Heriot-Watt University

Masters Thesis

Pruning Random Forests

Author:

Supervisor:

"Realists do not fear the results of their study."

Fyodor Dostoevsky

Abstract

Random Forest is one of the most popular classification techniques that was developed by Leo Breiman in 2001. It is an ensemble based technique with decision trees as its building blocks. The trees vote for an outcome when a new instance is being classified, and a majority vote is taken to decide the output. Many extensions have been proposed to optimize and further enhance its performance. Two main factors play an essential role in random forest's performance: correlation and diversity among trees in the forest. Correlation between trees decreases the accuracy, while diversity among trees is a desired feature. This dissertation aims at optimizing these two factors by using clustering analysis of trees in order to prune correlated trees while keeping the outliers to maintain diversity. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is one the most commonly used clustering techniques that is immune to outliers. It will be used to group correlated trees in the same cluster based on their prediction behaviour, and then choosing a representative of the cluster. In addition, the trees which are marked as outliers will be kept in the forest to maintain the diversity among trees.

Keywords: Machine Learning, Random Forests, Clustering, DBSCAN, Pruning.

Acknowledgements

I would like to thank my supervisor Dr. Hani Ragab for all his support since I joined Heriot-Watt University, and especially for his support and guidance in this dissertation.

Contents

List of Figures

List of Tables

Abbreviations

DBSCAN	**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
OPTICS	**O**rdering-**P**oints **T**o **I**dentify the **C**lustering **S**tructure
SFS	**S**equential-**F**orward **S**earch
SBS	**S**equential-**B**ackward **S**earch

Chapter 1

Introduction

Data mining and machine learning have been a very active research area in the past years. This is due to the amount of data being collected nowadays, and the availability of computational resources that enable researchers to execute machine learning techniques on big datasets. Those techniques are generally divided into supervised and unsupervised approaches [1]. The distinction is based on whether the desired output is known beforehand at the training time or not. Classification is a supervised learning technique in which we try to predict a class or a label for a given instance after acquiring the knowledge from the training samples [2]. Clustering, on the other hand, is an unsupervised learning technique in which similar instances are grouped in clusters which can be thought of as classes [3].

Decision trees are one of the most commonly used classifiers due to their simplicity and good performance [4]. Random forest is a classification method which consists of combining multiple decision trees as one ensemble and classify new instances based on the majority votes of individual trees [5]. Random forest tend to perform better than a single decision tree because it can generalize well due to the high number of trees. Correlation and diversity among trees in the forest play an important role in its classification accuracy. Correlation between trees decreases the performance, while diversity among trees is a desired feature. It was found to be the best classifier among 179 classifiers belonging to 17 different classifier families in a 2014 survey [6]. This highlights the importance of random forests, and

the motivation to further optimize its performance. Many extensions to enhance random forest classification accuracy have been proposed since its inception. Those enhancements include techniques like changing the voting mechanism [7, 8], using different attribute evaluation metrics [7], and pruning [9, 10, 11]. Pruning is done by removing correlated decision trees from a random forest which makes it faster and reduces the classification error. In this dissertation, we will propose a novel method to prune random forests using a clustering technique called Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [12]. DBSCAN clusters instances based on the distance between them and the density of resulting cluster. Correlated trees will be clustered, and a representative will be chosen from each cluster. Finally, being immune to outliers is one of DBSCAN's strengths and this will be utilized to increase diversity in the forest.

1.1 Aims and Objectives

The aim of this dissertation is to investigate the effectiveness of using DBSCAN clustering technique to prune random forests. Pruning Random Forests is performed to produce a smaller and faster model which requires less computational power while maintaining good accuracy. More specifically, the objectives of this research are:

- Determining DBSCAN and random forest parameters that optimize the accuracy.
- Investigating the different methods for choosing representative from a cluster.
- Investigating the effect of keeping the outliers on accuracy.
- Comparing the performance of the new model against the original random forest and other existing random forests enhancements.

The rest of this manuscript is organised as follows: First, we will give a summary of the necessary techniques used in this dissertation in addition to a review of the

related work in the literature. Then we will explain our methodology in investigating the problem before we define the requirements and evaluation method of our study. Finally, we will discuss the professional, legal and ethical issues and conclude by detailing the project plan.

Chapter 2

Background and Literature Review

In this chapter, we will introduce some of the core techniques and concepts in machine learning in order to grasp the objectives of this dissertation. Then, we will investigate notable works for pruning ensemble models and random forest in particular. Finally, we will describe and discuss the contribution and some possible future work.

2.1 Clustering

Unsupervised learning is a technique used in Machine Learning and Data Mining to extract patterns when the input consists of unlabelled data. Clustering is the most widely used technique which breaks down the input into smaller groups of similar data based on a predefined similarity measure [3]. Several clustering algorithms exist and are used depending on the nature of the problem. An essential step in clustering is defining a meaningful similarity measure since the objective is to group similar instances in the same group, and far away from dissimilar instances. The number of clusters is the second main step if it is known beforehand, or can be estimated using some heuristics or the help of experts then K-Means is the most

commonly used algorithm for cluster analysis in such cases [13]. On the other hand, DBSCAN [12], or Ordering Points To Identify the Clustering Structure (OPTICS) [14] clustering algorithms are used if the number of clusters cannot be estimated. These algorithms do not require a predefined number of clusters. We will explain some of the relevant techniques in the next subsections along with a critical analysis of their usage.

2.1.1 Similarity measures

As mentioned above, defining a meaningful similarity measure is an important step to carry out cluster analysis [3]. These measures usually depend on the nature of the problem and data types. The distance between numeric data is the easiest to measure since multiple well-established measures come from mathematics. Euclidean distance is the most common distance measures which is calculated using formula 2.1.

$$dist((x_1, y_1), (x_2, y_2))) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.1}$$

Categorical data or strings, on the other hand, have different measures. Edit distance measures how similar or dissimilar two strings are [15]. It measures the distance based on the number of operations required to generate one string from the other by defining three operations: Insertion, Deletion, and Substitution. Simple matching distance is a way of measuring the edit distance between two strings or two categorical vectors with the same length, which means it counts the number mismatches between the two vectors [16]. Let x and y be vectors of length n, then distance between them is calculated according to formula 2.2.

$$dist = \sum_{i=0}^{n} \delta(x_i, y_i) \tag{2.2}$$

$$\delta(x_i, y_i) = \begin{cases} 0 \text{ if } x_i = y_i \\ 1 \text{ otherwise;} \end{cases} \tag{2.3}$$

2.1.2 K-Means

K-means is a clustering algorithm for data with numerical attributes. As the name suggests, it has K clusters which are determined before running the algorithm [17]. K-means randomly chooses a centroid (cluster centre) for each cluster from the dataset at the first iteration. Then, it places every point in the cluster with the closest centroid until no point is left un-clustered. In the next iteration, a new centroid is calculated for each cluster, which represents the mean of the points in this cluster. All the points are then assigned to the cluster with the closest new centroid. This process continues until no point changes its cluster which means the algorithm has converged. Figure 2.1 illustrates the main drawback of K-Means which is its sensitivity to outliers since an outlier will shift the centroid of a cluster towards itself. Another challenge in K-Means is the initial choice of centroids since the highly affect the clustering results.

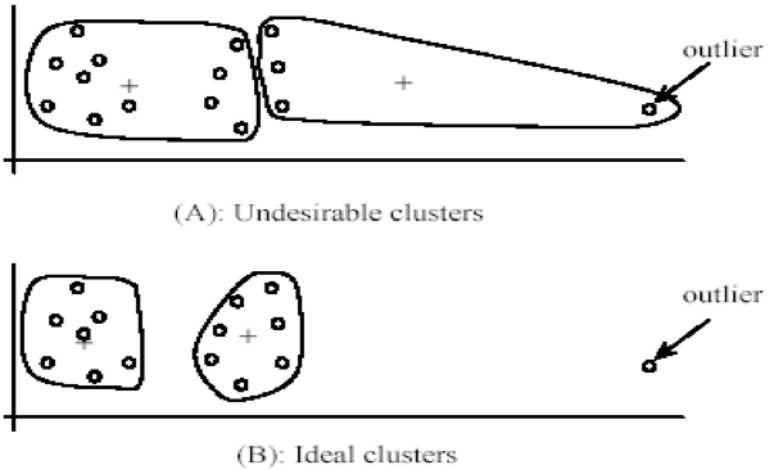

FIGURE 2.1: Outlier effect on K-Mean Results [18].

K-Modes is an extension from K-Means for dealing with categorical (non-numeric) data [16]. It follows the same steps as K-Means but calculates the mode of a cluster

instead of the mean to find the new centroids at each iteration. K-Modes faces the same challenges of K-Means which are: sensitivity to outliers, and the initial selection of the centroids.

2.1.3 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has many advantages over other clustering techniques as it is insensitive to outliers, can find clusters with arbitrary shapes, and does not require a predefined number of clusters [12]. These advantages are beneficial when the domain knowledge is unavailable, or the data is very noisy.

DBSCAN is based on two parameters; the first one is minPts which is the minimum number of neighbours to form a cluster. The second one is ϵ, which is the maximum distance between two points in a neighbourhood. Based on those two parameters, DBSCAN classifies points into three types; Core points which have minPts neighbours with distance less than or equal to ϵ. Border points are points which lie within an ϵ radius from a core point but have less than minPts neighbours in their ϵ radius. Finally, Outliers which are neither core or border points are left un-clustered. DBSCAN also defines a reachability notion which states that:

1. A point is directly reachable from a core point if they lie within ϵ distance from each other.

2. A point is reachable from a core point p if there is a connected path of core points from p such that; each two consecutive core points along the path are directly reachable from each other.

DBSCAN start by discovering the neighbours of a random point within ϵ distance. If it has minPts neighbours, then it is marked as core point. This is done for all points, and all core points that are within ϵ distance from each other are grouped in the same cluster. Border points are then assigned to the closest cluster that has a directly reachable point, and outliers are left un-clustered.

Figure 2.2 shows one of DBSCAN's strengths is its ability to find clusters with arbitrary shapes. This comes as a result of the minPts parameter and the reachability notion. One drawback of DBSCAN is worth mentioning here which is the fact that it is not deterministic since border points can belong to more than one cluster at the same time. Finally, like all algorithms with parameters, tuning DBSCAN's parameter can be difficult depending on the problem nature, especially when no domain knowledge is available

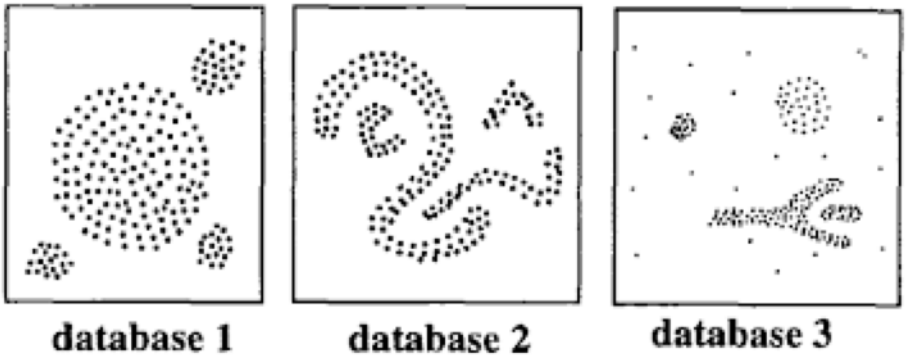

FIGURE 2.2: Arbitrary cluster shapes detected by DBSCAN [12].

2.2 Classification

Supervised learning takes pairs of input-output and tries to predict the output for the new instances based on the learned relations in the training stage. The most two common problems in supervised learning are Classification and Regression [19]. Classification is the problem of labelling new data with missing or unknown label or class. Regression is the problem of modelling the relationship between a set of independent variables (input) and a dependent variable (output). Generally speaking, Classification is concerned with finding a class based on a set of features and Regression is concerned with finding a numeric value based on a set of input variables. Several classifiers exist and are used depending on the nature of the problem. As our focus is Pruning Random Forests, we will only discuss Random Forests and Decision Trees which are the building blocks of Random Forests.

2.2.1 Decision Trees

Decision Trees are one of the most commonly used classifiers in data mining due to its efficiency for large datasets and its intuitive like structure. As shown in Figure 2.3, each decision tree consists of a root, internal nodes, and leaves which represent the class. C4.5 [20] is the most commonly used implementation of decision trees ,and our research will be using this implementation . C4.5 uses Gain Ratio to split at each node and construct the tree branches. It follows a greedy approach by choosing the feature with the highest Gain Ratio as the root and then applies the same at each level until all the instances belong to the same class or there are no more features. One advantage of C4.5 is that it can handle both numerical and categorical features with multiple values. It also does a pruning process in order to avoid the problem of overfitting where the model learns the training set and cannot handle unseen data. C4.5 does a post-pruning process by replacing sub-tress with leaf nodes if it minimizes the classification error; which is the number of misclassified instances over the total number of instances by a particular node.

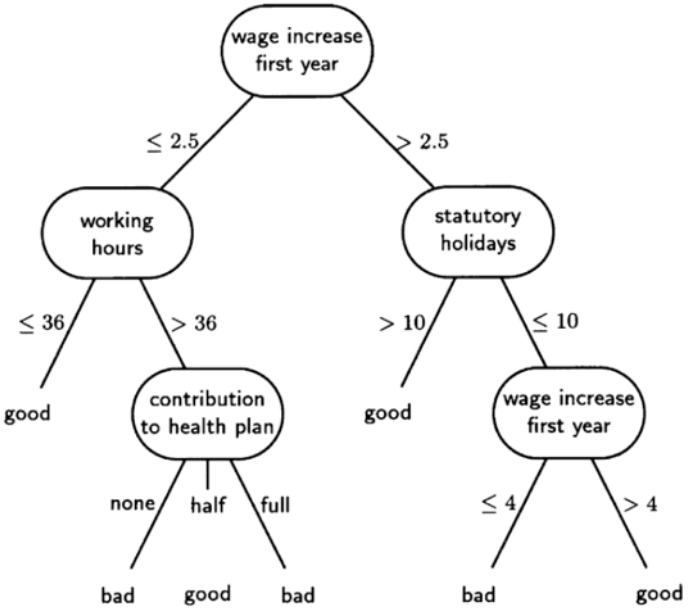

FIGURE 2.3: Decision Tree Example [20].

2.2.2 Random Forests

Ensemble models consist of several models from the same family or different types. They were proposed to overcome the limited predictive performance of a single classifier [21]. Each model in the ensemble produces an output, and the final result is decided according to a voting scheme. Diversity in the ensemble model plays a vital role in the accuracy of the model against unseen data [22]. If two models have different error rates when tested against unseen data, then we say they are diverse.

There are three commonly used ensemble techniques which are bagging, boosting, and stacking. Bagging uses a random subset of the dataset with replacement to build each model, and each model carries the same weight when voting for the output [23]. Random Forests are the most popular example of ensemble models via Bagging.

Boosting, on the other hand, builds an initial weak model and then tries to boost its performance by passing the instances where the output was not the expected one to the next model [24]. This process continues until the data is predicted correctly or a predefined threshold for the number of models is reached. This leads to the problem of overfitting compared with bagging [25].

Stacking is a heuristic technique which works by combining multiple models that belong to different families [26]. The output of these models is then passed to a level-two model which is trained using these predictions to produce the final output.

Random forests use bagging to construct a training set when growing each decision tree in addition to selecting a random subset of features for each tree based on a method proposed in [27], hence the name "Random Forests". According to [5], two main factors affect the classification error:

1. The correlation between trees increases the error, thus diversity is very crucial as mentioned above.

2. The higher the performance of individual trees, the smaller the error rate.

The main strengths of RF are its robustness against outliers and overfitting, better performance than bagging and boosting, and a better or similar accuracy to AdaBoost. With the implementation mentioned above, the number of trees in a forest is in the range of 100 to 500, which makes it unsuitable for real-time application [28]. Another drawback is when the number of features is very high, the correlation among trees will increase which will increase the error rate [5, 29]. Different approaches have been proposed to overcome these two weaknesses which will be mentioned in Related work section, and these two shortcomings are also the motivation behind this work where we are trying to reduce the number of trees while maintaining or enhancing the accuracy.

2.3 Related Work

Many techniques to optimize the performance of Random Forests exist in the literature. We will divide them into two categories pruning techniques, and non-pruning techniques. Non-Pruning techniques try to optimize the performance of the forest by tweaking the parameters of the random forest, or by changing the voting mechanism for example. Pruning techniques, on the other hand, take an ensemble model and reduce the number of models in it to reduce correlation and increase speed. Several such techniques will be explained in the next section.

2.3.1 Pruning Techniques

The objective of pruning ensemble models is twofold, producing a smaller ensemble in order to reduce computational and memory requirements and to boost the performance by eliminating correlated models [30]. Pruning techniques are classified into three types: ranking-based, optimization-based, and clustering-based [31].

In Ranking-based techniques, the models in the ensemble are ranked based on an evaluation measure like Cohen's Kappa, which measures the interrater agreement [32]. One advantage of k statistics is that it takes into account the agreement that could happen by chance. The models are then ranked in ascending order and pruned according to a predefined threshold.

Optimization-based techniques use local search algorithms [33], and genetic programming to find an optimal subset of the models the achieve an objective function [34]. This objective function can be accuracy, Receiver Operating Characteristic (ROC), root-mean-squared-error, or other metrics of evaluation. Hill Climbing was used in [35] to traverse a search space of states to find a local optima, and converges when a better score cannot be achieved by any of the following states. Each state is a subset of the ensemble, and each adjacent state is reached by adding or removing a model from the ensemble.

In Clustering-Based Techniques, the first step is to cluster similar models in the same cluster based on their prediction behaviour. The pruning is then done by choosing a single, or multiple representatives for each cluster. Many mechanisms for selecting a representative exist, such as selecting the centroid of the cluster as target values to construct a new model [36]. Another approach is choosing the model with the farthest distance from all other clusters; this way achieves more diversity among models [37]. Finally, a greedy approach where the models are selected according to their accuracy [38]. One difficulty with clustering like K-Means is determining the number of clusters, using DBSCAN we will try to overcome this issue.

2.3.2 Random Forests Enhancements

Several non-pruning techniques have been proposed such as changing the voting method from majority voting to a weighted voting scheme [7]. This way, the instances similar to the test instance are identified by the internal estimates. Trees that perform well on these instances are given a higher weight than the other trees.

This approach performed better than the original random forest implementation on multiple datasets as stated in [7]. Decreasing the correlation between trees in the forest was also proposed in [7]. This was achieved by using multiple attribute evaluation metrics which lead to a better performance too.

A genetic algorithm based approach was proposed in [10] that outperformed several classification techniques. Genetic algorithms are descendant of evolutionary algorithms; it does operations like mutation, selection, inheritance, and crossover to find an optimal subset. Each tree is represented with a bit, which indicates if it is in the subset or not. An initial random forest of size n is generated, and then a number of sub-forests are randomly drawn to form the initial population. Genetic algorithms operations are then used to optimize the objective function which is the classification accuracy of the sub forest. This approach outperformed several classification techniques including random forests.

Sequential Forward Search (SFS) and Sequential Backward Search (SBS) were used in [39] to add trees to the forest incrementally. The objective was not to find an optimal forest, but rather to prove that a subset of the forest might give a better performance. Their results showed that 50% or even 16% of the trees in the forest give good results compared to the original forest.

Two other approaches relies on overproducing the forest and then removing trees based on different measures were proposed in [40]. The first one is called “by prediction” in which the accuracy of the original forest with n trees is calculated, and then calculate the accuracy of all forests with size n-1. The removed tree that is associated with the weakest forest will then be removed. This will continue until the accuracy converges and removing trees will not produce a better forest. The same procedure is followed, but this time the removal is ”by similarity”. The correlation between the outputs of each two trees is calculated, and the average correlation for a tree with the rest of the forest is the overall similarity with the forest. The tree with the highest similarity is then removed. The results showed that ”by prediction” approach gave better results than the ”by similarity” approach.

McNemar non-parametric test of significance was used in Latinne et al. [9] a priori determine the smallest number of trees in a forest to achieve a similar accuracy to a larger forest. This method also achieved high classification speed and required less memory.

Figure 2.4 illustrates a clustering based approach that used K-Modes to prune random forests [11]. The trees are clustered based on the distance between the output vectors against the training set. The distance is simply the number of mismatches between the output vectors. The higher the distance, the more dissimilar the trees. Different values were tested for the number of clusters, and pruning levels in the range of 92% to 99% were reported. Different strategies for selecting a representative of a cluster were also tried, and the results showed that this approach performed same or better than the original forest.

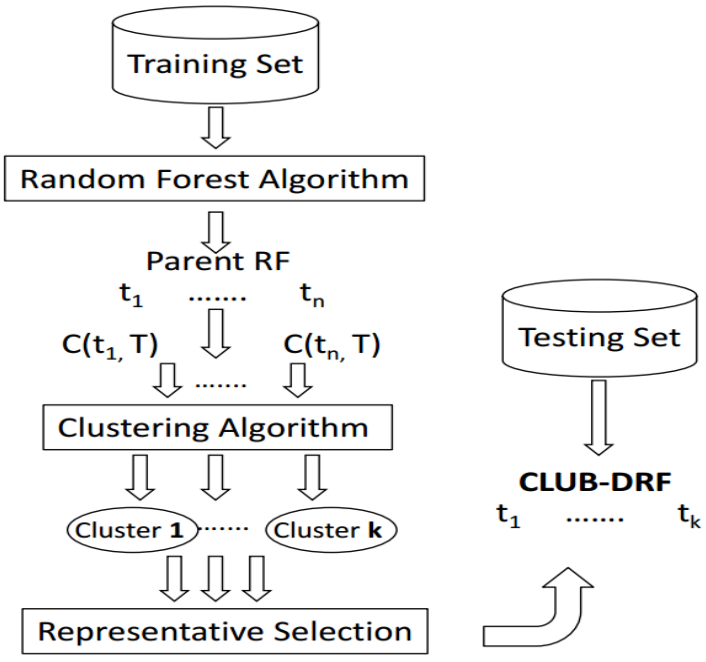

FIGURE 2.4: CLUB-DRF approach [11].

TABLE 2.1: Summary of Related Work

Pruning Technique	Pruning	Performance	Description
Weighted Voting [7]	No	Better over several datasets	The voting mechanism is a weighted one instead of majority voting.
Several feature evaluation measures[7]	No	Better performance	The correlation between trees was decreased which lead to a better performance.
Genetic Algorithms [10]	Yes	Outperformed Several Classifiers including random forests.	Genetic algorithms to find optimal sub-forests.
SFS&SBS [39]	Yes	Better performance with pruning level up to 84%.	Search for a subset of trees that perform better than the original forest using SFS & SBS.
Prune by prediction & by similarity [40]	Yes	Both performed better than random forests, "by prediction" outperformed "by similarity"	Eliminate trees one by one based on the sub-forest's accuracy or the tree's similarity to the forest.
McNemar non-parametric test [9]	Yes	Similar acuracy to original random forests, less computaional resources.	A priori determine the smallest number of trees in a forest
CLUB-DRF [11]	Yes	At least as good as random forests, 92% up to 99% pruning levels.	Cluster trees using K-Modes based on the similarity between output vectors.

2.4 Contribution

To the best of our knowledge, CLUB-DRF this is the only clustering based pruning approach for random forests in the literature. Figure 2.5 illustrates our overall approach which follows a similar approach to CLUB-DRF with two differences:

1. The number of clusters is not predefined in DBSCAN.
2. We will employ the outliers produced by DBSCAN to increase the diversity in the resulting forest.

2.5 Future work

Trying different similarity or correlation measures is an interesting issue to investigate as it affects the clustering results.

Ordering Points To Identify the Clustering Structure (OPTICS) is a variation of DBSCAN but has an advantage over DBSCAN, which is detecting clusters when there is a high variation in the dataset density. Investigating the usage of OPTICS to this particular problem is also another interesting approach to investigate.

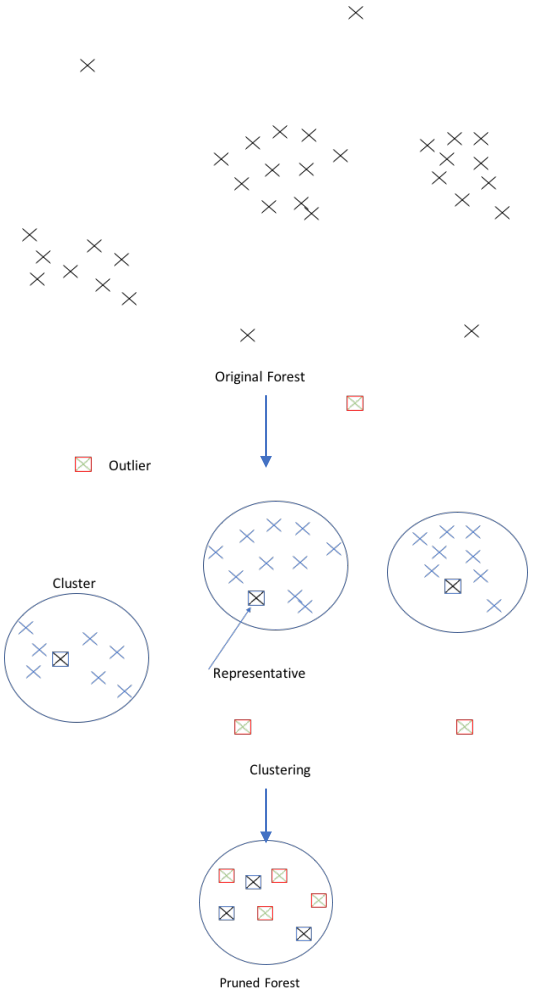

FIGURE 2.5: DBSCAN Pruning Approach.

Chapter 3

Methodology

This chapter describes the approach we will use to investigate the effectiveness of our proposed method. The general development approach is prototype-driven, where we built a simple prototype to test the method, and we will add features incrementally. This will enable us to focus on the validating our method and then worrying about the efficient implementation.

3.1 Software

We will proceed in two steps:

1. A wide range of tools is available to build machine learning prototypes easily. Weka, Knime Advanced Analytics Platform, and Rapid Miner are such examples, and we will use one of them to build the prototype.

2. If the results are promising and enough time is available, a package implemented in R, or Python will be implemented and released as a free open source software to share the benefits of the proposed method.

3.2 Development

Once we decide and evaluate the tools we will use, we will start developing the first prototype that extracts trees from a random forest and clusters them using DBSCAN. Then we will try some different methods for choosing a representative and fine-tune the parameters of random forests and DBSCAN. As mentioned above, these prototypes will be done in iterations following the spiral model in Figure 3.1. Given the nature of the project, this will help in adapting to any changes that might occur in the middle stages. We will follow the same approach in developing the package after analysing the results.

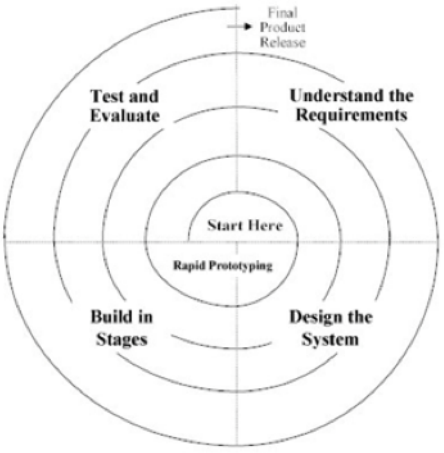

FIGURE 3.1: Rapid Prototyping[41].

3.3 Datasets

We will use 15 different datasets in Table 3.1 from UCI repository, which were used in [11] in order to also compare our results with CLUB-DRF. These datasets are publicly available and usually used as benchmarking datasets. Moreover, a bigger real-world dataset might be chosen later to test the validity of our model against such datasets.

TABLE 3.1: Datasets

Dataset	**Features**	**Records**
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

Chapter 4

Requirements Analysis

The outcome of this research will determine whether using DBSCAN to prune random forests is an effective method or not. A prototype to evaluate the approach will be implemented, and in case the results were promising, a package written in R or Python which performs pruning on random forests will be implemented. Although this is a research-oriented project, some basic requirements are laid out below and divided into functional and non-functional requirements.

4.1 Functional Requirements

- The prototype/package shall give options to configure the parameters of the random forest and DBSCAN
- The prototype/package **shall** give several options to select a representative for each cluster.
- The prototype/package **shall** provide the evaluation metrics for the resulting model against the original forest.
- The prototype/package **should** operate in two modes, starting with only a dataset, or starting with a pre-trained random forest model and performing pruning on it.

4.2 Non-Functional Requirements

- **Speed:** the resulting model **should** run faster than the original forest since it has fewer trees.
- **Usability:** the package **should** be easy to use and configure. The output should also be user-friendly.

4.3 Evaluation

The main evaluation criteria to determine the effectiveness of our proposed method will be the accuracy, speed, and pruning level of the resulting model compared with the original random forest. If it achieves a similar accuracy level as the original model, then the proposed method is successful. Moreover, we will test and evaluate the package to ensure that it meets all the mandatory requirements mentioned above.

The accuracy of a classifier is the number of correctly classified instances over the total number of instances. The algorithm will run 30 times against each dataset in Table 3.1 the average, minimum, and maximum accuracy will be recorded. The reason we will run the model multiple times against the same dataset is due to the random nature of random forests as mentioned earlier. These accuracies will be compared with the accuracy of the original model to determine the statistical significance of the improvement in performance, if any.

The runtime of the pruned model will be recorded against each dataset for each run. Again, the average, minimum, and maximum runtime will be recorded and compared with the original model. Runtime here is divided into two categories, training time, and prediction time. It is expected that the training time of our proposed method will be larger, while the classification time will be smaller and requires less memory.

Pruning level is the percentage of the removed trees from the original model over the total number of trees. This is an indication of how much the size of the forest has decreased.

Chapter 5

Professional, Legal and Ethical Issues

5.1 Professional and Legal Issues

All papers, code, or libraries will be referenced and used under the terms of the publisher licenses. The software produced will be tested, and documented according to the professional software engineering practices. The produced software will be published under the GNU GENERAL PUBLIC LICENSE [42]. The GNU license allows any user to modify, share, and distribute the code given that they make the source code available under the same license. They also must give a contribution to the original author.

5.2 Ethical Issues

This is a research-based project that does not involve users or sensitive datasets. All datasets are publicly available in the UCI machine learning repository and usually used for benchmarking. Thus, there is no risk of violating any ethics code.

Chapter 6

Project Plan

As this is a research-oriented project, the development revolves around investigating the effectiveness of the proposed approach by building prototypes. As mentioned above, we will adapt Rapid Prototyping methodology in developing this project and the project plan reflects that as shown in Gantt chart 6.1. The first prototype will form the main pipeline of the method, and in the second prototype we will carefully enhance the implementation and try to achieve better results.

If the results are promising, we will develop a package to be used in R or Python. Moreover, we will write a paper and submit to a conference or a journal depending on the time and strength of the model. The Gantt Chart below shows the breakdown of the tasks to be carried out through the whole dissertation.

Table 6.1 shows the risks associated with this project. Each risk identified was given an impact and likelihood level (low, medium, or high). The steps to deal with each risks in case it arises is also mentioned.

Table 6.1: Risk Assessment.

Risk	Likelihood	Impact	Mitigation
High workload at work	High	High;less time available for project development	Take an emergency leave from work
Supervisor goes on leave	Medium	High;difficulty in communication to get feedback	Email,phone, or Skype.
Source code loss	Low	High;work has to be done from scratch	Keep backups
Difficulties in tuning the approach	Medium	High; difficulties in obtaining good results	Use more complex tuning algorithms
Project falls behind plan	Medium	Medium;project plan has to be changed	Change plan an allocate more time to the project
Author sickness	Low	Medium;project progress will be affected	Ask for extension if the condition is sever
Datasets are not sufficient	Medium	High; approach evaluation will be affected	Try different datasets

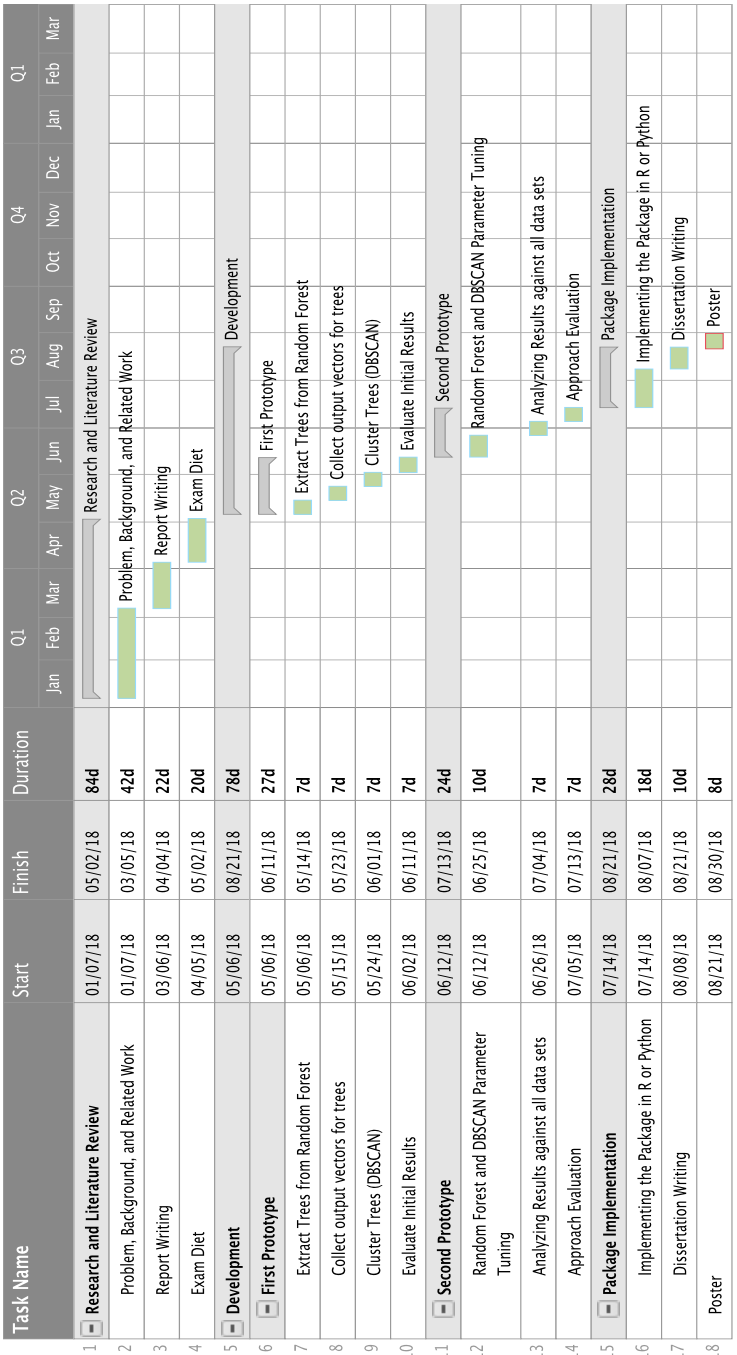

FIGURE 6.1: Project Plan.

Bibliography

[1] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.

[2] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[3] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*. Siam, 2007, vol. 20.

[4] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[5] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[6] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems," *J. Mach. Learn. Res*, vol. 15, no. 1, pp. 3133–3181, 2014.

[7] M. Robnik-Šikonja, "Improving random forests," in *European conference on machine learning*. Springer, 2004, pp. 359–370.

[8] A. Tsymbal, M. Pechenizkiy, and P. Cunningham, "Dynamic integration with random forests," in *European conference on machine learning*. Springer, 2006, pp. 801–808.

[9] P. Latinne, O. Debeir, and C. Decaestecker, "Limiting the number of trees in random forests," in *International Workshop on Multiple Classifier Systems*. Springer, 2001, pp. 178–187.

[10] M. Bader-El-Den and M. Gaber, "Garf: towards self-optimised random forests," in *International Conference on Neural Information Processing*. Springer, 2012, pp. 506–515.

[11] K. Fawagreh, M. M. Gaber, and E. Elyan, "Club-drf: A clustering approach to extreme pruning of random forests," in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Springer, 2015, pp. 59–73.

[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[13] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, no. 2, pp. 159–179, 1985.

[14] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," in *ACM Sigmod record*, vol. 28, no. 2. ACM, 1999, pp. 49–60.

[15] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 5, pp. 522–532, 1998.

[16] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data mining and knowledge discovery*, vol. 2, no. 3, pp. 283–304, 1998.

[17] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[18] B. Liu, *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media, 2007.

[19] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[20] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[21] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[22] Y. Yang, G. Wang, and K. He, "An approach for selective ensemble feature selection based on rough set theory," in *International Conference on Rough Sets and Knowledge Technology*. Springer, 2007, pp. 518–525.

[23] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[24] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[25] T. Bylander and L. Tate, "Using validation sets to avoid overfitting in adaboost." in *Flairs conference*, 2006, pp. 544–549.

[26] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[27] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[28] G. Williams, *Data mining with Rattle and R: The art of excavating data for knowledge discovery*. Springer Science & Business Media, 2011.

[29] S. Bernard, L. Heutte, and S. Adam, "A study of strength and correlation in random forests," in *International Conference on Intelligent Computing*. Springer, 2010, pp. 186–191.

[30] Y. Zhang, S. Burer, and W. N. Street, "Ensemble pruning via semi-definite programming," *Journal of Machine Learning Research*, vol. 7, no. Jul, pp. 1315–1338, 2006.

[31] G. Tsoumakas, I. Partalas, and I. Vlahavas, "An ensemble pruning primer," in *Applications of supervised and unsupervised ensemble methods*. Springer, 2009, pp. 1–13.

[32] D. D. Margineantu and T. G. Dietterich, "Pruning adaptive boosting," in *ICML*, vol. 97, 1997, pp. 211–218.

[33] P. Galinier and A. Hertz, "A survey of local search methods for graph coloring," *Computers & Operations Research*, vol. 33, no. 9, pp. 2547–2562, 2006.

[34] J. R. Koza, *Genetic Programming II, Automatic Discovery of Reusable Subprograms*. MIT Press, Cambridge, MA, 1992.

[35] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 18.

[36] B. Bakker and T. Heskes, "Clustering ensembles of neural network models," *Neural networks*, vol. 16, no. 2, pp. 261–269, 2003.

[37] G. Giacinto, F. Roli, and G. Fumera, "Design of effective multiple classifier systems by clustering of classifiers," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 2. IEEE, 2000, pp. 160–163.

[38] A. Lazarevic and Z. Obradovic, "Effective pruning of neural network classifier ensembles," in *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, vol. 2. IEEE, 2001, pp. 796–801.

[39] 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*. IEEE, 2009, pp. 302–307.

[40] H. Zhang and M. Wang, "Search for the smallest random forest," *Statistics and its Interface*, vol. 2, no. 3, p. 381, 2009.

[41] "Spiral software development methodology – the writepass journal." [Online]. Available: https://writepass.com/journal/2012/12/visualization-of-root-system-architecture/1-493/

[42] "Gnu general public license." [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html