DIVERSITY, MARGINS AND NON-STATIONARY LEARNING

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2012

By
Richard Stapenhurst
School of Computer Science

Contents

Word Count: 34041

List of Tables

List of Figures

Abstract

Diversity, Margins and Non-Stationary Learning
Richard Stapenhurst
A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy, 2012

Ensemble methods are frequently applied to classification problems, and generally improve upon the performance of individual models. *Diversity* is considered to be an important factor in this performance improvement; in the literature there is strong support for the idea that *high diversity* is crucial in ensembles. *Voting margins* provide an alternative explanation of the behaviour of ensembles; they have been prominently used in the interpretation of the Adaboost algorithm, and the literature suggests that *large margins* are beneficial. In this thesis, we examine these two quantities — which in both cases the literature suggests should be *increased* — and show that (in 2-class problems) they are *inversely* related, high diversity corresponding to small absolute margins. From this it can be seen that the views expressed in the literature are contradictory; we argue that ensemble behaviour can be sufficiently understood without the need to quantify 'diversity'.

However, in non-stationary learning scenarios — where we must process data that is not independent and identically distributed — the model must not only generalise well, but also *adapt* to changes in the distribution. Building on the work of Minku, we hypothesise that high diversity might be of special significance in such problems in determining the rate at which the model can adapt. We use the correspondence between diversity and margins to formulate the reasoning behind this intuition formally, and then derive an algorithm that explicitly manages diversity in order to test this hypothesis. An empirical investigation shows that

managing diversity *can*, under certain conditions, improve the ability of an ensemble to adapt to a new concept; however, it typically seems that other aspects of the learning algorithm, especially concept change detection, have a substantially larger impact on performance than diversity does.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487`), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see `http://www.manchester.ac.uk/library/aboutus/regulations`) and in The University's policy on presentation of Theses

Acknowledgements

I would like to thank my supervisor, Gavin, whose guidance was invaluable in producing this thesis. Academic gratitude also goes to the MLO group at Manchester University School of Computer Science, especially to Adam, Joe, Peter, Richard and Michalis, for their company, support and inspiration, and to my two thesis examiners, Josh and Lucy, for interesting discussion, comments and suggestions.

I thank my parents, Pat and Tim, for their invaluable financial contributions, and for only occasionally asking questions like "Have you started writing your thesis yet?" and "When are you going to get a real job?". I am grateful to my girlfriend, Jenny, for her patience and understanding, and for introducing me to BBC Radio 4.

I am also indebted to Humboldt University, who created this amazing game.

Chapter 1

Introduction

In this thesis, we present a number of contributions to the machine learning subfield of *ensemble methods*. As an introduction, we first give an overview of the problem domains that we consider (Section 1.1). Next, we describe the motivation for studying these domains (Section 1.2) and propose several contributions (Section 1.3). In Section 1.4 we outline the structure of the thesis, and finally present notation tables for reference (Section 1.5).

1.1 Context

We present an overview of three areas of interest for the thesis: supervised learning, ensemble methods and non-stationary learning.

1.1.1 Supervised Learning

One advantage conveyed by the ubiquity of computer systems is the ability to replace human labour with autonomous programs; this often brings advantages in both cost (running computers is cheaper than employing humans), and performance (computers can solve many problems faster and more accurately than humans). With tasks that can easily be expressed algorithmically, this replacement can be effected by following principles of software and systems design; for example, a computerised tax system can follow an algorithm to retrieve income data and apply tax rate and exemption rules to compute the payable tax.

However, many tasks cannot be feasibly solved by engineering an algorithm; when the task is complex or poorly understood, even experts who are able to

Index	**Features**				**Label**
Patient	Age	Gender	Blood Pressure (mm Hg)	...	Heart Disease?
1	63	Male	145	...	No
2	67	Male	160	...	Yes
3	41	Female	130	...	Yes
4	56	Male	120	...	No
5	57	Female	120	...	No

Table 1.1: Heart Disease Training Data

solve the problem themselves may be unable to describe the process by which they achieve this. For example, in medical diagnosis, Kononenko [46] describes an expert system for diagnosing heart disease in new-born babies; although it required several man-months to create, it was out-performed by a simple[1] learning algorithm. The supervised learning paradigm provides an alternative: instead of writing an algorithm to solve a task, we collect a set of input/output *examples*, and by examining these we hope to *derive* general rules that succeed on future inputs.

To continue with the diagnosis example, we might take measurements from groups of patients with heart disease, and the same measurements from patients without heart disease. These measurements would form the inputs (or *features*) of the *training data*; they might include quantities such as age or blood pressure, as well as categorical variables such as gender. The desired output (or *label*) would be a binary variable indicating whether each patient had heart disease. In Table 1.1, we show some examples from the Heart Disease dataset that is used later in this thesis.

A supervised learning *algorithm* takes this training data as an input. As an output, it returns a *model*. The model is able to take inputs (i.e. measurements from new patients) and return outputs (i.e. predictions of whether they have heart disease). The idea is that by examining the examples in the training data, the learning algorithm is able to produce a model that *accurately reflects the diagnosis process for heart disease*. Figure 1.1 gives a schematic overview of the supervised learning process.

[1] Naïve Bayes

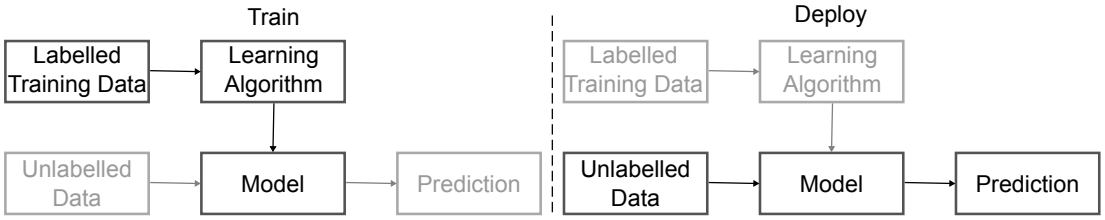

Figure 1.1: Supervised Learning Block Diagram. The *learning algorithm* is used in the training stage to produce a model. The *model* is used in the deployment stage to make predictions on unlabelled data.

1.1.2 Ensemble Methods and Diversity

Ensemble methods aggregate predictive models in an attempt to improve upon their individual performances, often using a voting system. The intuition behind this can be seen by considering the purpose of voting systems in other domains. Votes are often used where we expect *most* voters to make the best decision — for example, in the "Who Wants to be a Millionaire?" game-show [12], the *ask the audience* rule gives the contestant the option of agreeing with the majority vote of the audience. While it is unlikely for individual audience members to be knowledgeable enough to answer every question correctly, the aggregated audience answers will be accurate as long as individual audience members pick the correct answer more often than they pick the next most popular answer.

In an ensemble model, the 'voters' are *base learners* — individual models that each produce predictions on input data. In the simplest case, the 'ensemble prediction' is whichever prediction was most popular among the base learners. Figure 1.2 shows schematically how these voting predictions work. The primary concern of an ensemble algorithm is how to train individual models; in the literature there are many suggestions for what training procedures will result in the most accurate ensembles. One important and well-understood tool for investigating the behaviour of ensembles is the *voting margin*. This is a measurement of how many of the base learners in the ensemble make correct predictions, and can roughly be interpreted as the *confidence* of the ensemble in its prediction; we provide a precise definition in Section 2.2.3.

The idea of voting to improve the quality of a prediction relies upon *diversity* between the voters: it is okay for them to vote incorrectly sometimes, so long *most* voters vote correctly on every prediction. In a game of "Who Wants to be a Millionaire", diversity could be achieved by ensuring that the audience had a

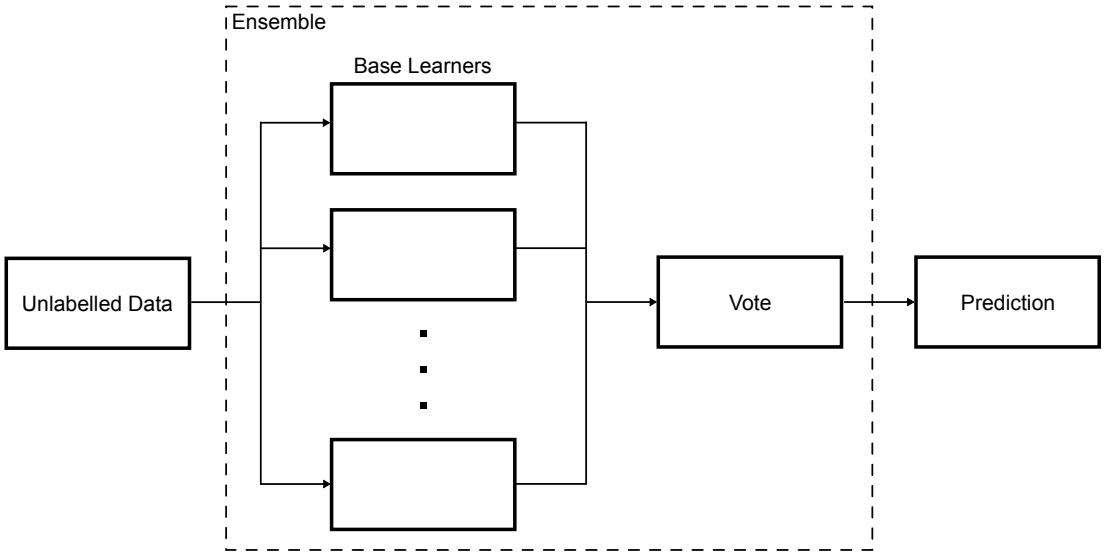

Figure 1.2: Ensemble Prediction Block Diagram. Base learners produce predictions, and the voting process chooses a single label as the ensemble's prediction.

wide variety of experts — some with deep knowledge of sports, or medicine, or politics, et cetera; if instead the audience comprised only experimental physicists, we would expect good answers on physics questions, but not on disparate topics like art history.

In ensemble methods, diversity is achieved through the algorithm that coordinates the training of component models. However, the literature abounds with speculation over how exactly diversity should be quantified and optimised; opinions range from enthusiasm [38]:

> *"it is almost an axiom that the base classifiers must be diverse in order for the ensemble to generalize well."*

through cautious acceptance [49]:

> *"there is still much room for heuristic in classifier combination, and diversity might be one of the lines for further exploration"*

to complete pessimism [75]:

> *"not only [does] no useful measure [of diversity] exist today, but it is unlikely that one will ever exist."*

In Sections 1.2 and 1.3, we outline how we intend to usefully contribute to this debate.

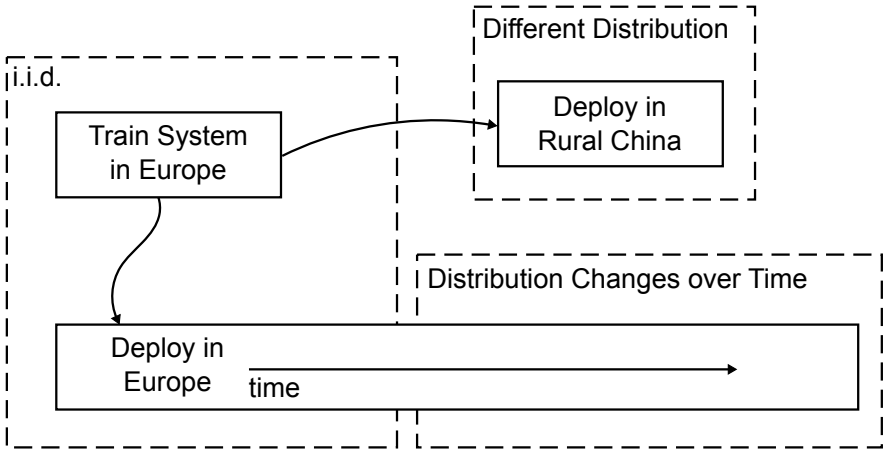

Figure 1.3: Schematic representation of violations of the i.i.d.ãssumption. When the system is deployed in rural China, the immediate change in demographic will damage its performance. Deployment in Europe is initially successful, since the unlabelled data is i.i.d., but over time gradual changes to lifestyle and diet cause the distribution to deviate too far from that of the training data.

1.1.3 Non-Stationary Learning

The supervised learning paradigm is quite general, and can be applied to many common tasks. However, it does make various assumptions that do not always hold in practice. In non-stationary learning, we address one such assumption: *all data is independent and identically distributed* (i.i.d.). This property requires essentially that there is a fixed way in which every item of data (both training data examples, and unlabelled data) is sampled, and implies that *training data is representative of unseen data*. Whether data is actually i.i.d.d̃epends partly on the data collection process (for example, if we collect data from European patients to train a medical diagnosis system on, the model may not perform well if we deploy it in rural China), but also on the nature of the problem (even if we carefully train and deploy a diagnosis system on patients from a fixed distribution, the long term success of the system is not assured; changes to lifestyle and diet will eventually destroy the i.i.d.p̃roperties of the data). These kinds of distribution change are illustrated in Figure 1.3.

In many scenarios where such issues occur, a viable solution is to follow a *non-stationary learning* procedure. The expectation that the distribution may change renders the idea of 'train-then-deploy' inadequate; it is necessary for *continual* training so that changes to the distribution can affect the model. Therefore,

in non-stationary learning, labelled training data arrives sequentially, and predictions on unlabelled data may be required at any time. This scenario is applicable to a number of high-profile tasks such as spam email detection, weather prediction, and economic forecasting. Existing high-quality non-stationary learning algorithms can be expected to improve performance and reduce manual intervention in real-world applications; interest in applying ensembles to such tasks is still relatively high, with a number of competitive algorithms proposed in recent years [5, 31, 45, 66, 67].

1.2 Motivation

In this section, we present some motivation for studying our areas of interest, and describe how our contributions will benefit future researchers and practitioners.

1.2.1 Understanding Ensemble Diversity

The field of ensemble diversity is characterised by strong intuitive arguments, useful heuristics, an abundance of diversity measures with varying degrees of theoretical and empirical support, and relatively few highly-cited algorithms (Random — and Rotation — Forests [9, 74] being the only examples of notorious algorithms with explicit diversity-based motivation of which we know). In Section 1.1.2, we presented three quotes that describe a spectrum of different views on diversity. Because of this uncertainty regarding its definition, purpose and utility, we believe that efforts to consolidate, relate to external fields, and experimentally corroborate aspects of ensemble diversity are extremely valuable.

The motivation behind our contributions here is exemplified by our own use of diversity in non-stationary learning. A consideration for researchers who attempts to rigorously apply diversity theory is *"How should diversity be measured, and what is the meaning of this measure?"*

For example, we would not be justified in selecting a diversity measure, accepting it as axiomatically beneficial, and then using it as a vacuous explanation for the behaviour of the ensemble; the idea that "the ensemble is successful because it has high diversity" should be treated with caution, since one of the most concrete conclusions of diversity literature is that there is *no simple link between high diversity and good ensemble performance* [49, 75].

If we can improve our understanding of how diversity interacts with other properties of ensembles, then it should become easier to *exploit* this understanding to suffix that incomplete explanation: "the ensemble is successful because it has high diversity, *implying X about its other properties, which influences performance because of Y*".

1.2.2 Diversity and Adaptation in Non-Stationary Learning

Non-stationary learning is an important class of supervised learning task. Since such problems occur in real-world scenarios, this alone is sufficient motivation to aim to improve our understanding of how algorithms can better adapt.

Our specific topic, *diversity in non-stationary learning*, reflects recent interest in applying ensembles to non-stationary learning tasks; in the Chapter 4, we will provide more intuition as to why ensembles are seen to be especially appropriate in these environments.

Of course, just because ensembles provide a promising approach to non-stationary learning tasks does not especially justify us in investigating *ensemble diversity* in these tasks. However, in Chapter 5 we explain how our insights from our contributions in ensemble diversity cause us to *expect* diversity to be influential in determining how quickly an ensemble adapts to a new concept. Depending on the degree to which our expectations are realised, diversity could form an important part of a successful non-stationary learning ensemble algorithm.

1.3 Contributions

Here, we present two main areas of contribution. The first of these — *Interpreting Diversity using Voting Margins* — is a significant insight into a fundamental theoretical connection between two aspects of ensemble methods. The next contributions exploit this connection, using our insights to understand how and why we might *Manage Diversity for Non-Stationary Learning.*

1.3.1 A Relationship between Diversity and Margins

In Chapter 3, we show that:

1. A large number of diversity measures from the literature [1, 15, 16, 18, 26, 29, 30, 35, 43, 48, 63, 81, 95] can be exactly described within the *voting margin* framework. 'High diversity' corresponds to voting margins that are close to zero.

2. Although high diversity is often considered to be beneficial, ideas from margin theory and our empirical investigation suggest that *low diversity* can reduce generalisation error in some circumstances.

3. By applying Taylor approximations, we can see two interesting relationships; one is between 'entropy' and 'disagreement' which answers an open question of Chen [16]. The other is between double fault diversity [30] and the exponential loss function used by Adaboost [76] — we investigate this relationship empirically.

1.3.2 Managing Diversity for Non-Stationary Learning

Our previous contributions demonstrate a strong theoretical connection between diversity and margins. The next part of the thesis (Chapter 5) investigates this relationship in a non-stationary environment. We:

1. Derive a novel incremental learning algorithm, DivBoost, that uses our diversity-margin correspondence from Chapter 3 to manage diversity according to a parameter.

2. Empirically identify (Using DivBoost) specific scenarios where high diversity contributes to the adaptation of the ensemble, applying techniques from the literature [65, 82] to generate toy data with known properties.

3. Extend our experimental study to include more datasets, both real and artificial, finding that other factors in non-stationary learning (such as change detection and windowing) dominate the impact of diversity in realistic scenarios.

1.4 Thesis Structure

The thesis is separated in to two parts — Part I starts with some background on ensembles, diversity and voting margins (Chapter 2). We then present our

first contributions on diversity and margins (Chapter 3). Part II starts with another background chapter on non-stationary learning and the use of ensembles for such tasks (Chapter 4), and continues by presenting the contributions relating to non-stationary learning (Chapter 5). We conclude with a summary of the contributions, and suggestions for future work (Chapter 6).

1.5 Notation

Here, we introduce the relevant notation for the thesis. Tables 1.2, 1.3 and 1.4 summarise all the appropriate values and functions. A small number of values are 'overloaded' to maintain consistency with other published works (e.g. some EDDM parameters), however in general we have converted others' notation in favour of internal consistency. Overloaded notation is only ever used in unambiguous contexts.

We have generally given some information about the definitions of notation here; either a full definition, or a description of its possible values. All notation is additionally introduced with greater depth in the body of the thesis.

Notation	Definition	Description
$Pr(a)$	$Pr(a) \in [0,1]$	Probability of a
$\delta[a]$	$\delta[a] \in \{0,1\}$	0/1 Indicator function
$Poiss$	$Poiss(\lambda) \in \mathbb{N}$	Poisson random number
λ	$\lambda \in [0,\infty]$	Poisson parameter
$\mathcal{X}$	$\mathcal{X} = \mathbb{R}^d$	Feature space
d	$d \in \mathbb{N}$	Number of features
$\mathcal{Y}$	$\mathcal{Y} = \{-1,1\}$	Target Space
$\mathbf{x}$	$\mathbf{x} \in \mathcal{X}$	Features
y	$y \in \mathcal{Y}$	Target
$\langle \mathbf{x}, y \rangle$	-	Example
N	$N \in \mathbb{N}$	Training dataset size
$\mathcal{T}$	$\{\langle \mathbf{x}_1, y_1 \rangle \dots \langle \mathbf{x}_N, y_N \rangle\}$	Training dataset
$H(\mathbf{x})$	$H : \mathcal{X} \to \mathcal{Y}$	Predictive model
e_{tr}	$\frac{1}{N}\sum_{i=1}^{N} \delta[H(\mathbf{x}_i) \neq y_i]$	Training Error
e_{gen}	$\int_{\mathbf{x},y \in \mathcal{X}\times\mathcal{Y}} \delta[H(\mathbf{x}) \neq y] Pr(\mathbf{x},y) d\mathbf{x}dy$	Generalisation Error
$h(\mathbf{x})$	$h : \mathcal{X} \to \mathcal{Y}$	Base learner
α	$\alpha \in \mathbb{R}$	Base learner weight
L	$L \in \mathbb{N}$	Ensemble size
$F(\mathbf{x})$	$\sum_{l=1}^{L} \alpha_l h_l(\mathbf{x})$	Ensemble weighted sum
$m(\mathbf{x},y)$	$\frac{yF(\mathbf{x})}{\sum_{l=1}^{L}\lvert\alpha_l\rvert}$	Margin
$\overline{m}$	$\frac{1}{N}\sum_{i=1}^{N} m_i$	Average Margin
θ	$\min_i m_i$	Minimum Margin
s	$E_{Pr(\mathbf{x},y)}[m(\mathbf{x},y)]$	Strength
c_i	$\frac{1}{L}\sum_{l=1}^{L} \delta[h_l(\mathbf{x}_i) = y_i]$	Proportion of correct predictions
$\mathcal{L}$	-	Learning algorithm
C	-	Cost Function
$D_l(i)$	$D_l(i) \in [-1,1]$	Example weight
ε	$\sum_{i=1}^{N} D_l(i)\delta[h_l(\mathbf{x}_i) \neq y_i]$	Weighted error
$\widehat{Pr}(y\lvert\mathbf{x})$	$\frac{1}{2}(1 + \frac{F(\mathbf{x})}{\sum_{l=1}^{L}\lvert\alpha_l\rvert})$	Predicted probability
$\mathbf{w}$	$\mathbf{w} \in \mathbb{R}^d$	Linear classifier weights
b	$b \in \mathbb{R}$	Linear classifier bias
m_{geo}	$\frac{y\cdot(\mathbf{w}^\top\mathbf{x}-b)}{\lVert\mathbf{w}\rVert_2}$	Geometric margin
v	-	VC-dimension
η	$\eta \in [0,1]$	Bound probability

Table 1.2: Notation

Notation	Definition	Description
f	$f \in [0, 1]$	Output flip rate parameter
γ	$\gamma \in [0, \frac{1}{2}]$	Adaboost weak learning parameter
Z	$Z = \sum_{i=1}^{N} D_l(i)$	Adaboost normalisation constant
p	$p \in \mathbb{N}$	DWM period
β	$\beta \in [0, 1]$	DWM discount
θ	$\theta \in [0, 1]$	DWM pruning threshold
f	$f \in \mathbb{R}$	DWM cumulative prediction
μ	$\mu \in [0, \infty]$	DivBoost quadratic minimum
p	$p \in [0, 1]$	DDM mean
s	$s \in [0, 1]$	DDM standard deviation
p'	$p' \in [o, \infty]$	EDDM mean
s'	$s' \in [0, \infty]$	EDDM standard deviation
α	$\alpha \in [\beta, 1]$	EDDM warning threshold
β	$\beta \in [0, \alpha]$	EDDM drift threshold
$N_t(y)$	$N_t(y) \in \mathbb{N}$	Naïve Bayes class-conditional count (at time t)
$\mu_t^{(j)}(y)$	$\mu_t^{(j)}(y) \in \mathbb{R}$	Naïve Bayes class-conditional mean (j^{th} feature)
$\gamma_t^{(j)}(y)$	$\gamma_t^{(j)}(y) \in [0, \infty]$	Naïve Bayes class-conditional sum-of-squares
$\sigma_t^{(j)}(y)$	$\sigma_t^{(j)}(y) \in [0, \infty]$	Naïve Bayes class-conditional standard deviation

Table 1.3: Algorithm Parameter Notation

	h_j correct	h_j wrong
h_k correct	$N_{j,k}^{11}$	$N_{j,k}^{01}$
h_k wrong	$N_{j,k}^{10}$	$N_{j,k}^{00}$

Table 1.4: Pairwise contingency table for a pair of base learners, j, k. Each cell is a *count over training data* of how often two classifiers, j and k, are correct/wrong in accordance with the superscript.

Part I

Diversity and Margins

Chapter 2

Background and Related Work

2.1 Ensemble Algorithms for Supervised Learning

In the previous chapter, we introduced the idea of ensemble algorithms and provided some intuitive explanations for their popularity. In this section, we describe supervised learning and voting ensembles in more detail and provide the relevant notation. We then introduce *Adaboost*, a popular ensemble algorithm, and show how *voting margins* are used to understand its behaviour. Finally, we discuss gradient descent and the *Anyboost* algorithm, which is a generalised version of Adaboost that we will use in our experiments later in this thesis.

2.1.1 Supervised Learning

Supervised learning problems can be characterised by a probability distribution over feature/target space in conjunction with a finite sample that is drawn independently from this distribution (i.i.d.). In the 2-class classification problems that we will consider, the feature space will be continuous, defined as $\mathcal{X} = \mathbb{R}^d$, where d is the number of features. The target space is binary, with positive and negative classes: $\mathcal{Y} = \{-1, 1\}$. The unknown distribution from which data is generated is $Pr(\mathbf{x}, y)$. *N training examples* are sampled i.i.d.from this distribution to give a *training set*[1] $\mathcal{T} = \{\langle \mathbf{x}_1, y_1 \rangle \ldots \langle \mathbf{x}_N, y_N \rangle\}$, with examples $\langle \mathbf{x}_i, y_i \rangle \in \mathcal{X} \times \mathcal{Y}$.

From $\mathcal{T}$, the learning task is to derive a model, $H(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{Y}$, which achieves

[1] Although the training set is described as a 'set', it may contain identical examples such that $\langle \mathbf{x}_j, y_j \rangle = \langle \mathbf{x}_k, y_k \rangle, j \neq k$.

low *generalisation error* with respect to $Pr(\mathbf{x}, y)$; generalisation error is defined as:

$$e_{\text{gen}} = \int_{\mathbf{x},y \in \mathcal{X} \times \mathcal{Y}} \delta[H(\mathbf{x}) \neq y] Pr(\mathbf{x}, y) d\mathbf{x} dy. \tag{2.1}$$

This corresponds to the probability that, given a random sample from $Pr(\mathbf{x}, y)$, H assigns the incorrect label. In noisy scenarios it may be possible for a given $\mathbf{x}$ to receive more than one label — either because the target variable is inherently non-deterministic, or because the features in $\mathbf{x}$ are not sufficient to predict y — in these cases, the *conditional probability* $Pr(y|\mathbf{x})$ for each label *given* the feature values can be between 0 and 1, so the deterministic model H cannot always correctly predict $\mathbf{x}$. The lowest achievable error occurs when H always predicts the *most probable* label. This is the *Bayes error*:

$$e_{\text{bayes}} = \int_{\mathbf{x},y \in \mathcal{X} \times \mathcal{Y}} \delta[\arg\max_{y'} Pr(y'|\mathbf{x}) \neq y] Pr(\mathbf{x}, y) d\mathbf{x} dy. \tag{2.2}$$

Therefore, the Bayes-optimal classifier is one where $H(\mathbf{x}) = \arg\max_y Pr(y|\mathbf{x})$. However, constructing such a classifier is difficult since $Pr(y|\mathbf{x})$ is unknown, and the only clue we have to the nature of this distribution is the training data set.

The intuition behind the formulation of the supervised learning task — i.e. the use of a training data set — is that the training data is representative of the unknown distribution, so that a model which accurately predicts the training data might be expected to achieve a low generalisation error. To this end, many supervised learning algorithms use the *training error* to guide their training process:

$$e_{\text{tr}} = \frac{1}{N} \sum_{i=1}^{N} \delta[H(\mathbf{x}_i) \neq y_i]. \tag{2.3}$$

2.1.2 Fitting the Model to the Training Data

It may seem that, using the law of large numbers, we can view e_{tr} as an estimate of e_{gen}; because of this, many approaches to supervised learning aim to learn models with low training error. However, to optimise *only* training error is not useful, since the dependence of H on $\mathcal{T}$ will cause e_{tr} to be biased. Figure 2.1 shows why this is a problem: a model that is *too* well fitted to the training data can often have worse performance than a simpler model with higher training error.

This phenomenon is called *overfitting*, and is an important consideration in supervised learning algorithms. "Occam's Razor" provides one solution: choose

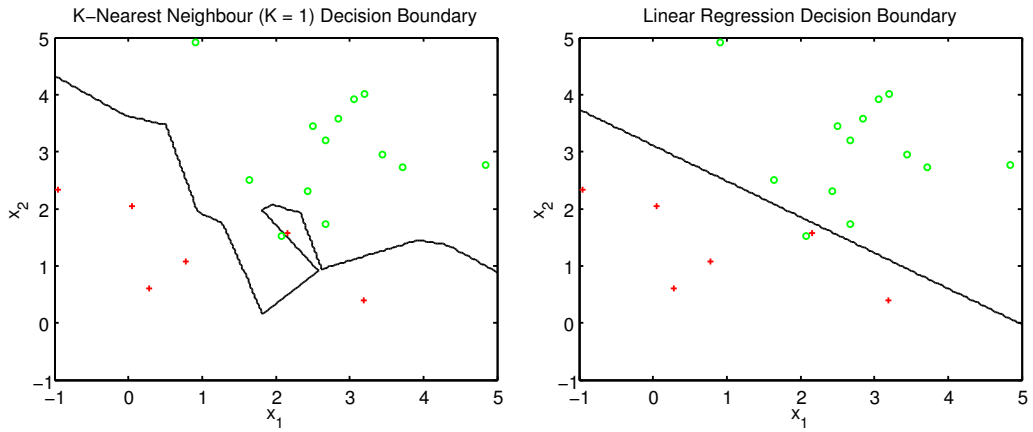

Figure 2.1: Comparison of the decision boundaries from KNN and linear regression on a toy Gaussian dataset (the dataset is described in Appendix A.1). The generalisation accuracies are 84.6% and 91.2% respectively.

the simplest model that 'explains' the data. The toy Gaussian task in Figure 2.1 shows that the linear model correctly classifies ('explains') most of the training data; furthermore, it is substantially 'simpler' than the KNN classifier[2] (a rough measure of simplicity is the number of model parameters, which is $d + 1 = 3$ for linear regression and effectively $N = 20$ for 1-nearest neighbour; though the VC-dimension [88] gives a more robust way to quantify this). The idea of encouraging simpler solutions is known as *regularisation*, and is usually incorporated into supervised learning algorithms, such as support vector machines (SVMs) which can use 'soft margins' to trade-off between training accuracy and model simplicity [17], or decision trees [11], which can enforce a maximum tree depth.

While using simple models can often alleviate overfitting problems, if the complexity of the learning task is unknown then we risk choosing a model that is *too* simple, which gives similarly poor performance. Ensemble algorithms take an alternative approach to supervised learning problems, exploiting multiplicities of individual models in ways which often mitigate overfitting while still obtaining sufficient expressive power to learn complicated tasks.

[2]K-Nearest Neighbour (KNN) classifiers predict each input with the majority label of the K nearest training examples, where K is a user-specified parameter.

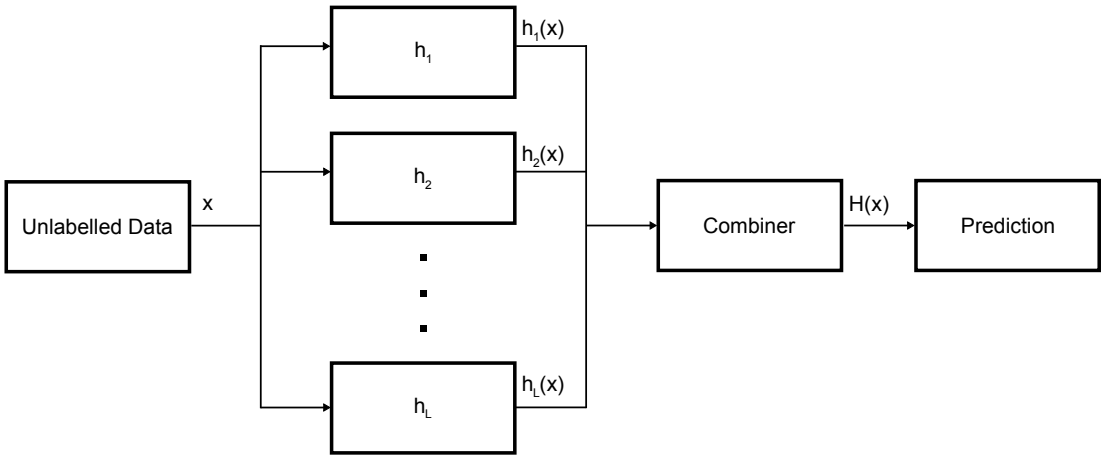

Figure 2.2: Block diagram of an ensemble. For weighted voting ensembles, the combiner takes the form of Equation 2.5.

2.1.3 Ensemble Algorithms

In an ensemble, multiple models are aggregated by some combination strategy. A popular combiner is a vote, as in *weighted ensembles*:

$$F(\mathbf{x}) = \sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}), \tag{2.4}$$

$$H(\mathbf{x}) = sign(F(\mathbf{x})), \tag{2.5}$$

where L *base learners* and associated weights $\alpha_1 \ldots \alpha_L$ are used to give a single prediction. F gives a real-valued output with a magnitude that depends on *how many* and *which* base learners made each prediction. We give a block diagram representation of an ensemble in Figure 2.2.

The success of ensembles has been attributed to a number of reasons. Dietterich [19] describes three: a *statistical* motivation because of the *averaging* effect of using multiple base learners. A *computational* advantage over individual models, where single learners may converge to a local optima while multiple base learners have more opportunities to converge to better optima, and are correspondingly more likely to arrive at a better solution. Finally, a *representation* benefit as the use of an ensemble expands the 'hypothesis space' in which models can be found.

In the next section, we describe a particular ensemble algorithm — Adaboost

— in detail. An important aspect of Adaboost is exploitation of the *representational* benefits of an ensemble; by combining a sufficient number of base learners, it is possible to 'boost' inaccurate models so that they predict training data perfectly.

2.2 Boosting

Boosting algorithms have formed a prominent subfield within ensemble learning literature for much of its history. The original idea of boosting has its roots in the *hypothesis boosting problem* [41]; whether there is equivalence between *weak* learnability (prediction with better accuracy than random guessing) and *strong* learnability (arbitrarily high accuracy on training data). Schapire [77] proved an affirmative answer to this question, and subsequently (with Freund) [76] derived the Adaboost algorithm, which takes as input a weak learning algorithm, and uses it to train a weighted ensemble of weak models which are combined to give a strong model. In this section, we introduce Adaboost, discuss some of the related theory on *margins* [78], and then describe Anyboost [62] — a generalised version of Adaboost that we will use later in this thesis.

2.2.1 The Adaboost Algorithm

Adaboost is an iterative algorithm, running for a pre-defined number of rounds and adding a single base learner in each round. The central notion is that of *weighting* both examples *and* base learners during the training process. Examples are weighted according to how difficult it has been to learn their true labels, while learner weights indicate how useful each learner is to the ensemble.

The specifics of Adaboost hinge upon the use of a surrogate cost function:

$$C_{\text{ada}} = \sum_{i=1}^{N} e^{-y_i F(\mathbf{x}_i)}, \tag{2.6}$$

$$\geq \sum_{i=1}^{N} \delta[H(\mathbf{x}_i) \neq y_i], \tag{2.7}$$

which upper-bounds the training error; by setting the example and learner weights such that this cost is reduced at each step, the classification error is *necessarily*

reduced, though not monotonically. Under modest assumptions[3], Adaboost can be shown to eventually achieve maximum training accuracy, if enough base learners are added to the ensemble. This is ensured by two update rules for D_l, the example weights to train the $(l+1)^{\text{th}}$ learner with, and α_l, the voting weight of the l^{th} learner:

$$D_l(i) = \frac{e^{-y_i \sum_{k=1}^{l} \alpha_k h_k(\mathbf{x}_i)}}{\sum_{j=1}^{N} e^{-y_j \sum_{k=1}^{l} \alpha_k h_k(\mathbf{x}_j)}}, \tag{2.8}$$

$$\alpha_l = \frac{1}{2} \log \frac{1-\varepsilon_l}{\varepsilon_l}, \tag{2.9}$$

where ε_l is the weighted error of the l^{th} base learner:

$$\varepsilon_l = \sum_{i=1}^{N} D_{l-1}(i) \delta[h_l(\mathbf{x}_i) \neq y_i]. \tag{2.10}$$

Schapire and Freund [76] show that training error is upper bounded by:

$$e_{\text{tr}} \leq (\sqrt{1-4\gamma^2})^L, \tag{2.11}$$

where $\varepsilon_l \leq \frac{1}{2} - \gamma$ represents the weak learning assumption on the base learners.

Algorithm 1 Adaboost

Require: $L \leftarrow$ number of base learners
Require: $\mathcal{L} \leftarrow$ base learning algorithm
Require: $\mathcal{T} \leftarrow$ training data set

$D_0(i) \leftarrow \frac{1}{N} \quad \forall i$
for $l = \{1 \dots L\}$ **do**
  $h_l \leftarrow \mathcal{L}(\mathcal{T}, D_l)$
  $\varepsilon_l \leftarrow \sum_{i=1}^{N} D_{l-1}(i) \delta[h_l(\mathbf{x}_i) \neq y_i]$
  if $\varepsilon_l \geq \frac{1}{2}$ **then**
    $L \leftarrow l - 1$
    break
  end if
  $\alpha_l \leftarrow \frac{1}{2} \log \frac{1-\varepsilon_l}{\varepsilon_l}$
  $D_l(i) \leftarrow \frac{D_{l-1}(i) e^{-y_i \alpha_l h_l(\mathbf{x}_i)}}{Z_l} \quad \forall i$
end for

[3]The base learning algorithm must perform at least a fixed but arbitrarily small amount better than random guessing on weighted training data.

Adaboost is defined in full in Algorithm 1. The $\mathcal{L}$ algorithm in Adaboost needs to train a base learner using the example weights provided with D_{l-1}; some existing learning algorithms can be naturally adapted to accommodate weighted data, but generally it is possible to replace $\mathcal{T}$ with a weighted sample-with-replacement from $\mathcal{T}$; the distribution of the weighted sample is an approximation to the distribution $D_l(i)$ [76, Section 4.1].

The conditional statement "**if** $\varepsilon_l \geq \frac{1}{2}$ **then**" considers the possibility that the weak learning assumption is violated. This could happen if the base learner is not sufficiently expressive to learn a particular weighting of the data — in which case $\varepsilon_l = \frac{1}{2}$ — or in the weighting-by-sampling scenario discussed above, a base learner that satisfies the weak learning assumption on the sampled dataset that it was trained on need not necessarily achieve $\varepsilon_l < \frac{1}{2}$ when considering the *weighted* data. However, we observe that:

- Base learners with weighted error $\frac{1}{2}$ can be ignored; if $\varepsilon_l = \frac{1}{2}$, then Equation 2.9 will set $\alpha_l = 0$.

- Base learners with weighted error of more than $\frac{1}{2}$ are 'inverted' by Equation 2.9; they receive a negative weight, with the effect of inverting all their predictions (in which case their effective weighted error becomes $1 - \varepsilon_l$ by the symmetry of 2-class problems), and the magnitude of the weight is correct for this 'inverted' model.

Combined, these allow us to omit the weak learning check in our implementation; Equation 2.9 ensures that α_l are chosen correctly regardless of what the weighted error is; at worst, we will end up including many 0-weighted learners.

2.2.2 Empirical Evaluation of Adaboost

Experimental studies on Adaboost have generally confirmed its value as a powerful learning algorithm. Many of these have compared Adaboost against Bagging [7], which is an algorithm for producing unweighted ensembles where each base learner is trained on a uniformly weighted sample-with-replacement from $\mathcal{T}$.

Bauer and Kohavi [4] perform experiments on 14 datasets, comparing Adaboost, Bagging and some variants with tree or Naïve Bayes base learners. Their results generally indicate that boosting algorithms outperform Bagging, and in almost all situations both ensemble algorithms improve upon the performance

of individual learners. Dietterich [20] experiments on 33 learning problems with Adaboost, Bagging and Randomisation using decision tree base learners, finding that Adaboost is almost always superior, but that the addition of noise to the target variable can be especially detrimental, and that Bagging is often better in scenarios exhibiting such noise. Maclin and Opitz [58] compare Bagging and Boosting with decision tree and neural network base learners, also confirming that the accuracy of Adaboost suffers when the labels are noisy; on low-noise datasets, however, Adaboost is generally the better algorithm of the two. Quinlan [72] performs detailed experiments with C4.5 base learners using Adaboost and Bagging, and arrives at similar conclusions to other empirical evaluations: Adaboost typically performs very well on low-noise data, but this performance rapidly degrades as artificial noise is added to target variables in the problem.

2.2.3 Understanding Ensemble Behaviour with Voting Margins

Schapire et al. [78] investigate an interesting aspect of Adaboost: it is experimentally observed that, in some cases, adding more base learners continues to improve generalisation accuracy *even after training accuracy reaches 100%*. Not only does this seem anomalous given that we expect Boosting to succeed purely because of its effect on training accuracy, but Freund and Schapire [76] had originally shown:

$$e_{\text{gen}} \leq e_{\text{tr}} + 2\sqrt{\frac{v'(\log \frac{2L}{v'} + 1) + \log \frac{9}{\eta}}{L}}, \tag{2.12}$$

$$\leq e_{\text{tr}} + \tilde{O}\Big(\sqrt{\frac{Lv}{N}}\Big), \tag{2.13}$$

with probability η, where v' is the VC-dimension[4] of the Adaboost ensemble; Freund and Schapire show this to be upper bounded by $v' \leq 2(v+1)(L+1)\log_2 e(L+1)$, where v is the base learner VC-dimension ($\tilde{O}$ is used to hide constant and logarithmic factors; $f(n) = \tilde{O}(g(n))$ means that $f(n) = O(g(n)\log^k g(n))$ for some k). This would suggest that *large Adaboost ensembles overfit* (since L is in the numerator), contradicting the aforementioned phenomenon. Schapire

[4]The VC-dimension is a measure of algorithm complexity; the VC-dimension of an algorithm $\mathcal{L}$ is the largest v such that there exists a set of v datapoints on which $\mathcal{L}$ can achieve 0 training error *regardless of how those datapoints are labelled* [88]. The VC-dimension of a linear classifier in d-dimensional space is $d+1$; the VC-dimension of a 1-nearest neighbour classifier is infinite.

et al. [78] propose an alternative explanation of generalisation error in Adaboost based on voting margins; here we shall introduce margins, and then present the results of Schapire et al. and others.

Geometric Margins

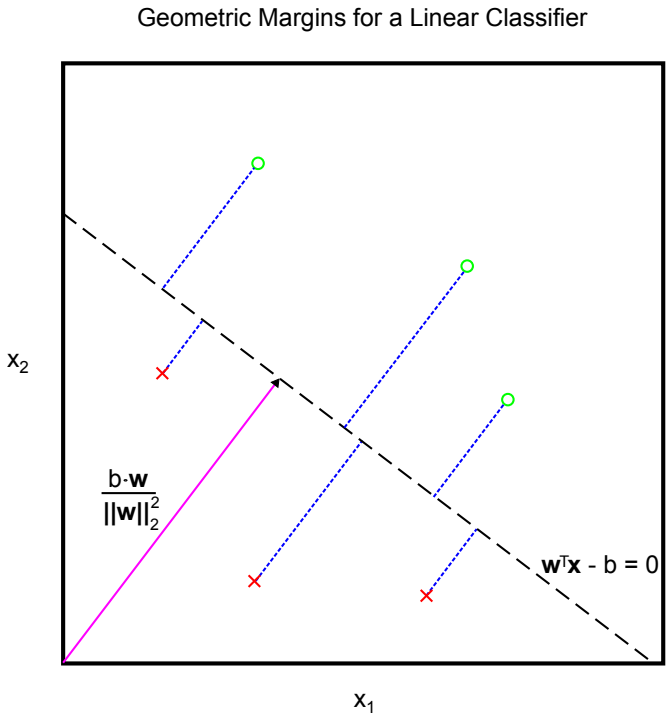

Figure 2.3: Geometric margins for a linear classifier. The solid line shows the direction of the weight vector, $\mathbf{w}$. The dashed line is the decision boundary, $\mathbf{x}$ s.t. $\mathbf{w}^\top \mathbf{x} - b = 0$. The many dotted lines show geometric margins for each of the examples.

A 'margin' represents the distance between a data point and a decision boundary. The concept of margins has been studied in the context of linear classifiers extensively; if we have a linear classifier with weight parameters $w_1 \ldots w_d$ (where d is the dimensionality of the input space) and bias b, then the *geometric margin* of a data point $\langle \mathbf{x}, y \rangle$ is:

$$m_{\text{geo}}(\mathbf{x}, y) = \frac{y \cdot (\mathbf{w}^\top \mathbf{x} - b)}{\|\mathbf{w}\|_2}. \tag{2.14}$$

This is illustrated in Figure 2.3. A margin of 0 would indicate that a point lies *on* the decision boundary. m_{geo} conveys valuable information about the quality

of a classifier; if margins on training data are large, then we can consider there to be a significant gap separating the classes, and the decision boundary cuts through the middle of the gap; smaller margins might tell us that either the data or the decision boundary is less conveniently positioned. Vapnik [88] showed that the geometric margin plays an important role in determining a bound on the generalisation error of support vector machine (SVM) classifiers.

Margins and Boosting

The *voting* margin is a quantity that can be meaningfully computed on any data point with reference to an ensemble. We present its definition for two-class problems, as well as two notational conveniences that we will use:

$$m(\mathbf{x}, y) = \frac{yF(\mathbf{x})}{\|\boldsymbol{\alpha}\|_1} \tag{2.15}$$

$$m_i = m(\mathbf{x}_i, y_i), \tag{2.16}$$

$$\overline{m} = \frac{1}{N}\sum_{i=1}^{N} m_i, \tag{2.17}$$

where $\boldsymbol{\alpha}$ is the vector comprising $\alpha_1 \ldots \alpha_L$. The voting margin quantifies the difference between support *for* the correct class, and support *against*. The sign of the margin indicates correctness or otherwise, so $e_{\text{tr}} = \frac{1}{N}\sum_{i=1}^{N} \delta[m_i < 0]$, while the magnitude of the margin will be between 1, when all base learners agree, and 0, when both classes receive equal support. Figure 2.4 illustrates voting margins in two-class problems.

Since $F(\mathbf{x})$ could alternatively be written as $\boldsymbol{\alpha}^\top \mathbf{x}$, there is a close correspondence between voting and geometric margins; geometric margins are normalised by the L_2 norm, voting margins by the L_1 norm. Equivalently [5], the geometric margin is the L_2 distance between the data point and the decision boundary, while the voting margin is the L_∞ (max norm) distance between the data point and the decision boundary (defined by $F(\mathbf{x}) = 0$) in a hypercube $[-1, 1]^L$ with vertices that represent the 2^L possible outputs of the base learners.

This relationship makes it possible to consider the problem of assigning weights to base learners in an ensemble as a normal supervised learning problem, and

[5] Due to Mangasarian [59, Theorem 2.1, Equation 7]: For a plane $A = \{\mathbf{a} | \mathbf{a}^\top \mathbf{w} - b = 0\}$, the distance between $\mathbf{q}$ and its projection on to a hyperplane A, $\text{proj}_A(\mathbf{q})$ is: $\|\mathbf{q} - \text{proj}_A(\mathbf{q})\|_z = \frac{|\mathbf{q}^\top \mathbf{w} - b|}{\|\mathbf{w}\|_{z'}}$, where the norms z and z' are *dual* to one another, $\frac{1}{z} + \frac{1}{z'} = 1$

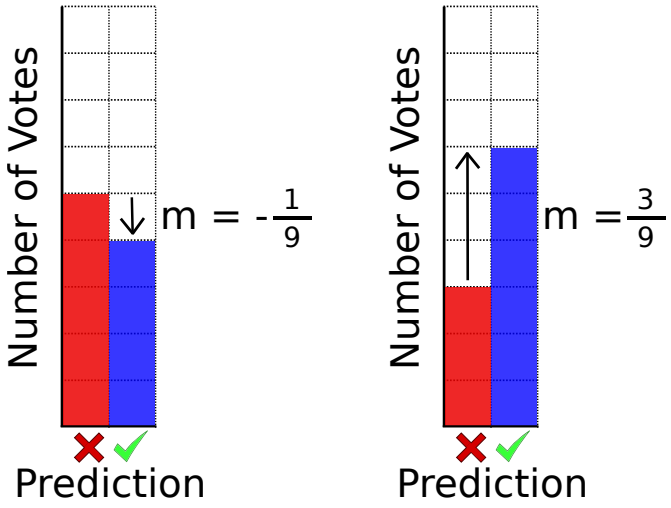

Figure 2.4: Illustration of voting margins. In the scenario on the left, 5 learners vote incorrectly and 4 vote correctly, so the resulting margin is negative (since the majority vote is incorrect) and its magnitude is $\frac{4-5}{9}$. Similarly, on the right, 3 learners are incorrect and 6 are correct, giving a margin of $\frac{6-3}{9}$.

hence generalisation bounds can be obtained based on the distance between data points and the margin, as with Vapnik's result on geometric margins. Schapire et al. [78] show that:

$$e_{\text{gen}} \leq \frac{1}{N}\sum_{i=1}^{N}\delta[m_i \leq \theta] + \frac{2N}{ve^{\frac{1}{2}}} + \Big[\frac{1}{2N}\big(\frac{4v}{\theta^2}\log\frac{N}{v}\log\frac{eN^2}{v} + \log\frac{16e^8}{\theta^2}\log\frac{N}{v}\big(\frac{4}{\theta^2}\log\frac{N}{v}+1\big)^2 - \log\eta\big)\Big]^{\frac{1}{2}}, \tag{2.18}$$

$$\leq \frac{1}{N}\sum_{i=1}^{N}\delta[m_i \leq \theta] + \tilde{O}\Big(\sqrt{\frac{v}{N\theta^2}}\Big), \tag{2.19}$$

for all values of θ with probability $1-\eta$, with $\tilde{O}$ defined as in Equation 2.13. In practice, the tightest bound is often achieved for $\theta = \min_i m_i$, and so this result can be seen as implying that *increasing the minimum margin will tighten a generalisation error bound*. They also show that Adaboost is an effective method for increasing the minimum margin, since the loss function applies exponentially larger weight to examples with small margins (Rätsch et al. [73] additionally illustrate parallels between the maximisation of the minimum voting margin in Adaboost and maximisation of the minimum geometric margin in SVMs). However, Breiman [8] shows that idea of increasing margins does not fully explain

the performance of Adaboost; he presents an alternative algorithm, arc-gv, that achieves more favourable margin distributions than Adaboost in terms of Equation 2.19 but which performs consistently worse than Adaboost. Breiman suggests as an explanation for this that Adaboost concentrates weight on fewer examples than arc-gv, and that this is a contributing factor to its success. Grove and Schuurmans [33] formulate boosting as a linear programming task, and derive an algorithm called LP-Boost which specifically optimises the minimum margin. Like Breiman, they find that this does not achieve the kind of performance improvement over Adaboost that Equation 2.19 would suggest.

Margin theoretic results related to Adaboost have progressed since the original bound of Schapire et al.. The view presented by Wang et al. [92] is more complex, giving a bound that is similar in flavour to Equation 2.19[6] — specifically, both bounds contain a θ value that can be altered to produce the tightest bound. However, the bound of Wang et al. is not usually minimised when $\theta = \min_i m_i$; instead, it is minimised by a term that they describe as the *equilibrium margin* (Emargin, θ^*). They say:

> *"... the Emargin depends, in a complicated way, on the whole margin distribution. Roughly, if most training examples have large margins, then θ^* is large ... The minimum margin is only a special case of the Emargin."*

From this result, it would seem that the relationship between voting margins and generalisation error is more complex than the 'minimum margin' explanation suggests. We take the view that *the whole margin distribution is important*, which is supported by the conclusions of Wang et al.; furthermore, we also conclude that, even though the precise relationship is not clear, *large margins are good* — by this we mean that if we have an opportunity to increase the margin on a single data point without affecting any other aspect of the ensemble, then this change can only have a positive effect on generalisation accuracy (The result of Wang et al. [92, Theorem 6] broadly says this; that increasing θ^* or reducing the proportion of examples with margins less than θ^*, the generalisation error bound is reduced).

[6] We omit full the definition [92, Theorem 7] due to its verbosity; given two functions D^{-1} and u, the bound is $e_{\text{gen}} \leq \frac{v^2+1}{N} + \min_{\theta^*} \frac{N-1}{N} D^{-1}(q, u[\theta^*])$, where q is the proportion of the training data with margins of θ^* or less; significantly, θ^* is chosen to produce the tightest bound.

2.2.4 The Gradient Descent Perspective on Boosting

An alternative view of boosting is based on the idea of stepwise gradient descent over the objective function. Friedman et al. [34] show that by minimising exponential cost $E_{Pr(\mathbf{x},y)}[e^{-yF(\mathbf{x})}]$, the weighted ensemble prediction can be used to estimate conditional probabilities via logistic regression:

$$Pr(y=1|\mathbf{x}) = \frac{e^{2F(\mathbf{x})}}{1+e^{2F(\mathbf{x})}}, \tag{2.20}$$

$$Pr(y=-1|\mathbf{x}) = \frac{e^{-2F(\mathbf{x})}}{1+e^{-2F(\mathbf{x})}}. \tag{2.21}$$

They derive this by considering the minimum of the cost function, at which:

$$F(\mathbf{x}) = \frac{1}{2}\log\frac{Pr(y=1|\mathbf{x})}{Pr(y=-1|\mathbf{x})}. \tag{2.22}$$

From this, they proceed to demonstrate that the example and learner weight updates in Adaboost correspond to *"stagewise estimation procedures for fitting an additive logistic regression model"*.

Subsequently, Mason et al. [62] showed that the stagewise procedures that Friedman et al. describe fit into a general gradient descent framework for optimising cost functions, which they call AnyBoost. For a cost function $C(yF(\mathbf{x}))$ there are appropriate choices of example weights and learner weights that will perform gradient descent on that function:

$$D_l(i) = \frac{\frac{\partial}{\partial y_i F(\mathbf{x}_i)} C(y_i F(\mathbf{x}_i))}{\sum_{j=1}^{N} \frac{\partial}{\partial y_j F(\mathbf{x}_j)} C(y_j F(\mathbf{x}_i))}, \tag{2.23}$$

$$\frac{\partial}{\partial \alpha_l}\frac{1}{N}\sum_{i=1}^{N} C(y_i F(\mathbf{x}_i)) = 0. \tag{2.24}$$

Equation 2.23 defines a direction of steepest descent over the cost on all examples, assuming the function C is convex and monotonic. Once a base learner h_l is trained, its predictions on the training data define another direction; the weak learning assumption implies that it is a descent of the cost function, although not necessarily the steepest descent as defined by the D_l. Equation 2.24 then computes the step size, α_l, that depends on the minimum of the 'slice' of the cost function defined by the base learner. These two update rules are used in an algorithm called *Anyboost*, which differs from Adaboost only in these more

general update rules.

It can be seen that when $C(yF(\mathbf{x})) = e^{-yF(\mathbf{x})}$ is plugged in to Equations 2.23 and 2.24, they give exactly the weight updates used by Adaboost.

2.2.5 Summary

In this section, we have introduced the idea of boosting weak learners to create ensembles with low (potentially 0) training error. We described the Adaboost algorithm, and explained the voting margin framework which has been used to investigate the properties of Adaboost. Finally, we described a general gradient descent algorithm, AnyBoost, that allows ensembles to perform gradient descent on arbitrary convex monotonic cost functions.

2.3 Diversity

Diversity is widely believed to be of paramount importance in ensemble models. Dietterich [20] states that:

> *"The goal of ensemble learning methods is to construct a collection (an ensemble) of individual classifiers that are diverse and yet accurate."*

Every ensemble training algorithm can be seen as a method for producing diverse base learners. This is the case simply because the alternative — the non-diverse ensemble — can easily be seen to be ineffective: there is no way to combine many identical base learners such that the ensemble is different from the individual classifiers.

While, in this sense, the topic of diversity pervades all of ensemble literature, the idea of *quantifying* diversity is not so prevalent. In this section, we will be examining these parts of the literature — where authors perceive that there is some advantage in measuring diversity. Our central contribution in Chapter 3 is an alternative interpretation of diversity that encompasses many of its existing definitions and applications; hence, it is important to examine these elements of the literature closely.

We will first discuss in more detail why diversity is considered to be so important, before presenting definitions for various diversity measures, showing existing work on the unification of these measures, and finally surveying some algorithms that explicitly exploit diversity.

2.3.1 Why is Diversity Important?

In this section, we give more detail on the motivation behind the study of diversity in the literature.

Independent Errors

The idea of ensemble diversity is proposed by Hansen and Salamon [35], where they analyse the impact of *independent errors* in neural network ensembles.

They make some assumptions for convenience — that base learner errors are independent and that all base learners achieve the same individual accuracy — so that ensemble errors can be modelled by a binomial distribution. We introduce some notation for $c(\mathbf{x}, y)$, the proportion of base learners that correctly classify $\langle \mathbf{x}, y \rangle$, its value on a training data point, c_i, and its average $\overline{c}$:

$$c(\mathbf{x}, y) = \frac{1}{L} \sum_{l=1}^{L} \delta[h_l(\mathbf{x}) = y], \tag{2.25}$$

$$c_i = c(\mathbf{x}_i, y_i), \tag{2.26}$$

$$\overline{c} = \frac{1}{N} \sum_{i=1}^{N} c_i. \tag{2.27}$$

Note that $\overline{c}$ is the average base learner accuracy. The result of Hansen and Salamon [35] states that, under these assumptions, the probability of k base learner errors occurring on any data point is:

$$Pr(c_i = 1 - \frac{k}{L}) = \binom{L}{k} \overline{c}^{L-k} (1 - \overline{c})^k, \tag{2.28}$$

and therefore the ensemble error rate is:

$$e_{\text{tr}} = \sum_{k=\lceil L/2 \rceil}^{L} Pr(c_i = 1 - \frac{k}{L}), \tag{2.29}$$

$$= \sum_{k=\lceil L/2 \rceil}^{L} \binom{L}{k} \overline{c}^{L-k} (1 - \overline{c})^k, \tag{2.30}$$

This alone is sufficient motivation to pursue independence of errors in ensembles (and therefore, to some extent, diversity); for $\overline{c} > 0.5$, Equation 2.30 will approach 0 as L increases, as illustrated in Figure 2.5.

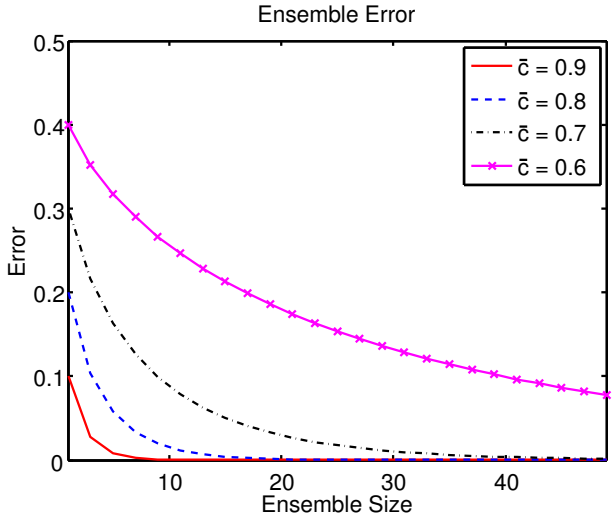

Figure 2.5: Relationship between ensemble size and ensemble error rate for base classifiers making independent errors (Equation 2.30). $\bar{c}$ is the average individual accuracy of base classifiers. For all $\bar{c} > 0.5$, ensemble error will eventually converge on 0.

In this context, 'ensemble diversity' could be considered to be the propensity of the base learners to make independent errors. Hansen and Salamon [35] subsequently derive a specific diversity measure, which we describe in Section 2.3.2, based on this idea. Note that independence of *errors* is different from having *independently trained base learners*: in algorithms such as Bagging [7] and Random Subspaces [37] the training data or feature subspaces are sampled independently for each base learner, which makes them *independently trained* (as opposed to Adaboost [76] where the training data for one learner depends on the predictions of the preceding learner). However, there is still commonality between base learners in the *training data* used, so we might still expect dependence between their errors (for example, 'easy' examples may be classified correctly by most learners, while 'hard' examples are classified wrongly by most learners). Statistical independence of *errors* implies that, for example, knowing $h_l(\mathbf{x}_i)$ does not provide any information about $h_k(\mathbf{x}_i)$ for $l \neq k$.

The general message from the work of Hansen and Salamon is that errors rates can be reduced by:

1. Increasing individual base learner accuracy.

2. Increasing the independence of base learner errors (diversity).

This idea that ensembles should be both accurate and diverse is a general theme in ensemble diversity literature.

Motivation from Regression Learning

The idea of ensemble diversity can be understood from the perspective of regression learning, where the target variable y is real-valued, as are the predictions made by base learners $h(\mathbf{x})$. The ensemble prediction can be a simple average over base learners $H(\mathbf{x}) = \frac{1}{L}\sum_{l=1}^{L} h_l(\mathbf{x})$. In regression, base learner errors can occur either when $h(\mathbf{x})$ is higher or lower than y; so there can be a clear advantage to 'diversity', if base learners that over-predict the target 'compensate' for those that under-predict it. The *ambiguity decomposition* of Krogh and Vedelsby [47] provides a useful perspective on the relationship between diversity and squared error[7] *even when base learners are dependent*:

$$(y - H(\mathbf{x}))^2 = \frac{1}{L}\sum_{l=1}^{L}(y - h_l(\mathbf{x}))^2 - \frac{1}{L}\sum_{l=1}^{L}(H(\mathbf{x}) - h_l(\mathbf{x}))^2, \tag{2.31}$$

where the three terms are described as *ensemble error*, *average base learner error*, and *ambiguity* respectively. Ambiguity describes the differences between base learner predictions and the ensemble prediction, and this decomposition shows that reducing ensemble error can be achieved either by reducing individual error, or increasing ambiguity.

The ambiguity decomposition is exploited in negative correlation learning (NCL) [57], where neural networks are trained with a regularised cost function, where λ is a trade-off parameter:

$$C_{\text{NCL}} = \frac{1}{2}(h_l(\mathbf{x}) - y)^2 + \lambda(h_l(\mathbf{x}) - H(\mathbf{x}))\sum_{j\neq l}^{L}(h_j(\mathbf{x}) - H(\mathbf{x})), \tag{2.32}$$

which encourages base learners to disagree with the ensemble; Brown [13] shows that the regularisation term from NCL can be rearranged to give the ambiguity term from the ambiguity decomposition. Therefore, NCL can be viewed as a procedure for minimising the ensemble squared error by training base learners to

[7]Squared error is a popular error function for regression learning; it captures the absolute difference between predictions and the target variable, with more emphasis placed on larger errors.

be both accurate (low individual error) and diverse (high ambiguity).

Theoretical Relationships

Kuncheva [49] suggests several reasons why it is important to study diversity. The first of these is for the purpose of *finding bounds and theoretical relationships*; the result of Hansen and Salamon [35] that we discussed in Section 2.3.1 is an example of such a result, but there exist several others in the literature which we will review in this section.

Breiman [10] presents an interesting result as part of his work on Random Forests. Random Forests use random feature subsets to encourage diversity, in addition to taking random samples of training data for each learner. By considering the correlation between base learner predictions (diversity) and the average margin (accuracy), Breiman derives a bound on ensemble error:

$$e_{\text{gen}} \leq \frac{\overline{\rho}(1-s^2)}{s^2}, \tag{2.33}$$

where $\overline{\rho}$ is the pairwise average correlation[8] and s is the expected margin $s = E_{Pr(\mathbf{x},y)}[m(\mathbf{x},y)]$. In this bound, $\overline{\rho}$ is considered to mean 'diversity', while s quantifies the average accuracy of individual base learners. The bound is reduced by low correlation between learners or high individual accuracy. Figure 2.6 gives an indication of the implications of the bound. It is assumed that $s > 0$ (i.e. on average, base learners perform better than random guessing), and that learners are generated independently (this applies for Bagging and Random Forests, but not for algorithms like Adaboost). These two assumptions, combined with the fact that $\overline{\rho}$ is an expectation over possible base learner pairs, ensure that both $\overline{\rho}$ and s are strictly positive, so that the bound must also remain positive.

Tumer and Ghosh [86] analyse the *added error* (i.e. error above the Bayes rate) of an ensemble by examining the region around the decision boundary. They show that added error is reduced by a factor of L when base learners have uncorrelated errors (this is the scenario considered by Hansen and Salamon [35]):

$$e_{\text{gen}} = e_{\text{bayes}} + \frac{e_{\text{add}}}{L}, \tag{2.34}$$

[8]The exact definition is $\overline{\rho} = \frac{1}{L(L-1)} \sum_{l=1}^{L} \sum_{k \neq l}^{L} \left(\text{corr}_{Pr(\mathbf{x})}[h_l(\mathbf{x}), h_k(\mathbf{x})] \right)$ where $\text{corr}_{Pr(\mathbf{x})}$ indicates Pearson's correlation with respect to the unknown distribution $Pr(\mathbf{x})$. We refer to this as $\overline{\rho}$ to avoid confusion with the diversity measure $\overline{\phi}$ introduced later, which is an *estimate* of $\overline{\rho}$ on training data.

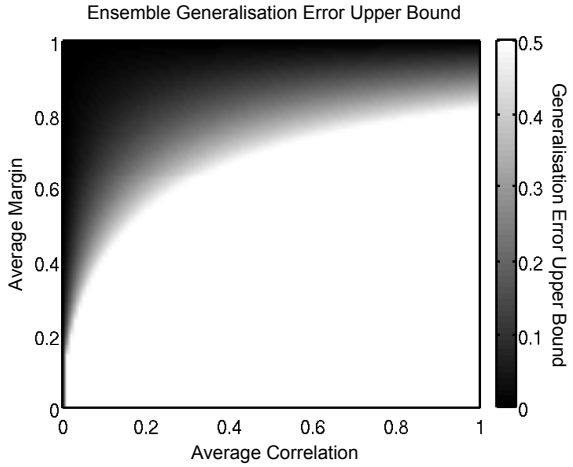

Figure 2.6: Generalisation error bound based on the average margins on generalisation data, and average correlation between base learners

where e_{add} is the added error of individual base learners. Tumer and Ghosh generalise this analysis to consider correlated base learner errors, giving:

$$e_{\text{gen}} = e_{\text{bayes}} + e_{\text{add}}\Big(\frac{1+\psi(L-1)}{L}\Big), \tag{2.35}$$

where (for 2-class problems), ψ is defined as:

$$\psi = \frac{1}{2}(Pr(y=1)\psi_1 + Pr(y=-1)\psi_{-1}), \tag{2.36}$$

$$\psi_c = \frac{1}{L(L-1)}\sum_{l=1}^{L}\sum_{k\neq l}^{L}\text{corr}_{Pr(\mathbf{x})}[\nu_c^l(\mathbf{x}), \nu_c^k(\mathbf{x})], \tag{2.37}$$

$$\nu_c^l(\mathbf{x}) = \delta[h_l(\mathbf{x}) = c] - Pr(y=c|\mathbf{x}). \tag{2.38}$$

The ν values describe the error of $h_l(\mathbf{x})$ with respect to the class probabilities for $\mathbf{x}$. ψ is a pairwise averaged correlation between these errors, weighted by class probabilities. The overall implication here is an interpolation between a correlation of $\psi = 1$, where the ensemble error matches the base learner error, and $\psi = 0$, where Equation 2.35 reduces to Equation 2.34.

Kuncheva [55] has suggested a link between the pairwise averaged Q statistic [94] on base learner outputs and ensemble accuracy under certain assumptions that she describes as "Patterns" of success or failure. Brown and Kuncheva [15] have developed a similar idea based on "Good" and "Bad" diversity, deriving an

expression for ensemble error, similar to the ambiguity decomposition:

$$e_{\text{tr}} = e_{\text{ind}} - D_{\text{Brown}}, \tag{2.39}$$

$$D_{\text{Brown}} = \frac{1}{N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)\frac{1}{L}\sum_{l=1}^{L}\frac{1}{2}(1 - h_l(\mathbf{x}_i)H(\mathbf{x}_i)), \tag{2.40}$$

where D_{Brown} describes the diversity of the ensemble; it can be positive or negative, depending on the sign of $y_i H(\mathbf{x}_i)$ — so if the ensemble makes a correct prediction, high diversity is "good", while an incorrect prediction will cause D_{Brown} to become negative, making diversity "bad". The ensemble prediction distributions that maximise D_{Brown} correspond exactly to those that are assumed under the "Pattern of Success" voting configuration in Kuncheva's analysis of the Q statistic.

These theoretical relationships based on ensemble diversity show that diversity does have some relevance in determining the performance of ensembles, and therefore they provide good motivation for better understanding the role of diversity.

Diversity for Visualisation

This is the second use of diversity that Kuncheva proposes [49]. Since diversity is often considered in conjunction with individual base learner error as a determining factor in ensemble behaviour, examining these properties and the interplay between them can provide interesting insights.

Kuncheva and Whitaker [53] provide bounds on pairwise diversity with respect to individual accuracy for a number of diversity measures. Visualising these bounds on plots of average individual accuracy against diversity helps to provide an intuition for how the two properties interact. Margineantu and Dietterich [60] originally suggested scatter-plots that show kappa diversity (defined in Section 2.3.2) and individual error rate. In Section 2.3.4, we describe how a Pareto front on these plots is used to prune trained ensembles.

Rodriguez et al. [74] use kappa-error plots to understand the behaviour of Rotation Forest ensembles — these visualisations show that Rotation Forests tend to achieve higher individual accuracy that Random Forests or Adaboost, but higher diversity than Bagging ensembles. This provides a more complete explanation for the success of the algorithm.

Diversity in Ensemble Algorithms

The final two motivations for investigating diversity that Kuncheva presents are both oriented around the design of ensemble algorithms — firstly, using diversity to guide classifier selection, and secondly in the process of building the ensemble. We will describe existing algorithms that make use of diversity in detail in Section 2.3.4.

2.3.2 Diversity Measure Definitions

Many of the most popular diversity measures were presented by Kuncheva [54], although they were previously defined outside of the ensemble literature. The relevant notation is introduced in Section 1.5.

Q Statistic

The Q statistic [94] is a measurement of independence between classifiers:

$$Q_{j,k} = \frac{N^{00}_{j,k}N^{11}_{j,k} - N^{01}_{j,k}N^{10}_{j,k}}{N^{00}_{j,k}N^{11}_{j,k} + N^{01}_{j,k}N^{10}_{j,k}}, \tag{2.41}$$

$$D_{\mathrm{Q}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} Q_{j,k}. \tag{2.42}$$

It is also possible to interpret Q as a transformation of the odds ratio:

$$Q_{j,k} = \frac{\mathrm{OR}_{j,k} - 1}{\mathrm{OR}_{j,k} + 1}, \tag{2.43}$$

$$\mathrm{OR}_{j,k} = \frac{N^{00}_{j,k}N^{11}_{j,k}}{N^{10}_{j,k}N^{01}_{j,k}}. \tag{2.44}$$

The odds ratio [22] describes the dependence between two binary variables (in this case, the outputs of two base classifiers), with $\mathrm{OR} = 1$ where there is independence between the two classifiers, $\mathrm{OR} > 1$ when the outputs are positively correlated, and $\mathrm{OR} < 1$ when they are negatively correlated. The meaning of the odds ratio can be broadly described as *the ratio between the probability that h_k is correct when h_j is correct and the probability that h_k is correct when h_j is wrong.*

The transformation applied to give Q adjusts the odds ratio such that 0 indicates independence, and it has minimum and maximum values of -1 and 1 respectively.

Kuncheva [54] has shown limits on the ensemble accuracy based on Q when the ensemble is configured in a 'pattern of success' or 'pattern of failure'. This analysis shows that small Q (high diversity) is beneficial for 'pattern of success' ensembles, but detrimental in 'pattern of failure' ensembles. We discuss this result in detail in Chapter 3.

Phi Coefficient

The phi coefficient is more commonly referred to as Pearson's correlation coefficient (D_ϕ is the special case of Pearson's, for 2-by-2 contingency tables). Like Q, it is a pairwise measure:

$$\phi_{j,k} = \frac{N^{00}_{j,k}N^{11}_{j,k} - N^{01}_{j,k}N^{10}_{j,k}}{\sqrt{(N^{11}_{j,k} + N^{10}_{j,k})(N^{11}_{j,k} + N^{01}_{j,k})(N^{00}_{j,k} + N^{10}_{j,k})(N^{00}_{j,k} + N^{01}_{j,k})}}, \tag{2.45}$$

$$D_\phi = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \phi_{j,k}. \tag{2.46}$$

The numerator in D_ϕ is the same as in Q; consequently, the sign of the two measures is always the same.

Disagreement

Disagreement is introduced by Skalak [81] as *"the percent of instances for which base and complementary classifiers make different predictions but for which one of them is correct"*, and used to investigate the behaviour of various boosting-like algorithms. It can be defined in terms of a contingency table as:

$$D_{\text{dis}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N^{01}_{j,k} + N^{10}_{j,k}}{N}. \tag{2.47}$$

Gatnar [29] proposed 'Hamann's measure' for measuring diversity. However, using the property of the contingency table that $N^{00} + N^{01} + N^{10} + N^{11} = N$, it can be seen to be a simple transformation of disagreement, so we omit it from

our analysis.

Double Fault

Double fault diversity [30] was introduced as a distance measure for clustering neural networks; similar networks would make many coincident errors, so clustering and choosing networks from each resulting cluster would give a 'diverse' ensemble of neural networks. Double fault diversity is defined as:

$$D_{\mathrm{DF}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N_{j,k}^{00}}{N}. \tag{2.48}$$

The same measure is described by Ali et al. as 'correlation' [1].

Kohavi-Wolpert Variance

Kohavi-Wolpert (KW) variance [43] is defined using the idea that the ensemble output can be interpreted as a predicted probability, $\widehat{Pr}(y = 1|\mathbf{x})$. The variance of this probability is:

$$D_{\mathrm{KW}} = \frac{1}{2N} \sum_{i=1}^{N} (1 - \widehat{Pr}(y = -1|\mathbf{x}_i)^2 - \widehat{Pr}(y = 1|\mathbf{x}_i)^2). \tag{2.49}$$

Kuncheva [54] suggests that these probabilities can be quantified based on the proportion of base learners predicting the class in question; her derivation concludes with a correspondence between KW variance and disagreement:

$$D_{\mathrm{KW}} = \frac{(L-1)}{2L} D_{\mathrm{dis}}. \tag{2.50}$$

Interrater Agreement

Interrater agreement (or the kappa coefficient) is a statistical measure [26] which quantifies a level of agreement between 'raters'.

$$D_\kappa = 1 - \frac{L}{N(L-1)} \sum_{i=1}^{N} \frac{c_i(1-c_i)}{\overline{c}(1-\overline{c})}. \tag{2.51}$$

There is also a pairwise version of interrater agreement (which does not, when averaged over all pairs, correspond to D_κ):

$$D_k = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{2(N_{j,k}^{11} N_{j,k}^{00} - N_{j,k}^{01} N_{j,k}^{10})}{(N_{j,k}^{11} + N_{j,k}^{10})(N_{j,k}^{01} + N_{j,k}^{00}) + (N_{j,k}^{11} + N_{j,k}^{01})(N_{j,k}^{10} + N_{j,k}^{00})}. \tag{2.52}$$

This differs from the phi coefficient D_ϕ only by the denominator; both Chen [16] and Kuncheva [54] have observed this similarity empirically. When the two kinds of pairwise disagreement are balanced (i.e. $N^{01} = N^{10}$), the pairwise and non-pairwise kappa coefficients are identical [51] (See Appendix B.2.12 for details).

Difficulty

The difficulty diversity measure [35] is simply defined as the variance of the predicted probability of error over all classifiers:

$$D_{\text{diff}} = \text{Var}_{Pr(\mathbf{x},y)}[c(\mathbf{x}, y)]. \tag{2.53}$$

This variance tells us whether all training examples are equally 'hard' (high diversity), or whether there is variability in the amount of agreement exhibited on each example (low diversity).

Entropy

Entropy is a popular measure from information theory which quantifies the 'randomness' of a distribution. Kuncheva defines one measure based on entropy [54]:

$$D_{\text{ent}} = \frac{L}{N(L - \lceil L/2 \rceil)} \sum_{i=1}^{N} \min\{c_i, 1 - c_i\}, \tag{2.54}$$

while Cunningham uses another quantity [18]:

$$D_H = -\frac{1}{N}\sum_{i=1}^{N} \widehat{Pr}(y=-1|\mathbf{x}_i)\log \widehat{Pr}(y=-1|\mathbf{x}_i) + \widehat{Pr}(y=1|\mathbf{x}_i)\log \widehat{Pr}(y=1|\mathbf{x}_i), \tag{2.55}$$

where we can again interpret $\widehat{Pr}$ as we did for KW variance.

Generalised Diversity

Generalised diversity is based on the probability of exactly k classifiers correctly predicting a randomly chosen example. From this quantity, $Pr(c_i = \frac{k}{L})$, we consider the probability that either one or two *randomly chosen* base learners make an error. Generalised diversity is one minus the ratio between these two quantities; however, we present a simpler form due to Tang [84]:

$$D_{\text{GD}} = 1 - \frac{\sum_{l=1}^{L} l(l-1)Pr(c_i = \frac{L-l}{L})}{\sum_{l=1}^{L} l(L-1)Pr(c_i = \frac{L-l}{L})}. \tag{2.56}$$

This measurement was applied in software failure analysis by Partridge et al. [48].

Coincident Failure Diversity

Another measure defined by Partridge et al. [48] is coincident failure diversity, where $Pr(c_i = \frac{k}{L})$ is evaluated for other values of k:

$$D_{\text{CFD}} = \begin{cases} 0 & \text{if } Pr(c_i = 1) = 1 \\ \frac{\sum_{l=1}^{L}(L-l)Pr(c_i=\frac{L-l}{L})}{(L-1)(1-Pr(c_i=1))} & \text{otherwise} \end{cases} \tag{2.57}$$

Ambiguity

Because of the ambiguity decomposition [47], the term 'ambiguity' is often applied in diversity measures for classification. Chen proposed a decomposition of classification error which features an ambiguity term [16]:

$$D_{\text{Chen}} = \frac{1}{2N}\sum_{i=1}^{N}\sum_{l=1}^{L} y_i\left(\frac{1}{L}H(\mathbf{x}_i) - \alpha_l h_l(\mathbf{x}_i)\right). \tag{2.58}$$

For clarity, we will refer to this measure as "Ambiguity (Chen)". Tsymbal et al. [85] define an alternative ambiguity measure based on applying a squared error function to classification problems:

$$D_{\text{amb}} = \frac{1}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}(\delta[h_l(\mathbf{x}_i) = 1] - \widehat{Pr}(y = 1|\mathbf{x}_i))^2 + (\delta[h_l(\mathbf{x}_i) = -1] - \widehat{Pr}(y = -1|\mathbf{x}_i))^2, \quad (2.59)$$

which we refer to as "Ambiguity (Tsymbal)". Zenobi [95] uses an ambiguity function based on 0/1 cost, defined as:

$$D_{\text{Zenobi}} = \frac{1}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}\delta[h_l(\mathbf{x}_i) = H(\mathbf{x}_i)]. \quad (2.60)$$

Melville et al. [63] measure diversity based on how often base learners disagree with the ensemble prediction; this is the same as D_{Zenobi} from Equation 2.60 except that it measures inequality, rather than equality.

Brown and Kuncheva [15] derive a diversity measure based on a decomposition of classification error, which we will call "Ambiguity (Brown)":

$$D_{\text{Brown}} = -\tfrac{1}{2NL}\textstyle\sum_{i=1}^{N} y_i H(\mathbf{x}_i)\sum_{l=1}^{L}(1 - h_l(\mathbf{x}_i)H(\mathbf{x}_i)). \quad (2.61)$$

where $y_i H(\mathbf{x}_i)$ — the correctness of the ensemble output — determines whether 'diversity' is good or bad.

Information Theoretic Measures

Brown [14] gives a decomposition of ensemble mutual information which could be interpreted as a description of ensemble diversity:

$$I(\{h_1 \dots h_L\}; Y) = \sum_{l=1}^{L} I(h_l, Y) + \sum_{\substack{\{\mathbf{h}\}\subseteq\{h_1\dots h_L\} \\ |\{\mathbf{h}\}|=2\dots L}} \Big(- I(\{\mathbf{h}\}) + I(\{\mathbf{h}\}|Y)\Big). \quad (2.62)$$

The term on the left, $I(\{h_1 \dots h_L\}; Y)$, is the mutual information between base learner outputs and target labels. Using Fano's inequality [25] and a bound of Hellman & Raviv [36], it can be shown that low conditional entropy (and therefore, high mutual information) minimises upper and lower bounds on the

error of the optimal combiner.

The three terms in the decomposition are described as *relevancy*, *redundancy* and *conditional redundancy*. Relevancy is most closely related to individual accuracy. The other two terms collectively describe how individual base learners should differ from each other; redundancy captures mutual information between subsets of base learners (unlike pairwise diversity measures, redundancy measures mutual information between subsets of *all* cardinalities). Conditional redundancy measures class-conditional differences between base learners; since this term is positive, Brown suggests that ensembles should aim for small within-class variance, but large between-class variance.

Zhou and Li [96] further refine Brown's interpretation to describe diversity with multi-information terms:

$$I(\{h_1 \dots h_L\}; Y) = \sum_{l=1}^{L} I(h_l; Y) + \mathcal{I}(\{h_1 \dots h_L\}|Y) - \mathcal{I}(\{h_1 \dots h_L\}).$$

These terms can, in a sense, be decomposed over the base learners; for example, $\mathcal{I}(\{h_1 \dots h_L\}) = \sum_{l=1}^{L} I(h_l; \{h_1 \dots h_{l-1}\})$. Zhou uses this decomposition to estimate redundancy and conditional redundancy in ensembles.

In Chapter 3, we will not discuss these information theoretic measures in detail. The quantity $I(\{h_1 \dots h_L\}; Y)$ is most significant when considering the *theoretically optimal combiner*; however, in this thesis we will deal only with *voting combiners*, and as such the mutual information over-estimates the amount of information that can be extracted from the base learners. We illustrate this in

	x_1	x_2	x_3	x_4
h_1	-1	-1	1	1
h_2	-1	1	-1	1
y	-1	1	1	-1
y'	-1	-1	-1	1

Table 2.1: Classifier outputs with label y, which is the XOR of h_1 and h_2, and label y' which is the AND of h_1 and h_2.

Table 2.1, which shows base learner outputs and target labels in an ensemble of two learners. In both cases, the mutual information between base learner outputs and the target label suggests that *the optimal combiner predicts y and y' with perfect accuracy*. However, when we consider the form of this 'optimal combiner', it is clear that it cannot be linear in the case of y, while it can be for y'.

Note that the 'optimal combiner' is not quite the same as an optimal *oracle* combiner; an oracle combiner may examine an example's features to determine which base learner (or base learners) to include in the majority vote. An *optimal* oracle would be one which always produces a correct prediction *as long as at least one base learner is correct.* However, the 'optimal combiner' must make a prediction based *only* on base learner outputs, but is not constrained to *agree* with any of the base learners.

We would expect the case where the optimal combiner *can* predict perfectly, but the optimal *linear* combiner cannot (as illustrated in Table 2.1) to be relatively common, since an optimal combiner could be a look-up table between then base learner outputs and the labels; such a combiner would predict perfectly so long as all differently-labelled examples can be distinguished based on the base learner predictions. We illustrate this in Figure 2.7, where the descriptive power of the information theoretic interpretation does not describe the observed behaviour of a voting ensemble[9]; the theoretical minimum error reached zero very quickly, while the linear combiner did not achieve 0 training error even for $L = 100$.

Because the mutual information measure makes an assumption about the combiner that is inappropriate (in the context of this thesis — voting ensembles), the diversity terms in the decompositions of mutual information likewise describe behaviour in ensembles with very general combiners. Of course, if we did want to consider diversity outside the domain of voting ensembles, this could be a natural measure for doing so.

2.3.3 Interpretations of Diversity Measures

In addition to Kuncheva's efforts in proposing, collecting and investigating diversity measures [49, 53–55], other researchers have made progress in *unifying* diversity measures, showing direct theoretical correspondences between them. Since one of our major contributions is of this nature, we consider our work to build directly on the results of Tang [84] and Saitta [75], who describe several diversity measures within common frameworks.

[9]For this illustration, we train Adaboost on the entire heart disease dataset. Since Adaboost combination weights are chosen in a forward search, we also train a linear SVM on the base learner outputs — this gives a result that is closer to the optimal linear combiner.

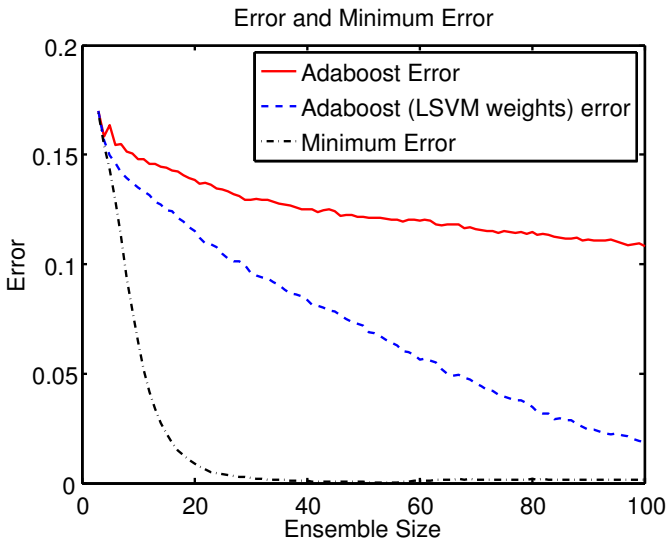

Figure 2.7: Error rate and theoretical minimum error rate for Adaboost ensembles (both using Adaboost combination weights, and a linear SVM combiner) on the Heart Disease dataset (270 training examples). The theoretical minimum error is the error obtained by a lookup table between base learner outputs and predictions — so errors only occur when two examples with opposite labels are predicted identically by all the base learners.

Tang's Diversity Framework

Tang's framework [84] is based on two properties of voting ensembles: the number of base learners that *incorrectly* predict each example (l_i in Tang's notation), and the average base learner accuracy (P in Tang's notation). In our notation, these two quantities are:

$$l_i = L(1 - c_i), \tag{2.63}$$

$$P = \overline{c} \tag{2.64}$$

respectively. We present Tang's results in our notation for consistency.

Tang derives expressions for 6 diversity measures: disagreement, double fault, KW variance, interrater agreement, generalised diversity and difficulty. He observes that all the measures include a term:

$$\sum_{i=1}^{N} (1 - c_i)^2, \tag{2.65}$$

which is associated with low diversity. Occurrences of $\overline{c}$ are less easily interpreted,

so Tang applies the assumption: *"regard the average classification accuracy ... of the base classifiers as a constant"*, meaning that he considers scenarios where c_i can vary but $\overline{c}$ must remain constant. Tang solves a constrained optimisation problem to find the minimum of $\sum_{i=1}^{N}(1-c_i)^2$ subject to the constraint implied by constant $\overline{c}$ and the definition of $\overline{c}$ in Equation 2.27. However, we prefer to understand this by considering Jensen's Inequality:

$$\Big(\frac{1}{N}\sum_{i=1}^{N}(1-c_i)\Big)^2 \leq \frac{1}{N}\sum_{i=1}^{N}(1-c_i)^2, \tag{2.66}$$

where we know that the left-hand side of the equation is constrained to be a constant $(1-\overline{c})^2$ and we wish to minimise the right-hand side. In Jensen's inequality, equality is obtained when all the terms have the same value.

Constant c_i with respect to i implies that all training examples are correctly classified by the same number of base learners. Tang calls this the *uniformity condition*, deriving the equality:

$$c_i = \overline{c} \quad \forall i. \tag{2.67}$$

Since the margin is a simple transformation of number of correct predictions, Tang is able to use the fact that the minimum of a sample is necessarily upper bounded by the average to relate the minimum margin and the average accuracy:

$$m_i = 2c_i - 1, \tag{2.68}$$

$$\min_i c_i \leq \overline{c}, \tag{2.69}$$

$$\min_i m_i \leq 2\overline{c} - 1, \tag{2.70}$$

$$\max \min_i m_i = 2\overline{c} - 1. \tag{2.71}$$

Combining this with the fact that the uniformity condition (Equation 2.67) must hold if the ensemble is maximally diverse, Tang states:

> *"If [average individual accuracy] is regarded as constant and the maximum diversity is achievable, maximising the diversity among the base classifiers is equivalent to maximising the minimum margin of the ensemble on the training samples."*

Tang interprets this as suggesting that maximising diversity can be viewed as an

implicit way to maximise the minimum margin of the ensemble. In this thesis we show our own margin-based interpretation of diversity; while will not dispute the correctness of Tang's result, we will give a view that does not rely on assumptions of constant individual accuracy and achievable maximum diversity.

Firstly, we emphasise that the *process* of maximising diversity is not necessarily related to that of maximising the minimum margin; Tang shows that the *maximum* values will occur in the same situation, but not that there is any monotonic relationship between the two.

Furthermore, although 'maximising the minimum margin' may sound tantalisingly like processes that occur in Adaboost or SVMs, since the maximum of a sample is *lower bounded* by the sample average, we can symmetrically argue that:

$$\max m_i \geq 2\overline{c} - 1, \tag{2.72}$$

and hence Tang's result equally implies that maximising diversity is related to finding the *minimum maximum margin*. From that perspective, high diversity seems less promising; and as discussed in Section 2.2.3, recent margin theory results suggest that generally increasing all margins (not just the minimum) is beneficial.

Saitta's Diversity Framework

Saitta [75] presents interpretations of various diversity measures based on properties of oracle output matrices: Q statistic, disagreement, double fault, entropy (Kuncheva), KW variance, inter-rater agreement, difficulty, generalised diversity and coincident failure diversity. Many of the interpretations are either analogous to those of Kuncheva or Tang, or derived by considering behaviour in certain scenarios. The view expressed by Saitta is that:

> *"not only no useful measure exists today, but it is unlikely that one will ever exist."*

The justification for this is that diversity has no monotonic relationship with performance, and that iterative construction of an ensemble based on diversity optimisation does not exhibit optimal substructure. To an extent, we agree that Saitta's work supports this negative conclusion, although of course the usefulness or otherwise of a diversity measure is contingent on how *'usefulness'* is defined.

The idea that diversity should be derived from an oracle matrix and be predictive of ensemble accuracy seems strange, since ensemble accuracy itself can be computed with an oracle matrix (Saitta mentions this in her conclusions). Both the work of Kuncheva that we have referenced previously, and our own developments in Chapter 3, seem to take views that are not especially opposed to that of Saitta, but rather disagree on what it would mean for diversity to be *useful*. Certainly, some of Kuncheva's proposals [49] can be seen to be satisfied by existing diversity measures to some extent; for example, visualising the behaviour of ensembles in kappa-error diagrams has provided an improved understanding of ensemble algorithms [74].

2.3.4 Using Diversity in Ensemble Algorithms

We now attend to some applications of diversity for the purpose of algorithm design. Kuncheva [49] describes two broad techniques for applying diversity to ensemble creation: 'overproduce & select', and 'building ensembles'. In 'overproduce and select' algorithms, many base learners are generated, and diversity is used to guide a selection process that defines the final ensemble. 'Building ensembles' refers to the idea that diversity could be used while creating base learners. We will describe a number of approaches from the literature that fit into one or other of these categories.

Pruned Adaboost

Margineantu and Dietterich [60] define several pruning algorithms that can be applied to Adaboost ensembles. Two of these — 'Kappa pruning' and 'Kappa-Error convex hull pruning', rely on the Kappa diversity measure to guide the pruning process.

The simplest procedure, 'Kappa pruning', computes pairwise Kappa for each of $\binom{L}{2}$ classifier pairs. It then performs a forward search to add the most diverse classifier pairs to the ensemble until the desired size is reached. This does not necessarily give the classifier subset that maximises average pairwise kappa, and also fails to account for the accuracy of individual classifiers.

'Kappa-Error convex hull pruning' addresses this second issue by computing the convex hull of the classifier pairs in a two dimensional space of pairwise kappa and average individual error. The final ensemble comprises only those classifiers

contained in pairs that fall on the convex hull. Kuncheva [49] suggests using the Pareto front as an alternative to the convex hull, since the convex hull could be very sparse, and in such cases the ensemble could vary dramatically based on slight noise in the estimates of D_κ and error.

Empirically, Margineantu and Dietterich found that Kappa pruning was competitive with other pruning methods based on estimating error on a hold-out set al.hough pruning rarely improved performance substantially beyond that of unpruned Adaboost, but did achieve smaller ensemble sizes.

GASEN

Genetic Algorithm based Selective ENsemble (GASEN), proposed by Zhou et al. [97] is an overproduce and select algorithm that uses a genetic algorithm to evolve base learner weights, before using those weights as a threshold to guide pruning.

GASEN relies on a decomposition of ensemble error based on error correlation:

$$e_{\text{tr}} = \sum_{j=1}^{L} \sum_{k=1}^{L} \alpha_j \alpha_k (h_j(\mathbf{x}) - y)(h_k(\mathbf{x}) - y) \tag{2.73}$$

This measure is estimated on hold-out training data, and used as a fitness function to guide a genetic algorithm. GASEN is designed around quadratic loss; in the case of 0/1 outputs from base learners and uniform weights, the term inside the summation reduces to double fault diversity.

Label Switching

Breiman [9] proposes promotion of diversity via 'output randomisation'. The idea is to build an ensemble of base learners that are trained on *all* the training data, but to add different random noise to target values prior to training each base learner. In the context of regression, this involves adding Gaussian distributed random noise to each output label. For classification problems, there were two approaches: 'Output Smearing' involves converting the problem to a multiple output regression problem, and adding Gaussian noise as in previous case. 'Output flipping' is an alternative approach based on flipping the target label with a certain probability. Breiman suggests that this should occur with probabilities that retain consistent class proportions, so prior probabilities $Pr(y)$ are estimated

from training data. A flip rate parameter is specified, f, where higher values will result in a greater likelihood of flipping labels.

For an example with true label y, the label is unchanged with probability $1 - \frac{f(1-Pr(y))}{1-\sum_{y' \in \mathcal{Y}} Pr(y')^2}$, and is changed to $k \neq y$ with probability:

$$\frac{fPr(k)}{1 - \sum_{y' \in \mathcal{Y}} Pr(y')^2}, \tag{2.74}$$

In the case of output flipping, f is chosen on validation data. Empirically, randomised outputs often outperform Bagging but not Adaboost.

Martínez-Muñoz and Suárez [61] extend Breiman's work by considering an alternative relabelling scheme. In their algorithm, the label is unchanged with probability $1 - f$, and changed to k with probability:

$$\frac{f}{|\mathcal{Y}| - 1} \tag{2.75}$$

where $|\mathcal{Y}|$ is the number of classes, such that there is no longer any consideration for the class priors. The rationale behind this adjustment is that the range of f can be increased; for 'convergence' (by which the authors mean that an example should receive its true label at least half of the time), it is necessary that:

$$f \quad < \quad \frac{|\mathcal{Y}| - 1}{|\mathcal{Y}|}, \tag{2.76}$$

while Breiman's transition probabilities require for convergence that f be less than the proportion of examples in the minority class.

These transition probabilities are further motivated by the kind of analysis that we described in Section 2.3.1: if the base learners are sufficiently strong (i.e. if they will exactly fit their training data), then it is possible to use label flipping to create truly independent training errors between base learners. This creates a scenario where we can be sure that training error will converge to zero.

DECORATE

Melville and Mooney [63] propose an algorithm for Diverse Ensemble Creation by Oppositional Relabelling of Artificial Training Examples (DECORATE). This is an iterative ensemble construction algorithm that uses artificial data to promote diversity. The algorithm generates artificial data from a Gaussian distribution

that approximates the input distribution. The data is labelled *oppositionally* — in the two-class case, if the ensemble predicts y with probability $\widehat{Pr}(y|\mathbf{x})$, then the next learner is trained on $\langle \mathbf{x}, y \rangle$ with probability $1 - \widehat{Pr}(y|\mathbf{x})$ and $\langle \mathbf{x}, -y \rangle$ with probability $\widehat{Pr}(y|\mathbf{x})$.

This data is used in addition to the original training data, so each base learner is trained with a different set of oppositional artificial data, fostering diversity. The oppositional nature of the relabelling procedure encourages base learners to disagree with the rest of the ensemble. Empirically, DECORATE is competitive with Bagging and Adaboost, and especially effective when there is very little training data [63].

2.3.5 Summary

We have introduced the idea of diversity, presented motivation from the literature for quantifying and studying it, and enumerated 16 existing measures. We then highlighted some previous interpretations of diversity measures in unifying frameworks, and finally surveyed some algorithms that are designed to exploit ensemble diversity.

Chapter 3

Interpreting Diversity using Voting Margins

3.1 Introduction

The primary contribution of this chapter is to establish a link between diversity and voting margins. In Chapter 2, we introduced many diversity measures from the literature, describing where and why they have been previously used. Section 3.2 shows direct theoretical relationships between 15 of these measures and expressions based on the margin distribution. We also consider measures that cannot be described precisely via the margin distribution, specifically examining their use in proving theoretical properties of diversity. We discuss the support and utility of the margin theory framework via analysis of previous experiments and two additional empirical investigations. One of these examines the influence of diversity on generalisation error, where the hypothesis arising from the conventional view of diversity contradicts the hypothesis that we arrive at from the margin theory perspective. Our second experiment considers the relationship between a quadratic loss function, based on the double fault diversity measure, and an exponential loss function as is used in Adaboost, showing how a Taylor approximation to exponential loss gives rise to a set of interpolating loss functions; we investigate the convergence of ensembles trained on successively closer approximations to exponential loss.

Understanding diversity from the perspective of margin theory is valuable because:

1. The rich margin theory literature enables us to make predictions about how

diversity affects ensemble behaviour.

2. There are known techniques for *optimising* functions of the voting margin; by implication, these techniques are also affecting *diversity*.

3. Our analysis reveals further redundancies and relationships in the plethora of existing diversity measures.

3.2 Interpreting Diversity with Voting Margins

In Section 2.3, we introduced various measures of ensemble diversity. Now we analyse each measure in turn. In the majority of cases, we show that the measures correspond directly to some property of the margin distribution.

3.2.1 Related to the Absolute Margin

Here we show how entropy (Kuncheva), ambiguity (Zenobi) and diversity (Melville) all measure essentially the same quantity, which is based on the absolute value of the voting margin. We also examine ambiguity (Brown) and ambiguity (Chen), showing their relationship with the absolute margin.

Theorem 1. *Entropy (Kuncheva) [54] is a function of the absolute margin. Original Form:*

$$D_{ent} = \frac{L}{N(L - \lceil L/2 \rceil)} \sum_{i=1}^{N} \min\{c_i, 1 - c_i\}. \tag{3.1}$$

Margin Interpretation:

$$D_{ent} = \frac{L}{(L-1)}(1 - \overline{|m|}). \tag{3.2}$$

Proof. In Appendix B.2.1. □

Note that the coefficient is constant for a fixed ensemble size (L), and tends towards 1 for large ensembles.

Theorem 2. *Ambiguity (Zenobi) [95] is a function of the absolute margin.*
Original Form:

$$D_{Zenobi} = \frac{1}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}\delta[h_l(\mathbf{x}_i) = H(\mathbf{x}_i)]. \tag{3.3}$$

Margin Interpretation:

$$D_{Zenobi} \ = \tfrac{1}{2}(1 - \overline{|m|}). \tag{3.4}$$

Proof. In Appendix B.2.2. □

Due to the similarity in the definition of diversity (Melville) [63] (recall that it uses inequality in its definition, where D_{Zenobi} uses equality), the relationship for diversity (Melville) follows analogously.

Theorem 3. *The two diversity terms described by Brown [15] and Chen [16] are identical, and they are asymmetric versions of the absolute margin diversity measures.*
Original Forms:

$$D_{Brown} = -\frac{1}{2NL}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)\sum_{l=1}^{L}(1 - h_l(\mathbf{x}_i)H(\mathbf{x}_i)), \tag{3.5}$$

$$D_{Chen} = \frac{1}{2N}\sum_{i=1}^{N}\sum_{l=1}^{L} y_i(\frac{1}{L}H(\mathbf{x}_i) - \alpha_l h_l(\mathbf{x}_i)). \tag{3.6}$$

Margin Interpretation:

$$D_{Chen} = -D_{Brown} = \frac{1}{2N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)(1 - |m_i|). \tag{3.7}$$

Proof. In Appendix B.2.3. □

These measures contain the same absolute margin term as other diversity measures, but also a $y_i H(\mathbf{x}_i)$ term. This is a result of fitting this diversity term into a decomposition of ensemble error; the sign of $y_i H(\mathbf{x}_i)$ determines whether high diversity on $\mathbf{x}_i$ is beneficial or detrimental. In this sense, we could see the $(1 - |m_i|)$ term describes 'diversity', while $y_i H(\mathbf{x}_i)$ describes the effect that diversity will have on ensemble error.

Chen does consider this decomposition of his ambiguity measure [16, Section 3.3], and directly notes that smaller $|m_i|$ corresponds to higher diversity.

3.2.2 Related to the Squared Margin

Next, we show several diversity measures that relate to the squared margin: KW variance, disagreement, ambiguity (Tsymbal), entropy (Cunningham), and double fault. These are also closely related to the measures from the previous section, since $|m| = \sqrt{m^2}$.

Theorem 4. *KW variance [43] is a function of the squared margin.*
Original Form:

$$D_{KW} = \frac{1}{2N}\sum_{i=1}^{N}(1 - \widehat{Pr}(y = -1|\mathbf{x}_i)^2 - \widehat{Pr}(y = 1|\mathbf{x}_i)^2). \tag{3.8}$$

Margin Interpretation:

$$D_{KW} = \frac{1}{4}(1 - \overline{m^2}) \tag{3.9}$$

Proof. In Appendix B.2.4. □

From here the known relationship with disagreement [81] gives:

$$D_{\text{dis}} = \frac{2L}{L-1}D_{\text{KW}}, \tag{3.10}$$

$$= \frac{L}{2(L-1)}(1 - \overline{m^2}), \tag{3.11}$$

which extends to 'Hamann's measure' [29] by the simple transformation mentioned in the background section.

Theorem 5. *Ambiguity (Tsymbal) [85] is a function of the squared margin.*
Original Form:

$$D_{amb} = \frac{1}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}(\delta[h_l(\mathbf{x}_i) = 1] - \widehat{Pr}(y = 1|\mathbf{x}_i))^2 + (\delta[h_l(\mathbf{x}_i) = -1] - \widehat{Pr}(y = -1|\mathbf{x}_i))^2. \tag{3.12}$$

Margin Interpretation:

$$D_{amb} = \frac{1}{2}(1 - \overline{m^2}). \tag{3.13}$$

Proof. In Appendix B.2.5. □

Entropy (Cunningham) is a more complex measure to interpret — it can be expressed precisely in terms of the margin distribution, but the interpretation is not easy to relate to other measures. However, using a Taylor expansion to express the log terms ($\log x = (x-1) + \frac{(x-1)^2}{2}$, to second order) shows that entropy does closely approximate other $\overline{m^2}$ measures.

Theorem 6. *Entropy (Cunningham) [18] is, to second-order Taylor approximation, a function of the squared margin.*
Original Form:

$$\begin{aligned} D_H = -\frac{1}{N}\sum_{i=1}^{N} &\widehat{Pr}(y=-1|\mathbf{x}_i)\log\widehat{Pr}(y=-1|\mathbf{x}_i) \\ &+ \widehat{Pr}(y=1|\mathbf{x}_i)\log\widehat{Pr}(y=1|\mathbf{x}_i) \end{aligned} \tag{3.14}$$

Margin Interpretation:

$$D_H \approx \frac{5}{8}(1 - \overline{m^2}) \tag{3.15}$$

Proof. In Appendix B.2.6. □

This approximation, in addition to showing a clear link between entropy and voting margins, also explains an observation of Chen [16, page 46]:

> "the high correlation between entropy and disagreement measure is somewhat surprising and we currently do not know how to explain this"

We make a novel contribution here by showing the second-order Taylor approximation of entropy to disagreement, which explains why entropy behaves very similarly (but not the same as) disagreement and KW variance. We show this relationship in Figure 3.1.

Finally, we examine double fault diversity, which is an asymmetric measure, unlike the previous three. Therefore, its margin-based interpretation does not have such a close resemblance to the previous measures.

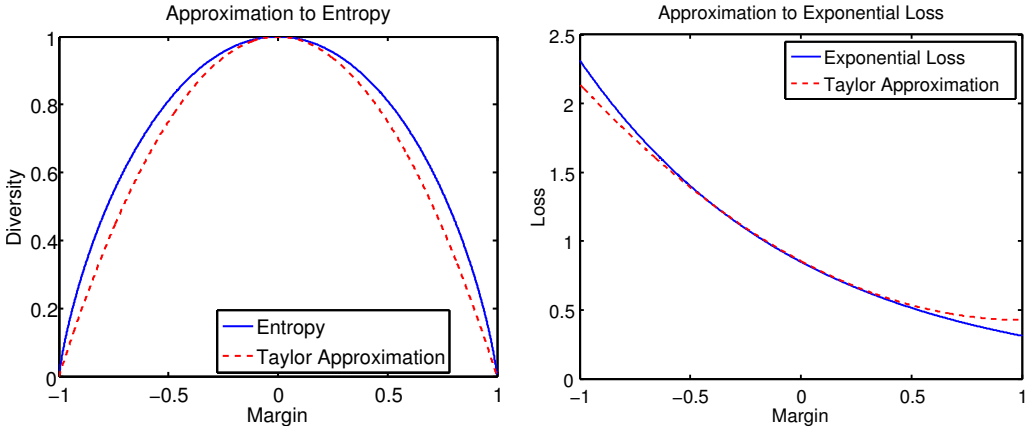

Figure 3.1: We illustrate the second-order Taylor approximations for entropy and exponential loss. In both cases, we see links with diversity measures; in the first case, entropy is approximated by KW variance (scaled such that 1 is the maximum value). For exponential loss, the Taylor approximation is very similar to the double fault diversity measure (scaled to give the same area under the curve).

Theorem 7. *Double fault diversity [30] is a quadratic function of the margin. Original Form:*

$$D_{DF} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N_{j,k}^{00}}{N}. \tag{3.16}$$

Margin Interpretation:

$$D_{DF} = \frac{1}{2}(1 - \overline{m}) - \frac{L}{4(L-1)}(1 - \overline{m^2}). \tag{3.17}$$

Proof. In Appendix B.2.7. □

The result contains two terms: $(1-\overline{m})$ which favours accurate ensembles, and $(1-\overline{m^2})$ which captures the same notion of diversity as KW variance, ambiguity (Tsymbal) and entropy (Cunningham). This first term is what gives double fault its asymmetry. Also note that, although double fault is generally considered a 'pairwise' diversity measure, this reinterpretation considers the margins of the whole ensemble, rather than the behaviour of classifier pairs.

We can show another interesting property of double fault diversity by using a Taylor approximation. Taking the exponential loss function used by Adaboost,

$C_{\text{ada}} = \sum_{i=1}^{N} e^{-m_i}$, and computing the second-order Taylor approximation, gives:

$$C_{\text{ada}} = \sum_{i=1}^{N} e^{-m_i}, \tag{3.18}$$

$$\approx (1 - \overline{m}) + \frac{1}{2}\overline{m^2}, \tag{3.19}$$

$$D_{\text{DF}} = (1 - \overline{m}) + \frac{1}{2}\overline{m^2} - \frac{1}{2}, \tag{3.20}$$

when L is sufficiently large that $\frac{L}{L-1} \approx 1$. This illustrates the close link between double fault diversity and a Taylor approximation to exponential loss; they are scaled and translated versions of one another. We show the approximation in Figure 3.1. We investigate this relationship experimentally in Section 3.4.2.

3.2.3 Related to Other Functions of the Margin Distribution

Some of the other diversity measures have more complex interpretations that are nevertheless expressible in terms of the margin distribution. Difficulty is the variance in the proportion of learners that predict the correct class; in fact, this is almost the variance of the margin distribution:

Theorem 8. *Difficulty [35] is closely related to the variance of the margin distribution.*
Original Form:

$$D_{diff} = Var_{Pr(\mathbf{x},y)}[c(\mathbf{x}, y)]. \tag{3.21}$$

Margin Interpretation:

$$D_{diff} = \frac{1}{4}(\overline{m^2} - \overline{m}^2). \tag{3.22}$$

Proof. In Appendix B.2.8. □

Difficulty measures only the variance of the margin distribution, so it does not have a minimum at $m = 0$ as the absolute or squared margin measures do. However, if individual base learner accuracy is fixed, then the only changes to difficulty come from the $\overline{m^2}$ term, and in this sense it agrees with squared margin measures.

Theorem 9. *Non-pairwise Interrater Agreement [26] can be described as a function of the margins.*
Original Form:

$$D_\kappa = 1 - \frac{L}{N(L-1)} \sum_{i=1}^{N} \frac{c_i(1-c_i)}{\overline{c}(1-\overline{c})}. \tag{3.23}$$

Margin Interpretation:

$$D_\kappa = 1 - \frac{L}{L-1}\Big(\frac{1-\overline{m^2}}{1-\overline{m}^2}\Big). \tag{3.24}$$

Proof. In Appendix B.2.9. □

As with difficulty, interrater agreement reduces to the squared margin when individual accuracy is fixed.

Generalised diversity is similar to difficulty and interrater agreement, but exhibits some explicit asymmetry.

Theorem 10. *Generalised Diversity [48] can be described as a function of the margins.*
Original Form:

$$D_{GD} = 1 - \frac{\sum_{l=1}^{L} l(l-1)Pr(c_i = \frac{L-l}{L})}{\sum_{l=1}^{L} l(L-1)Pr(c_i = \frac{L-l}{L})}. \tag{3.25}$$

Margin Interpretation:

$$D_{GD} = \frac{L}{L-1}\Big(\frac{1-\overline{m^2}}{2(1-\overline{m})}\Big). \tag{3.26}$$

Proof. In Appendix B.2.10. □

This interpretation shows that generalised diversity is related to other squared margin measures, but that it additionally considers the average margin, which will quantify the accuracy of the ensemble to an extent.

Theorem 11. *Coincident Failure Diversity [48] can be described as a function of the margins.*

Original Form:

$$D_{CFD} = \begin{cases} 0 & \text{if } Pr(c_i = 1) = 1 \\ \frac{\sum_{l=1}^{L}(L-l)Pr(c_i=\frac{L-l}{L})}{(L-1)(1-Pr(c_i=1))} & \text{otherwise} \end{cases} \tag{3.27}$$

Margin Interpretation:

$$D_{CFD} = \frac{L}{L-1}\Big(1 - \frac{1-\overline{m}}{2(1-\frac{1}{N}\sum_{i=1}^{N}\delta[m_i=1])}\Big). \tag{3.28}$$

Proof. In Appendix B.2.11. □

As discussed in the appendix, we have omitted the $Pr(c_i = 1)$ special case for CFD due to the necessary condition (i.e. all base learners always make correct predictions)[1]; the equation we have given here is undefined in such a scenario. The general pattern of CFD is asymmetric, with one component for the average margin, and one for the number of perfectly predicted examples.

3.2.4 Discussion of Terms in the Margin Framework.

So our interpretations have shown that most diversity measures can be described using various properties of the margin distribution. We now give some attention to what each of the terms involved mean.

$\overline{|m|} \in [0, 1]$ is the average absolute margin. This quantifies the average difference in support for the two classes, and is symmetric with respect to the true label.

$\overline{m^2} \in [0, 1]$ is the average squared margin. Its meaning is similar to $\overline{|m|}$, but squaring the margins increases the influence of outliers (i.e. margins close to -1 or 1). Jensen's Inequality describes a relationship between absolute and squared margins: $\overline{|m|} \leq \sqrt{\overline{m^2}}$, with equality when $\forall i, j : |m_i| = |m_j|$.

$\overline{m} \in [-1, 1]$ is the average margin. This is a simple transformation of the average base learner accuracy. In practice we can often assume that $\overline{m} > 0$ (although this does not have to be the case). $\overline{m}$ is asymmetric with respect to the true label, so we should expect diversity measures that contain $\overline{m}$ to have a stronger relationship with ensemble accuracy.

[1]Since $Pr(c_i = 1)$ can be expressed using margins, our omission here does not indicate that coincident failure diversity cannot be perfectly expressed using the margins; we left the special case out for clarity, and because it will only occur in an ensemble where base learners never make errors.

$\overline{m}^2 \in [0, 1]$ is the squared average margin. This term is interesting as it is symmetric in a sense (if we generated completely random ensembles, $\overline{m}^2$ would have no relationship to accuracy), but in trained ensembles where $\overline{m} > 0$ holds, squaring $\overline{m}^2$ will not change the sign of $\overline{m}$, and therefore it acts as an asymmetric measure. As above, Jensen's Inequality implies a relationship: $\overline{m}^2 \leq \overline{m^2}$. These two terms are also related as an empirical estimate of the variance of the margin distribution: $\text{Var}[m] = \overline{m^2} - \overline{m}^2$.

3.2.5 Agreement with Other Empirical Results

Our theoretical interpretations have suggested that many diversity measures can be expressed simply using either $\overline{|m|}$ or $\overline{m^2}$. A simple way to verify our conclusions is to examine the empirical results from the literature, where authors have analysed correlations between various diversity measures.

Kuncheva produces tables of rank correlation coefficients for 10 measures, based on results from ensembles of linear and quadratic classifiers generated with varying feature subsets on the UCI Breast Cancer dataset [54, Section 6.4].

KW Variance, Disagreement and Entropy (Kuncheva)

Kuncheva's experiments show that KW variance and disagreement are identical, and often behave the same as entropy (Kuncheva). This agrees with the known theoretical connection between KW variance and disagreement [54]. With regards to entropy, for the first set of experiments, the ensemble size $L = 3$ would prevent linear correlation from making a distinction between $\overline{|m|}$ and $\overline{m^2}$, so the 100% correlation in Kuncheva's Table 7 agrees with our analysis. Kuncheva's subsequent clustering experiments always assign entropy to the same group as KW variance and disagreement; however, her discussion (*"... equivalence between KW and Dis, and the similarity to E..."*) supports the idea that entropy behaves similarly, but not identically.

Q Statistic, Phi Coefficient, Pairwise Kappa Coefficient, Generalised Diversity

These measures are shown to be related, but not to the same extent as those in the previous section. For $L = 3$, there are 4 possible margins and 6 possible classifier pairs; case by case analysis gives each pairwise count as a function of

counts of examples with specific margins (for example $N^{11} = 6|\{i : m_i = 1\}| + 2|\{i : m_i = \frac{2}{3}\}|$). In this situation, the denominator in D_k is the square of half the denominator in D_ϕ; this close similarity agrees with the strong empirical correlation between the two.

These measures have empirical similarity to generalised diversity and the 3 measures we discussed previously. Pairwise and non-pairwise kappa should be expected to exhibit a close relationship (although non-pairwise kappa is not included in the experiments); given this link, we should expect measures that are related to pairwise kappa (D_{Q} and D_ϕ) to be slightly similar to measures that are based on the squared margin.

Difficulty, Double Fault

Kuncheva's experiments on difficulty are especially interesting. In the first set of experiments, difficulty correlates well with double fault but poorly with everything else. In the second experiments, difficulty is included in the cluster with all the measures discussed above. For double fault, $\overline{m}$ is combined with $\overline{m^2}$ at a ratio of 2 : 1, meaning that the asymmetric component ($\overline{m}$) is most significant. Difficulty combines $\overline{m}^2$ and $\overline{m^2}$ at parity, so although $\overline{m}^2$ still provides some information about average base learner accuracy (assuming $\overline{m} > 0$), the symmetric information associated with $\overline{m^2}$ will have far more influence than it does in double fault. But considering how much *more* significant the asymmetric component in double fault is, we can see that the size of $\overline{m}$ will have some role in determining this since that term is *squared* in difficulty. So we expect difficulty to be *more like double fault* when $\overline{m}$ is large, and *more like squared-margin measures* when $\overline{m}$ is close to 0.

This seems to fit with the experimental results, assuming the stronger learners (and ease of achieving good performance on the breast cancer dataset) in the first experiment cause $\overline{m}$ to be large; in the second experiment, some ensembles are trained with random weak learners, which would suggest that $\overline{m}$ will be far smaller. If these assumptions about $\overline{m}$ are true, then the experiments agree with our interpretation of difficulty and double fault diversity.

Correlation with accuracy

Kuncheva found that *"In general, the relationship between accuracy and diversity was strongest for the DF and CFD measures with the majority vote accuracy".*

These two measures are both strongly asymmetric; they both contain a $\overline{m}$ term that gives them a relationship with average base learner accuracy; so there is strong corroboration for our interpretation, since the measures that are most predictive of accuracy are also most closely related to the average margin.

Chen's Experiments with Generalisation Accuracy

Chen's experiments [16, Section 3.4] examined correlations between various diversity measures and generalisation error, using Bagging ensembles of CART learners. Like Kuncheva, Chen found that D_{KW} and D_{dis} are identical; Chen measured entropy as defined by Cunningham, and found that it was also very similar to KW variance, which we have shown with a theoretical link via a Taylor approximation. D_k, D_ϕ and D_{Q} were very similar to each other, and quite closely correlated with D_{dis}, D_{KW} and D_H, as was generalised diversity. Difficulty was not strongly correlated with any other measure (double fault diversity was not investigated here), and neither was coincident failure diversity. This agrees with our experience of difficulty in high average base learner accuracy situations (CART is a relatively strong base learner). Ambiguity (Chen) was shown to be most strongly correlated with generalisation accuracy; this would agree with our interpretation showing that ambiguity is the only measure to explicitly contain $y_i H(\mathbf{x}_i)$, which essentially quantifies the training accuracy. The other measures that Chen finds are most closely related to generalisation accuracy are also asymmetric: generalised diversity, coincident fault diversity and difficulty.

Kapp et al. Empirical Study of Diversity and Voting Margins

Kapp et al. [40] investigate the relationship between diversity measures and voting margins, as well as training and generalisation accuracy. They find that some measures (Q, disagreement, ambiguity (Zenobi), KW variance) were 'weakly' related to individual and ensemble accuracy, while other measures (generalised diversity, difficulty, double fault) were 'strongly' related. These two groups correspond to symmetric and asymmetric measures respectively.

Our result in Section 3.2.2 showed that there is a similarity between double fault diversity and exponential loss. Kapp et al. observe:

> *"Maybe Double-Fault diversity has produced a stable behaviour because if strong classifiers are available (high average margin), this measure*

seeks to decrease the probability of identical errors"

as an explanation for the good relationship between double fault diversity and ensemble accuracy; this observation fits well with the success of exponential loss as an optimisation criterion.

Kapp et al. also find a measure that they describe as 'CI-measure' to be successful; below (in Equation 3.41), we show that this measure (a version of the Chebyshev inequality on the margin distribution) is equivalent to the non-pairwise kappa coefficient.

3.3 Relationship with Existing Theoretical Results

In this section, we address some existing theoretical results that are related to diversity, and show how they can be viewed within the margin theory framework.

3.3.1 Breiman's Random Forest Result

This result is important because it contains D_ϕ — the correlation between base learners. This has been proposed as a diversity measure, but does not have a direct interpretation from the margin theory perspective. Therefore, we should examine the use of D_ϕ here to determine whether the same result can be derived using only margin theory, or whether measuring diversity with D_ϕ does allow us to learn new things about ensemble generalisation error.

Breiman motivates the diversity-inducing aspects of Random Forests with an upper bound on the generalisation error of independent ensembles. The bound describes the generalisation error of an ensemble of independent base learners in terms of the average margin and the correlation between learners.

$$e_{\text{gen}} \leq \frac{\overline{\rho}(1-s^2)}{s^2}, \tag{3.29}$$

(see Section 2.3.1 for definitions of $\overline{\rho}$ and s). Because diversity measures are defined with respect to *training data*, it is simplest to translate Breiman's bound

into estimates of all the variables from training data [2]:

$$e_{\text{tr}} \leq \frac{D_\phi(1-\overline{m}^2)}{\overline{m}^2}, \tag{3.30}$$

which suggests that good performance can be achieved by reducing correlation, or increasing the margin. This is useful as it bounds the error of the ensemble with the correlation (which we can control to some extent while maintaining base learner independence) and margin (which can be optimised independently by the individual base learners).

The bound is derived from Chebyshev's inequality on the margin distribution:

$$Pr(|m - E[m]| < k\sqrt{\text{Var}[m]}) \leq \frac{1}{k^2}, \tag{3.31}$$

$$Pr(m < E[m] + k\sqrt{\text{Var}[m]}) \leq \frac{1}{k^2}, \qquad \text{let } k = \frac{-E[m]}{\sqrt{\text{Var}[m]}} \tag{3.32}$$

$$Pr(m < 0) \leq \frac{\text{Var}[m]}{E[m]}, \tag{3.33}$$

where all the expectations and variance are w.r.t.the distribution $Pr(\mathbf{x}, y)$.

The additional steps of Breiman's derivation use the independence between base learners to bound $\text{Var}[m]$ with $\overline{\rho}$. Since learners are independent, we can consider each base learner to have been drawn from a distribution $Pr(h)$. The 'strength' of an individual learner is closely related to its generalisation accuracy: $s(h) = E_{Pr(\mathbf{x},y)}[yh(\mathbf{x})]$, and the expectation of this over all base learners gives the expected margin: $E_{Pr(h)}[s(h)] = E_{Pr(\mathbf{x},y)}[m(\mathbf{x}, y)]$. Therefore:

$$\text{Var}[m] \leq \overline{\rho}\text{Var}_{Pr(h)}[s(h)], \tag{3.34}$$

$$= \overline{\rho}(E_{Pr(h)}[s(h)^2] - E_{Pr(h)}[s(h)]^2), \tag{3.35}$$

$$\leq \overline{\rho}(1 - E[m]^2), \tag{3.36}$$

which is combined with Equation 3.33 to give the final bound from Equation 3.29.

The beauty of this result is that correlation and *average* margin can both be optimised *for independent base learners*, while directly optimising the *variance*

[2] Breiman's statement that *"the bound ... fulfills the same suggestive function for random forests as VC-type bounds do for other types of classifiers"* is often quoted, and perhaps misinterpreted; the bound does not show a relationship between generalisation error and quantities that can be measured on training data. Breiman's implication seems to have been that it describes the error as a trade-off — in this case between correlation and individual accuracy.

of the margin would have required dependence between the learners. However, the result is often interpreted as implying that optimising diversity is *crucial* for ensemble learning; for example, Gatnar states [29, Section 3]:

> "The conclusion from the above equations [including Equation 3.29] is obvious: the stronger the correlation between members of the ensemble, the higher the ensemble classification error."

However, by examining the origin of Breiman's bound, we have seen that the relationship between correlation and error is as part of a bound on the variance of the margins which is itself part of a bound on the error.

For our research, the most important issue regarding Breiman's Random Forest bound is whether itit ascribes special significance to the D_ϕ diversity measure that could not equally apply to other diversity measures. If it does, then this would suggest that taking a non-margin theory view of diversity could permit a better understanding of diversity in ensembles of independent learners.

Since Breiman's bound is based on the initial application of Chebyshev's inequality on the margin distribution, we can actually see that the margin theory explanation of diversity allows us to draw the same conclusions from Equation 3.33, for example:

$$e_{\text{tr}} \leq \frac{4D_{\text{diff}}}{L^2\overline{m}^2}, \tag{3.37}$$

or even, with some manipulations, the (non-pairwise) kappa coefficient (we approximate for large $L : \frac{L}{L-1} \approx 1$):

$$D_\kappa = 1 - \frac{1 - \overline{m^2} - \text{Var}[m] + \text{Var}[m]}{1 - \overline{m}^2}, \tag{3.38}$$

$$= 1 - \frac{1 - \text{Var}[m] - \overline{m}^2}{1 - \overline{m}^2}, \tag{3.39}$$

$$= \frac{\text{Var}[m]}{1 - \overline{m}^2}, \tag{3.40}$$

$$e_{\text{tr}} \leq D_\kappa. \tag{3.41}$$

(Equation 3.41 may seem surprising since D_κ is symmetric with respect to accuracy, but the bound requires $\overline{m} > 0$, which is where the asymmetry necessary to upper bound error comes from). We arrive at similar expressions for other

measures that include $\overline{m^2}$:

$$e_{\text{tr}} \leq 1 + \frac{2D_{\text{KW}} - 1}{2(1 - \overline{m^2})}, \tag{3.42}$$

$$e_{\text{tr}} \leq 1 - \frac{2D_{\text{GD}}}{1 + \overline{m}}. \tag{3.43}$$

So, these results show that various diversity measures can form the 'diversity' component of a bound on generalisation error; in fact, they give an even tighter bound than D_ϕ, since the inequality from Equation 3.36 ($E_{Pr(h)}[s(h)^2] \leq 1$) is not required.

To summarise, we have shown the origins of Breiman's bound on generalisation error, which suggests that D_ϕ (which cannot be expressed using the margin distribution) has a unique application in ensemble theory. However, by examining the bound in more detail, we showed that it was derived from an instantiation of Chebyshev's inequality, which itself is a bound on generalisation error in terms of the margin distribution. Furthermore, we illustrated how several diversity measures could be inserted into Chebyshev's inequality to express a tighter version of Breiman's bound without requiring D_ϕ.

3.3.2 Kuncheva's Q Statistic Result

The Q statistic is another significant diversity measure that cannot be expressed directly in terms of the margin distribution. Furthermore, it is one of the most widely used measures, and was specifically recommended by Kuncheva following an empirical study of diversity measures [54].

In this section, we will study Kuncheva's bounds on majority vote accuracy in terms of D_{Q}. Since these results concern *training* accuracy, it may seem foolish to consider the margin distribution; of course the training accuracy is exactly the proportion of the margin distribution that lies above 0. However, the utility of these bounds is that they show that *for a constant individual base learner accuracy, how to distribute votes so as to achieve maximal training accuracy.*

The essential idea in Kuncheva's analysis is that of "Patterns of Success" and "Patterns of Failure". These describe voting patterns that achieve maximum (or minimum) training error[3] assuming all base learners achieve the same individual

[3]For consistency, we have converted Kuncheva's results to describe error rates, rather than accuracy. We also assume L is odd, which is a reasonable assumption in unweighted voting ensembles.

error. The results show that two error values, e_{succ} and e_{fail} can be defined in terms of D_{Q}, under the success/failure configurations.

$$e_{\text{succ}} = \frac{2L}{(L+1)} \frac{1}{(2-D_{\text{Q}})} - \frac{L-1}{L+1}, \tag{3.44}$$

$$e_{\text{fail}} = \frac{2L}{(L-1)} \frac{(1-D_{\text{Q}})}{(2-D_{\text{Q}})}. \tag{3.45}$$

Significantly, it seems that small D_{Q} (high diversity) is beneficial in the pattern of success, but detrimental in the pattern of failure.

The "Pattern of Success" actually describes a family of margin distributions. These distributions are zero everywhere except for $m = -1$ and $m = \frac{1}{L}$, which are populated based on the accuracy of individual base learners. This distribution matches the optimal distribution for the D_{Brown} function introduced by Kuncheva and Brown [15] — if $y_i H(\mathbf{x}_i)$ is negative (i.e. a misclassification), then the margin should be -1, if $y_i H(\mathbf{x}_i)$ is positive, the margin should be as small as possible. This pattern is the result of the idea that diverse ensembles should achieve the highest ensemble accuracy possible for a fixed individual accuracy.

The derivation of the result has two main components: firstly, a description of e_{succ} and e_{fail} in terms of individual accuracy ($\overline{c}$) and L:

$$e_{\text{succ}} = 1 - \frac{L\overline{c}}{(L+1)}, \tag{3.46}$$

$$e_{\text{fail}} = \frac{2L(1-\overline{c})}{(L+1)}. \tag{3.47}$$

The other component of the derivation is an expression for D_{Q} in terms of $\overline{c}$:

$$D_{\text{Q}}^{\text{succ}} = \frac{1-2\overline{c}}{1-\overline{c}}, \tag{3.48}$$

$$D_{\text{Q}}^{\text{fail}} = \frac{2\overline{c}-1}{\overline{c}}. \tag{3.49}$$

The expressions for e and D_{Q} are finally combined to give the results from Equation 3.45.

The relationship between this bound and the margin distribution can be seen by examining Equations 3.47 and 3.49 individually. Both of these can be rewritten to give $e_{\text{succ}}, e_{\text{fail}}, D_{\text{Q}}^{\text{succ}}$ and $D_{\text{Q}}^{\text{fail}}$ in terms of the average margin, simply using $\overline{c} = \frac{1}{2}(\overline{m}+1)$;

$$e_{\text{succ}} = 1 - \frac{L}{2(L+1)}(\overline{m} + 1), \tag{3.50}$$

$$e_{\text{fail}} = 1 - \frac{L}{L+1}(\overline{m} - 1), \tag{3.51}$$

$$D_{\text{Q}}^{\text{succ}} = \frac{2\overline{m}}{\overline{m} - 1}, \tag{3.52}$$

$$D_{\text{Q}}^{\text{fail}} = \frac{2\overline{m}}{\overline{m} + 1}. \tag{3.53}$$

These results imply that, with the assumptions that votes follow patterns of success/failure and that all learners have the same individual accuracy, the diversity and ensemble accuracy are both exactly determined by what the individual accuracy of the base learners is; i.e. both e and D_{Q} are functions of $\overline{c}$. So, the link between e and D_{Q} does not provide evidence that D_{Q} has some special significance in determining training accuracy, since the same relationship can be described concisely using the margin distribution as in Equation 3.51; rather, we should see this as meaning that *whatever aspect of D_Q cannot be described by the margin distribution, is precluded by one or both of the assumptions introduced in order to derive Equations 3.45 and 3.44.*

3.3.3 Tang's Minimum Margin Result

Tang uses a common framework to express six of the diversity measures that we described earlier [84]. Although he doesn't specifically use the margin in his definitions, he does make the connection between c_i: the proportion of votes for the correct class on $\langle \mathbf{x}_i, y_i \rangle$, $\overline{c}$, the average accuracy of the ensemble, and the margin:

$$m_i = 2c_i - 1, \tag{3.54}$$

$$\overline{m} = 2\overline{c} - 1, \tag{3.55}$$

$$\min_i m_i \leq 2\overline{c} - 1. \tag{3.56}$$

From this, Tang concludes:

> "If [average accuracy] is regarded as a constant and the maximum diversity is achievable, maximising the diversity among the base classifiers is equivalent to maximising the minimum margin of the ensemble

on the training samples."

This can be seen because the maximum value of $\min_i m_i$ occurs when all the margins are the same, i.e. $\forall i : m_i = 2\overline{c} - 1$, and Tang's derived diversity measures are maximised when c_i is uniform over all i.

This result seems contrary to our own exposition from Section 3.2; we generally show that diversity is maximised when m_i is close to 0, which typically implies *reducing the margins to increase diversity* (when the margins are positive). The reason why Tang shows such a surprising result is the assumption under which it holds: $\overline{c}$ (equivalently, $\overline{m}$) is regarded as constant.

When considering the margin distribution, it becomes clear why constant $\overline{m}$ causes this behaviour: to maintain constant $\overline{m}$, any increase in the minimum margin must be 'balanced' by a corresponding decrease in the margin of another data point; hence, the minimum margin provides some measure of how uniform the margin distribution is around $\overline{m}$ (although not a very sensitive one). Similarly, diversity terms that are based on $\overline{m^2}$ provide an alternative measure of uniformity when $\overline{m}$ is fixed.

The 'equivalence' of maximising the minimum margin and maximising diversity is actually not very strong; even with the assumption of constant $\overline{m}$, the two quantities are not *monotonic* with respect to one another — so unless the optimal scenario (uniform m_i) is achievable, there is no necessary similarity between the maximisation of the quantities. Furthermore, the general situations in which the minimum margin is maximised are not well described by the assumption of constant $\overline{m}$; realistically $\overline{m}$ needs to vary to facilitate the maximisation.

In the next section, we perform experiments to illustrate the applicability of our interpretation. Our findings suggest that high diversity should generally reduce the size of voting margins, which would negatively impact generalisation performance. This disagrees with sentiment in the diversity literature, which generally suggests that high diversity should improve generalisation performance [16, 38, 63].

3.4 Experiments

In this section, we present two experiments. The first investigates the relationship between diversity and generalisation error, while the second examines a link between the double fault diversity measure and the exponential loss function.

3.4.1 Diversity and Generalisation Error

In this section, we perform an investigation which illustrates the utility of the margin theory interpretation of diversity. Specifically, we evaluate the hypothesis:

Hypothesis: *For ensembles with identical training error, high diversity will be associated with higher generalisation error.*

This question is important because:

- In diversity literature it is suggested that *high diversity will indicate low generalisation error.*
- In margin theory literature there are bounds that suggest that *large margins will indicate low generalisation error.*
- Our connection between diversity and margins suggests that (for positive margins), these two statements are in conflict.

Procedure

There are three main components to our experimental procedure:

1. We generate ensemble that achieve training error of less than 0.5, but are otherwise random.
2. We control for training error by 'binning' ensembles according to their performance on training data.
3. We measure Pearson correlation between diversity and generalisation error within each bin.

This second step may seem strange, since the 'training data' is not actually used for training. However, within each bin, the error on training data constitutes a biased (optimistic) estimate of generalisation error. In this sense, we can look at data in a single bin and ask: "Can any measure extracted from the training data provide additional information about generalisation error?"

We define our procedure in Algorithm 2:

Algorithm 2 Diversity and Generalisation Error Experiment

```
for i = 1...100000 do
    Sample 20 training data points
    Generate an ensemble of 25 random linear classifiers
    if Ensemble training error > 0.5 then
        Invert all the ensemble predictions
    end if
    Compute e_tr, ensemble training error
    Compute |m|, average absolute margin on training data
    Compute D_Q, Q statistic on training data
    Compute e_gen, generalisation error on hold-out data
end for
for For all ensembles within each distinct e_tr do
    Compute correlation between |m| and e_gen
    Compute correlation between D_Q and e_gen
end for
```

Our experimental procedure involves generating many (100000) random unweighted ensembles of 25 linear classifiers. We use 20 training examples to potentially invert the predictions of each ensemble, such that every ensemble has training error of 0.5 or less. Since the small number of training examples leaves only 11 possible values for training error, we can partition our 100000 ensembles according to training error (this will control for the relationship between diversity and training error). Within each set, we compute the correlation between average absolute margin on training data and generalisation error. Even though our analysis has suggested that the absolute margin is representative of 'diversity', we additionally show correlation between D_{Q} and generalisation error since D_{Q} has been prominently used in the literature.

Because ensembles were generated at random, the cardinality of the bins varies dramatically, with many thousands of ensembles achieving error around 0.5, and very few achieving error of 0. This will affect the statistical significance of correlation in very sparse bins.

To investigate this effect further, we perform the same experiment, but generating Bagging ensembles of 25 decision stumps or CART, and Adaboost ensembles of 15 decision stumps — in these cases we *train* on the 20 sampled data point. With Adaboost, we only measure $\overline{|m|}$, since the ensemble is weighted.

Results

The procedure gives us correlation values for each possible value of training error. We present these in Table 3.1, where bold highlighting indicates statistically significant correlation at a level of $p = 0.01$. The experiment was repeated for 8 datasets. Note that both $|m|$ and D_{Q} are *inversely* proportional to 'diversity', so a negative correlation indicates that *diversity is positively correlated with generalisation error*. Values for correlation are typically very low, but this is to be expected, since $\overline{m}$ and D_{Q} are measured on training data, and training error has been controlled for.

In Table 3.2, we show the behaviour in Bagging ensembles of Decision Stumps. In this case, the correlation is *positive* in many cases, which contradicts our hypothesis. We find similar results with CART bagging ensembles (Table 3.3 — due to the strength of CART models, we only show low error rates). Finally, the results for Adaboost ensembles are displayed in Table 3.4.

e_{tr}	heart $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	breast $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	ionosphere $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	diabetes $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0.5	-0.0018	0.0028	-0.0041	-0.0075	-0.0091	-0.0105	0.0018	0.0079
0.45	**-0.0794**	**-0.0525**	**-0.0865**	**-0.0452**	**-0.0856**	**-0.0689**	**-0.0717**	**-0.0478**
0.4	**-0.1622**	**-0.1023**	**-0.1802**	**-0.0831**	**-0.1642**	**-0.1237**	**-0.1393**	**-0.0966**
0.35	**-0.2249**	**-0.1512**	**-0.2391**	**-0.1061**	**-0.2049**	**-0.1485**	**-0.1898**	**-0.1282**
0.3	**-0.2827**	**-0.1922**	**-0.2964**	**-0.1264**	**-0.2172**	**-0.1465**	**-0.2267**	**-0.1546**
0.25	**-0.3094**	**-0.1880**	**-0.3569**	**-0.1587**	**-0.2028**	**-0.1111**	**-0.2524**	**-0.1705**
0.2	**-0.3428**	**-0.2341**	**-0.4050**	**-0.1804**	**-0.1948**	**-0.1048**	**-0.2855**	**-0.1958**
0.15	**-0.3770**	**-0.2392**	**-0.4381**	**-0.2034**	**-0.1520**	-0.0588	**-0.2646**	**-0.1600**
0.1	**-0.3738**	**-0.2077**	**-0.4421**	**-0.1597**	**-0.1366**	0.0186	**-0.2899**	-0.1403
0.05	**-0.3410**	-0.0997	**-0.4106**	**-0.1050**	-0.1661	-0.0710	**-0.3748**	-0.2431
0	-0.3782	-0.2089	**-0.4605**	-0.0163	-0.0341	0.4977	-0.5480	0.1750

e_{tr}	transfusion $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	iris-twoclass $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	gauss $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	checkerboard $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0.5	-0.0026	0.0164	-0.0022	0.0000	-0.0077	-0.0081	0.0205	0.0180
0.45	**-0.3079**	**0.2382**	**-0.0880**	**-0.0336**	**0.0767**	**0.0624**	0.0168	0.0133
0.4	**-0.3374**	**-0.2457**	**-0.1607**	**-0.0667**	**0.1460**	**0.1203**	**0.0348**	**0.0310**
0.35	**-0.2854**	**-0.2004**	**-0.1678**	**-0.0493**	**0.1946**	**0.1612**	**0.0618**	**0.0591**
0.3	**-0.2100**	**-0.1479**	**-0.1575**	-0.0154	**0.2558**	**0.2322**	**0.0799**	**0.0727**
0.25	**-0.1500**	**-0.1093**	**-0.1328**	0.0154	**0.2460**	**0.2447**	**0.1231**	**0.1186**
0.2	**-0.1134**	**-0.0791**	**-0.1356**	**0.0369**	**0.2633**	**0.2687**	**0.1400**	**0.1322**
0.15	**-0.0760**	**-0.0548**	**-0.1140**	**0.0654**	**0.2449**	**0.3003**	0.1143	0.1194
0.1	**-0.0654**	**-0.0496**	**-0.1476**	0.0437	**0.1738**	**0.2458**	0.3046	0.3595
0.05	-0.0364	-0.0163	**-0.1412**	0.0450	0.1154	**0.2421**	-0.0934	-0.0152
0	-0.0775	-0.1319	-0.0886	-0.0072	0.2283	-0.1022	N/A	N/A

Table 3.1: Table of correlations between diversity measures and generalisation error in ensembles of 25 random linear classifiers. Bold values indicate statistical significance at $p = 0.01$. Since both $\overline{|m|}$ and D_{Q} are inversely related to diversity, positive values in this table indicate negative correlation between diversity and generalisation error.

e_{tr}	heart $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	breast $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	ionosphere $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	diabetes $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0.05	**1.0000**	**1.0000**	N/A	N/A	N/A	N/A	N/A	N/A
0.1	0.3151	0.2045	N/A	N/A	N/A	N/A	**0.6210**	0.4107
0.15	**0.2813**	0.1130	0.4194	0.3027	0.0905	0.1862	**0.2425**	0.0333
0.2	**0.3501**	**0.1812**	0.2146	0.2535	0.0436	0.0294	0.0867	-0.0455
0.25	**0.3136**	**0.2019**	**0.4333**	**0.3793**	**0.1108**	**0.1501**	**-0.1320**	**-0.1389**
0.3	**0.1915**	**0.1445**	**0.4164**	**0.2863**	**0.1434**	**0.1777**	**-0.1966**	**-0.1556**
0.35	**0.1355**	**0.1202**	0.0341	0.0229	**0.1681**	**0.2231**	**-0.1975**	**-0.1496**
0.4	**0.1377**	**0.1176**	**-0.3413**	**-0.1595**	**0.1514**	**0.2014**	**-0.1656**	**-0.1100**
0.45	**0.1680**	**0.1122**	**-0.1361**	0.0081	**0.1443**	**0.1286**	**-0.1481**	**-0.0776**
0.5	**0.1755**	**0.1269**	**0.1153**	**0.1979**	**0.1295**	0.0281	**-0.1576**	**-0.0674**
e_{tr}	transfusion $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	iris-twoclass $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	gauss $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	checkerboard $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0	**-1.0000**	**-1.0000**	N/A	N/A	N/A	N/A	-0.0072	-0.0407
0.05	-0.1587	-0.1386	N/A	N/A	-0.0754	-0.0804	**0.1231**	-0.1032
0.1	**-0.2109**	-0.1203	N/A	N/A	-0.0683	-0.0023	**0.1213**	**0.1160**
0.15	**-0.1673**	**-0.0659**	-0.5295	-0.1920	**0.0959**	0.0164	**0.1280**	**0.1232**
0.2	**-0.2007**	**-0.1238**	**0.7249**	-0.4293	**0.4037**	**0.0920**	**0.1463**	**0.1486**
0.25	**-0.2316**	**-0.1780**	**0.5916**	**0.3573**	**0.8047**	**0.5208**	**0.1930**	**0.1918**
0.3	**-0.2327**	**-0.2117**	**0.5188**	-0.2523	**0.6847**	**0.5971**	**0.2351**	**0.2300**
0.35	**-0.2223**	**-0.2135**	-0.0885	**-0.3969**	**0.1316**	**0.1983**	**0.3081**	**0.2792**
0.4	**-0.2020**	**-0.1987**	**-0.1667**	**-0.0812**	-0.0049	**0.0966**	**0.3659**	**0.3228**
0.45	**-0.2005**	**-0.1484**	**0.1208**	**0.1250**	**0.1042**	**0.1142**	**0.4309**	**0.3747**
0.5	0.0935	**-0.1189**	**0.0852**	**0.0223**	**0.2160**	**0.2950**	**0.4440**	-0.1437

Table 3.2: Table of correlations between diversity measures and generalisation error in Bagging ensembles of 25 decision stumps. Bold values indicate statistical significance at $p = 0.01$. Since both $\overline{|m|}$ and D_{Q} are inversely related to diversity, positive values in this table indicate negative correlation between diversity and generalisation error.

	heart		breast		ionosphere		diabetes	
e_{tr}	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0.15	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0.1	0.0116	0.1287	N/A	N/A	**-1.0000**	**1.0000**	-0.2636	-0.0591
0.05	0.1233	0.0531	0.1515	0.0002	**0.2584**	0.1272	-0.1269	-0.0231
0	**0.0456**	**0.0805**	**0.2913**	**0.2223**	**0.2885**	**0.2281**	**-0.1168**	-0.0069
	transfusion		iris-twoclass		gauss		checkerboard	
e_{tr}	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$	$\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	$\rho(D_{\text{Q}}, e_{\text{gen}})$
0.15	-0.3448	-0.2804	N/A	N/A	N/A	N/A	N/A	N/A
0.1	**-0.3032**	-0.1988	0.8109	0.9058	-0.4716	0.0736	-0.0336	-0.3680
0.05	**-0.2636**	-0.0551	**0.3168**	0.1128	0.1673	0.1113	**0.1560**	0.1226
0	**-0.2502**	0.0064	**0.1578**	**0.0707**	**0.0995**	**0.1323**	**0.1700**	**0.0815**

Table 3.3: Table of correlations between diversity measures and generalisation error in Bagging ensembles of 25 CART base learners. Bold values indicate statistical significance at $p = 0.01$. Since both $\overline{|m|}$ and D_{Q} are inversely related to diversity, positive values in this table indicate negative correlation between diversity and generalisation error.

e_{tr}	heart $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	breast $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	ionosphere $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	diabetes $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	transfusion $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	iris-twoclass $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	gauss $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$	checkerboard $\rho(\overline{\lvert m\rvert}, e_{\text{gen}})$
0.25	N/A	N/A	N/A	N/A	**-0.2878**	N/A	-1.0000	0.1122
0.2	-0.7044	N/A	N/A	-0.4105	**-0.2102**	N/A	-0.3629	**0.1662**
0.15	-0.1253	N/A	N/A	-0.3822	**-0.2106**	N/A	-0.2004	**0.1422**
0.1	-0.1280	N/A	N/A	**-0.2160**	**-0.2337**	N/A	-0.1713	**0.1386**
0.05	-0.1169	N/A	N/A	**-0.2008**	**-0.2160**	1.0000	**-0.2160**	**0.1629**
0	**-0.0849**	**0.2120**	**0.2203**	**-0.1723**	**-0.1718**	**0.1309**	**0.0929**	**0.2096**

Table 3.4: Table of correlations between diversity measures and generalisation error in Adaboost ensembles of 15 decision stumps. Bold values indicate statistical significance at $p = 0.01$. Since both $\overline{|m|}$ and D_{Q} are inversely related to diversity, positive values in this table indicate negative correlation between diversity and generalisation error.

Conclusions

These experiments examined the relationship between diversity and generalisation error. It was unclear, based on previous literature, whether generalisation error would be positively influenced by high diversity (as is often suggested in diversity-oriented literature), or large voting margins (and, equivalently, low diversity). Our experiments examined correlations between the average absolute margin and generalisation error in various ensembles, while controlling for training error.

Most of the correlations we observed in random ensembles were positive and statistically significant. For training errors of 0.5, correlations were approximately 0, which is expected due to the symmetry of 2-class problems. This generally positive correlation supports the idea that large margins are indicative of good performance, and conversely that high diversity is not *in general* beneficial outside of its influence on training error.

However, whether the correlation between these two values was positive or negative depended on how the ensemble was generated; in Bagging ensembles, low generalisation error was often associated with *high* diversity. Of course, the quantity we measured ($\overline{|m|}$) describes only one aspect of the margin distribution, so from the perspective of margin theory, it is not too surprising that it does not correlate positively with generalisation error. While our initial experiment showed that our hypothesis was supported when ensembles were essentially random, the learning algorithm and dataset also clearly plays a large role in how the margin distribution behaves.

Similar experiments have previously been performed by Chen [16] and Kuncheva [54], but with slightly different experimental goals. Chen investigates this correlation without controlling for training error, so as to include the influence of the relationship between diversity and *training* error. Kuncheva is interested in the improvement of ensemble error over average individual error, and therefore partitions ensembles based on average individual error, rather than ensemble training error. We emphasise that, due to our controlling for training error, there is no conflict in the outcome of those experiments and the one we have performed here.

3.4.2 Double Fault Diversity and Exponential Loss

In Section 3.2.2, we showed a relationship between a quadratic function (double fault diversity), and exponential loss. In these experiments we will verify this

behaviour and develop our understanding of polynomial approximations to the exponential function in boosting algorithms.

We know that, using a Taylor expansion, we can derive polynomial functions that approximate e^{-m}, with higher degree polynomials giving closer approximations. However, we do not know to what extent this similarity will apply to the *behaviour* of ensembles that optimise such functions. We would like to answer the question *if we optimise double fault diversity in an ensemble, how similar is this to an Adaboost ensemble?* Additionally, the ability to use higher order Taylor approximations will allow us to examine the convergence of polynomial loss ensemble to exponential loss. This gives us a hypothesis:

Hypothesis: *By training with loss functions that approximate exponential loss, we can create boosted ensembles that behave similarly to Adaboost.*

To perform the investigation we use the AnyBoost algorithm that we introduced in Section 2.2.4. The allows us to derive learner and example weight update rules from both polynomial and exponential loss function (although we are slightly restricted in that we require the polynomial functions to be *convex*).

For example, we show second and forth degree polynomial cost functions:

$$C_{k=2} = \sum_{i=1}^{N} 1 - m_i + \frac{1}{2}m_i^2, \tag{3.57}$$

$$C_{k=4} = \sum_{i=1}^{N} 1 - m_i + \frac{1}{2}m_i^2 - \frac{1}{6}m_i^3 + \frac{1}{24}m_i^4. \tag{3.58}$$

we analytically compute the appropriate derivatives to train AnyBoost (Equations 2.23 and 2.24), and use Laguerre's method[4] to find their roots during the training process.

Experimental Procedure

We define our procedure precisely in Procedure 3:

We produce 500 random train/test splits of the data (100 training and 100 testing examples). For each split, we train AnyBoost ensembles with $k \in \{2, 4, 8, 16, 32, 64, 128\}$, and a single Adaboost ensemble. Finally, we compute the average

[4]We perform up to 10^5 iterations with an error tolerance of 10^{-5}.

Algorithm 3 Polynomial and Exponential Loss Behaviour

```
for i = 1 ... 500 do
    Sample 100 training data points
    for k = 2, 4, 8 ... 128 do
        Train Anyboost using a k-degree cost function, C_k
        Compute e^k_gen, generalisation error
    end for
    Train Adaboost
    Compute e^ada_gen, generalisation error
end for
```

generalisation accuracy difference between polynomial learners and Adaboost. The hypothesis is confirmed if these differences tend to 0 as k increases. We repeat the experiment on 6 datasets for 4 different base learning algorithms — Gaussian Naïve Bayes, CART, perceptron (learning rate 0.01, 10 epochs), decision stumps. In every case, the ensembles contain 15 base learners. More general information on our experimental environment is provided in Appendix A.

Results

In Figure 3.2, we show the results for all the datasets studied, with polynomial degree up to 128. With the exception of the breast cancer dataset, all other experiments show convergence to the error rate of Adaboost. The convergence on the breast cancer dataset seems to be ongoing at $k = 128$ but we have not tried higher degree polynomials due to numerical precision issues. For two base learners — CART and decision stumps — low degree polynomials were consistently worse or better (respectively) than exponential loss, while for Naïve Bayes and perceptron learners this was dataset dependent.

Conclusions

The general conclusion here is that *Adaboost with exponential loss can be seen an algorithm that explicitly optimises diversity*; in a sense this is already known — that Adaboost introduces diversity by example weighting — but our experiments show how exponential loss relates to existing diversity measures. Quadratic loss represents a trade-off between squared-margin diversity (e.g. KW variance) and average margin (i.e. average individual accuracy). The interpolating loss functions — higher order polynomials — imply the same intuition, with negatively

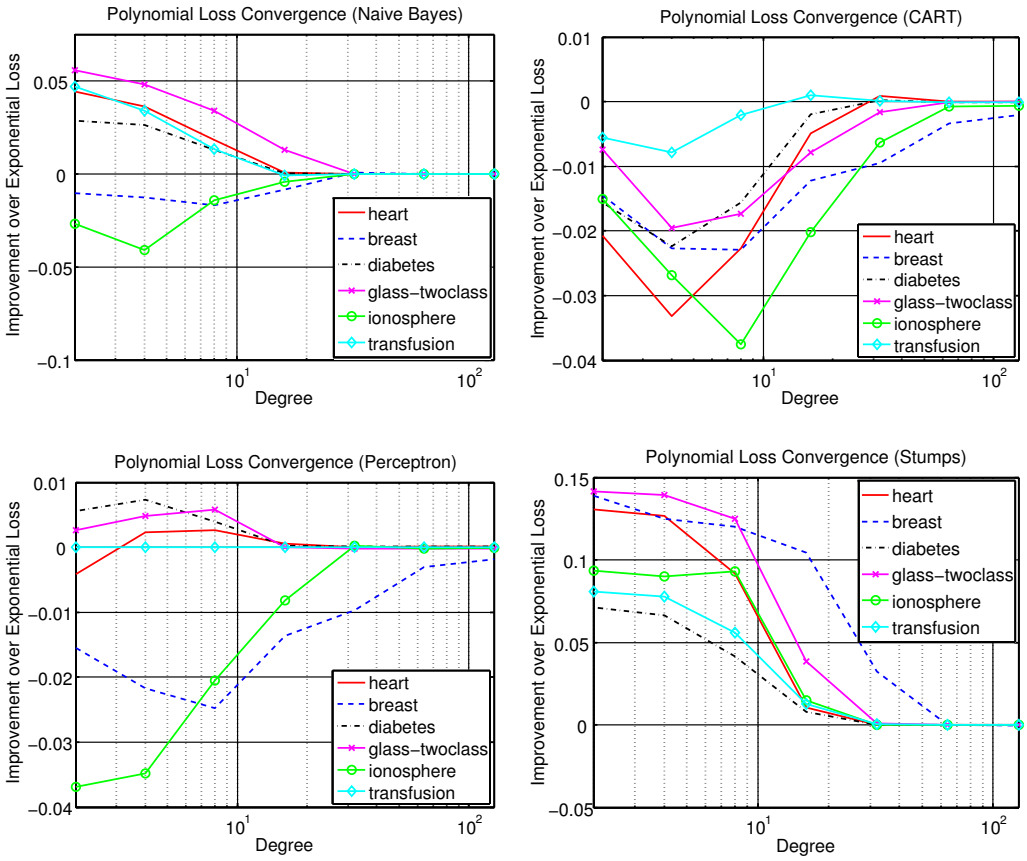

Figure 3.2: Convergence of AnyBoost with Polynomial loss to Exponential loss, as degree of Taylor approximation increases. Top left: Gaussian Naïve Bayes, Top right: CART, Bottom left: Perceptron (lr = 0.01, epochs = 10), Bottom right: Decision Stumps. Polynomial degrees for all powers of two up to 128.

	x_1	x_2	x_3	x_4	x_5	x_6
h_1	1	-1	-1	1	-1	1
h_2	-1	1	1	-1	-1	1
m	0	0	0	0	-1	1

	x_1	x_2	x_3	x_4	x_5	x_6
h_1	-1	-1	-1	1	-1	1
h_2	1	1	1	-1	-1	1
m	0	0	0	0	-1	1

Table 3.5: Two possible scenarios for an ensemble of two base learners making predictions on a dataset of six examples. The ensemble on the left has $\phi = -\frac{1}{3}$, while the ensemble on the right has $\phi = -\frac{1}{4}$; however, the margin distributions are identical.

weighted asymmetric components (encouraging accuracy) and positively weighted symmetric components (encouraging diversity).

3.5 Alternatives to the Margin Interpretation

Much of the argument we have presented for viewing diversity as a property of the margin distribution has been objectively valid (i.e. our derivations show that most diversity measures *are* margin measures). One possible conclusion at this point would be *quantifying diversity is not useful since the margin distribution describes it anyway.* In this section, we aim to pose questions that, when answered, can help to determine the validity or otherwise of that conclusion.

3.5.1 Diversity as Correlation

The phi coefficient cannot be described by examining the margin distribution. Table 3.5 shows an example of a scenario where correlation differentiates between two ensembles with identical margin distributions.

The implication here is that the margin distributions tells us *how much* disagreement there is in the ensemble, while correlation also considers *which base learners* are disagreeing. This is the basis for what we consider to be the strongest case for non-margin-based diversity, and gives us an initial question to consider:

Do the two ensembles illustrated in Table 3.5 have different amounts of diversity?

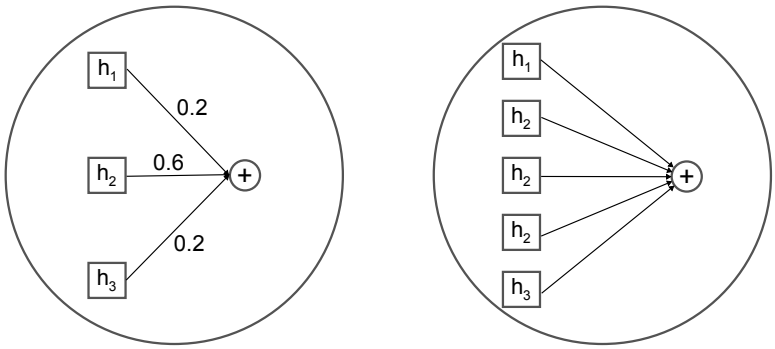

Figure 3.3: We show how a weighted ensemble (left) can be converted to a larger, unweighted ensemble, such that its behaviour (and margin distribution) is preserved. Each base learner is duplicated some number of times so that it accounts for the correct proportion of the unweighted ensemble (i.e. since $\alpha_2 = 3\alpha_1$, h_2 occurs three times as often as h_1 in the unweighted ensemble).

3.5.2 Invariance over Ensemble Resizing

The issue of correlation vs. margins also arises when we consider resizing or weighting an ensemble. Consider an ensemble $L = 3$ and its margin distribution; now double the size of the ensemble by duplicating its base learners ($h_4 = h_1, h_5 = h_2, h_6 = h_3$). The margin distribution does not change, but we might consider diversity to be lower since half of the ensemble comprises duplicate base learners. The difference between the ensembles would be captured by correlation, even though their margin distributions are identical.

The invariance of the margin distribution when enlarging the ensemble in this way does have some advantage though — it gives us a natural way to describe the diversity of a weighted ensemble. For all of the measures that we interpreted using margins, the weighting of learners could be achieved by considering the margins of weighted ensembles; $m(\mathbf{x}, y) = y \sum_{l=1}^{L} \alpha_l h_l(\mathbf{x})$. Existing weighted diversity measures (such as those presented by Tang [84]) are consistent with this interpretation. Therefore, the margin distribution of a weighted ensemble can be achieved[5] by an unweighted ensemble where each distinct base learner has been duplicated in proportion to its weight.

According to the margin interpretation of diversity, such ensembles (as in Figure 3.3) would have *the same* diversity. A correlation-oriented diversity measure,

[5]Assuming weights are rational numbers; otherwise the margin distribution is only *approximated*.

however, would assign different amounts of correlation. So our next question is:

If two ensembles of different sizes contain the same distinct base learners in the same proportions (either via weighting or duplication), do they necessarily have the same amount of diversity?

3.5.3 Tumer and Ghosh Correlation Result

While we have interpreted theoretical results of Breiman [10], Kuncheva [55] and Tang [84] within the voting margin framework, we have shown no such interpretation for the work of Tumer and Ghosh [86]. The correlation term used in their result is not exactly the D_ϕ term we defined in Chapter 2; however, it is based on the correlation between base learner errors. Whether or not the main theoretical result here (Equation 2.35) can be expressed in terms of the margin distribution is an open problem; their framework is quite complex and we have not found any margin-based interpretation. To gain substantial evidence for or against the margin interpretation of diversity, we would need to answer:

Is it possible to express the result of Tumer and Ghosh (Equation 2.35) in terms of the margin distribution?

3.5.4 Learner- and Data-dependent Effects

While we consider that the margin distribution describes the important aspects of ensemble behaviour in general, we can imagine that correlation-like measures would have significance in specific scenarios. It is hard to construct an example without considering convoluted learning algorithms or datasets; however, if there were any cases where, for example, measuring correlation could differentiate between different types of noise or be used to detect when a base learning algorithm was inappropriate for the task, then this would give us a good reason to give deeper consideration to measures of correlation between learners.

Are there specific learning algorithms or classes of learning problem where the correlation between base learner predictions on training data provides useful information that is not contained in the margin distribution?

3.6 Conclusions

In this chapter we have made several novel contributions. We have:

1. Shown direct theoretical links between many diversity measures and the margin distribution.

2. Discussed corroboration of our results by existing empirical studies.

3. Analysed how other theoretical results from the diversity literature relate to the voting margin framework.

4. Presented an experiment that tests two contrasting predictions about how diversity affects generalisation error.

5. Performed a further experiment showing the relationship between polynomial loss functions (e.g. double fault diversity) and exponential loss (Adaboost).

These results are valuable for several reasons. Being able to interpret diversity within the voting margin framework enriches our understanding of what diversity is and how it should be used.

The impact for practitioners could be, as we suggested in Section 3.5, that *quantifying diversity is not useful since the margin distribution describes it anyway.* In this sense, the quantitative aspect of ensemble diversity should be considered within the framework of voting margins — for example, to say 'the algorithm encourages diversity' really means 'the algorithm encourages the margin distribution to have a certain shape'.

Is the future really so bleak for 'diversity'? Not necessarily: in Section 3.5 we posed four questions — in the first two cases, we asked about what it meant for two ensembles to have the same amount of diversity. The next questions were open problems regarding diversity. If any of the questions were answered affirmatively, it would imply that diversity *does* have some interesting and valuable meaning outside of the margin distribution. In this case, our work has contributed by *refining* the definition of diversity — specifically, we have shown that it would *only* make sense to measure diversity as correlation or the Q statistic, and that even in this case, it should be viewed *from a holistic perspective that considers its relationship with the voting margins.*

Understanding this link between diversity and margins also enables our further contributions in subsequent chapters; in Chapter 5, we show that diversity can be optimised using gradient descent on a margin-based loss function, using a similar

approach to our experiment from Section 3.4.2. We then exploit this to address the issue of adaptation in non-stationary learninga

Part II

Diversity in Non-Stationary Learning

Chapter 4

Background and Related Work

In this chapter, we introduce the specific set of assumptions that characterise *non-stationary learning*, in preparation for Chapter 5, where we develop a diversity-oriented approach to non-stationary learning. After describing the problem, we discuss some important issues and state-of-the-art techniques.

4.1 From Ensemble Diversity to Non-Stationary Learning

In the previous part of this thesis, we presented a link between ensemble diversity and voting margins. This contributed towards a more unified understanding of ensemble algorithms, and opened up several interesting research directions. The ability to quantify diversity using voting margins makes it possible to *manage* diversity using existing techniques based on voting margins; as such, it becomes easier to evaluate hypotheses relating to the impact of high — or low — diversity.

In the field of non-stationary learning, there has been recent interest in applying ensemble methods [70]. Ensembles can be of use here because of their inherent decomposition of complex tasks; diversity is one way of describing how effectively a learning task has been 'decomposed', and as such we can imagine that it will be especially important in non-stationary learning.

In this chapter, we will first introduce the domain of non-stationary learning and describe some popular approaches — especially those based on ensemble algorithms. Then, in Chapter 5, we consider intuitively and theoretically how we should expect diversity to affect the ability of an ensemble to adapt to changes.

Finally, we derive an algorithm and evaluate it on toy and real datasets to test our hypotheses.

4.2 What Makes Non-Stationary Learning Difficult?

Non-stationary learning differs from the conventional supervised learning framework that we described previously in two respects: the dataset is ordered and we should be able to make predictions at any point in the sequence, and examples are not drawn from a fixed distribution. Both of these properties imply a *relaxation* of assumptions that we normally have in supervised learning (so in this sense, non-stationary learning is *at least as hard* as stationary learning).

4.2.1 Incremental Learning

Incremental Learning is the process of learning from sequential data, where data *is* still i.i.d. In this case, the primary challenges are:

1. To train a learner such that predictions can be made after processing any number of examples (Anytime prediction).

2. For an incrementally trained learner to be as similar as possible to a batch learner trained on the same data (Losslessness).

3. For the time and space complexity of the learner to be constant irrespective of the amount of data (Efficiency).

Achieving all three of these is possible for certain types of learning model. Examples of lossless algorithms are the categorical and Gaussian Naïve Bayes algorithms that we use in our experiments [68, Section 2.3]. To illustrate this, we first present the batch learning rules for Gaussian Naïve Bayes models, before giving

equivalent incremental update equations:

$$Pr(y|\mathbf{x}) \propto Pr(y) \prod_{j=1}^{d} \mathcal{N}(\mathbf{x}^{(j)}; \mu^{(j)}(y), \sigma^{(j)}(y)), \tag{4.1}$$

$$\widehat{Pr}(y) = \frac{N(y)}{N}, \tag{4.2}$$

$$\mu^{(j)}(y) = \frac{1}{N(y)} \sum_{i=1}^{N} \delta[y_i = y] \mathbf{x}_i^{(j)}, \tag{4.3}$$

$$\sigma^{(j)}(y) = \sqrt{\frac{\sum_{i=1}^{N} \delta[y_i = y](\mathbf{x}_i^{(j)} - \mu^{(j)}(y))^2}{N(y) - 1}} \tag{4.4}$$

Here, we use a Bernoulli distribution for the prior, with $N(y)$ denoting the number of examples with $y_i = y$. The class-conditional distributions for the features are Gaussians, with means $\mu^{(j)}(y)$ (for feature j and class y) and standard deviations $\sigma^{(j)}(y)$. All of these are estimated from the data, and prediction is achieved by $\arg\max_y Pr(y|\mathbf{x})$.

To approximate this in incremental learning, we need to be able to *update* all the parameters each time we receive a new training example. For incremental problems, we will use t to denote the time step (i.e. number of training examples seen), so subscript t will refer to a quantity *at time* t. Therefore, we have update rules for Gaussian Naïve Bayes, based on Welford's technique for incrementally updating means and variances [93]:

$$\widehat{Pr}_t(y) = \frac{N_t(y)}{t}, \tag{4.5}$$

$$N_0(y) = 0, \quad N_t(y) = N_{t-1}(y) + \delta[y_t = y], \tag{4.6}$$

$$\mu_0^{(j)}(y) = 0, \quad \mu_t^{(j)}(y) = \mu_{t-1}^{(j)}(y) + \frac{\delta[y_t = y](\mathbf{x}_t^{(j)} - \mu_{t-1}^{(j)}(y))}{N_t(y)}, \tag{4.7}$$

$$\gamma_0^{(j)}(y) = 0, \quad \gamma_t^{(j)}(y) = \gamma_{t-1}^{(j)}(y) + \delta[y_t = y](\mathbf{x}_t^{(j)} - \mu_{t-1}^{(j)}(y))(\mathbf{x}_t^{(j)} - \mu_t^{(j)}(y)), \tag{4.8}$$

$$\sigma_t^{(j)}(y) = \sqrt{\frac{\gamma_t^{(j)}(y)}{N_t(y) - 1}}. \tag{4.9}$$

These equations show how incremental learning works in a Gaussian Naïve Bayes model; $N_t(y)$ indicates the total number of y-labelled instances seen at or before time t, $\mu_t^{(j)}(y)$ indicates the class conditional mean of the j^{th} feature after t

training examples, while $\sigma_t^{(j)}(y)$ is its standard deviation. We use $\gamma_t^{(j)}(y)$ to simplify the process of updating $\sigma_t^{(j)}(y)$; $\gamma_t^{(j)}(y)$ stores the sum of squared differences between the j^{th} feature and its mean for class y on the first t examples.

All the parameters computed in this incremental fashion will be the same as if they had been computed using the batch algorithm on all data up to time t. Furthermore, the updates require $O(1)$ computation[1], so the Gaussian Naïve Bayes models satisfies all three desiderata of anytime prediction, losslessness, and efficiency.

However, such a convenient conversion from batch to incremental learning is not possible for most types of model; for example, building a lossless decision tree incrementally is not possible to do with constant update time — inserting a value normally requires a search of the tree, and the splitting criterion needs to be re-evaluated after each insertion. Because of this, *approximate* tree learning algorithms like ITI (Incremental Tree Inducer) [87] and Hoeffding Trees [21] are necessary to perform incremental learning efficiently.

4.2.2 Non-Stationarity

The assumption of identical and independently distributed data is an essential part of supervised learning; it implies that the training data is somehow representative of the true distribution. If we *completely* discard this assumption, then learning becomes impossible — the distribution that we are interested in for prediction would not be related to the distribution that our data was sampled from.

The assumption of non-stationarity lies somewhere between these two extremes — we assume that historic data is *usually* representative of the current true distribution, or that it is *almost* representative of it. Žliobaitė [91] describes this as the *"future assumption"*. She presents three possible assumptions that could be made about the relationship between the distributions from which $\langle \mathbf{x}_t, y_t \rangle$ and $\langle \mathbf{x}_{t+1}, y_{t+1} \rangle$ are sampled:

1. The distributions are the same,
2. The latter distribution can be predicted with $\mathbf{x}_{t+1}$,
3. The latter distribution can be predicted with $\langle \mathbf{x}_1, y_1 \rangle \ldots \langle \mathbf{x}_t, y_t \rangle$.

[1] Within an arithmetic model of computation (i.e. constant time arithmetic operations).

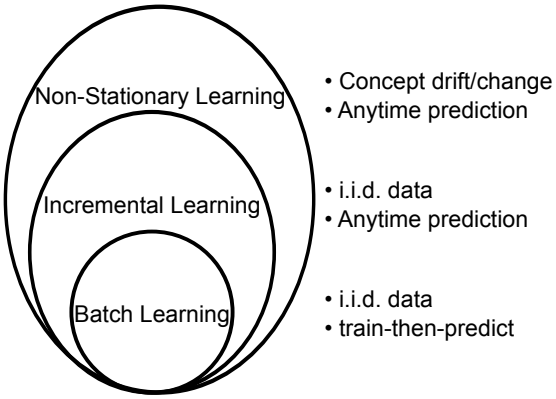

Figure 4.1: Relationship between the assumptions in batch, incremental and non-stationary learning paradigms.

However, alternative assumptions are described elsewhere in the literature; for example, Bach and Maloof [2] describe an algorithm which computes a probability distribution over *possible change points*. A *change point* is a value of t at which the distribution exhibits a single, arbitrary change; a common assumption in non-stationary learning is that such change points occur *rarely*. If the change really is arbitrary, then the optimal strategy in such problems is to maintain a model that has been trained on *all data since the previous change point*. We discuss non-stationary learning with change point detection in more detail in Section 4.3.2. Another kind of assumption occurs in *concept drift* problems: the data distribution may change slowly over time. Here, it is appropriate to assume that the amount of drift between any two examples is *small*, so that recently seen data is more relevant to the current task than distantly historic data.

We summarise the differences between batch, incremental and non-stationary learning in Figure 4.1.

4.2.3 Parameter Selection and Evaluation

Without the assumption of i.i.d. data, selecting parameters and evaluating algorithms becomes more difficult. A simple and general metric for evaluation is *test-then-train* accuracy [6]: if we denote a model that is trained on the first t examples as H_t, then the test-then-train error is:

$$e_{\text{ttt}} = \frac{1}{N} \sum_{t=1}^{N} \delta[H_{t-1}(\mathbf{x}_t) \neq y_t], \tag{4.10}$$

so for each example, we first test the ensemble prediction, and then train the ensemble on that example. To understand what this means, we introduce a time-indexed distribution, $Pr_t(\mathbf{x}, y)$, which is the distribution from which example $\langle \mathbf{x}_t, y_t \rangle$ is drawn. Therefore, we can consider the probability of error at time t:

$$e_{\text{gen}}(t) = \int_{\mathbf{x},y \in \mathcal{X} \times \mathcal{Y}} \delta[H_{t-1}(\mathbf{x}) \neq y] Pr_t(\mathbf{x}, y) d\mathbf{x} dy. \tag{4.11}$$

Test-then-train error is an estimate of this quantity summed over all t:

$$e_{\text{ttt}} \approx \frac{1}{N} \sum_{t=1}^{N} e_{\text{gen}}(t). \tag{4.12}$$

Test-then-train error tells us, for a problem with same sequence of distributions $Pr_1(\mathbf{x}, y) \ldots Pr_N(\mathbf{x}, y)$, what the expected overall error would be. Obviously as an estimate of $e_{\text{gen}}(t)$ for any single t, e_{ttt} is very poor — it only measures error on a single data point. However, since the testing occurs *before* training, e_{ttt} is unbiased, and so for large N, it will converge to the value in Equation 4.11.

So averaging test-then-train over the whole learning process gives a single value — e_{ttt} — that is representative of overall performance. However, often (especially when *developing* algorithms) we want to know what performance is at a *specific point during training*. The test-then-train error for a single t is either 0 or 1, which is not very informative. One option is *windowed error*, where we average test-then-train error over a window of K examples:

$$e_{\text{win}}(t) = \frac{1}{K} \sum_{k=t-K}^{t} \delta[H_{k-1}(\mathbf{x}_k) \neq y_k]. \tag{4.13}$$

There is a trade-off here in choosing K: a large K will give a low-variance estimate, while small K will give a low-bias estimate. A variant of e_{win} is *prequential error*, as is used by Baena-García et al. [3], where we average over test-then-train error on all previous examples (or, in some cases, all examples since a known change point). For a start time K_0, prequential error is:

$$e_{\text{preq}}(t) = \frac{1}{t - K_0} \sum_{k=K_0}^{t} \delta[H_{k-1}(\mathbf{x}_k) \neq y_k]. \tag{4.14}$$

When $K_0 = 0$ and $t = N$, prequential error is test-then-train error averaged over

all the data.

In toy scenarios, or problems where we know more about the nature of the concept change, it can be possible to compute error rates using hold-out data. Basically, for this to be possible, we need to have, at each time t, a substantial amount of data sampled from $Pr_t(\mathbf{x}, y)$. In problems with *abrupt concept change*, for example, we may have several hundred data points from a stationary distribution, followed by hundreds more from a different stationary distribution. If we know where the change points occur, then the data can be segmented into a number of stationary concepts; within each concept, we can keep hold-out training data, and use techniques such as a cross-validation to perform repeat experiments or select parameters.

Without hold-out data, we know of no robust way to select parameters in non-stationary learning problems; Kuncheva and Žliobaitė [52] subsample from non-stationary data, selecting every k^{th} example, to produce k different datasets with approximately the same distribution, though this increases the rate of change in the datasets by a factor of k. In our work, we will generally choose parameters based on what works well on stationary data.

4.3 Non-Stationary Learning Algorithms

There are many algorithms that have been developed to address non-stationary learning problems. In some cases, these are modifications of existing algorithms [42, 56, 69, 87], while other approaches are entirely novel [3, 28, 45]. Since there is a lot of variety in non-stationary learning problems (e.g. abrupt concept change, gradual drift, recurring concepts) different algorithms can be applicable in different situations, so we first describe broad categories into which these algorithms fit. We then describe non-ensemble or model-independent algorithms for incremental and non-stationary learning, before finally reviewing *ensemble* algorithms.

4.3.1 Approaches

Žliobaitė [91] describes a taxonomy of adaptive supervised learning techniques based on their methods of determining 'when' to adapt and 'how' to adapt. The question of 'when' to adapt can be broadly separated into two approaches, which Žliobaitė describes as "trigger-based" and "evolving". Trigger-based algorithms rely on a detection mechanism to provide a signal when the data distribution is

considered to have changed substantially, while evolving algorithms will not explicitly model concept changes, but rather aim to adapt continuously. Because of this, evolving algorithms must necessarily be *lossy*, so as to be able to 'forget' old data. Trigger-based algorithms may be normal incremental learning algorithms that are simply reset or otherwise modified when a change is detected.

With regards to *how* non-stationary learning algorithms adapt, Žliobaitė presents a few general strategies: training set formation and model manipulation.

In training set formation techniques, the model itself is not directly affected by the detection or adaptation strategy; instead, training data is manipulated to describe the non-stationarity. For example, windowing algorithms select a window of recent examples on which to train the learner, while the learner itself trains on the data inside the window as though it were stationary.

In model manipulation, the parameters of the model are explicitly modified in response to the changes in the distribution. These types of algorithm include tree restructuring techniques like VFDTc [27]. Other algorithms, like windowed KNN [56] and SVMs [42] are a combination of training set formation and model manipulation.

4.3.2 Non-Ensemble Algorithms

In this section we describe a number of techniques for addressing non-stationary learning without applying ensemble algorithms. In some cases, we will apply these techniques in our empirical evaluations, and therefore will provide a precise definition for the algorithm too.

First, we describe some *windowing* techniques. The idea of windowing is to train a learner on a set of recent training examples, the size of which is independent of N. This achieves constant-time updating, while also *forgetting* about data prior to the start of the window. There are three primary issues in windowing:

1. Which examples should be kept in the window?
2. How should the learner be trained on windowed data?
3. How large should the window be?

Often, the answer to the first question is 'the most recently seen examples', but sometimes, especially when the task is close to stationary, it is more important that the window of examples is *representative* (spread out over the input space)

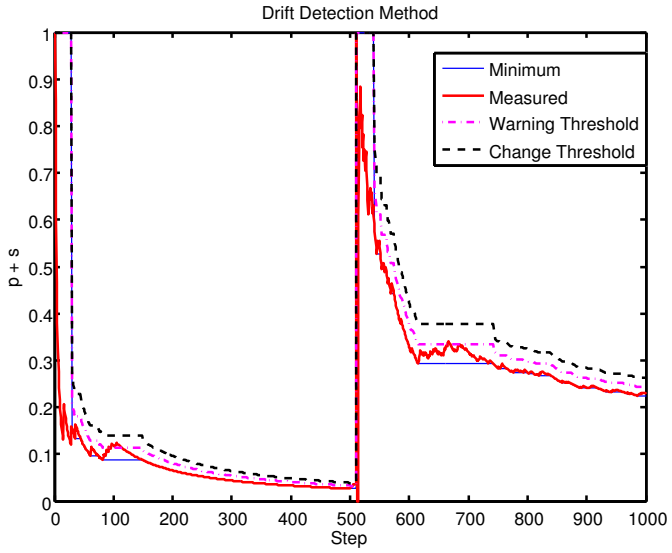

Figure 4.2: Change detection measurements in a DDM change detector. Errors are taken from a Naïve Bayes learner on the first two concepts of STAGGER. The change occurs at $t = 500$ and is detected at $t = 513$.

rather than *recent*. Training a learner can be accomplished in two ways: the simplest solution is to completely retrain the model every time the contents of the window change, which has the advantage of not requiring any special considerations on the part of the learner, although its time complexity is linear in the size of the window, even if constant with respect to N; the alternative is to train incrementally on examples that are added to the window, and *untrain* on examples that leave the window. In this case, the amortised time complexity[2] of each update is $O(1)$, but the learning algorithm must not only learn incrementally, but also be able to unlearn data.

The size of a window is an important issue in windowing algorithms [52, 89]. In concept *change* problems, we often want to choose a window that encompasses every example since the change point. Gama et al. [28] propose a Drift Detection Method (DDM) which detects changes in the distribution of prediction errors. DDM models errors as a binomial distribution with probability of failure:

$$p_t = \frac{1}{t}\sum_{k=1}^{t} \delta[H_{k-1}(\mathbf{x}_k) \neq y_k], \tag{4.15}$$

where t denotes the number of examples since the previous detected change, and

[2]This is *amortised* time complexity if the size of the window varies; worst case time for a single update is linear in the maximum window size.

p_t is the error rate since then (i.e. prequential error rate since previous detected change). The standard deviation of this distribution is:

$$s_t = \sqrt{\frac{p_t(1-p_t)}{t}}. \tag{4.16}$$

As training progresses, the value of t for which $p_t + s_t$ was at a minimum is stored. If the prequential error rate decreases below this, then a new minimum value is stored. The ratio between the current and minimum $p_t + s_t$ values can be used to compute a *probability* that the distribution of errors has changed. This is used for determining *warning* and *drift* situations — the warning state indicates that subsequent examples should be stored in preparation for a drift. When a drift is detected, the window is cleared and replaced with only examples that occurred after the warning state. This reflects the fact that it may take some time to differentiate between noise and drift, but that if we observe apparently noisy behaviour immediately prior to detecting a drift, then that data is likely to be from the new concept. DDM therefore makes a strong assumption about the nature of concept drift: that it occurs in abrupt transitions between unrelated concepts. Furthermore, because the detection is based on measures of the learner's error rate, DDM is only able to detect drifts that *increase the error rate*; while these may be the most important drifts in many applications, there are drifts that would not be recognised by DDM but where we would benefit from reducing the window size. Figure 4.2 shows DDM detecting a concept change in STAGGER (See Appendix A for a dataset description). The longer it takes to detect a change, the lower the thresholds become — this reflects the fact that each new prediction will affect p by only a small amount. Note that the measured value sometimes exceeds the warning threshold when there is no change — at $t = 100$ and $t = 660$ — but only exceeds the change detection threshold after the genuine change at $t = 510$.

EDDM [3] (Early Drift Detection Method) follows a similar principle to DDM, but instead of examining the distribution of *prediction errors*, it measures the distribution of *distances between prediction errors*. There are two rules for warning/drift levels (p'_t and s'_t are the mean and standard deviations of the distances

between errors up to time t):

$$\alpha > \frac{p'_t + 2s'_t}{p'_{\max} + 2s'_{\max}}, \tag{4.17}$$

$$\beta > \frac{p'_t + 3s'_t}{p'_{\max} + 2s'_{\max}}, \tag{4.18}$$

where α and β are parameters that define the warning and drift levels. In their experiments, Baena-García et al. use $\alpha = 0.9$, $\beta = 0.95$, and suggest choosing parameters by experimentation. They propose that EDDM will detect changes faster than DDM, especially gradual changes. This could be seen to be the case because, although p'_t is related to p_t by $p'_t = \frac{1}{p_t}$, the standard deviation of the distance captures some additional property of the error distribution; so s'_t will be especially small when the distances between errors is *changing*, while in DDM s_t is entirely determined by p_t.

Kuncheva and Žliobaitė [52] discuss window sizes in abrupt concept change scenarios, showing a trade-off between large windows (low error during stationary periods) and small windows (shorter transition period). By expressing the theoretical error of a classifier trained on K examples in terms of asymptotic error ($K \to \infty$) and a classifier/data dependent function, it is possible to compute error rates for windowed classifiers where all K examples are drawn from a stationary distribution. They then consider the error immediately after a concept change — when the entire window contains data from the old concept — and show the interpolation between this error rate and the stationary error rate for the new concept. By approximating the error rate during adaptation with a straight line, they produce an expression for the overall error rate in terms of K, the total size of the dataset, and some classifier and data dependent functions. Finally, they derive a window resizing algorithm for nearest mean classifiers[3] based on this bound. In Figure 4.3, we give an example of the concept change scenario they considered. The window size determines the error on stationary data and the amount of time taken to adapt.

Klinkenberg [42] proposes a variable window size strategy for online SVMs, based on the ease of estimating approximate upper bounds on their leave-one-out training error. The window adjustment algorithm selects a window size that minimises these estimates, by training multiple SVMs on different windows. This

[3]Nearest mean classifiers compute class means from input data, and then assign labels based on which class mean is closest (via euclidean distance).

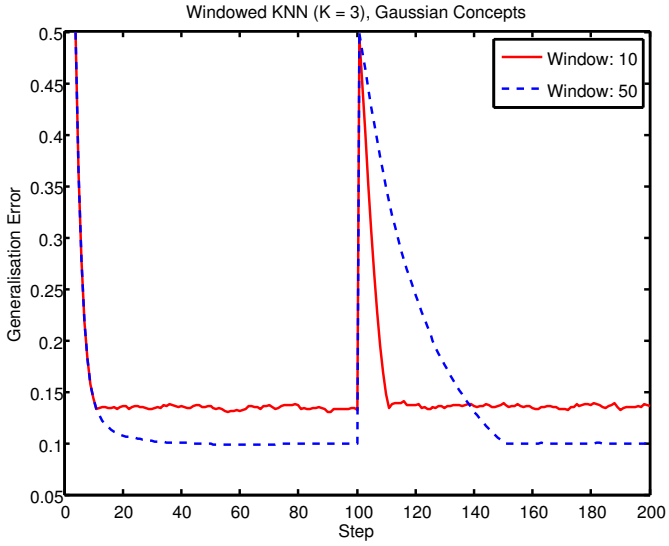

Figure 4.3: Comparison of window sizes with KNN (K = 3) learners, window sizes of 10 and 50. The learning task comprises two unit variance Gaussians; in the first concept ($t \in [1, 100]$), class 1 has mean $(0, 0)$ and class 1 has mean $(2, 2)$. In the second concept ($t \in [101, 200]$), class 1 has mean $(2, 0)$ and class 2 has mean $(0, 2)$. Small window sizes give high error during stationary concepts, but faster adaptation to concept changes, while larger window sizes perform better during stationarity and adapt more slowly to changes.

is proposed to react better to non-stationarity because the leave-one-out error estimate is more stable for small amounts of data than, for example, the error rates estimated by DDM.

Perceptron and Multilayer Perceptron (MLP) learners can be easily modified to work in non-stationary environments; the normal training procedure for these is based on multiple epochs, as in Algorithm 4.

Algorithm 4 Multi-Epoch Perceptron/MLP training procedure

Require: Number of epochs, K

```
Initialise learner H randomly
for k = {1 ... K} do
    for i = {1 ... N} do
        Train H on ⟨x_i, y_i⟩
    end for
end for
```

Training MLPs or perceptrons in this way requires batch access to the entire training dataset, which is no longer possible in incremental learning. In many

cases, a large value of K is required for good performance. However, if we were in a *stationary* but *incremental* setting, we could omit the outer for-loop (equivalently, let $K = 1$), and still expect good performance *for large* N. This is a lossy version of the training procedure on batch data, because we would expect $K = 1$ to give worse performance until N is sufficiently large. Furthermore, in non-stationary learning, it may be that the distribution changes before convergence can occur. However, using a single epoch could be seen to be an advantage when the concept is changing, because more recently seen examples will have more impact on the behaviour of the classifier. MLPs and Perceptrons also include a *learning rate* parameter, which determines the size of update that occurs after training on a single example. High values of learning rate will promote forgetting, while lower values are more appropriate if earlier data is still considered relevant.

4.3.3 Ensemble Algorithms

We now describe *ensemble algorithms* for non-stationary learning; ensemble algorithms are inherently modular, so there is an intuitive argument for applying them in situations where it may be necessary to forget old knowledge and adapt to new concepts. Žliobaitė suggests that most 'evolving' approaches to non-stationary learning are ensemble algorithms [91], with the combination rules being a primary tool for promoting adaptation. Kuncheva [50] surveys a number of ensemble-based approaches, suggesting in her conclusion that the accuracy and flexibility of ensembles made them especially appropriate for non-stationary problems. In this section, we introduce some important ensemble algorithms; for algorithms that have special significance in the context of this thesis, we also provide a full definition.

Dynamic Weighted Majority [45] (DWM) is a popular ensemble algorithm for dealing with concept drift. In DWM, heuristic rules guide the addition, weighting and removal of base learners, based on their performance on recently seen data. Since we will later use this algorithm to benchmark against our own, we provide the full description[4] in Algorithm 5.

DWM trains base learners incrementally on all data. However, by strategically weighting, removing and replacing base learners, DWM can implicitly adapt to non-stationarity. The period parameter, p, determines how often weight decreases, pruning and learner addition occur — larger values here can reduce

[4] $x \equiv y \mod z$ is used to denote x being congruent to y, mod z (i.e. $x \bmod z = y$).

Algorithm 5 DWM Algorithm (2-class)

Require: $p \leftarrow$ update period
Require: $\beta \leftarrow$ error discount
Require: $\theta \leftarrow$ pruning threshold
  $L \leftarrow 1$
  Create h_1, new base learner
  $\alpha_1 \leftarrow 1$
  for Each example $\langle \mathbf{x}_1, y_1 \rangle \ldots \langle \mathbf{x}_N, y_N \rangle$ **do**
    $f = 0$
    for $l = \{1 \ldots L\}$ **do** ▷ Collect base learner predictions
      if $t \equiv 0 \mod p$ **and** $h_l(y) \neq y_t$ **then** ▷ Decrease weight on error
        $\alpha_l \leftarrow \beta \alpha_l$
      end if
      $f \leftarrow f + \alpha_l h_l(\mathbf{x})$
    end for
    $\hat{y} \leftarrow sign(f)$
    if $t \equiv 0 \mod p$ **then**
      $\alpha \leftarrow \frac{\alpha}{\max_l \alpha_l}$ ▷ Normalise weights
      for all $l : \alpha_l < \theta$ **do** ▷ Prune base learners
        Remove h_l from the ensemble
        $L \leftarrow L - 1$
      end for
      if $\hat{y} \neq y_t$ **then** ▷ Add new learner on ensemble error
        $L \leftarrow L + 1$
        Create h_L, new base learner
        $\alpha_L \leftarrow 1$
      end if
    end if
    for $l = \{1 \ldots L\}$ **do** ▷ Train base learners
      Train h_l on $\langle \mathbf{x}_t, y_t \rangle$
    end for
  end for

execution time, but smaller values will adapt more aggressively to changes. The discount parameter determines the degree to which learners are 'punished' for making mistakes. There is subtle interplay between the weight normalisation step and the weight discounting and pruning parameters: note that, without normalisation, DWM would behave poorly in difficult learning scenarios, since all the base learners would make errors, have their weight reduced, and be pruned, frequently. Instead, normalisation ensures that only learners who perform significantly worse than the best learner will be pruned.

AddExp [44] is similar to DWM, the essential difference being that α_L (the weight of a new learner) is chosen as a proportion of the *sum* of learner weights; in this sense, AddExp can be seen as equivalent to DWM but where normalisation sets $\sum_{l=1}^{L} \alpha_l = 1$ instead of $\max_l \alpha_l = 1$. An analysis of AddExp shows how its performance compares to learners that are trained on only part of the data. Specifically, for any partition of the data, AddExp achieves an error that differs by a coefficient from a learner trained *only* on that partition. Furthermore, when pruning is implemented in AddExp, it is possible to bound the maximum size of the ensemble, while DWM ensembles could theoretically grow indefinitely.

Bach and Maloof's Bayesian conditional model comparison (BCMC) algorithm [2] works on a similar premise, by creating a new base learner at every time step, and then calculating a probability distribution over time steps when a concept change could have occurred. This distribution over possible change points is determined by the performance of the base learners training on data starting at those change points. In producing an ensemble prediction, base learners are weighted according to the probability that a change occurred immediately before they were created.

These approaches — DWM, AddExp and BCMC — all use ensembles to achieve the same goal: to approximate the prediction of a single base learner trained on only data that occurred since the most recent change point. This purpose differs to the common reason for applying ensembles in stationary learning problems, which is to improve performance over an individual model.

Various boosting algorithms have been developed for non-stationary learning [32, 67, 69, 80]. The general approach is modelled upon online boosting for incremental, stationary learning [68], but with modifications to the weighting strategy and the addition or removal of base learners.

Oza described two important algorithms for incremental learning [68]: Online

Bagging and Online Boosting. In both cases, it was shown that the models learned by the online algorithms converge[5] to their batch equivalents for large N. We describe both algorithms here.

Algorithm 6 Online Bagging

Require: Ensemble size: L
Create base learners: $h_1 \dots h_L$

For each incoming example $\langle \mathbf{x}, y \rangle$:
for each learner l **do**
Sample $k \leftarrow Poiss(1)$
Repeat k times: train h_l on $\langle \mathbf{x}, y \rangle$
end for

Prediction:
$H(\mathbf{x}) = sign\Big(\sum_{l=1}^{L} h_l(\mathbf{x}) \Big)$

The Online Bagging algorithm (Algorithm 6) trains each base learner on each example $Poiss(1)$ times. When training with batch Bagging, examples are sampled with replacement, so that it is possible for an example to occur multiple times in the dataset, or exactly once, or not at all. As $N \to \infty$, the probability distribution for this number approaches a Poisson distribution.

Online Adaboost is structured in a similar fashion, processing examples sequentially and training each learner on the same example some number of times. However, instead of using 1 as the parameter to the Poisson distribution, Online Adaboost computes an example weight in a way that corresponds to the weighting procedure of batch Adaboost. We show Online Adaboost in Algorithm 7. Achieving a good approximation with Adaboost is much harder than for Bagging, however, since, for example, determining the weight assigned to the first example after training on h_1 requires us to know α_1 — in batch Adaboost α_1 depends on the performance of h_1 on *all* the data. For this reason, Online Adaboost tends to perform significantly worse than batch Adaboost [68].

Online Non-Stationary Boosting (ONSBoost) [69] is a modification of Online Boosting [68] which uses a window of recent examples to guide the replacement of poorly performing ensemble members. With user-defined frequency, a floating search is performed to identify the base learner which, when removed from the

[5]The convergence for Online Adaboost was only proven when the base learners were Naïve Bayes.

Algorithm 7 Online Adaboost

Require: Ensemble size: L
Create base learners: $h_1 \dots h_L$
Set correct/wrong sums $\lambda_1^{\text{sc}} \dots \lambda_L^{\text{sc}}, \lambda_1^{\text{sw}} \dots \lambda_L^{\text{sw}}$ to 0.

For each incoming example $\langle \mathbf{x}, y \rangle$:
$\lambda \leftarrow 1$
for each learner l **do**
  Sample $k \leftarrow Poiss(\lambda)$
  Repeat k times: train h_l on $\langle \mathbf{x}, y \rangle$
  if $h_l(\mathbf{x}) = y$ **then**
    $\lambda_l^{\text{sc}} \leftarrow \lambda_l^{\text{sc}} + \lambda$ ▷ Update total correct weight
    $\epsilon_l \leftarrow \frac{\lambda_l^{\text{sw}}}{\lambda_l^{\text{sc}} + \lambda_l^{\text{sw}}}$ ▷ Update weighted error
    $\lambda \leftarrow \frac{\lambda}{2(1-\epsilon_l)}$ ▷ Update example weight
  else
    $\lambda_l^{\text{sw}} \leftarrow \lambda_l^{\text{sw}} + \lambda$ ▷ Update total wrong weight
    $\epsilon_l \leftarrow \frac{\lambda_l^{\text{sw}}}{\lambda_l^{\text{sc}} + \lambda_l^{\text{sw}}}$ ▷ Update weighted error
    $\lambda \leftarrow \frac{\lambda}{2\epsilon_l}$ ▷ Update example weight
  end if
  $\alpha_l \leftarrow \log \frac{1-\epsilon_l}{\epsilon}$ ▷ Update learner weight
end for

Prediction:
$H(\mathbf{x}) = sign\Big(\sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}) \Big)$

ensemble, reduces ensemble error. If such a classifier exists, it is replaced with a new base learner. This replacement strategy ensures that base learners are replaced if they became irrelevant to the current concept. However, in boosted ensembles there is a hazard associated with base learner removal, since the *order* of base learners during training is important — so removing a single base learner can impact the way data is weighted for all subsequent learners.

IBoost [32] also uses a window of recent examples. When new data arrives, the window is updated and used to modify learner weights in a similar fashion to Online Adaboost. Additional learners may be added at a fixed frequency, with a maximum ensemble size that is enforced by removing a learner. Rather than using a floating search like ONSBoost, IBoost always removes the learner with the lowest weight.

Learn^{++} [67] and KBS-Stream [80] operate with *batches* of data. In some applications, this is a natural property of the task — for example, an insurance risk prediction system may be trained at the end of each working day — while in other cases, sequential data would need to be collected and buffered to create batches. In both these cases, it is assumed that data *within* each batch is stationary.

The original Learn^{++} [71] algorithm was designed for stationary data; it trains a new ensemble using a boosting-like algorithm on each batch of data. Predictions are made using *all* ensembles from previous batches. For dealing with non-stationary problems, Learn^{++}.NSE [24] adds only one learner after each batch. All learners are evaluated on *every* batch, and error rates on all previous batches are stored; the final prediction is computed with:

$$H(\mathbf{x}) = \arg\max_{y} \sum_{l=1}^{L} \delta[h_l(\mathbf{x}) = y] \log \frac{1}{\sum_{j=0}^{L-l} w_l^{L-j} \frac{\epsilon_l^{L-j}}{1-\epsilon_l^{L-j}}} \tag{4.19}$$

where w_l^j is a coefficient which discounts historic weight for the l^{th} learner from the j^{th} data batch sigmoidally, and ϵ_l^{L-j} is the weighted error of base learner h_l on the j^{th} batch of data. This weighting system is similar to that of Adaboost, but it applies more emphasis to examples from recent batches of data.

The KBS-Stream algorithm [80] considers two possibilities after each batch of data: either the new batch is from the same distribution as the previous one, or it is from a different distribution. Therefore, KBS-Stream maintains two ensembles; one ensemble is trained only on the most recent batch of data, and the other has

been trained on all batches since the last 'change'. By comparing these ensembles' performance on the most recent batch, KBS-Stream can ensure that the ensemble benefits from multiple batches of data when the distribution is stationary, but is replaced when the distribution changes. Additionally, the base learners within each ensemble are reweighted, and a new base learner is added, after each batch.

Note that Learn^{++} and KBS-Stream both rely on diversity in their ensembles, to some extent: old learners are not retrained at any point, and so their diversity determines how effectively the ensemble predictions can change when their weights are updated; this is not explicitly discussed in the literature, but it is an example of part of our motivation for investigating diversity in non-stationary learning; we examine this in more detail in Section 5.2.

4.4 Diversity in Non-Stationary Learning

DDD [66] is an ensemble algorithm for non-stationary learning that specifically uses diverse ensembles. It works on the same assumptions that we will discuss in Chapter 5 — regarding diverse learners as being better positioned to adapt to novel concepts — where diversity is created using a variant of Online Bagging [68]. Up to four ensembles are created and trained, and complex heuristics govern what the overall prediction should be, and how to organise the ensembles during training. A change detection algorithm based on EDDM [3] is used to guide the switching between ensembles. The motivation behind DDD is presented empirically, with an investigation of how diversity affects adaptation in Online Bagging ensembles [65], and after deriving the algorithm, Minku evaluates it extensively [64, 66].

Overall, DDD achieves performance that is competitive with other state-of-the-art algorithms, can be applied to a wide range of learning problems, has a modular design that is agnostic with respect to base learning algorithm, the ensemble algorithm (as long as it can run in both low and high diversity modes), and the change detection algorithm. The sophisticated training strategy also provides robustness against false change detection, and is able to distinguish between several *types* of drift and react in an empirically justified way.

Since DDD is the only algorithm of which we know that operates on the premise that diversity is important in non-stationary learning, we aim to summarise Minku's work here, so as to be able to indicate where our work is similar

or different.

4.4.1 Motivation

As we discussed in Section 2.3, the role of diversity is reasonably well studied (if not so well understood) in normal supervised learning tasks. Minku et al. [65] motivate an empirical study into diversity with two observations; firstly, that *"no study of the role of diversity in the presence of concept drift has ever been done"*, and secondly because:

> *"the literature does not contain any deep study of why [ensembles] can be helpful for [concept drift] and which of their features can contribute or not to deal with concept drift. Diversity could be expected to be one of the features that help in dealing with concept drifts when using ensembles."*

The subsequent empirical study shows, to some extent, that diverse ensembles adapt better to new concepts (we discuss the study in more detail in the next section). Based on these empirical observations, they develop the DDD algorithm, which we describe in Section 4.4.3.

4.4.2 Controlling Diversity in Online Bagging

The method proposed by Minku et al. for controlling the diversity of incrementally trained ensembles is based on a small modification to Online Bagging [68]. The ensemble is trained in a similar fashion to normal Online Bagging, with a number of duplicated examples (k) sampled from a Poisson distribution. However, instead of consistently using 1 as the parameter to that distribution, a user-specified parameter, λ, is used. Setting $\lambda = 1$ gives the original Online Bagging algorithm, while smaller values of λ will decrease the similarity in each base learner's training set (in principle, $\lambda > 1$ could also be used to reduce diversity further). The proportion of the dataset 'seen' by each base learner in batch Bagging is well known:

$$1 - \left(\frac{N-1}{N}\right)^N, \tag{4.20}$$

since each learner is trained on N samples, and each sample has a $\frac{N-1}{N}$ probability of *not* being $\langle \mathbf{x}_i, y_i \rangle$ for some specific i. As $N \to \infty$, this probability approaches

the probability of not drawing 0 from a $\lambda = 1$ Poisson distribution[6]:

$$Pr(Poiss(\lambda) = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \tag{4.21}$$

$$\sum_{k=1}^{\infty} Pr(Poiss(\lambda) = k) = 1 - Pr(Poiss(\lambda) = 0), \tag{4.22}$$

$$= 1 - e^{-\lambda}. \tag{4.23}$$

A final point that we note regarding the Poisson distribution is its expectation:

$$E[Poiss(\lambda)] = \lambda. \tag{4.24}$$

Minku et al. show, by training ensembles with various choices of λ, that there is a strong correlation between λ and D_{Q}, the Q statistic diversity measure. For this reason, future experiments use λ as a *proxy* for diversity. Given the strength and monotonicity of the relationship between λ and D_{Q}, the idea that ensemble diversity can be controlled in Online Bagging using the λ parameter is well supported by the results.

Here we briefly identify three important issues in the use of Bagging with a modified λ parameter. Firstly, the algorithm relies upon *training data dissimilarity* to promote diversity, and this dissimilarity is achieved primarily by training base learners on datasets that are close to disjoint — however, note that if training data is i.i.d., diversity will converge to 0 (or, D_{Q} will converge to 1) as N increases. The actual convergence is probably not much of a concern — for small λ, it can require very large N — but, as we show in Figure 4.4, the similarity between base learners increases with N, such that relationship between λ and D_{Q} depends on N. However, this is not such an issue on non-stationary data, where changes in concept prevent a high degree of convergence.

A second observation regarding Online Bagging is that changing λ changes the *amount* of training data seen by each base learner; in fact, each base learner sees on average $E[Poiss(\lambda)]N$ examples, which by Equation 4.24 is λN. This is significant in Online Bagging since some of the values of λ used in Minku's experiments are very low[7] — in the most extreme case, each base learner was trained on 1 out of every 2000 examples. One implication of this on smaller

[6]This result holds by examining Equation 4.20 in the limit of large N. See Oza [68, Section 3.4] for more detailed and general results.

[7]For initial experiments, $\lambda \in \{0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$

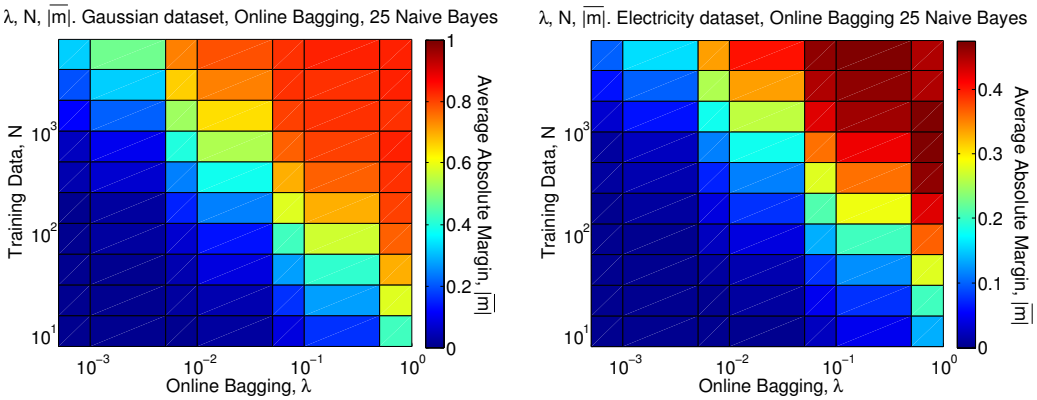

Figure 4.4: We train Online Bagging ensembles with varying amounts of training data and varying values of λ. Diversity ($\overline{|m|}$) is affected by both λ and N.

datasets is that many base learners will never be trained on any data[8].

Another issue occurs if λ *changes*; for lossless base learners, their 'learning rate' is determined by how much data they have been trained on; for example, Equation 4.7 shows that, for Naïve Bayes base learners, the change to the estimate of class conditional mean involves a denominator of $N(y)$ — the total number of y labelled examples seen so far. If λ remains constant, then this learning rate is constant too — for lower λ, each example will affect the base learner more, but fewer examples are seen, and these two factors balance. If λ were *changed* during training, this would either increase or decrease the learning rate; in the subsequent section we will discuss what impact this has for DDD.

So, what is the implication of these three issues for generating diversity ensembles with Online Bagging? The first issue, regarding the relationship between diversity and N, is unlikely to be too problematic — there is no noticeable effect in Minku's experiments, probably due to the use of small amounts of training data and fairly complex datasets.

The second issue — amount of training data seen by each base learner — seems to be significant in Minku's initial experiments [65]. The datasets used (See [65, Table 3]) are UCI datasets; concept change is simulated by changing class labels. In many of the artificial drifts, *more than* 50% of the examples have their labels changed; in such cases, high accuracy on the initial concept is *detrimental* on the second concept. This will, in general, favour learners with very low accuracy immediately after a concept change. An example of this occurs

[8]Note that in some cases, untrained base learners return *random* predictions, in which case untrained learners are always 'high diversity'

in [65, Figure 9], where some of the plots show Online Bagging with very low λ achieving approximately the error rate of random guessing throughout.

In the next section, we introduce DDD, and discuss how the third issue we identified here — the impact of changing λ — is exhibited in DDD.

4.4.3 Diversity for Dealing with Drifts (DDD)

Diversity for Dealing with Drifts (DDD) [66] is an ensemble algorithm that works on the premise that diverse ensembles are able to adapt more quickly to concept changes. The essential components of the algorithm are:

1. Maintain various ensembles that have high or low diversity and have been trained on varying amounts of data.

2. Detect concept changes with a stand-alone change detection algorithm.

3. Use switching heuristics based on change detection and error estimation to decide when to create/replace ensembles.

4. Make predictions either with a single ensemble, or as a weighted vote, again based on heuristic rules.

The full algorithm is presented in [66], and extensively evaluated in [64]. The goal of the algorithm can be stated quite simply though: *a diverse ensemble is maintained, and where appropriate used to adapt more quickly to a concept change*. Many of the complexities of the algorithm are designed to correctly identify situations where the diverse ensemble will be useful, based on insights from previous experiments. The issue of very high severity concept changes was alluded to in these experiments [64, Section 5.4.2]:

> *"... the new concept has almost no similarities to the old concept. So, an ensemble which learnt the old concept either partly or fully will not be so helpful (and could even be harmful) for the accuracy on the new concept."*

This acknowledgement and further analysis of DDD on a wider range of low severity concept changes, as well as on real data where concept change is expected to be fairly gradual, suggests that the algorithm does more than just exploiting the poor performance of very diverse ensembles. However, at this point we highlight

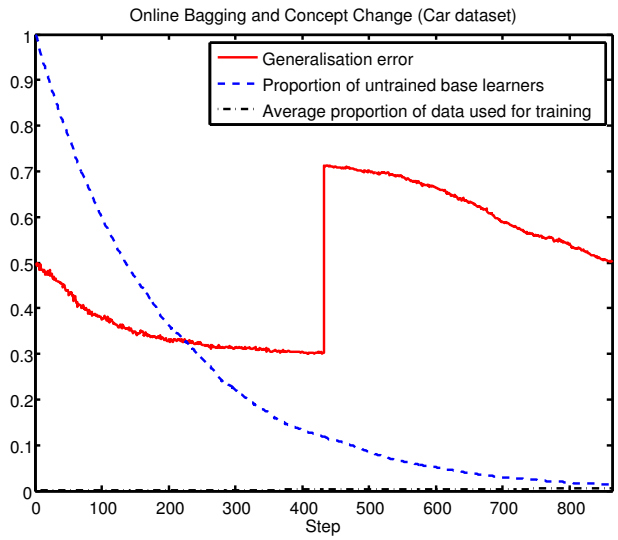

Figure 4.5: Online Bagging (25 Naïve Bayes) with $\lambda = 0.005$ trained on the car dataset. The proportion of untrained base learners decreases according to Equation 4.25, while the average proportion of data that each base learner is trained on remains at about $\frac{1}{200}$.

the third issue that was introduced in the previous section: essentially *the learning rate of lossless base learners depends on the amount of data they were trained on.*

Many of the experiments in [64, Section 6.3] use either Naïve Bayes or ITI base learners, although MLPs are also used in some cases. Because of the losslessness of NB and ITI, it would be expected that in situations where diversity is actually exploited by DDD — where an ensemble that had high diversity on the previous concept is then trained with low diversity on the new concept — the effect would be similar to training the base learners on a small number of examples from the previous concept, and then many examples on the new concept. Obviously this would compare favourably to situations where base learners see approximately N examples from the previous concept; adaptation to the new concept requires new examples to form a large proportion of the total data seen by the base learner.

In Figure 4.5, we show a few quantities of interest in Online Bagging ($\lambda = 0.005$) on the modified Car dataset[9] [65]. Specifically, we count the *proportion* of base learners that are completely untrained. For arbitrary λ, the probability

[9]Apart from the generalisation error, the other values shown in Figure 4.5 are base learner and data independent.

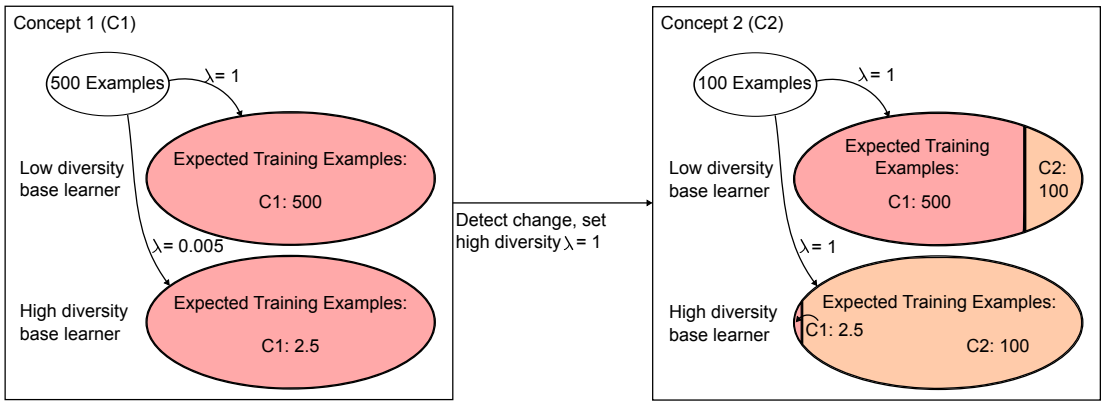

Figure 4.6: Illustration of the relative amounts of data from different concepts in an example scenario.

that a base learner is untrained after t examples is:

$$Pr(Poiss(\lambda) = 0)^t = e^{-\lambda t}, \tag{4.25}$$

which is therefore also the expected proportion of base learners that are untrained after t examples (the blue dashed line in Figure 4.5). Similarly, we know that the expectation of a Poisson distribution is λ, and so the proportion of training examples that each base learner has been trained on is also λ (shown on the black dot-dash line in Figure 4.5).

The impact of these phenomena occurs when a change is detected in DDD; the λ parameter in the 'old high diversity' ensemble is changed to 1, so that it can become the 'new low diversity' ensemble. The adaptation of this ensemble to the new concept is the essential part of 'diversity in non-stationary learning' that motivates the whole algorithm. However, we would attribute the behaviour to other aspects of the state of the 'old high diversity' ensemble — specifically, its base learners are trained on a very small amount of data.

For example, consider the scenario illustrated in Figure 4.6, where a stationary concept experiences an abrupt shift at $t = 500$. At this point, base learners in a diverse ensemble with $\lambda = 0.005$ have been trained on an average of 2.5 examples, while learners in a non-diverse ensemble ($\lambda = 1$) have been trained on 500. We set λ in the high diversity ensemble to 1, and then data from the new concept arrives. At $t = 600$, base learners in the formally diverse ensemble have seen on average 2.5 examples from the old concept and 100 from the new concept, while base learners in the non-diverse ensemble have seen 500 from the old concept

and 100 from the new. If the base learners are *lossless* then the data from the previous concept will dominate the behaviour of the low diversity ensemble.

We believe this effect is likely to be responsible for much of the success of DDD, along with sensible heuristics that decide when it is appropriate to keep the low diversity ensemble from the previous concept (i.e. when there is not much concept change) or to create an entirely new ensemble (i.e. when concept change is severe). In fact, Minku observes [64, Section 6.3.4] that:

> *"...DDD obtained worse accuracy than DWM during the first and last thirds of the learning when using MLPs, but similar (or slightly better) when using NB."*

MLPs are lossy incremental learners (they are trained for a single epoch with a fixed learning rate), so they will adapt to a new concept at the same rate regardless of how many examples they have previously seen (their rate of adaptation might be affected by *what* the previously seen data was, but not *how much* of it there was). This observation supports the hypothesis that a significant part of DDD's success is related to the amount of data that base learners are trained on, rather than the diversity of the ensembles.

We emphasise at this point that our criticism is not of the DDD algorithm — in fact, our analysis here provides further support for the idea that DDD *should* adapt quickly to new concepts. However, what we have illustrated is that the λ parameter in Online Bagging does not control diversity in isolation, but rather affects several factors within the model. It seems very likely that one of these factors — the amount of training data — accounts for most or all of the success that Minku attributes to 'diversity'; there is considerable study of *windowing* algorithms in the literature, so it is well known that models adapt quickly when they are only trained on a small amount of data from a previous concept.

4.5 Summary

In this section, we have presented some important issues and approaches relating to non-stationary learning tasks. We have described various assumptions, and shown how these assumptions relate to stationary, incremental and non-stationary problems. We described some algorithms that can be applied to incremental and non-stationary scenarios, highlighting the idea of *lossless* learners. We then discussed algorithms that specifically exploit the idea of an *ensemble* for adaptation.

Finally, we analysed the DDD algorithm in detail, since it is highly relevant to the content of Chapter 5.

Although DDD is presented as 'diversity for dealing with drifts', we have actually suggested that *diversity* is not the main factor in determining adaptation of ensembles to new concepts. However, our criticism does not extend to the performance of DDD; empirical evaluations show that it performs competitively on a wide range of learning problems, and in fact our analysis of the algorithm here has provided *theoretical support* for the success of DDD. However, because of this alternative explanation that we supply for the success of DDD, it is no longer clear whether:

1. DDD is successful because of the effect of λ on the amount of training data supplied to each base learner, *and* because of the effect of λ on ensemble diversity, or

2. DDD is successful because of the effect of λ on the amount of training data supplied to each base learner, and *diversity is inconsequential*.

This uncertainty over the exact conclusions that should be drawn from DDD and the associated investigations provides one good reason for further investigating diversity in non-stationary learning. Additionally, our contributions in Chapter 3 have opened new possibilities for studying diversity; specifically, we can consider how *margins* interact with non-stationarity, and derive further theoretical predictions about what role diversity plays in non-stationary learning.

Chapter 5

Managing Diversity for Non-Stationary Learning

5.1 Introduction

In this chapter, we apply our insights from Chapter 3 to investigate the role of diversity in non-stationary learning. Our approach consists of:

1. Using the connection between diversity and margins to consider how we *expect* diversity to affect adaptation in non-stationary learning problems.

2. Deriving an algorithm that explicitly manages ensemble diversity using the margin distribution.

3. Using a similar technique to Oza [68] so that we can apply our diversity-managing algorithm to incremental problems.

4. Developing a method of exploiting diversity in non-stationary learning, similar to Minku's DDD algorithm [66].

5. Examining the behaviour of our algorithm on toy and real-world problems.

Our final goal — to investigate the role of diversity in non-stationary learning — is identical to that of Minku [64]. In Section 4.4, we described Minku's approach in detail, concluding that its success should not be entirely attributed to *diversity*, but that it exploited other useful effects when adapting to new concepts. In our analysis, we build on Minku's work by giving explicit consideration to issues

such as our initial motivation and expectations, and performing further empirical analysis in scenarios where the effects of other factors are controlled.

Due to the comparative simplicity of the 'abrupt concept change' scenarios considered by many authors [2, 44, 52, 65], our analysis only extends to these scenarios. Based on our theoretical results in Section 5.2, we are only able to see an obvious potential benefit to diversity in situations of abrupt or very fast drift, and since our experimental results suggest that realising this advantage is very difficulty even in problems with abrupt change, we do not investigate our hypotheses in gradual concept change scenarios.

5.2 Motivation for Using Diversity in Non-Stationary Learning

Our primary hypothesis is that *ensembles with higher diversity are better positioned to adapt to new concepts.*

Diversity in non-stationary learning has only been discussed in previous literature by Minku [64]:

> "... in offline mode, diversity among base learners is an issue that has been receiving lots of attention ... Many authors believe that the success of ensembles algorithms depends on both the accuracy and the diversity among base learners. However, no study of diversity has even been done in online changing environments."

While Minku adeptly identifies diversity in non-stationary learning to be of interest, he does not actually present any motivational reason for why we might expect diversity to be of special importance in these situations.

5.2.1 Intuitive Argument

We first discuss a qualitative interpretation of diversity in non-stationary learning, without alluding to the correspondence between diversity and margins, or attempting to quantify adaptivity. We do this by considering how ensembles will behave if they encounter a concept change where the new concept is *unrelated* to the old one.

Non-Diverse Ensembles

In a low-diversity ensemble, base learners are very similar, and there is necessarily a high degree of similarity between the base learner predictions and the ensemble predictions. Therefore, if the ensemble is irrelevant for predicting a new concept, so are its base learners. In order for adaptation to occur, a majority of the ensemble will either need to change or be replaced.

Diverse Ensembles

When the ensemble is diverse, a change of concept that renders its predictions irrelevant does not necessary imply that the base learners themselves are no longer useful. Because base learners are, in a sense, 'spread out', it is possible that some will already be useful for solving the new concept. This intuitively seems like a strong position for the ensemble to be in — adapting to the new concept should require less updating/reweighting/replacement of base learners.

In fact, some concept change algorithms already implicitly exploit the diversity of their base learners. For example, in KBS-Stream [80], base learners are never retrained; only the voting weights change. If there was no diversity, then changing the learner weights would not be sufficient to adapt to new concepts.

5.2.2 Quantitative Argument

We now discuss what changes must occur in an ensemble to cause 'adaptation', and how diversity affects these changes.

When a concept change occurs, learners suffer because they make suboptimal predictions on the affected data points. When we talk about 'adapting to a change', we mean that these predictions should be changed as quickly as possible. Hence, we can describe 'adaptation' as:

$$sign(m_i) \neq sign(m_i'), \tag{5.1}$$

where m_i is the original margin and m_i' is the 'adapted' margin.

There are two ways in which the margin can change — either base learners change their predictions, or the voting weights change. Both of these affect the

margin in basic ways:

$$h_l(\mathbf{x}_i) \neq h_l(\mathbf{x}_i)' \implies m_i' = m_i + 2\alpha_l y_i h_l(\mathbf{x}_i)', \tag{5.2}$$

$$\alpha_l \neq \alpha_l' \implies m_i' = m_i + (\alpha_l' - \alpha_l) y_i h_l(\mathbf{x}_i). \tag{5.3}$$

In the first case, a learner changes its prediction, so the margin changes by some amount. Similarly, in the second case a learner weight is changed, and therefore that learner's impact on the prediction changes.

From this we can see that the margin gives us a natural way of expressing the *amount of change necessary to 'swing' the prediction*. If $|m_i|$ is large, then changing the sign of m_i requires many learners to change their prediction, and/or a large change to be made to learner weights. For a small $|m_i|$, a few learners changing predictions, or a small change to learner weights, is sufficient to alter the sign.

However, while this provides us with good motivation to investigate how diversity affects adaptation, it does not *necessarily* mean that high diversity will improve adaptation. This would be the case if the amount of change to h_l and α_l were constant — however, in practice we might expect these changes to depend themselves on the magnitude of the margins; for example, Adaboost will implement larger updates when m_i is negative, due to the steepness of the exponential loss. Therefore, diversity is not a factor that we should consider in isolation; there will be interactions between it and other aspects of the ensemble learning algorithm which may diminish (or amplify) its efficacy.

5.3 Diversity Optimisation via Quadratic Loss

In this section, we describe our 'DivBoost' algorithm, which manages diversity in incremental learning. Figure 5.1 compares various margin-based loss functions: in general, such functions are *monotonic*, i.e. they decrease as the margin increases. This is seen as appropriate since large margins can imply strong generalisation [76, 92]. The relationship between diversity and margins makes it clear that *higher diversity occurs when margins are closer to 0* — so a loss function that promotes high diversity should have a minimum that is close to 0 – and hence, be *non-monotonic*. Therefore, we first examine the gradient descent interpretation of boosting, and modify the existing framework to allow non-monotonic loss

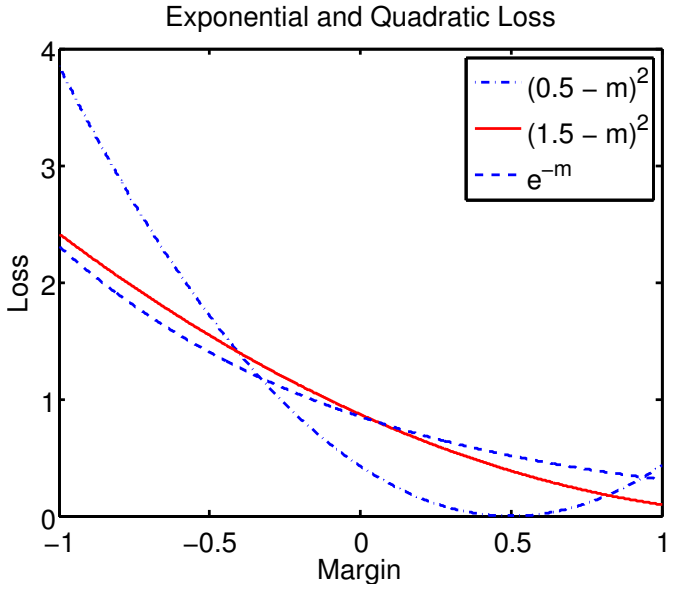

Figure 5.1: We show various loss functions (scaled for comparison). The solid line, exponential loss, is used by Adaboost and encourages large margins; however, to explicitly promote diversity, we need to encourage margins to take specific values, closer to 0. Quadratic loss can achieve this, since we can set the minimum to any value.

functions. Next, we present a loss function that permits diversity to be managed by a parameter. Finally, we extend the algorithm for online learning.

5.3.1 Non-Monotonic Loss in AnyBoost

In Section 2.2.4, we described the AnyBoost algorithm, which performs gradient descent in functional space to train a weighted ensemble that optimises a convex monotonic loss function. Example and base learner weight updates are effected by two equations:

$$D_l(i) = \frac{\frac{\partial}{\partial m_i} C(m_i)}{\sum_{j=1}^{N} \frac{\partial}{\partial m_i} C(m_i)}, \tag{5.4}$$

$$\frac{\partial}{\partial \alpha_l} \frac{1}{N} \sum_{i=1}^{N} C(m_i) = 0. \tag{5.5}$$

However, the AnyBoost framework is only meaningful when $C(m_i)$ is a convex *monotonic* function. Mason et. al. reason that any "sensible cost function of the margin will be monotonically decreasing" [62, Section 2.4], with this intuition

having implications in the derivation of Equation 5.4. While it is indeed sensible to accommodate only monotonic loss functions in many learning scenarios, our plans for optimising diversity suggest that the loss function should be *non-monotonic* with respect to the margin; this requires us to modify Equation 5.4 thus:

$$D_l(i) = \frac{-\frac{\partial}{\partial m_i} C(m_i)}{\sum_{j=1}^{N} |\frac{\partial}{\partial m_i} C(m_j)|} \propto -\frac{\partial}{\partial m_i} C(m_i), \tag{5.6}$$

since the negative gradient implied by monotonic loss had previously allowed the negative in the numerator to cancel the absolute in the denominator.

5.3.2 Incorporating Diversity in a Loss Function

We first need to find a loss function that will interpolate between accuracy (large margins) and diversity (small absolute margins) according to a parameter μ. One solution is to have a function that is convex with a minimum at μ; we will examine quadratic loss, since its derivatives have convenient forms. We show various quadratic losses and compare them to exponential loss in Figure 5.1.

$$C(m_i) = \frac{1}{2}(\mu - m_i)^2. \tag{5.7}$$

Computing the necessary derivatives gives:

$$\frac{\partial}{\partial \alpha_l} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2}(\mu - m_i)^2 = 0, \tag{5.8}$$

$$\alpha_l = \sum_{i=1}^{N} y_i h_l(\mu - y_i \sum_{l=1}^{L-1} \alpha_l h_l(\mathbf{x}_i)), \tag{5.9}$$

for the learner weights, and:

$$D_l(i) \propto -\frac{\partial}{\partial m_i} \frac{1}{2}(\mu - m_i)^2 = \mu - m_i, \tag{5.10}$$

for example weights. The offline version of DivBoost, therefore, is obtained by using these update equations within the AnyBoost framework.

5.3.3 Online DivBoost

We now modify this offline version of DivBoost to learn incrementally, using techniques analogous to those in Online Adaboost (See Algorithm 7) [68]. This algorithm is simpler than its Online Adaboost counterpart, due to the linearity of the derivative of the loss: updates to the learner and example weights are additive at each step. We use a Poisson distribution instead of weighted sampling, since this removes the need for normalising example weights. Similarly, learner weights are only maintained up to proportionality, but this gives the same result (We normalise them when making predictions to give margins in $[-1, 1]$).

Hence Equation 5.9 becomes (we now index data with t, the current timestep):

$$\alpha_{l,t} \leftarrow \alpha_{l,t-1} + y_t h_l(\mathbf{x}_t)\big(\mu - y_t \sum_{k=1}^{l-1} \alpha_{l,t-1} h_k(\mathbf{x}_t)\big), \tag{5.11}$$

and Equation 5.10 is replaced by a variable λ, which is updated as:

$$\lambda_{l,t} \leftarrow \mu - y_t \sum_{k=1}^{l} \alpha_{k,t-1} h_k(\mathbf{x}_t). \tag{5.12}$$

such that, as in Online Adaboost, the only disparity between this and a batch learning version is due to changes in α and h over the course of training. In some cases, this can severely impact convergence of the algorithm, since good values of α_{l+1} and h_{l+1} depend on the stability of α_l and h_l.

Note that we substitute Equation 5.12 into Equation 5.11 in our definition of the algorithm, for brevity. We show our online gradient descent boosting algorithm in Algorithm 8.

μ can be chosen here based on the required diversity: $\mu = 1.5$ gives a monotonic loss for high accuracy/low diversity, while a value such as $\mu = 0.25$ gives a very diverse ensemble.

Unlike Online Adaboost, we allow λ to become negative, in which case we choose k based on $|\lambda|$ and invert the target value of y during training. This indicates a situation where the prediction on $\langle \mathbf{x}, y \rangle$ is *too* correct — something which can only happen with non-monotonic loss. Clearly, if the minimum of the loss function (μ) is less than 1, then sometimes the optimal base learner will need to reduce the margin.

The updates to α_l (base learner weights), $\hat{y}$ (prediction of $\langle \mathbf{x}, y \rangle$ by the first l

Algorithm 8 DivBoost(μ)

Require: $\mu \in \mathbb{R}$: Target margin, L: Ensemble size
Set $\alpha_l \leftarrow 0$ for all base learners

For each incoming example $\langle \mathbf{x}, y \rangle$:
Let $\lambda = \mu$, $\hat{y} = 0$
for each learner l **do**
  Repeat $Poiss(|\lambda|)$ times: train h_l on $\langle \mathbf{x}, sign(\lambda) \cdot y \rangle$
  $\alpha_l \leftarrow \alpha_l + \lambda y h_l(\mathbf{x})$
  $\hat{y} \leftarrow \hat{y} + \alpha_l y h_l(\mathbf{x})$
  $\lambda \leftarrow \mu - (\hat{y} / \sum_{k=1}^{l} \alpha_k)$
end for

Prediction:
$H(\mathbf{x}) = sign\Big(\sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}) \Big)$

learners), and λ (relative example weight) are straightforward due to the convenient form of the loss function.

As a pragmatic concern, we observe that if $\lambda = 0$ then no additional learning will occur. A similar event can happen in Adaboost, should the weighted error ever become 0. We handle this by resetting $\lambda \leftarrow \mu$ in this event.

5.3.4 How DivBoost can be applied in Non-Stationary Learning

Algorithm 9 Adaptive DivBoost

$H_{\text{low}} \leftarrow$ DivBoost (low diversity)
$H_{\text{high}} \leftarrow$ DivBoost (high diversity)
for $t \in 1 \ldots N$ **do**
  Train H_{low}, H_{high} on $\langle \mathbf{x}_t, y_t \rangle$
  if change detected **then**
    $H_{\text{low}} \leftarrow$ duplicate H_{high}
  end if
end for

In Algorithm 9, we define a simple methodology for exploiting the diverse ensembles produced by DivBoost in a non-stationary learning environment. Like DDD [66], the idea is to maintain multiple ensembles (in this case, only 2), predicting with one while training both. In our procedure, 'Adaptive DivBoost',

we always make predictions with a low diversity ensemble ($\mu = 1.5$) while training a separate high diversity ensemble ($\mu = 0.5$). A change detection algorithm (such as DDM [28]) monitors the predictions of H_{low}: when a concept change is detected, the low diversity ensemble is replaced by the diverse ensemble, with its μ parameter set to 1.5. In preparation for future drifts, we also *duplicate* the diverse ensemble and continue training it with $\mu = 0.5$.

By duplicating the diverse ensemble and continuing its training, we ensure that there is always a trained diverse ensemble that can be used if a change is detected; if we started to train a new diverse ensemble, then we might detect another concept change before it had been sufficiently trained.

Of course, an alternative would be to use DivBoost as the ensemble algorithm in DDD instead of Online Bagging. This could be an interesting future research direction, although our experimental conclusions suggest that we would not expect DDD with DivBoost to outperform DDD with Online Bagging. Similarly, we do not present Adaptive DivBoost as a competitive algorithm; an algorithm like DDD has far more useful heuristic built in to deal with a variety of concept change scenarios. The simplicity of DivBoost will be beneficial though, when we try to understand its behaviour in an empirical setting.

5.4 Experiments

5.4.1 Experimental Aims

In these experiments, we aim to investigate:

1. Does DivBoost behave as we expect (i.e. does it allow us to control the margin distribution, and hence diversity, without a large negative effect on accuracy)?

2. What kind of problems should we consider applying Adaptive DivBoost to?

3. How do other components of the learning system (i.e. base learner choice and change detection) affect the behaviour of Adaptive DivBoost?

4. Should we expect to achieve a measurable improvement in performance on real-world problems?

Our general experimental environment is described in Appendix A.

Learners and Parameters

We compare several algorithms in this section. The "Keep" algorithm is a simple instantiation of DivBoost with $\mu = 1.5$, with no accommodation for concept change. "Replace" is similar, but when it is notified of a change, the ensemble is replaced with a new, untrained DivBoost ensemble (still $\mu = 1.5$). Online Bagging is used as an alternative diversity-optimising algorithm [66]. We train an Online Bagging ensemble with $\lambda = 0.05$ prior to drift, when a drift is detected we set $\lambda = 1$ and continue training. Adaptive DivBoost is our algorithm as described in Section 5.3.4. For benchmarking against an existing state-of-the-art algorithm, it might seem appropriate to compare against DDD, but because there does not seem to be evidence that DDD *does* use diversity (see Section 4.4.3), we instead use the more popular and well-established DWM [45] algorithm.

We selected parameters that performed well in general, based on settings from other authors, but did not tune them to specific datasets; since there is no i.i.d.ãssumption, choosing parameters based on 'validation data' is futile (there is no fixed distribution from which to draw the data), and choosing parameters that perform well on the dataset would lead to overfitting; parameter selection in non-stationary learning is a challenging open problem. We used an ensemble size of 15 throughout. In DWM, we set weight decay $\beta = 0.9$, removal threshold $\theta = 0.2$ and period $p = 5$.

In most experiments (i.e. unless otherwise specified) we used multilayer perceptrons (MLPs) as base learners. We used sigmoidal transfer functions, 5 hidden units and 0.1 learning rate. To facilitate online learning, we trained for 1 epoch on each example — hence, the momentum parameter is inconsequential. We used the implementation from the Netlab toolbox.

We chose MLPs since their learning rate is independent of the amount of data; this allows us to investigate the effect of ensemble diversity on adaptation without confounding factors. In Section 5.4.4, we compare with Gaussian Naïve Bayes (NB) base learners; this will illustrate how dramatically the base learner can impact adaptation. In Section 4.4, we described the issues associated with base learners with quantity of data dependent learning rates. Since μ in DivBoost will affect the number of examples sent to each base learner, we do not attempt to evaluate the utility of diversity when using such base learners.

5.4.2 Does DivBoost Effectively Manage Diversity?

First, we perform two experiments with stationary data to establish the validity of DivBoost.

Hypothesis: *In DivBoost, margin distributions will be tightly clustered around μ, and the accuracy will not suffer substantially even for quite diverse ensembles.*

Procedure

Our first experiment produces cumulative distribution plots using DivBoost ensembles. We train DivBoost ensembles with various μ on stationary data from 4 UCI datasets. We then evaluate their margin distributions on a hold-out test set. We perform 100 repetitions for each value of μ, using half of the data for training and half for testing. Finally, we display cumulative margin distributions, aggregating the margins from all repetitions.

The second experiment compares the behaviour of DivBoost to an alternative diversity-promoting algorithm — Online Bagging. We organise the data as in the previous experiment. We vary the λ and μ parameters non-uniformly between 0.002 and 1.5 (the full range is $\{0.002, 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5\}$), generating 50 ensembles for each value and computing diversity and accuracy for each ensemble. We then plot the average diversity and accuracy within each λ or μ value. This allows us to demonstrate how accuracy is affected as diversity increases.

Results

In Figure 5.2, we show margin distributions produced by DivBoost. We see that μ controls the skew of the distribution well, and in some cases the clustering of the distribution is tight (e.g. breast cancer). In general, most of the distribution lies below the 'target' margin of μ; the distribution will depend on the difficulty of the learning problem, so easier tasks like breast cancer permit margins to be generally closer to μ, while distributions are more diffuse for difficult problems like diabetes.

In Figure 5.3, we show diversity and accuracy as μ and λ are varied in DivBoost and Online Bagging respectively. The results show that the explicit diversity optimisation effected by DivBoost brings significant accuracy benefits, as well as being able to encourage more diverse ensembles.

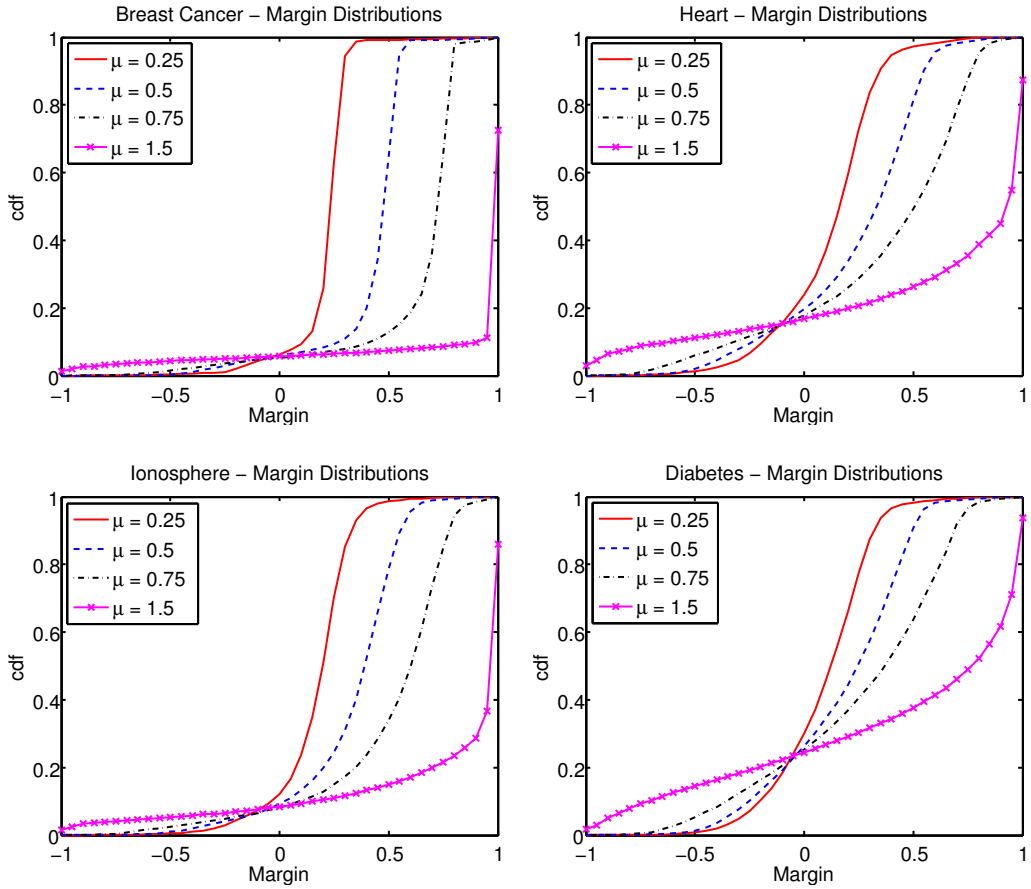

Figure 5.2: Cumulative margin distributions on generalisation data on 4 UCI datasets. Steep lines indicate tightly clustered distributions, with larger margins when the distribution is skewed towards the right.

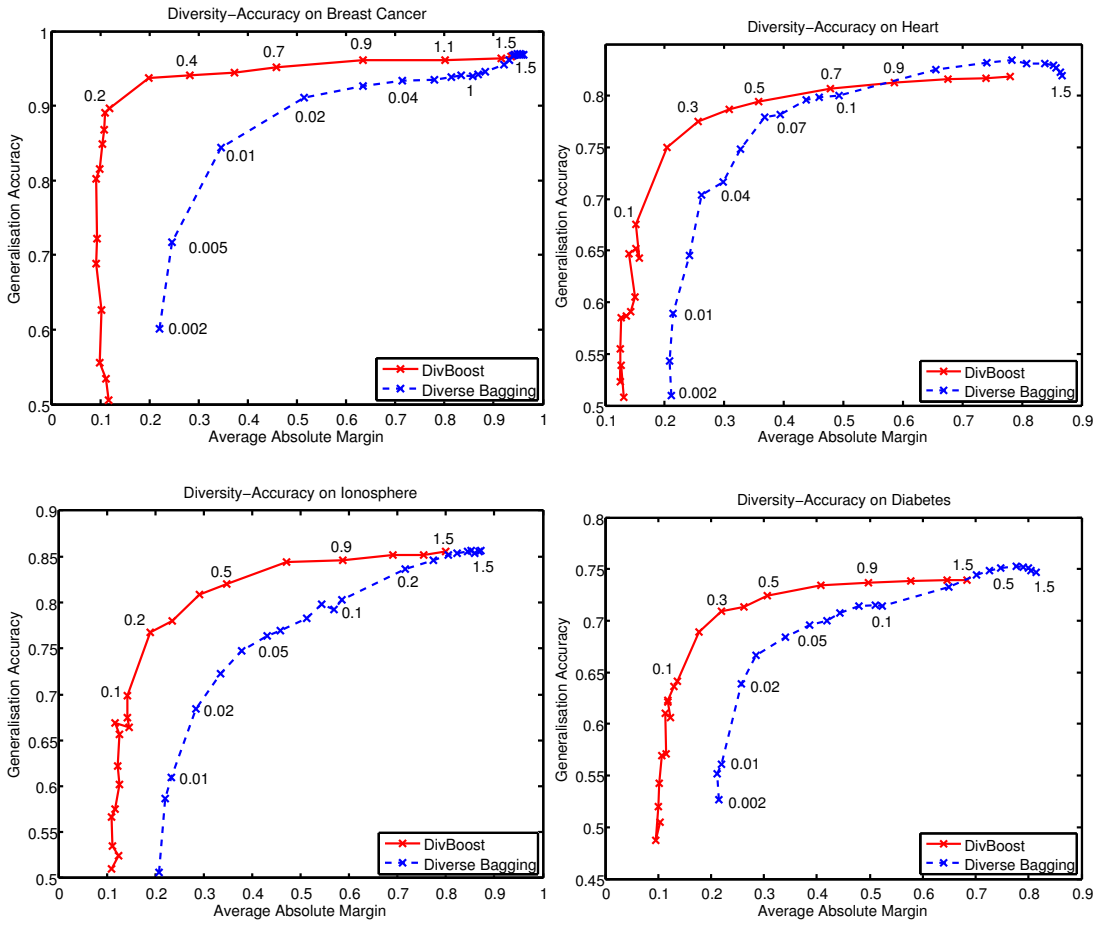

Figure 5.3: Average diversity and accuracy with varying μ or λ. The numbers on the plot indicate the μ or λ values used to generated those ensembles — they monotonically increase along the lines, so we omit some labels to avoid clutter.

Overall, we can see that the behaviour of DivBoost agrees with the theoretical premises from which it was derived. The μ parameter allows an efficient trade-off between diversity and accuracy, and (when evaluated on this criterion), seems to consistently achieve better accuracy that a recently-proposed alternative when training high-diversity ensembles.

5.4.3 Oracle Change Detection

To investigate our second and third experimental aims, we use an *oracle* to provide change detection information. This allows us to see how quickly DivBoost adapts given perfect change detection information. We vary the severity of concept drifts to delineate situations in which DivBoost has an advantage.

After this, we show the effect of using alternative base learners — specifically investigating the differences between Naïve Bayes and MLPs. We also change the oracle change detection to provide late concept change information, to establish whether poor change detection is likely to neutralise DivBoost's performance advantage.

Hypothesis: (from Section 5.2) *Ensembles with higher diversity are better positioned to adapt to new concepts*, and

Hypothesis: *The choice of base learner and efficacy of change detection will have a significant impact on the applicability of this approach.*

Procedure

First, we aim to delineate the applicability of DivBoost given perfect change detection. We use toy problems where we can control the position and severity of a single abrupt change (See Section A.1). We train DivBoost with high diversity on the 'initial concept'. We then simulate a concept change, adjust μ in DivBoost to 1.5 (i.e. low diversity), and evaluate the performance of DivBoost on hold-out data as it adapts to the new concept. We use UCI datasets, with 50% of the data in each concept. On the second concept, we only train on the first 50 examples, using the rest to estimate generalisation accuracy.

For comparison, we use the similar technique with Online Bagging instead of DivBoost (using a λ parameter of 0.05 for 'high diversity' and 1 for 'low diversity') — the effectiveness of this method is an essential aspect of the DDD algorithm, although DDD has other mechanisms that will improve over this performance in

some situations. We also compare with Keep and Replace.

Note that this experiment does not consider the accuracy of the ensembles before concept change — we only measure their performance as they adapt to the new concept.

Next, we vary the base learner used in the ensembles — this will show the difference between learners with constant learning rate (like MLPs), and learners with learning rates that depend on the amount of data that they have been trained on (like Naïve Bayes).

Finally, we simulate late change detection, simply by waiting for 10 time steps before making the appropriate changes (i.e. learner replacement, changes to μ and λ). This scenario is more likely in real problems, since change detection is not generally able to detect concept change instantly.

Results

In Figure 5.4, we show adaptation to new concepts after changes of varying severity. The main effect we are looking for is a difference between DivBoost and Keep; this would indicate that a trained *but diverse* ensemble has value immediately after a concept change. We see that extremes of very little concept change (12 overlap) and complete concept change (0 overlap) favour Keep and Replace respectively, since these correspond to scenarios where those approaches are optimal. However, in general, *either* Keep or Replace performs well, while Adaptive DivBoost retains consistent performance across various amounts of drift. Online Bagging tends to mix the behaviour of Adaptive DivBoost and Replace.

From this experiment, we conclude that DivBoost may be valuable both when concept changes are moderate in severity, or when change detection performs badly and is likely to generate a high proportion of false or missed detections.

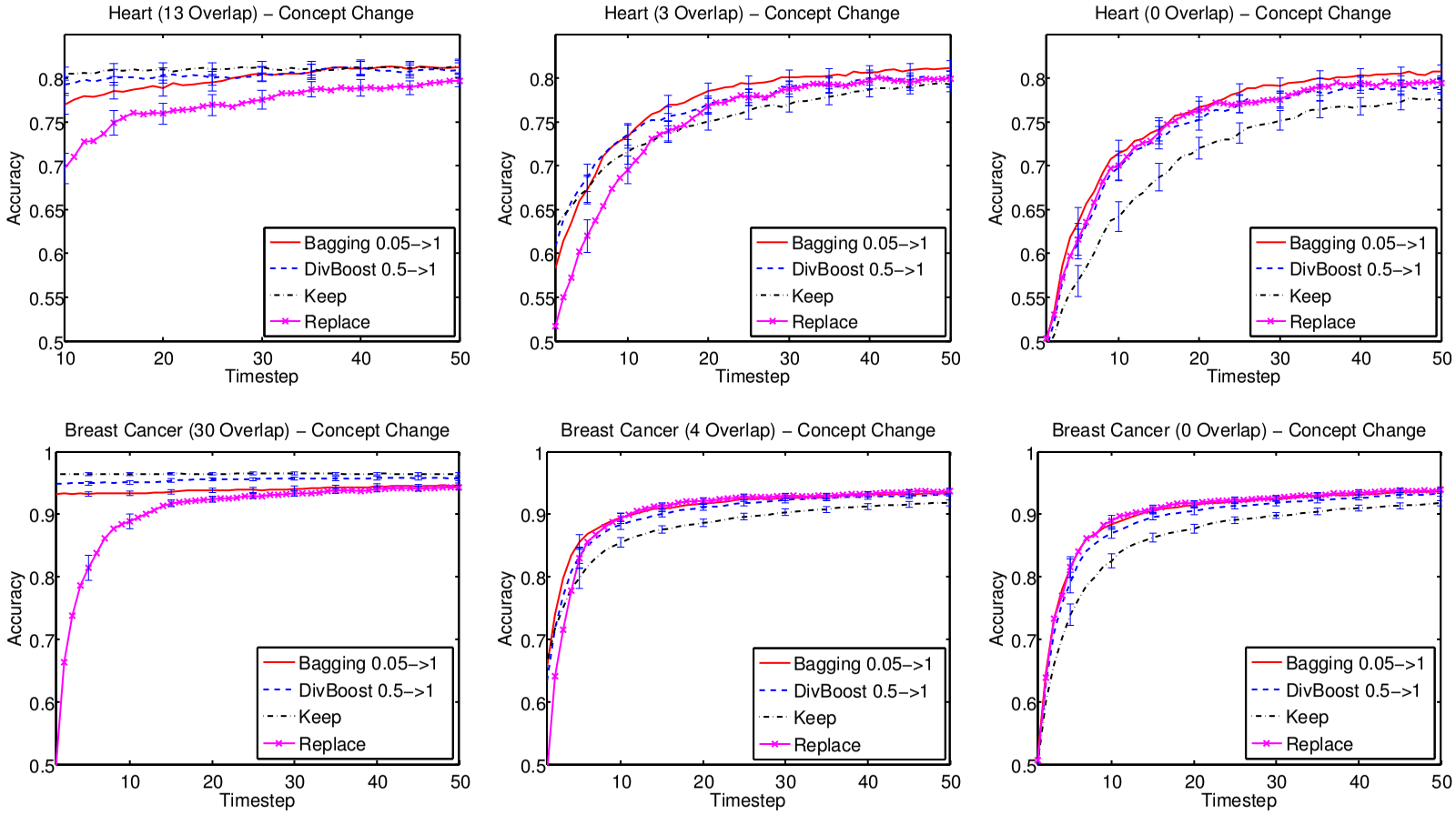

Figure 5.4: Performance on UCI datasets when adapting to concept changes of varying severity. The x axis indicates the number of training examples seen *after* the concept change. The largest overlaps indicate no concept change, while overlap of 0 indicates that the new concept is unrelated to the old one. Continued in Figure 5.5.

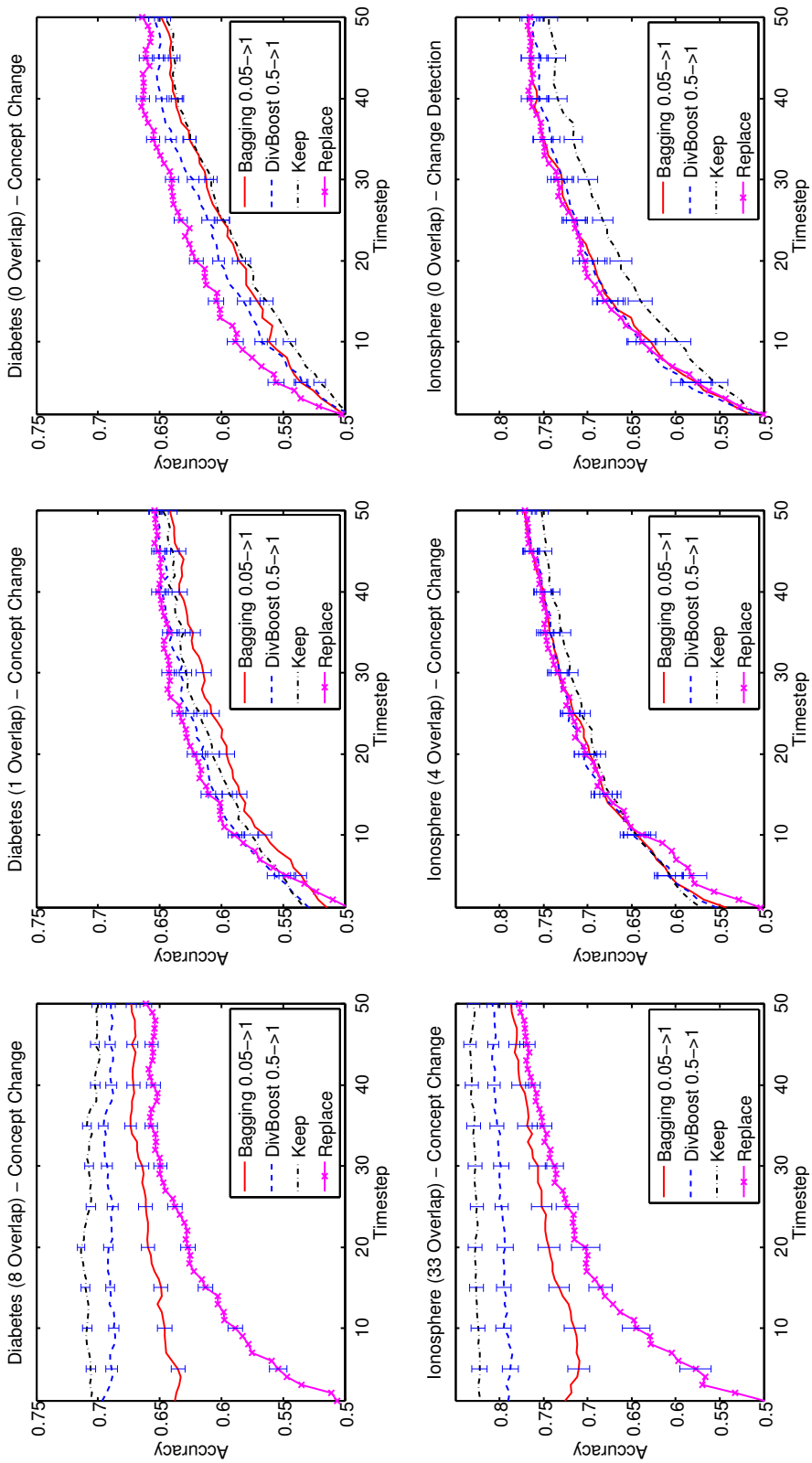

Figure 5.5: Continuation of Figure 5.4, for Diabetes and Ionosphere UCI datasets.

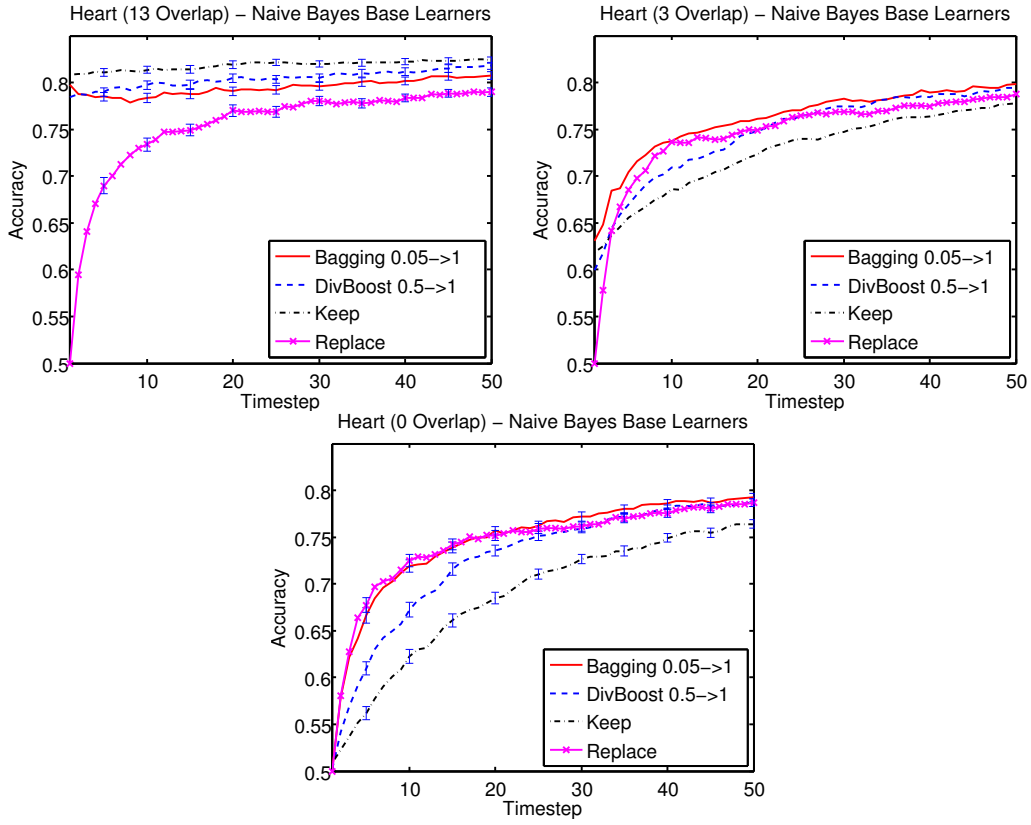

Figure 5.6: Performance on Heart Disease dataset with Naïve Bayes learners. The experiments are identical to those in Figure 5.4, except that the base learners are Naïve Bayes models. Replace and Bagging outperform DivBoost and Keep when the base learners are lossless.

Figure 5.6 shows how the choice of base learner impacts performance. The learning rate of Naïve Bayes depends on the number of training examples previously seen. Therefore, Replace performs well (since the replacement model has a high learning rate), while Keep performs poorly (its learning rate decreases as it trains on additional examples). There is a significant difference between Online Bagging and DivBoost now, due to the methods that they use to encourage diversity — since the λ parameter in Online Bagging corresponds to the mean of a Poisson distribution, this directly translates into *prior to the concept change, each base learner in Online Bagging trains on an average of λN examples*, while the explicit diversity optimisation used by DivBoost means that its base learners train many more examples — generally around N.

Our conclusion from this experiment is that the effect of learning rates in

lossless base learner dominates any impact that diversity has; for such learners, controlling the learning rate will be much more important than diversity in determining performance.

Finally, in Figure 5.7, we show the effect of late change detection (again, the base learners are MLPs). In this case, the Keep strategy has an advantage (since it did not rely on change detection anyway), while Replace performs very poorly. The original effect we observed (DivBoost outperforming Keep on moderate and severe changes) does not appear here.

Overall, these experiments have shown that DivBoost has a small potential for applicability; although it performs quite well regardless of change severity, it does not behave well when the base learner is lossless, and furthermore its advantage over Keep (when base learners are lossy) is mostly neutralised by the presence of late or false change detections.

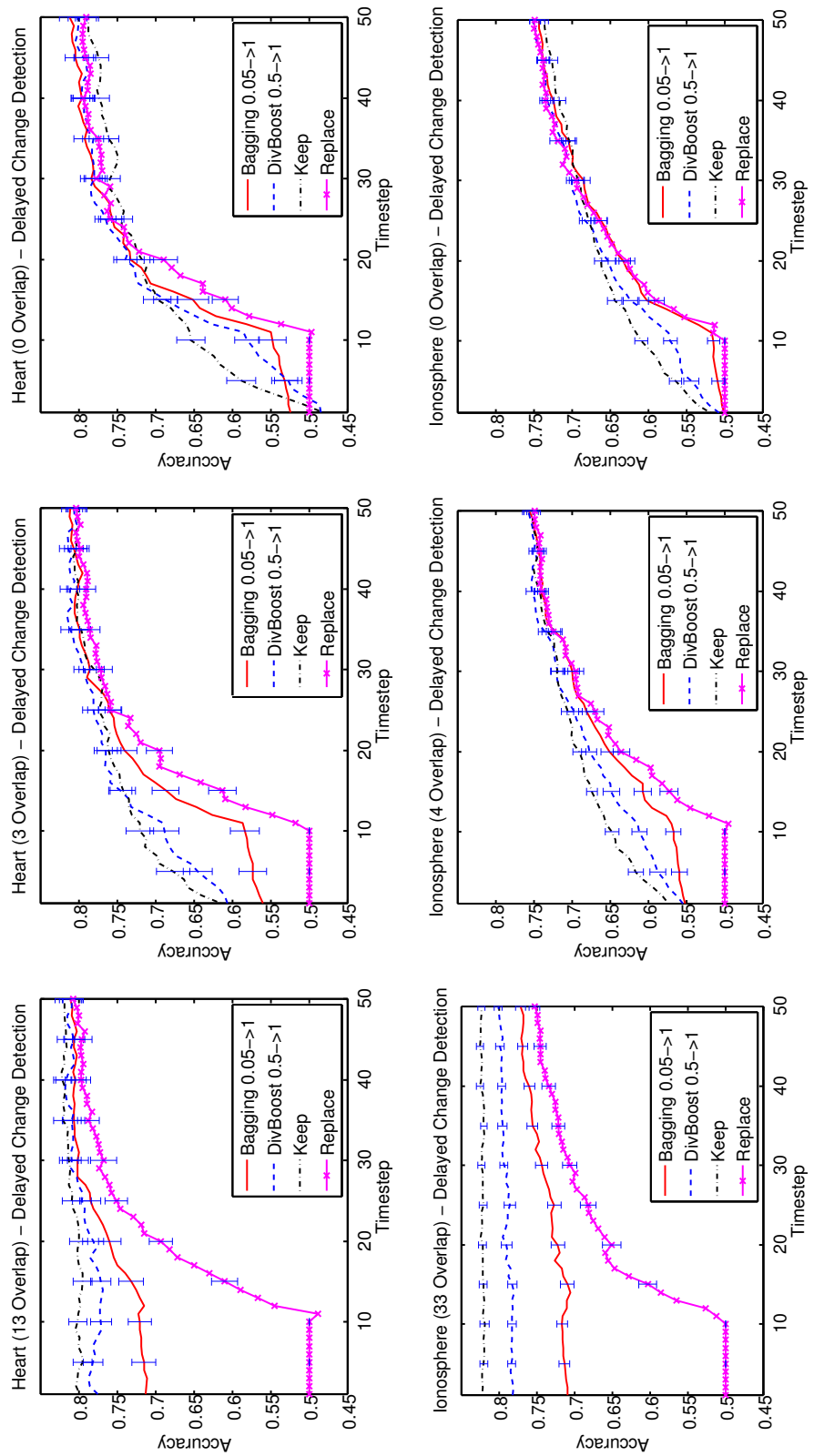

Figure 5.7: Performance on Heart Disease and Ionosphere datasets after a concept change with oracle detection delayed by 10 steps. With delayed detection, Replace and Online Bagging perform much worse.

5.4.4 DDM Change Detection

Addressing our final experimental aim, we replace oracle change detection with the DDM algorithm [28]. This introduces genuine late and false change detections, and also allows us to evaluate performance on real-world datasets where underlying distributions are not known.

Hypothesis: *The exploitation of diversity by DivBoost, although it can produce small improvements in adaptation in restricted scenarios, does not provide significant utility under realistic conditions, where change detection and choice of base learner have more impact on performance.*

Procedure

We first experiment on toy datasets, as we have done previously, but using DDM to provide concept change signals — this affects the behaviour of the Adaptive DivBoost and Replace algorithms; in both cases, a DDM model is trained on their predictions. Wherever such change detection is used, we provide cumulative plots showing the distribution of change detections — these show (for all t) the *proportion* of change detections that occurred *on or before* t; in the legend we specify what the *total* number of detections was. When multiple repetitions were performed, we aggregate change detections so that the cumulative plots display averages over all repetitions.

We use a hold out test set to evaluate performance after every step — we use 50 examples from each stationary concept, and take averages over 200 independent runs. Unlike the previous section, this time we show performance over the whole learning process, since false detections can affect learning even on stationary data.

Finally, we show experiments on two real-world datasets — Australian Electricity and Luxembourg Internet Usage. In both cases we display the prequential test-then-train accuracy. Drawing robust conclusions from such results is difficult, but they do give some indication as to the practical utility of a DivBoost oriented approach to non-stationary learning.

Results

Figure 5.8 demonstrates the impact of using real change detection to guide the behaviour of Adaptive DivBoost; as foreshadowed in the results from Figure 5.7, we see that Adaptive DivBoost no longer exhibits any advantage over Keep.

Similarly, Figure 5.9 shows that no benefit is realised for DivBoost on any of the toy problems we examined earlier.

Finally, we perform experiments on Electricity and Luxembourg. These results broadly follow the pattern from Figure 5.8, suggesting that common real-world drift is not amenable to Adaptive DivBoost; presumably, the drifts in these datasets are gradual enough that MLP base learners are able to adapt sufficiently. Performing the same experiment with Naïve Bayes base learners gives the result that might be expected, with DWM and Replace being more effective as they implement policies of base learner replacement.

With Luxembourg, we found that changes were never detected; this is likely due to relatively slow drift and the fact that (with MLPs) the task took a long time to learn (i.e. the benefit of training on more data outweighed the effect of concept change). Since lack of detection caused all the same behaviour in Keep, Replace and Adaptive DivBoost, we inserted false change detection to better understand what its impact would be in this scenario; every 500 steps, we informed all learners of a change detection and reset the parameters in DDM.

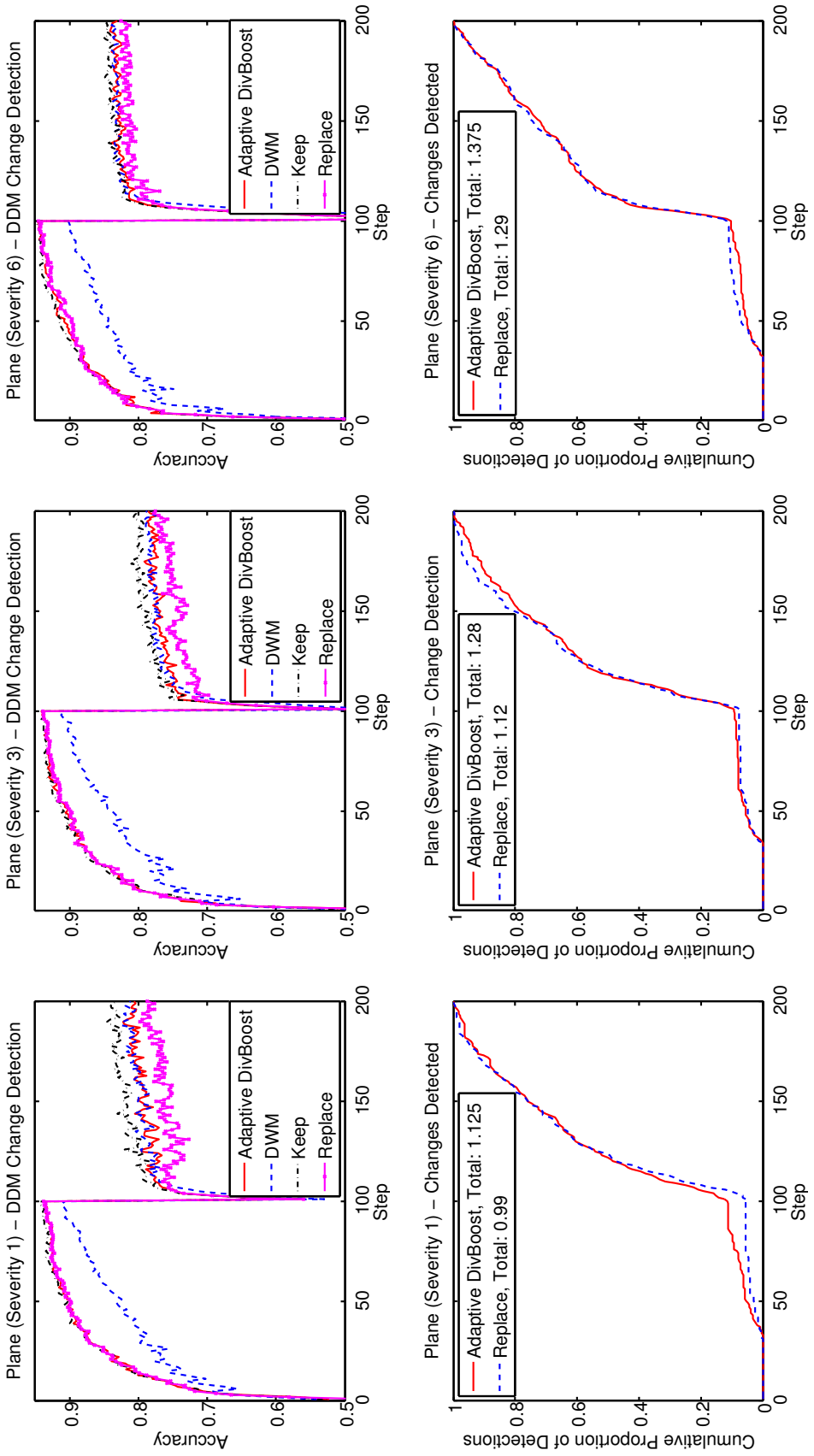

Figure 5.8: Various adaptive algorithms on low and high severity concept changes. Keep performs consistently well; Adaptive DivBoost is generally better than Replace, but never out-performs Keep.

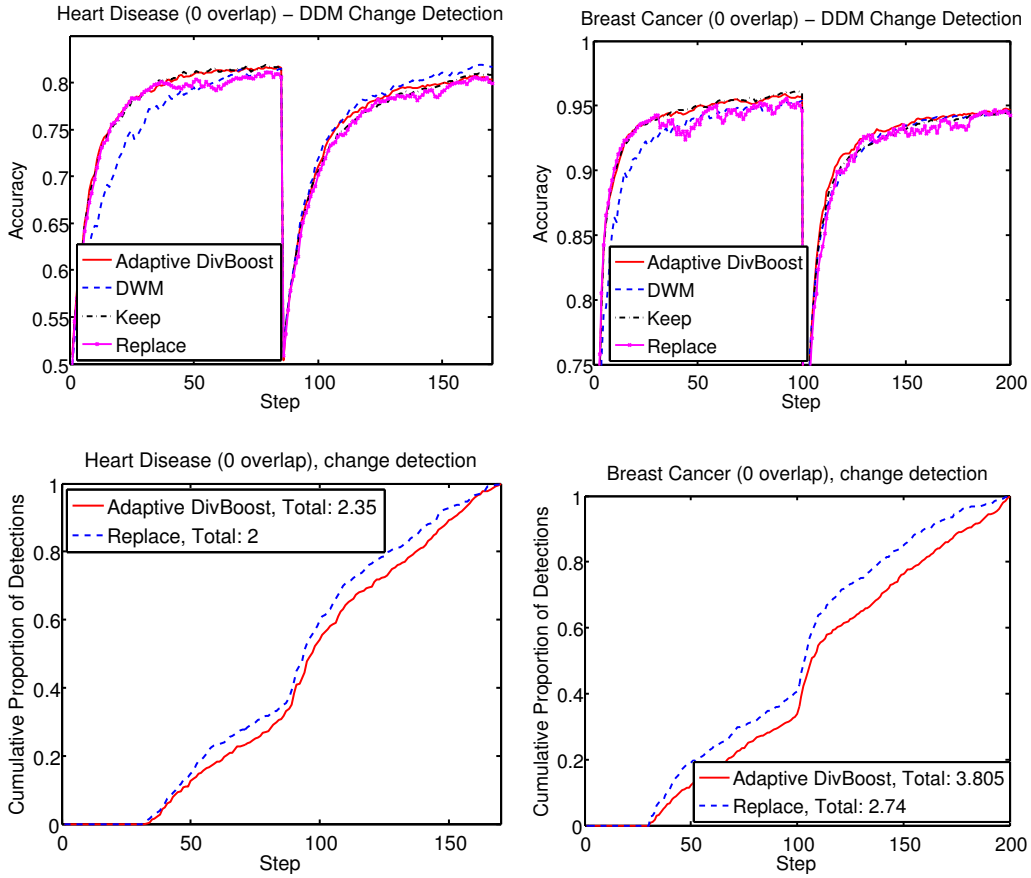

Figure 5.9: Various adaptive algorithms on high severity concept changes. Lower severity datasets were omitted since DDM did not detect changes on them. These results show no discernible advantage to DivBoost over Keep. Continued in Figure 5.10.

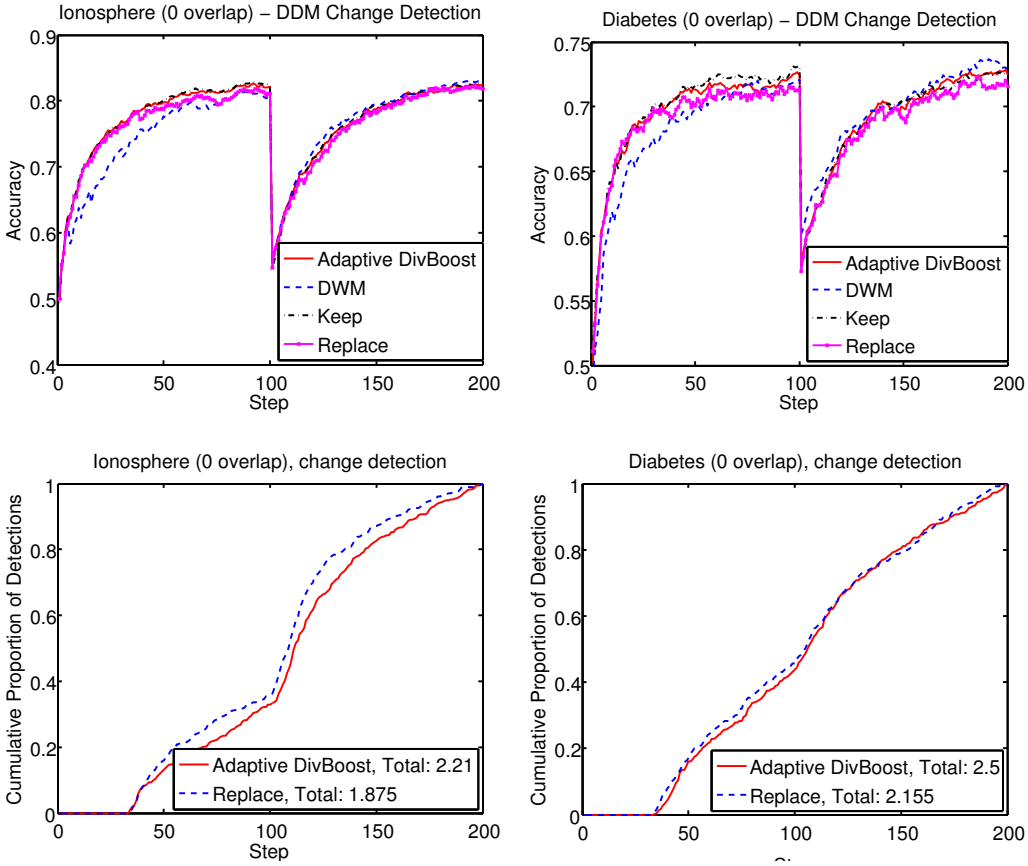

Figure 5.10: Continuation of Figure 5.9.

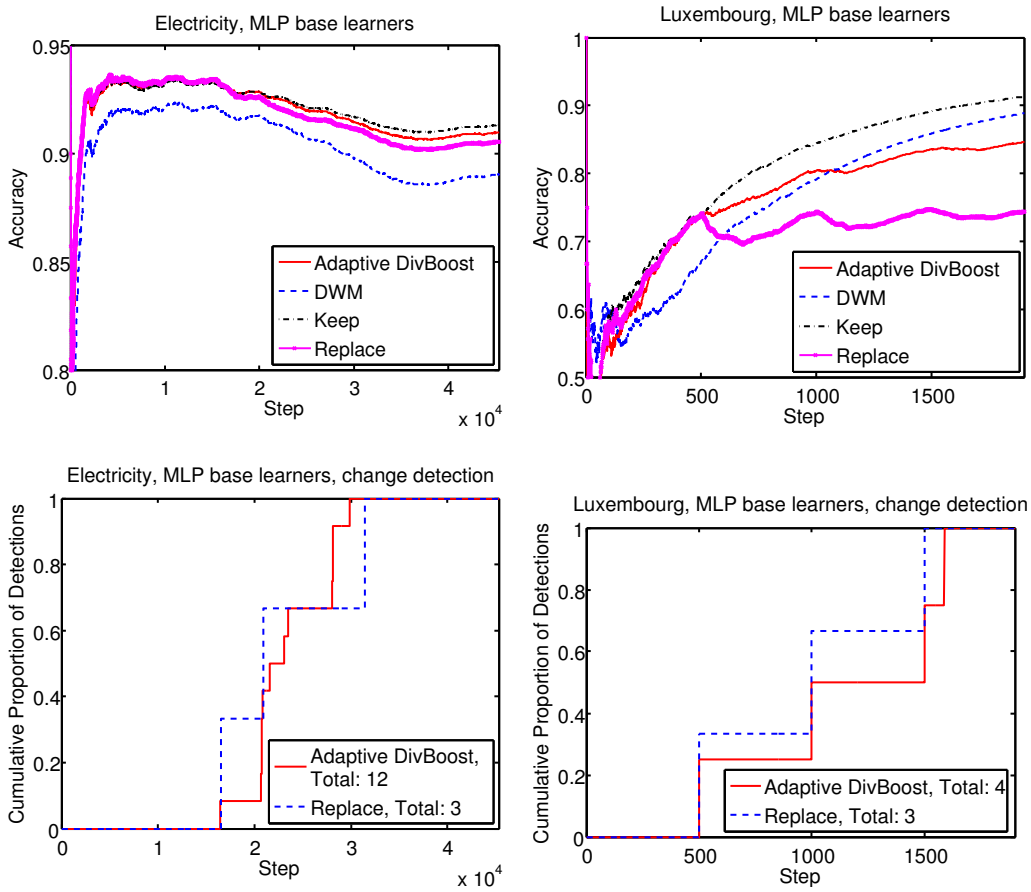

Figure 5.11: Various adaptive algorithms on real non-stationary problems. With MLP base learners, changes are rarely detected; this suggests that MLPs alone are able to adapt sufficiently fast for these problems. Since we are never able to detect drift in the Luxembourg dataset, we initiate false change detections every 500 steps — Replace and Adaptive DivBoost do worse because of these false detections.

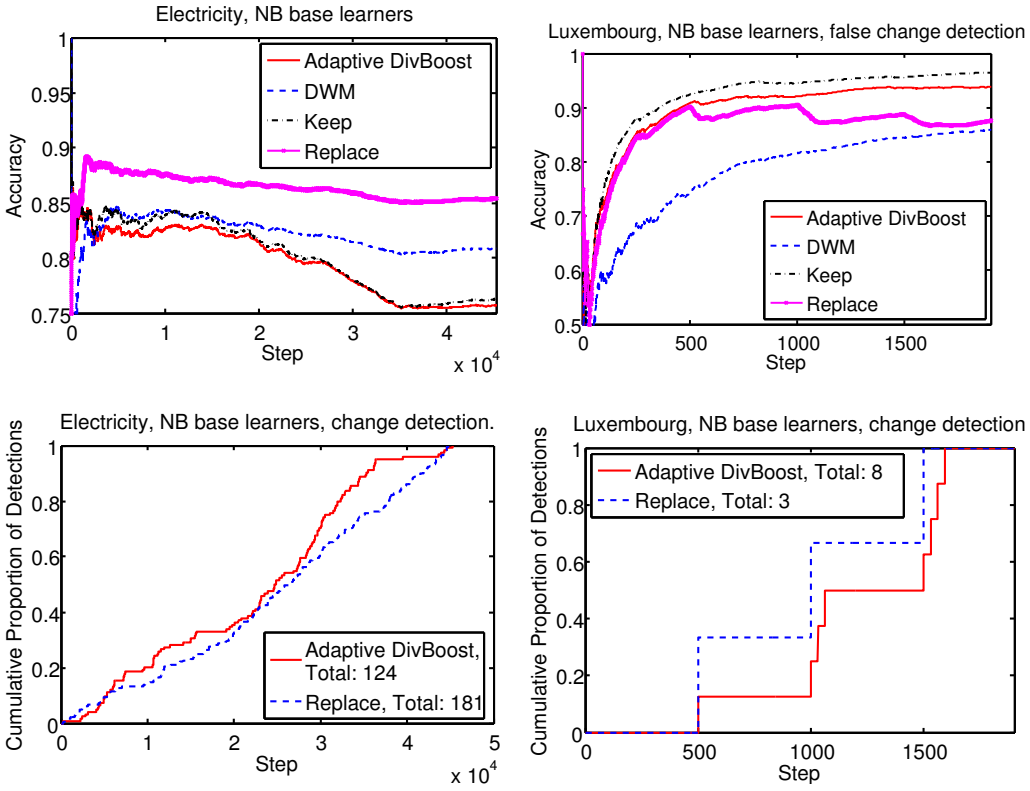

Figure 5.12: The same experiments as in Figure 5.11, but with Naïve Bayes base learners. Note that there are many changes detected, and Replace is most successful here; DWM also outperforms strategies that do not replace base learners. As before with Luxembourg, changes were not detected, so we inserted false change detections ever 500 steps — here, Adaptive DivBoost does not suffer much from the false detections.

5.5 Conclusions

In this chapter we have presented an algorithm that manages diversity so as to facilitate fast adaptation to new concepts in non-stationary learning. We motivated our investigation from a theoretical perspective using our conclusions from Chapter 3. We then discussed how gradient descent boosting could be applied to the task of managing diversity; by modifying the AnyBoost algorithm and taking inspiration from Online Adaboost, we were able to create DivBoost, an algorithm that achieved an amount of diversity that depended on the parameter μ, while retaining good accuracy and being able to learn incrementally.

We used DDM, a change detection algorithm, in combination with a simple non-stationary learning framework, to apply DivBoost to artificial and real non-stationary learning problems.

Our experimental evaluations gave mixed results; many of our conjectures were confirmed — we showed that DivBoost worked correctly, and that it could be used to improve adaptation in certain situations. However, when we tried to use DivBoost with real change detection we found that its utility was greatly diminished; we conclude that:

1. The adaptation benefit of high diversity was observed on toy datasets, when moderate or severe changes occurred — in these cases, the diverse ensemble adapted faster than a non-diverse one (i.e. Keep).

2. The effects of latency and false alarms from change detection were more significant factors than diversity in determining adaptation. For the lossy base learners we studied (MLPs), keeping a fixed ensemble and ignoring change detection became more favourable when false or late changes were detected.

3. Controlling diversity in ensembles of lossless base learners (such as Naïve Bayes) also affected the amount of training data seen by the base learners, which itself had a large effect on the adaptation of the ensemble.

4. The effect of varying amounts of training data in ensembles of lossless base learners highlighted the fact that approaches that exploit these effects (like windowing algorithms, or DWM, where base learners are replaced), could achieve substantially better performance than algorithms that just manipulated diversity.

Chapter 6

Summary and Conclusions

6.1 Summary

This thesis examined two properties of ensemble models — diversity and voting margins. In diversity literature, it has generally been considered that high diversity is beneficial; similarly, the theory of voting margins suggests that large voting margins are important in ensembles. Our work shows a relationship between these two quantities — surprisingly, *high diversity* corresponds to *small margins* — so it would be contradictory for ensembles to benefit from both high diversity and large margins. The first part of the thesis established this connection (Section 3.2) and investigated its implications (Sections 3.3 to 3.5).

In the next part of the thesis, we explored the case of non-i.i.d. learning problems, finding evidence that diversity had some special significance in this domain (Sections 5.2 and 5.4.3). To achieve this, we derived an algorithm (DivBoost) that *explicitly managed* diversity (Section 5.3), and applied it to toy problems with known concept changes. Finally, we examined how the situation changed in more realistic scenarios, finding that the small benefits measured on carefully constructed toy problems were nullified by the effects of false and late change detections.

6.2 Conclusions

In Chapter 3, we made several main contributions, showing that:

1. *Most existing measures of ensemble diversity are also measures of the voting*

margin distribution (Section 3.2): In general, small absolute margins were shown to be associated with 'high diversity'. We showed empirical and theoretic validation of our interpretation in Sections 3.2.5 and 3.3 respectively.

2. *High diversity can be associated with low generalisation accuracy under certain conditions* (Section 3.4.1): Although diversity has relevance in achieving good *training* performance, the idea that diversity should positively influence *generalisation* accuracy contradicts its interpretation within the voting margin framework. Experimentally we showed that high diversity *could* be detrimental to generalisation accuracy with certain training algorithms and datasets, although diverse Bagging ensembles seem to achieve high generalisation accuracy.

3. *The exponential loss function used by Adaboost is approximated by double fault diversity* (Section 3.4.2): this shows a link between Adaboost and existing diversity measures; furthermore, we empirically investigated the approximation by deriving loss functions that *interpolated* between double fault diversity and exponential loss.

This novel view of ensemble diversity improves our ability to use diversity in ensemble learning scenarios; we have demonstrated this utility already, both in making predictions about the effect of diversity (Item 2 in the above list) and considering diversity in the derivation of algorithms (Chapter 5, below). We anticipate further utility in similar future endeavours.

Chapter 5 exploits the diversity-margin correspondence to examine the role of diversity in non-stationary learning. Our contributions were:

1. *Deriving a novel algorithm, DivBoost, that manages diversity according to a parameter and can learn incrementally* (Section 5.3): we showed empirically that DivBoost was able to manage diversity in a desirable fashion.

2. *Identifying specific scenarios where high diversity contributes to the adaptation of the ensemble* (Section 5.4): experimental results indicated that using a diverse ensemble after concept change was usually better than using a non-diverse ensemble (although for the largest changes, the best strategy was to replace the model). This indicated a small potential for application in situations where moderately severe concept changes occur.

3. *Showing that other factors in non-stationary learning dominate the impact of diversity in realistic scenarios* (Section 5.4): prompt change detection was shown to be vital for exploiting diversity in the scenarios where it could be useful. In many artificial situations, and all real datasets that we tested, we failed to observe any advantage of switching to a diverse ensemble (Adaptive DivBoost) over retaining a normal ensemble (Keep) once we used a real change detection algorithm — even though such effects had been observed with oracle change detection.

The DivBoost algorithm is valuable outside of the domain of non-stationary learning: it provides a way of explicitly controlling diversity, and can be applied in both batch and incremental scenarios. Our investigation of diversity in non-stationary learning has not immediately suggested a competitive state-of-the-art algorithm based on diversity, but it has identified areas where diversity *could* be of value, and also shown that performance seems to depend far more on correctly detecting concept change and influencing the adaptation of base learners.

6.3 Limitations

In the previous section, we enumerated the main contributions of the thesis and their implications; however, it is also important to clearly state the limitations of our work — in many cases we will discuss these again in Section 6.4 as potential further work.

All of our contributions have relied to some extent on the restriction of supervised learning tasks to *classification* with *two classes*. Although we provide some insight into how more general conclusions might be drawn (See Appendix C), the two fundamental properties that we have discussed throughout the thesis — diversity and voting margins — both benefit from a degree symmetry exhibited by two-class problems that does not apply in the multi-class case. As such, there is no trivial extension of our work to multi-class problems.

We have also described some potential semantic issues regarding our conclusions about diversity and voting margins (see Section 3.5); in a sense, due to the imprecise definition of 'diversity', it is not certain that the word should be applied to our contributions in Part II of the thesis. This would not especially harm our conclusions though; it would simply imply that we investigated 'manipulation of the margin distribution' in non-stationary learning, rather than 'diversity'.

Our experiments in Part II are somewhat specific to MLP base learners; in some of our experiments we illustrated why manipulating diversity in lossless learners could cause unexpected results. Of course we should expect some degree of learner and data dependence in our results, and so it would be wrong to draw too general a conclusion.

Another aspect of our experiments was the nature of concept drift that we investigated: with the exception of the real world datasets (Electricity and Luxembourg), all the problems we looked at exhibited *abrupt* concept change. Furthermore, our motivations in Section 5.2 only considered abrupt changes. Therefore, we have not addressed the issue of 'ensemble diversity in concept drift problems'; however, we can also present no argument as to why we should *expect* diversity to have special significance in such problems.

6.4 Further Work

In our work on diversity, we highlighted a novel connection between ensemble diversity and voting margins. We propose three extensions to this work:

1. *Elaborating on the relationship between diversity and margins*; for example, both diversity and margins are most easily defined with respect to two-class problems, and this thesis only considered such interpretations. However, it seems that the multi-class definition of the voting margin and diversity measures based on the dichotomy between correct and incorrect predictions could be investigated to establish whether the link between diversity and margins generalises to these multi-class definitions. In Appendix C, we provide some more details on possibilities for this work.

2. *Exchange of insight between diversity and margins*; in Section 3.4.1, we took an insight from the literature on voting margins (large margins are indicative of better generalisation accuracy) and translated it into an insight about diversity (high diversity is indicative of worse generalisation accuracy, under certain circumstances). Given the originality of the margin-diversity connection, it is conceivable that there are similar exchanges of insight between these two fields. For example, in noisy environments it is understood that large margins are not necessarily beneficial (this is shown empirically to some extent [20], and exploited by the DOOM II algorithm of Mason

et al. [62]) — it is possible that this have some relevance for the role of *diversity* in such scenarios.

3. *Exploiting margin optimisation techniques*; In Chapter 5, we used the AnyBoost framework to manage ensemble diversity. This was possible because of the correspondence between diversity and margins. Therefore, in any scenario where there is a hypothesis that diversity is beneficial or detrimental, we can use this algorithm to evaluate such a hypothesis.

Our contributions related to diversity in non-stationary learning can also be extended in a number of ways:

1. *Use of diversity in a holistic approach to non-stationary learning*; we found that managing ensemble diversity could be beneficial in some scenarios — but our techniques for managing diversity precluded the exploitation of other considerations such as windowing. Future work could involve diversity management as a small component of a larger system that also includes other state-of-the-art approaches.

2. *Alternative loss function in DivBoost*; we use the quadratic loss function in DivBoost because of its non-monotonicity, parameterised minimum, and analytic convenience. However, quadratic loss does have some undesirable properties — the function will strongly penalise very large or small margins, and must be symmetric around μ. One improvement might be a negative Gaussian loss function, which could work well in noisy scenarios, in the same way that sigmoidal loss functions can work well as alternatives to exponential loss [62].

3. *Further study of non-stationary learning strategies not involving diversity*; although there might be some value in including diversity in a non-stationary learning algorithm, our results show that other factors tend to be far more important. While there has been significant attention applied to change detection and adaptive window sizing techniques, further attention to these may be more fruitful than attempting to exploit diversity, which at best seems to give only a modest improvement in performance in a restricted set of situations.

Appendix A

Experimental Preliminaries

In this section, we describe non-specific aspects of our experimental environment.

A.1 Datasets

We present information on the various datasets used in this thesis in Table A.1. For experiments of stationary data, we use UCI and toy datasets. For the Iris dataset, we converted the original multiclass problem to 2-class by grouping two classes (Iris Setosa and Iris Versicolour) as one class, and the third class (Iris Virginica) as the other (this class grouping is *not* linearly separable).

For non-stationary problems, we use some non-stationary toy data, some real data, and some modified UCI datasets. With the UCI datasets, we generate artificial drifts using the technique described in [82] — we pad data with additional zero mean unit variance noise features, and then swap the indices of the features, giving two distinct concepts with a parameterised number of common features. By varying this parameter — the number of overlapping features — we can experiment with different amounts of severity. When overlap is 0, the two concepts are independent, and when overlap is the same as the number of features, the two concepts are identical.

The Plane dataset is based on the toy dataset used by Minku [65], which is a generalisation of SEA [83]. Minku chooses a parameter a_0, and changes it to create a new concept. For our experiments, we use 6 different transitions for a_0 (denoted as Severity 1 to Severity 6): $\{-1.5 \rightarrow -3.5, -1.4 \rightarrow -3.6, -1.3 \rightarrow -3.7, -1.2 \rightarrow -3.8, -1.1 \rightarrow -3.9, -1 \rightarrow -4\}$. Unlike SEA, the class priors are 0.5 in all concepts.

Dataset Name	Type	Examples	Features	Class Balance
Breast Cancer	UCI	569	30	0.37
Heart Disease	UCI	270	13	0.44
Blood Transfusion	UCI	748	4	0.24
Ionosphere	UCI	351	33	0.64
Diabetes	UCI	768	8	0.35
Iris (2-class)	UCI	150	4	0.33
Plane	Toy	N/A	3	0.5
Checkerboard	Toy	N/A	2	0.5
Gaussians	Toy	N/A	2	0.5
STAGGER [79]	Toy	N/A	3	0.5
Australian Electricity	Real	45312	7	0.42
Luxembourg [90] [39]	Real	1901	31	0.49

Table A.1: Datasets used in experiments.

The Checkerboard dataset is based on a dataset described by Elwell and Polikar [23], in which a checkerboard can be rotated to create concept drift. Our checkerboard dataset is a stationary version of this, with uniformly distributed features in $[-1, 1]^2$. Tiles were squares with edges of length 0.75, tiled with vertices at the origin, and rotated about the origin by 1 radian $\approx 57°$.

Our Gaussian toy dataset is a two-class problem, with classes drawn from Gaussians with means $(1, 1)$ and $(3, 3)$ respectively and unit variance.

A.2 Evaluation

When evaluating hypotheses, we often measure training or generalisation accuracy on data. For UCI datasets, we estimate generalisation error on hold-out data. In toy datasets, we simply use the data generating algorithm to supply additional i.i.d. data for evaluation. To obtain robust estimates, we perform multiple repetitions of many experiments. For these, we choose random train-test partitions of the data at each repetition.

For displaying results, we indicate significance with 95% confidence intervals on the accuracy. In test-then-train evaluations, we display the prequential accuracy:

$$\text{acc}_T = \frac{1}{T} \sum_{t=1}^{T} \delta[H(x_t) = y_t]. \tag{A.1}$$

Note that this can give misleading results, as performance is dampened by the

$\frac{1}{T}$ factor for large T. This can make it harder to answer questions like "how does the performance change when the underlying concept shifts?"; usually, we use artificial data to evaluate such questions. Furthermore, drawing significant conclusions about performance on real-world data in a test-then-train scenario is difficult: the sequential nature of the data makes it impossible to perform repetitions, so the only statistically robust statements we can make are regarding the average performance over a large number of steps.

A.3 Learners and Parameters

Here, we will give a review of learners and their parameters. Since parameters may vary depending on experiment and dataset, this section simply describes what they are; we supply actual values later in the thesis when actually performing experiments.

A.3.1 Adaboost

Algorithm: Defined by Freund and Schapire [76]. Online version defined by Oza [68] *Parameters:* Ensemble size, base learner algorithm.

Adaboost is a popular supervised learning ensemble algorithm.

A.3.2 AnyBoost

Algorithm: Defined by Mason et al. [62]. *Parameters:* Ensemble size, base learner algorithm, loss function.

AnyBoost generalises Adaboost to cater to a wide range of convex, monotonic loss functions.

A.3.3 Bagging

Algorithm: Defined by Breiman [7]. Online version defined by Oza [68]. *Parameters:* Ensemble size, base learner algorithm.

Bagging is a simple ensemble algorithm where base learners are generated by sampling the training data.

A.3.4 Online Diverse Bagging

Algorithm: Defined by Minku [66]. *Parameters:* Ensemble size, base learner algorithm, Poisson parameter.

A version of Online Bagging where data is sampled less frequently (according to the Poisson parameter) to promote higher diversity.

A.3.5 Dynamic Weighted Majority

Algorithm: Defined by Maloof and Kolter [45] *Parameters:* Period, threshold, discount factor, base learner algorithm

An algorithm for non-stationary learning. With a frequency determined by the period parameter, base learners are down-weighted by the discount factor if they make a mistake. Learners with weight below the threshold are removed.

A.3.6 Incremental Naïve Bayes

Algorithm: Defined in Section 4.2.1. *Parameters:* None.

Naïve Bayes is a well-known algorithm. We implement lossless incremental update rules. Unit pseudo-counts are used to avoid assigning probabilities of 0.

A.3.7 Classification and Regression Trees (CART)

Algorithm: Defined by Breiman et al. [11]. *Parameters:* Minimum examples for split, splitting criterion.

We use the MATLAB implementation of classification and regression trees, with the Gini Diversity Index as a splitting criterion. We set the minimum number of examples required to split a node to $\lfloor \log N \rfloor$.

A.3.8 Decision Stump

Algorithm: Exhaustive search over splitting points. *Parameters:* Splitting criterion.

Our decision stump algorithm searches for a feature and split point that minimises empirical error rate.

A.3.9 Multi-layer Perceptrons

Algorithm: Multi-Layer Perceptrons with back propagation. *Parameters:* Number of hidden units, learning rate, activation function, epochs.

We use the Netlab MLP implementation. To precisely specify the options vector supplied to the mlp function: x precision = 0.001, objective function precision = 0.001, line minimisation via the learning rate. For online learning, only 1 epoch is ever performed, so the momentum parameter is not used. The transfer functions are sigmoidal, and the activation function is logistic (target variables are projected into $\{0, 1\}$, and a threshold of 0.5 is used to indicate the prediction).

In the experiments with DDD (Chapter 5), we use the same implementation and default parameter settings as Minku [64].

Incremental Tree Inducer (ITI)

Algorithm: Defined by Utgoff et al. [87]. *Parameters:* None.

ITI is an incremental algorithm for producing decision trees. It produces an approximation to the tree that would be induced in batch mode; significant for our work is the fact that ITI uses example counts to maintain information about previous data; because of this, the influence of the N^{th} example diminishes as N increases.

Appendix B

Proofs of Theorems

B.1 Lemmas

Here we present a few lemmas that are used in several of the subsequent theorems.

Lemma 12. *Multiplying the average of base learner predictions by the ensemble prediction gives the absolute margin.*

$$H(\mathbf{x}) \sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}) = |m(\mathbf{x}, y)| \tag{B.1}$$

Proof. This is the case because $H(\mathbf{x}) = sign(\sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}))$; therefore:

$$H(\mathbf{x}) \sum_{l=1}^{L} \alpha_l h_l(\mathbf{x}) = |\sum_{l=1}^{L} \alpha_l h_l(\mathbf{x})|, \tag{B.2}$$

$$= |y \sum_{l=1}^{L} \alpha_l h_l(\mathbf{x})|, \tag{B.3}$$

$$= |m(\mathbf{x}, y)|. \tag{B.4}$$

The lemma holds regardless of the true label y, since $|m(\mathbf{x}, y)| = |m(\mathbf{x}, -y)|$. It will also apply when learners are unweighted, by substituting $\alpha_l = \frac{1}{L}$. □

Lemma 13. *Identity functions on the equality of two variables in* $\{-1, 1\}$ *can be*

expressed using a product.

$$\delta[a = b] = \frac{1}{2}(1 + ab), \tag{B.5}$$

$$\delta[a \neq b] = \frac{1}{2}(1 - ab). \tag{B.6}$$

Proof. Using a case-by-case analysis, we construct a truth table for the four possible combinations of a and b:

a	b	$\delta[a = b]$	$\frac{1}{2}(1 + ab)$	$\delta[a \neq b]$	$\frac{1}{2}(1 - ab)$
-1	-1	1	1	0	0
-1	1	0	0	1	1
1	-1	0	0	1	1
1	1	1	1	0	0

Since the table enumerates all the possible values for a and b, and the equalities from the lemma are seen to hold, the lemma is valid. □

B.2 Diversity and Margins

B.2.1 Entropy (Kuncheva)

Proof. (Theorem 1)

Entropy (Kuncheva), E, is defined as:

$$D_{\text{ent}} = \frac{L}{N(L - \lceil L/2 \rceil)} \sum_{i=1}^{N} \min\{c_i, 1 - c_i\}. \tag{B.7}$$

First, we assume that L is odd [1]. This allows us to replace the $\lceil L/2 \rceil$ with $\frac{L+1}{2}$, and hence express the coefficient to the summation as $\frac{2L}{N(L-1)}$. Now, by using $c_i = \frac{1}{2}(1 + m_i)$, we can express the term inside the summation as $\min\{\frac{1}{2}(1 + m_i), \frac{1}{2}(1 - m_i)\}$; a case-by-case analysis gives:

$$\min\{\frac{1}{2}(1 + m_i), \frac{1}{2}(1 - m_i)\} = \begin{cases} \frac{1}{2}(1 + m_i) & \text{if } m_i \leq 0 \\ \frac{1}{2}(1 - m_i) & \text{otherwise,} \end{cases} \tag{B.8}$$

[1] For unweighted ensembles (as is assumed for D_{ent}), the assumption of odd L is reasonable, since even-sized ensembles have no additional expressive power, and additionally sometimes cause tied votes (where both classes receive the same number of votes).

which, since we add m_i when it is negative or subtract it when it is positive, means we can instead subtract $|m_i|$ unconditionally. This gives our final expression:

$$D_{\text{ent}} = \frac{2L}{N(L-1)} \sum_{i=1}^{N} \frac{1}{2}(1 - |m_i|), \tag{B.9}$$

$$= \frac{L}{(L-1)}(1 - \overline{|m|}). \qquad \square$$

B.2.2 Ambiguity (Zenobi)

Proof. (Theorem 2)

Ambiguity (Zenobi) is defined as:

$$D_{\text{Zenobi}} = \frac{1}{NL} \sum_{i=1}^{N} \sum_{l=1}^{L} \delta[h_l(\mathbf{x}_i) = H(\mathbf{x}_i)]. \tag{B.10}$$

We can interpret the identity function using Lemma 13, and subsequently apply Lemma 12 to arrive at the margin:

$$D_{\text{Zenobi}} = \frac{1}{NL} \sum_{i=1}^{N} \sum_{l=1}^{L} \frac{1}{2}(1 - H(\mathbf{x}_i)h_l(\mathbf{x}_i)), \tag{B.11}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2}(1 - H(\mathbf{x}_i) \frac{1}{L} \sum_{l=1}^{L} h_l(\mathbf{x}_i)), \tag{B.12}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2}(1 - |m_i|), \tag{B.13}$$

$$= \frac{1}{2}(1 - \overline{|m|}). \qquad \square$$

Diversity (Melville) will follow an analogous process.

B.2.3 Ambiguity (Brown) and Ambiguity (Chen)

Proof. (Theorem 3)

For Chen's definition of ambiguity:

$$D_{\text{Chen}} = \frac{1}{2N}\sum_{i=1}^{N}\sum_{l=1}^{L} y_i(\frac{1}{L}H(\mathbf{x}_i) - \alpha_l h_l(\mathbf{x}_i)), \tag{B.14}$$

$$= \frac{1}{2N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)(1 - H(\mathbf{x}_i)\sum_{l=1}^{L}\alpha_l h_l(\mathbf{x}_i)), \tag{B.15}$$

$$= \frac{1}{2N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)(1 - |m_i|). \tag{B.16}$$

where we use Lemma 12 to arrive at the absolute margin. Similarly, for Brown's ambiguity term:

$$D_{\text{Brown}} = -\frac{1}{2NL}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)\sum_{l=1}^{L}(1 - h_l(\mathbf{x}_i)H(\mathbf{x}_i)), \tag{B.17}$$

$$= -\frac{1}{2N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i)(1 - |m_i|). \qquad \square$$

B.2.4 KW Variance

Proof. (Theorem 4)

Starting with KW variance, we observe that the interpretation of $\widehat{Pr}(y = 1|\mathbf{x}_i)$ as the proportion of base learners predicting $h(\mathbf{x}_i) = 1$ allows us to write $\widehat{Pr}(y = 1|\mathbf{x}_i) = \frac{1}{2}(1 + y_i m_i)$ [2], after which the derivation is straightforward:

$$D_{\text{KW}} = \frac{1}{2N}\sum_{i=1}^{N}(1 - \widehat{Pr}(y = -1|\mathbf{x}_i)^2 - \widehat{Pr}(y = 1|\mathbf{x}_i)^2), \tag{B.18}$$

$$= \frac{1}{2N}\sum_{i=1}^{N} 1 - \frac{1}{4}(1 + y_i m_i)^2 - \frac{1}{4}(1 - y_i m_i)^2, \tag{B.19}$$

$$= \frac{1}{2N}\sum_{i=1}^{N} 1 - \frac{1}{4}(1 + 2y_i m_i + m_i^2) - \frac{1}{4}(1 - 2y_i m_i + m_i^2), \tag{B.20}$$

$$= \frac{1}{4}(1 - \overline{m^2}), \qquad \square$$

[2] Note that $y_i m_i$ has the same sign as $H(\mathbf{x}_i)$.

where the y_i terms disappear when squared since $y_i \in \{-1, 1\}$.

B.2.5 Ambiguity (Tsymbal)

Proof. (Theorem 5)

Ambiguity (Tsymbal) uses the transformation of $\widehat{Pr}$ to margins (as with KW variance), as well as Lemma 13. With y_i and $h_l(\mathbf{x}_i)$ terms, since they are always either -1 or 1, squaring gives 1.

$$
\begin{aligned}
D_{\text{amb}} &= \frac{1}{NL}\sum_{i=1}^{N}\sum_{l=1}^{L}(\delta[h_l(\mathbf{x}_i)=1] - \widehat{Pr}(y=1|\mathbf{x}_i))^2 + \\
&\quad (\delta[h_l(\mathbf{x}_i)=-1] - \widehat{Pr}(y=-1|\mathbf{x}_i))^2, && \text{(B.21)}\\
&= \frac{1}{LN}\sum_{i=1}^{N}\sum_{l=1}^{L}(\frac{1}{2}(1+h_l(\mathbf{x}_i)) - \frac{1}{2}(1+y_i m_i))^2 \\
&\quad + (\frac{1}{2}(1-h_l(\mathbf{x}_i)) - \frac{1}{2}(1-y_i m_i))^2, && \text{(B.22)}\\
&= \frac{1}{LN}\sum_{i=1}^{N}\sum_{l=1}^{L}\frac{1}{2}(1+m_i^2 - 2h_l y_i m_i), && \text{(B.23)}\\
&= \sum_{i=1}^{N}\frac{1}{2}(1+m_i^2 - \frac{2y_i m_i}{L}\sum_{l=1}^{L}h_l(\mathbf{x}_i)), && \text{(B.24)}\\
&= \frac{1}{2}(1-\overline{m^2}). && \square
\end{aligned}
$$

B.2.6 Entropy (Cunningham)

Proof. (Theorem 6)

As with KW Variance, we convert estimated probabilities to margin-based terms and simplify:

$$\begin{aligned} D_H &= -\frac{1}{N}\sum_{i=1}^{N} \widehat{Pr}(y=-1|\mathbf{x}_i)\log \widehat{Pr}(y=-1|\mathbf{x}_i) \\ &\quad + \widehat{Pr}(y=1|\mathbf{x}_i)\log \widehat{Pr}(y=1|\mathbf{x}_i), && \text{(B.25)} \\ &= -\frac{1}{N}\sum_{i=1}^{N} \frac{1}{2}(1+y_i m_i)\log \frac{1}{2}(1+y_i m_i) \\ &\quad + \frac{1}{2}(1-y_i m_i)\log \frac{1}{2}(1-y_i m_i). && \text{(B.26)} \\ & && \text{(B.27)} \end{aligned}$$

This shows that entropy (Cunningham) is a function of the margins, and we could proceed slightly to simplify the result. However, in order to better compare it with other measures, we wish to find a *second order* polynomial expression. For this reason, we will use a Taylor approximation of log:

$$\begin{aligned} \log x &= \sum_{k=1}^{\infty}(-1)^{k+1}\frac{(x-1)^k}{k}, && \text{(B.28)} \\ &\approx (x-1)-\frac{1}{2}(x-1)^2. && \text{(B.29)} \end{aligned}$$

We can see that Equation B.26 has two components — one for $\widehat{Pr}(y=1|\mathbf{x}_i)$ and one for $\widehat{Pr}(y=-1|\mathbf{x}_i)$. Once these probabilities are converted to margin-based terms, the only difference is the sign of y_i. Therefore, we will consider a single component, and our conclusions will apply symmetrically to the other.

$$\begin{aligned} \log \frac{1}{2}(1+y_i m_i) &\approx \frac{1}{2}(1+y_i m_i) - 1 - \frac{1}{2}\Big(\frac{1}{2}(1+y_i m_i)-1\Big)^2, && \text{(B.30)} \\ &= \frac{1}{2}(y_i m_i - 1) - \frac{1}{8}(y_i m_i - 1)^2, && \text{(B.31)} \\ &= -\frac{5}{8} + \frac{3}{4}y_i m_i - \frac{1}{8}m_i^2. && \text{(B.32)} \end{aligned}$$

We now multiply by $\frac{1}{2}(1 + y_i m_i)$ to compute the entire component:

$$\frac{1}{2}(1 + y_i m_i) \log \frac{1}{2}(1 + y_i m_i) \approx \frac{1}{2}(1 + y_i m_i)(-\frac{5}{8} + \frac{3}{4} y_i m_i - \frac{1}{8} m_i^2), \tag{B.33}$$

$$= \frac{1}{2}(-\frac{5}{8} - \frac{1}{8} y_i m_i + \frac{5}{8} m_i^2 + \frac{5}{8} y_i m_i^3). \tag{B.34}$$

Now recall that this term must be added to the corresponding $\widehat{Pr}(y = -1|\mathbf{x}_i)$ term. That term will differ only in that occurrences of y_i will be inverted; so to find the sum we can[3] cancel all of the terms that contain y_i:

$$D_H \approx -\frac{1}{N} \sum_{i=1}^{N} -\frac{5}{8} + \frac{5}{8} m_i^2, \tag{B.35}$$

$$\approx \frac{5}{8}(1 - \overline{m^2}). \qquad \square$$

[3]This is true, but perhaps not obviously so; equivalently, one could repeat the derivation for $\widehat{Pr}(y = -1|\mathbf{x}_i)$; we use the trick with y_i for brevity.

B.2.7 Double Fault

Proof. (Theorem 7)

Although double fault diversity is generally seen as a pairwise measure, we show that by pulling a sum over training data to the front of the expression, the pairwise terms can be 'collapsed' into margins:

$$D_{\text{DF}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N_{j,k}^{00}}{N}, \tag{B.36}$$

$$= \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{1}{N} \sum_{i=1}^{N} \delta[h_j(\mathbf{x}_i) \neq y_i] \delta[h_k(\mathbf{x}_i) \neq y_i], \tag{B.37}$$

$$= \frac{1}{N(L-1)} \sum_{i=1}^{N} \sum_{j=1}^{L} \delta[h_j(\mathbf{x}_i) \neq y_i] \Big(\frac{1}{2}(1 - y_i \frac{1}{L} \sum_{k=1}^{L} h_k(\mathbf{x}_i)) - \frac{1}{L} \Big), \tag{B.38}$$

$$= \frac{L}{N(L-1)} \sum_{i=1}^{N} \Big(\frac{1}{2}(1 - m_i) - \frac{1}{L} \Big) \frac{1}{L} \sum_{j=1}^{L} \frac{1}{2}(1 - y_i h_j(\mathbf{x}_i)), \tag{B.39}$$

$$= \frac{L}{(L-1)} \Big(\frac{1}{2}(1 - \overline{m}) - \frac{1}{L} \Big) \frac{1}{2}(1 - \overline{m}), \tag{B.40}$$

$$= -\frac{L-1}{2(L-1)} \overline{m} + \frac{L-1}{2(L-1)} - \frac{L}{4(L-1)} + \frac{L}{4(L-1)} \overline{m}^2, \tag{B.41}$$

$$= \frac{1}{2}(1 - \overline{m}) - \frac{L}{4(L-1)}(1 - \overline{m^2}). \qquad \square$$

Arriving at the final neat form requires various manipulations. Note the $\frac{1}{L}$ term that arise due to the $j \neq k$ condition in the pairwise definition.

B.2.8 Difficulty

Proof. (Theorem 8)

$$D_{\text{diff}} = \text{Var}_{Pr(\mathbf{x},y)}[c(\mathbf{x}, y)], \tag{B.42}$$

$$= \frac{1}{4}(E[(m(\mathbf{x}, y) + 1)^2] - E[(m(\mathbf{x}, y) + 1)]^2), \tag{B.43}$$

$$= \frac{1}{4}(\overline{m^2} + 2\overline{m} + 1 - m^2 - 2\overline{m} - 1), \tag{B.44}$$

$$= \frac{1}{4}(\overline{m^2} - \overline{m}^2). \qquad \square$$

The expectations are over $Pr(\mathbf{x}, y)$; hence, the conversion to $\overline{m}$ terms is not strictly correct. However, since diversity is typically measured on training data, this derivation shows the margin equivalent of the measured quantity (which is only an approximation to difficulty). The second line (with expectations) shows the exact relationship with the margin distribution.

B.2.9 Interrater Agreement

Proof. (Theorem 9)

Interrater Agreement has a complicated definition, but Kuncheva shows a link to disagreement that we use to simplify our derivation [54]:

$$D_\kappa = 1 - \frac{1}{2\overline{c}(1-\overline{c})} D_{\text{dis}}, \tag{B.45}$$

$$= 1 - \frac{1}{2\frac{1}{2}(1+\overline{m})(1-\frac{1}{2}(1+\overline{m}))} D_{\text{dis}}, \tag{B.46}$$

$$= 1 - \frac{1}{(1+\overline{m})\frac{1}{2}(1-\overline{m})} D_{\text{dis}}, \tag{B.47}$$

$$= 1 - \frac{2}{1-\overline{m}^2} \frac{L}{2(L-1)} (1 - \overline{m^2}), \tag{B.48}$$

$$= 1 - \frac{L}{L-1} \Big(\frac{1-\overline{m^2}}{1-\overline{m}^2} \Big). \qquad \square$$

B.2.10 Generalised Diversity

Proof. (Theorem 10)

We start from the definition given by Tang [84]:

$$D_{\text{GD}} = \frac{L}{L-1} \Big(1 - \frac{1}{N} \sum_{i=1}^{N} \frac{(1-c_i)^2}{(1-\overline{c})} \Big), \tag{B.49}$$

$$= \frac{L}{L-1} \Big(1 - \frac{1}{N} \sum_{i=1}^{N} \frac{(\frac{1}{2}(1-m_i))^2}{\frac{1}{2}(1-\overline{m})} \Big), \tag{B.50}$$

$$= \frac{L}{L-1} \Big(1 - \frac{(1 - 2\overline{m} + m_i^2)}{2(1-\overline{m})} \Big), \tag{B.51}$$

$$= \frac{L}{L-1} \Big(\frac{1-\overline{m^2}}{2(1-\overline{m})} \Big). \qquad \square$$

B.2.11 Coincident Failure Diversity

Proof. (Theorem 11)

First we show an expression for $\sum_{l=0}^{L}(L-l)Pr(c=\frac{L-l}{L})$, based on Tang's generalised diversity proof [84]:

$$\sum_{l=0}^{L}(L-l)Pr(c=\frac{L-l}{L}) = \frac{1}{N}\sum_{l=0}^{L}\sum_{i=1}^{N}(L-l)\delta[Lc_i = L-l], \tag{B.52}$$

$$= \frac{1}{N}\sum_{i=1}^{N} Lc_i, \tag{B.53}$$

$$= L\overline{c}, \tag{B.54}$$

$$= L\frac{1}{2}(1+\overline{m}) \tag{B.55}$$

since we can reverse the order of the two sums, and then the sum over L has exactly one non-zero term. We consider only the main case of the D_{CFD} definition:

$$D_{\text{CFD}} = \frac{\sum_{l=1}^{L}(L-l)Pr(c=\frac{L-l}{L})}{(L-1)(1-Pr(c=1))}, \tag{B.56}$$

$$= \frac{L\frac{1}{2}(1+\overline{m})-1+1-LPr(c=1)}{(L-1)(1-Pr(c=1))}, \tag{B.57}$$

$$= \frac{L}{L-1}\Big(1-\frac{1-\overline{m}}{2(1-Pr(c=1))}\Big), \tag{B.58}$$

$$= \frac{L}{L-1}\Big(1-\frac{1-\overline{m}}{2(1-\frac{1}{N}\sum_{i=1}^{N}\delta[m_i=1])}\Big). \qquad \square$$

Note that in the case of $Pr(c=1)=1$, then $\overline{m}=1$ and the second term becomes $\frac{0}{0}$; this necessitates the special case which defines $Pr(c=1)=1 \implies D_{\text{CFD}}=1$. We omit this for clarity, and since $Pr(c=1)=1$ only occurs when all base learners make perfect predictions — it seems unlikely that such a special case should have any impact on our analysis [4].

B.2.12 Pairwise and Non-Pairwise Interrater Agreement

Pairwise (D_k) and non-pairwise (D_κ) interrater agreement (kappa) were introduced in Section 2.3.2. Kuncheva [51] discusses the behaviour of pairwise kappa

[4] Though in an implementation of D_{CFD}, this special case should be included.

under the assumption of $N^{01} = N^{10}$, although she does not specifically mention that, under this assumption, $D_k = D_\kappa$. We are therefore unsure whether this relationship is considered to be well-known in the literature, or whether our exposition here is the only explicit reference to it. We start from Kuncheva's definition of a lower bound on D_k, which is tight when $N^{01} = N^{10}$:

$$D_k \geq \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} 1 - \frac{N_{j,k}^{01}}{N\overline{c}(1-\overline{c})}, \tag{B.59}$$

$$= 1 - \frac{1}{N\overline{c}(1-\overline{c})} \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} N_{j,k}^{01}. \tag{B.60}$$

We have a known relationship between the disagreement measure and the margins:

$$D_{\text{dis}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N_{j,k}^{01} + N_{j,k}^{10}}{N}, \tag{B.61}$$

$$= \frac{L}{2(L-1)}(1 - \overline{m^2}). \tag{B.62}$$

We can substitute disagreement into Equation B.60 to give:

$$D_k \geq 1 - \frac{N D_{\text{dis}}}{2N\overline{c}(1-\overline{c})}, \tag{B.63}$$

$$= 1 - \frac{L}{2(L-1)}(1 - \overline{m^2}) \frac{1}{2\overline{c}(1-\overline{c})}, \tag{B.64}$$

$$= 1 - \frac{L}{(L-1)} \Big(\frac{1 - \overline{m^2}}{4\overline{c}(1-\overline{c})} \Big), \tag{B.65}$$

$$= 1 - \frac{L}{(L-1)} \Big(\frac{1 - \overline{m^2}}{1 - \overline{m}^2} \Big), \tag{B.66}$$

$$= D_\kappa. \tag{B.67}$$

Therefore, pairwise and non-pairwise kappa are equivalent when $N^{01} = N^{10}$, and otherwise non-pairwise kappa is a lower bound on pairwise kappa.

Appendix C

Multiclass Diversity and Margins

This thesis has dealt only with binary prediction tasks ($\mathcal{Y} = \{-1, 1\}$). We have presented no relationship between diversity and voting margins in the multiclass case ($\mathcal{Y} = \{1 \dots K\}, K \in \mathbb{N}$). In many learning problems, there are more than two possible predictions (for example, a vehicle recognition system could predict car, motorbike, van, lorry, bus etc.). In this section, we explain why the multiclass definition of the voting margin will not be directly related to multiclass diversity measures, and suggest that a probabilistic interpretation of ensemble behaviour might lead to a good multiclass measure of 'diversity'. This content is primarily for the suggestion of future directions.

C.1 Multiclass Diversity

As in the binary case, there are many definitions for different diversity measures. For measures which are based on pairwise quantities, the definitions look essentially the same — for example,

$$D_{\text{dis}} = \frac{1}{L(L-1)} \sum_{j=1}^{L} \sum_{k \neq j}^{L} \frac{N_{j,k}^{01} + N_{j,k}^{10}}{N}, \tag{C.1}$$

since $N_{j,k}^{10}$ is based on a dichotomy between *correct* and *wrong* predictions. In the two-class case, we could interpret $N_{j,k}^{10}$ as "the number of examples $\langle \mathbf{x}, y \rangle$ for which $h_j(\mathbf{x}) = y$ and $h_k(\mathbf{x}) = -y$"; however, now we must subtly loosen this definition to say that h_k can predict anything other than y. We can express this

as a sum over training data:

$$N_{j,k}^{10} = \sum_{i=1}^{N} \delta[h_j(\mathbf{x}_i) = y_i]\delta[h_k(\mathbf{x}_i) \neq y_i]. \tag{C.2}$$

In the two-class case, this measurement also described whether the classifier pair agreed. In multiclass cases, we still know that $N_{j,k}^{11}$ counts *agreement* (there is only one correct class), and $N_{j,k}^{10}, N_{j,k}^{01}$ count disagreement, but $N_{j,k}^{00}$ could describe agreement or disagreement.

If we follow the same kind of manipulations that we used to relate diversity to margins in the two-class case, we arrive at:

$$D_{\text{dis}} = \frac{2L}{(L-1)}\overline{c}(1-\overline{c}), \tag{C.3}$$

($\overline{c}$ is the average classification accuracy of the base learners). In the two-class case, we could connect this to the margin with $\overline{c} = \frac{1}{2}(1+\overline{m})$.

C.2 Multiclass Voting Margins

Approaching from the perspective of voting margins, the multi-class margin is defined as:

$$m(\mathbf{x}, y) = \frac{1}{L}\sum_{l=1}^{L} \delta[h_l(\mathbf{x}) = y] - \arg\max_{y' \neq y} \frac{1}{L}\sum_{l=1}^{L} \delta[h_l(\mathbf{x}) = y']. \tag{C.4}$$

Unlike the correct/wrong dichotomy used for diversity measures, the multiclass margin makes distinctions between votes for the *correct* answer, and votes for the *most popular wrong* answer. This does not seem to have any correspondence to existing diversity measures.

C.3 Future Work

One possibility is that, in the general case, diversity should be related to a *probabilistic* interpretation of base learner predictions; we have the two-class relationship between the margin and an estimated probability:

$$m(\mathbf{x}, y) = \frac{1}{2}(1 + \widehat{Pr}(y|\mathbf{x})). \tag{C.5}$$

For example, in the two-class case, the entropy of the distribution $\widehat{Pr}(y|\mathbf{x})$ quantifies the diversity and would meaningfully generalise to multiclass problems. Unlike the voting margin, or measures based on contingency matrices, it would be completely symmetric with respect to true class labels.

One issue with generalising ensemble-specific quantities is that we ideally want the result to still specifically apply to ensembles; for example, in Section 2.3.2, we described some information theoretic approaches to diversity; these were related to a decomposition that permitted upper and lower bounds on the error rate of the optimal combiner — which would not, typically, be a linear combiner as in voting ensembles. Therefore, to achieve results that *exploit* the specific properties of voting ensembles, there may be some value in considering measures that *only make sense for voting ensembles*; at this stage it is unclear whether a probabilistic perspective would retain this specificity.

Bibliography

[1] K. M. Ali and M. J. Pazzani. Error reduction through learning multiple descriptions. *Machine Learning*, 24:173–202, 1996.

[2] S. Bach and M. Maloof. A Bayesian approach to concept drift. *Advances in Neural Information Processing Systems*, 23:127–135, 2010.

[3] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno. Early drift detection method. *International Workshop on Knowledge Discovery From Data Streams*, 4:77–86, 2006.

[4] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–139, 1999.

[5] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà. New ensemble methods for evolving data streams. *International Conference on Knowledge Discovery and Data Mining*, pages 139–148, 2009.

[6] A. Bifet and R. Kirkby. Data stream mining: a practical approach. Technical report, University of Waikato, 2009.

[7] L. Breiman. Bagging predictors. *Machine Learning*, pages 123–140, 1996.

[8] L. Breiman. Prediction games and arcing algorithms. *Neural Computing*, 11(7):1493–1517, 1999.

[9] L. Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242, 2000.

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

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

[12] D. Briggs, M. Whitehill, and S. Knight. Who wants to be a millionaire. produced by Celador, aired on ITV, 1998.

[13] G. Brown. *Diversity in Neural Network Ensembles*. PhD thesis, University of Birmingham, 2004.

[14] G. Brown. An information theoretic perspective on multiple classifier systems. *Multiple Classifier Systems*, pages 344–353, 2009.

[15] G. Brown and L. I. Kuncheva. "Good" and "bad" diversity in majority vote ensembles. *Multiple Classifier Systems*, pages 124–133, 2010.

[16] H. Chen. *Diversity and Regularization in Neural Network Ensembles*. PhD thesis, University of Birmingham, 2008.

[17] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[18] P. Cunningham and J. Carney. Diversity versus quality in classification ensembles based on feature selection. *European Conference on Machine Learning*, pages 109–116, 2000.

[19] T. G. Dietterich. Ensemble methods in machine learning. *Multiple Classifier Systems*, pages 1–15, 2000.

[20] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40:139–157, 2000.

[21] P. Domingos and G. Hulten. Mining high-speed data streams. *International Conference on Knowledge Discovery and Data Mining*, 6:71–80, 2000.

[22] A. W. F. Edwards. The measure of association in a 2 x 2 table. *Journal of the Royal Statistical Society*, 126(1):109–114, 1963.

[23] R. Elwell and R. Polikar. Incremental learning of variable rate concept drift. *Multiple Classifier Systems*, pages 142–151, 2009.

[24] R. Elwell and R. Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011.

[25] R. Fano. *Transmission of Information: A Statistical Theory of Communications*. The MIT Press, Cambridge, MA, 1961.

[26] J. Fleiss. *Statistical Methods for Rates and Proportions*. John Wiley and Sons, 1981.

[27] J. Gama, R. Fernandes, and R. Rocha. Decision trees for mining data streams. *Intelligent Data Analysis*, 10(1):23–45, 2006.

[28] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. *Brazilian Symposium on Artificial Intelligence*, pages 286–295, 2004.

[29] E. Gatnar. A diversity measure for tree-based classifier ensembles. *Data Analysis and Decision Support*, pages 30–38, 2005.

[30] G. Giacinto, F. Roli, and G. Fumera. Design of effective multiple classifier systems by clustering of classifiers. *International Conference on Pattern Recognition*, 15:3–8, 2000.

[31] J. B. Gomes, E. Menasalvas, and P. A. C. Sousa. Tracking recurrent concepts using context. *International conference on Rough Sets and Current Trends in Computing*, 7:168–177, 2010.

[32] M. Grbovic and S. Vucetic. Tracking concept change with incremental boosting by minimization of the evolving exponential loss. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 516–532, 2011.

[33] A. J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. *National Conference of Artificial Intelligence*, 15, 1998.

[34] Friedman J. H., T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[35] L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 993–1001, 1990.

[36] M. E. Hellman and J. Raviv. Probability of error, equivocation and the chernoff bound. *IEEE Transactions on Information Theory*, 16:368–372, 1970.

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

[38] U. Johansson, T. Lofstrom, and L. Niklasson. The importance of diversity in neural network ensembles - an empirical investigation. *International Joint Conference on Neural Networks*, pages 661–666, 2007.

[39] R. Jowell. European social survey 2002/2003; 2004/2005; 2006/2007. Technical report, Centre for Comparative Social Surveys, City University, 2003, 2005, 2007.

[40] M. N. Kapp, R. Sabourin, and P. Maupin. An empirical study on diversity measures and margin theory for ensembles of classifiers. *Journal on Information Fusion*, 10:1 –8, 2007.

[41] M. Kearns. Thoughts on hypothesis boosting. Unpublished manuscript, project for MIT machine learning course, obtained from the author's personal website, 1988.

[42] R. Klinkenberg and T. Joachims. Detecting concept drift with support vector machines. *International Conference on Machine Learning*, 17:487–494, 2000.

[43] R. Kohavi and D. H. Wolpert. Bias plus variance decomposition for zero-one loss functions. *International Conference on Machine Learning*, 13:275–283, 1996.

[44] J. Z. Kolter and M. A. Maloof. Using additive expert ensembles to cope with concept drift. *International Conference on Machine Learning*, 22:449–456, 2005.

[45] J. Z. Kolter and M. A. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8:2755–2790, December 2007.

[46] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23:89–109, 2001.

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

[48] W. Krzanowski and D. Partridge. Software diversity: Practical statistics for its measurement and exploitation. *Information and Software Technology*, 39:39–707, 1996.

[49] L. I. Kuncheva. That elusive diversity in classifier ensembles. *Pattern Recognition and Image Analysis*, 2652:1126–1138, 2003.

[50] L. I. Kuncheva. Classifier ensembles for changing environments. *Multiple Classifier Systems*, pages 1–15, 2004.

[51] L. I. Kuncheva. A bound on kappa-error diagrams for analysis of classifier ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 99, 2011.

[52] L. I. Kuncheva and I. Žliobaitė. On the window size for classification in changing environments. *Intelligent Data Analysis*, 13(6):861–872, 2009.

[53] L. I. Kuncheva and C. J. Whitaker. Ten measures of diversity in classifier ensembles: Limits for two classifiers. *IEEE Workshop on Intelligent Sensor Processing*, pages 1–10, 2001.

[54] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.

[55] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. W. Duin. Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis and Applications*, 6:22–31, 2003.

[56] Y. Law and C. Zaniolo. An adaptive nearest neighbor classification algorithm for data streams. *Principles and Practice of Knowledge Discovery in Databases*, pages 108–120, 2005.

[57] Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neural Networks*, 12:1399–1404, 1999.

[58] R. Maclin and D. Opitz. An empirical evaluation of bagging and boosting. *National Conference of Artificial Intelligence*, 14:546–551, 1997.

[59] O. L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24(1–2):15–23, 1999.

[60] D. D. Margineantu and T. G. Dietterich. Pruning adaptive boosting. *International Conference on Machine Learning*, pages 211–218, 1997.

[61] G. Martínez-muiñoz and A. Suárez. Switching class labels to generate classification ensembles. *Pattern Recognition*, 38:1483–1494, 2005.

[62] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. *Advances in Neural Information Processing Systems*, 12:512–518, 2000.

[63] P. Melville and R. J. Mooney. Creating diversity in ensembles using artificial data. *Journal on Information Fusion*, 6:99–111, 2004.

[64] L. L. Minku. *Online ensemble learning in the presence of concept drift*. PhD thesis, University of Birmingham, 2011.

[65] L. L. Minku, A. P. White, and X. Yao. The impact of diversity on online ensemble learning in the presence of concept drift. *IEEE Transactions on Knowledge and Data Engineering*, 22(5):730–742, 2010.

[66] L. L. Minku and X. Yao. DDD: A new ensemble approach for dealing with concept drift. *IEEE Transactions on Knowledge and Data Engineering*, 24(4):619–633, 2012.

[67] M. D. Muhlbaier, A. Topalis, and R. Polikar. Learn^{++}.nc: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes. *IEEE Transactions on Neural Networks*, 20(1):152–168, 2009.

[68] N. C. Oza. *Online Ensemble Learning*. PhD thesis, The University of California, Berkeley, CA, 2001.

[69] A. Pocock, P. Yiapanis, J. Singer, M. Luján, and G. Brown. Online non-stationary boosting. *Multiple Classifier Systems*, pages 205–214, 2010.

[70] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6(3):21–45, 2006.

[71] R. Polikar, L. Upda, S.S. Upda, and V. Honavar. Learn++: an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 31(4):497–508, 2001.

[72] J. R. Quinlan. Bagging, boosting, and C4.5. *National Conference of Artificial Intelligence*, 13:725–730, 1996.

[73] G. Rätsch, B. Schölkopf, S. Mika, and K.-R. Müller. SVM and Boosting: One class. Technical Report 119, GMD FIRST, 2000.

[74] J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1619–1630, 2006.

[75] L. Saitta. Hypothesis diversity in ensemble classification. *Foundations of Intelligent Systems*, pages 662–670, 2006.

[76] R. Schapire and Y. Freund. Decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, 55(1):119–139, 1997.

[77] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[78] Robert E. Schapire and Yoav Freund. Boosting the margin: a new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26:322–330, 1998.

[79] J. C. Schlimmer and R. H. Granger, Jr. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354, 1986.

[80] M. Scholz and R. Klinkenberg. Boosting classifiers for drifting concepts. *Intelligent Data Analysis*, 11(1):3–28, 2007.

[81] D. B. Skalak. The sources of increased accuracy for two proposed boosting algorithms. *American Association for Artificial Intelligence, Integrating Multiple Learned Models Workshop*, pages 120–125, 1996.

[82] R. Stapenhurst and G. Brown. Theoretical and empirical analysis of diversity in non-stationary learning. *IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments*, pages 25–32, 2011.

[83] W. N. Street and Y. Kim. A streaming ensemble algorithm (SEA) for large-scale classification. *International Conference on Knowledge Discovery and Data Mining*, 7:377–382, 2001.

[84] E. K. Tang, P. N. Suganthan, and X. Yao. An analysis of diversity measures. *Machine Learning*, 65(1):247–271, 2006.

[85] A. Tsymbal, M. Pechenizkiy, and P. Cunningham. Diversity in random subspacing ensembles. *Data Warehousing and Knowledge Discovery*, pages 309–319, 2004.

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

[87] P. E. Utgoff, N. C. Berkman, J. A. Clouse, and D. Fisher. Decision tree induction based on efficient tree restructuring. *Machine Learning*, pages 5–44, 1996.

[88] V. N. Vapnik. *Estimation of dependences based on empirical data*. Springer-Verlag New York, Inc., 1982.

[89] I. Žliobaitė and L. I. Kuncheva. Determining the training window for small sample size classification with concept drift. *IEEE International Conference on Data Mining Workshops*, pages 447–452, 2009.

[90] Indrė Žliobaitė. Combining time and space similarity for small size learning under concept drift. *International Symposium on Methodologies for Intelligent Systems*, 5722:412–421, 2009.

[91] Indrė Žliobaitė. Learning under concept drift: an overview. *Computing Research Repository*, abs/1010.4784, 2010.

[92] L. Wang, M. Sugiyama, Z. Jing, C. Yang, Z. Zhou, and J. Feng. A refined margin analysis for boosting algorithms via equilibrium margin. *Journal of Machine Learning Research*, pages 1835–1863, 2011.

[93] B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[94] G. U. Yule. On the association of attributes in statistics. *Philosophical Transactions of the Royal Society of London*, 194:257–319, 1900.

[95] G. Zenobi and P. Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. *European Conference on Machine Learning*, 12:576–587, 2001.

[96] Z. Zhou and N. Li. Multi-information ensemble diversity. *Multiple Classifier Systems*, pages 134–144, 2010.

[97] Z. Zhou, J. Wu, Y. Jiang, and S. Chen. Genetic algorithm based selective neural network ensemble. *International joint conference on Artificial Intelligence*, 17:797–802, 2001.