.4</td>
<td>23.8</td>
</tr>
<tr>
<td>audiology</td>
<td>15.8</td>
<td>15.0</td>
<td>17.7</td>
<td>33.7</td>
</tr>
<tr>
<td>auto</td>
<td>15.7</td>
<td>15.6</td>
<td>22.5</td>
<td>19.0</td>
</tr>
<tr>
<td>balance</td>
<td>15.7</td>
<td>18.9</td>
<td>12.3</td>
<td>19.7</td>
</tr>
<tr>
<td>breast-w</td>
<td>3.7</td>
<td>3.1</td>
<td>2.4</td>
<td>3.4</td>
</tr>
<tr>
<td>breast-y</td>
<td>25.5</td>
<td>26.9</td>
<td>25.9</td>
<td>28.6</td>
</tr>
<tr>
<td>chess</td>
<td>0.5</td>
<td>0.3</td>
<td>1.6</td>
<td>0.9</td>
</tr>
<tr>
<td>cleveland</td>
<td>42.9</td>
<td>41.6</td>
<td>41.6</td>
<td>39.6</td>
</tr>
<tr>
<td>coding</td>
<td>16.5</td>
<td>15.4</td>
<td>16.8</td>
<td>17.7</td>
</tr>
<tr>
<td>credit-a</td>
<td>12.6</td>
<td>14.3</td>
<td>13.0</td>
<td>14.5</td>
</tr>
<tr>
<td>credit-q</td>
<td>23.5</td>
<td>22.4</td>
<td>25.7</td>
<td>24.3</td>
</tr>
<tr>
<td>dna</td>
<td>5.1</td>
<td>4.8</td>
<td>26.5</td>
<td>3.7</td>
</tr>
<tr>
<td>echo</td>
<td>34.9</td>
<td>37.4</td>
<td>34.2</td>
<td>34.3</td>
</tr>
<tr>
<td>flare</td>
<td>18.9</td>
<td>17.5</td>
<td>19.2</td>
<td>18.3</td>
</tr>
<tr>
<td>glass</td>
<td>22.8</td>
<td>21.4</td>
<td>22.9</td>
<td>25.3</td>
</tr>
<tr>
<td>hayes</td>
<td>18.1</td>
<td>16.9</td>
<td>21.9</td>
<td>14.4</td>
</tr>
<tr>
<td>hepatitis</td>
<td>20.0</td>
<td>14.1</td>
<td>15.5</td>
<td>16.7</td>
</tr>
<tr>
<td>horse</td>
<td>13.6</td>
<td>22.5</td>
<td>17.9</td>
<td>15.2</td>
</tr>
<tr>
<td>hypno</td>
<td>1.1</td>
<td>0.8</td>
<td>1.7</td>
<td>0.8</td>
</tr>
<tr>
<td>ionosphere</td>
<td>5.7</td>
<td>5.4</td>
<td>8.5</td>
<td>6.3</td>
</tr>
<tr>
<td>iris</td>
<td>5.3</td>
<td>4.0</td>
<td>4.7</td>
<td>3.3</td>
</tr>
<tr>
<td>labor</td>
<td>5.0</td>
<td>15.7</td>
<td>3.3</td>
<td>14.0</td>
</tr>
<tr>
<td>led24</td>
<td>28.3</td>
<td>27.8</td>
<td>30.3</td>
<td>28.3</td>
</tr>
<tr>
<td>led7</td>
<td>26.6</td>
<td>28.1</td>
<td>26.9</td>
<td>26.7</td>
</tr>
<tr>
<td>liver</td>
<td>25.8</td>
<td>29.6</td>
<td>27.9</td>
<td>32.5</td>
</tr>
<tr>
<td>lymph</td>
<td>15.0</td>
<td>19.1</td>
<td>14.3</td>
<td>18.2</td>
</tr>
<tr>
<td>nursery</td>
<td>0.7</td>
<td>0.9</td>
<td>2.2</td>
<td>1.4</td>
</tr>
<tr>
<td>pima</td>
<td>24.3</td>
<td>25.0</td>
<td>24.6</td>
<td>23.4</td>
</tr>
<tr>
<td>post</td>
<td>37.8</td>
<td>30.0</td>
<td>36.7</td>
<td>43.3</td>
</tr>
<tr>
<td>primary</td>
<td>56.3</td>
<td>56.9</td>
<td>57.2</td>
<td>55.2</td>
</tr>
<tr>
<td>satimage</td>
<td>8.5</td>
<td>8.1</td>
<td>10.4</td>
<td>8.3</td>
</tr>
<tr>
<td>segment</td>
<td>1.6</td>
<td>1.8</td>
<td>3.1</td>
<td>2.3</td>
</tr>
<tr>
<td>sick</td>
<td>2.3</td>
<td>2.2</td>
<td>5.7</td>
<td>2.3</td>
</tr>
<tr>
<td>solar</td>
<td>30.0</td>
<td>25.7</td>
<td>30.3</td>
<td>28.8</td>
</tr>
<tr>
<td>sonar</td>
<td>15.4</td>
<td>15.9</td>
<td>15.9</td>
<td>18.7</td>
</tr>
<tr>
<td>soybean</td>
<td>5.4</td>
<td>6.2</td>
<td>6.0</td>
<td>11.7</td>
</tr>
<tr>
<td>threeOf9</td>
<td>0.2</td>
<td>0.0</td>
<td>0.6</td>
<td>1.2</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>2.1</td>
<td>1.2</td>
<td>9.7</td>
<td>6.3</td>
</tr>
<tr>
<td>vehicle</td>
<td>24.2</td>
<td>23.3</td>
<td>27.1</td>
<td>24.2</td>
</tr>
<tr>
<td>vote</td>
<td>4.4</td>
<td>4.5</td>
<td>5.3</td>
<td>4.4</td>
</tr>
<tr>
<td>waveform21</td>
<td>15.2</td>
<td>15.6</td>
<td>14.7</td>
<td>15.7</td>
</tr>
<tr>
<td>waveform40</td>
<td>15.8</td>
<td>15.1</td>
<td>17.0</td>
<td>16.0</td>
</tr>
<tr>
<td>wine</td>
<td>4.0</td>
<td>5.6</td>
<td>1.1</td>
<td>1.7</td>
</tr>
<tr>
<td>zoo</td>
<td>2.0</td>
<td>3.0</td>
<td>2.0</td>
<td>2.9</td>
</tr>
<tr>
<td>mean</td>
<td>15.6</td>
<td>15.9</td>
<td>16.8</td>
<td>17.4</td>
</tr>
</tbody>
</table>

Table 5.4: Experiment 3 summary:
Comparison of four ensemble methods in wins, losses and draws. Scores are read from top to left. Bold faced are significant scores using sign tests at 5% level of significance.

<table>
<thead>
<tr>
<th>wins, losses, draws</th>
<th>Max-diverse.α</th>
<th>C5 Boosting</th>
<th>Random Forests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max-diverse Ensemble</td>
<td>32, 12, 1</td>
<td>26, 17, 2</td>
<td>24, 21, 0</td>
</tr>
<tr>
<td>Random Forests</td>
<td>26, 14, 5</td>
<td>30, 14, 1</td>
<td></td>
</tr>
<tr>
<td>C5 Boosting</td>
<td>21, 24, 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
that Max-diverse.α sometimes chose an α value that performs worse than Max-diverse Ensemble. This happened in a few data sets such as abalone, balance and breast-w.

In addition, Max-diverse.α takes about eleven times longer to run than Max-diverse Ensemble. This is because the existing estimation method assesses eleven different α values for training error evaluation. These issues indicate a need to improve the α_e estimation process.

One interesting finding in Table 5.4 is that, contrary to Breiman’s (2001) claim, our experiment shows that boosting is significantly better than Random Forests using a sign test at 5% level of significance. In our experiment, we employ forty five data sets, two times more than the nineteen data sets used in Breiman’s experiment. This may be attributable to the fact that we used C5 boosting (the state-of-the-art boosting algorithm) instead of Adaboost (Freund and Schapire, 1996). The increased size of the data set provides better confidence for this finding.

Following up the results Table 5.4, we will discuss the similarities and differences between Random Forests and Max-diverse.α in order to discern why they performed differently despite of the fact that they appear to be very similar. The most obvious difference between Random Forests and Max-diverse.α is the way they randomise the feature selection. Random Forests uses an F parameter to vary the degree of randomness, whereas Max-diverse.α uses α for the same purpose. Other differences such as combinatorial function and sampling technique are already discussed in Section 5.2 and they are not included in this section.

Concerning Random Forests, Breiman (Breiman, 2001) found that accuracy is not sensitive to the setting of the F parameter. On the other hand, our analysis in Section 5.3.2 shows clearly that varying the degree of randomness (using α) has a significant impact on the performance of the ensemble. Hence, it is important to identify the different behaviours resulting from these two similar but different randomisation processes.

In terms of similarities, Random Forests and Max-diverse.α both use deterministic and random feature selections in the tree induction process. However, the ways they apply these feature selections are different from each other. Random Forests applies both of these feature selections in each node where F is the number of features to be randomly selected. A deterministic feature selection is applied after the random selection on selected features. The process resembles a card game where one receives a few shuffled cards and then tries to find the best card among them. As long as the best card is received in the first place, the final selected card is inevitably the ‘best’ card after all. In the same way, no matter what F is, as long as the best feature is selected in the first place, Random Forests will inevitably select the best feature. Hence, its randomness is limited. This helps to explain why accuracy is insensitive to Random Forests’ F parameter.

## 5.5 Summary

In this chapter, three important experiments in exploring the utility of randomness are reported. The first experiment aims at finding the key components of complete-random tree ensembles. In building complete-random tree ensembles, it suggests that using simply the full height trees trained on unmodified training samples and combining via probability averaging are the best settings for complete-random tree ensembles. Through observations, complete-random tree ensembles generally do not overfit which dispels the concern of potential deficiency.

The second experiment aims at exploring the impact of variable randomness on ensembles’ performance. Visual evidence shows that representational power of random tree ensembles changes with the degrees of randomness. By plotting the corresponded strength
and correlation measures, it suggests that variable randomness can be served as a mechanism to survey the boundary of strength and correlation. The experimental results show that (a) accuracy is sensitive to different degrees of randomness, (b) better accuracy can be found in certain range of randomness. As a result, it leads to an idea of estimating appropriate randomness prior to evaluation.

The last experiment reports the results of Max-diverse.α using αe estimation procedure. The results show that the proposed algorithm Max-diverse.α performs comparably to the state-of-the-art C5 boosting and performs significantly better than Random Forests.

Through these experiments, the utility of randomness in decision tree ensembles are extended from simply providing diversity to surveying the appropriate strength and correlation for random tree ensembles. Using variable randomness, we are able to lift the accuracy of random tree ensembles to a new height and now it is in direct competition with the state-of-the-art boosting algorithm, C5 boosting.
Chapter 6

Conclusion

The first goal of this thesis is to investigate the utility of randomness in decision tree ensembles. This thesis examines the full spectrum of randomness from completely random to completely deterministic in order to elicit the significance of variable randomness in achieving better predictive accuracy.

The second goal is to design a better random tree algorithm. This thesis studies a complete-random model and proposes a novel variable-random model for building decision tree ensembles. This variable-random model is capable of finding and applying the appropriate randomness for individual data sets.

At present, the use of randomness in decision tree ensemble was merely a way to increase arbitrary amounts of diversity. In this thesis, we have seen an extended use of randomness in surveying the appropriate level of strength and diversity. This extension provides an avenue to further improve ensembles’ accuracy. It also allows us to see clearly and empirically the relationship between randomness, strength of individual trees, diversity and accuracy of random tree ensembles.

In this chapter, Section 6.1 summarise the contributions of this thesis and concludes this research and Section 6.2 discusses the prospects for future research.

6.1 Conclusions

This thesis is an analytical and empirical exploration in which the intricate idea of randomness in decision tree ensembles is explored. In response to the investigation of randomness, the contributions of this thesis are summarised as follows:

- The taxonomy of randomisation techniques provides a clear distinction of randomisation used so far. The thesis concentrates on the partial-random and complete-random test selection (the second category of the taxonomy).

- By analysing the benefits and problems of randomisations, we gain a better understanding of the different effects of randomisations.

- We visualise and examine the decision boundaries and find that complete-random tree ensembles have a distinctively high representational power to model non-axis-parallel decision boundary and hard Boolean function. Probability Imbrication is a way to explain how random trees increase the representational power of the ensemble by forming small decision segments. The representational power is capitalised in the implementation of Max-diverse Ensemble.

- Using Max-diverse Ensemble, it empirically confirms Breiman’s analysis of strength and correlation in that maximising diversity is an alternative to increasing strength
in lowering generalisation errors. It relaxes the requirements to maintaining strength in random tree ensemble and allows alternative ways to achieve higher accuracy.

- Experimental results show that Max-diverse Ensemble is not subject to overfitting. Max-diverse Ensemble behaves similarly to Random Forests which dispels a concern of overfitting with the use of probability averaging.

For designing a better random tree algorithm, the proposed algorithm has the following features:

- Most importantly, the proposed variable-random tree ensemble algorithm Max-diverse.\(\alpha\) performs significantly better than Random Forests, improves from its predecessor Max-diverse Ensemble and performs comparably to the state-of-the-art C5 Boosting.

- Max-diverse.\(\alpha\) is an effective mechanism that surveys possible combinations of strength and diversity in order to adapt to individual data sets.

- The experimental results of Max-diverse.\(\alpha\) confirm that accuracy is sensitive to different degrees of randomness as opposite to Random Forests’ insensitivity.

- The internal estimation in Max-diverse.\(\alpha\) is able to find an effective \(\alpha\) setting for individual data sets based purely on training errors.

- Max-diverse.\(\alpha\) is simple and easy to implement because it only utilises one randomisation technique to provide variable randomness. In contrast, the traditional random tree ensembles often implement multiple randomisation techniques which made them harder to implement.

\section{6.2 Future Research}

For the future works of this thesis, efficiency of Max-diverse.\(\alpha\) will be crucial to the development of random tree ensembles. An obvious extension of this research is to find a more efficient method to estimate \(\alpha\). If randomness can be progressively adjusted to the appropriate level while the ensemble is growing, such a mechanism would be significant as it would greatly reduce the time complexity of the estimation procedure.

In a bigger scope, Random tree ensembles may be used for regression problems. In the future, it is possible to extend this research to regression problems.

Again, the scope of this investigation is limited to decision tree ensembles. The research can possibly be extended to investigate the utility of randomness in other classifier-based ensembles as well. For examples, neural networks, support vector machines and the nearest neighbour algorithms.
Appendix A

Data attributes

Forty five data sets are used in this thesis. They are selected from the UCI repository (Blake and Merz, 1998). Data attributes including number of instances (size), number of attributes (att.), number of class labels and a description are listed.
### Table A.1: Data attributes for data used in the experiments

<table>
<thead>
<tr>
<th>datasets</th>
<th>size</th>
<th>#att.</th>
<th>#class</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>4177</td>
<td>8</td>
<td>2</td>
<td>Abalone growth</td>
</tr>
<tr>
<td>anneal</td>
<td>898</td>
<td>38</td>
<td>6</td>
<td>Steel annealing</td>
</tr>
<tr>
<td>audiology</td>
<td>226</td>
<td>69</td>
<td>23</td>
<td>Standardised Audiology Database</td>
</tr>
<tr>
<td>auto</td>
<td>205</td>
<td>25</td>
<td>7</td>
<td>1985 Auto Imports Database</td>
</tr>
<tr>
<td>balance</td>
<td>625</td>
<td>4</td>
<td>3</td>
<td>Balance Scale weight and Distance Database</td>
</tr>
<tr>
<td>breast-w</td>
<td>699</td>
<td>10</td>
<td>2</td>
<td>Wisconsin breast cancer database</td>
</tr>
<tr>
<td>breast-y</td>
<td>286</td>
<td>9</td>
<td>2</td>
<td>Breast cancer from Institute of Oncology, Ljubljana, Yugoslavia</td>
</tr>
<tr>
<td>chess</td>
<td>3196</td>
<td>35</td>
<td>2</td>
<td>Chess end games</td>
</tr>
<tr>
<td>cleveland</td>
<td>303</td>
<td>13</td>
<td>5</td>
<td>Heart disease database from Cleveland Clinic Foundation</td>
</tr>
<tr>
<td>coding</td>
<td>20000</td>
<td>15</td>
<td>2</td>
<td>Coding database</td>
</tr>
<tr>
<td>credit-a</td>
<td>690</td>
<td>13</td>
<td>2</td>
<td>Australian Credit database</td>
</tr>
<tr>
<td>credit-g</td>
<td>1000</td>
<td>24</td>
<td>2</td>
<td>German credit database</td>
</tr>
<tr>
<td>dna</td>
<td>3186</td>
<td>60</td>
<td>3</td>
<td>Primate splice-junction gene sequences</td>
</tr>
<tr>
<td>echo</td>
<td>133</td>
<td>7</td>
<td>2</td>
<td>Echocardiogram data</td>
</tr>
<tr>
<td>flare</td>
<td>1066</td>
<td>10</td>
<td>2</td>
<td>Predicting solar flare</td>
</tr>
<tr>
<td>glass</td>
<td>214</td>
<td>9</td>
<td>7</td>
<td>Glass identification database</td>
</tr>
<tr>
<td>hayes</td>
<td>160</td>
<td>4</td>
<td>3</td>
<td>Hayes-Roth &amp; Hayes-Roth database</td>
</tr>
<tr>
<td>hepatitis</td>
<td>155</td>
<td>19</td>
<td>2</td>
<td>Hepatitis Domain</td>
</tr>
<tr>
<td>horse</td>
<td>368</td>
<td>22</td>
<td>2</td>
<td>Horse colic database</td>
</tr>
<tr>
<td>hypo</td>
<td>3163</td>
<td>25</td>
<td>2</td>
<td>Thyroid disease database</td>
</tr>
<tr>
<td>ionosphere</td>
<td>351</td>
<td>34</td>
<td>2</td>
<td>Radar returns from the ionosphere</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>Iris plants database</td>
</tr>
<tr>
<td>labor</td>
<td>57</td>
<td>16</td>
<td>2</td>
<td>Final settlements in labour negotiations</td>
</tr>
<tr>
<td>led24</td>
<td>3200</td>
<td>24</td>
<td>10</td>
<td>LED display + 17 irrelevant attributes</td>
</tr>
<tr>
<td>led7</td>
<td>3200</td>
<td>7</td>
<td>10</td>
<td>LED display with no irrelevant attribute</td>
</tr>
<tr>
<td>liver</td>
<td>345</td>
<td>6</td>
<td>2</td>
<td>BUPA liver disorder</td>
</tr>
<tr>
<td>lymph</td>
<td>148</td>
<td>18</td>
<td>4</td>
<td>Lymphography domain</td>
</tr>
<tr>
<td>nursery</td>
<td>12960</td>
<td>8</td>
<td>5</td>
<td>Nursery database</td>
</tr>
<tr>
<td>pima</td>
<td>768</td>
<td>8</td>
<td>2</td>
<td>Diabetes of female Pima Indians</td>
</tr>
<tr>
<td>post</td>
<td>90</td>
<td>8</td>
<td>3</td>
<td>Postoperative patient data</td>
</tr>
<tr>
<td>primary</td>
<td>339</td>
<td>17</td>
<td>22</td>
<td>Primary tumor domain</td>
</tr>
<tr>
<td>satimage</td>
<td>6435</td>
<td>36</td>
<td>7</td>
<td>Satellite image data set from NASA</td>
</tr>
<tr>
<td>segment</td>
<td>2310</td>
<td>19</td>
<td>7</td>
<td>Image segmentation data</td>
</tr>
<tr>
<td>sick</td>
<td>3163</td>
<td>25</td>
<td>2</td>
<td>Sick-euthyroid data</td>
</tr>
<tr>
<td>solar</td>
<td>323</td>
<td>12</td>
<td>6</td>
<td>Solar data set</td>
</tr>
<tr>
<td>sonar</td>
<td>208</td>
<td>60</td>
<td>2</td>
<td>Classification of sonar signals</td>
</tr>
<tr>
<td>soybean</td>
<td>683</td>
<td>35</td>
<td>19</td>
<td>Soy bean disease diagnosis</td>
</tr>
<tr>
<td>threeOf9</td>
<td>512</td>
<td>9</td>
<td>2</td>
<td>The concept of three of nine</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>958</td>
<td>9</td>
<td>2</td>
<td>Tic-Tac-Toe board configurations</td>
</tr>
<tr>
<td>vehicle</td>
<td>846</td>
<td>18</td>
<td>4</td>
<td>Vehicle silhouette data set</td>
</tr>
<tr>
<td>vote</td>
<td>435</td>
<td>16</td>
<td>2</td>
<td>Votes for U.S. House of Representatives Congressmen</td>
</tr>
<tr>
<td>waveform21</td>
<td>5000</td>
<td>21</td>
<td>3</td>
<td>Waveform data</td>
</tr>
<tr>
<td>waveform40</td>
<td>5000</td>
<td>40</td>
<td>3</td>
<td>Waveform data with 19 noise attributes</td>
</tr>
<tr>
<td>wine</td>
<td>178</td>
<td>13</td>
<td>3</td>
<td>Wine recognition data</td>
</tr>
<tr>
<td>zoo</td>
<td>101</td>
<td>16</td>
<td>7</td>
<td>Zoo database</td>
</tr>
</tbody>
</table>
Appendix B

Covergences of Max-diverse Ensemble

We group the following diagrams into three groups:

1. The data sets in which Max-diverse Ensemble distinctively performs better than Bagging and Random Forests.
2. The data sets in which Bagging and Random Forests perform well with.
3. The data sets in which all methods have a similar performance.

Figure B.1: The data sets in which Max-diverse Ensemble distinctively performs better than Bagging and Random Forests (Part 1).
Figure B.2: The data sets in which Max-diverse Ensemble distinctively performs better than Bagging and Random Forests (Part 2).
Figure B.3: The data sets in which Bagging and Random Forests perform well with.
Figure B.4: The data sets in which all methods have a similar performance (Part 1).
Figure B.5: The data sets in which all methods have a similar performance (Part 2).
Appendix C

Error surfaces of Max-diverse. $\alpha$

We group the following diagrams into three groups:

2. Slide shape - where error rates form a slide or imbalance convex curve.
3. Irregular shape - rare but possible, where error rates do not form a smooth surface.

Left diagrams are error rate samples using different values of $\alpha$ factor at one thousand trees. Right diagrams are the three dimensional surfaces in terms of number of trees, $\alpha$ factors and error rates.

Figure C.1: Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 1).
Figure C.2: Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 2).
Figure C.3: Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 3).
Figure C.4: Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 4).
Figure C.5: Error rates of different $\alpha$ values and three dimensional error surfaces (Convex part 5).
Figure C.6: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 1).
Figure C.7: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 2).
Figure C.8: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 3).
Figure C.9: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 4).
Figure C.10: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 5).
Figure C.11: Error rates of different $\alpha$ values and three dimensional error surfaces.

Dataset: threeOf9

Dataset: waveform 21

Dataset: wave40

Dataset: wine
Figure C.12: Error rates of different $\alpha$ values and three dimensional error surfaces (Slide shape part 7).

Figure C.13: Error rates of different $\alpha$ values and three dimensional error surfaces (Irregular shape).
Appendix D

Strength and Correlation of Max-diverse.\(\alpha\)

The following diagrams are group by whether they conform to the Buttrey and Kobayashis’s (2003) conjecture or not. We show that Max-diverse.\(\alpha\) forms boundaries as conjectured and include the strength and correlation of Random Forests and Bagging for comparison.
Figure D.1: Strength and correlation plots of data sets that conform to the conjecture (part 1).
Figure D.2: Strength and correlation plots of data sets that conform to the conjecture (part 2).
Figure D.3: Strength and correlation plots of data sets that conform to the conjecture (part 3).
Figure D.4: Strength and correlation plots of data sets that conform to the conjecture (part 4).
Figure D.5: Strength and correlation plots of data sets that do not conform to the conjecture (part 1).
Figure D.6: Strength and correlation plots of data sets that **DO NOT** conform to the conjecture (part 2).
Vita

Publications arising from this thesis include:


Permanent Address: Gippsland School of Computing and Information Technology
Monash University
Australia

This thesis was typeset with $\LaTeX$ by the author.

---

$\LaTeX$ is an extension of $\TeX$. $\TeX$ is a collection of macros for $\TeX$. $\TeX$ is a trademark of the American Mathematical Society. The macros used in formatting this thesis were written by Glenn Maughan and modified by Dean Thompson and David Squire of Monash University.
References


REFERENCES


