Robert Gordon University

Doctoral Thesis

On Pruning and Feature Engineering in Random Forests

Author:
Khaled Fawagreh

Supervisors:
Dr. Mohamed Gaber
Dr. Eyad ELYAN

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

in the

Data Mining Research Group
School of Computing Science and Digital Media

October 2016

Declaration of Authorship

I, Khaled Fawagreh, declare that this thesis titled, 'On Pruning and Feature Engineering in Random Forests' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"Success is never final. Failure is never fatal. Courage is what counts."

Sir Winston Churchill

ROBERT GORDON UNIVERSITY

Abstract

Faculty of Design and Technology
School of Computing Science and Digital Media

Doctor of Philosophy

On Pruning and Feature Engineering in Random Forests

by Khaled Fawagreh

Random Forest (RF) is an ensemble classification technique that was developed by Leo Breiman over a decade ago. Compared with other ensemble techniques, it has proved its accuracy and superiority. Many researchers, however, believe that there is still room for optimizing RF further by enhancing and improving its performance accuracy. This explains why there have been many extensions of RF where each extension employed a variety of techniques and strategies to improve certain aspect(s) of RF. The main focus of this dissertation is to develop new extensions of RF using new optimization techniques that, to the best of our knowledge, have never been used before to optimize RF. These techniques are clustering, the local outlier factor, diversified weighted subspaces, and replicator dynamics. Applying these techniques on RF produced four extensions which we have termed *CLUB-DRF*, *LOFB-DRF*, *DSB-RF*, and *RDB-DR* respectively. Experimental studies on 15 real datasets showed favorable results, demonstrating the potential of the proposed methods. Performance-wise, *CLUB-DRF* is ranked first in terms of accuracy and classification speed making it ideal for real-time applications, and for machines/devices with limited memory and processing power.

Keywords: Random Forests, Ensemble Classification, Clustering, Local Outlier Factor, Replicator Dynamics

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Dr. Mohamad Gaber, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research as well as on my career have been priceless. I would also like to thank my second supervisor Dr. Eyad Elyan for his valuable input and support, and also for his good sense of humor.

A special thanks to my family. Words cannot express how grateful I am for all of the sacrifices that you've made on my behalf. Your prayer for me was what sustained me thus far. I would also like to thank all of my friends who supported me in writing, and incented me to strive towards my goal. At the end I would like to express appreciation to my beloved wife Naseem who spent sleepless nights with me and was always my support in the moments when there was no one to answer my queries.

Contents

List of Figures

List of Tables

Abbreviations

APP	**A**bsolute **P**redictive **P**ower
BIRCH	**B**alanced **I**terative **R**educing and **C**lustering using **H**ierarchies
CART	**C**lassification and **R**egression **T**ree
CCVA	**C**omputer **C**oded **V**erbal **A**utopsy
CLARANS	**CL**ustering **A**lgorithm based on **RAN**domized **S**earch
CLUB-DRF	**CLU**stering-**B**ased **D**iverse **R**andom **F**orest
CURE	**C**lustering **U**sing **RE**presentatives
DBSCAN	**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
DSB-RF	**D**iversified Subspaces-**B**ased **R**andom **F**orest
DFM	**D**ouble **F**ault **M**easure
DM	**D**isagreement **M**easure
DS	**D**ynamic **S**election
DV	**D**ynamic **V**oting
DVS	**D**ynamic **V**oting with **S**election
GD	**G**eneralized **D**iversity
HOG	**H**istogram of **O**riented **G**radients
KW	**K**ohavi **W**olpert
k-NN	**k**-**N**earest **N**eighbors
LOFB-DRF	**L**ocal **O**utlier **F**actor-**B**ased **D**iverse **R**andom **F**orest
LOF	**L**ocal **O**utlier **F**actor
MLP	**M**ulti **L**ayer **P**erception
PCVA	**P**hysician **C**ertified **V**erbal **A**utopsy
RPP	**R**elative **P**redictive **P**ower
RD	**R**eplicator **D**ynamics
RF	**R**andom **F**orest

ROC	**R**eceiver **O**perating **C**haracteristic
SDP	**S**emi **D**efinite **P**rogramming
SVM	**S**upport **V**ector **M**achine
VIM	**V**ariable **I**mportance **M**easure

Part I

Preliminaries

Chapter 1

Introduction

1.1 Background

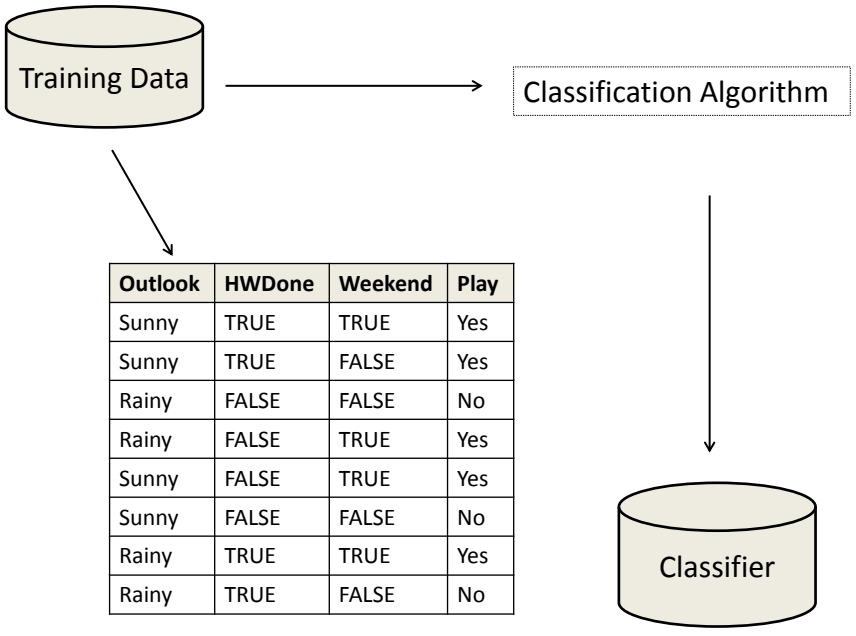
Data mining is an important and interesting subfield in computer science that has received significant attention from the research community particularly over the past decade. In a nutshell, data mining can be defined as the process of extracting knowledge and/or unknown interesting patterns from structured, semi-structured or unstructured data Jiawei and Kamber (2011). Its importance and significance stem from the fact that it specializes in analyzing the data from different perspectives and summarizing it into useful information. Such information can be used to increase revenues, cut costs, or both Fawagreh et al. (2014a). What makes it an interesting field is that it applies methods at the intersection of multiple disciplines including artificial intelligence, machine learning, statistics, and database systems Fayyad et al. (1996).

Following the taxonomy used in Fayyad et al. (1996), data mining involves seven common classes of tasks: anomaly detection, association rule learning, clustering, classification, regression, summarization, and sequential pattern mining. The research work reported in this dissertation deals with classification which builds models from previously classified data to predict future data with unknown classes.

The bulk of data mining methods can be traced back to an area known as machine learning. It initially evolved from the study of pattern recognition and computational learning theory in artificial intelligence, and deals with the study and construction of algorithms that can learn from data and make predictions on data Kohavi and Provost

(1998). Machine learning algorithms are classified into supervised and unsupervised learning. Supervised learning involves establishing a mapping between a set of input variables X and an output variable Y and applying this mapping to predict the outputs for unseen data. One typical example of supervised learning is classification which, as defined above, builds models from labeled data that are able to predict class labels for future data. Classification algorithms normally use a training set where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model, also called a classifier, as shown in Figure 1.1. Once the model is validated through the testing data as shown in Figure 1.2, it is deployed in the field.

Unsupervised learning, on the other hand, attempts to discover hidden patterns in unlabeled data. A typical example of unsupervised learning is clustering which is a technique used to classify objects by splitting a diverse group into smaller groups of similar objects whose characteristics of similarity are not known in advance. As outlined in Section 1.5, we will be using clustering as a technique to meet one of our research objectives.



Outlook	HWDone	Weekend	Play
Sunny	TRUE	TRUE	Yes
Sunny	TRUE	FALSE	Yes
Rainy	FALSE	FALSE	No
Rainy	FALSE	TRUE	Yes
Sunny	FALSE	TRUE	Yes
Sunny	FALSE	FALSE	No
Rainy	TRUE	TRUE	Yes
Rainy	TRUE	FALSE	No

FIGURE 1.1: Classification Process - Classifier Construction

Since single classifier systems have limited predictive performance Yan and Goebel (2004), Polikar (2006), Maclin and Opitz (1999), Rokach (2010), to boost the performance of classification, ensemble classification was developed. Ensemble classification

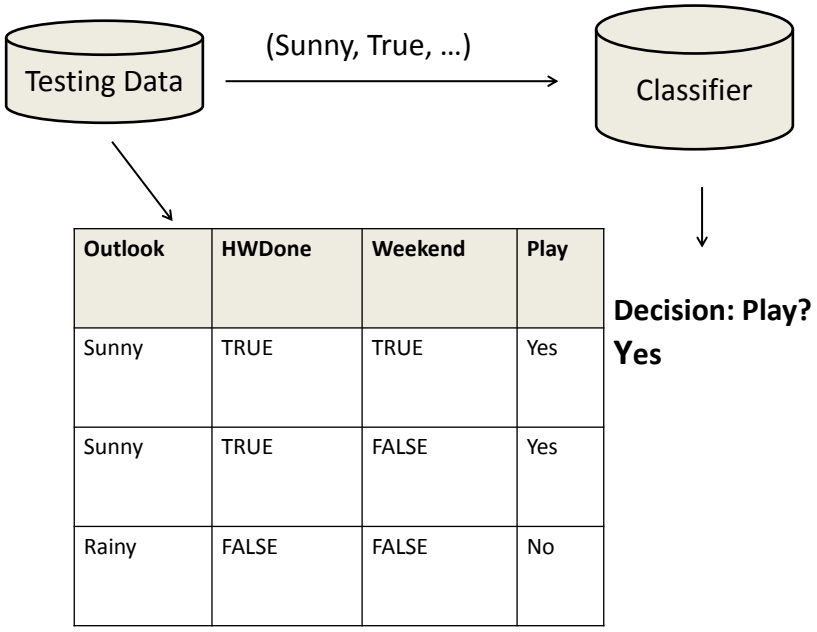


FIGURE 1.2: Classification Process - Prediction

is an application of ensemble learning which involves using multiple models that jointly work together to solve the same problem Polikar (2006), Rokach (2010), Kuncheva and Whitaker (2003). Likewise, in ensemble classification, multiple classifiers are used and tend to perform better than the individual classifiers in the ensemble. A popular ensemble classification technique that has attracted significant attention from the research community is Random Forest (RF) Breiman (2001). Ever since its inception back in 2001, researchers have striven to produce several extensions of RF in order to further enhance its performance. In an evaluation study made by Fernández-Delgado et al. (2014) where 179 classifiers arising from 17 families (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearestneighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods) were evaluated, RF has proven to be the best family of classifiers. The main aim of this dissertation is to enhance RF in two directions: extreme pruning and feature engineering. Extreme pruning produces a much more smaller ensemble than the original one with performance comparable and mostly better than the initial one. Feature engineering deals with improving feature interaction which is not an inherent characteristic of RF.

1.2 Problem Statement

A well established principle in ensemble classification is that ensembles tend to perform better when the individual classifiers in the ensemble exhibit a high level of diversity Kuncheva and Whitaker (2003), Brown et al. (2005), Adeva et al. (2005), Tang et al. (2006). With one common goal in mind, many researchers have exploited this principle and developed several expansions of RF in order to boost its performance. In this dissertation, we will exploit this principle even further by considering new diversity techniques that, to the best of our knowledge, have not been exploited before to produce more accurate RF ensembles. As will be discussed in the subsequent chapters, these techniques are clustering, the Local Outlier Factor, diversified weighted subspaces, and replicator dynamics.

Extensions of RF, regardless of the techniques they use, should improve performance accuracy as well as provide timely classification. In 2006, an international provider of on-demand Internet streaming media, called Netflix Inc., held an open competition to find a collaborative filtering algorithm that would best predict whether or not a user would like a particular film or TV show based on previous ratings. In 2009, a team known as "BellKor's Pragmatic Chaos" won the grand prize of $1 million. Though the team's algorithm was 10% more accurate than Netflix's own prediction system, Netflix never implemented the team's solution into its own service. The main reason being, as explained by Netflix, was that the team's 100 algorithm ensemble was too complex and computationally demanding. Of the 100 algorithms, only 2 were implemented by Netflix. The additional accuracy gain achieved by the rest of the algorithms, as Netflix justified, was not worth the engineering effort to integrate them into a production environment.

The Netflix story puts a huge emphasis on three highly desirable properties of classification algorithms, namely, simplicity, accuracy and speed. In this dissertation, we have developed non-computationally demanding RF methods/algorithms to support these properties. Speed wise and as outlined in Section 1.5, some the proposed methods developed perform ensemble pruning producing a much more smaller ensemble of the original one and therefore, run much faster. Accuracy wise, because high correlated trees in an RF can increase the forest error rate Breiman (2001), Bernard et al. (2010), the added diversity that has been injected by these methods is likely to reduce the correlation between trees and consequently, reduce the forest error rate.

1.3 Dissertation Goal

Considering the aforementioned evaluation study made by Fernández-Delgado et al. (2014) where RF has produced the best results, the main goal of this dissertation is to build on the success of RF and use different techniques and strategies to produce new methods of RF that produce more accurate ensembles. Since classification accuracy and classification speed are highly desirable properties for many applications, it is of paramount importance to develop methods that outperform the traditional RF. Examples of such applications include but not limited to medical diagnosis, autopsy, agriculture, and credit rating. Such applications have zero tolerance for misclassification that could lead to severe consequences.

1.4 Research Questions

In our quest for developing more accurate RF ensembles, we have identified the following questions. Answering them will help us achieve our goal:

- What techniques can be used to produce the least correlated trees in an RF ensemble?
- Can we achieve high pruning levels that allow RF to operate in real-time applications?
- Can we improve feature interactions in RF?
- Can we add a dynamic characteristic in RF that makes the model easily incrementally updatable?

1.5 Aims, Objectives and Approaches

There are two aims of this dissertation. The first is to perform extreme pruning on RF ensembles in order to produce ensembles that are much smaller in size than the original ones, and yet, perform as least as good as the original ones. The second aim is to apply feature engineering techniques in order to improve interactions among the features.

To meet the first aim, two objectives need to be addressed. In the first objective, trees are clustered into groups where each group contains similar trees. To eliminate redundant trees, a tree is then selected from each group. One approach to meet this objective is to utilize a known diversity technique called clustering Jain et al. (1999) to find diverse trees in the ensemble. This can be done by clustering the trees in the ensemble into a predefined number of clusters and then selecting a representative from each cluster as shown in Figure 1.3. In this figure, the original ensemble consisted of 20 trees. Since 4 clusters were used, the resulted ensemble after pruning contained only 4 trees. Therefore, the pruning level achieved was 80%.

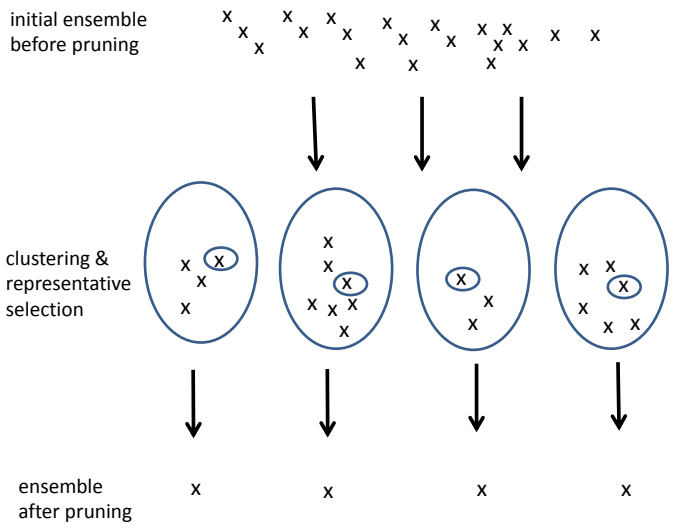


FIGURE 1.3: Extreme Pruning via Clustering

In the second objective, trees to be ranked according to a certain evaluation measure and then the ones with the highest ranking are selected. Doing so will eliminate many redundant trees. This approach can be met by utilizing the Local Outlier Factor (LOF) Breunig et al. (2000) for the first time ever to extreme prune RF ensembles by assigning each tree an LOF value and then selecting the top k (where k is a predefined integer) trees with the highest weighted LOF values as shown in Figure 1.4. In this figure, an 80% pruning level has also been achieved since the top 4 trees were picked to form the pruned ensemble.

By addressing two more objectives, the second aim can be met. In the first one, random projections of the training data are performed prior bagging in order to inject an additional level of diversity into the constituent trees of the ensemble. Such projections

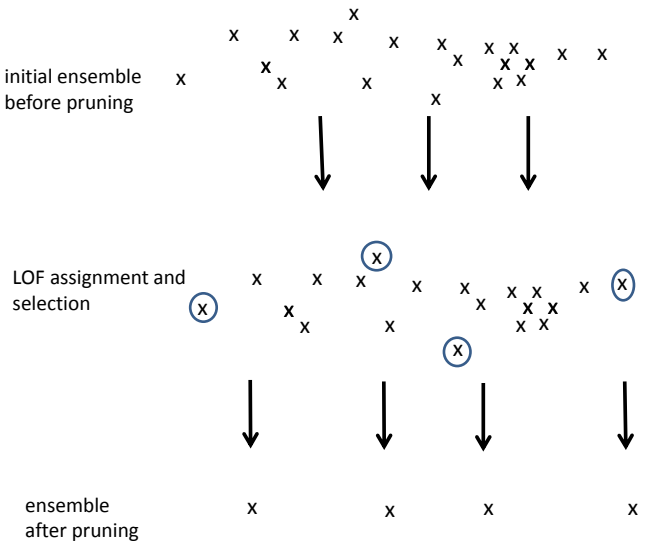


FIGURE 1.4: Extreme Pruning via Local Outlier Factor

are then given a weight to reflect their predictive power. The approach to meet this objective is to use diversified weighted subspaces where each subspace contains a random subset of the features. Each subspace eventually corresponds to a sub-forest. When all the sub-forests are taken collectively, they form a diversified Random Forest as shown in Figure 1.5. The trees in such a forest are likely to have improved feature interaction. The

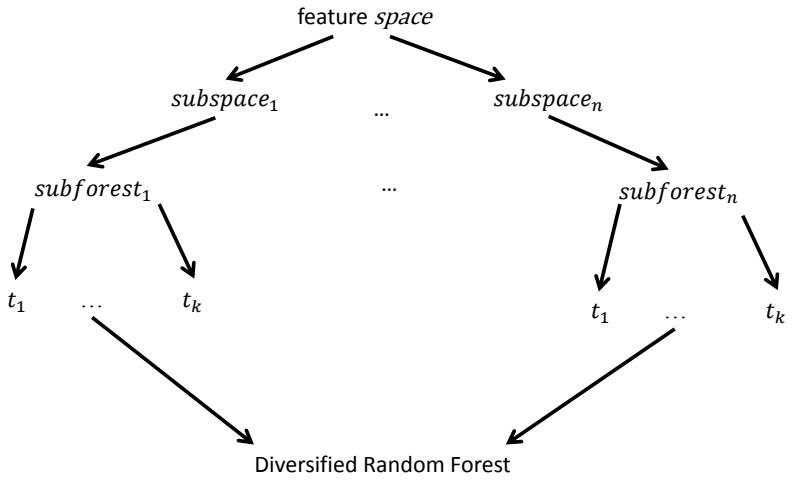


FIGURE 1.5: Diversified Random Forest via Diversified Subspaces

second objective aims at further improving the diversified forest of the previous objective by growing/shrinking the sub-forests. Poorly performing sub-forests, possibly due to noisy and correlated features are shrunk, and well performing sub-forests are grown. This objective can be met by using an approach heavily used in evolutionary game theory known as replicator dynamics Schuster and Sigmund (1983). The goal here is to

make the ensemble easily incrementally updatable by growing some sub-forests with new trees that can improve the performance, and shrink others by removing existing trees that can worsen the performance as shown in Figure 1.6.

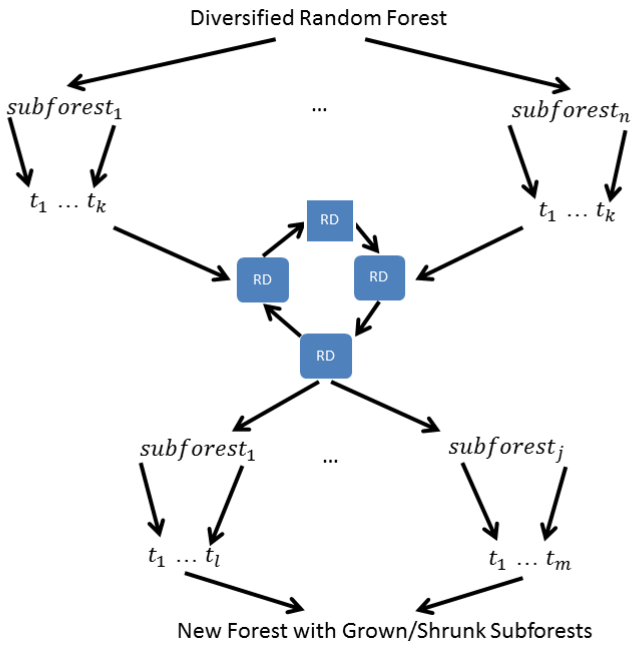


FIGURE 1.6: Applying Replicator Dynamics (RD) to a Diversified Random Forest

1.6 Research Contributions

The following contributions have been made and published as outlined in Section 1.8:

- To meet the objectives pertaining to the extreme pruning aim, the *CLUB-DRF* and *LOFB-DRF* methods have been developed. The first method is based on clustering and the second one is based on LOF. Though clustering was used before to prune other types of ensembles Bakker and Heskes (2003), Giacinto et al. (2000), Lazarevic and Obradovic (2001), Qiang et al. (2005), it has not been used before to prune RF ensembles, and the pruning levels achieved on other types of ensembles were not extreme. These methods were proposed to fill this research gap. As will be discussed in Chapters 4 and 5, both our ensemble pruning techniques have achieved an unprecedented pruning level reaching as high as 99% at the time of

boosting the predictive accuracy of the ensemble! This notably high pruning level makes these techniques ideal candidates for real-time applications.

- To meet the objectives pertaining to the feature engineering aim, the *DSB-RF* and *RDB-DRF* methods were developed. The *DSB-RF* method produces a diversified forest with diversified sub-forests based on diversified subspaces. Such a forest is likely to have improved feature interaction. *RDB-DRF* extends *DSB-RF* by applying RD to make it incrementally updatable.

1.7 Organization of the Thesis

The thesis will be divided into four main parts. In this section, the chapter(s) of each part are identified and a brief overview of the main topics discussed in each chapter is given.

- **Part I (Preliminaries)**: This part is composed of the first 3 chapters (including this chapter) and aims at laying out the foundation of the thesis. Following the introduction in this chapter, topics related to pruning, diversification, and Random Forests are covered. Chapter 2 covers topics related to ensemble pruning and diversification. The different categories of pruning techniques will be covered as well as topics related to diversity creation methods. An overview of Random Forests is presented in Chapter 3 where many related topics will be discussed including background, applications, strengths, and weaknesses.

- **Part II (Ensemble Pruning)**: consists of two contributions related to ensemble pruning covered in Chapters 4 and 5 respectively. Chapter 4 introduces *CLUB-DRF*: the first ensemble pruning method that is clustering-based. Key topics to be covered in this chapter include clustering, *CLUB-DRF*, and experimental study. Chapter 5 presents *LOFB-DRF*: the other ensemble pruning method that is based on the Local Outlier Factor (LOF). Topics to be covered in this chapter include LOF, *LOFB-DRF*, and experimental study.

- **Part III (Feature Engineering)**: contains two other contributions related to feature engineering and Replicator Dynamics (RD) covered in Chapters 6 and 7 respectively. Chapter 6 introduces *DSB-RF*: the method that applies feature

engineering to produce a diversified RF based on diversified subspaces. Key topics to be covered in this chapter include *DSB-RF* and experimental study. Chapter 7 enhances *DSB-RF* by applying RD. After a brief introduction to RD, details, coupled with an experimental study, are given to show how it can be applied to enhance *DSB-RF*.

- **Part IV (Final Remarks)**: describes conclusion and future work which are covered in Chapter 8.

1.8 Thesis Outcomes

The following papers have been published as a result of the research conducted in this thesis:

- Fawagreh, Khaled, Mohamed Medhat Gaber, and Eyad Elyan. **"Random forests: from early developments to recent advancements."** Systems Science & Control Engineering: An Open Access Journal 2.1 (2014): 602-609.
- Fawagreh, Khaled, Mohamed Medhat Gaber, and Eyad Elyan. **"Diversified Random Forests Using Random Subspaces."** Intelligent Data Engineering and Automated Learning–IDEAL 2014. Springer International Publishing, 2014. 85-92.
- Fawagreh, Khaled, Mohamed Medhat Gaber, and Eyad Elyan. **"CLUB-DRF: A Clustering Approach to Extreme Pruning of Random Forests"**,Research and Development in Intelligent Systems XXXII. Springer International Publishing, 2015. 59-73.
- Fawagreh, Khaled, Mohamed Medhat Gaber, and Eyad Elyan. **"A Replicator Dynamics Approach to Collective Feature Engineering in Random Forests"**, Research and Development in Intelligent Systems XXXII. Springer International Publishing, 2015. 25-41.
- Fawagreh, Khaled, Mohamed Medhat Gaber, and Eyad Elyan. **"An Outlier Detection-based Tree Selection Approach to Extreme Pruning of Random Forests"**, 17th International Conference on Engineering Applications of Neural Networks–EANN 2016. Springer International Publishing, 2016. 267-282.

Chapter 2

Ensemble Pruning and Diversification

2.1 Overview

This chapter provides a literature review on ensemble pruning and diversification that are relevant to the research conducted in this thesis. For ensemble pruning, this chapter looks at the different ensemble pruning techniques used to produce pruned ensembles. For diversification, the different diversity techniques are discussed and some common ensemble diversity measures are described.

2.2 Background

Ever since the inception of RF back in the year of 2001, several extensions have been developed to boost its performance. Such extensions can be classified into two main categories: those that use an ensemble pruning technique (see Section 2.4) to produce pruned RF ensembles and those that do not. Extensions in the first category start with an ensemble of a particular size and tend to produce a subset of this ensemble that performs at least as good as the original one. The ones in the second category use different techniques/strategies to construct more accurate ensembles without doing any pruning. Following an introduction to ensemble classification and pruning, we will explore these two categories in greater detail.

2.3 Ensemble Classification

As an application of ensemble learning, ensemble classification was developed to boost the accuracy of classification since single classifier systems have limited predictive performance Yan and Goebel (2004), Polikar (2006), Maclin and Opitz (1999), Rokach (2010). Ensemble learning involves using multiple models that jointly work together to solve the same problem Polikar (2006), Rokach (2010), Kuncheva and Whitaker (2003). Likewise, in ensemble classification, multiple classifiers are used and tend to perform better than the individual classifiers in the ensemble Polikar (2006), Rokach (2010), Maclin and Opitz (1999). To classify an instance, a voting scheme is used. A popular and simple voting scheme is majority voting Lam and Suen (1997) where each classifier in the ensemble casts a vote for its predicted class label. The class label that wins the majority of the votes is returned as the chosen candidate of the ensemble. An alternative voting scheme is veto voting where other classifiers decisions are vetoed by a single classifier Shahzad and Lavesson (2012), Sun and Dance (2012). As an extension of veto voting, a recent voting scheme is called trust-based veto voting Shahzad and Lavesson (2013). In this voting scheme, the trust of each classifier is considered to determine whether a classifier or a set of classifiers can veto the decision.

The accuracy and diversity of the individual classifiers in the ensemble are likely to make the ensemble produce favorable results. A classifier with an error rate less than random guessing is considered an accurate classifier, and two classifiers are considered diverse if they produce different prediction errors on unseen data Yang et al. (2007). The more diversity the classifiers exhibit, the more accurate results they tend to produce Brown et al. (2005). In fact, empirically speaking, ensembles tend to perform better when the constituent models exhibit high level of diversity Kuncheva and Whitaker (2003). This justifies why many ensemble methods strive to promote diversity among their constituent models Brown et al. (2005), Adeva et al. (2005).

Popular ensemble methods include boosting, bagging, and stacking. Boosting follows an incremental approach where, after building a sequence of classifiers, each classifier works on the incorrectly classified instances of the previous one in the sequence. A representative of this class of techniques is AdaBoost Freund and Schapire (1997). However, AdaBoost is known to suffer from overfitting Bylander and Tate (2006).

Bootstrap Aggregating (Bagging) Breiman (1996) is another class of ensemble approaches. In this approach, a randomly drawn sample with replacement of the data is used to build each classifier in the ensemble. When labeling unlabeled instances, each classifier gives an equal vote. With respect to model overfitting, bagging is known to be more impervious than boosting. A main representative of bagging is Random Forests Breiman (2001).

Stacking is historically one of the first ensemble learning methods. It combines several base classifiers, which can belong to absolutely different classes of machine learning methods, by means of a "meta-classifier" that takes as its inputs the output values of the base classifiers Wolpert (1992), Seewald (2002). Although stacking is a heuristic method and does not guarantee improvement in all cases, in many practical studies, it shows excellent performance.

2.4 Ensemble Pruning Techniques

Several enhancements have been made in recent years in order to produce a subset of an ensemble that performs as well as, or better than, the original ensemble. The purpose of ensemble pruning (also known as ensemble selection, selective ensemble and ensemble thinning) is to search for such a good subset. This is particularly useful for large ensembles that require extra memory usage, computational costs, and occasional decreases in efficiency Zhang et al. (2006). A survey of ensemble pruning techniques can be found in Tsoumakas et al. (2009) where such techniques were classified into three main categories: clustering-based, ranking-based, and optimization-based. The following subsections explore these techniques in greater detail.

2.4.1 Clustering-Based Techniques

Clustering based-techniques consist of two stages. In the first stage, a clustering algorithm is employed in order to discover groups of models that make similar predictions. Pruning each cluster then follows in the final stage. In this stage, several approaches have been used. One approach is by using the cluster centroids as values of the target variable, a new model is trained for each cluster Bakker and Heskes (2003). Another interesting approach was proposed by Giacinto et al. (2000) which involved selecting a

representative classifier from each cluster that is the farthest distant from the rest of the clusters. A yet different approach by Lazarevic and Obradovic (2001) that does not guarantee the selection of a single model from each cluster was by iteratively removing models from the least to the most accurate, until the accuracy of the entire ensemble starts to increase. Selecting the most accurate model from each cluster was proposed by Qiang et al. (2005).

When using a clustering algorithm that requires the number of clusters to be specified in advance (as in K-means), one interesting and challenging question that immediately comes to mind is how to determine this number so that it is not high and it is not low. Mardi Mardia et al. (1980) proposed a simple rule of thumb: $number\_of\_clusters \approx \sqrt{\frac{n}{2}}$, where n refers to the number of data points to be clustered. Following this rule can alleviate the empty cluster problem Pakhira (2009) from happening.

2.4.2 Ranking-Based Techniques

Ranking-based techniques use an evaluation measure in order to rank models. A diversity measure called Kappa pruning is used in Margineantu and Dietterich (1997) as an evaluation measure. By calculating the κ statistic of agreement on the training dataset, all pairs of classifiers are ranked. Kappa pruning can be generalized by accepting a parameter to specify any pairwise diversity measure for either classification or regression models. Another ranking-based pruning method for ensembles is orientation ordering Martínez-Muñoz and Suárez (2006). The signature vector of a classifier h_t is an important concept in orientation ordering. It is defined as an N-dimensional vector with elements taking the value +1 if $h_t(x_i) = y_i$ and -1 if $h_t(x_i) \neq y_i$. The ensemble signature vector is the average signature vector of all classifiers in an ensemble, and is an indicative of the ensemble's ability to correctly classify each example in the pruning set (the training set in this method) using majority voting for classifier combination.

In ranking-based ensemble pruning methods, an interesting issue that arises is the choice of the final number of models to choose following the ranking process. Selecting a fixed user-specified amount or percentage of models is a common method. In Kappa pruning for example, classifier pairs are selected in ascending order of agreement until a specified number of models has been reached.

2.4.3 Optimization-Based Techniques

Optimization-based techniques are classified into three categories: genetic algorithms, semi definite programming, and hill climbing. In the following subsections, the original ensemble H is denoted as $H = \{h_t \mid t = 1, 2, ..., T\}$ where T refers to the number of training samples in the training dataset, and h_t refes to a particular tree in the ensemble.

2.4.3.1 Genetic Algorithms

Genetic algorithms fall under the umbrella of evolutionary algorithms. Using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover Mitchell (1998), Haupt and Haupt (2004), Banzhaf et al. (1998), evolutionary algorithms generate solutions to optimization problems. The Gasen-b method Zhou and Tang (2003) used a genetic algorithm to perform stochastic search in the space of model subsets. Using one bit for each model, the ensemble is represented as a bit string. Depending on the value of the corresponding bit, models are included or excluded from the ensemble. In addition to the standard mutation and crossover operations, Gasen-b defined a fitness function for an individual $S \subseteq H$ as the accuracy of S on a separate validation set using voting for model combination.

2.4.3.2 Semi-Definite Programming

As a mathematical problem, Zhang et al. (2006) expressed ensemble pruning and then applied semi-definite programming (SDP) techniques to solve it. Initially, ensemble pruning was expressed as a quadratic integer programming problem to find a fixed-size subset of k classifiers with high accuracy and maximum divergence. Subsequently, this quadratic integer programming problem was found to resemble *the max cut with size k* problem, which, using an algorithm based on SDP, can be approximately solved. As an input parameter, the algorithm developed in Zhang et al. (2006) required the number of classifiers to be kept. It ran in polynomial time.

2.4.3.3 Hill Climbing

Hill climbing employs a greedy search algorithm where, from the neighborhood of the current state, the next state is visited. States represent the different subsets of models. The neighborhood of a subset $S \subseteq H$ contains the subsets that can be built by adding or removing one model from S. The greedy search utilized traverses the search space from one end representing an empty ensemble to the other end representing a complete ensemble. Two types of search approaches can be applied depending on the direction of the graph: forward selection Caruana et al. (2004), Fan et al. (2002), Partalas et al. (2009), Margineantu and Dietterich (1997), Martınez-Munoz and Suárez (2004) and backward elimination Banfield et al. (2005), Partalas et al. (2008), Yang et al. (2005). Forward selection involves starting with no variables in the model, testing the addition of each variable using a chosen model comparison criterion, adding the variable (if any) that improves the model the most, and repeating this process until none improves the model. Backward elimination, on the other hand, involves starting with all candidate variables, testing the deletion of each variable using a chosen model comparison criterion, deleting the variable (if any) that improves the model the most, and repeating this process until no further improvement is possible.

Like ranking-based methods, hill climbing ensemble pruning methods use an evaluation measure which can be either performance-based or diversity-based. For performance-based measures, accuracy was used in Margineantu and Dietterich (1997), Fan et al. (2002), Yang et al. (2005). In addition to accuracy, Caruana et al. (2004) experimented with other performance metrics including root-mean-squared-error, mean cross-entropy, lift, precision/recall break-even point, precision/recall F-score, average precision, and Receiver Operating Characteristic (ROC) area. A cost model based measure was used in Fan et al. (2002). Diversity-based measures, on the other hand, use a diversity metric to ensure that the models in the ensembles are diverse. To measure diversity, several diversity measures including disagreement, double fault, Kohavi-Wolpert variance, inter-rater agreement, generalized diversity and difficulty are used Tang et al. (2006). These are all covered in Section 2.6.

2.4.4 Other Techniques

Techniques that do not fall in any of the above categories are described here. A method based on the McNemar non-parametric test of significance was proposed in Latinne et al. (2001). The method a priori determines the minimum number of trees in the RF to use in order to obtain prediction accuracy comparable to the one obtained with larger ensembles. In addition to maintaining accuracy with fewer trees, the method significantly improves classification speed and reduces memory costs.

Pruning an ensemble of heterogeneous classifiers using statistical tests was proposed in Tsoumakas et al. (2005), Tsoumakas et al. (2004). Such tests determine whether the differences in predictive performance among the classifiers of the ensemble are significant. Only the classifiers with significantly better performance than the rest are retained and subsequently combined with the method of (weighted) voting.

A reinforcement learning approach to ensemble pruning was proposed in Partalas et al. (2006), Partalas et al. (2009). In specific, the problem of pruning an ensemble of T classifiers is modeled as an episodic task, where an agent takes T sequential actions, each one corresponding to either the inclusion or exclusion of a classifier $h_t \mid t = 1...T$, from the final ensemble. The Q-learning Watkins and Dayan (1992) algorithm is then used to approximate the optimal policy for this task.

2.5 Pruned/Unpruned RF Ensembles

Not much work has been reported to produce pruned RF ensembles that are smaller in size than the original ones. Consequently, this area is considered a hot research area that is worth investigating. In fact, as we shall see later in Chapters 4 and 5, we will explore this area further by introducing novel methods. The only prominent extension found in this area was by Latinne et al. (2001). As described in the previous subsection, the McNemar non-parametric test of significance was used to a priori determine the minimum number of trees in the RF.

As for unpruned RF ensembles, many extensions were developed to produce more accurate ensembles without reducing their initial size. What follows describes, in chronological order by year of publication, some of these extensions emphasizing on the techniques/strategies used by the perspective researchers to boost the performance.

New ways were investigated by Robnik-Šikonja (2004) to improve the performance of RF. By using several attribute evaluation measures instead of just one, the correlation between trees is decreased without any loss in their strength. Another way to improve the performance of RF as proposed by Robnik-Šikonja (2004) is to change the voting method. Instead of using majority voting, weighted voting is used. With this voting technique, internal estimates are used to identify instances most similar to the instance being labeled. The votes of the corresponding trees are then weighted with the strength they demonstrate on these near instances. Improvements were demonstrated on several classification datasets.

By replacing majority voting with more sophisticated dynamic integration techniques, Tsymbal et al. (2006) found a way to improve the performance of RF on some datasets. Three techniques were used: Dynamic Selection (DS), Dynamic Voting (DV), and Dynamic Voting with Selection (DVS). Using DV and DVS integration strategies, experimental studies showed that dynamic integration was able to improve the accuracy of RF on 12 out of 27 datasets.

The significance decline in RF when the number of features is large and the number of truly informative features is small (as in the DNA microarray dataset) was investigated by Amaratunga et al. (2008). A novel and simple approach to remedy this decline was proposed by picking the eligible subsets of features to split each node by weighted random sampling instead of simple random sampling, with the weights tilted in favor of the informative features. The approach demonstrated superior performance when applied to several actual microarray datasets.

To remedy the limitations of the off-line algorithm which has limited usability for many practical problems, Saffari et al. (2009) proposed a novel on-line RF algorithm. Ideas from on-line bagging, extremely randomized forests, and on-line decision tree growing procedures were combined to produce the on-line version of the algorithm. To boost performance, a temporal weighting scheme for adaptively discarding some trees based

on their out-of-bag-error was added. Experiments have shown that the performance of the on-line algorithm proved comparable to the off-line version.

Finally, Bader-El-Den and Gaber (2012) developed an approach to enhance the accuracy of RF by using genetic algorithms Goldberg (1989) . The approach was called Genetic Algorithm based Random Forest (GARF). Experiments have shown that GARF outperformed other state-of-the-art classification techniques including AdaBoost.

Table 8.1 below shows a summary, ordered in chronological order by year of publication, of the aforementioned extensions to boost the performance of RF.

TABLE 2.1: Summary of RF Extensions

Ref.	Technique(s)	Summary
Robnik-Šikonja (2004)	several attributes measures, weighted voting	Decreased correlation between trees by using several attribute evaluation measures. Used weighted voting instead of majority voting
Tsymbal et al. (2006)	dynamic intergration techniques	Replaced majority voting with more sophisticated dynamic integration techniques: Dynamic Selection (DS), Dynamic Voting (DV), and Dynamic Voting with Selection (DVS)
Amaratunga et al. (2008)	weighted random sampling	Improved the declining performance when the number of features is large and the number of truly informative features is small by using weighted random sampling instead of simple random sampling when picking features to split each node
Saffari et al. (2009)	on-line RF algorithm	Introduced a novel on-line RF algorithm to remedy the limitations of the off-line algorithm
Bader-El-Den and Gaber (2012)	genetic algorithms	Used genetic algorithms to boost the performance of RF

2.6 Diversity Creation Methods

Because of the vital role diversity plays on the performance of ensembles, it had received a lot of attention from the research community. The work done to date in this domain was summarized by Brown et al. (2005) from two main perspectives. The first is a review

of the various attempts that were made to provide a formal foundation of diversity. The second is a survey of the various techniques to produce diverse ensembles. For the latter, two types of diversity methods were identified: implicit and explicit. While implicit methods tend to use randomness to generate diverse trajectories in the hypothesis space, explicit methods, on the other hand, choose different paths in the space deterministically. In light of these definitions, bagging and boosting in the previous section are classified as implicit and explicit respectively.

Ensemble diversity techniques were also categorized in Brown et al. (2005) into three categories: starting point in hypothesis space, set of accessible hypotheses, and manipulation of training data. Methods in the first category use different starting points in the hypothesis space, therefore, influencing the convergence place within the space. Because of their poor performance of achieving diversity, such methods are used by many authors as a default benchmark for their own methods Maclin and Opitz (1999). Methods in the second category vary the set of hypotheses that are available and accessible by the ensemble. For different ensembles, these methods vary either the training data used or the architecture employed. In the third category, the methods alter the way space is traversed. Occupying any point in the search space gives a particular hypothesis. The type of the ensemble obtained will be determined by how the space of the possible hypotheses is traversed.

2.6.1 Diversity Measures

Though there have been many diversity measures developed over the past years as will be seen shortly, it is important to emphasize that measuring diversity is not a straightforward process due to the lack of a generally accepted formal definition Kuncheva and Whitaker (2003). The first doubt on diversity measures was presented by Kuncheva and Whitaker (2003), where several experiments revealed that the effectiveness of existing diversity measures is discouraging, since diversity measurements seem to have no relation with ensemble performance and therefore, are not true indicators of performance. In fact, because the right formulation and measures for diversity remain unsolved, Zhou (2012) argues that understanding ensemble diversity remains a holy grail problem.

Despite the surprizing revealation described above, ensemble diversity measures were nevertheless developed to measure the diversity of a certain technique or perhaps to

compare the diversity of two techniques. A theoretical analysis on six existing diversity measures was presented in Tang et al. (2006): disagreement measure Skalak (1996), double fault measure Giacinto and Roli (2001), Kohavi Wolpert variance Kohavi et al. (1996), inter-rater agreement Fleiss et al. (2013), generalized diversity Partridge and Krzanowski (1997), and measure of difficulty Tang et al. (2006). The goal was not only to show the underlying relationships between them, but also to relate them to the concept of margin, which is one of the contributing factors to the success of ensemble learning algorithms.

The following notation will be used in the following subsections:

L: total number of base classifiers

N: total number of training samples

P: average classification accuracy of the base classifiers on the training data

l_i: product of L and sum of the weights of the base classifiers that classify the training sample x_i incorrectly, $l_i = L \sum_{O_{ij}=-1} w_j$ where:

o_{ij}: is oracle output of the base classifier h_j on the training sample x_i (defined as 1 if the training sample x_i is classified correctly by the base classifier h_j, or -1 otherwise)

w_j: is the weight of the base classifier h_j

2.6.1.1 Disagreement Measure

The Disagreement Measure (DM) is used to measure the diversity between two base classifiers h_j and h_k, and is calculated as follows:

$$DM_{j,k} = \frac{n(1,-1) + n(-1,1)}{n(1,1) + n(1,-1) + n(-1,1) + n(-1,-1)} \tag{2.1}$$

where

- $n(1,-1)$: means number of training instances that were correctly classified by h_j, but are incorrectly classified by h_k
- $n(-1,1)$: means number of training instances that were incorrectly classified by h_j, but are correctly classified by h_k

- $n(1,1)$: means number of training instances that were correctly classified by h_j and h_k
- $n(-1,-1)$: means number of training instances that were incorrectly classified by h_j and h_k

The higher the DM, the more diverse the classifiers are. By taking the average of all pairs of base classifiers, diversity within the whole set of base classifiers is then calculated as:

$$DM = \frac{2}{L(L-1)} \sum_{j=1}^{L} \sum_{k=j+1}^{L} DM_{j,k} \tag{2.2}$$

In Equation 2.1, since $n(1,1) + n(1,-1) + n(-1,1) + n(-1,-1) = N$, the previous equation can be represented as

$$DM = \frac{2}{NL(L-1)} \sum_{j=1}^{L} \sum_{k=j+1}^{L} (n_{j,k}(1,-1) + n_{j,k}(-1,1)) \tag{2.3}$$

Diversity is proportional with DM and tends to increase when it increases.

2.6.1.2 Double-Fault Measure

The Double Fault Measure (DFM) uses a slightly different approach where the diversity between two classifiers is calculated as:

$$DFM_{j,k} = \frac{n(-1,-1)}{n(1,1) + n(1,-1) + n(-1,1) + n(-1,-1)} \tag{2.4}$$

Similar to DM, since the denominator=N, the diversity within the whole set of base classifiers is calculated as:

$$DFM = \frac{2}{NL(L-1)} \sum_{j=1}^{L} \sum_{k=j+1}^{L} n_{j,k}(-1,-1) \tag{2.5}$$

Diversity is conversely proportional with DFM and tends to decrease when the value of DFM increases.

2.6.1.3 Kohavi-Wolpert Variance

In their decomposition formula of the classification error of a classifier, Kohavi and Wolpert introduced the Kohavi-Wolpert (KW) variance in 1996. This measure originated from the bias-variance decomposition of the error of a classifier. The following equation gives the KW variance in its simplest form

$$KW = \frac{1}{NL^2} \sum_{i=1}^{N} l_i(L - l_i) \tag{2.6}$$

Diversity is proportional with KW and tends to increase when it increases.

2.6.1.4 Measurement of inter-rater agreement

A measure of inter-rater (inter-classifier) reliability was developed and called k to measure the level of agreement within a set of classifiers. To be diverse, this measure is based on the assumption that a set of classifiers should disagree with one another. The k is calculated as:

$$k = 1 - \frac{\sum_{i=1}^{N} l_i(L - l_i)l_i}{NL(L-1)P(1-P)} \tag{2.7}$$

Diversity is conversely proportional with k and tends to decrease when it increases.

2.6.1.5 Generalized Diversity

Proposed by Partidge and Krzanowski in 1997, this measure is similar to DFM. Given two classifiers, Partidge and Krzanowski argued that maximum diversity is achieved when one classifier yields incorrect classification while the other yields correct classification. They also argued that minimum diversity occurs when the two classifiers fail together. For a sample x_i that is randomly drawn from the training set, let T_j denote the probability that $l_i = j$, the Generalized Diversity (GD) is defined as:

$$GD = 1 - \frac{\sum_{j=1}^{L} \frac{j(j-1)}{L(L-1)} T_j}{\sum_{j=1}^{L} \frac{j}{L} T_j} \tag{2.8}$$

Diversity is proportional with GD and tends to increase when it increases.

2.6.1.6 The measure of "difficulty"

Developed by Hansen and Salamon in 1990. For a sample x_i that is randomly drawn from the training set, a discrete random variable V is defined where $V_i = (L - l_i)/L$. The measure of difficulty was defined as the variance of V over the whole training set as given by the following equation:

$$diff = var(V_i) \tag{2.9}$$

Diversity is conversely proportional with the measure of difficulty and tends to decrease when it increases.

2.7 Discussion

Looking back at the work done to boost the performance of RF in Section 2.5, we can see that several researchers focused on improving two of RF key merits, namely, feature selection and voting. For feature selection, Robnik-Šikonja (2004) and Amaratunga et al. (2008) provided alternative methods for selecting the best features to split at each node. As for voting, Robnik-Šikonja (2004) and Tsymbal et al. (2006) proposed new voting techniques that proved to be more efficient than the traditional majority voting technique.

As will be discussed in Chapter 3, increasing the strength of the individual trees in the RF, and decreasing the correlation between trees, are key factors in improving the performance by reducing the RF error rate. This approach was used by Bernard et al. (2010).

In Section 2.3, we have stressed the importance of diversity to achieve better performance. This principle was exploited extensively in pruning ensembles but unfortunately, other than RF ensembles. For example, the pruning in Bakker and Heskes (2003), Lazarevic and Obradovic (2001), Qiang et al. (2005) was performed on neural network ensembles. Pruning of multiple heterogeneous classifier systems including neural networks, MLPs , SVMs, and k-NN was addressed in Giacinto et al. (2000), Partalas et al. (2010). AdaBoost ensembles were pruned in Margineantu and Dietterich (1997), Zhang et al. (2006), and a novel ensemble pruning method for bagging ensembles was presented in Martínez-Muñoz and Suárez (2006).

In ensemble pruning, a key performance factor is to achieve a high pruning level. The higher the pruning level is, the fewer classifiers are needed, which, in turn, improves the classification speed. Table 2.2 depicts the maximum pruning level achieved by some authors in the area of ensemble pruning.

TABLE 2.2: Maximum Pruning Level Achieved

Ref.	Max Pruning Level
Qiang et al. (2005)	78.5%
Lazarevic and Obradovic (2001)	68%
Bakker and Heskes (2003)	98%
Latinne et al. (2001)	95%
Martınez-Munoz and Suárez (2004)	90%
Fan et al. (2002)	97%

As we shall see later in Chapters 4 and 5, we have achieved comparable and even better pruning levels reaching as high as 99% while retaining or outperforming the accuracy of the original ensemble. Moreover, as mentioned in Chapter 1, for the proposed methods, new techniques have been used that, to the best of our knowledge, have not been used before to extreme prune RF ensembles. These techniques are clustering (Chapter 4) and

the Local Outlier Factor (Chapter 5). *Because clustering has been used extensively as a diversity technique in many applications* Li et al. (2010), Kuncheva and Hadjitodorov (2004), Brown and Martin (1998), Shemetulskis et al. (1995), Lee et al. (2008), Sharpton et al. (2012), *it likely has the potential to diversify trees in Random Forests. Likewise, because the Local Outlier Factor has the ability to achieve diversity, and was actually one of 3 strategies used to obtain diversity when constructing an ensemble for the KDDCup 1999 dataset* Erich and Zimek (2011), *we would like to see if it is capable of diversifying trees in Random Forests.*

As for feature selection, new feature engineering techniques will be used that are different than the techniques developed by other researchers as discussed in Section 2.5. *To produce diverse trees with reduced correlation among them, we will build on the random subspace method that was initially introduced by* Ho (1998) *by assigning a weight to each subspace (that is constructed from a fixed randomized subset of the total number of features) according to its predictive power as described in Chapter 5. Each subspace eventually produces a sub-forest and during the voting process, a weighted voting technique will be applied to select the class label that receives the highest weight.*

To improve the previous technique further, in Chapter 7, we will use replicator dynamics to grow and shrink the sub-forests as appropriate. *The rationale here is that by growing the well-performing sub-forests, more weight is given to them during the voting process, and less weight is given to the poorly-performing sub-forests by shrinking them.*

2.8 Summary

After an introduction to ensemble classification, this chapter looked at the different ensemble pruning techniques. It then described the main distinction between pruned and unpruned RF ensembles, and identified a research gap in the area of pruning RF ensembles as not much work was reported in this area. Categories of ensemble pruning techniques were identified and the different diversity measures were described. For pruned and unpruned extensions of RF covered in Section 2.5, the key merits of RF that researchers manipulated have been highlighted, namely, feature selection and voting. To achieve comparable (and even better) pruning levels in our proposed pruning

enhancements of RF, the maximum pruning level achieved by the respective authors in pruning other types of ensembles has been highlighted.

Chapter 3

Random Forests

3.1 Overview

Findings reported in this chapter have been published in Fawagreh et al. (2014a). This chapter gives a gentle introduction to Random Forests. The main aim here is not only to show the reader how Random Forests work as ensembles, but also to identify their strengths and weaknesses. For the latter, we refer to specific weaknesses that the thesis will address and improve upon.

3.2 Background

Random Forests (RFs) is an ensemble learning method used for classification and regression. Developed by Breiman Breiman (2001) over a decade ago, the method combines Breiman's bagging sampling approach Breiman (1996), and the random selection of features, introduced independently by Ho (1995) Ho (1998) and Amit and Geman Amit and Geman (1997), in order to construct a collection of decision trees with controlled variation. Using bagging, each decision tree in the ensemble is constructed using a sample with replacement from the training data. Statistically, the sample is likely to have about 64% of instances appearing at least once in the sample. Instances in the sample are referred to as in-bag-instances, and the remaining instances (about 36%), are referred to as out-of-bag (OOB) instances. Each tree in the ensemble acts as a base classifier to determine the class label of an unlabeled instance. This is done via majority

voting where each classifier casts one vote for its predicted class label, then the class label with the most votes is used to classify the instance.

3.2.1 Example

To illustrate RFs and majority voting, consider the training data depicted in Table 3.1 which consist of eight samples and four features. An RF will be created to predict the value of the *Play* feature which will determine whether a child can play or not, given the predefined values for the other features, namely, *Outlook*, *HWDone*, and *Weekend*. For example, it is obvious from the training data that a child can play if (s)he finished the homework on a sunny day regardless of whether it is a weekend or weekday. The child cannot play on a rainy weekday even if (s)he finished the homework. To aid in classifying new samples, an RF of three trees was created as shown in Figure 3.1. Table 3.2 shows the result of casting votes to classify the sample (rainy,false,true,?), where a ? mark is used to indicate the value of the *Play* feature to be determined. As shown in Table 3.2, trees A and C voted for "yes", whereas tree B voted for "no". By majority voting, the winning vote is therefore "yes" (child can play).

TABLE 3.1: Training Data

Outlook	HWDone	Weekend	Play
sunny	true	true	yes
sunny	true	false	yes
sunny	false	true	yes
sunny	false	false	no
rainy	true	true	yes
rainy	true	false	no
rainy	false	true	yes
rainy	false	false	no

TABLE 3.2: Vote Casting for (rainy,false,true,?)

Tree	Vote
A	yes
B	no
C	yes

During the construction of the individual trees in RFs, randomization is also applied when selecting the best node to split on. Typically, this is equal to $\sqrt{F}$ where F is the number of features in the dataset.

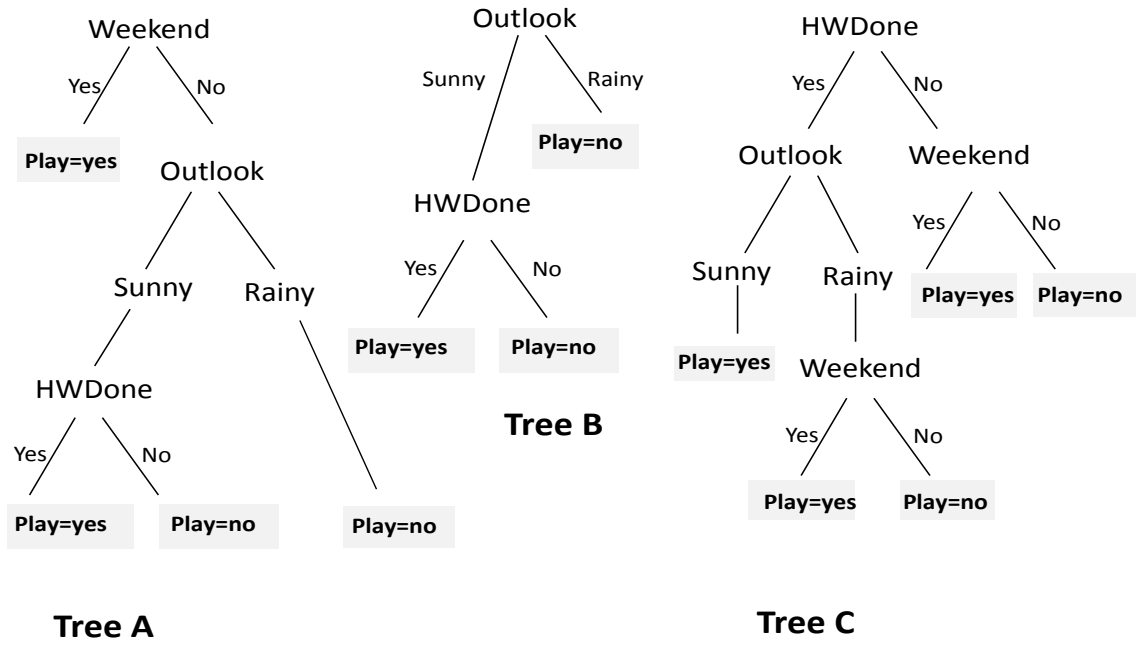


FIGURE 3.1: Random Forest of Three Trees

3.2.2 Algorithm

Algorithm 1 depicts the RFs algorithm where N is the number of training samples, and S refers to the bootstrap sample with replacement that will be used for growing the tree. As shown in the algorithm, N trees are constructed where each tree is grown as follows. Using Bootstrap sampling, a sample S is sampled with replacement out of all features F. This sample will be the training set for growing the tree. At each node in the tree, best split features are selected using the Gini index and are used to split the node. Each tree is grown to the largest extent possible and is unpruned.

In the original paper on Random Forests Breiman (2001), it was shown that the RF error rate depends on *correlation* and *strength*. Increasing the correlation between any two trees in an RF increases the forest error rate. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the RF error rate. Such findings seem to be consistent with a study made by Bernard et al. (2010) which showed that the error rate statistically decreases by jointly maximizing the strength and minimizing the correlation.

Algorithm 1 Random Forests Algorithm

{User Settings}
input N, S
{Process}
Create an empty vector $\overrightarrow{RF}$
for $i = 1 \rightarrow N$ **do**
  Create an empty tree T_i
  repeat
    Sample S out of all features F using Bootstrap sampling
    Create a vector of the S features $\overrightarrow{F_S}$
    Find Best Split Features $B(\overrightarrow{F_S})$
    Create A New Node using $B(\overrightarrow{F_S})$ in T_i
  until No More Instances To Split On
  Add T_i to the $\overrightarrow{RF}$
end for
{Output}
A vector of trees $\overrightarrow{RF}$

3.3 Applications

Over the past decade, many applications of RFs were developed in virtually all disciplines, and new applications are yet to be uncovered. The ones chosen in this section are by no means exhaustive as there are many; in fact, a number of papers can be written about RFs applications developed to date. In this section, we have selected some interesting ones.

In *Ecology*, Cutler et al. (2007) compared the accuracies of RFs and four other commonly used statistical classifiers using various species data collected from multiple locations in the USA. The results demonstrated superiority of RFs over the other techniques.

In *Medicine*, Klassen et al. (2008) conducted some experiments to explore several attribute selection methods with RFs that precisely classified cancer using a published benchmark dataset. Experimental results showed that RFs performed well for microarray data in terms of speed and accuracy with several different gene sets. Hu (2009) applied RFs to study the prediction of pathologic complete response in breast cancer. Results showed that the feature selection scheme of RFs was able to identify important genes of biological significance.

In *Astronomy*, Gao et al. (2009) conducted some experiments on multi-wavelength data

classification. Results showed that RFs proved effective for astronomical object classification. Thanks to its own virtues in classification, feature selection, feature weighting, and detection of outliers, RFs have proved to be superior.

In *Autopsy*, Flaxman et al. (2011) introduced a new Computer-Coded Verbal Autopsy (CCVA) method using RFs to predict cause of death. This was done by training RFs to distinguish between each pair of causes, and then combining the results through a novel ranking technique. The new method outperformed Physician-Certified Verbal Autopsy (PCVA) and was recommended for analyzing past and current verbal autopsies.

In *Traffic and Transport Planning*, Zaklouta et al. (2011) used K-d trees and RFs to classify 43 types of traffic signs using different size Histogram of Oriented Gradients (HOG) descriptors and distance transforms. Results showed that RFs outperformed K-d trees by achieving a classification rate of 97.2% and 81.8% on HOG and distance transforms respectively.

In *Agriculture*, Löw et al. (2012) used a combination of RFs and Support Vector Machine (SVM) classifiers to improve crop classification accuracy, and to provide spatial information on map uncertainty. Results showed that the feature selection merit of RFs improved the performance of SVM. Using this hybrid classifier improved classification accuracy compared with single classifiers and user's and producer's accuracy.

In *Bioinformatics* and *Computational Biology*, Boulesteix et al. (2012) amalgamated ten years of RFs development. Practical aspects of RFs including selection of parameters, available RFs implementations, important pitfalls, and biases of RFs and their Variable Importance Measures (VIM) were covered. The paper also surveyed recent developments relevant to Bioinformatics as well as some representative examples of RFs applications in this domain. Table 3.3 below depicts a summary of the RFs applications covered in this section.

TABLE 3.3: Summary of RFs Applications

Ref.	Domain	Aim	Summary
Cutler et al. (2007)	Ecology	Species Classification	Compared RFs accuracy with other techniques to classify various species. RFs proved to be superior to others.
Klassen et al. (2008)	Medicine	Cancer Classification	Did experiments to prove that RFs were able to precisely classify cancer. Also found that RFs performed well for microarray data in terms of speed and accuracy with several different gene sets.
Hu (2009)	Medicine	Prediction of Pathologic Complete Response in Breast Cancer	Applied RFs to study the prediction of pathologic complete response in breast cancer. Results showed that the feature selection scheme of RFs was able to identify important genes of biological significance.
Gao et al. (2009)	Astronomy	Astronomical Object Classification	Conducted some experiments on multi-wavelength data classification. Results showed that RFs proved effective for astronomical object classification.
Flaxman et al. (2011)	Autopsy	Cause of Death Prediction	Introduced a new computer-coded verbal autopsy (CCVA) method using RFs to predict cause of death.
Zaklouta et al. (2011)	Traffic and Transport Planning	Traffic Signs Classification	Used K-d trees and RFs to classify 43 types of traffic signs using different size Histogram of Oriented Gradients (HOG) descriptors and distance transforms.
Löw et al. (2012)	Agriculture	Accurate Classification of Crops	Used a combination of RFs and Support Vector Machine (SVM) classifiers to improve crop classification accuracy, and to provide spatial information on map uncertainty.
Boulesteix et al. (2012)	Bioinformatics	Recent Developments Survey in Bioinformatics	Amalgamated ten years of RFs development including representative examples of RFs applications in this domain.

3.4 Strengths

Key advantages of RF over its AdaBoost counterpart are robustness to noise and overfitting Breiman (2001), Liaw and Wiener (2002), Robnik-Šikonja (2004), Boinee et al. (2005). Overfitting generally occurs when a model is constructed in such a way that it fits the data more than is warranted. A model which overfits will generally have poor predictive performance, as it does not generalize well. By generalization we mean how well will the model make predictions for cases that are not in the training set. Hawkins et al. (2004) pointed out that overfitting adds complexity to a model without any gain in performance, or even worse, leads to poorer performance. A classifier that suffers from overfitting is likely to have a low error rate for the training instances (in-bag-instances), and a higher error rate for the out-of-bag instances.

Other strengths as listed in the original paper on RFs Breiman (2001) are (1) accuracy is as good as AdaBoost and sometimes better; (2) it's faster than bagging or boosting; (3) it gives useful internal estimates of error, strength, correlation and variable importance; and (4) it's simple and easily parallelized.

3.5 Weaknesses

Though RFs have proven to be the leading ensemble classification technique according to an evaluation study made by Fernández-Delgado et al. (2014), where 179 classifiers arising from 17 families were evaluated, RFs do have some weaknesses. Firstly, because current RFs algorithms tend to build ensembles with the number of trees in the range 100 to 500 Williams (2011), such ensembles are not suitable for real-time applications where fast classification is an important desideratum. Secondly, for high-dimensional datasets, RFs algorithms are likely to produce highly-correlated trees which can increase the forest error rate Breiman (2001), Bernard et al. (2010). Thirdly, feature interaction is not an inherent characteristic of RFs where decision trees are constructed without considering that certain features interact well with some features but not well with others.

To remedy the first weakness, we have developed the enhancements *CLUB-DRF* and *LOFB-DRF* in Chapters 4 and 5 respectively to extreme prune RFs. In doing so, the classification speed is improved significantly making these enhancements ideal for

real-time applications. As for the second and third weaknesses, we have developed the enhancements *DSB-RF* and *RDB-DRF* in Chapters 6 and 7 respectively to improve feature interaction and to reduce the correlation among trees.

3.6 Summary

A gentle introduction to Random Forests was presented in this chapter. As a leading state of the art classification and regression technique that is robust to overfitting, Random Forests have been used extensively in many applications as outlined in Section 3.3. Needles to say that Random Forests have proven to be the best family of classifiers among 17 families Fernández-Delgado et al. (2014), and a former winner of predictive modeling and analytics competitions held by Kaggle Narayanan et al. (2011). A Random Forest ensemble consists of a collection of decision trees where each decision tree is constructed from a sample with replacement from the training data as depicted in Algorithm 1. Strengths and weaknesses have been described with special attention to the weaknesses that will be addressed in this thesis. In the next chapter, we introduce the *CLUB-DRF* enhancement to produce pruned Random Forests ensembles that are much smaller in size than the original ones, and tend to perform better.

Part II

Ensemble Pruning

Chapter 4

CLUstering-Based Diverse Random Forest (CLUB-DRF)

4.1 Overview

Findings reported in this chapter have been published in Fawagreh et al. (2015). This chapter introduces the first enhancement related to extreme pruning which we have termed *CLUB-DRF*. As will be demonstrated in the experimental study, we have achieved extreme pruning levels reaching as high as 99% (to the best of our knowledge, this is the highest pruning level ever reported) while retaining or outperforming the accuracy of the original RF from which *CLUB-DRF* was derived. This makes *CLUB-DRF* ideal for real-time applications where classification speed is an important factor.

4.2 Background

Compared with other ensemble techniques, RFs, introduced in the previous chapter, have proved its accuracy and superiority Fernández-Delgado et al. (2014). Many researchers, however, believe that there is still room for enhancing and improving its performance accuracy. This explains why, over the past decade, there have been many extensions of RF where each extension employed a variety of techniques and strategies to improve certain aspect(s) of it. As discussed in Section 2.3, since it has been proven empirically that ensembles tend to yield better results when there is a significant diversity among

the constituent models, the objective of this chapter is twofold. First, it investigates how data clustering (a well known diversity technique) can be applied to identify groups of similar decision trees in RFs in order to eliminate redundant trees by selecting a representative from each group (cluster). Second, these likely diverse representatives are then used to produce an enhancement of RF, termed *CLUB-DRF*, that is much smaller in size than RF, and yet performs at least as good as RF, and mostly exhibits higher performance in terms of accuracy. The latter refers to a known technique called ensemble pruning (see Section 2.4 in Chapter 2). Experimental results on 15 real datasets from the UCI repository prove the superiority of our proposed enhancement over the traditional RF. Most of the experiments achieved unprecedented extreme pruning levels reaching as high as 99% while retaining or outperforming the RF accuracy.

4.2.1 Motivation

Current RF algorithms tend to build between 100 and 500 trees Williams (2011). Some empirical and theoretical studies have also clearly demonstrated that adding more trees to an RF beyond a certain number (i.e. 500) won't necessarily improve the RF accuracy Bernard et al. (2009). Our research aims at pruning RF ensembles by producing subsets of the original ones that are significantly smaller in size and yet, have accuracy performance that is at least as good as that of the original RF from which they were derived. In other words, we are aiming at finding the optimial or near-optimal number of trees that will be used to generate an accurate RF.

4.3 Clustering

Clustering has been used extensively as a diversity technique in many applications Li et al. (2010), Kuncheva and Hadjitodorov (2004), Brown and Martin (1998), Shemetulskis et al. (1995), Lee et al. (2008), Sharpton et al. (2012). Unlike classification, clustering is an unsupervised learning technique that attempts to organize objects into clusters (groups) where the members in one cluster are more similar to each other than those members in other clusters. Each group is referred to as a cluster, hence, a cluster is a group of similar objects which are dissimilar to other objects belonging to other clusters. Clustering is considered a data exploration method as it helps to unveil the

natural grouping in a dataset without a prior knowledge of the groups to be produced. One of the earliest and most popular clustering algorithms is called K-means. It was developed by MacQueen et al. MacQueen et al. (1967) in the late sixties and despite its seniority, it is still considered as one of the most widely used algorithms, mainly due to its simplicity, efficiency, and empirical success Jain (2010).

Unfortunately, however, this algorithm has a limitation that it only works with numerical data. To overcome this limitation, there have been some extensions of this algorithm to work with categorical data Huang (1998) Huang and Ng (1999) San et al. (2004). Huang Huang (1998) developed an extension of K-means called K-modes that uses modes instead of means, and can handle categorical data using the following simple matching dissimilarity measure:

$$d_1 = \sum_{j=1}^{m} \delta(x_j, y_j) \tag{4.1}$$

where x, y be two categorical objects with m categorical attributes, x_j and y_j (j=1..m) refer to the categorical attributes of X and Y respectively and

$$\delta(x_j, y_j) = \begin{cases} 0, & \text{if } x_j = y_j \\ 1, & \text{otherwise} \end{cases} \tag{4.2}$$

According to the above dissimilarity measure, for the two categorical objects provided, the total number of discrepancies, of the corresponding attribute categories, is calculated. The similarity of the two objects is conversely proportional to the number of mismatches; the smaller it is, the more similar the two objects are. Algorithm 2 outlines the main steps involved in the K-modes algorithm Huang (1997) where the constant k refers to the number of clusters to be created and *Objects* refers to the set of objects to be clustered.

The algorithm starts by assigning an initial mode to each cluster. Each object is then allocated to the cluster whose mode is nearest to it according to Equation 4.1. After each allocation, the modes are updated and the dissimilarity of each object against the current modes is retested. If the object's nearest mode belongs to another cluster, it is moved to that cluster. This process is repeated until no further movement of objects is possible between clusters.

Algorithm 2 K-modes Algorithm

```
{User Settings}
input k, Objects
{Process}
Create a set Clusters containing k clusters
Select k initial modes, one for each cluster
for i = 1 → Objects.size() do
  for j = 1 → k do
    if (isNearestTo(Objects.element(i), Clusters.element(j))) then
      Allocate(Objects.element(i),Clusters.element(j))
      UpdateMode(Clusters.element(j))
    end if
  end for
end for
repeat
  for i = 1 → Objects.size() do
    for j = 1 → k do
      if (isNearestTo(Objects.element(i), Clusters.element(j))) then
        m=findCurrentCluster(Objects.element(i))
        Reallocate(Objects.element(i), Clusters.element(j))
        UpdateMode(Clusters.element(m))
        UpdateMode(Clusters.element(j))
      end if
    end for
  end for
until No More Movement of Objects Between Clusters
{Output}
Clusters
```

In the experimental stage we will be using a popular machine learning software suite called Waikato Environment for Knowledge Analysis (WEKA) Hall et al. (2009). This suite comes with a clustering algorithm called SimpleKMeans that integrates K-means and K-modes to work with both numerical and categorical data.

4.4 CLUB-DRF

In this section, we propose an enhancement of RF called *CLUB-DRF* that spawns a child RF that is 1) much smaller in size than the parent RF and 2) has an accuracy that is at least as good as that of the parent RF. In the remainder of this chapter, we will refer to the parent/original traditional RF as simply *RF*, and refer to the resulted child RF based on our method as *CLUB-DRF*.

4.4.1 CLUB-DRF Algorithm

Figure 4.1 shows the *CLUB-DRF* approach and the corresponding algorithm is displayed in Algorithm 3 below where T refers to the training dataset and N refers to the number of training samples. The constant k refers to the number of clusters to be created which we define as a multiple of 5 in the range 5 to 40. This way and as we shall see in the experimental section, we can compare the performance of *CLUB-DRF* of different sizes with that of *RF*. As outlined in the experimental section, the size of the parent *RF* to be created is 500 trees. When the number of clusters is multiple of 5 in the range 5 to 40, this means the pruning levels will be in the range 99% to 92% respectively, which we consider a reasonable range for extreme pruning. Initially, we considered a maximum of 50 clusters to achieve pruning levels in the range 99% to 90%, however, an empty cluster problem Pakhira (2009) surfaced so we had to reduce the maximum to 40 clusters.

As shown in Algorithm 3, a clustering-based technique is applied to produce diverse groups of trees in *RF*. Assuming that the trees in *RF* are denoted by the vector $\overrightarrow{RF}$=<t_1, t_2,$\cdots$,t_n> (where n is number of trees in *RF*), and the training set is denoted by T=$\{r_1,r_2,\cdots,r_m\}$. Each tree in *RF* will then be used to classify each record in the training set to determine the class label c. We use $C(t_i, T)$ (where $t_i \in \overrightarrow{RF}$) to denote a vector of class labels obtained after having t_i classify the training set T. That is, $C(t_i, T) = <c_{i1}, c_{i2}, \cdots, c_{im}>$. The result obtained of having each tree classify the training records will therefore be a super vector $\overrightarrow{\overset{\circ}{A}}$ containing class labels vectors as classified by each tree: $\overrightarrow{\overset{\circ}{A}} = C(t_1, T) \cup C(t_2, T) \cup ...C(t_n, T)$
This set will be fed as input to a clustering algorithm as shown in Figure 4.1. When clustering is completed, we will have a set of clusters where each cluster contains vectors that are similar and likely to have the least number of discrepancies. For example, using a training set of 5 records with the class label being a boolean (Y/N), the vectors $<Y, Y, Y, N, N>$ and $<Y, Y, Y, Y, N>$ are likely to fall in the same cluster. However, the vectors $<Y, Y, Y, N, N>$ and $<Y, N, N, Y, Y>$ are likely to appear in different clusters because there are many discrepancies in the class labels. As defined above, since a cluster contains similar objects that are not similar to other objects belonging to other clusters, a vector of class labels as classified by a tree in one cluster will be dissimilar (different) from another vector belonging to another cluster, and this means that the two vectors are diverse.

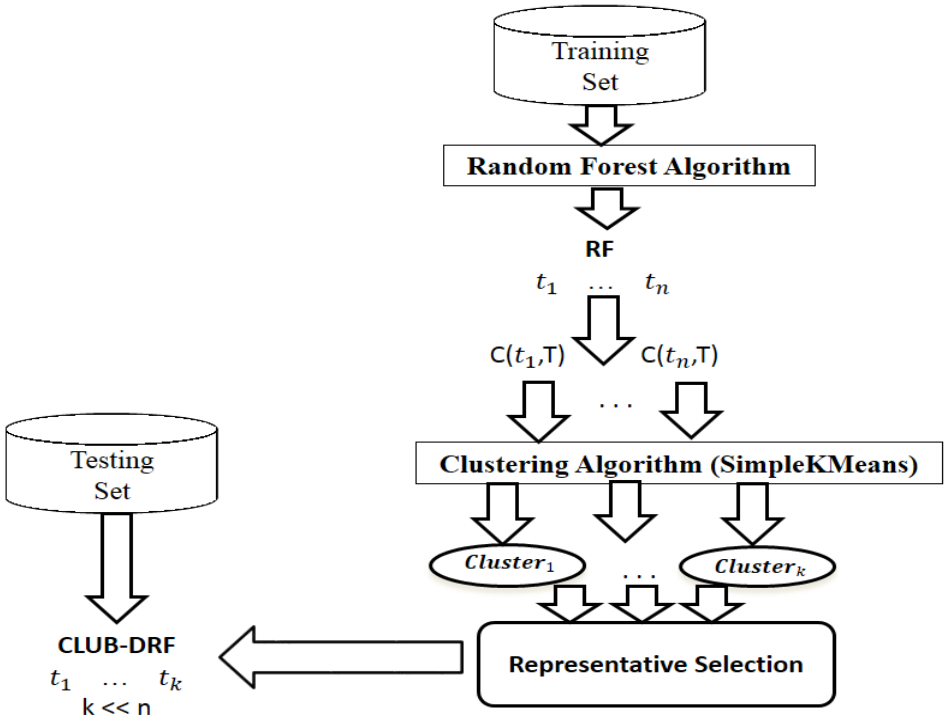


FIGURE 4.1: CLUB-DRF Approach

It is important to remember that the size of the resulted *CLUB-DRF* is determined by the number of clusters used. For example, if the number of the clusters is 5, then the resulted *CLUB-DRF* will have size 5, and so on.

The final step in the algorithm is to select a representative from each cluster. For this, three variations will be used and are discussed next.

4.4.1.1 Best Representative on Training CLUB-DRF

In this variation, from each cluster, we loop over the instances in each cluster and find the accuracy of their corresponding trees on the training data. The representative tree selected from each cluster is the one that has achieved the highest performance on the training data. According to Algorithm 3, the tree will then be added to *CLUB-DRF*. The resulted *CLUB-DRF* that uses this variation will be refrred to as *BestRepOnTraining CLUB-DRF*.

Algorithm 3 CLUB-DRF Algorithm

{User Settings}
input T, N, k
{Process}
Create an empty vector $\overrightarrow{\mathring{A}}$
Create an empty vector $\overrightarrow{RF}$
Create an empty vector $\overrightarrow{CLUB\text{-}DRF}$
Using N, call Random Forests Algorithm (Algorithm 1, *Chapter* 3) to create $\overrightarrow{RF}$
for $i = 1 \rightarrow \overrightarrow{RF}.size()$ **do**
  $\overrightarrow{\mathring{A}} = \overrightarrow{\mathring{A}} \cup C(\overrightarrow{RF}.tree(i), T)$
end for
Cluster $\overrightarrow{\mathring{A}}$ into a set of k clusters: $cluster_1$ ... $cluster_k$
for $i = 1 \rightarrow k$ **do**
  Find a representative $C(\overrightarrow{RF}.tree(j), T)$ in $cluster_i$
  Add its $\overrightarrow{RF}.tree(j)$ to $\overrightarrow{CLUB\text{-}DRF}$
end for
{Output}
A vector of trees $\overrightarrow{CLUB\text{-}DRF}$

4.4.1.2 Best Representative on OOB CLUB-DRF

In this variation, a similar approach is used as in the previous variation except that we pick the tree that has achieved the highest performance on the out-of-bag (OOB) instances. As mentioned in Section 3.2 in Chapter 3, these are the instances that were not included in the sample with replacement that was used to build the tree, and they account for about 36% of the total number of instances. Using the OOB samples to evaluate a tree gives an unbiased estimate of its predictive accuracy since, unlike training data that was seen by the tree when it was built, OOB data was not seen and therefore, it is a more accurate measure of the tree's predictive accuracy. The resulted *CLUB-DRF* that uses this variation will be referred to as *BestRepOnOOB CLUB-DRF*.

4.4.1.3 Random Representative CLUB-DRF

In this variation, from each cluster, we randomly pick an instance and select its corresponding tree without assessing its performance. The resulted *CLUB-DRF* that uses this variation will be referred to as *RandomRep CLUB-DRF*. Unlike the previous two variations which have been overfitted on the training and OOB samples respectively, this variation does not suffer from overfitting as accuracy was not involved as a selection criterion in the selection of the representative.

4.5 Experimental Study

For our experiments, 15 real datasets with varying characteristics from the UCI repository Bache and Lichman (2013) have been used. Table 4.1 below lists these datasets in ascending order by the number of instances. These datasets will also be used in all the other proposed enhancements in Chapters 5, 6, and 7.

TABLE 4.1: Experiments Datasets

Name	Number of Features	Number of Instances
pasture	23	36
squash-unstored	24	52
squash-stored	25	52
white-clover	32	63
sonar	61	208
glass	10	214
breast-cancer	10	286
vote	17	435
soybean	36	683
eucalyptus	20	736
diabetes	9	768
vehicle	19	846
credit	21	1000
car	7	1728
audit	13	2000

To use the holdout testing method, each dataset was divided into 2 sets: training and testing. Two thirds (66%) were reserved for training and the rest (34%) for testing. The size of the parent *RF* (refer back to Figure 4.1) was 500 trees; a typical upper limit setting for RFs Williams (2011). This upper limit was chosen for two main reasons. First, the more trees we have, the more diverse ones we can get. Secondly, for many clusters, the more trees there are, the more unlikely the problem of empty clusters Pakhira (2009) will surface.

The *CLUB-DRF* algorithm described above was implemented using the Java programming language utilizing the API of WEKA Hall et al. (2009). It was run 10 times on each dataset where a new *RF* was created in each run. The average of the 10 runs for each resulted *CLUB-DRF* was calculated to produce the average for a variety of metrics including accuracy, standard deviation, F-Measure, and Area Under Curve (AUC). The minimum and maximum accuracy obtained over the 10 runs for each cluster increment

were also calculated. For *RF*, only the average accuracy, F-Measure, and AUC have been calculated as shown in the last 3 columns in Tables 4.2, 4.4, and 4.6.

Due to the lack of a generally accepted formal definition for measuring diversity in ensembles, and the discouraging effectiveness of existing diversity measures Kuncheva and Whitaker (2003) as discussed in Subsection 2.6.1, no diversity measure was ever implemented in this chapter's experiments, and hence, no diversity metrics were ever reported. This also applies to experiments carried out in other chapters.

4.5.1 Results

Results of the above 3 variations of *CLUB-DRF* on the 15 datasets are reported next. In the result tables of all variations, the average accuracy of the resulted *CLUB-DRF* has been highlighted in boldface when it is greater than that of *RF*, and underlined when it is equal to that of *RF*. In this section, for simplicity, we will refer to the resulted forest produced by each variation as *CLUB-DRF*.

4.5.1.1 Best Representative on Training CLUB-DRF Results

Table 4.2 compares the performance of *CLUB-DRF* and *RF*. Taking a closer look at this table, observe that *CLUB-DRF* outperformed *RF* on 14 datasets. Interestingly enough, of the 14 datasets, in all settings of *CLUB-DRF*, *CLUB-DRF* completely outperformed *RF* on 5 of the datasets, namely, *pasture*, *eucalyptus*, *car*, *sonar*, and *vehicle*. More precisely, a pruned RF using *CLUB-DRF*, regardless of its size, outperformed *RF*. For the *sonar* dataset, the poor performance by both *RF* and *CLUB-DRF* is attributed to the fact that this dataset is a timer-series dataset, and trees in general do not work well with such datasets. Nevertheless, as shown in Table 4.2, the pruned *CLUB-DRF* ensemble outperformed the original *RF* ensemble which is what ensemble pruning is all about.

When comparing the best performer *CLUB-DRF* accuracy with that of *RF* for each dataset, the result is statistically significant as demonstrated by a p-value of 0.003237 using the paired t-test with 95% confidence.

TABLE 4.2: BestRepOnTraining CLUB-DRF & RF

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure	AUC
breast-cancer								
5	69.07	60.82	72.16	3.23	0.65	0.59	69.79	0.63	0.58
10	**72.37**	70.10	77.32	2.10	0.64	0.59		
15	**70.31**	67.01	72.16	1.65	0.64	0.59		
20	**71.34**	67.01	74.23	2.39	0.64	0.59		
25	**70.21**	67.01	75.26	2.54	0.64	0.58		
30	69.69	65.98	73.20	2.07	0.63	0.58		
35	69.48	65.98	73.20	2.36	0.64	0.58		
40	**70.82**	69.07	73.20	1.39	0.64	0.58		
audit								
5	96.15	93.81	97.35	0.93	0.92	0.89	96.56	0.92	0.88
10	**96.70**	96.46	96.90	0.14	0.93	0.89		
15	96.42	95.87	96.61	0.20	0.92	0.88		
20	**96.68**	96.46	96.90	0.15	0.92	0.89		
25	96.36	95.87	96.61	0.22	0.92	0.88		
30	96.55	96.31	96.90	0.20	0.92	0.88		
35	96.47	96.31	96.90	0.18	0.92	0.88		
40	**96.59**	96.46	96.76	0.10	0.92	0.88		
credit								
5	**77.88**	69.12	88.24	6.03	0.68	0.63	77.18	0.67	0.61
10	**77.29**	71.76	81.76	3.28	0.68	0.62		
15	**78.15**	75.29	80.29	1.87	0.68	0.62		
20	**77.65**	75.88	80.00	1.45	0.67	0.62		
25	76.62	75.00	78.53	0.94	0.67	0.62		
30	76.74	74.71	78.24	1.09	0.67	0.61		
35	76.41	74.12	79.12	1.40	0.67	0.61		
40	76.47	73.82	78.24	1.27	0.67	0.61		
pasture								
5	**48.33**	25.00	66.67	13.84	0.48	0.60	42.50	0.43	0.57
10	**60.00**	33.33	83.33	15.72	0.45	0.57		
15	**45.00**	25.00	66.67	13.02	0.43	0.56		
20	**51.67**	33.33	66.67	11.06	0.44	0.56		
25	**47.50**	33.33	75.00	12.94	0.44	0.57		
30	**54.17**	41.67	75.00	10.70	0.45	0.58		
35	**44.17**	25.00	58.33	9.17	0.43	0.56		
40	**49.17**	41.67	58.33	6.92	0.43	0.57		
squash-unstored								
5	**75.00**	41.67	91.67	13.44	0.61	0.71	68.33	0.58	0.70
10	**74.17**	58.33	83.33	7.86	0.61	0.72		
15	**71.67**	41.67	91.67	12.47	0.60	0.71		
20	**74.17**	58.33	91.67	10.83	0.61	0.71		
25	66.67	58.33	83.33	9.13	0.59	0.71		
30	**72.50**	58.33	83.33	9.17	0.60	0.71		
35	**70.00**	58.33	83.33	6.67	0.59	0.71		
40	66.67	50.00	75.00	10.54	0.59	0.69		
squash-stored								
5	**62.50**	41.67	83.33	11.33	0.56	0.64	57.50	0.50	0.58
10	**63.33**	50.00	75.00	7.64	0.54	0.62		
15	56.67	50.00	75.00	7.26	0.52	0.60		
20	**61.67**	50.00	66.67	5.53	0.53	0.61		
25	55.00	41.67	75.00	8.50	0.53	0.61		
30	56.67	50.00	66.67	6.24	0.53	0.60		
35	54.17	41.67	66.67	7.68	0.52	0.60		
40	57.50	41.67	66.67	6.92	0.50	0.59		
white-clover								
5	71.43	64.29	78.57	5.53	0.62	0.55	75.00	0.60	0.55
10	69.29	50.00	78.57	9.61	0.62	0.57		
15	74.29	64.29	85.71	6.55	0.62	0.57		

Continued on next page

Table 4.2 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC		
20	**76.43**	64.29	92.86	8.48	0.62	0.56			
25	72.14	64.29	85.71	6.74	0.61	0.56			
30	69.29	50.00	78.57	7.86	0.61	0.57			
35	75.00	71.43	85.71	5.76	0.61	0.56			
40	**75.71**	71.43	85.71	4.74	0.60	0.55			
eucalyptus									
5	**60.74**	50.92	71.17	7.72	0.53	0.71	47.67	0.51	0.70
10	**54.42**	49.08	63.19	4.24	0.53	0.72			
15	**51.60**	48.47	56.44	2.46	0.52	0.71			
20	**51.66**	46.01	57.06	3.52	0.52	0.71			
25	**50.25**	46.01	54.60	2.29	0.51	0.70			
30	**51.41**	47.24	56.44	2.87	0.51	0.70			
35	**50.12**	47.85	52.15	1.45	0.51	0.70			
40	**50.00**	47.24	52.76	1.72	0.51	0.70			
soybean									
5	**84.97**	80.79	89.40	3.09	0.79	0.90	82.45	0.75	0.89
10	**83.25**	77.48	88.08	3.48	0.76	0.89			
15	**82.65**	79.47	88.74	2.85	0.76	0.89			
20	81.92	77.48	85.43	2.79	0.75	0.89			
25	**82.52**	77.48	85.43	2.28	0.76	0.89			
30	**82.72**	80.13	84.11	1.05	0.75	0.89			
35	82.12	80.79	83.44	0.84	0.75	0.89			
40	**82.58**	80.13	86.75	1.80	0.75	0.89			
diabetes									
5	80.08	73.56	85.82	4.92	0.71	0.68	81.26	0.71	0.67
10	**81.38**	77.01	84.67	2.17	0.71	0.68			
15	79.96	75.86	83.52	2.31	0.71	0.68			
20	80.23	78.16	83.14	1.42	0.71	0.67			
25	79.69	76.63	82.76	1.80	0.71	0.67			
30	80.34	78.16	82.38	1.52	0.71	0.67			
35	**81.34**	79.31	82.76	1.00	0.71	0.67			
40	80.88	79.31	82.76	0.99	0.71	0.67			
glass									
5	**74.93**	65.75	84.93	6.51	0.66	0.77	68.90	0.63	0.75
10	**71.23**	65.75	79.45	4.58	0.65	0.76			
15	**71.23**	67.12	76.71	2.74	0.65	0.76			
20	**70.68**	65.75	80.82	3.98	0.65	0.76			
25	68.90	64.38	75.34	3.19	0.64	0.76			
30	**70.41**	65.75	72.60	2.14	0.64	0.76			
35	68.77	64.38	71.23	2.36	0.64	0.76			
40	**69.04**	65.75	71.23	1.64	0.64	0.76			
car									
5	**63.62**	61.90	66.33	1.33	0.56	0.79	62.41	0.56	0.78
10	**62.98**	61.73	64.46	0.79	0.55	0.78			
15	**62.60**	60.88	64.97	1.11	0.56	0.78			
20	**62.89**	61.90	63.78	0.57	0.56	0.79			
25	**62.74**	61.56	63.95	0.67	0.56	0.78			
30	**62.47**	61.73	63.27	0.54	0.56	0.78			
35	**62.53**	60.54	63.95	0.97	0.56	0.78			
40	**62.55**	61.56	63.61	0.60	0.56	0.78			
sonar									
5	**12.68**	4.23	36.62	10.82	0.26	0.00	0.70	0.30	0.00
10	**6.62**	1.41	12.68	2.96	0.26	0.00			
15	**4.79**	1.41	7.04	2.11	0.27	0.00			
20	**4.23**	0.00	12.68	3.27	0.28	0.00			
25	**3.10**	1.41	5.63	1.64	0.29	0.00			
30	**3.94**	0.00	7.04	2.25	0.29	0.00			
35	**3.94**	1.41	7.04	1.97	0.29	0.00			
40	**2.39**	0.00	5.63	1.89	0.29	0.00			

Continued on next page

Table 4.2 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC		
vehicle									
5	**73.72**	67.71	83.68	4.47	0.67	0.78	69.69	0.65	0.77
10	**73.02**	68.40	77.78	3.15	0.66	0.77			
15	**72.74**	68.40	76.74	2.45	0.66	0.77			
20	**70.90**	68.40	75.69	2.21	0.66	0.77			
25	**70.28**	68.75	73.96	1.47	0.65	0.77			
30	**69.90**	68.06	71.53	1.10	0.65	0.76			
35	**70.38**	69.10	72.92	1.07	0.66	0.77			
40	**70.45**	67.71	72.92	1.34	0.65	0.77			
vote									
5	94.73	92.57	95.95	0.95	0.92	0.95	95.95	0.91	0.94
10	95.47	94.59	96.62	0.68	0.92	0.94			
15	95.41	94.59	95.95	0.51	0.91	0.94			
20	95.61	94.59	95.95	0.45	0.91	0.94			
25	95.68	94.59	95.95	0.45	0.92	0.94			
30	95.68	95.27	95.95	0.33	0.91	0.94			
35	95.61	94.59	95.95	0.45	0.91	0.94			
40	95.74	95.27	95.95	0.31	0.91	0.94			

As demonstrated in Table 4.2, since for each dataset, 8 *CLUB-DRF*s of multiple of 5 sizes in the range 5 to 40 have been generated, Table 4.3 shows some performance statistics for this variation including *RF* (the parent) performance, the number of winning *CLUB-DRF*s, and accuracy of best and worst *CLUB-DRF* performers. As demonstrated in the 3rd column of the table, it is easy to observe that *CLUB-DRF* outperformed *RF* on 14/15 datasets. For the *vote* dataset where *RF* outperformed *CLUB-DRF*, the difference was by only 0.21% when compared with the best performer *CLUB-DRF*.

TABLE 4.3: Performance Statistics of BestRepOnTraining CLUB-DRF

Dataset	RF Accuracy	# of Winning CLUB-DRFs	Best Accuracy	Worst Accuracy
breast-cancer	69.79%	5	72.37%	69.07%
audit	96.56%	3	96.70%	96.15%
credit	77.18%	4	78.15%	76.41%
pasture	42.50%	8	60.00%	44.17%
squash-unstored	68.33%	6	75.00%	66.67%
squash-stored	57.50%	3, 1 tie	63.33%	54.17%
white-clover	75.00%	2, 1 tie	76.43%	69.29%
eucalyptus	47.67%	8	60.74%	50.00%
soybean	82.45%	6	84.97%	81.92%
diabetes	81.26%	2	81.38%	79.69%
glass	68.90%	6, 1 tie	74.93%	68.77%
car	62.41%	8	63.62%	62.47%
sonar	0.70%	8	12.68%	2.39%
vehicle	69.69%	8	73.72%	70.28%
vote	95.95%	0	95.74%	94.73%

4.5.1.2 Best Representative on OOB CLUB-DRF Results

Table 4.4 compares the performance of *CLUB-DRF* and *RF*. As demonstrated in this table, *CLUB-DRF* outperformed *RF* on 13 datasets. Of the 13 datasets, interestingly enough, *CLUB-DRF*, regardless of its size, outperformed *RF* on 6 of the datasets, namely, *breast-cancer*, *pasture*, *eucalyptus*, *glass*, *sonar*, and *vehicle*. Table 4.5 shows performance statistics for this variation. As shown in the table, for the *audit* and *vote* datasets where *RF* outperformed *CLUB-DRF*, the outperformance was only by 0.06% (in the case of *audit*), and only by 0.14% (in the case of *vote*). After applying the paired t-test for this variation, a p-value of 0.001087 was obtained indicating that the result is statistically significant.

TABLE 4.4: Performance Metrics of BestRepOnOOB CLUB-DRF & RF

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
breast-cancer							
5	**71.24**	65.98	77.32	3.34	0.64	0.59	69.18 0.63 0.58
10	**72.16**	69.07	76.29	2.53	0.64	0.59	
15	**70.21**	67.01	75.26	2.28	0.64	0.59	
20	**72.58**	67.01	75.26	2.45	0.64	0.59	
25	**69.59**	67.01	73.20	1.61	0.64	0.59	
30	**71.65**	69.07	74.23	1.68	0.64	0.59	
35	**69.48**	65.98	72.16	1.68	0.64	0.59	
40	**71.44**	67.01	73.20	2.01	0.64	0.58	
audit							
5	95.86	93.22	97.05	1.19	0.92	0.88	96.53 0.92 0.88
10	96.18	94.99	96.76	0.49	0.91	0.87	
15	96.06	95.43	96.76	0.50	0.91	0.88	
20	96.47	95.58	97.05	0.41	0.91	0.87	
25	96.30	95.87	96.61	0.31	0.91	0.87	
30	96.42	96.17	96.61	0.15	0.91	0.87	
35	96.39	95.72	96.61	0.25	0.91	0.87	
40	96.42	96.17	96.76	0.18	0.91	0.87	
credit							
5	**79.68**	72.06	87.94	5.06	0.68	0.62	77.47 0.67 0.61
10	**78.74**	75.00	82.65	2.38	0.68	0.62	
15	**77.97**	75.59	80.29	1.84	0.68	0.62	
20	76.24	73.24	78.53	1.60	0.67	0.62	
25	77.35	75.00	80.29	1.53	0.67	0.61	
30	77.00	75.29	78.82	1.29	0.67	0.61	
35	76.94	75.29	78.53	1.02	0.67	0.61	
40	77.29	75.00	78.53	1.10	0.67	0.61	
pasture							
5	**43.33**	16.67	66.67	15.72	0.45	0.55	41.67 0.43 0.57
10	**55.83**	33.33	75.00	13.46	0.45	0.58	
15	**45.00**	16.67	66.67	15.90	0.43	0.56	
20	**49.17**	41.67	66.67	9.46	0.43	0.56	
25	**45.00**	16.67	58.33	13.02	0.44	0.55	
30	**45.83**	33.33	66.67	10.03	0.42	0.55	
35	**42.50**	25.00	58.33	10.17	0.44	0.57	
40	**45.00**	25.00	66.67	11.90	0.42	0.55	
squash-unstored							
5	**76.67**	50.00	100.00	12.25	0.63	0.72	72.50 0.58 0.70
10	**80.00**	75.00	83.33	4.08	0.64	0.73	
15	70.83	58.33	83.33	10.03	0.62	0.72	
20	**79.17**	66.67	91.67	7.68	0.60	0.72	

Continued on next page

Table 4.4 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
25	**75.83**	66.67	83.33	4.49	0.61	0.72	
30	72.50	50.00	83.33	12.94	0.59	0.70	
35	**75.00**	58.33	91.67	9.13	0.59	0.70	
40	70.00	50.00	83.33	12.47	0.58	0.70	
squash-stored							
5	**63.33**	50.00	91.67	13.02	0.57	0.64	55.83 0.50 0.58
10	**65.00**	50.00	83.33	9.72	0.55	0.62	
15	54.17	41.67	66.67	8.54	0.51	0.59	
20	**65.00**	58.33	83.33	7.26	0.54	0.62	
25	53.33	41.67	58.33	6.67	0.52	0.61	
30	55.00	50.00	66.67	6.67	0.52	0.60	
35	52.50	41.67	66.67	6.51	0.50	0.59	
40	**56.67**	50.00	66.67	6.24	0.52	0.60	
white-clover							
5	75.00	57.14	92.86	10.23	0.62	0.58	77.14 0.60 0.55
10	75.71	50.00	85.71	11.16	0.62	0.57	
15	67.86	57.14	85.71	10.23	0.61	0.55	
20	72.14	57.14	78.57	7.46	0.60	0.55	
25	72.14	57.14	85.71	10.81	0.62	0.57	
30	**79.29**	57.14	92.86	9.82	0.62	0.57	
35	73.57	64.29	85.71	6.43	0.62	0.57	
40	73.57	64.29	78.57	4.57	0.62	0.56	
eucalyptus							
5	**55.46**	45.40	61.96	5.78	0.52	0.71	48.40 0.51 0.70
10	**56.20**	46.63	63.80	5.74	0.52	0.71	
15	**52.52**	47.24	59.51	3.45	0.52	0.71	
20	**50.74**	45.40	52.15	2.05	0.51	0.70	
25	**52.15**	47.85	55.21	1.96	0.52	0.71	
30	**49.26**	45.40	53.99	2.65	0.51	0.70	
35	**51.23**	47.85	55.21	2.00	0.51	0.70	
40	**50.18**	45.40	53.99	2.66	0.51	0.70	
soybean							
5	**83.97**	78.15	92.72	4.10	0.76	0.89	82.32 0.75 0.89
10	**82.52**	80.13	84.77	1.57	0.76	0.89	
15	**84.11**	80.79	88.08	2.05	0.76	0.89	
20	82.05	78.81	84.11	1.86	0.75	0.89	
25	82.05	79.47	84.77	2.00	0.75	0.89	
30	81.32	76.16	84.77	2.33	0.75	0.88	
35	82.05	79.47	84.11	1.55	0.75	0.88	
40	81.99	78.81	84.11	1.44	0.75	0.89	
diabetes							
5	80.80	74.71	84.29	3.53	0.72	0.68	81.26 0.71 0.67
10	81.15	74.71	84.29	3.56	0.71	0.68	
15	79.85	77.39	83.14	1.96	0.71	0.67	
20	**81.42**	79.31	83.14	1.24	0.71	0.67	
25	80.96	78.93	82.76	1.31	0.71	0.67	
30	80.88	78.54	82.76	1.14	0.71	0.67	
35	79.81	77.39	81.99	1.40	0.71	0.67	
40	**81.38**	80.08	83.14	0.94	0.71	0.67	
glass							
5	**76.16**	64.38	84.93	5.62	0.65	0.76	67.53 0.63 0.75
10	**68.36**	61.64	75.34	4.52	0.64	0.76	
15	**71.37**	65.75	78.08	3.65	0.64	0.76	
20	**70.82**	67.12	73.97	2.30	0.64	0.76	
25	**68.22**	64.38	71.23	2.01	0.63	0.75	
30	**70.41**	67.12	76.71	2.75	0.64	0.76	
35	**69.32**	67.12	72.60	1.86	0.63	0.76	
40	**68.77**	63.01	71.23	2.51	0.64	0.76	
car							

Continued on next page

Table 4.4 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
5	**64.17**	62.41	67.52	1.33	0.56	0.78	62.26 0.56 0.78
10	**63.01**	61.56	64.29	0.75	0.56	0.78	
15	**62.36**	60.71	64.29	1.12	0.56	0.78	
20	**62.35**	61.22	63.78	0.82	0.56	0.78	
25	**62.69**	60.88	63.95	0.85	0.56	0.78	
30	62.18	61.05	63.10	0.82	0.56	0.78	
35	61.96	60.88	63.61	0.72	0.56	0.78	
40	61.99	61.05	62.59	0.54	0.55	0.78	
sonar							
5	**12.25**	7.04	18.31	3.34	0.26	0.00	0.14 0.29 0.00
10	**9.15**	0.00	16.90	5.20	0.28	0.00	
15	**6.34**	0.00	14.08	4.47	0.29	0.00	
20	**3.38**	0.00	8.45	2.76	0.29	0.00	
25	**3.10**	0.00	7.04	2.42	0.28	0.00	
30	**1.83**	0.00	4.23	1.27	0.28	0.00	
35	**3.38**	0.00	4.23	1.29	0.28	0.00	
40	**3.38**	0.00	9.86	2.69	0.28	0.00	
vehicle							
5	**72.01**	67.36	81.25	4.76	0.66	0.77	69.90 0.65 0.77
10	**72.43**	64.58	81.25	5.22	0.66	0.77	
15	**71.25**	67.71	76.04	2.76	0.65	0.77	
20	**71.49**	69.44	73.96	1.28	0.65	0.77	
25	**70.94**	69.44	72.92	1.42	0.65	0.77	
30	**71.25**	68.75	75.00	2.03	0.66	0.77	
35	**70.10**	67.71	71.53	1.12	0.65	0.77	
40	**70.45**	68.75	72.57	1.07	0.65	0.77	
vote							
5	94.53	91.89	97.30	1.37	0.91	0.94	95.95 0.91 0.94
10	95.74	95.27	96.62	0.43	0.91	0.94	
15	95.27	93.92	96.62	0.80	0.91	0.94	
20	95.61	94.59	96.62	0.54	0.91	0.94	
25	95.74	94.59	95.95	0.43	0.91	0.94	
30	95.81	95.27	96.62	0.41	0.91	0.94	
35	95.68	95.27	95.95	0.33	0.91	0.94	
40	95.81	94.59	96.62	0.51	0.91	0.94	

4.5.1.3 Random Representative CLUB-DRF Results

Table 4.6 compares the performance of *CLUB-DRF* and *RF*. As expected and as shown in the table, *CLUB-DRF* did not perform as well as the previous 2 variations of *CLUB-DRF*. This can be attributed to the fact that, unlike the previous 2 variations where accuracy was the decisive factor in selecting a representative from each cluster, in this variation, a representative tree was randomly picked without assessing its performance accuracy. This emphasizes the importance of using some sort of accuracy measure when selecting a representative from each cluster. Table 4.7 shows performance statistics for this variation. Despite the fact that *RF* outperformed *CLUB-DRF*, the interesting observation is that, for the datasets where *RF* outperformed *CLUB-DRF*, the outperformance between *RF* and the best performer *CLUB-DRF* for such datasets was less

Table 4.5: Performance Statistics of BestRepOnOOB CLUB-DRF

Dataset	RF	# of Winning CLUB-DRFs	Best Performer	Worst Performer
breast-cancer	69.18%	8	72.58%	69.48%
audit	96.53%	0	96.47%	95.86%
credit	77.47%	3	79.68%	76.24%
pasture	41.67%	8	55.83%	42.50%
squash-unstored	72.50%	5, 1 tie	80.00%	70.00%
squash-stored	55.83%	4	65.00%	52.50%
white-clover	77.14%	1	79.29%	67.86%
eucalyptus	48.40%	8	56.20%	49.26%
soybean	82.32%	3	84.11%	81.32%
diabetes	81.26%	2	81.42%	79.81%
glass	67.53%	8	76.16%	68.22%
car	62.26%	5	64.17%	61.96%
sonar	0.14%	8	12.25%	1.83%
vehicle	69.90%	8	72.43%	70.10%
vote	95.95%	0	95.81%	94.53%

than 5%. Due to the unsatisfactory performance of this variation, the p-value for this variation using the paired t-test was 0.9739 which is statistically insignificant.

Table 4.6: RandomRep CLUB-DRF & RF

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure	AUC
breast-cancer								
5	67.42	53.61	75.26	6.07	0.64	0.56	71.75	0.65	0.58
10	66.70	60.82	74.23	4.42	0.64	0.57		
15	70.10	65.98	73.20	2.73	0.65	0.58		
20	70.41	67.01	73.20	2.16	0.65	0.58		
25	70.93	67.01	77.32	3.12	0.65	0.58		
30	68.97	65.98	72.16	2.18	0.65	0.58		
35	**72.89**	65.98	79.38	3.54	0.65	0.58		
40	71.75	65.98	74.23	2.07	0.65	0.58		
audit								
5	94.10	92.21	95.44	1.07	0.89	0.86	96.22	0.90	0.88
10	94.22	93.09	96.03	0.76	0.89	0.86		
15	95.40	94.71	96.18	0.51	0.89	0.86		
20	95.41	94.85	96.03	0.38	0.89	0.86		
25	95.79	94.71	96.32	0.43	0.89	0.86		
30	95.62	95.15	96.03	0.26	0.89	0.86		
35	95.75	95.29	96.47	0.37	0.89	0.86		
40	95.51	95.00	96.18	0.44	0.89	0.86		
credit								
5	61.62	47.94	68.82	7.65	0.65	0.59	76.00	0.66	0.60
10	71.50	67.65	74.71	1.94	0.66	0.60		
15	71.74	67.06	75.88	2.61	0.67	0.60		
20	72.44	70.59	74.12	1.09	0.66	0.60		
25	74.50	71.47	77.06	1.96	0.66	0.60		
30	75.09	73.53	76.76	1.01	0.66	0.60		
35	74.74	72.65	77.06	1.33	0.66	0.60		
40	74.68	72.65	76.18	1.17	0.66	0.60		
pasture								

Continued on next page

Table 4.6 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC		
5	**43.33**	8.33	75.00	20.00	0.42	0.55	38.33	0.42	0.56
10	24.17	0.00	41.67	13.67	0.40	0.53			
15	27.50	8.33	41.67	10.57	0.37	0.54			
20	31.67	16.67	50.00	8.16	0.41	0.56			
25	34.17	8.33	58.33	16.01	0.41	0.56			
30	30.00	8.33	50.00	12.47	0.40	0.54			
35	**40.00**	16.67	58.33	11.67	0.41	0.56			
40	30.00	16.67	50.00	10.00	0.41	0.54			
squash-unstored									
5	58.33	50.00	72.22	8.33	0.51	0.63	60.56	0.51	0.64
10	50.56	38.89	66.67	8.77	0.49	0.62			
15	56.67	38.89	72.22	9.23	0.50	0.63			
20	48.89	38.89	61.11	5.44	0.50	0.64			
25	53.89	44.44	66.67	7.05	0.48	0.62			
30	55.56	44.44	72.22	8.24	0.50	0.63			
35	50.00	38.89	61.11	6.57	0.49	0.62			
40	51.67	44.44	55.56	3.56	0.50	0.63			
squash-stored									
5	42.22	22.22	66.67	13.43	0.48	0.54	55.56	0.50	0.56
10	47.78	27.78	61.11	8.68	0.51	0.57			
15	44.44	33.33	61.11	8.24	0.46	0.54			
20	49.44	38.89	61.11	7.22	0.50	0.56			
25	52.78	33.33	66.67	9.38	0.49	0.54			
30	50.56	33.33	55.56	7.22	0.50	0.56			
35	48.33	38.89	55.56	5.00	0.49	0.55			
40	48.33	38.89	55.56	6.11	0.48	0.54			
white-clover									
5	57.62	42.86	71.43	9.15	0.54	0.63	63.33	0.51	0.61
10	50.00	33.33	61.90	8.84	0.50	0.60			
15	59.52	52.38	66.67	4.39	0.51	0.61			
20	57.62	42.86	66.67	7.51	0.51	0.60			
25	60.48	52.38	66.67	5.24	0.51	0.60			
30	61.43	52.38	71.43	5.81	0.51	0.61			
35	62.86	52.38	71.43	5.13	0.51	0.60			
40	60.48	52.38	66.67	4.29	0.50	0.60			
eucalyptus									
5	17.32	4.40	28.80	8.00	0.19	0.56	19.60	0.21	0.57
10	16.40	5.60	24.00	5.87	0.21	0.57			
15	**20.64**	12.40	28.80	5.04	0.22	0.57			
20	18.84	11.20	24.00	4.66	0.21	0.57			
25	18.12	11.60	25.20	4.21	0.21	0.57			
30	18.56	11.60	23.60	3.68	0.22	0.57			
35	17.24	10.00	22.80	3.72	0.21	0.57			
40	19.04	16.40	23.60	2.24	0.21	0.57			
soybean									
5	68.02	57.76	76.29	6.93	0.71	0.84	77.76	0.73	0.85
10	70.99	65.09	76.72	4.33	0.71	0.84			
15	70.99	66.81	75.43	2.90	0.70	0.84			
20	73.06	68.53	82.76	4.57	0.71	0.84			
25	72.67	68.97	77.16	2.51	0.71	0.84			
30	72.84	66.38	82.33	3.90	0.71	0.84			
35	73.15	68.53	78.88	3.21	0.72	0.85			
40	72.63	69.83	75.86	1.88	0.71	0.84			
diabetes									
5	67.01	57.47	72.41	5.08	0.68	0.66	73.72	0.68	0.66
10	71.57	65.90	75.10	3.13	0.68	0.66			
15	69.96	67.43	72.03	1.52	0.68	0.66			
20	72.07	68.20	74.33	1.86	0.68	0.66			
25	72.68	70.11	75.10	1.45	0.68	0.66			

Continued on next page

Table 4.6 – continued from previous page

CLUB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure	AUC
30	72.61	68.58	76.25	2.12	0.68	0.66		
35	73.64	70.88	75.48	1.38	0.68	0.66		
40	71.95	69.73	75.10	1.46	0.68	0.66		
glass								
5	56.99	50.68	64.38	4.95	0.58	0.72	63.56	0.58	0.71
10	58.63	52.05	64.38	3.51	0.57	0.71		
15	60.96	58.90	64.38	1.65	0.57	0.71		
20	60.82	57.53	63.01	1.75	0.57	0.71		
25	60.96	56.16	65.75	2.76	0.57	0.71		
30	61.37	58.90	67.12	2.44	0.57	0.71		
35	61.37	57.53	65.75	2.58	0.57	0.71		
40	61.37	60.27	64.38	1.34	0.57	0.71		
car								
5	61.60	60.20	64.12	1.36	0.56	0.78	62.36	0.56	0.78
10	61.75	61.05	62.59	0.50	0.56	0.78		
15	61.97	60.88	63.27	0.76	0.55	0.78		
20	61.85	60.88	62.76	0.64	0.56	0.78		
25	61.67	60.71	62.76	0.59	0.55	0.78		
30	61.55	60.54	62.41	0.63	0.55	0.78		
35	61.70	60.54	62.24	0.54	0.56	0.78		
40	61.84	60.71	62.93	0.72	0.56	0.78		
sonar								
5	**8.45**	0.00	18.31	6.07	0.26	0.00	0.28	0.29	0.00
10	**6.20**	1.41	14.08	3.79	0.30	0.00		
15	**5.92**	2.82	14.08	3.26	0.32	0.00		
20	**3.94**	0.00	8.45	3.14	0.29	0.00		
25	**2.82**	0.00	8.45	2.36	0.30	0.00		
30	**3.80**	0.00	7.04	2.44	0.30	0.00		
35	**3.66**	1.41	7.04	1.80	0.30	0.00		
40	**2.54**	0.00	5.63	2.07	0.30	0.00		
vehicle								
5	65.10	52.43	73.26	6.12	0.70	0.80	73.89	0.69	0.80
10	69.58	64.24	75.69	3.78	0.69	0.80		
15	70.28	67.71	73.96	1.95	0.69	0.79		
20	71.70	68.40	75.00	2.01	0.69	0.79		
25	72.85	70.49	75.00	1.46	0.69	0.80		
30	72.47	68.75	74.31	1.69	0.68	0.79		
35	73.02	71.53	75.00	1.19	0.69	0.80		
40	73.16	71.88	74.65	0.98	0.69	0.80		
vote								
5	96.62	95.27	98.65	1.00	0.95	0.97	97.97	0.95	0.97
10	96.69	95.27	97.97	0.82	0.95	0.97		
15	97.57	97.30	97.97	0.33	0.95	0.97		
20	97.50	97.30	97.97	0.31	0.95	0.97		
25	97.70	96.62	97.97	0.54	0.95	0.97		
30	97.84	97.30	97.97	0.27	0.95	0.97		
35	97.70	97.30	97.97	0.33	0.95	0.97		
40	97.43	96.62	97.97	0.41	0.95	0.97		

TABLE 4.7: Performance Statistics of RandomRep CLUB-DRF

Dataset	RF	# of Winning CLUB-DRFs	Best Performer	Worst Performer
breast-cancer	71.75%	1, 1 tie	72.89%	66.70%
audit	96.22%	0	95.79%	94.10%
credit	76.00%	0	75.09%	61.62%
pasture	38.33%	2	43.33%	24.17%
squash-unstored	60.56%	0	58.33%	48.89%
squash-stored	55.56%	0	52.78%	42.22%
white-clover	63.33%	0	62.86%	50.00%
eucalyptus	19.60%	1	20.64%	16.40%
soybean	77.76%	0	73.15%	68.02%
diabetes	73.72%	0	73.64%	67.01%
glass	63.56%	0	61.37%	56.99%
car	62.36%	0	61.97%	61.55%
sonar	0.28%	8	8.45%	2.54%
vehicle	73.89%	0	73.16%	65.10%
vote	97.97%	0	97.84%	96.62%

4.5.2 Analysis of the Results

4.5.2.1 Best Representative on Training CLUB-DRF Analysis

Using this variation and based on the information given in Table 4.2, for each dataset, Figure 4.2 shows the number of *CLUB-DRF* pruned ensembles that have outperformed *RF*. It is easy to see in the figure that *CLUB-DRF* has outperformed *RF* on 14/15 datasets, of which, *CLUB-DRF* has completely outperformed *RF* (which happens when the number of outperformers is 8) on 5 datasets: *pasture*, *eucalyptus*, *car*, *sonar*, and *vehicle*. More precisely, a pruned RF using *CLUB-DRF*, regardless of its size, outperformed *RF* on these datasets.

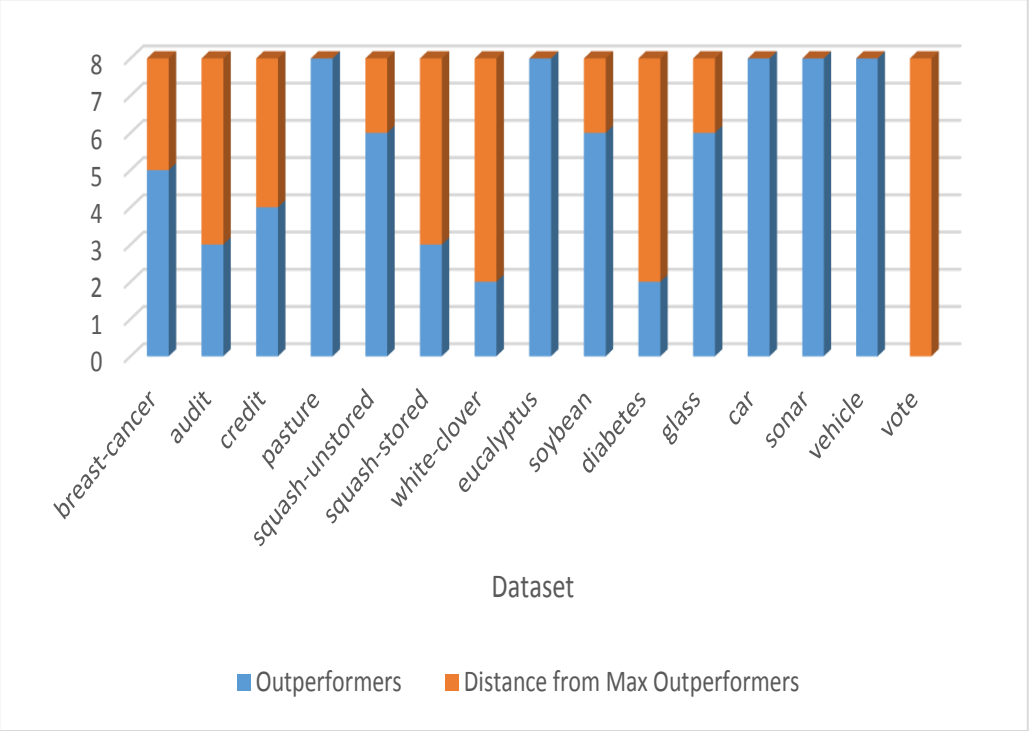


FIGURE 4.2: Number of BestRepOnTraining CLUB-DRF Outperformers per Dataset

4.5.2.2 Best Representative on OOB CLUB-DRF Analysis

From the information given in Table 4.4, Figure 4.3 shows similar results for this variation of *CLUB-DRF*. As shown in the figure, this variation outperformed *RF* on 13/15 datasets. Of the 13 datasets, *CLUB-DRF* completely outperformed *RF* on 6 of them, namely, *breast-cancer*, *pasture*, *eucalyptus*, *glass*, *sonar*, and *vehicle*.

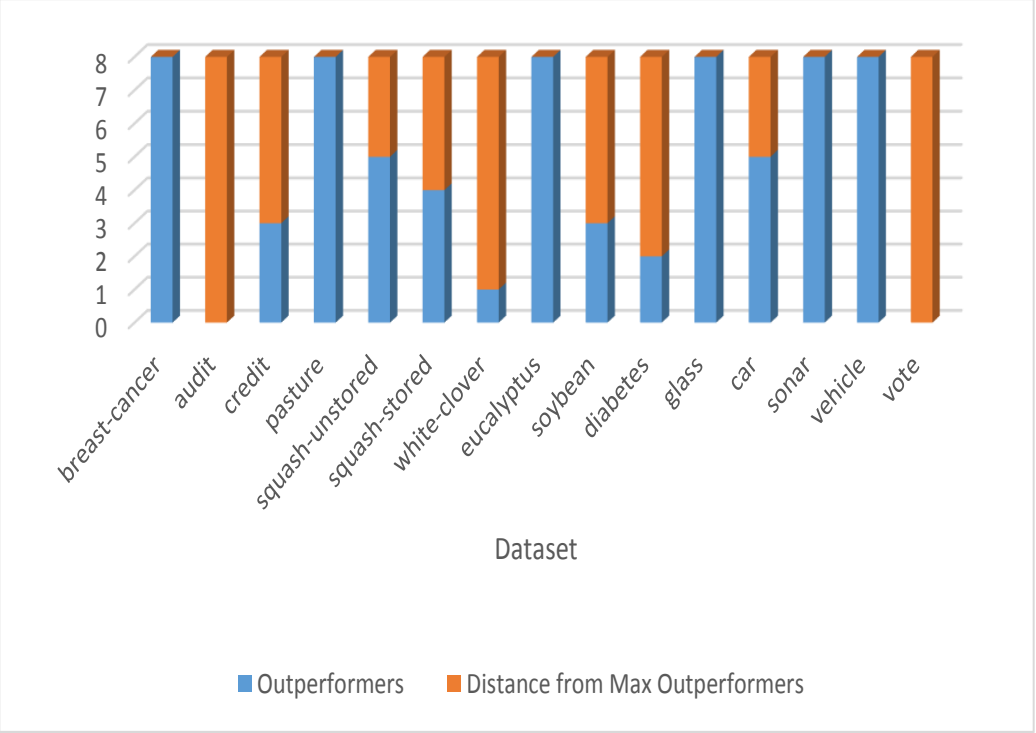


FIGURE 4.3: Number of BestRepOnOOB CLUB-DRF Outperformers per Dataset

4.5.2.3 Random Representative CLUB-DRF Analysis

As previously stated, this variation did not perform well. This is reflected in Figure 4.4 which is based on the information presented in Table 4.6. The figure shows that *CLUB-DRF* outperformed *RF* on only 4 datasets, of which, *CLUB-DRF* completely outperformed *RF* on 1 dataset. As previously explained, this is due to the lack of an accuracy measure used in this variation.

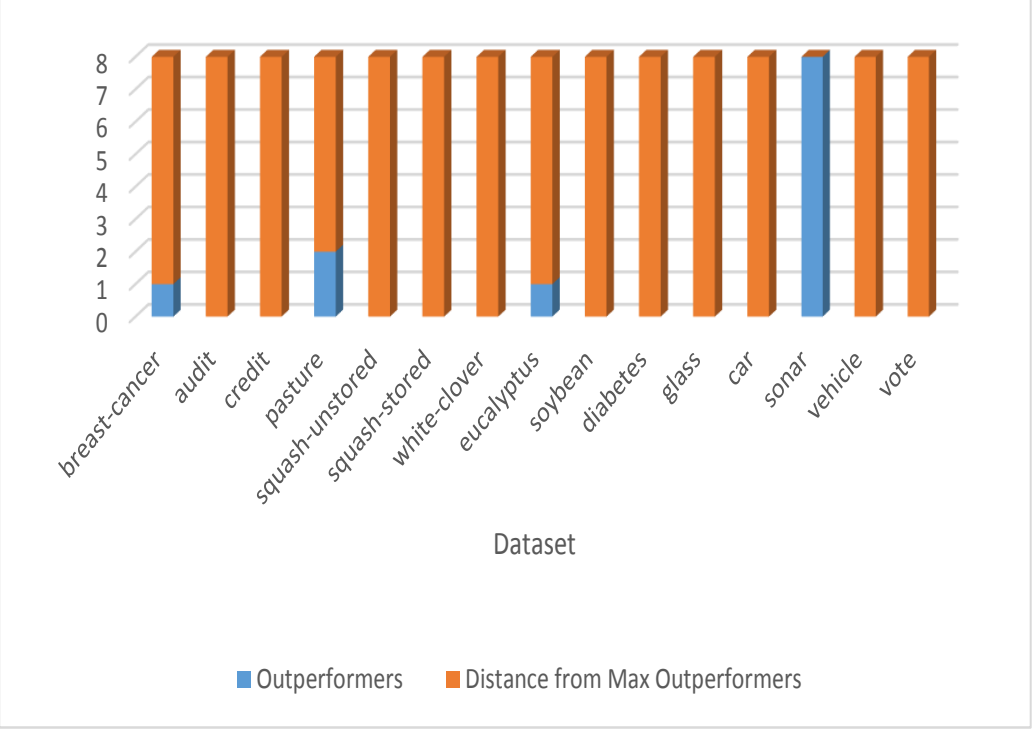


FIGURE 4.4: Number of RandomRep CLUB-DRF Outperformers per Dataset

4.5.3 Outperformance Range

For each dataset, Tables 4.8, 4.9, and 4.10 below depict the outperformance range of each *CLUB-DRF*, produced by each of the 3 variations respectively, over *RF*. This range has the format *min-max* where *min* is calculated as the difference between the worst performer (that outperformed *RF*) and *RF*. Likewise, *max* is calculated as the difference between the best performer (that outperformed *RF*) and *RF*. When *RF* has outperformed all *CLUB-DRF*s (as it is the case for example with the dataset *vote* in Table 4.2), we list the difference of performance between *RF* and the best performer *CLUB-DRF*, and show it as a negative number (as shown in the last entry in Table 4.8) to indicate that *RF* was superior. These tables demonstrate a maximum outperformance range of 17.5%, 14.16%, and 8.17% achieved by the *BestRepOnTraining*, *BestRepOnOOB*, and *RandomRep* variations respectively.

4.5.4 Pruning Level

By applying the above proposed clustering technique, we managed to achieve two objectives. First, *CLUB-DRF* ensembles with diverse trees were produced. Second and more importantly, we manged to significantly reduce the size of *RF*. The resulted pruned *CLUB-DRF* ensembles mostly outperformed the original *RF* ensemble, as depicted in Figures 4.2 and 4.3. In ensemble

TABLE 4.8: Outperformance Range of BestRepOnTraining CLUB-DRF over RF

Dataset	Range
breast-cancer	0.42% - 2.58%
audit	0.03% - 0.14%
credit	0.11% - 0.97%
pasture	1.67% - 17.5%
squash-unstored	1.67% - 6.67%
squash-stored	4.17% - 5.83%
white clover	0.71% - 1.43%
eucalyptus	2.33% - 13.07%
soybean	0.07% - 2.52%
diabetes	0.08% - 0.12%
glass	0.14% - 6.03%
car	0.06% - 1.21%
vehicle	0.21% - 4.03%
vote	-0.21%

TABLE 4.9: Outperformance Range of BestRepOnOOB CLUB-DRF over RF

Dataset	Range
breast-cancer	0.30% - 3.40%
audit	-0.06%
credit	0.50% - 2.21%
pasture	0.83% - 14.16%
squash-unstored	2.50% - 7.50%
squash-stored	0.84% - 9.17%
white clover	2.15% (only one outperformer found)
eucalyptus	0.86% - 7.80%
soybean	0.20% - 1.79%
diabetes	0.12% - 0.16%
glass	0.69% - 8.63%
car	0.09% - 1.91%
vehicle	0.20% - 2.53%
vote	-0.14%

pruning, a pruning level refers to the reduction ratio between the original ensemble and the pruned one. For example, if the size of the original ensemble is 500 trees and the pruned one is of size 50, then $100\% - \frac{50}{500} \times 100\% = 90\%$ is the pruning level that was achieved in the pruned ensemble. This means that the pruned ensemble is 90% smaller than the original one.

4.5.4.1 Best Representative on Training CLUB-DRF Pruning Level

Using the information in Table 4.2, Table 4.11 was compiled to show the pruning levels for *BestRepOnTraining CLUB-DRF*. The first column in this table shows the maximum possible pruning level for a *CLUB-DRF* that has outperformed *RF*, and the second column shows the

TABLE 4.10: Outperformance Range of RandomRep CLUB-DRF over RF

Dataset	Range
breast-cancer	1.14% (only one outperformer found)
audit	-0.43%
credit	-0.91%
pasture	1.67% - 5.00%
squash-unstored	-2.23%
squash-stored	-2.78%
white clover	-0.47%
eucalyptus	1.04% (only one outperformer found)
soybean	-4.61%
diabetes	-0.08%
glass	-2.19%
car	-0.39%
vehicle	-0.73%
vote	-0.13%

pruning level of the best performer *CLUB-DRF*. We can see that with extremely healthy pruning levels ranging from 96% to 99%, *CLUB-DRF* outperformed *RF*. The interesting observation in this table is that, in both columns, the mode is 99%. This mode refers to the highest unprecedented pruning level that was achieved by *CLUB-DRF* which corresponds to 5 trees only.

TABLE 4.11: BestRepOnTraining CLUB-DRF: Maximum Pruning Level with Best Possible Performance

Dataset	Maximum Pruning Level	Best Performer Pruning Level
breast-cancer	98%	98%
audit	98%	98%
credit	99%	99%
pasture	99%	98%
squash-unstored	99%	99%
squash-stored	99%	98%
white clover	96%	96%
eucalyptus	99%	99%
soybean	99%	99%
diabetes	98%	98%
glass	99%	99%
car	99%	99%
vehicle	99%	99%

4.5.4.2 Best Representative on OOB CLUB-DRF Pruning Level

Similarly, using the information in Table 4.4, pruning levels for *BestRepOnOOB CLUB-DRF* are shown in Table 4.12. We can see that in this variation, at extremely healthy pruning levels that

range from 94% to 99%, *CLUB-DRF* outperformed *RF*. In this table, note that 99% (highest pruning level) is the mode in the first columns and 98% (next highest pruning level) is the mode in the second column.

TABLE 4.12: BestRepOnOOB CLUB-DRF: Maximum Pruning Level with Best Possible Performance

Dataset	Maximum Pruning Level	Best Performer Pruning Level
breast-cancer	99%	96%
credit	99%	99%
pasture	99%	98%
squash-unstored	99%	98%
squash-stored	99%	98%
white clover	94%	94%
eucalyptus	99%	98%
soybean	99%	97%
diabetes	96%	96%
glass	99%	99%
car	99%	99%
vehicle	99%	98%

4.5.4.3 Random Representative CLUB-DRF Pruning Level

Using the information in Table 4.6, Table 4.13 depicts similar results for *CLUB-DRF* produced by this variation. As shown in the table, only 3 datasets outperformed *RF* with a pruning level ranging from 93% to 99%.

TABLE 4.13: RandomRep CLUB-DRF: Maximum Pruning Level with Best Possible Performance

Dataset	Maximum Pruning Level	Best Performer Pruning Level
breast-cancer	93%	93%
pasture	99%	99%
eucalyptus	97%	97%

4.5.5 Suitability for Real-Time Applications

Looking back at the high pruning levels achieved by the 3 variations of *CLUB-DRF*, it is obvious that *CLUB-DRF* ensembles are ideal candidates for real-time applications where fast classification is an important desideratum. They are also typical for machines and mobile devices with limited memory and processing power because of their small size in comparison with the size of the classical RF ensembles.

4.6 Time Complexity Analysis

In this section, we provide, using Big O notation, a time complexity analysis of our *CLUB-DRF* and compare it with that of *RF* (the parent). Let n refer to the number of trees in *RF*, α such that $0 < \alpha < 1$, refer to the pruning level achieved in the pruned *CLUB-DRF* ensemble, m refer to the number of trees in *CLUB-DRF*, and f refer to the number of features in the dataset. Since traversing the average number of nodes of all trees of in a *CLUB-DRF* is in the order of $O(f)$, the time complexity to classify an instance after having each tree in *RF* casts a vote for its predicted class label is in the order of $O(n \times O(f))$. Since $m = n - n \times \alpha$, the time complexity to classify an instance after having each tree in *CLUB-RF* casts a vote for its predicted class label is therefore given by $O(m \times O(f))$.

Because *CLUB-DRF* is known of its ability to perform extreme pruning as was described throughout this chapter, $m << n$. It is therefore clear that the time required by *CLUB-DRF* to classify an instance is going to be much smaller than that of *RF*. As an example, consider an *RF* of 500 trees (i.e., n is 500). As aforementioned, the time complexity of *RF* is therefore $O(500 \times O(f))$. If we assume a 99% pruning level achieved by *CLUB-DRF*, then $m = 500 - 500 \times 0.99 = 5$. Therefore, the time complexity of *CLUB-DRF* is $O(5 \times O(f))$ which is approximately 99% smaller than that of *RF*, hence, *CLUB-DRF* runs approximately 99% faster than *RF*. This example demonstrates that the pruning level is proportional to the classification speed which tends to increase when the pruning level increases. Since trees in RF ensembles are bushy trees with excessive number of nodes, and since the number of nodes traversed grows proportionally with the number of features, for high-dimensional datasets, classifying an instance is likely to take more time than low-dimensional datasets. Hence, for high-dimensional datasets, *CLUB-DRF* ensembles would be typical as they will yield faster classification time, making them cost effective not only in terms of accuracy, but also in terms of classification speed.

4.7 Summary

A new method called *CLUB-DRF* was developed for the extreme pruning of Random Forests. It was featured by clustering the trees based on their predictions on the training data. Each cluster in the ensemble can be represented by one or more trees. To achieve extreme pruning, only one representative from each cluster is selected. For the selection of a representative from each cluster, three variations were used. By comparing the performance of the three variations in the charts given in Figures 4.2, 4.3, and 4.4 respectively, we can see that *BestRepOnTraining* has performed the best. It outperformed *RF* on 14/15 of the datasets and completely outperformed *RF* on the 5/14 datasets. As depicted in the second column of Table 4.11, for the winning datasets where *CLUB-DRF* outperformed *RF*, the pruning level achieved by *CLUB-DRF* was

mostly 99% (highest pruning level ever reported to date), which corresponds to *CLUB-DRF* ensembles of size 5 trees only. The *BestRepOnOOB* variation produced less favorable results as it outperformed *RF* on 13/15 of the datasets and completely outperformed *RF* on the 6/13 datasets. As shown in the second column of Table 4.12, *BestRepOnOOB* also achieved 99% for the winning datasets.

The *RandomRep* variation did not perform well compared with the others as depicted by the chart in Figure 4.4. This is attributed to the fact that, unlike the first two variations where accuracy was the main criteria used for the selection of a representative from each cluster, accuracy in *RandomRep* was not used at all. The selection process in this variation simply involved picking a random representative from each cluster. As previously discussed, this emphasizes the importance of having an accuracy measure when selecting a representative from each cluster.

Thanks to the extreme pruning achieved by *CLUB-DRF*, it makes it usable not only for real-time applications where speed is a significant factor, but also for machines and devices with limited memory and processing power.

Chapter 5

Local Outlier Factor-Based Diverse Random Forest (LOFB-DRF)

5.1 Overview

Findings reported in this chapter have been published in Fawagreh et al. (2016). In this chapter, a new method is proposed to prune RF ensembles which is termed *LOFB-DRF*. Unlike *CLUB-DRF* in the previous chapter, *LOFB-DRF* applies the Local Outlier Factor instead of clustering. Like *CLUB-DRF*, extreme pruning levels reaching as high as 99% have been achieved while retaining or outperforming the accuracy of the original RF from which *LOFB-DRF* was derived.

5.2 Background

In this chapter, we will utilize another diversity technique to prune RF. As discussed in Section 2.3, empirically speaking, it has been proven that ensembles tend to perform better when the constituent models exhibit high level of diversity. Following the footsteps of the *CLUB-DRF* approach introduced in the previous chapter, the objective of this chapter is twofolds. First, it investigates how an unsupervised learning technique, namely, Local Outlier Factor (LOF), can be used to identify diverse trees in an RF. Second, trees with the highest weighted LOF scores are then used to produce a pruned RF ensemble, termed *LOFB-DRF*, that is much smaller in size than RF, and yet performs at least as good as RF, but mostly exhibits higher performance

in terms of accuracy (this is called ensemble pruning as discussed in Section 2.4 in Chapter 2). Experimental results on 15 real datasets prove the superiority of our proposed extension over the traditional RF. Like *CLUB-DRF*, unprecedented pruning levels reaching as high as 99% have been achieved at the time of boosting the predictive accuracy of the original ensemble. These notably high pruning levels make the technique a good candidate for real-time applications.

5.2.1 Motivation

The main aim is to minimize the number of trees built by RF algorithms which is typically in the range 100 to 500 trees Williams (2011). Our approach in this chapter aims at pruning RF ensembles by producing a subset of the original ones that are significantly smaller in size and yet, have accuracy performance that is at least as good as that of the original RF from which they were derived. In other words, we are aiming at finding the optimal or near-optimal number of trees that will be used to generate an accurate RF.

As discussed in the next section, the Local Outlier Factor (LOF) was used before as a diversity technique. Since the pruning methods developed in this thesis (*CLUB-DRF* in the previous chapter and *LOFB-DRF* in this chapter), produce pruned ensembles by selecting diverse trees, LOF is applied as a technique to aid in the pruning process by selecting the top k trees with the highest weighted LOF scores.

5.3 Local Outlier Factor

The LOF algorithm was developed by Breunig et al. Breunig et al. (2000) to measure the peculiarity/outlierness of an object. The higher the LOF value assigned to an object, the more isolated the object is with respect to its neighbors. It is considered a very powerful anomaly detection technique in machine learning and classification. Earlier work on outlier detection was investigated in Arning et al. (1996) Ruts and Rousseeuw (1996) Knox and Ng (1998) Knorr and Ng (1999), however, the work was limited by treating an outlier as a binary property to classify an object as an outlier or not, without assigning it a value to measure its outlierness as was done in Breunig et al. (2000).

The LOF can be used as a method to achieve diversity. It was one of 3 strategies used to obtain diversity when constructing an ensemble for the KDDCup 1999 dataset Erich and Zimek (2011). The use of normalized LOF values is what makes our approach similar to Erich and Zimek (2011). What makes it different, however, is that, in addition to LOF, we have used an accuracy measure to weigh each tree. Also, we have selected the top k (where k is multiple of 5 in the range 5 to 40) trees with the highest weighted LOF scores. In Erich and Zimek

(2011), $k = 20, 40, 80, 120, 160$. Furthermore, the ensembles used in the their experiments were not RF ensembles but rather, ensembles constructed using different strategies: feature bagging, LOF with different values of k on the full-dimensional dataset, and combination of four different methods (LOF, LDOF, kNN, and aggregate kNN).

TABLE 5.1: Height of Grade 5 Students

Student	Height (cm)
S1	130
S2	132
S3	138
S4	136
S5	131
S6	153
S7	131
S8	133
S9	129
S10	133
S11	110
S12	132
S13	129
S14	134
S15	135
S16	132
S17	135
S18	134
S19	133
S20	132
S21	130
S22	131
S23	134
S24	135
S25	135
S26	134
S27	136
S28	133
S29	133
S30	130

For the construction of advanced outlier detection ensembles using LOF variants and other algorithms, Schubert et al. (2012) developed methods for measuring similarity and diversity.

Formally, Breunig et al. Breunig et al. (2000) introduced the concept of reachability distance in order to calculate the LOF. If the distance of object A to the k nearest neighbor is denoted by $k\_distance(A)$, where the k nearest neighbors is denoted by $N_k(A)$, the following equation defines the reachability distance (rd):

$$rd_k(A, B) = max\{k\_distance(B), d(A, B)\} \tag{5.1}$$

where $d(A, B)$ is the distance between objects A and B. The local reachability density of object A is then defined by:

$$lrd(A) = \frac{\sum_{B \in N_k(A)} rd_k(A, B)}{|N_k(A)|} \tag{5.2}$$

Using the local reachability density of object A as defined in Equation 5.2, the LOF for object A is given by:

$$LOF_k(A) = \frac{\sum_{B \in N_k(A)} \frac{lrd(B)}{lrd(A)}}{|N_k(A)|} \tag{5.3}$$

The degree of the outlierness of object A is proportional with $LOF_k(A)$ and tends to increase when $LOF_k(A)$ increases.

As an example to illustrate calculating LOF for a set of objects, consider Table 5.1 above that lists the height (in cm) for grade 5 students. Using $k = 3$, we first get the *k_distance* neighborhood for each student. The furthest among the 3 nearest neighbors gives the *k_distance* for each student. To calculate the reachability distance for each student p with regard to each of its 3 nearest neighbors, for each neighbor o, we take the maximum between the distance $d(s, o)$ and *k_distance* of o. The reachability density for each student is the average between the reachability distances with regard to its nearest neighbors. The final step is to calculate the LOF for each student. All these calculations are shown in Table 5.2.

TABLE 5.2: LOF Calculation for Grade 5 Students

Student	3 Nearest Neighbors	$k\_distance$	Reach$_dist_3$(p)	$1/lrd_3$(p)	lrd_3(p)	LOF
S1	{S21,S22,S28}	3.35	{2.24,2.69,3.35}	2.76	0.36	1.77
S2	{S16,S19,S20}	1	{1.12,1,1}	1.04	0.96	1.00
S3	{S27,S4,S25}	3	{2.24,2.50,3}	2.58	0.38	2.11
S4	{S27,S15,S24}	1.12	{1.41,1.12,1.12}	1.21	0.38	1.02
S5	{S7,S12,S30}	1.12	{1.12,1.12,1.12}	1.12	0.89	0.94
S6	{S3,S4,S27}	20.25	{19.21,19.98,20.25}	19.81	0.05	13.13
S7	{S5,S30,S12}	1.12	{1.12,1.12,1.12}	1.12	0.89	0.94
S8	{S28,S29,S19}	1	{1.41,1,1}	1.14	0.88	1.08
S9	{S13,S30,S7}	2.06	{2.24,1.12,2.16}	1.84	0.54	1.30
S10	{S17,S1,S28}	8.5	{7.28,7.61,8.50}	7.80	0.13	4.17
S11	{S13,S9,S21}	20.05	{19.01,19.03,20.06}	19.37	0.05	10.00
S12	{S5,S20,S7}	1.12	{1.12,1,1.12}	1.08	0.93	0.97
S13	{S9,S30,S7}	2.24	{2.06,1.41,2.24}	1.90	0.53	1.34
S14	{S18,S23,S26}	1	{1,0.5,1}	0.83	1.20	0.94
S15	{S24,S4,S26}	1.12	{1.12,1.12,1.12}	1.12	0.89	1.09
S16	{S2,S20,S19}	1.12	{1,1,1.12}	1.04	0.96	1.00
S17	{S25,S28,S14}	2.69	{2,2.5,2.69}	2.40	0.42	2.07
S18	{S14,S23,S26}	1	{1,0.5,1}	0.83	1.20	0.87
S19	{S29,S2,S8}	1	{1,1,1}	1	1	0.95
S20	{S16,S2,S12}	1	{1.12,1,1.12}	1.08	0.93	1.02
S21	{S22,S1,S13}	2.24	{1.41,3.35,2.24}	2.33	0.43	1.48
S22	{S2,S16,S21}	1.41	{1.12,1.41,2.24}	1.59	0.63	1.24
S23	{S14,S18,S26}	0.5	{1,1,1}	1	1	1.20
S24	{S15,S4,S26}	1.12	{1.12,1.12,1.12}	1.12	0.89	1.09
S25	{S14,S18,S23}	1.41	{1.12,1.12,1.41}	1.22	0.82	1.38
S26	{S23,S14,S18}	1	{0.5,1,1}	0.83	1.20	0.94
S27	{S4,S15,S24}	1.41	{1.12,1.41,1.41}	1.31	0.76	1.14
S28	{S8,S29,S14}	1.41	{1,1,1.41}	1.17	0.85	1.21
S29	{S8,S19,S14}	1	{1,1,1}	1	1	1.03
S30	{S7,S5,S9}	1.12	{1.12,1.12,2.06}	1.43	0.70	1.10

5.4 LOFB-DRF

In this section, we again propose a new pruning method called *LOFB-DRF* that spawns a child RF that is 1) much smaller in size than the parent RF and 2) has an accuracy that is at least as good as that of the parent RF. In this enhancement, we use the LOF discussed in the previous section. As was the case with *CLUB-DRF*, in the remainder of this chapter, we will refer to the parent/original traditional RF as simply *RF*. We will refer to the resulted child RF based on our method as *LOFB-DRF*.

5.4.1 LOFB-DRF Algorithm

Figure 5.1 shows the *LOFB-DRF* approach and the corresponding algorithm is displayed in Algorithm 4 where T is the training set and N refers to the number of training samples. The

constant k refers to the number of trees that will have the highest weighted LOF values as will be discussed later. The domain of this constant is multiple of 5 in the range 5 to 40. The rationale behind this range was covered in Subsection 4.4.1 which was mainly to achieve extreme pruning levels in the range 92% to 99%. This way and as we shall see in the experiments section, we can compare the performance of *RF* with an *LOFB-DRF* of different sizes.

As shown in the algorithm, each tree predictions on the training dataset (denoted by the vector $C(t_i, T)$) is assigned a normalized LOF value that indicates the degree of its outlierness. Following a weighting process (to be discussed next), the top k trees corresponding to these predictions with the highest weighted LOF values are then selected to become members of the resulted *LOFB-DRF*.

It is important to remember that the size of the resulted *LOFB-DRF* is determined by the constant k. For example, if k is 5, then the resulted *LOFB-DRF* will have size 5, and so on.

Algorithm 4 LOFB-DRF Algorithm

{User Settings}
input T, N, k
{Process}
Create an empty vector $\overrightarrow{treesPredictions}$
Create an empty vector $\overrightarrow{LOFB-RF}$
Using N, call Random Forests Algorithm (Algorithm 1, *Chapter* 3) to create $\overrightarrow{RF}$
for $i = 1 \rightarrow RF.size()$ **do**
  $\overrightarrow{treesPredictions} = \overrightarrow{treesPredictions} \cup$ C(RF.tree(i), T)
end for
for $i = 1 \rightarrow \overrightarrow{treesPredictions}.size()$ **do**
  assignNormalizedLOF($\overrightarrow{treesPredictions}$.element(i))
end for
for $i = 1 \rightarrow \overrightarrow{treesPredictions}.size()$ **do**
  assignWeight($\overrightarrow{treesPredictions}$.element(i))
end for
Select the top k instances in $\overrightarrow{treesPredictions}$ with highest weighted LOF values
Select the corresponding trees from RF and add them to $\overrightarrow{LOFB-DRF}$
{Output}
A vector of trees $\overrightarrow{LOFB-DRF}$

5.4.2 Selection of Trees

With reference to Algorithm 4, the selection of trees in *RF* that will become members of *LOFB-DRF* proceeds as follows. First, predictions of each tree t_i on the training dataset T is computed as a vector $C(t_i, T)$ and added to the vector $\overrightarrow{treesPredictions}$. At the conclusion of the first **for** loop, $\overrightarrow{treesPredictions}$ becomes a super vector containing vectors where each vector stores the predictions of each tree. In the second **for** loop, each instance in $\overrightarrow{treesPredictions}$ is then

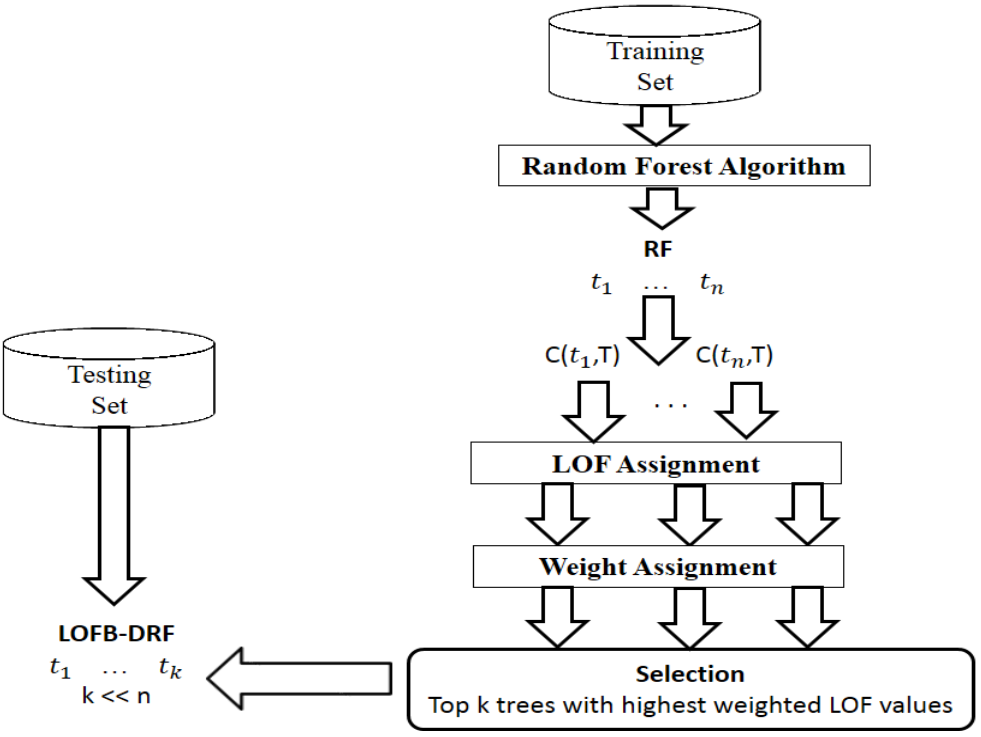


FIGURE 5.1: LOFB-DRF Approach

assigned a normalized LOF value between 0 and 1. This way, each normalized value describes the degree of the instance being an outlier Erich and Zimek (2011). In the third **for** loop, a weight is then assigned to each instance. For this, two variations will be used and are discussed next.

5.4.2.1 LOFB-DRF Weighted on Training

In this variation, a weight is calculated as the product of the normalized LOF value and the accuracy of the corresponding tree on the training data. Formally, let $C(t_i, T)$ be an instance in the super vector $\overrightarrow{treesPredictions}$, $LOF\_C(t_i, T)$ be the normalized LOF value assigned to this instance, and $AccuracyRate(t_i, T)$ be the accuracy of the tree (t_i on the training dataset T. The weight assigned to this instance is given by:

$$weight = LOF\_C(t_i, T) \times AccuracyRate(t_i, T) \tag{5.4}$$

The instances are then sorted in descending order by this weight and the corresponding top k trees are then selected. We will refer to the resulted *LOFB-DRF* that uses this variation as *WeightedOnTraining LOFB-DRF*.

5.4.2.2 LOFB-DRF Weighted on OOB

In this variation, instead of using the accuracy on the training data, we use the accuracy on the OOB. As discussed in Section 3.2, OOB refers to the instances that were not included in the sample with replacement that was used to construct the tree. The weight in this variation is calculated as:

$$weight = LOF\_C(t_i, T) \times AccuracyRate(t_i, OOB) \tag{5.5}$$

Similarly, the instances are then sorted in descending order by this weight and the corresponding top k trees are then selected. We will refer to the resulted *LOFB-DRF* that uses this variation as *WeightedOnOOB LOFB-DRF*.

5.5 Experimental Study

For the experiments, we used the 15 real datasets used in the previous chapter. As was the case with *CLUB-DRF*, the *RF* (parent) in Figure 5.1 had a size of 500 trees (a typical upper limit setting for RF Williams (2011)), and the *LOFB-DRF* algorithm described above was implemented using the Java programming language utilizing the API of WEKA. Again, this algorithm was run 10 times on each dataset where a new *RF* was created in each run. The average of the 10 runs for each resulted *LOFB-DRF* was calculated to produce the average for a variety of metrics including accuracy, standard deviation, F-Measure, and AUC. The minimum and maximum accuracy obtained over the 10 runs for each cluster increment were also calculated. For *RF*, only the average accuracy, F-Measure, and AUC have been calculated as shown in the last 3 columns in Tables 5.3 and 5.5.

5.5.1 Results

In this subsection, we report results of running the experiments on the two variations of *LOFB-DRF* described above.

5.5.1.1 LOFB-DRF Weighted on Training Results

Table 5.3 compares the performance of WeightedOnTraining *LOFB-DRF* and *RF* on the 15 datasets used in the experiment. To show the superiority of *LOFB-DRF*, we have highlighted in

boldface the average accuracy of *LOFB-DRF* when it is greater than that of *RF*. With the exception of the *audit*, *credit*, *white clover*, *glass*, *vehicle*, and *vote* datasets, *LOFB-DRF* outperformed *RF*. Interestingly enough, in all settings of *LOFB-DRF*, *LOFB-DRF* completely outperformed *RF* on 3 of the datasets, namely, *squash-stored*, *eucalyptus*, and *sonar*. Performance related statistics for this variation are depicted in Table 5.4. As demonstrated in the 3rd column of the table, it is easy to observe that *LOFB-DRF* outperformed *RF* on 9/15 datasets. For the datasets where *RF* outperformed *LOFB-DRF*, the difference of performance between *RF* and the best performer *LOFB-DRF* for such datasets is a small negligible fraction of less than 1%. The result of the paired t-test for this variation was statistically significant as a p-value of 0.03512 was achieved.

TABLE 5.3: Performance Metrics of WeightedOnTraining LOFB-DRF & RF

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure	AUC
breast-cancer								
5	67.01	61.86	74.23	3.16	0.65	0.57	71.13 0.65	0.58
10	67.22	64.95	69.07	1.71	0.66	0.58		
15	**71.34**	67.01	76.29	3.12	0.65	0.58		
20	69.48	67.01	73.20	2.62	0.66	0.58		
25	**71.86**	69.07	74.23	1.46	0.65	0.58		
30	70.41	68.04	72.16	1.53	0.65	0.58		
35	70.62	65.98	73.20	1.91	0.65	0.58		
40	69.18	64.95	72.16	2.14	0.65	0.58		
audit								
5	95.63	94.26	96.47	0.72	0.91	0.89	96.31 0.90	0.88
10	95.74	95.00	96.18	0.35	0.90	0.88		
15	95.99	95.29	96.47	0.35	0.90	0.88		
20	96.06	95.29	96.76	0.39	0.90	0.88		
25	96.22	95.88	96.47	0.25	0.91	0.89		
30	96.03	95.59	96.47	0.25	0.90	0.88		
35	96.26	95.88	96.47	0.18	0.90	0.88		
40	96.00	95.59	96.47	0.27	0.90	0.87		
credit								
5	63.00	47.65	72.65	8.58	0.67	0.61	76.06 0.66	0.60
10	71.56	66.18	74.71	2.39	0.67	0.61		
15	71.29	68.24	75.88	2.52	0.67	0.60		
20	73.65	71.47	77.06	1.78	0.67	0.60		
25	73.82	71.76	77.35	1.41	0.67	0.61		
30	74.50	71.76	76.47	1.53	0.67	0.60		
35	74.76	72.65	77.94	1.38	0.67	0.60		
40	75.24	71.47	76.47	1.51	0.67	0.60		
pasture								
5	**53.33**	41.67	75.00	10.67	0.48	0.60	41.67 0.43	0.56
10	33.33	8.33	58.33	16.67	0.42	0.54		
15	40.00	8.33	66.67	18.56	0.45	0.56		
20	26.67	8.33	50.00	13.33	0.42	0.54		
25	35.83	16.67	66.67	15.39	0.41	0.54		
30	33.33	16.67	58.33	11.18	0.42	0.55		
35	39.17	25.00	58.33	10.57	0.43	0.55		
40	37.50	16.67	58.33	11.33	0.44	0.55		
squash-unstored								
5	58.89	44.44	83.33	12.47	0.58	0.66	61.11 0.52	0.64
10	54.44	33.33	66.67	9.56	0.56	0.66		
15	60.56	50.00	83.33	8.77	0.55	0.65		
20	60.00	50.00	66.67	5.98	0.54	0.66		

Continued on next page

Table 5.3 – continued from previous page

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
25	**63.33**	55.56	77.78	7.93	0.54	0.65	
30	58.33	44.44	77.78	8.70	0.53	0.65	
35	**67.22**	50.00	83.33	10.08	0.54	0.66	
40	57.78	50.00	66.67	6.19	0.53	0.65	
squash-stored							
5	**56.67**	38.89	66.67	9.56	0.57	0.59	55.56 0.51 0.56
10	**59.44**	44.44	66.67	7.05	0.54	0.58	
15	**58.33**	50.00	66.67	4.48	0.54	0.58	
20	**58.33**	50.00	61.11	3.73	0.55	0.58	
25	**58.33**	50.00	66.67	5.12	0.53	0.57	
30	**56.67**	55.56	61.11	2.22	0.52	0.56	
35	**56.11**	55.56	61.11	1.67	0.52	0.57	
40	**56.11**	55.56	61.11	1.67	0.52	0.56	
white-clover							
5	50.95	38.10	57.14	6.41	0.52	0.63	63.81 0.51 0.61
10	55.24	42.86	61.90	6.10	0.53	0.63	
15	60.95	47.62	71.43	6.67	0.52	0.62	
20	58.57	52.38	66.67	4.29	0.52	0.63	
25	62.86	57.14	71.43	4.15	0.52	0.62	
30	59.52	52.38	66.67	3.84	0.52	0.62	
35	61.90	52.38	71.43	4.76	0.52	0.62	
40	59.05	47.62	66.67	6.10	0.52	0.62	
eucalyptus							
5	**25.80**	11.20	40.40	8.73	0.26	0.60	19.92 0.21 0.57
10	**21.00**	12.40	28.40	4.70	0.24	0.59	
15	**24.32**	14.80	32.00	5.01	0.24	0.58	
20	**24.48**	15.60	29.60	4.55	0.23	0.58	
25	**24.68**	21.20	29.60	2.35	0.23	0.58	
30	**24.80**	14.80	33.60	5.13	0.23	0.58	
35	**23.96**	20.00	34.40	4.20	0.23	0.58	
40	**21.16**	15.20	28.00	3.69	0.22	0.57	
soybean							
5	77.28	60.78	85.78	6.80	0.79	0.88	77.59 0.73 0.85
10	**78.45**	70.69	85.34	5.46	0.75	0.87	
15	**79.57**	72.84	83.62	3.50	0.76	0.87	
20	76.85	74.57	78.88	1.26	0.74	0.86	
25	76.90	74.14	79.31	1.88	0.74	0.86	
30	76.85	72.41	81.47	2.43	0.74	0.86	
35	77.33	71.98	82.33	3.66	0.73	0.86	
40	76.59	71.98	81.03	2.59	0.73	0.85	
diabetes							
5	65.63	59.39	74.71	4.26	0.70	0.68	73.52 0.68 0.66
10	71.03	67.05	75.10	2.08	0.69	0.67	
71.19	68.58	76.25	2.00	0.69	0.67		
20	71.42	67.82	73.56	1.84	0.69	0.67	
25	73.49	71.65	75.10	1.07	0.69	0.67	
30	72.53	69.35	74.33	1.51	0.69	0.66	
35	**73.64**	71.65	76.25	1.43	0.69	0.66	
40	72.57	70.88	75.10	1.58	0.68	0.66	
glass							
5	57.40	49.32	67.12	5.21	0.59	0.73	63.97 0.57 0.71
10	62.88	60.27	65.75	1.98	0.59	0.72	
15	61.51	58.90	64.38	1.56	0.58	0.72	
20	61.51	58.90	65.75	2.41	0.58	0.72	
25	62.60	58.90	65.75	2.04	0.58	0.72	
30	62.05	58.90	65.75	2.30	0.58	0.72	
35	63.01	60.27	65.75	1.37	0.58	0.72	
40	61.78	57.53	64.38	1.78	0.58	0.72	
car							

Continued on next page

Table 5.3 – continued from previous page

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
5	61.80	60.03	63.27	0.92	0.56	0.79	62.21 0.56 0.78
10	**62.69**	60.37	63.95	1.06	0.56	0.78	
15	62.09	60.71	63.95	0.87	0.56	0.78	
20	**62.77**	62.07	63.95	0.66	0.56	0.79	
25	**62.26**	60.71	63.95	0.83	0.56	0.78	
30	**62.33**	60.88	63.27	0.81	0.56	0.78	
35	**62.53**	61.73	63.78	0.67	0.56	0.78	
40	61.92	60.54	63.27	0.80	0.56	0.78	
sonar							
5	**5.49**	1.41	11.27	3.11	0.24	0.00	0.14 0.29 0.00
10	**4.65**	0.00	12.68	3.57	0.27	0.00	
15	**2.96**	0.00	8.45	2.55	0.27	0.00	
20	**4.08**	1.41	8.45	2.13	0.30	0.00	
25	**4.23**	0.00	9.86	2.82	0.29	0.00	
30	**2.39**	0.00	4.23	1.27	0.29	0.00	
35	**2.54**	0.00	5.63	1.87	0.28	0.00	
40	**1.83**	0.00	4.23	1.55	0.28	0.00	
vehicle							
5	65.24	52.08	73.96	8.14	0.70	0.80	73.58 0.69 0.80
10	70.24	66.67	73.26	2.30	0.70	0.80	
15	72.43	68.75	76.04	2.45	0.70	0.80	
20	72.40	68.75	75.35	1.66	0.69	0.80	
25	72.64	70.49	75.35	1.38	0.69	0.80	
30	73.09	71.88	74.65	0.87	0.70	0.80	
35	72.43	70.49	74.65	1.23	0.69	0.80	
40	73.33	71.53	75.69	1.25	0.69	0.80	
vote							
5	96.82	95.27	97.97	0.80	0.96	0.98	97.97 0.95 0.97
10	97.09	95.27	97.97	0.86	0.96	0.97	
15	97.57	96.62	97.97	0.45	0.95	0.97	
20	97.43	96.62	97.97	0.51	0.95	0.97	
25	97.57	96.62	97.97	0.45	0.95	0.97	
30	97.70	97.30	97.97	0.33	0.95	0.97	
35	97.64	96.62	97.97	0.45	0.95	0.97	
40	97.64	96.62	97.97	0.45	0.95	0.97	

5.5.1.2 LOFB-DRF Weighted on OOB Results

Table 5.5 compares the performance of WeightedOnOOB *LOFB-DRF* and *RF* on the 15 datasets used in the experiment. As demonstrated in the table, this variation did not perform as well as the previous one. With the exception of the datasets *squash-unstored*, *eucalyptus*, and *sonar*, *RF* outperformed *LOFB-DRF*. Performance related statistics for this variation are depicted in Table 5.6. Though *RF* outperformed *LOFB-DRF* on the majority of the datasets, the outperformance range is small and insignificant. Due to the unsatisfactory performance of this variation, the result of the paired t-test was statistically insignificant as a p-value of 0.2159 was achieved.

Table 5.4: Performance Statistics of WeightedOnTraining LOFB-DRF

Dataset	RF	Winning LOFB-DRFs	Best Performance	Worst Performance
breast-cancer	71.13%	2	71.86%	67.01%
audit	96.31%	0	96.26%	95.63%
credit	76.06%	0	75.24%	63.00%
pasture	41.67%	1	53.33%	26.67%
squash-unstored	61.11%	2	67.22%	54.44%
squash-stored	55.56%	8	59.44%	56.11%
white-clover	63.81%	0	62.86%	50.95%
eucalyptus	19.92%	8	25.80%	21.00%
soybean	77.59%	2	79.57%	76.59%
diabetes	73.52%	1	73.64%	65.63%
glass	63.97%	0	63.01%	57.40%
car	62.21%	5	62.77%	61.80%
sonar	0.14%	8	5.49%	1.83%
vehicle	73.58%	0	73.33%	65.24%
vote	97.97%	0	97.70%	96.82%

Table 5.5: Performance Metrics of WeightedOnOOB LOFB-DRF& RF

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
breast-cancer							
5	62.78	57.73	68.04	3.52	0.61	0.55	71.96 0.65 0.58
10	61.34	56.70	68.04	3.24	0.62	0.56	
15	67.94	62.89	72.16	2.93	0.63	0.57	
20	66.70	60.82	72.16	3.26	0.63	0.57	
25	69.48	63.92	75.26	3.33	0.63	0.57	
30	68.25	62.89	74.23	3.57	0.63	0.57	
35	69.38	62.89	77.32	3.94	0.63	0.57	
40	68.25	61.86	72.16	3.32	0.63	0.57	
audit							
5	89.46	87.21	92.50	1.71	0.83	0.799	96.31 0.90 0.88
10	89.28	87.79	91.18	1.09	0.84	0.79	
15	92.26	91.62	93.24	0.49	0.84	0.79	
20	91.94	91.03	93.53	0.63	0.84	0.80	
25	93.12	91.91	94.41	0.73	0.84	0.80	
30	92.88	91.91	93.53	0.50	0.84	0.80	
35	93.56	92.79	94.12	0.38	0.84	0.80	
40	93.38	92.65	93.97	0.48	0.85	0.80	
credit							
5	68.71	66.47	73.24	2.04	0.64	0.58	75.71 0.66 0.60
10	66.56	62.65	69.71	2.03	0.65	0.59	
15	73.41	72.06	74.71	0.89	0.65	0.59	
20	71.03	69.41	73.24	1.15	0.65	0.59	
25	74.00	72.65	76.76	1.18	0.65	0.59	
30	71.88	69.71	75.00	1.49	0.65	0.59	
35	74.26	72.06	76.47	1.28	0.65	0.59	
40	72.47	70.88	73.82	1.05	0.65	0.59	
pasture							
5	39.17	16.67	58.33	13.46	0.42	0.57	40.83 0.43 0.57
10	31.67	8.33	66.67	16.16	0.43	0.57	
15	35.00	25.00	50.00	7.26	0.42	0.56	
20	31.67	8.33	50.00	12.25	0.41	0.55	
25	35.83	25.00	58.33	10.57	0.41	0.55	

Continued on next page

Table 5.5 – continued from previous page

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
30	27.50	8.33	50.00	11.81	0.41	0.55	
35	32.50	16.67	41.67	7.86	0.41	0.55	
40	25.83	16.67	41.67	8.70	0.41	0.55	
squash-unstored							
5	52.78	33.33	66.67	11.98	0.49	0.61	58.89 0.51 0.64
10	51.11	33.33	66.67	9.23	0.52	0.63	
15	**61.67**	50.00	72.22	6.31	0.52	0.63	
20	54.44	44.44	72.22	6.94	0.52	0.63	
25	**61.11**	50.00	72.22	6.57	0.51	0.63	
30	57.78	38.89	72.22	9.69	0.51	0.63	
35	58.33	38.89	72.22	10.02	0.50	0.63	
40	55.00	38.89	72.22	8.77	0.50	0.62	
squash-stored							
5	47.78	16.67	66.67	12.96	0.45	0.54	55.56 0.51 0.56
10	41.67	16.67	55.56	12.23	0.46	0.55	
15	50.56	27.78	66.67	11.77	0.47	0.55	
20	48.33	27.78	61.11	8.98	0.47	0.55	
25	50.56	38.89	66.67	8.41	0.47	0.55	
30	47.78	38.89	61.11	7.54	0.47	0.55	
35	48.89	38.89	61.11	6.48	0.47	0.55	
40	46.11	33.33	61.11	8.26	0.47	0.55	
white-clover							
5	47.14	28.57	61.90	11.94	0.45	0.57	64.29 0.51 0.61
10	37.14	23.81	47.62	8.19	0.46	0.58	
15	48.10	28.57	61.90	11.94	0.47	0.58	
20	44.76	14.29	66.67	14.32	0.47	0.59	
25	51.43	33.33	66.67	8.98	0.48	0.59	
30	50.48	33.33	61.90	8.83	0.48	0.59	
35	52.38	38.10	61.90	7.38	0.48	0.59	
40	49.52	42.86	61.90	6.80	0.48	0.59	
eucalyptus							
5	**20.88**	14.40	26.00	3.99	0.22	0.57	20.16 0.21 0.57
10	14.56	8.40	19.20	3.47	0.21	0.57	
15	18.92	12.00	22.80	3.15	0.21	0.57	
20	16.76	11.60	22.00	2.88	0.21	0.57	
25	18.24	14.40	23.20	2.50	0.21	0.57	
30	16.72	12.00	22.00	3.07	0.21	0.57	
35	17.96	11.60	21.60	2.73	0.21	0.57	
40	16.84	10.80	20.40	3.11	0.21	0.57	
soybean							
5	58.71	50.43	63.79	3.88	0.60	0.79	77.59 0.73 0.85
10	54.74	49.14	58.62	3.18	0.61	0.79	
15	61.25	58.19	63.79	1.88	0.61	0.79	
20	59.18	56.90	62.07	1.53	0.62	0.79	
25	61.42	59.05	63.36	1.56	0.62	0.80	
30	60.09	56.47	63.36	1.68	0.62	0.80	
35	62.33	59.91	64.66	1.50	0.63	0.80	
40	61.98	59.48	64.22	1.22	0.63	0.80	
diabetes							
5	71.65	67.43	74.33	2.07	0.67	0.65	73.64 0.68 0.66
10	66.97	63.22	70.11	2.00	0.67	0.65	
15	72.22	70.88	75.10	1.13	0.67	0.65	
20	69.81	68.20	71.65	1.12	0.67	0.65	
25	73.03	71.26	74.71	1.05	0.67	0.65	
30	71.23	68.58	73.18	1.22	0.67	0.65	
35	73.10	70.88	75.86	1.41	0.67	0.65	
40	72.11	69.73	74.71	1.57	0.67	0.65	
glass							
5	56.03	52.05	60.27	2.63	0.52	0.68	63.42 0.58 0.71

Continued on next page

Table 5.5 – continued from previous page

LOFB-DRF Size	AVG	MIN	MAX	SD	F-Measure	AUC	AVG F-Measure AUC
10	53.42	47.95	58.90	3.78	0.53	0.68	
15	57.67	50.68	63.01	3.32	0.53	0.68	
20	55.89	50.68	60.27	3.67	0.54	0.69	
25	58.63	54.79	64.38	3.05	0.54	0.69	
30	57.12	52.05	63.01	3.36	0.54	0.69	
35	59.45	56.16	64.38	2.68	0.54	0.69	
40	59.04	57.53	63.01	1.88	0.55	0.69	
car							
5	61.12	60.54	61.56	0.35	0.52	0.72	62.18 0.56 0.78
10	60.70	60.20	61.05	0.27	0.52	0.73	
15	60.90	60.54	61.05	0.18	0.53	0.73	
20	60.70	60.37	60.88	0.18	0.53	0.73	
25	60.71	60.37	60.88	0.15	0.53	0.74	
30	60.66	60.54	60.88	0.13	0.53	0.74	
35	60.73	60.37	60.88	0.18	0.53	0.74	
40	60.66	60.37	60.88	0.15	0.53	0.74	
sonar							
5	**13.10**	4.23	22.54	5.91	0.38	0.00	0.0 0.29 0.00
10	**7.04**	1.41	14.08	3.45	0.36	0.00	
15	**10.14**	4.23	19.72	4.17	0.38	0.00	
20	**6.06**	1.41	12.68	2.96	0.37	0.00	
25	**8.03**	2.82	14.08	3.51	0.37	0.00	
30	**6.06**	2.82	11.27	2.52	0.36	0.00	
35	**6.62**	2.82	9.86	2.28	0.35	0.00	
40	**5.49**	2.82	8.45	2.13	0.35	0.00	
vehicle							
5	70.24	65.63	72.57	2.21	0.66	0.78	74.13 0.69 0.80
10	67.12	64.93	69.79	1.68	0.67	0.78	
15	71.84	69.79	73.26	1.00	0.67	0.78	
20	70.17	69.10	71.53	0.74	0.67	0.78	
25	72.53	70.49	74.31	1.21	0.67	0.78	
30	71.08	70.14	72.57	0.88	0.67	0.78	
35	72.99	71.88	74.65	0.74	0.67	0.78	
40	72.01	70.49	73.96	1.02	0.67	0.78	
vote							
5	95.95	94.59	97.30	0.80	0.93	0.96	97.97 0.95 0.97
10	95.27	93.24	97.30	1.17	0.93	0.96	
15	96.35	95.95	97.30	0.45	0.93	0.96	
20	96.01	94.59	97.30	0.88	0.93	0.96	
25	96.62	95.27	97.30	0.60	0.93	0.96	
30	96.42	95.27	97.97	0.80	0.93	0.96	
35	97.09	96.62	97.97	0.53	0.93	0.96	
40	96.62	95.95	97.97	0.60	0.93	0.96	

5.5.2 Analysis of the Results

5.5.2.1 LOFB-DRF Weighted on Training Analysis

Using the information depicted in Table 5.3, for each dataset, Figure 5.2 shows the number of WeightedOnTraining *LOFB-DRF* pruned ensembles that have outperformed *RF*. It is easy to

TABLE 5.6: Performance Statistics of WeightedOnOOB LOFB-DRF

Dataset	RF	Winning LOFB-DRFs	Best Performance	Worst Performance
breast-cancer	71.96%	0	69.48%	61.34%
audit	96.31%	0	93.56%	89.28%
credit	75.71%	0	74.26%	66.56%
pasture	40.83%	0	39.17%	25.83%
squash-unstored	58.89%	2	61.67%	51.11%
squash-stored	55.56%	0	50.56%	41.67%
white-clover	64.29%	0	52.38%	37.14%
eucalyptus	20.16%	1	20.88%	14.56%
soybean	77.59%	0	62.33%	54.74%
diabetes	73.64%	0	73.10%	66.97%
glass	63.42%	0	59.45%	53.42%
car	62.18%	0	61.12%	60.66%
sonar	0.0%	8	13.10%	5.49%
vehicle	74.13%	0	72.99%	67.12%
vote	97.97%	0	97.09%	95.27%

see in the figure that *LOFB-DRF* has outperformed *RF* on 9/15 datasets, of which, *LOFB-DRF* has completely outperformed *RF* on 3 datasets: *squash-stored*, *eucalyptus*, and *sonar*. More precisely, a pruned RF using *LOFB-DRF*, regardless of its size, outperformed *RF* on these datasets.

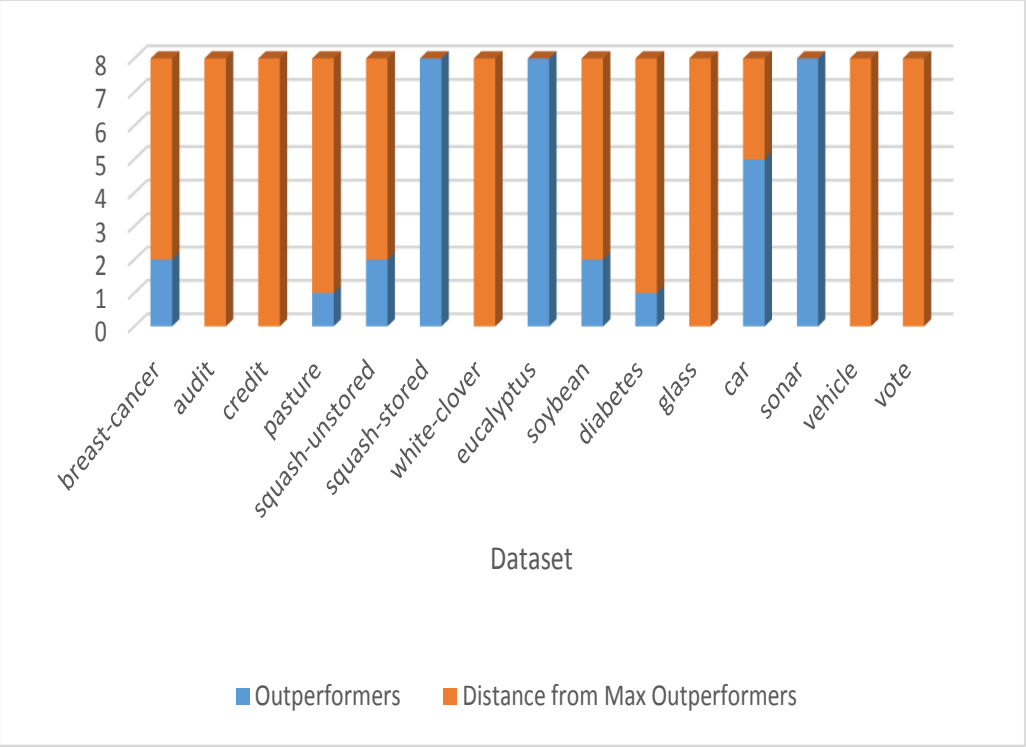


FIGURE 5.2: Number of WeightedOnTraining LOFB-DRF Outperformers per Dataset

5.5.2.2 LOFB-DRF Weighted on OOB Analysis

Since WeightedOnOOB *LOFB-DRF* pruned ensembles did not perform well as discussed above, less favorable results are obtained as shown in Figure 5.3. This figure is based on the information presented in Table 5.5.

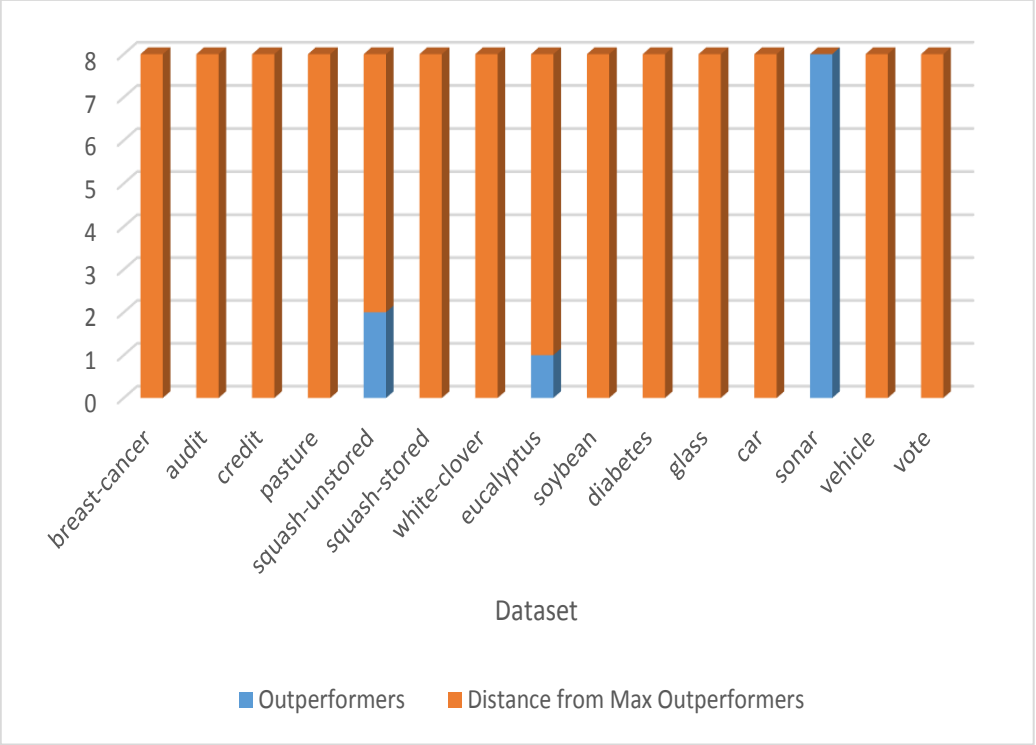


FIGURE 5.3: Number of WeightedOnOOB LOFB-DRF Outperformers per Dataset

5.5.3 Outperformance Range

Tables 5.7 and 5.8 depict the outperformance range of the two variations of *LOFB-DRF* over *RF*. A negative number indicates that *RF* was superior to *LOFB-DRF* and the absolute value of this number refers to the performance difference between *RF* and best performer *LOFB-DRF*. Taking a closer look at Table 5.7, we can see that *LOFB-DRF* outperformed *RF* on 9/15 datasets with a maximum outperformance range of 11.66%. For the few datasets where *RF* outperformed *LOFB-DRF*, the outperformance difference for such datasets was only less than 1%. As for Table 5.8, *LOFB-DRF* outperformed *RF* on 3 datasets with a maximum outperformance range of 13.10%. Though *RF* outperformed *LOFB-DRF* on the majority of the datasets, the outperformance range, with the exception of the *white-clover* and *soybean* datasets, is small and insignificant.

TABLE 5.7: Outperformance Range of WeightedOnTraining LOFB-DRF over RF

Dataset	Range
breast-cancer	0.21% - 0.73%
audit	-0.05%
credit	-0.82%
pasture	11.66% (only one outperformer found)
squash-unstored	2.22% - 6.11%
squash-stored	0.55% - 3.88%
white clover	-0.95%
eucalyptus	1.08% - 5.88%
soybean	0.86% - 1.98%
diabetes	0.12% (only one outperformer found)
glass	-0.96%
car	0.05% - 0.56%
vehicle	-0.25%
vote	-0.27%

TABLE 5.8: Outperformance Range of WeightedOnOOB LOFB-DRF over RF

Dataset	Range
breast-cancer	-2.48%
audit	-2.75%
credit	-1.45%
pasture	-1.66%
squash-unstored	2.22% - 2.78%
squash-stored	-5.00%
white clover	-11.91%
eucalyptus	0.72% (only one outperformer found)
soybean	-15.26%
diabetes	-0.54%
glass	-3.97%
car	-1.06%
vehicle	-1.14%
vote	-0.88%

5.5.4 Pruning Level

5.5.4.1 LOFB-DRF Weighted on Training Pruning Level

Pruning level in ensemble pruning was discussed in Subsection 4.5.4. Table 5.9 shows the pruning levels where the first column shows the maximum possible pruning level for an *LOFB-DRF* that has outperformed *RF*, and the second column shows the pruning level of the best performer *LOFB-DRF*. With extremely healthy pruning levels ranging from 93% to 99% as depicted in the first column, our technique outperformed *RF*. Like *CLUB-DRF*, this makes *LOFB-DRF* a natural choice for real-time applications, where fast classification is an important desideratum

(refer back to Subsection 4.5.5 for more details). The interesting observation in this table is that, in both columns, the mode is 99% which refers to the highest pruning level that was achieved by *LOFB-DRF* which corresponds to 5 trees only.

TABLE 5.9: Maximum Pruning Level with Best Possible Performance (WeightedOnTraining LOFB-DRF)

Dataset	Maximum Pruning Level	Best Performance Pruning Level
breast-cancer	97%	95%
pasture	99%	99%
squash-unstored	95%	93%
squash-stored	99%	98%
eucalyptus	99%	99%
soybean	98%	97%
diabetes	93%	93%
car	98%	96%

5.5.4.2 LOFB-DRF Weighted on OOB Pruning Level

Table 5.10 shows the pruning levels for this variation. As expected, only 2 datasets are listed as these are the datasets that outperformed *RF*. Note that the mode in both columns is also 99%.

TABLE 5.10: Maximum Pruning Level with Best Possible Performance (WeightedOnOOB LOFB-DRF)

Dataset	Maximum Pruning Level	Best Performance Pruning Level
squash-unstored	97%	97%
eucalyptus	99%	99%

5.6 Time Complexity Analysis

The time complexity analysis of *CLUB-DRF* in Section 4.6 also holds for *LOFB-DRF*. Reason being is that both techniques achieve the same pruning level, hence, depending on the pruning level used, pruned ensembles produced by both techniques will have identical sizes. Therefore, they are likely to have the same time complexity given by $O(m \times O(f))$, where f refers to the number of features in the dataset, $O(f)$ refers to order of traversing the average number of nodes of all trees in an LOFB-DRF, $m = n - n \times \alpha$, is the number of trees in *LOFB-DRF*, n is the number of trees in *RF*, and $0 < \alpha < 1$ is the pruning level.

5.7 Summary

We have used the Local Outlier Factor as a new ensemble pruning technique for the extreme pruning of Random Forests. This technique was used to produce an enhancement of Random Forests called *LOFB-DRF*. As demonstrated in the charts in Figures 5.2 and 5.3, the *WeightedOnTraining LOFB-DRF* variation performed better than the *WeightedOnOOB LOFB-DRF* variation. By outperforming *RF* on 9/15 datasets and a maximum outperformance range of 11.66%, the extremely pruned *LOFB-DRF* ensembles using the *WeightedOnTraining* variation are not only typical for real-time applications, but also for machines and mobile devices with limited memory and processing power. Interestingly enough, a 99% pruning level was the mode in both columns in Tables 5.9 and 5.10. This means that for the majority of the datasets, the smallest *LOFB-DRF* that outperformed *RF* is of size 5, and the best performer is also of size 5. Size 5 is the size of the smallest pruned ensemble produced by *LOFB-DRF* (and also by *CLUB-DRF*) and corresponds to a 99% pruning level; the highest pruning level achieved by both *LOFB-DRF* and *CLUB-DRF*.

Part III

Feature Engineering

Chapter 6

Diversified Subspaces-Based Random Forest (DSB-RF)

6.1 Overview

Findings reported in this chapter have been published in Fawagreh et al. (2014b). This chapter introduces the first enhancement related to feature engineering which is termed *DSB-RF*. Our method aims at improving feature interaction which is not an inherent characteristic of RFs. As subsequently discussed in this chapter, this is done by using weighted subspaces where each subspace contains a fixed random subset of features.

6.2 Background

By continuing to utilize the principle that diversity can lead to better performance, this chapter proposes a new method to promote diversity in RFs. The method uses randomly selected subspaces, giving a weight to each subspace according to its predictive power, and using this weight in majority voting. Such random subspaces are likely to improve feature interaction which is not an inherent characteristic of RFs, and to reduce the correlation among them. Because highly correlated trees in an RF can increase the forest error rate Breiman (2001) Bernard et al. (2010), we expect our method to reduce the error rate and therefore, improve the performance. The new forest is termed Diversified Subspaces-Based Random Forest (*DSB-RF*). What distinguishes *DSB-RF* from the standard RF is that, in the latter, all features are used during the construction of each tree in the forest. In *DSB-RF*, however, only a subset of the features is used to construct each tree in each sub-forest, where each sub-forest, as described in the next section, corresponds

to a subspace. Experimental study on 15 real datasets showed favorable results, demonstrating the potential of the proposed method.

6.3 DSB-RF

As aforementioned, the method uses randomly selected subspaces. It is worth mentioning that the random subspace method was initially introduced by Ho (1998), however, and unlike the approach proposed in this chapter, the subspaces were not weighted according to their predictive power.

To some extent, the standard RF already applies some diversity to the classifiers being built during the construction of the RF. In a nutshell, there are two levels of diversity being applied. The first level is when each decision tree is constructed using sampling with replacement from the training data. The samples are likely to have some diversity among each other as they were drawn at random. The second level is achieved by randomization which is applied when selecting the best node to split on.

The ultimate objective of this chapter is to add a third level of diversity by injecting more diversity in an RF. From the training set, a number of subspaces is created. The number of subspaces is determined as follows:

$$Subspaces = \lfloor \alpha \times Z \rfloor \tag{6.1}$$

where α denotes the subspace factor such that $0 < \alpha \leq 1$ ($\alpha \in \mathbb{R}$), and Z is the size of the *DSB-RF* to be created. The floor function ($\lfloor \rfloor$) returns the largest integer less than or equal to a given number. Each subspace will contain a fixed randomized subset of the total number of features and will eventually correspond to a sub-forest. A projected training dataset will be created for each subspace and will be used to create the trees in the corresponding sub-forest using bagging. Likewise, from the testing dataset, a projected testing dataset is created for each subspace. The number of trees in each sub-forest is given by the equation

$$Trees = \frac{Z}{Subspaces} \tag{6.2}$$

The resulted *DSB-RF* is shown in Figure 6.1.

A weight is then assigned to each projected training dataset using the Absolute Predictive Power (APP) given by Cuzzocrea et al. (2013). Given a dataset S, the APP is defined by the following equation

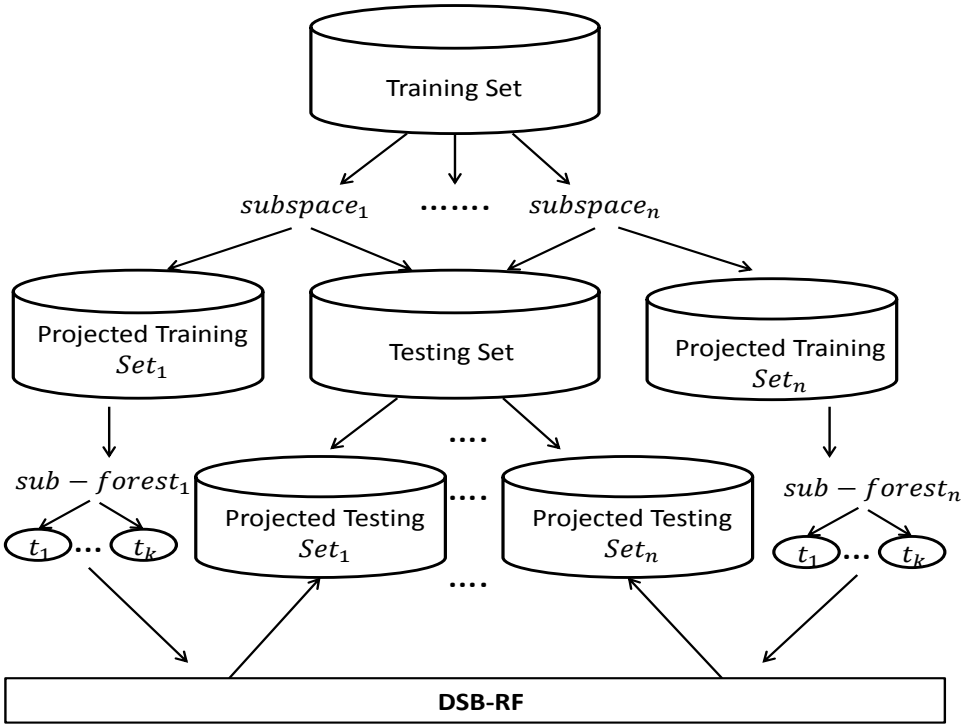


FIGURE 6.1: DSB-RF

$$APP(S) = \frac{1}{|Att(S)|} \times \sum_{A \in Att(S)} \frac{I(S,A)}{E(S)} \tag{6.3}$$

where $E(S)$ is the entropy of a given dataset S, having K instances, and $I(S,A)$ is the information gain of a given attribute A in a dataset S. $E(S)$ is a measure of the uncertainty in a random variable and is given by the following equation

$$E(S) = \sum_{i=1}^{K} -p_i(x_i) \log_2 p_i(x_i) \tag{6.4}$$

where x_i refers to a generic instance of S and $p_i(x_i)$ denotes the probability that the instance x_i occurs in S. $I(S,A)$ is given by

$$I(S,A) = E(S) - \sum_{v \in Val(A)} \left(\frac{|S_v|}{|S|} \right) E(S_v) \tag{6.5}$$

where $E(S)$ denotes the entropy of S, $Val(A)$ denotes the set of possible values for A, S_v refers to the subset of S for which A has the value v, and $E(S_v)$ denotes the entropy of S_v.

This weight will be inherited by the corresponding sub-forest and will be used in the voting process. This means that the standard voting technique currently used in the standard RF is going to be replaced by a weighted voting technique.

As an example to illustrate how to calculate APP for a given dataset S, consider the the training data depicted in Table 3.1 of Chapter 3. As shown in the table, the feature space consists of the features *Outlook*, *HWDone*, *Weekend*, and *Play*. Assume that the subspace chosen is *Outlook*, *Weekend*, and *Play*, the projected training data according to this subspace is shown in Table 6.1 below.

TABLE 6.1: Projected Training Data

Outlook	Weekend	Play
sunny	true	yes
sunny	false	yes
sunny	true	yes
sunny	false	no
rainy	true	yes
rainy	false	no
rainy	true	yes
rainy	false	no

Assuming that that projected training data in Table 6.1 is referred to as S, the APP of S according to Equation 6.3 is given as:

$$APP(S) = \frac{1}{2} \times \left(\frac{I(S, Outlook)}{E(S)} + \frac{I(S, Weekend)}{E(S)} \right) \tag{6.6}$$

Starting with the denominator in Equation 6.6, according to Equation 6.4, $E(S)$ is calculated as:

$$E(S) = -\frac{2}{8}\log_2\frac{2}{8} + -\frac{2}{8}\log_2\frac{2}{8} + -\frac{2}{8}\log_2\frac{2}{8} + -\frac{2}{8}\log_2\frac{2}{8} = 4 \times (-\frac{2}{8} \times -2) = 2 \tag{6.7}$$

As for the numerator, $I(S, Outlook)$, according to Equation 6.5, is calculated as:

$$\begin{aligned} I(S, Outlook) &= 2 - \left(\frac{|S_{sunny}|}{|S|} E(S_{sunny}) + \frac{|S_{rainy}|}{|S|} E(S_{rainy}) \right) \\ &= 2 - \left(\frac{4}{8} E(S_{sunny}) + \frac{4}{8} E(S_{rainy}) \right) \end{aligned} \tag{6.8}$$

According to Equation 6.4:

$$E(S_{sunny}) = E(S_{rainy}) = -\frac{4}{8}\log_4\frac{4}{8} = \frac{1}{2}$$

Substituting $E(S_{sunny})$ and $E(S_{rainy})$ into Equation 6.8 yields:

$$I(S, Outlook) = 2 - \left(\frac{4}{8} \times \frac{1}{2} + \frac{4}{8} \times \frac{1}{2}\right) = 1\frac{1}{2}$$

Repeating the same calculations for $I(S, Weekend)$ in Equation 6.6 yields:

$$\begin{aligned} I(S, Weekend) &= 2 - \left(\frac{|S_{true}|}{|S|} E(S_{true}) + \frac{|S_{false}|}{|S|} E(S_{false})\right) \\ &= 2 - \left(\frac{4}{8} E(S_{true}) + \frac{4}{8} E(S_{false})\right) \end{aligned} \tag{6.9}$$

According to Equation 6.4:

$$E(S_{true}) = E(S_{false}) = -\frac{4}{8} \log_4 \frac{4}{8} = \frac{1}{2}$$

Substituting $E(S_{true})$ and $E(S_{false})$ into Equation 6.9 yields:

$$I(S, Weekend) = 2 - \left(\frac{4}{8} \times \frac{1}{2} + \frac{4}{8} \times \frac{1}{2}\right) = 1\frac{1}{2}$$

Finally, putting everything together by substituting the values $I(S, Outlook)$, $I(S, Weekend)$, and $E(S)$ into Equation 6.6 produces:

$$APP(S) = \frac{1}{2} \times (1.5 + 1.5) = \frac{3}{4} = 0.75 \tag{6.10}$$

6.3.1 Algorithm

Algorithm 5 summarises the main steps involved in the construction of a *DSB-RF* according to the aforementioned discussion where Z is the desired size of the *DSB-RF* to be created, α is the subspace factor, and $featurePercent$ is desired percentage of features in the subspace.

As shown in the algorithm above, after creating empty vectors for the *DSB-RF* to be generated and for the weights that will be assigned to the sub-forests, using Equation 6.1, the algorithm creates N subspaces each containing $featurePercent$ of all features F drawn at random. Each subspace is then used to create projected training and testing datasets. A weight to each projected training set is assigned using Equation 6.3. After calculating the number of trees in each

Algorithm 5 DSB-RF Algorithm

{User Settings}
input Z, α, $featurePercent$
{Process}
Create an empty vector $\overrightarrow{DSB\text{-}RF}$
Create an empty vector $\overrightarrow{Weights}$
Using Equation 6.1, create N subspaces each containing $featurePercent$ of the features chosen at random from all features F
For each subspace i in the previous step, create a projected training set TR_i
Repeat the previous step to create a projected testing set TS_i for each subspace (this is required for the testing phase)
Using Equation 6.3, assign a weight to each projected training set and add this weight to $\overrightarrow{Weights}$
Using Equation 6.2, determine the number of trees in each subspace:

$$\text{treesPerSubspace} = Z/N$$

for $i = 1 \rightarrow N$ **do**
  for $j = 1 \rightarrow treesPerSubspace$ **do**
    Create an empty tree T_j
    repeat
      Sample S out of all features in the corresponding projected training set TR_i using Bootstrap sampling
      Create a vector of the S features $\overrightarrow{F_S}$
      Find Best Split Feature $B(\overrightarrow{F_S})$
      Create a New Node using $B(\overrightarrow{F_S})$ in T_j
    until No More Instances To Split On
    Add T_j to the $\overrightarrow{DSB\text{-}RF}$
  end for
end for
{Output}
A vector of trees $\overrightarrow{DSB\text{-}RF}$
A vector of weights $\overrightarrow{Weights}$ (to replace standard voting by weighted voting)

subspace ($treesPerSubspace$) using Equation 6.2, the outer and inner **for** loops create N sub-forests where each sub-forest contains $treesPerSubspace$ trees. Each tree in each sub-forest is constructed from a sample S out of all features in the corresponding projected training set using Bootstrap sampling. The output of the algorithm are a vector of trees $\overrightarrow{DSB\text{-}RF}$, and a vector of weights $\overrightarrow{Weights}$ (to be used in the voting process).

6.4 Experimental Study

Traditional RFs and *DSB-RF*s were tested on 15 real datasets from the UCI repository Bache and Lichman (2013) and both had an initial size of 500 trees. To allow enough diversity, two types of *DSB-RF*s will be created: the first contains 10 sub-forests with 50 trees each, and the second contains 20 sub-forests with 25 trees each. To create such *DSB-RF*s, we used as subspace

factors 2% and 4% respectively. By Equations 6.1 and 6.2, these subspace factors produced these two types of *DSB-RF*s. In the remainder of this chapter, we will refer to these types as *Type I* and *Type II* respectively. We used a random 50%, 60%, 70%, 80%, and 90% of the features for each subspace.

6.4.1 Results

Detailed results of the experiments, based on the average of 10 runs, are depicted in Tables 6.2 and 6.3. In these tables, we have highlighted in boldface the average accuracy of the resulted *DSB-RF* when it is greater than that of RF, and underlined the accuracy when it is equal to that of RF. Interestingly enough, taking a closer look at Tables 6.2 and 6.3, you will see that *DSB-RF* has performed consistently well on the medical datasets, namely, *diabetes* and *breast-cancer*.

As demonstrated in Table 6.4 in the next subsection, *Type I* performed well on 70% of the features, and *Type II* performed well on 90% of the features. The result of the paired t-test for these two cases is a p-value of 0.3947 and 0.6795 respectively. These values suggest no statistical significance. A further analysis of the proposed method will be given next followed by a proposed solution using Replicator Dynamics (RD) in the next chapter.

TABLE 6.2: Accuracy Comparison Between RF & DSB-RF Type I

	RF	DSB-RF				
Dataset		50%	60%	70%	80%	90%
pasture	40.83%	36.67%	40.83%	40.83%	40.83%	40.83%
squash-unstored	60.56%	**61.67%**	**61.67%**	**61.11%**	60.56%	60.00%
squash-stored	55.56%	54.44%	55.00%	53.33%	54.44%	55.00%
white-clover	63.33%	62.86%	63.33%	**63.81%**	63.33%	62.86%
glass	12.33%	10.82%	10.82%	12.33%	11.92%	11.92%
breast-cancer	71.65%	**77.53%**	**76.19%**	**75.46%**	**74.02%**	**73.51%**
diabetes	72.76%	72.64%	**74.37%**	**79.35%**	**73.72%**	**73.98%**
vote	97.97%	96.35%	97.09%	97.36%	97.43%	97.57%
car	62.31%	59.25%	60.36%	59.93%	61.51%	61.04%
vehicle	73.96%	71.35%	71.84%	73.96%	72.81%	73.82%
eucalyptus	19.72%	**20.48%**	**20.36%**	**22.76%**	**21.20%**	19.68%
sonar	0.28%	**1.55%**	**0.99%**	**0.99%**	0.00%	**0.70%**
credit	75.85%	75.68%	**75.91%**	**76.15%**	**76.00%**	**76.35%**
audit	96.28%	**96.78%**	**96.60%**	**96.29%**	**96.54%**	**96.43%**
soybean	77.59%	73.36%	73.97%	75.43%	75.09%	76.85%

6.4.2 Analysis of the Results

Table 6.4 summarizes the data in these tables, where we reported the number of wins, losses, and ties. By wins, losses, and ties we refer to the number of datasets where *DSB-RF*, accuracy-wise,

TABLE 6.3: Accuracy Comparison Between RF & DSB-RF Type II

	RF	DSB-RF				
Dataset		50%	60%	70%	80%	90%
pasture	40.83%	40.83%	40.83%	40.00%	**41.67%**	40.83%
squash-unstored	60.56%	60.00%	60.00%	**61.11%**	60.00%	**61.11%**
squash-stored	55.56%	55.56%	55.56%	52.22%	54.44%	55.56%
white-clover	63.33%	62.38%	**63.81%**	63.33%	**63.81%**	**63.81%**
glass	12.33%	10.68%	11.10%	12.33%	11.64%	12.19%
breast-cancer	71.65%	**75.98%**	**78.76%**	**77.94%**	**73.61%**	**71.86%**
diabetes	72.76%	**72.84%**	**73.79%**	72.76%	**73.98%**	**73.75%**
vote	97.97%	97.23%	97.09%	97.70%	97.50%	97.77%
car	58.61%	**60.15%**	**60.36%**	**59.97%**	**60.51%**	**61.55%**
vehicle	73.96%	71.88%	72.92%	73.02%	73.54%	73.37%
eucalyptus	19.72%	19.12%	**21.04%**	**23.32%**	**20.96%**	19.48%
sonar	0.28%	0.28%	0.28%	**0.85%**	**0.42%**	0.00%
credit	75.85%	**76.21%**	**76.09%**	**76.47%**	75.82%	**76.38%**
audit	96.28%	**97.18%**	**96.81%**	**96.68%**	**96.65%**	**96.44%**
soybean	77.59%	74.66%	76.47%	75.60%	75.69%	**77.63%**

outperformed, underperformed, and equal performed RF respectively. As shown in Table 6.4, *DSB-RF Type I* has outperformed RF on the majority of the datasets when the percentage of features used is 60% and 70% (best at 70%). On the other hand, *DSB-RF Type II*, was superior to RF when 60%, 70%, 80% and 90% of the features were used (best at 90%). Bar charts summarizing the results of *DSB-RF Type I* and *DSB-RF Type II* are shown in Figures 6.2 and 6.3 respectively.

TABLE 6.4: Performance Comparison of RF & DSB-RF

DSB-RF	**Percentage of Features**	**Wins**	**Losses**	**Ties**
Type I				
	50%	5	10	0
	60%	7	6	2
	70%	8	4	3
	80%	5	7	3
	90%	5	9	1
Type II				
	50%	5	7	3
	60%	7	5	3
	70%	7	5	3
	80%	8	7	0
	90%	8	5	2

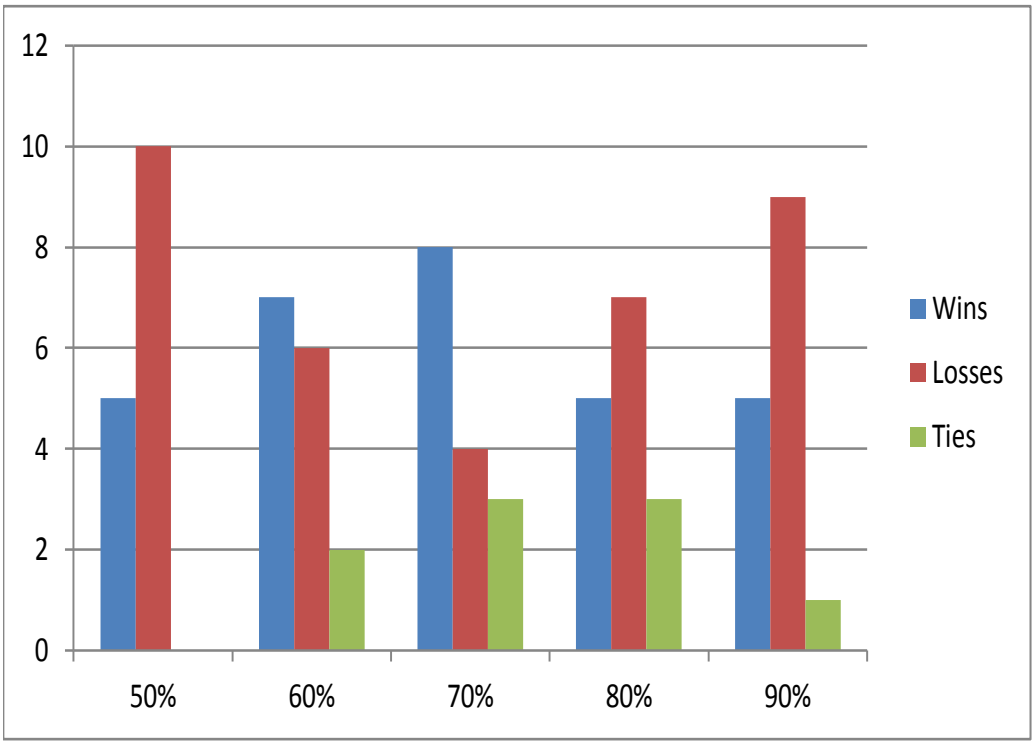


FIGURE 6.2: DSB-DRF Type I Results

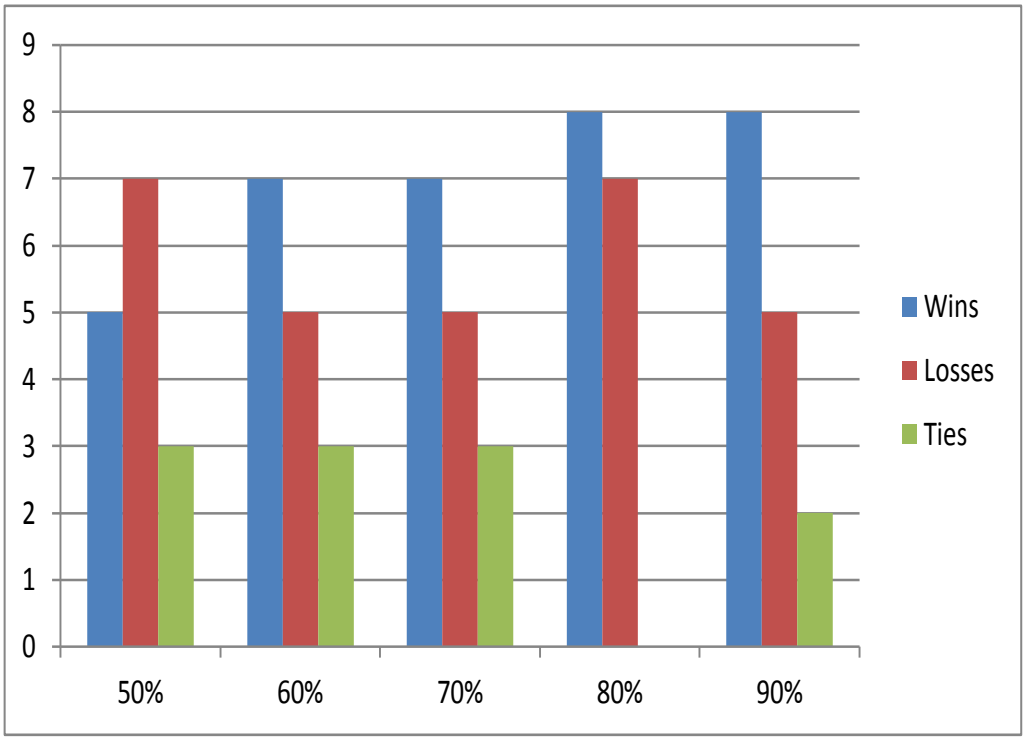


FIGURE 6.3: DSB-DRF Type II Results

6.4.3 Outperformance Range

For the winning datasets, Table 6.5 below depicts the outperformance range of *DSB-RF* over the traditional RF. As shown in the table, *Type I* and *Type II* achieved a maximum outperformance range of 6.59% and 7.11% respectively.

TABLE 6.5: Outperformance Range of RF Over DSB-RF

DSB-RF	Percentage of Features	Range
Type I		
	50%	0.50% - 5.88%
	60%	0.06% - 4.54%
	70%	0.01% - 6.59%
	80%	0.15% - 2.37%
	90%	0.15% - 1.86%
Type II		
	50%	0.08% - 4.33%
	60%	0.24% - 7.11%
	70%	0.40% - 6.29%
	80%	0.14% - 1.96%
	90%	0.04% - 2.94%

6.5 Time Complexity Analysis

Using Big O notation, a time complexity analysis of *DSB-RF* is presented. If n refer to the number of trees in a *DSB-RF*, f refers to the number of features in the dataset, and $O(f)$ refers to the order of traversing the average number of nodes of all trees in a *DSB-RF*, then the time complexity to classify an instance after each tree in a *DSB-RF* casts a vote for its predicted class label is given by $O(n \times O(f))$. It is worth mentioning that since *DSB-RF* does not do any pruning as the previous two enhancements, its time complexity is likely to be comparable to the time complexity of a traditional RF of the same size. This means that *DSB-RF* does not run any faster than the traditional RF. Because *DSB-RF* produces unpruned ensembles, it is considered less suitable than *CLUB-DRF* and *LOFB-DRF* for real-time applications and for machines/devices with limited memory and precessing power.

6.6 Summary

In *DSB-RF*, an additional level of diversity has been injected to improve feature interaction by using weighted subspaces where each subspace contained a fixed random percentage of features. Eventually each subspace mapped to a sub-forest, thus, creating a forest with diverse sub-forests.

As depicted in Table 6.4, *DSB-RF Type I*, with 10 sub-forests and 50 trees each, good results were obtained at 60% and 70% of the features (best at 70%). As for the *DSB-RF Type II*, with 20 sub-forests and 25 trees each, as shown in the same table, good results were obtained at 60%, 70%, 80% and 90% of the features (best at 90%). As an interesting observation, *DSB-RF* has demonstrated promising results especially for the medical datasets (namely *breast-cancer* and *diabetes*) as demonstrated in Tables 6.2 and 6.3. For such datasets, it can be seen that *DSB-RF* has performed consistently well regardless of the percentage of features used.

Due to two main issues that can affect the performance of *DSB-RF*, in the next chapter, RD will be used to overcome these issues. The first issue is that having a large number of correlated features in the subspace (an hence in the corresponding sub-forest) can invalidate the weight (given by Equation 6.3) assigned to the subspace. Therefore, the absence of correlation among features is desirable as it is likely to produce more accurate estimate of the weight.

The second issue is noisy features. A noisy feature is a feature that is not related to the class label to be predicted and hence, can increase the classification error rate. When the training dataset from which we the subspaces are created (refer to the top of Figure 6.1) contains many noisy features, this can result in sub-forests containing only noisy features. Such sub-forests are likely to have poor performance due to high classification error rate, therefore, they can affect the overall performance of *DSB-RF*.

As shall be seen in the next chapter, RD aims at adjusting the size of the sub-forests by growing the well-performing ones and shrinking the poorly-performing ones. Because highly correlated and noisy features are likely to produce poorly-performing sub-forests, RD overcomes these issues by shrinking the sub-forests that suffer from these issues, hence, giving them less weight in the voting process.

Chapter 7

Replicator Dynamics-Based Diverse Random Forest (RDB-DRF)

7.1 Overview

Findings reported in this chapter have been published in Fawgreh et al. (2015). This chapter introduces the second technique related to feature engineering which we have termed *RDB-DRF*. This enhancement aims to further improve the performance of *DSB-RF* introduced in the previous chapter by applying Replicator Dynamics (RD). As subsequently discussed in this chapter, the evolution model of RD will be used to grow the well-performing sub-forests and shrink the poorly-performing ones.

7.2 Background

In the previous chapter, it has been demonstrated how random subspaces can be used to create an RF termed *DSB-RF* with diverse trees to promote the principle that diversity can lead to better performance. This chapter investigates how to further improve the performance of *DSB-RF* using RD in order to resolve two issues that can affect the performance of *DSB-RF*, namely, correlated and noisy features, as previously discussed in the previous chapter. Though RD has been used in evolutionary game theory and other domains as discussed in the next section, to the best of our knowledge, however, it has not been used in RF or any extension of it. Experimental

study on 15 real datasets showed favorable results, demonstrating the potential of the proposed method.

7.3 Replicator Dynamics

Replicator Dynamics (RD) by Schuster and Sigmund (1983) is a simple model of evolution used extensively in evolutionary game theory Taylor and Jonker (1978), Hofbauer and Sigmund (2003), Hilbe (2011), Hauert (2010), Nowak and Sigmund (2004), Hauert et al. (2002), Roca et al. (2009). It provides a convenient way to represent selection among a population of diverse types. To illustrate how it works, assume that selection occurs between periods after dividing time into discrete intervals. The proportion of each type in the next period is given by the replicator equation as a function of the type's payoffs and its current proportion in the population. Types that score above the average payoffs increase in proportion, while types that score below the average payoffs decrease in proportion. The amount of increase or decrease depends on a type's proportion in the current population and on its relative payoffs.

In addition to being used extensively in the domain of evolutionary game theory, to a lesser extent, however, it was also used in other domains. In medicine, Lohmann and Bohn (2002) used RD to analyze fMRI data of the human brain. In mathematical ecology, Bomze (1995) used RD to describe the interaction of two populations over time. In mathematical biology, Hutson and Schmitt (1992) used RD in the study of permanence, which, in a set of populations, studies the long-term survival of each species. Multi-agent learning was another domain of RD Galstyan (2013), Tuyls et al. (2006). Finally, in social networks, Olfati-Saber (2007) used RD to understand origins of social patterns and the dominant behavioral and cultural directions in social networks.

The most general continuous form of RD is given by the differential equation

$$\dot{x_i} = x_i[f_i(x) - \phi(x)] \tag{7.1}$$

such that

$$\phi(x) = \sum_{j=1}^{n} x_j f_j(x) \tag{7.2}$$

where the proportion of type i in the population is given by x_i, the distribution of types vector in the population is given by $x = (x_1, ..., x_n)$, the fitness of type i, which depends on the population, is given by $f_i(x)$, and $\phi(x)$ is the average population fitness which is calculated as the weighted average of the fitness of the n types in the population.

In the next section, we shall see how to apply RD in order to boost the performance of a *DSB-RF* introduced in the previous chapter. To the best of our knowledge, RD has never been used before in ensemble learning.

7.4 Applying Replicator Dynamics to a DSB-RF

Figure 6.1 in Chapter 6 depicts a *DSB-RF*. As shown in the figure, the population is n sub-forests. Since each sub-forest corresponds to a subspace containing a randomized subset of features, it represents, and is analogous to a diverse type. The sub-forest's size is analogous to the proportion of each type, the sub-forest's accuracy is analogous to the type's payoffs, and the average accuracy of the entire forest is analogous to the average payoffs. The discrete periods mentioned in the previous section correspond to loop iterations. At each iteration, the accuracy of the sub-forest being processed is compared with the average accuracy of the entire *DSB-RF* which is calculated as the average accuracy of the sub-forests. If it is found greater than, then the size of the sub-forest grows, and if it is found less than, the size shrinks.

For growing and shrinking a sub-forest, two variations will be used. In the first one, the size grows/shrinks by a fixed number of trees as shown in the following equations:

$$\text{treesToAdd} = \beta \tag{7.3}$$

$$\text{treesToRemove} = \gamma \tag{7.4}$$

It is worth pointing out that growing/shrinking a sub-forest by a fixed number of trees as per this variation is not a standard evolution technique in RD. This variation was meant to act as a form of regularization to allow the sub-forests to slowly grow/shrink in a controlled fashion which may result in better performance.

In the second variation, the sub-forest grows/shrinks by adding/removing a variable number of trees according to the following equations:

$$treesToAdd = \lfloor (subforestAccuracy(i) - ForestAccuracy) \times numTrees \rfloor \tag{7.5}$$

$$treesToRemove = \lfloor (ForestAccuracy - subforestAccuracy(i)) \times numTrees \rfloor \tag{7.6}$$

where *subforestAccuracy(i)* refers to the accuracy of the i^{th} sub-forest being processed on the OOB instances. As mentioned in Section 3.2 in Chapter 3, these are the instances that were not included in the sample with replacement that was used to build the tree, and they account for about 36% of the total number of instances. Using the OOB samples to evaluate a tree gives an unbiased estimate of its predictive accuracy since, unlike training data that was seen by the tree when it was built, OOB data was not seen and therefore, it is a more accurate measure of the tree's predictive accuracy. *numTrees* (given by Equation 6.2 in Chapter 6) refers to the initial number of trees that was used to construct the sub-forest. The *ForestAccuracy* refers to the average accuracy of the entire *DSB-RF* which can be calculated as follows:

$$ForestAccuracy = \frac{1}{Subspaces} \sum_{i=1}^{Subspaces} subforestAccuracy(i) \tag{7.7}$$

where the constant *Subspaces* is given by Equation 6.1 in Chapter 6. Figure 7.1 shows how RD can be applied to a *DSB-RF* to further improve its performance by growing/shrinking the sub-forests. As shown the figure, the process starts by first creating a *DSB-RF* which consists of n sub-forests each containing a randomized subset of features. This *DSB-RF* is given as input to the *RDB-DRF* algorithm described in the next subsection where a predefined number of loop iterations is applied to the input *DSB-RF*. During each iteration, the accuracy of the sub-forest being processed is compared with the accuracy of the entire forest. Depending on whether it is greater/less than the forest's overall accuracy, trees in the sub-forest are grown/shrunk. The end result is a new forest which we have termed *RDB-DRF* with grown/shrunk sub-forests.

7.4.1 RDB-DRF Algorithm

Algorithm 6 outlines the main steps involved in applying RD to a *DSB-RF*, which is given as input to the algorithm. The other input *numIterates* refers to the desired number of loop iterations.

The outer **for** loop iterates over the number of iterations given by *numIterates*, and the inner **for** loop iterates over each sub-forest in the input *DSB-RF*. During each iteration in the inner loop, the accuracy of each sub-forest is compared with the accuracy of *DSB-RF*. If it is found greater than, then Equation 7.3 or Equation 7.5 is used to determine how many trees to add to the sub-forest. If it is found less than, then Equation 7.4 or Equation 7.6 is used to determine how many trees to remove from the sub-forest. Since a sub-forest is grown when its performance outperforms the performance of the entire forest, and is shrunk otherwise, the rationale here is that by growing the well-performing sub-forests, more weight is given to them during the voting process, and less weight is given to the poorly-performing sub-forests by shrinking them. This weight adjustment is likely to yield correct classification of the instance being processed.

As shown in the *Output* section of the algorithm, we refer to the resulted *DSB-RF* forest after growing/shrinking it as *RDB-DRF*.

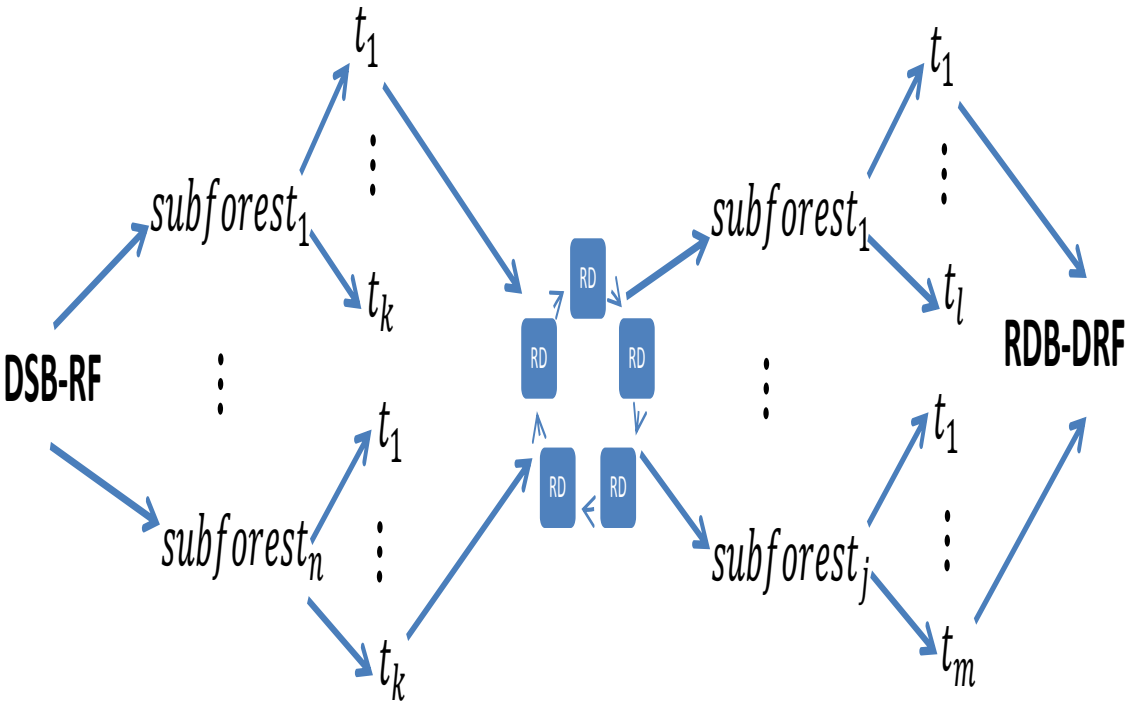


FIGURE 7.1: Producing RDB-DRF from DSB-RF by Applying Replicator Dynamics

Algorithm 6 Applying Replicator Dynamics to a DSB-RF

{User Settings}
input *DSB-RF*,*numIterates*
{Process}
for $i = 1 \rightarrow numIterates$ **do**
  for $j = 1 \rightarrow DSB\text{-}RF.numberofSubforests()$ **do**
    forestAccuracy ← calculateForestAccuracy(*DSB-RF*)
    subforestAccuracy ← calculateSubforestAccuracy(*DSB-RF*.subforest(j))
    if $(subforestAccuracy > forestAccuracy)$ **then**
      Apply Equation 7.3 or Equation 7.5 to determine treestoAdd
      addTrees(*DSB-RF*.subforest(j), treestoAdd)
    else if $(subforestAccuracy < forestAccuracy)$ **then**
      Apply Equation 7.4 or Equation 7.6 to determine treestoRemove
      removeTrees(*DSB-RF*.subforest(j), treestoRemove)
    end if
  end for
end for
{Output}
DSB-RF with grown or shrunk sub-forests (we refer to this forest as *RDB-DRF*)

7.5 Experimental Study

Since the two types of *DSB-RFs* in the previous chapter (refer to *Type I* and *Type II* in Section 6.4) produced favorable results, we decided to use them in our experiments in this chapter. As a refresher, *Type I* refers to 10 sub-forests with 50 trees each, and *Type II* refers to 20 sub-forests with 25 trees each. We used a random 70% of the features for each subspace in both types. We chose 70% because in Table 6.4 (Chapter 6), this percentage of features was a common denominator as it performed well in both types (*Type I* and *Type II*) of *DSB-RF* used in the experiments.

As for the number of iterations (refer to *numIterates* in Algorithm 6 above), we have used various numbers of iterations including 25, 50, 100, 150, and 1000 iterations. By taking into account the 2 types of *DSB-RF*s mentioned above, and the various numbers of iterations, Table 7.1 lists the different experimental scenarios that will be addressed later in this chapter.

TABLE 7.1: Scenario Reference Table

Scenario#	# of sub-forests	# of Trees Per sub-forest	# of Iterations
1	10	50	25
2	10	50	50
3	10	50	100
4	10	50	150
5	10	50	1000
6	20	25	25
7	20	25	50
8	20	25	100
9	20	25	150
10	20	25	1000

7.5.1 Experiments Details

Four experiments sets were carried out. In the first and second set of experiments, we used the first variation given by Equations 7.3 and 7.4 above. Here we chose $\beta = \gamma = 1$, therefore, only 1 tree may be added/removed in each iteration. In the first experiments set, we compared the performance of the resulted *RDB-DRF* with the initial *DSB-RF* to see whether there is a performance gain or not. In the second experiments set, performance was compared with traditional RFs of identical size (500 trees). Similarly, the third and fourth set of experiments were performed using the second variation where a variable number of the initial number of trees in the sub-forest is added or removed as per Equations 7.5 and 7.6 above. In the third experiments set, performance of the resulted *RDB-DRF* was compared with the initial *DSB-RF* to see whether there is a performance gain or not, and in the fourth experiments set, performance was compared with traditional RFs of identical size.

In each set of experiments, a new experiment was performed for each scenario listed in Table 7.1 where a new *DSB-RF* was created after applying Algorithm 6 above as per the parameters specified for each scenario.

7.5.2 Results

For the first experiments set, Tables 7.2 and 7.3 show the accuracy of the winning and losing datasets respectively. Performance in these tables is compared between *RDB-DRF* and the initial *DSB-RF*. The *Difference* column in both tables shows the difference of accuracy where the smallest difference is underlined and the largest difference is displayed in boldface.

Tables 7.4 and 7.5 show the accuracy of the winning and losing datasets respectively for the second experiments set. Performance in these tables is compared between *RDB-DRF* and a traditional RF of identical size.

Similarly, Tables 7.6 and 7.7 show the results of the third experiments set, and Tables 7.8 and 7.9 show the results of the fourth experiments set.

7.5.3 Analysis

Using the information in Tables 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, and 7.3, we recorded the number of wins, losses, and ties. By wins, losses, and ties we refer to the number of datasets where *RDB-DRF*, accuracy-wise, outperformed, underperformed, and equal performed either the initial *DSB-RF* (first and third experiments sets), or the traditional RF (second and fourth experiments sets) respectively.

Results for the first experiments set in terms of wins, losses, and ties are shown in the bar chart in Figure 7.2. As reflected in the chart, the number of wins (where *RDB-DRF* outperformed the initial *DSB-RF*) exceeded the number of losses in all of the scenarios except Scenario 10 (wins equal losses), and Scenario 2 (losses exceed wins by 1 dataset only).

TABLE 7.2: First Experiments Set Winning Datasets Accuracy

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
1	diabetes	72.41%	73.18%	0.77%
	vehicle	73.61%	75.00%	1.39%
	eucalyptus	21.20%	26.40%	**5.20%**
	credit	75.29%	76.18%	0.89%
	soybean	73.71%	76.29%	2.58%
2	squash-unstored	55.56%	61.11%	**5.60%**
	eucalyptus	21.60%	24.00%	2.40%
	credit	74.12%	76.18%	2.06%

Continued on next page

Table 7.2 – continued from previous page

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
3	squash-unstored	61.11%	66.67%	5.56%
	glass	10.96%	12.33%	1.37%
	diabetes	74.33%	74.71%	0.38%
	vehicle	70.83%	72.57%	1.74%
	eucalyptus	18.00%	24.00%	**6.00%**
	sonar	0.00%	1.41%	1.41%
4	squash-unstored	61.11%	72.22%	**11.11%**
	squash-stored	44.44%	50.00%	5.56%
	white-clover	61.90%	66.67%	4.77%
	vehicle	72.57%	72.92%	0.35%
	eucalyptus	20.80%	26.40%	5.60%
	credit	77.94%	78.82%	0.88%
5	squash-unstored	50.00%	61.11%	**11.11%**
	white-clover	61.90%	66.67%	4.77%
	vehicle	72.92%	73.26%	0.34%
	eucalyptus	17.20%	24.00%	6.80%
	soybean	73.28%	75.43%	2.15%
6	squash-unstored	61.11%	66.67%	**5.56%**
	diabetes	73.56%	75.10%	1.54%
	vehicle	72.22%	73.26%	1.04%
	eucalyptus	22.00%	26.40%	4.40%
	credit	75.29%	76.18%	0.89%
	audit	96.47%	96.76%	0.29%
7	squash-unstored	61.11%	66.67%	**5.56%**
	diabetes	73.56%	73.95%	0.39%
	vehicle	73.26%	73.61%	0.35%
	eucalyptus	28.00%	29.20%	1.20%
	audit	97.06%	97.21%	0.15%
8	squash-unstored	61.11%	77.78%	**16.67%**
	vote	96.62%	97.30%	0.68%
	vehicle	72.92%	74.65%	1.73%
	eucalyptus	23.60%	29.60%	6.00%
	audit	96.76%	96.91%	0.15%
	soybean	77.59%	78.88%	1.29%
9	squash-unstored	61.11%	77.78%	**16.67%**
	vehicle	72.57%	73.96%	1.39%
	eucalyptus	28.00%	28.80%	0.80%
	sonar	0.00%	1.41%	1.41%
	audit	96.62%	97.06%	0.44%
	soybean	73.71%	85.78%	12.07%
10	pasture	33.33%	41.67%	8.34%
	squash-unstored	61.11%	72.22%	**11.11%**
	eucalyptus	23.60%	30.40%	6.80%
	soybean	77.16%	79.74%	2.58%

TABLE 7.3: First Experiments Set Losing Datasets Accuracy

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
1	breast-cancer	77.32%	75.26%	**2.06%**
2	breast-cancer	78.35%	74.23%	**4.12%**
	diabetes	73.95%	73.56%	0.39%
	sonar	2.82%	0.00%	2.82%
	soybean	79.74%	78.02%	1.72%
3	breast-cancer	78.35%	71.13%	**7.22%**
	vote	96.62%	95.95%	0.67%
	soybean	77.59%	77.16%	0.43%
4	breast-cancer	79.38%	75.26%	**4.12%**
	vote	97.30%	96.62%	0.68%
5	breast-cancer	76.29%	74.23%	2.06%
	diabetes	74.71%	71.65%	**3.15%**

Continued on next page

Table 7.3 – continued from previous page

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
	credit	76.76%	74.71%	2.05%
6	breast-cancer	81.44%	72.16%	**9.28%**
	sonar	1.41%	0.00%	1.41%
	soybean	76.72%	76.29%	0.43%
7	breast-cancer	79.38%	74.23%	**5.15%**
	vote	97.97%	96.62%	1.35%
	sonar	1.41%	0.00%	1.41%
	credit	77.35%	76.47%	0.88%
8	breast-cancer	77.32%	72.16%	**5.16%**
	diabetes	73.18%	72.80%	0.38%
	sonar	2.82%	0.00%	2.82%
	credit	76.18%	74.71%	1.47%
9	breast-cancer	77.32%	74.23%	**3.09%**
	diabetes	73.18%	72.80%	0.38%
	vote	97.97%	97.30%	0.67%
	credit	77.35%	75.88%	1.47%
10	breast-cancer	76.29%	72.16%	**4.13%**
	diabetes	75.48%	74.71%	0.77%
	vote	97.97%	97.30%	0.67%
	credit	77.65%	76.76%	0.89%

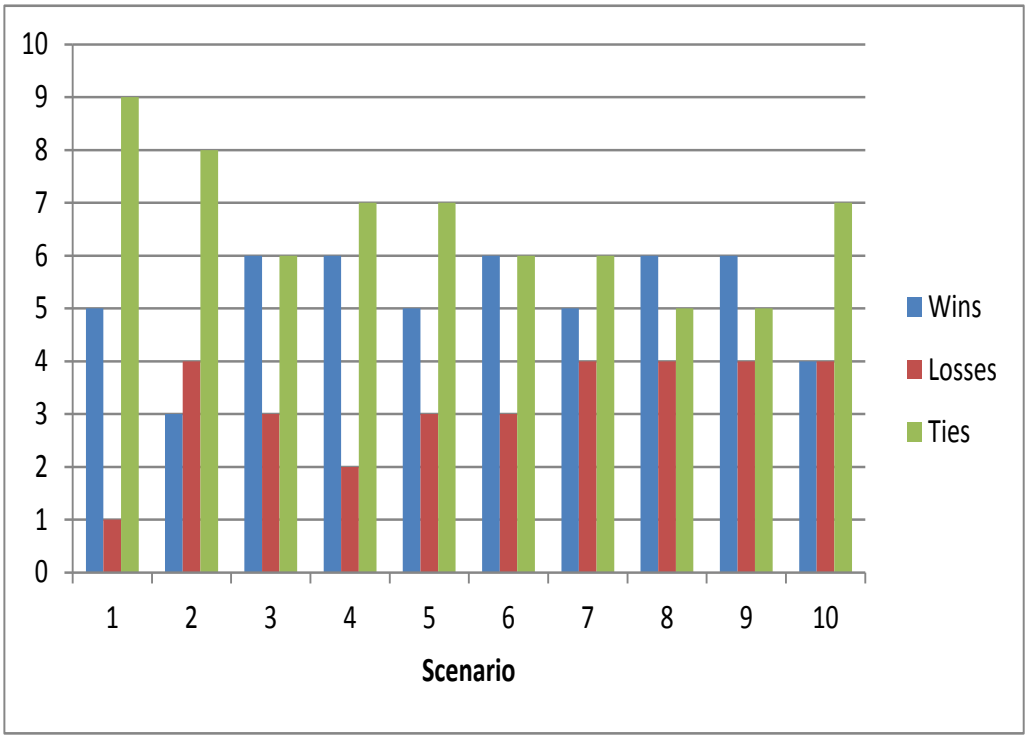


FIGURE 7.2: First Experiments Set Results Bar Chart

Similarly, the bar chart in Figure 7.3 shows the results for the second experiments set. Once again and as reflected in the chart, the number of wins (where *RDB–DRF* outperformed a traditional RF of identical size) exceeded the number of losses in all of the scenarios except for Scenarios 8 and 5 (losses exceed wins by 1 and 2 datasets respectively).

TABLE 7.4: Second Experiments Set Winning Datasets Accuracy

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
1	breast-cancer	72.16%	75.26%	3.10%
	vehicle	72.92%	75.00%	2.08%
	eucalyptus	21.20%	26.40%	**5.20%**
	sonar	0.00%	1.41%	1.41%
	credit	75.88%	76.18%	0.30%
	audit	96.32%	96.76%	0.44%
	soybean	75.86%	76.29%	0.43%
2	breast-cancer	72.16%	74.23%	2.07%
	eucalyptus	21.20%	24.00%	**2.80%**
	credit	75.88%	76.18%	0.30%
	audit	96.32%	97.06%	0.74%
	soybean	75.86%	78.02%	2.16%
3	squash-unstored	61.11%	66.67%	**5.56%**
	diabetes	73.56%	74.71%	1.15%
	eucalyptus	21.20%	24.00%	2.80%
	sonar	0.00%	1.41%	1.41%
	credit	75.88%	76.76%	0.88%
	audit	96.32%	97.06%	0.74%
	soybean	75.86%	77.16%	1.30%
4	squash-unstored	61.11%	72.22%	**11.11%**
	breast-cancer	72.16%	75.26%	3.10%
	eucalyptus	21.20%	26.40%	5.20%
	credit	75.88%	78.82%	2.94%
	audit	96.32%	96.91%	0.59%
	soybean	75.86%	78.02%	2.16%
5	breast-cancer	72.16%	74.23%	2.07%
	vehicle	72.92%	73.26%	0.34%
	eucalyptus	21.20%	24.00%	**2.80%**
	audit	96.32%	97.21%	0.89%
6	squash-unstored	61.11%	66.67%	**5.56%**
	diabetes	73.56%	75.10%	1.54%
	vehicle	72.92%	73.26%	0.34%
	eucalyptus	21.20%	26.40%	5.20%
	credit	75.88%	76.18%	0.30%
	audit	96.32%	96.76%	0.44%
	soybean	75.86%	76.29%	0.43%
7	squash-unstored	61.11%	66.67%	5.56%
	breast-cancer	72.16%	74.23%	2.07%
	diabetes	73.56%	73.95%	0.39%
	vehicle	72.92%	73.61%	0.69%
	eucalyptus	21.20%	29.20%	**8.00%**
	credit	75.88%	76.47%	0.59%
	audit	96.32%	97.21%	0.89%
8	squash-unstored	61.11%	77.78%	**16.67%**
	vehicle	72.92%	74.65%	1.73%
	eucalyptus	21.20%	29.60%	8.40%
	audit	96.32%	96.91%	0.59%
	soybean	75.86%	78.88%	3.02%
9	squash-unstored	61.11%	77.78%	**16.67%**
	breast-cancer	72.16%	74.23%	2.07%
	vehicle	72.92%	73.96%	1.04%
	eucalyptus	21.20%	28.80%	7.60%
	sonar	0.00%	1.41%	1.41%
	audit	96.32%	97.06%	0.74%
	soybean	75.86%	85.78%	9.92%
10	squash-unstored	61.11%	72.22%	**11.11%**
	diabetes	73.56%	74.71%	1.15%
	vehicle	72.92%	73.96%	1.04%
	eucalyptus	21.20%	30.40%	9.20%
	sonar	0.00%	1.41%	1.41%

Continued on next page

Table 7.4 – continued from previous page

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
	credit	75.88%	76.76%	0.88%
	audit	96.32%	96.91%	0.59%
	soybean	75.86%	79.74%	3.88%

TABLE 7.5: Second Experiments Set Losing Datasets Accuracy

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
1	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	73.18%	0.38%
	vote	97.97%	96.62%	1.35%
	car	61.90%	60.37%	**1.53%**
2	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	vote	97.97%	96.62%	1.35%
	car	61.90%	60.37%	1.53%
3	white-clover	66.67%	61.90%	**4.77%**
	breast-cancer	72.16%	71.13%	1.03%
	vote	97.97%	95.95%	2.02%
	car	61.90%	60.54%	1.36%
	vehicle	72.92%	72.57%	0.35%
4	squash-stored	55.56%	50.00%	**5.56%**
	diabetes	73.56%	73.18%	0.38%
	vote	97.97%	96.62%	1.35%
	car	61.90%	60.20%	1.70
5	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	71.65%	1.91%
	vote	97.97%	96.62%	1.35%
	car	61.90%	59.52%	**2.38%**
	credit	75.88%	74.71%	1.17%
	soybean	75.86%	75.43%	0.43%
6	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	vote	97.97%	97.30%	0.67%
	car	61.90%	60.37%	1.53
7	white-clover	66.67%	61.90%	**4.77%**
	vote	97.97%	96.62%	1.35%
	car	61.90%	59.69%	2.21%
	soybean	75.86%	72.84%	3.02%
8	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	72.80%	0.76%
	vote	97.97%	97.30%	0.67%
	car	61.90%	59.86%	2.04%
	credit	75.88%	74.71%	1.17%
9	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	72.80%	0.76%
	vote	97.97%	97.30%	0.67%
	car	61.90%	60.03%	**1.87%**
10	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	vote	97.97%	97.30%	0.67%
	car	61.90%	60.20%	1.70%

For the third and fourth experiments set, results are shown in the bar charts in Figures 7.4 and 7.5 respectively. It is clear from both figures that these experiment sets produced less favorable

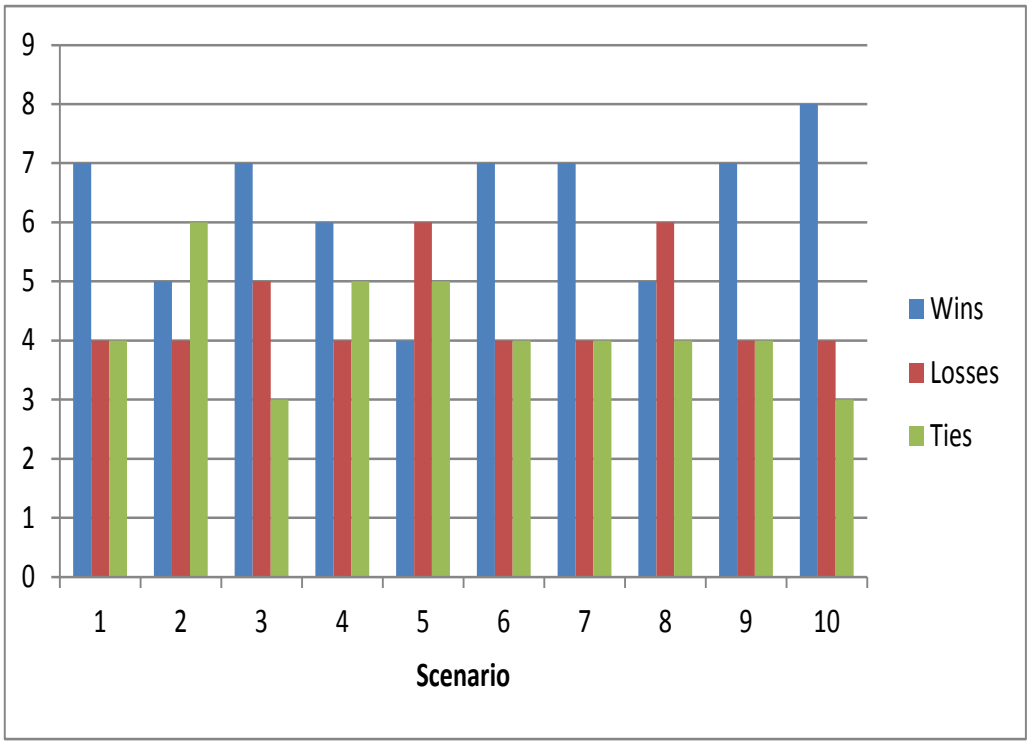


FIGURE 7.3: Second Experiments Set Results Bar Chart

results than the previous experiment sets as the number of wins exceeded the number of losses in 5 scenarios only (third experiments set) and 4 scenarios only (fourth experiments set).

TABLE 7.6: Third Experiments Set Winning Datasets Accuracy

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
1	squash-unstored	55.56%	61.11%	5.55%
	squash-stored	50.00%	55.56%	**5.56%**
	eucalyptus	19.20%	19.60%	0.40%
	audit	96.03%	97.06%	1.03%
2	squash-unstored	61.11%	77.78%	**16.67%**
	vehicle	72.22%	73.26%	1.04%
	eucalyptus	24.80%	28.40%	3.60%
	soybean	72.41%	75.00%	2.59%
3	squash-unstored	50.00%	55.56%	**5.56%**
	eucalyptus	16.40%	16.80%	0.40%
	soybean	78.02%	80.17%	2.15%
4	squash-unstored	66.67%	77.78%	11.11%
	squash-stored	50.00%	55.56%	5.56%
	vote	96.62%	97.30%	0.68%
	vehicle	70.83%	72.92%	2.09%
	eucalyptus	20.00%	34.00%	**14.00%**
	soybean	72.41%	73.28%	0.87%
5	squash-unstored	61.11%	66.67%	5.56%
	white-clover	61.90%	66.67%	4.77%
	eucalyptus	17.60%	27.20%	**9.60%**
	soybean	77.59%	79.74%	2.15%
6	squash-unstored	61.11%	66.67%	**5.56%**
	eucalyptus	22.80%	24.00%	1.20%
	audit	96.32%	96.76%	0.44%

Continued on next page

Table 7.6 – continued from previous page

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
7	diabetes	73.18%	74.33%	1.15%
	vehicle	71.88%	72.92%	1.04%
	eucalyptus	22.80%	25.20%	**2.40%**
	audit	96.76%	96.91%	0.15%
8	eucalyptus	22.00%	27.20%	**5.20%**
	audit	96.32%	96.91%	0.59%
9	squash-unstored	61.11%	66.67%	**5.56%**
	eucalyptus	19.60%	24.00%	4.40%
	audit	96.62%	96.76%	0.14%
10	squash-unstored	55.56%	66.67%	**11.11%**
	eucalyptus	24.40%	26.80%	2.40%
	audit	96.62%	96.76%	0.14%
	soybean	77.59%	86.21%	8.62%

TABLE 7.7: Third Experiments Set Losing Datasets Accuracy

Scenario	Dataset	Accuracy DSB-RF	Accuracy RDB-DRF	Difference
1	pasture	41.67%	33.33%	**8.34%**
	breast-cancer	81.44%	76.29%	5.15%
	vote	97.97%	97.30%	0.67%
	credit	75.88%	75.59%	0.29%
	soybean	78.02%	75.00%	3.02%
2	breast-cancer	77.32%	76.29%	**1.03%**
	credit	76.18%	75.88%	0.30%
3	breast-cancer	78.35%	73.20%	**5.15%**
	diabetes	75.10%	73.95%	1.15%
	credit	74.12%	73.82%	0.30%
4	breast-cancer	79.38%	69.07%	**10.31%**
	diabetes	75.48%	73.18%	2.30%
	credit	76.76%	76.47%	0.29%
5	breast-cancer	75.26%	72.16%	**3.10%**
	diabetes	75.10%	73.56%	1.54%
	credit	77.35%	76.47%	0.88%
6	breast-cancer	80.41%	73.20%	**7.21%**
	diabetes	74.33%	73.56%	0.77%
	soybean	71.55%	70.26%	1.29%
7	breast-cancer	77.32%	72.16%	**5.16%**
	soybean	72.84%	71.98%	0.86%
8	breast-cancer	75.26%	71.13%	**4.13%**
	diabetes	73.18%	72.80%	0.38%
	soybean	77.16%	76.72%	0.44%
9	diabetes	73.56%	72.41%	1.15%
	vote	97.97%	96.62%	**1.35%**
	soybean	77.59%	76.72%	0.87%
10	breast-cancer	79.38%	75.26%	**4.12%**
	diabetes	75.48%	73.95%	1.53%

TABLE 7.8: Fourth Experiments Set Winning Datasets Accuracy

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
1	breast-cancer	72.16%	76.29%	**4.13%**
	diabetes	73.56%	74.71%	1.15%
	car	61.90%	64.29%	2.39%
	vehicle	72.92%	73.61%	0.69%
	audit	96.32%	97.06%	0.74%
2	squash-unstored	61.11%	77.78%	**16.67%**
	breast-cancer	72.16%	76.29%	4.13%
	vehicle	72.92%	73.26%	0.34%

Continued on next page

Table 7.8 – continued from previous page

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
	eucalyptus	21.20%	28.40%	7.20%
	audit	96.32%	96.91%	0.59%
3	breast-cancer	72.16%	73.20%	1.04%
	diabetes	73.56%	73.95%	0.39%
	sonar	0.00%	1.41%	1.41%
	audit	96.32%	97.06%	0.74%
	soybean	75.86%	80.17%	**4.31%**
4	squash-unstored	61.11%	77.78%	**16.67%**
	eucalyptus	20.00%	34.00%	14.00%
	sonar	0.00%	1.41%	1.41%
	credit	75.88%	76.47%	0.59%
	audit	96.32%	97.21%	0.89%
5	squash-unstored	61.11%	66.67%	5.56%
	vehicle	72.92%	73.26%	0.34%
	eucalyptus	21.20%	27.20%	**6.00%**
	credit	75.88%	76.47%	0.59%
	audit	96.32%	96.91%	0.59%
	soybean	75.86%	79.74%	3.88%
6	squash-unstored	61.11%	66.67%	**5.56%**
	breast-cancer	72.16%	73.20%	1.04%
	eucalyptus	21.20%	24.00%	2.80%
	audit	96.32%	96.76%	0.44%
7	diabetes	73.56%	74.33%	0.77%
	eucalyptus	21.20%	25.20%	**4.00%**
	credit	75.88%	76.76%	0.88%
	audit	96.32%	96.91%	0.59%
8	squash-unstored	61.11%	66.67%	5.56%
	eucalyptus	21.20%	27.20%	**6.00%**
	audit	96.32%	96.91%	0.59%
	soybean	75.86%	76.72%	0.86%
9	squash-unstored	61.11%	66.67%	**5.56%**
	breast-cancer	72.16%	74.23%	2.07%
	car	61.90%	62.59%	0.69%
	eucalyptus	21.20%	24.00%	2.80%
	audit	96.32%	96.76%	0.44%
	soybean	75.86%	76.72%	0.86%
10	squash-unstored	61.11%	66.67%	5.56%
	breast-cancer	61.11%	75.26%	**14.15%**
	diabetes	73.56%	73.95%	0.39%
	eucalyptus	21.20%	26.80%	5.60%
	audit	96.32%	96.76%	0.44%
	soybean	75.86%	86.21%	10.35%

TABLE 7.9: Fourth Experiments Set Losing Datasets Accuracy

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
1	pasture	41.67%	33.33%	8.34%
	white-clover	66.67%	57.14%	**9.53%**
	vote	97.97%	97.30%	0.67%
	eucalyptus	21.20%	19.60%	1.60%
	credit	75.88%	75.59%	0.29%
	soybean	75.86%	75.00%	0.86%
2	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	72.80%	0.76%
	vote	97.97%	96.62%	1.35%
	car	61.90%	60.20%	1.70%
	soybean	75.86%	75.00%	0.86%
3	squash-unstored	61.11%	55.56%	**5.55%**
	white-clover	66.67%	61.90%	4.77%

Continued on next page

Table 7.9 – continued from previous page

Scenario	Dataset	Accuracy Traditional RF	Accuracy RDB-DRF	Difference
	vote	97.97%	97.30%	0.67%
	car	61.90%	61.39%	0.51%
	vehicle	72.92%	72.57%	0.35%
	eucalyptus	21.20%	16.80%	4.40%
	credit	75.88%	73.82%	2.06%
4	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	breast-cancer	72.16%	69.07%	3.09%
	diabetes	73.56%	73.18%	0.38%
	vote	97.97%	97.30%	0.67%
	car	61.90%	59.86%	2.04%
	soybean	75.86%	73.28%	2.58%
5	glass	12.33%	10.96%	1.37%
	vote	97.97%	97.30%	0.67%
	car	61.90%	58.84%	**3.06%**
6	vote	97.97%	97.30%	0.67%
	car	61.90%	59.86%	2.04%
	soybean	75.86%	70.26%	**5.60%**
7	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	vote	97.97%	97.30%	0.67%
	car	61.90%	60.54%	1.36%
	soybean	75.86%	71.98%	3.88%
8	glass	12.33%	10.96%	**1.37%**
	breast-cancer	72.16%	71.13%	1.03%
	diabetes	73.56%	72.80%	0.76%
	car	61.90%	60.54%	1.36%
9	white-clover	66.67%	61.90%	**4.77%**
	glass	12.33%	10.96%	1.37%
	diabetes	73.56%	72.41%	1.15%
	vote	97.97%	96.62%	1.35%
10	white-clover	66.67%	61.90%	**4.76%**
	glass	12.33%	10.96%	1.37%
	car	61.90%	59.69%	2.21%
	vehicle	72.92%	72.57%	0.35%
	credit	75.88%	75.29%	0.59%

7.5.4 Outperformance Range

To show how well *RDB-DRF* performed relative to *DSB-RF* (first and third experiment sets), and relative to traditional RFs (second and fourth experiment sets), in this subsection, we show the outperformance range of *RDB-DRF* over *DSB-RF* and traditional RF. For the first, second, third, and fourth experiment sets, such a range is displayed in Tables 7.10, 7.11, 7.12, and 7.13 respectively.

By taking a closer look at these tables, note that a maximum outperformance range of 16.67% has been achieved in each table. Specifically speaking, Scenarios 8 and 9 in Tables 7.10 and 7.11, Scenario 2 in Table 7.12, and Scenarios 2 and 4 in Table 7.13, have all achieved this maximum outperformance range. As per Table 7.1, both Scenarios 8 and 9 refer to 20 sub-forests with 25 trees each, but 100 iterations for Scenario 8, and 150 iterations for Scenario 9. Scenarios 2

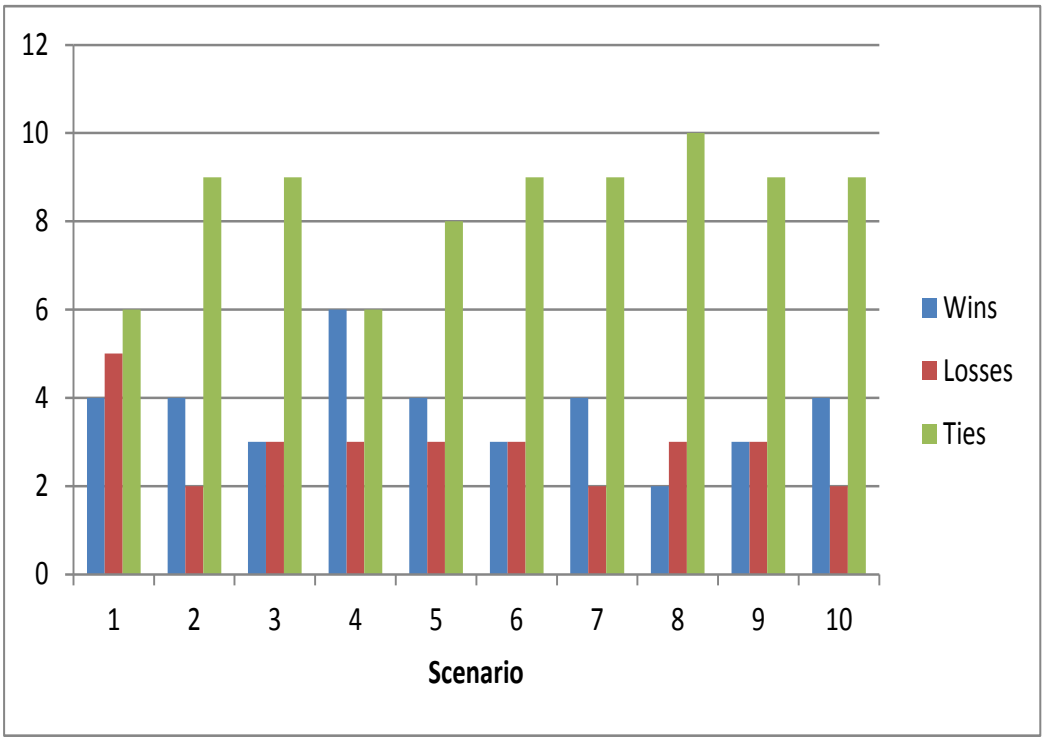


FIGURE 7.4: Third Experiments Set Results Bar Chart

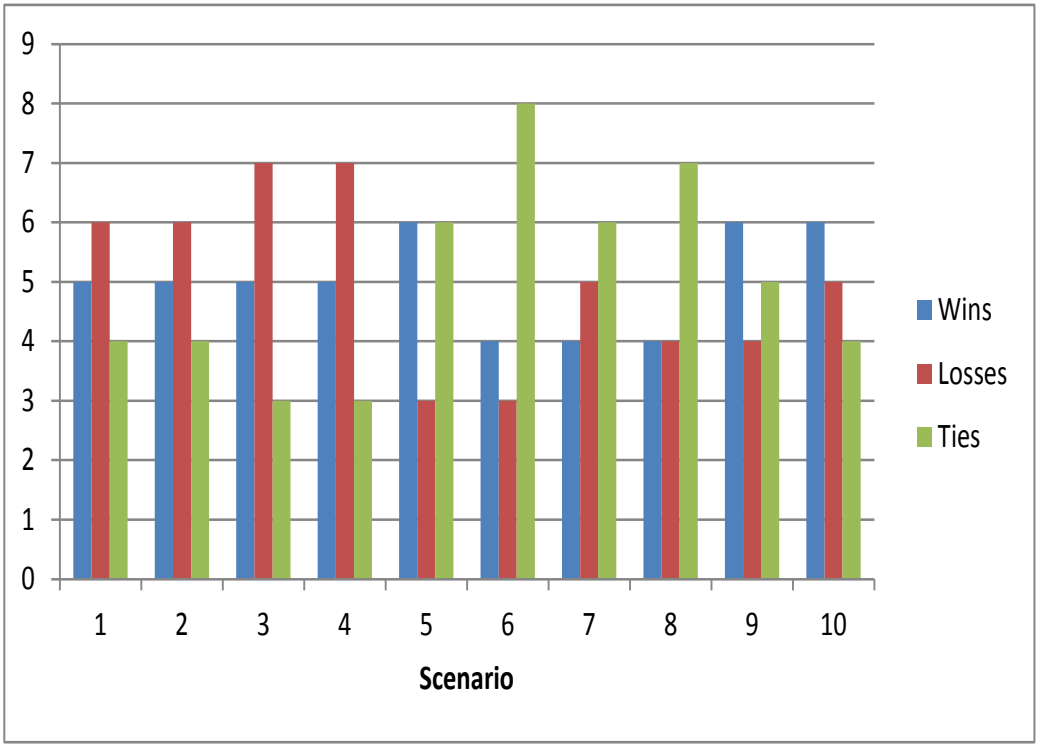


FIGURE 7.5: Fourth Experiments Set Results Bar Chart

and 4 both refer to 10 sub-forests with 50 trees each, but 50 iterations for Scenario 2, and 150 iterations for Scenario 4.

Table 7.10: Outperformance Range of RDB-DRF over DSB-RF (1^{st} Experiments Set)

Scenario	Range
1	0.77% - 5.20%
2	2.06% - 5.60%
3	0.38% - 6.00%
4	0.35% - 11.11%
5	0.34% - 11.11%
6	0.29% - 5.56%
7	0.15% - 5.56%
8	0.15% - 16.67%
9	0.44% - 16.67%
10	2.58% - 8.34%

Table 7.11: Outperformance Range of RDB-DRF over Traditional RF (2^{nd} Experiments Set)

Scenario	Range
1	0.30% - 5.20%
2	0.30% - 2.80%
3	0.74% - 5.56%
4	0.59% - 11.11%
5	0.34% - 2.80%
6	0.30% - 5.56%
7	0.39% - 8.00%
8	0.59% - 16.67%
9	0.74% - 16.67%
10	0.59% - 11.11%

Table 7.12: Outperformance Range of RDB-DRF over DSB-RF (3^{rd} Experiments Set)

Scenario	Range
1	0.40% - 5.56%
2	1.04% - 16.67%
3	0.40% - 5.56%
4	0.68% - 14.00%
5	2.15% - 9.60%
6	0.44% - 5.56%
7	0.15% - 2.40%
8	0.59% - 5.20%
9	0.14% - 5.56%
10	0.14% - 11.11%

TABLE 7.13: Outperformance Range of RDB-DRF over Traditional RF (4^{th} Experiments Set)

Scenario	Range
1	0.69% - 4.13%
2	0.34% - 16.67%
3	0.39% - 4.31%
4	0.59% - 16.67%
5	0.34% - 6.00%
6	0.44% - 5.56%
7	0.59% - 4.00%
8	0.59% - 6.00%
9	0.44% - 5.56%
10	0.39% - 14.15%

7.6 Time Complexity Analysis

Using Big O notation, a time complexity analysis of *RDB-DRF* is presented. If n refer to the number of trees in *RDB-DRF*, f refers to the number of features in the dataset, and $O(f)$ refers to the order of traversing the average number of nodes of all trees in a *RDB-DRF*, then the time complexity to classify an instance after each tree in a *RDB-DRF* casts a vote for its predicted class label is given by $O(n \times O(f))$. Since the sub-forests can grow/shrink, it is worth mentioning that a *RDB-DRF* can be faster, slower, or simlar to that of the initial *DSB-RF*. Therefore, like *DSB-RF*, *RDB-DRF* may not be a good choice for real-time applications and for machines/devices with limited memory and precessing power. For such applications and devices, we recommend the *CLUB-DRF* and *LOFB-DRF* enhancements developed in Chapters 4 and 5 respectively.

7.7 Summary

For the first time ever, we have used Replicator Dynamics to improve the performance of *DSB-RF*, which we have introduced in the previous chapter. We have termed the new method *RDB-DRF*. Favorable results were obtained as demonstrated by the first and second experiments sets whose results were displayed in Figures 7.2 and 7.3 respectively. In such experiments, the first variation given by Equations 7.3 and 7.4 was used. Since we chose $\beta = \gamma = 1$, only 1 tree may be added/removed in each iteration. Taking a closer look at these figures, note that the number of wins exceeds the number of losses in 80% of the scenarios. Such results proved that the first variation given by Equations 7.3 and 7.4, which meant to act as a regularization technique as previously discussed, produced favorable results.

The third and fourth experiments sets, that were based on the second variation given by Equations 7.5 and 7.6, produced less favorable results as demonstrated in Figures 7.4 and 7.5 respectively. Hence, for future experiments, we do recommend the first variation. As demonstrated in Tables 7.10, 7.11, 7.12, and 7.13, all experiments sets achieved a maximum outperformance range of 16.67%.

Part IV

Final Remarks

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Research conducted in this dissertation was motivated by how diversity in ensembles tends to yield better results Kuncheva and Whitaker (2003), Brown et al. (2005), Adeva et al. (2005), Tang et al. (2006). As such, this dissertation investigated new diversity-based techniques that have not been used before to produce more accurate Random Forests ensembles. These techniques are clustering, the Local Outlier Factor (LOF) , weighted random subspaces, and Replicator Dynamics (RD). These techniques were used to produce the enhancements *CLUB-DRF*, *LOFB-DRF*, *DSB-RF*, and *RDB-DRF* respectively.

Clustering is a known diversity technique as data is grouped into clusters (groups) where each cluster contains members that are similar to each other, but are dissimilar to other members in other clusters. When a member is selected from each cluster, the end result is a set of members that is diverse. When clustering is applied to RFs, it can produce pruned ensembles that are much smaller in size than the original ones. This happens when trees are grouped into clusters, and a representative tree from each cluster is selected. Depending on the number of trees in the RF, and how many clusters are created, clustering can be used as a technique to extreme prune RFs, as it is likely to achieve extreme pruning levels. Similarly, the LOF can be used as a diversity technique by assigning each tree an LOF score, and selecting the top k trees with the highest weighted LOF score. Extreme pruning levels can be achieved when the number of trees in the RF is large and k is small.

As a feature engineering technique, weighted random subspaces were used to inject another level of diversity in RF trees. This was done by creating random subspaces where each subspace contained a random fixed subset of features, and assigning a weight to each subspace according

to its Absolute Predictive Power (APP) given by Cuzzocrea et al. (2013). Each subspace was used to create a sub-forest that contains a random subset of features, and when taken collectively, a diversified RF was formed which we have termed *DSB-RF*. To resolve 2 issues that can affect the performance of a *DSB-RF*, namely noisy and correlated features, RD was applied to further improve its performance. This was done by growing the well-performing sub-forests and shrinking the poorly-performing ones. By doing this, at voting time, more weight is given to the well-performing sub-forests and less weight is given to the poorly-performing ones.

We have been able to fulfill the aims and objectives that were outlined in Section 1.5 of this thesis. The first aim was the extreme pruning of Random Forests. As discussed in Section 1.5, there were two objectives to support this aim. The first objective was met by exploiting clustering and the second objective was met by exploiting the Local Outlier Factor, as previously discussed in this section. To the best of our knowledge, such techniques were not used before to extreme prune Random Forests.

The second aim as discussed in Section 1.5 was improving feature interaction using new feature engineering techniques. To support this aim, two objectives were met. The first objective was met by using diversified weighted subspaces, and the second objective was met using RD, as discussed above.

The following table lists the aims, approaches taken for each aim, and the research contributions that were developed to meet these objectives.

Table 8.1: Summary of Aims, Approaches & Research Contributions

Aims	Approaches	Research Contributions
Extreme Pruning of RFs	Clustering Local Outlier Factor	*CLUB-DRF* *LOFB-DRF*
Improving Feature Interaction in RFs	Weighted Subspaces Replicator Dynamics	*DSB-RF* *RDB-DRF*

Next, we summarize the results of the experimental study that was performed for each method. Starting with *CLUB-DRF* and comparing the performance of the three variations in the charts in Figures 4.2, 4.3, and 4.4, we can see that *BestRepOnTraining* has performed the best, and *BestRepOnOOB* as the second best. In the *BestRepOnTraining* variation, from each cluster, the tree that has achieved the highest performance on the training data is selected. This variation outperformed *RF* on 14/15 of the datasets, and completely outperformed *RF* on the 5/14 datasets. The *BestRepOnOOB* variation, where from each cluster, the tree that has achieved the highest performance on the out-of-bag (OOB) data is chosen, produced less favorable results as it outperformed *RF* on 13/15 of the datasets, and completely outperformed *RF* on the 6/13 datasets.

The *RandomRep* variation did not perform well compared with the others as depicted by the chart in Figure 4.4. This is attributed to the fact that, unlike the first two variations where accuracy was the main criteria used for the selection of a representative from each cluster, a core *CLUB-DRF* component, which is accuracy, was removed in *RandomRep*. The selection process in this variation simply involved selecting a random representative from each cluster in the absence of accuracy. As previously discussed, this emphasizes the importance of having an accuracy measure when selecting a representative from each cluster.

As for *LOFB-DRF* and as shown in the charts in Figures 5.2 and 5.3, the *WeightedOnTraining* variation performed better than the *WeightedOnOOB* variation as it outperformed *RF* on 9/15 datasets . In the *WeightedOnTraining* variation, the top k trees having the highest weight (the product of the normalized LOF value and the accuracy on the training data), are selected. On the other hand, in the *WeightedOnOOB* variation, the top k trees having the highest weight (the product of the normalized LOF value and the accuracy on the OOB data), are selected.

For the maximum and the best performer pruning levels depicted in both columns in Tables 4.11, 4.12, 5.9, and 5.10, both *CLUB-DRF* and *LOFB-DRF* have done extremely well. In these tables, the pruning level achieved was mostly 99% (highest pruning level ever reported to date). This means that for the majority of the datasets, the smallest *CLUB-DRF*/*LOFB-DRF* that outperformed *RF* is of size 5, and the best performer is also of size 5. Size 5 is the size of the smallest pruned ensemble produced by *CLUB-DRF*/*LOFB-DRF* and corresponds to a 99% pruning level; the highest pruning level achieved by both *LOFB-DRF* and *CLUB-DRF*. This makes the extremely pruned *CLUB-DRF*/*LOFB-DRF* ensembles typical not only for real-time applications, but also for machines and mobile devices with limited memory and processing power.

As depicted in Table 6.4, *Type I* of *DSB-RF* (10 sub-forests with 50 trees each), good results were obtained at 60% and 70% of the features. As for *Type II* (20 sub-forests with 25 trees each), good results were obtained at 60%, 70%, 80% and 90% of the features. Interestingly enough, both types of *DSB-RF* have demonstrated promising results especially for medical datasets as demonstrated in Tables 6.2 and 6.3.

To deal with two issues that can affect the performance of *DSB-RF*, namely, noisy and correlated features as discussed in Subsection 6.4.3, *RDB-DRF* was developed. The performance of *RDB-DRF* was the best in the first and second experiments sets depicted in Figures 7.2 and 7.3 respectively. In such experiments, the first variation given by the Equations 7.3 and 7.4 was used. Since we adopted $\beta = \gamma = 1$, only 1 tree may be added/removed in each iteration to allow the sub-forests to grow/shrink in a slow controlled fashion. Taking a closer look at these figures, you will see that the number of wins exceeded the number of losses in 80% of the scenarios. The second variation where a fraction of the initial number of trees in the sub-forest is added or

removed as per Equations 7.5 and 7.6, less favorable results were obtained as depicted in Figures 7.4 and 7.5 respectively. Taking a closer look at Figure 7.4, the number of wins exceeded the number of losses in 5 scenarios, and tied losses in 3 scenarios. In Figure 7.5, the number of wins exceeded the number of losses in 4 scenarios, and tied losses in 1 scenario.

Performance-wise, *CLUB-DRF* is ranked first in terms of accuracy and classification speed making it ideal for real-time applications, and for machines/devices with limited memory and processing power.

8.2 Future Work

In the following two subsections, we suggest both direct and indirect future work. Direct future work aims at using large datasets in both size and features, and re-running the experiments using different parameters scenarios. Indirect future work, on the other hand, is a kind of visionary work that aims at assessing the performance the new methods proposed in this thesis, on real applications and datasets, by having them participate in various local and global competitions.

8.2.1 Direct Future Work

For *CLUB-DRF*, another interesting research direction would be to use other clustering algorithms like DBSCAN Ester et al. (1996), CLARANS Ng and Han (2002), BIRCH Zhang et al. (1996), and/or CURE Guha et al. (1998). There is a potential that the way clusters are formed by each algorithm may have an impact on the performance of *CLUB-DRF*. This can happen when representatives selected from the clusters of one algorithm are more/less diverse than others selected from clusters produced by another algorithm. Different sizes for the initial *RF* and also different cluster increments can also be tried.

To improve the performance of *LOFB-DRF*, a hybrid approach can be used that combines LOF with clustering to boost diversity up. Using this approach, we first create clusters of trees, then from each cluster, we select a representative that corresponds to the instance with the highest weighted LOF value.

For both *CLUB-DRF* and *LOFB-DRF* which are typical for real-time applications as previously discussed in Chapters 4 and 5, it would be useful to apply these methods to such applications and assess their performance.

For *DSB-RF*, the Absolute Predictive Power (APP) has been used to weigh each projected dataset. As future work, two other scenarios can be considered. In the first one, we could use the Relative Predictive Power (RPP), also by Cuzzocrea et al. (2013), instead of APP. A

second and more interesting scenario would be to use a hybrid approach where some projected datasets are weighed according to APP and others according to RPP. It would be interesting to compare the performance of the three scenarios to determine the one that yields the best results. Furthermore, in our experiments, we have used a subspace factor of 2% and 4%, a size of 500 trees for the *DSB-RF* forest to be created, and 50%, 60%, 70%, 80%, and 90% of the features in each subspace. In the future, we will attempt different values for these parameters.

As for *RDB-DRF*, since the first variation given by the Equations 7.3 and 7.4 produced favorable results where we adopted $\beta = \gamma = 1$, it would be interesting to experiment with other values of β and γ other than 1.

For all new methods proposed in this dissertation, we would like to extend the testing to high dimensional and large datasets. This would be quite interesting particularly for *DSB-RF* and *RDB-DRF* as high dimensional datasets are likely to produce more diverse sub-forests, hence, more diverse ensembles. Because diversity can lead to better performance as was repeatedly stated throughout the thesis, we expect higher performance for high dimensional datasets than low dimensional datasets.

Further research can also be conducted to show how the proposed methods in this dissertation perform when used in new and emerging applications.

8.2.2 Indirect Future Work

As a platform for predictive modeling and analytics competitions, Kaggle Narayanan et al. (2011) offers competitions based on real applications and datasets for experimenting with different machine learning classification techniques. Winning teams in each competition are rewarded with valuable prizes. As previously mentioned in Chapter 1, an evaluation study made by Fernández-Delgado et al. (2014) revealed that RF has proven to be the best family of classifiers. It comes at no surprize, then, that RF was a former winner of Kaggle competition to predict the set of bird species present in audio recordings, collected in field conditions Fodor (2013). Since it has been proven empirically that the enhancements developed in this thesis outperformed the traditional RF, it would be a good idea to have these enhancements participate in Kaggle competitions. By doing so, the ultimate objective here is not to win a prize but rather, to determine how well these methods perform in the field when applied to real applications and real datasets.

Another future work direction would be to conduct an evaluation study similar to Fernández-Delgado et al. (2014) where the methods developed in this dissertation are evaluated against other methods. Such an evaluation study would be useful to determine the overall ranking of these methods relative to others. Furthermore, depending on the ranking achieved by these

methods, the study can further provide an insight as to whether further improvements are needed to these methods or not.

Bibliography

Han Jiawei and Micheline Kamber. Data mining: concepts and techniques. *San Francisco, CA, itd: Morgan Kaufmann*, 5, 2011.

Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering: An Open Access Journal*, 2(1):602–609, 2014a.

Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

Ron Kohavi and Foster Provost. Glossary of terms. *Machine Learning*, 30(2-3):271–274, 1998.

Weizhong Yan and Kai F Goebel. Designing classifier ensembles with constrained performance requirements. In *Defense and Security*, pages 59–68. International Society for Optics and Photonics, 2004.

Robi Polikar. Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45, 2006.

Richard Maclin and David Opitz. Popular ensemble methods: An empirical study. *Journal Of Artificial Intelligence Research*, 11(1-2):169–198, 1999.

Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

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

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

Juan José Garcıa Adeva, U Beresi, and R Calvo. Accuracy and diversity in ensembles of text categorisers. *CLEI Electronic Journal*, 9(1), 2005.

E Ke Tang, Ponnuthurai N Suganthan, and Xin Yao. An analysis of diversity measures. *Machine Learning*, 65(1):247–271, 2006.

Simon Bernard, Laurent Heutte, and Sébastien Adam. A study of strength and correlation in random forests. In *Advanced Intelligent Computing Theories and Applications*, pages 186–191. Springer, 2010.

Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000.

Peter Schuster and Karl Sigmund. Replicator dynamics. *Journal of theoretical biology*, 100(3): 533–538, 1983.

Bart Bakker and Tom Heskes. Clustering ensembles of neural network models. *Neural networks*, 16(2):261–269, 2003.

Giorgio Giacinto, Fabio Roli, and Giorgio Fumera. Design of effective multiple classifier systems by clustering of classifiers. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 2, pages 160–163. IEEE, 2000.

Aleksandar Lazarevic and Zoran Obradovic. Effective pruning of neural network classifier ensembles. In *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, volume 2, pages 796–801. IEEE, 2001.

Fu Qiang, Hu Shang-Xu, and Zhao Sheng-Ying. Clustering-based selective neural network ensemble. *Journal of Zhejiang University Science A*, 6(5):387–392, 2005.

Louisa Lam and Ching Y Suen. Application of majority voting to pattern recognition: an analysis of its behavior and performance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 27(5):553–568, 1997.

Raja Khurram Shahzad and Niklas Lavesson. Veto-based malware detection. In *Availability, Reliability and Security (ARES), 2012 Seventh International Conference on*, pages 47–54. IEEE, 2012.

Yu-An Sun and Christopher Dance. When majority voting fails: Comparing quality assurance methods for noisy human computation environment. *arXiv preprint arXiv:1204.3516*, 2012.

Raja Khurram Shahzad and Niklas Lavesson. Comparative analysis of voting schemes for ensemble-based malware detection. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 4(1):98–117, 2013.

Yong Yang, Guoyin Wang, and Kun He. An approach for selective ensemble feature selection based on rough set theory. In *Rough Sets and Knowledge Technology*, pages 518–525. Springer, 2007.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Tom Bylander and Lisa Tate. Using validation sets to avoid overfitting in adaboost. In *FLAIRS Conference*, pages 544–549, 2006.

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

David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

Alexander K Seewald. How to make stacking better and faster while also taking care of an unknown weakness. In *Proceedings of the nineteenth international conference on machine learning*, pages 554–561. Morgan Kaufmann Publishers Inc., 2002.

Yi Zhang, Samuel Burer, and W Nick Street. Ensemble pruning via semi-definite programming. *The Journal of Machine Learning Research*, 7:1315–1338, 2006.

Grigorios Tsoumakas, Ioannis Partalas, and Ioannis Vlahavas. An ensemble pruning primer. In *Applications of supervised and unsupervised ensemble methods*, pages 1–13. Springer, 2009.

Kantilal Varichand Mardia, John T Kent, and John M Bibby. Multivariate analysis. 1980.

Dragos D Margineantu and Thomas G Dietterich. Pruning adaptive boosting. In *ICML*, volume 97, pages 211–218. Citeseer, 1997.

Gonzalo Martínez-Muñoz and Alberto Suárez. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd international conference on Machine learning*, pages 609–616. ACM, 2006.

Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

Randy L Haupt and Sue Ellen Haupt. *Practical genetic algorithms*. John Wiley & Sons, 2004.

Wolfgang Banzhaf, Peter Nordin, Robert E Keller, and Frank D Francone. *Genetic programming: an introduction*, volume 1. Morgan Kaufmann San Francisco, 1998.

Zhi-Hua Zhou and Wei Tang. Selective ensemble of decision trees. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, pages 476–483. Springer, 2003.

Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, page 18. ACM, 2004.

Wei Fan, Fang Chu, Haixun Wang, and Philip S Yu. Pruning and dynamic scheduling of cost-sensitive ensembles. In *AAAI/IAAI*, pages 146–151, 2002.

Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. Pruning an ensemble of classifiers via reinforcement learning. *Neurocomputing*, 72(7):1900–1909, 2009.

Gonzalo Martınez-Munoz and Alberto Suárez. Aggregation ordering in bagging. In *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*, pages 258–263. Citeseer, 2004.

Robert E Banfield, Lawrence O Hall, Kevin W Bowyer, and W Philip Kegelmeyer. Ensemble diversity measures and their application to thinning. *Information Fusion*, 6(1):49–62, 2005.

Ioannis Partalas, Grigorios Tsoumakas, and Ioannis P Vlahavas. Focused ensemble selection: A diversity-based method for greedy ensemble selection. In *ECAI*, pages 117–121, 2008.

Ying Yang, Kevin Korb, Kai Ming Ting, and Geoffrey I Webb. Ensemble selection for superparent-one-dependence estimators. In *AI 2005: Advances in Artificial Intelligence*, pages 102–112. Springer, 2005.

Patrice Latinne, Olivier Debeir, and Christine Decaestecker. Limiting the number of trees in random forests. In *Multiple Classifier Systems*, pages 178–187. Springer, 2001.

Grigorios Tsoumakas, Lefteris Angelis, and Ioannis Vlahavas. Selective fusion of heterogeneous classifiers. *Intelligent Data Analysis*, 9(6):511–525, 2005.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective voting of heterogeneous classifiers. In *Machine Learning: ECML 2004*, pages 465–476. Springer, 2004.

Ioannis Partalas, Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Ensemble pruning using reinforcement learning. In *Advances in Artificial Intelligence*, pages 301–310. Springer, 2006.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Marko Robnik-Šikonja. Improving random forests. In *Machine Learning: ECML 2004*, pages 359–370. Springer, 2004.

Alexey Tsymbal, Mykola Pechenizkiy, and Pádraig Cunningham. Dynamic integration with random forests. In *Machine Learning: ECML 2006*, pages 801–808. Springer, 2006.

Dhammika Amaratunga, Javier Cabrera, and Yung-Seop Lee. Enriched random forests. *Bioinformatics*, 24(18):2010–2014, 2008.

Amir Saffari, Christian Leistner, Jakob Santner, Martin Godec, and Horst Bischof. On-line random forests. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1393–1400. IEEE, 2009.

Mohamed Bader-El-Den and Mohamed Gaber. Garf: towards self-optimised random forests. In *Neural Information Processing*, pages 506–515. Springer, 2012.

David E Goldberg. Genetic algorithms in search, optimization, and machine learning. 1989.

Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.

David B Skalak. The sources of increased accuracy for two proposed boosting algorithms. In *Proc. American Association for Artificial Intelligence, AAAI-96, Integrating Multiple Learned Models Workshop*, volume 1129, page 1133. Citeseer, 1996.

Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9):699–707, 2001.

Ron Kohavi, David H Wolpert, et al. Bias plus variance decomposition for zero-one loss functions. In *ICML*, pages 275–283, 1996.

Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. *Statistical methods for rates and proportions*. John Wiley & Sons, 2013.

Derek Partridge and W Krzanowski. Software diversity: practical statistics for its measurement and exploitation. *Information and software technology*, 39(10):707–717, 1997.

Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. *Machine Learning*, 81(3):257–282, 2010.

Jian Li, Ke Yi, and Qin Zhang. Clustering with diversity. In *Automata, Languages and Programming*, pages 188–200. Springer, 2010.

Ludmila I Kuncheva and Stefan Todorov Hadjitodorov. Using diversity in cluster ensembles. In *Systems, man and cybernetics, 2004 IEEE international conference on*, volume 2, pages 1214–1219. IEEE, 2004.

RD Brown and YC Martin. An evaluation of structural descriptors and clustering methods for use in diversity selection. *SAR and QSAR in Environmental Research*, 8(1-2):23–39, 1998.

Norah E Shemetulskis, James B Dunbar Jr, Bonnie W Dunbar, David W Moreland, and Christine Humblet. Enhancing the diversity of a corporate database using chemical database clustering and analysis. *Journal of Computer-Aided Molecular Design*, 9(5):407–416, 1995.

Jungwon Lee, Yakun Sun, Rohit Nabar, and Hui-Ling Lou. Cluster-based transmit diversity scheme for mimo ofdm systems. In *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th*, pages 1–5. IEEE, 2008.

Thomas Sharpton, Guillaume Jospin, Dongying Wu, Morgan Langille, Katherine Pollard, and Jonathan Eisen. Sifting through genomes with iterative-sequence clustering produces a large, phylogenetically diverse protein-family resource. *BMC bioinformatics*, 13(1):264, 2012.

Hans-Peter Kriegel Peer Kröger Erich and Schubert Arthur Zimek. Interpreting and unifying outlier scores. In *11th SIAM International Conference on Data Mining (SDM), Mesa, AZ*, 2011.

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

Tin Kam Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995.

Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural computation*, 9(7):1545–1588, 1997.

D Richard Cutler, Thomas C Edwards Jr, Karen H Beard, Adele Cutler, Kyle T Hess, Jacob Gibson, and Joshua J Lawler. Random forests for classification in ecology. *Ecology*, 88(11): 2783–2792, 2007.

Myungsook Klassen, Matt Cummings, and G Seldana. Investigation of random forest performance with cancer microarray data. In *Proceedings of ISCA 24th International Conference ON Computers and Their Applications*, 2008.

Wei Hu. Identifying predictive markers of chemosensitivity of breast cancer with random forests. *cancer*, 13:14, 2009.

Dan Gao, Yan-Xia Zhang, and Yong-Heng Zhao. Random forest algorithm for classification of multiwavelength data. *Research in Astronomy and Astrophysics*, 9(2):220, 2009.

Abraham D Flaxman, Alireza Vahdatpour, Sean Green, Spencer L James, Christopher JL Murray, et al. Random forests for verbal autopsy analysis: multisite validation study using clinical diagnostic gold standards. *Popul Health Metr*, 9(29):2, 2011.

Fatin Zaklouta, Bogdan Stanciulescu, and Omar Hamdoun. Traffic sign classification using kd trees and random forests. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2151–2155. IEEE, 2011.

Fabian Löw, Gunther Schorcht, Ulrich Michel, Stefan Dech, and Christopher Conrad. Per-field crop classification in irrigated agricultural regions in middle asia using random forest and support vector machine ensemble. In *SPIE Remote Sensing*, pages 85380R–85380R. International Society for Optics and Photonics, 2012.

Anne-Laure Boulesteix, Silke Janitza, Jochen Kruppa, and Inke R. König. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 2(6):493–507, 2012.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3): 18–22, 2002.

Praveen Boinee, Alessandro De Angelis, and Gian Luca Foresti. Meta random forests. *International Journal of Computationnal Intelligence*, 2(3):138–147, 2005.

Douglas M Hawkins et al. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12, 2004.

Graham Williams. *Use R: Data Mining with Rattle and R: the Art of Excavating Data for Knowledge Discovery*. Springer, 2011.

Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. Club-drf: A clustering approach to extreme pruning of random forests. In *Research and Development in Intelligent Systems XXXII*, pages 59–73. Springer, 2015.

S. Bernard, L. Heutte, and S. Adam. On the selection of decision trees in random forests. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 302–307, June 2009. doi: 10.1109/IJCNN.2009.5178693.

James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA, 1967.

Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8): 651–666, 2010.

Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.

Zhexue Huang and Michael K Ng. A fuzzy k-modes algorithm for clustering categorical data. *Fuzzy Systems, IEEE Transactions on*, 7(4):446–452, 1999.

Ohn Mar San, Van-Nam Huynh, and Yoshiteru Nakamori. An alternative extension of the k-means algorithm for clustering categorical data. *International Journal of Applied Mathematics and Computer Science*, 14(2):241–248, 2004.

Zhexue Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *DMKD*, page 0. Citeseer, 1997.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. *The WEKA data mining software: an update*, volume 11. ACM, 2009.

Malay K Pakhira. A modified k-means algorithm to avoid empty clusters. *International Journal of Recent Trends in Engineering*, 1(1):1, 2009.

K. Bache and M. Lichman. Uci machine learning repository, 2013.

Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. A linear method for deviation detection in large databases. In *KDD*, pages 164–169, 1996.

Ida Ruts and Peter J Rousseeuw. Computing depth contours of bivariate point clouds. *Computational Statistics & Data Analysis*, 23(1):153–168, 1996.

Edwin M Knox and Raymond T Ng. Algorithms for mining distance-based outliers in large datasets. In *Proceedings of the International Conference on Very Large Data Bases*. Citeseer, 1998.

Edwin M Knorr and Raymond T Ng. Finding intensional knowledge of distance-based outliers. In *VLDB*, volume 99, pages 211–222, 1999.

Erich Schubert, Remigius Wojdanowski, Arthur Zimek, and Hans-Peter Kriegel. On evaluation of outlier rankings and outlier scores. In *Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA, 2012*, pages 1047–1058, 2012.

Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. Diversified random forests using random subspaces. In *Intelligent Data Engineering and Automated Learning–IDEAL 2014*, pages 85–92. Springer, 2014b.

Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. An Outlier Detection-based Tree Selection Approach to Extreme Pruning of Random Forests. In *17th International Conference on Engineering Applications of Neural Networks–EANN 2016* (submitted).

Alfredo Cuzzocrea, Shane Leo Francis, and Mohamed Medhat Gaber. An information-theoretic approach for setting the optimal number of decision trees in random forests. In *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pages 1013–1019. IEEE, 2013.

Khaled Fawgreh, Mohamed Medhat Gaber, and Eyad Elyan. A replicator dynamics approach to collective feature engineering in random forests. In *Research and Development in Intelligent Systems XXXII*, pages 25–41. Springer, 2015.

Peter D Taylor and Leo B Jonker. Evolutionary stable strategies and game dynamics. *Mathematical biosciences*, 40(1):145–156, 1978.

Josef Hofbauer and Karl Sigmund. Evolutionary game dynamics. *Bulletin of the American Mathematical Society*, 40(4):479–519, 2003.

Christian Hilbe. Local replicator dynamics: A simple link between deterministic and stochastic models of evolutionary game theory. *Bulletin of mathematical biology*, 73(9):2068–2087, 2011.

Christoph Hauert. Replicator dynamics of reward & reputation in public goods games. *Journal of theoretical biology*, 267(1):22–28, 2010.

Martin A Nowak and Karl Sigmund. Evolutionary dynamics of biological games. *science*, 303 (5659):793–799, 2004.

Christoph Hauert, Silvia De Monte, Josef Hofbauer, and Karl Sigmund. Replicator dynamics for optional public good games. *Journal of Theoretical Biology*, 218(2):187–194, 2002.

Carlos P Roca, José A Cuesta, and Angel Sánchez. Evolutionary game theory: Temporal and spatial effects beyond replicator dynamics. *Physics of Life Reviews*, 6(4):208–249, 2009.

Gabriele Lohmann and Stefan Bohn. Using replicator dynamics for analyzing fmri data of the human brain. *Medical Imaging, IEEE Transactions on*, 21(5):485–492, 2002.

Immanuel M Bomze. Lotka-volterra equation and replicator dynamics: new issues in classification. *Biological cybernetics*, 72(5):447–453, 1995.

Vivian Hutson and Klaus Schmitt. Permanence and the dynamics of biological systems. *Mathematical Biosciences*, 111(1):1–71, 1992.

Aram Galstyan. Continuous strategy replicator dynamics for multi-agent q-learning. *Autonomous agents and multi-agent systems*, 26(1):37–53, 2013.

Karl Tuyls, Pieter Jan T Hoen, and Bram Vanschoenwinkel. An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems*, 12 (1):115–153, 2006.

R Olfati-Saber. Evolutionary dynamics of behavior in social networks. In *Decision and Control, 2007 46th IEEE Conference on*, pages 4051–4056. IEEE, 2007.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *Knowledge and Data Engineering, IEEE Transactions on*, 14(5):1003–1016, 2002.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996.

Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. In *ACM SIGMOD Record*, volume 27, pages 73–84. ACM, 1998.

Arvind Narayanan, Elaine Shi, and Benjamin IP Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1825–1834. IEEE, 2011.

Gábor Fodor. The ninth annual mlsp competition: first place. In *Machine Learning for Signal Processing (MLSP), 2013 IEEE International Workshop on*, pages 1–2. IEEE, 2013.