HYBRID GENETIC RANDOM FOREST ALGORITHM FOR THE IDENTIFICATION OF ISI-INDEXED ARTICLES

MOHAMMADREZA MOOHEBAT

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2017

HYBID GENETIC RANDOM FOREST ALGORITHM FOR THE IDENTIFICATION OF ISI-INDEXED ATRICLES

MOHAMMADREZA MOOHEBAT

THESIS SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2017

UNIVERSITY OF MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: MOHAMMADREZA MOOHEBAT

Registration/Matric No: WHA110014

Name of Degree: DOCTOR OF PHILOSOPHY HYBID GENETIC RANDOM FORESTS ALGORITHM FOR THE INDETOFICATION OF ISI-INDEXED ARTICLES

Field of Study:

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date:

Subscribed and solemnly declared before,

Witness's Signature Date:

Name:

Designation:

ABSTRACT

In the past, the growth of human knowledge was slow and limited. For instance, when an innovation was created in the 18th century in the UK, it took several months or even years for the news to reach other parts of the globe. The advent of more modern technology and the current educational structure has accelerated this growth. Today, human knowledge grows every hour, and it is more accessible than ever before. This speed of knowledge growth highlights the role of scientific manuscripts in spreading this valuable knowledge around the world. We can trust that articles published in scientific journals concentrate on the cutting edge of knowledge. Presently, most of these journals are published in the English language, but many scientists are not proficient in English. This leads to a high rejection rate for publications and the loss of good research and talent due to the use of inappropriate terms or syntactical style. Reviewing scientific articles for high-quality journals is time-consuming (some cases take up to a year). Furthermore, many inexperienced authors do not follow the scientific writing style of high-quality journals (ISI journals) and get rejected after waiting several months. Having a tool that advises authors whether their writing style is following ISI journal standards can be helpful and save time. In this research study, I proposed an automated system for detecting the similarity of an article with well-written academic writing by noticing various term forms. I chose to advance a novel classification technique to recognize the existing academic patterns. However, it was first necessary to be confident that the classification technique could handle this job. Moreover, the result of this section was essential for me as a benchmark. After ensuring that the classification technique was able to accomplish this work, Hybrid Genetic Random Forests (HGRF) was introduced as a new ensemble classifier based on a Random Forest algorithm, but altered slightly with some innovations. In order to measure performance of the proposed algorithm, evaluation was done by several independent UCI datasets

and the results were compared with RF and some individual classifiers. In the final stage, it was tested by creating datasets for ISI and non-ISI papers and the result was promising. In most cases, HGRF successfully distinguished ISI articles from non-ISI articles.

ABSTRAK

Sebelum ini pertumbuhan pengetahuan manusia adalah perlahan dan terhad. Sebagai contoh, apabila inovasi yang telah diwujudkan pada 18 abad di UK, ia mengambil masa beberapa bulan atau tahun untuk mencapai beritanya ke bahagian lain di dunia. Kemunculan teknologi dan struktur pendidikan benar, mempercepatkan irama ini. Pada masa kini, pengetahuan manusia semakin meningkat setiap jam. Kelajuan ini menekankan peranan manuskrip saintifik untuk menyebarkan pengetahuan ini di seluruh dunia. Pertumbuhan pesat ini menarik kita dalam jumlah yang sangat besar manuskrip saintifik. Berikut adalah tempat yang berkualiti tinggi jurnal saintifik berguna. Kita boleh percaya bahawa artikel yang diterbitkan dalam jurnal itu, memberi tumpuan di pinggir pengetahuan. Masalahnya ialah bahawa kebanyakan jurnal-jurnal ini menyiarkan dalam bahasa Inggeris dan ramai saintis tidak profesional dalam bahasa Inggeris. Ini menyebabkan kadar penolakan yang tinggi untuk penerbitan dan kehilangan beberapa kajian yang baik dan bakat kerana menggunakan istilah yang tidak sesuai dan bentuk sintaksis kata-kata. Memproses artikel ISI memakan masa (beberapa kes sampai ke setahun). Ramai penulis tidak berpengalaman tidak mengikuti gaya penulisan saintifik jurnal ISI dan mendapatkan penolakan selepas menunggu beberapa bulan. Sedia ada alat yang menasihati penulis sama ada gaya penulisan yang mengikuti kepada jurnal ISI boleh membantu dan menjimatkan masa mereka. Dalam kajian ini, saya cuba untuk mencadangkan sistem automatik untuk mengesan persamaan artikel baru dengan perasan bentuk jangka berbeza artikel ISI sebagai salah satu jurnal diindeks terkemuka dan popular di dunia saintifik.Ia memutuskan untuk memajukan teknik pengelasan novel untuk mengiktiraf corak akademik yang sedia ada. Walau bagaimanapun, ia adalah perlu untuk menjadi yakin bahawa teknik pengelasan boleh mengendalikan kerja ini. Selain itu, hasil daripada seksyen ini adalah penting bagi saya

sebagai penanda aras. Selepas memberi jaminan bahawa teknik klasifikasi mampu menyelesaikan pekerjaan ini, Hybrid Genetic Random Forest (HGRF) diperkenalkan sebagai pengelas itere baru berdasarkan algoritma Forest rawak tetapi dengan beberapa inovasi. Ia memutuskan untuk memajukan teknik pengelasan novel untuk mengiktiraf corak akademik yang sedia ada. Walau bagaimanapun, ia adalah perlu untuk menjadi yakin bahawa teknik pengelasan boleh mengendalikan kerja ini. Selain itu, hasil daripada seksyen ini adalah penting bagi saya sebagai penanda aras. Selepas memberi jaminan bahawa teknik klasifikasi mampu menyelesaikan pekerjaan ini, Hybrid Genetic Random Forest (HGRF) diperkenalkan sebagai pengelas itere baru berdasarkan algoritma Forest rawak tetapi dengan beberapa inovasi. Dalam usaha untuk mengukur prestasi algoritma dicadangkan, ia dinilai oleh beberapa dataset UCI bebas dan hasilnya berbanding RF dan beberapa penjodoh bilangan individu. Di peringkat akhir, ia diuji dengan mewujudkan dataset untuk ISI dan kertas bukan ISI dan hasilnya adalah memberangsangkan. Dalam kebanyakan kes, HGRF dibezakan ISI dari artikel bukan ISI jayanya.

ACKNOWLEDGEMENTS

I would like to thanks to all people, who helped me during doing my PhD. There is no doubt that without them this journey could be more difficult.

I would like to thanks my supervisors. Dr. Ram Gopal Raj and Dr. Sameem Binti Abdul Kareem. I really appreciate Dr. Ram Gopal Raj's patience and support during these years. His guidance and suggestions were very helpful for me. Moreover, he kept me on the right track and motivated me to finish this journey. I should thanks to Dr. Sameem Binti Abdul Kareem, for supporting me when I was confused and helping me to find the correct way. Presence of her without doubt was a big milestone in my PhD progress. I learnt a lot from her.

Finally, I should say thanks to all my friends in University of Malaya for their support in difficult days and giving hope and motivation to me and my dear friends in Iran who did not leave me alone and all these years were in contact with me and made me happy and strong.

Dedication:

This thesis is dedicated to my beloved family. My parents, who have been supported me in any conditions and have been my best friends in whole of my life and myy dear wife, Sanaz, who gave me hope and was patient during this journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
BoW	Bag of Words
CART	Classification And Regression Tree
CHAID	CHi-squared Automatic Interaction Detection
ERP	Enterprise Resource Planning
GA	Genetic Algorithm
HGRF	Hybrid Genetic Random Forest
HTML	Hyper Text Markup Language
ID3	Iterative Dichotomiser 3
ISI	Institute of Scientific
KNN	K-Nearest Neighbors
ML	Machine Learning
NLP	Natural Language Processing
PAC	Probably Approximately Correct
RF	Random Forest
SVM	Support Vector Machine
Tf-IDF	Term Frequency-Inverse Document Frequency

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

1.1 Background

There is no debate that writing is an essential skill for everyone, irrespective of one's position in society. People who are active in science should not only be knowledgeable in basic writing, but must also be proficient in academic or scientific writing. The goal of scientific writing is to convey discoveries and information that were previously unknown to readers. According to Lindsay (2011), scientific writing should be precise, clear and brief. Nevertheless, writing in a scientific manner is not easy for many people, especially for those whose English is still developing. Proof of this is found in the research of Santos (1988), who conducted research to record the opinions of experienced professors about the writing of non-English speaking students (mostly Chinese and Korean students). It was found that 178 professors believed that the writing of these students suffered from serious lexical errors, and, based on their experience, was not publishable.

With the prevalence of computers and the Internet around the globe, academic writing has also found its way into the virtual world. Scientific journals now publish online and authors no longer need to worry about the long process of submission that was previously prevalent. A turning point occurred when computer science was able to help analyze digital content and, consequently, text analysis was born. Montes-y-Gómez et al. (2002) defined text analysis as "knowledge discovery in large text collections". Due to the importance of analyzing digital content, text analysis has experienced rapid growth in comparison to the past, when scripts were not digital. The scientific world has

also benefited greatly from text analysis; for instance, the detection of plagiarism, grammar checking and many other applications are the result of this innovation.

Text analysis constitutes a broad domain, and includes Machine Learning (defined as a process by which a machine gains the capability to solve a problem by examining examples or data (Michalski 1983)) and Natural Language Processing.[1] In this study, we chose to concentrate on the Machine Learning (ML) aspect by focusing on the classification[2] methods and their application in scientific text analysis. Text classification aims to assign classes to textual documents, in which the classes must be pre-defined (Finzen, Kintz & Kaufmann 2012; Ko and Seo 2009a; Lin and Hong 2011; Sudhamathy and Jothi Venkateswaran 2012; Thorleuchter, den Poel, and Prinzie 2010). The classification task belongs to the supervised learning category in ML.

The Institute for Scientific Information (ISI) was founded by Eugene Garfield in 1960. It was acquired by Thomson Scientific & Healthcare in 1992, (Thomson Corporation acquired ISI in 1991) and became known as Thomson ISI. It is now a part of the Intellectual Property & Science business of Thomson Reuters. ISI Journal Citation Reports on the Web (JCR Web) provides a systematic and objective means to critically evaluate the world's leading research journals. JCR Web citation data is drawn from approximately 7,000 journals covered by ISI, representing over 1,400 publishers worldwide in over 200 disciplines. Due to the meticulous and trusted process of selecting high-quality journals, publishing in ISI-indexed journals is honorable and a

[1] Short for natural language processing, a branch of artificial intelligence that deals with analyzing, understanding and generating the languages that humans use naturally in order to interface with computers in both written and spoken contexts using natural human languages, instead of computer languages.

[2] A classification is a structure imposed on the space of automata that groups cellular automata with related properties (Gutowitz 1990)

clear sign of proficiency in that scientific domain. Proof of this is the annual report of Academic Ranking World Universities, which is published by Shanghai Jiao Tong University. In this ranking, having researchers that publish ISI highly cited papers is one of the main metrics in ranking universities (Ranking Criteria and Weights, 2013).

This study is conducted with the aim of proposing a new classification technique for identifying ISI and non-ISI indexed papers. The methodology and system design are discussed in the related chapters. In the remainder of this chapter, we explain the problem statement and the objectives of this study. Finally, we mention the research methodology and respective research questions.

1.2 Motivation

Currently, the number of published ISI-indexed articles plays an important role in university ranking (Ranking Criteria and Weights, 2013). On the other hand, getting one's work published, especially in an ISI-indexed journal, is a prestigious but somewhat complex endeavor. A novel methodology and solid findings are mandatory for the acceptance of a paper. However, other aspects are also important, including the style of writing. The writing style needs to be clear, concise and comprehensible to the reader. It has been demonstrated that using an inappropriate writing style is the most common reason for rejection in scientific journals, as it creates conceptual barriers in the transmission of the authors' intentions (Bornmann, Weymuth & Daniel 2009). We believe that text mining can provide the ability to distinguish ISI from non-ISI articles.

1.3 Problem Statement

Publishing a scientific paper in high-ranking journals has never been an easy task for young scholars. One aspect of this achievement is the excellent research quality that these journals expect from authors. However, another aspect that many inexperienced

researchers underestimate is the importance of language and their lexical vocabulary in such journals. According to Meneghini and Packer (2007), the most common reason for rejection in scientific journals is conceptual barriers in transmitting an author's intentions. The growth of information technology in recent years resulted in new tools and techniques to help novice researchers, for instance, grammar-checking software, reference manager tools and editorial platforms. However, these endeavors are still not enough to assist an inexperienced writer who wants to create a scientific article for high-quality journals.

To elaborate this problem, I try to highlight the key aspects of the discussed issue. It will help us to know the problem better before proposing an appropriate solution for it.

PS1. **Existing Differences in the lexical domain of scientific scripts:** Susan Conrad (1996) conducted research comparing two different academic scripts. She chose common composition textbooks in the field of ecology and articles from ecology-related scientific journals. She found that each of these texts was different from the other. For example, scientific articles had a lower rate of type/token ratio than textbooks, but higher level information can be transmitted to reader. She noted that these kinds of differences in various scientific scripts would be a barrier for students' skill in scientific writing (Conrad, 1996). It is well known that the texture of different scientific scripts is different. However, it is not clear whether such a difference helps computers to differentiate them.

PS2 Application of classification techniques in discovering patterns among scientific scripts: Using Machine Learning and, more specifically, supervised learning in text analysis is not a new idea. However, analyzing academic text is only a small subset of text mining, which is why has received less attention (Coxhead, 2012).

Gaizauskas et al (2000) applied a data-extraction technique on biological scientific journals in order to extract information about enzymes and proteins from scientific papers. They extracted information about enzymes, metabolic pathways and protein structure. In another study, Szarvas (2008) used a classification technique for hedge categorization in biomedical articles. He argued that since facts or statements in a hedge or negated context typically appear as false positives, the proper handling of these language phenomena is highly important in biomedical text mining. Donaldson et al. (2003) used an SVM trained on the words for MEDLINE abstracts to identify abstracts containing information on protein–protein interactions, prior to curating this information into their BIND database.

Despite the existence of related research on the use of classification in the scientific domain, few of them focused on using classification for investigating the quality of scientific writing. One of the objectives of this study is to determine whether classification is an appropriate technique for distinguishing quality of scientific writings?

PS3. Competition between classifiers for better performance and precision: Naïve Bayesian is one of the oldest classification algorithms that still performs well (Wu & Vipin, 2009). Two decades after the creation of the Naïve Bayesian classifier and the evolution of various kinds of classifiers with different techniques, this area remains interesting for researchers and scientists who are active in machine learning. Why do we not use the best classifiers to finish the best classifier competition? The best answer to this question is given by Wolpert and Macready (1997) in the form of the No Free Lunch Theorem (NFLT). According to NFLT, universal optimization is impossible. In other words, you cannot acquire knowledge "for free" just by looking at training instances. Why not? Well, the fact is that the only things you

know about data is what you have seen as training data. Therefore, each dataset has special characteristics, making it impossible for all techniques to work well on the data. For example, a certain kind of basic neural network, the perceptron, is biased towards learning only linear functions and does not work properly if the data has a linear pattern (Ho & Pepyne, 2002). Therefore, different classifiers work for certain datasets and trial and error is usually the process by which the right sort of classifiers is discovered for the appropriate dataset.

For this reason, proposing a new classification algorithm that performs well with high accuracy for scientific text is possible. This is one of the problems that requires an answer, and this research seeks to accomplish that.

PS4. **Existing various syntactical forms in different text:** Eggins (1994) suggested a metric for gauging the lexicon densities of documents. According to his definition, lexical density is measured by dividing the number of content words (nouns, base verbs, adjectives and adverbs) by the running words (prepositions, congestion, auxiliary verbs, pronouns and determinants). Eggins (1994) stated that lexical density in academic manuscripts is higher than in other scripts. In another study, Halliday and his colleagues alleged that lexical density in every clause of an academic manuscript is two or three times greater than that of a normal manuscript (Halliday, Michael Alexander Kirkwood & Martin, 1993) .

Thus, firstly, it is important to understand if there is any clue to show the difference between using terms in high- and low-quality scientific articles. Secondly, we are curious to determine whether supervised learning can help us differentiate those writing samples from each other. Lastly, if classification is able to perform the classification task successfully, we seek to design a classification algorithm with higher performance

for distinguishing between high- and low-quality articles. Finally, we aim to discover which forms of the lexicon are more common in various types of writing.

1.4 Research Objectives

In this research, in order to propose a solution for the discussed problems, I have chosen to use supervised learning. The reason for choosing supervised learning is due to the successful background of classification techniques in the application of text analysis and scientific writing. This background is discussed extensively in Chapter 2.

For each step of this research, an objective is defined. These objectives are:

- To collect and create a reliable dataset for ISI and non-ISI index articles
- To investigate the ability of pattern discovery through machine-learning techniques for ISI articles
- To investigate the prevalent syntactical terms' forms in ISI and non-ISI articles.
- To develop a novel classification algorithm for distinguishing ISI articles from non-ISI papers with high accuracy
- To evaluate the performance of the improved algorithm

1.5 Research Methodology:

In the first step, to identify the research problem, it was necessary to conduct a comprehensive literature review. As Figure 1.1 depicted, the scope of this research was a subset of different research domains, such as machine learning, academic writing and text analysis.

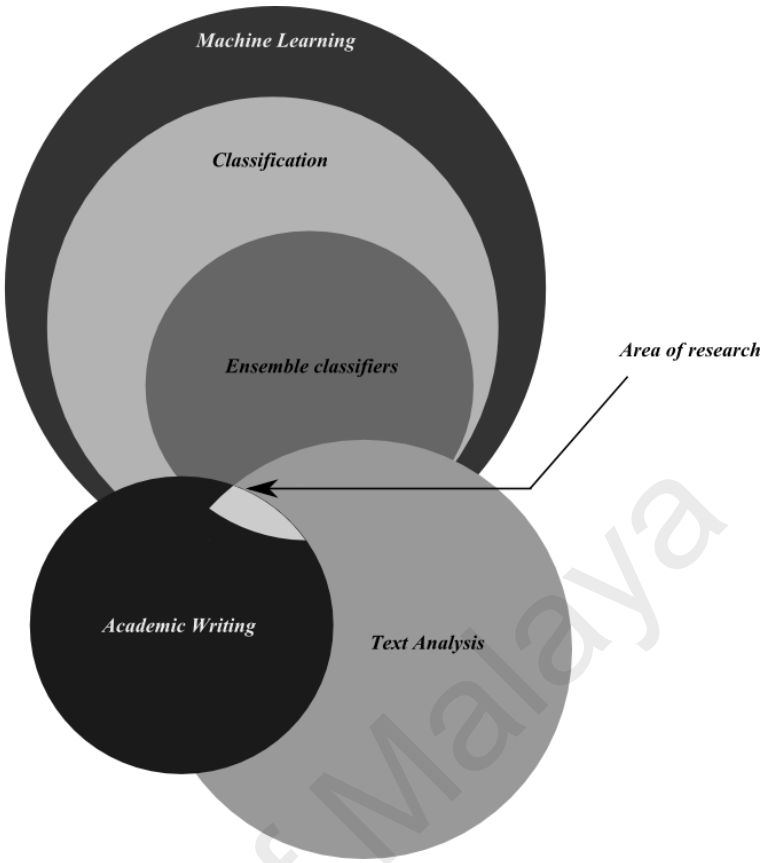


Figure 1.1: Research Domain

After this step, the problem statement became clear, leading to the definition of objectives of the research, which are presented in section 1.4. To begin the research, it was necessary to have a dataset on which to implement and test the machine-learning techniques. Chapter 3 has clarified this step and explains how it was done. Having the dataset in place paved the way to test some of the classic and standard algorithms. Chapter 4 details how this step was completed and presents the results. Analyzing and comparing syntactical forms of ISI and non-ISI is also included in Chapter 4 to highlight the differences between these two sets.

The promising results of Chapter 4 encouraged me to further improve the result and for this goal, the Hybrid Genetic Random Forest was introduced as a new classifier. Chapter 5 is dedicated to HGRF and its evaluation. Finally, Chapter 6 summarizes this

research, highlights the findings, and suggests future works. Figure 1.2 summarizes these steps and visually presents the phases of this research.

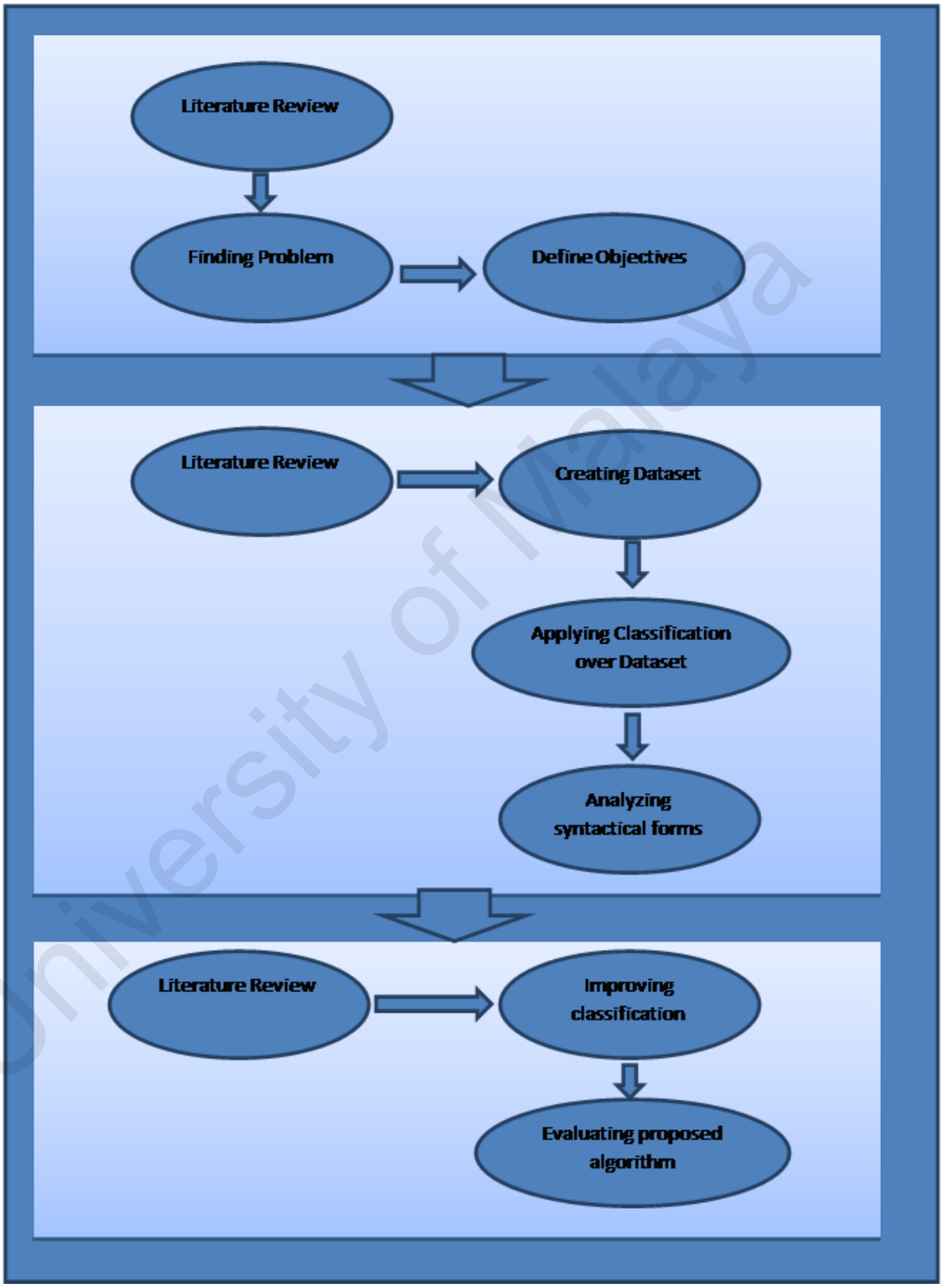


Figure 1.2: Research Methodology

1.6 Research Questions

- How should one collect and create a reliable dataset for ISI and non-ISI index articles?
- How is the classification performance for distinguishing ISI and non-ISI index articles?
- What is a novel and reliable classification algorithm with better performance for distinguishing ISI articles from non-ISI articles?
- What is the precision of the proposed algorithm?
- What are the common syntactical differences between ISI and non-ISI index papers?

1.7 Thesis Overview

Chapter 1 is an introduction to the thesis. First, the research motivation states why this research was interesting to pursue. The problem statement answers why it was necessary to accomplish this work. In the next stage, a brief background and the research objectives are presented. Following this, the research methodology summarizes the required steps of the research from beginning to end. Finally, we discuss the questions that we are seeking to answer.

In Chapter 2, the background of the research topic is explored in detail. We start with the history of supervised learning and introduce some base-line algorithms, such as K-nearest neighbor (KNN), Linear Regression, Logistic Regression, Support Vector Machine (SVM) and Decision Trees. Later, we move to Ensemble classifiers and introduce Bagging, Boosting, AdaBoost and Random Forest. One sub-section is assigned to the Genetic Algorithm, which is also used in the proposed Ensemble

classifier. Following this, text mining history and related techniques are covered and narrowed down to their application in academic scripts.

Chapter 3 demonstrates different elements of the proposed solution step by step. This makes it possible for other scholars to understand how this research was done and provides an outline for conducting similar research.

In Chapter 4, we run the experiment with classical classification techniques, such as KNN, Naïve Bayesian and SVM, to prove that a classification technique is an appropriate method for differentiating ISI and non-ISI articles.

Chapter 5 introduces Hybrid Genetic Random Forests (HGRF) as a novel classification algorithm and we investigate the performance of HGRF on standard UCI datasets. Finally, HGRF is tested on ISI and non-ISI data and its results are evaluated.

Chapter 6 summarizes the major contributions made in the thesis, followed by suggestions for future work.

CHAPTER 2: MACHINE LEARNING AND TEXT MINING

2.1 Introduction

In this chapter, some of the key topics related to this study are reviewed. The domain of this research is interdisciplinary and encompasses Text Mining and Machine Learning (ML). If we look at the whole picture, we can consider text mining as one of the subsets of ML. Mainly, there are two different, but related, topics within ML; *supervised learning* and *unsupervised learning*. In s*upervised learning*, we assume that we have some existing data that has been labeled and attempt to discover the hidden pattern in the data and assign the un-labeled data to existing labeled categories. On the other hand, in unsupervised learning, there is no labeled data and researchers must discover similarities or dissimilarities from the nature of existing data and assign them to different groups or clusters (Ko & Seo, 2009b). This research focuses on Supervised Learning, as it is believed that we have enough ISI articles online to use as labeled samples. Later, we talk about supervised learning application in text, and link it to the text analysis problem. Some of the essential methods of text analysis, which are used in this research, are also explained.

2.2 Machine Learning

Machine learning (ML) refers to algorithms that build analytical models from data. ML is one of the subsets of Artificial Intelligence. Machine learning has its roots in computational statistics, a discipline that also focuses on prediction-making through the use of computers. It also has a strong relationship with mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning has different and various applications, such as spam detection, computer vision and

many others. We can break down ML techniques into two main parts: Unsupervised learning (2.3) and Supervised learning (2.4) (Aggarwal & Zhai, 2012).

2.3 Unsupervised Learning

Unsupervised learning, or Clustering, organizes data instances into similar groups, called clusters, such that the data instances in the same cluster are similar to each other and data instances in different clusters are different from each other. Clustering is often called unsupervised learning, because unlike supervised learning, class values denoting an *a priori* partition or grouping of the data are not given (B. Liu, 2007).

2.4 Supervised Learning

Supervised learning, or classification, in machine learning is analogous to how humans learn in ordinary life, as it is a form of learning based on previous experience to acquire knowledge (Manning et al., 2008). Similarly, computers learn from data to gain new knowledge. In contrast to unsupervised learning, supervised learning uses labeled data. Generally, in supervised learning, a dataset is separated into two sets – training and test – in which the first is used for training the system, while the second is used for evaluating the accuracy of the generated trained model.

Machine learning is the process of learning a set of rules from instances (cases in a training set), or, informally, the responsibility of the classifier is to detect the class of new cases. In classification (supervised learning), the first step is to build the dataset. If an expert is available, they can advise which attributes or features should be considered. Otherwise, the simplest method is called "brute-force", which means considering all the existing features in the hope that the most informative and relevant attributes can be isolated. However, in most cases, the created dataset is not suitable for direct use in the machine learning process, as it usually includes noise and missing attribute values, and

therefore requires significant pre-processing (S. Zhang et al., 2002). Some of the common problems that cause the data to become impure include:

- Difficulty in detecting some of the noise
- Existing missing values
- Digressive input attributes are present in the data at hand

Detecting noise is one of the first activities that should be done in the pre-processing phase (Biau, 2012). Hodge and Austin (2004) presented contemporary methods for outlier (noise) detection. The issue of impure data is an unavoidable problem when dealing with most real-world data sources. In most cases, certain important issues need to be considered when processing unknown attribute values. One of the critical problems is known as the source of "unknown-ness", in which (a) a value is missing because it was forgotten or lost, (b) a certain feature is not applicable for a given instance (e.g., it does not exist for a given instance), or (c) for a given observation, the designer of a training set does not care about the value of a certain feature (so-called "don't-care values"). Based on the source of the unknown-ness, researchers have a number of techniques to handle missing data, such as ignoring missing data, imputing missing data with a replacement value, imputing the missing data and accounting for the fact that these were imputed with uncertainty, and using statistical models to allow for missing data, making assumptions about their relationships with the available data (Batista & Monard, 2003). For example, KNN imputation considers *K* neighbors of the missing value and uses their average amount as a substitution (de Souto, Jaskowiak & Costa, 2015).

Supervised learning can be categorized in different ways; for instance, it can be based on the techniques being used, such as regression-based classifiers or probabilistic

classifiers. However, for simplicity, the classifiers are placed according to two categories that are more general – individual and ensemble classifiers – each of which is described with examples in the following sections (Meinshausen, 2006).

2.5 Individual Classifiers

By individual classifier, we mean the classifiers that perform the classification task without helping other existing classifiers. The method could have originated from probabilistic models, such as Naïve Bayesian, or pure mathematical models, such as logistic regression and Support Vector Machine. In this section, we introduce some of the most well-known individual classifiers (Rodríguez, Kuncheva & Alonso, 2006).

2.5.1 K-Nearest Neighbors (K-NN)

The K-NN method follows a straightforward and effective idea in classification by testing each sample in a given vector space with the majority class of its *K*-nearest neighbors. *K*, as the only parameter of this classification method, can have various values. The nature of K-NN is based on this greedy fact that the new sample should be similar to its neighbors. K-NN measures similarity by the Euclidean or Cosine distance between different vectors. The decision boundary in K-NN is defined by *Voronoi tessellation*, which is a set of *Voronoi cells* (Figure 2.1). Voronoi cells are the polygon space around each training sample that consists of all the points close to it, while the decision boundary is formed by concatenating tiles that belong to the same classes. The border between the two different classes is called the decision boundary. The points on the decision boundary have the same distance from both training examples. K-NN is simple and flexible at the same time. The decision boundary can be shaped in very complex forms. However, the drawback of K-NN is its intrinsic sensitivity to outliers. To solve this problem, K-NN typically uses more than one neighbor (K>1) and the class

that has more cases in that particular area is selected for the new sample in the test set (Wu et al., 2007).

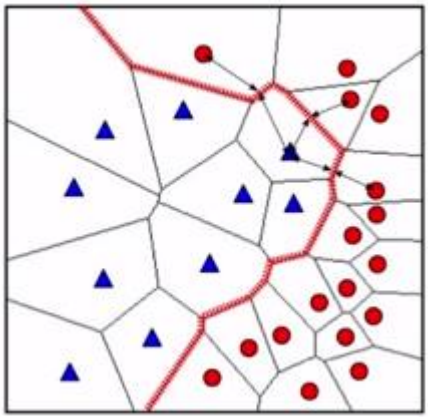


Figure 2.1: Voronoi cells in KNN algorithm

The role of the *K* value is very important, and, when changed, can lead to different results. Choosing too large of a value can include all the neighbors, which changes the K-NN to a poor classifier, while a small *K* makes the K-NN very sensitive to outliers, which causes overfitting of the training data. The best method to determine which *K* is most suitable for solving the problem is cross-validation (2.8).

The main advantage of the K-NN algorithm is its ease of implementation; it does not assume anything about the data and usually has acceptable performance. For example, Kwon and Lee (2003) reported acceptable results for K-NN concerning the classification of Korean websites. However, the drawback is that K-NN is computationally expensive. The time required to calculate the distance between different samples is *O (ND); n* is the number of cases in the training set and *d* represents the dimensions of the samples.

To make K-NN clearer, I explain it through a numerical example. Consider that some data has been collected in a survey from some people to determine their opinion about a

special type of paper tissue (Table 2-1). Two features have been measured: Acid Durability and Strength. All samples belong to two classes (Good and Bad quality).

Table 2.1: Paper tissue survey data

X1= Acid Durability	X2= Strength	Y= Class
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Assume that we get another result (X1=3, X2=7) without any label and decide to classify it as Good or Bad. KNN tries to find the distance of the new case with the older data, as described in Table 2.2. This example finds the distance between two cases with the Euclidian distance formula ($\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$).

Table 2.2: Calculating square distance of new case to other examples

X1= Acid Durability	X2= Strength	Square distance to query instance
7	7	$(7-3)^2 + (7-7)^2$=16
7	4	$(7-3)^2 + (4-7)^2 = 25$
3	4	$(3-3)^2 + (4-7)^2 = 9$
1	4	$(1-3)^2 + (4-7)^2 = 13$

If we suppose that *K=3* and sort the existing values according to their distance from the new sample, by considering the 3 nearest cases and majority voting, we can assign the new case to one of the classes. Finally, KNN classifies the new case as Good (Table 2.3).

Table 2.3: KNN results on paper tissue data

X1= Acid Durability	X2= Strength	Square distance to query instance	Rank	Included in 3 neighbors	Class
7	7	$(7-3)^2+(7-7)^2$=16	3	Yes	Bad
7	4	$(7-3)^2+(4-7)^2=25$	4	NO	-
3	4	$(3-3)^2+(4-7)^2=9$	1	Yes	Good
1	4	$(1-3)^2+(4-7)^2=13$	2	Yes	Good

2.5.2 Naïve Bayesian

The Naïve Bayesian (NB) classifier is a probabilistic learning algorithm that is derived from Bayesian decision theory (Mitchell 1997). The probability of a message *d* being in class *c*, *P (c | d),* is computed as:

$$P(c|d) \propto P(c)\prod_{K=1}^{m} P(t_k|c) \tag{2.1}$$

where $P(t_k|c)$ is the conditional probability of feature t_k occurring in a message of class c, and P (c) is the prior probability of a message occurring in class c.

$P(t_k|c)$ can be used to measure how much evidence t_k contributes that c is the correct class (Manning et al., 2008). In email classification, the class of a message is determined by finding the most likely or maximum a posteriori (MAP) class c_{MAP}, defined by Equation 2.2.

$$C_{MAP} = \arg max_{c\in(c_l,c_s)} P(c|d) = \arg max_{c\in(c_l,c_s)} p(c)\prod_{k=1}^{m} P(t_k|c) \tag{2.2}$$

Since Equation 2.2 involves the multiplication of many conditional probabilities, one for each feature, the computation can result in a floating point underflow.

In practice, the multiplication of probabilities is often converted to an addition of logarithms of probabilities and, therefore, the maximization of the equation is alternatively performed by Equation 2.3

$$C_{MAP} = \arg max_{c\in(c_l,c_s)} \left[\log P(c) + \sum_{k=1}^{m} logP(t_k, c) \right] \quad (2.3)$$

All model parameters, i.e., class priors and feature probability distributions, can be estimated with relative frequencies from the training set D. Note that when the class and message features do not occur together in the training set, the corresponding frequency-based probability estimate will be zero, which would make the right-hand side of Equation 2.3 undefined. This problem can be mitigated by incorporating some method of correction, such as Laplace smoothing, in all probability estimates. NB is a simple probability-learning model and can be implemented very efficiently with a linear complexity. It applies a simplistic or naive assumption that the presence or absence of a

feature in a class is completely independent of any other features (Wu et al., 2007). Despite the fact that this oversimplified assumption is often inaccurate (in particular for text domain problems), NB is one of the most widely used classifiers and possesses several properties (Zhang, 2004) that make it surprisingly useful and accurate.

2.5.3 Linear Regression

Linear regression is a very common method for prediction and forecasting in statistics (James et al., 2013). In this model, as always, we have training data, such as

$X^T = \{X_1, X_2, \dots, X_m\}$, and we try to determine any trends with a linear function that is called the *learning function*. Parameters θ_j are coeffients of X in the equation (e.g., $Y = \theta_0 + \theta_1 X_1 + \theta_2 X_2$), for which the complete form would be in Equation 2.4.

$$Y = \theta_0 + \sum_{j=1}^{m} X_j \theta_j \tag{2.4}$$

$\boldsymbol{\theta_0}$ is named bias. If we assume that X_0 is equal to Constant 1, we can rewrite the above formula with numeric algebraic symbols in a compact form (Equation 2.5):

$$Y = X^T \theta \tag{2.5}$$

Y is *m×1* vector, X^T is the transposed matrix with *m×(n+1)* dimensions and θ is *(n+1)×1*.

The question concerns how we measure the accuracy of the estimated learning function. In other words, how do we fit the model? There are different techniques to determine this, but one of the most popular is the *least squares error* (Equation 2.6).

$$J(\theta) = \sum_{i=1}^{m} (Y_i - X_i^T \theta)^2 \tag{2.6}$$

The goal is to have a small $j(\theta)$. $j(\theta)$ shows the error rate of the learning function with respect to the real output. The best learning function produces the least squares error. $j(\theta)$ is a quadratic function and a minimum exists for it.

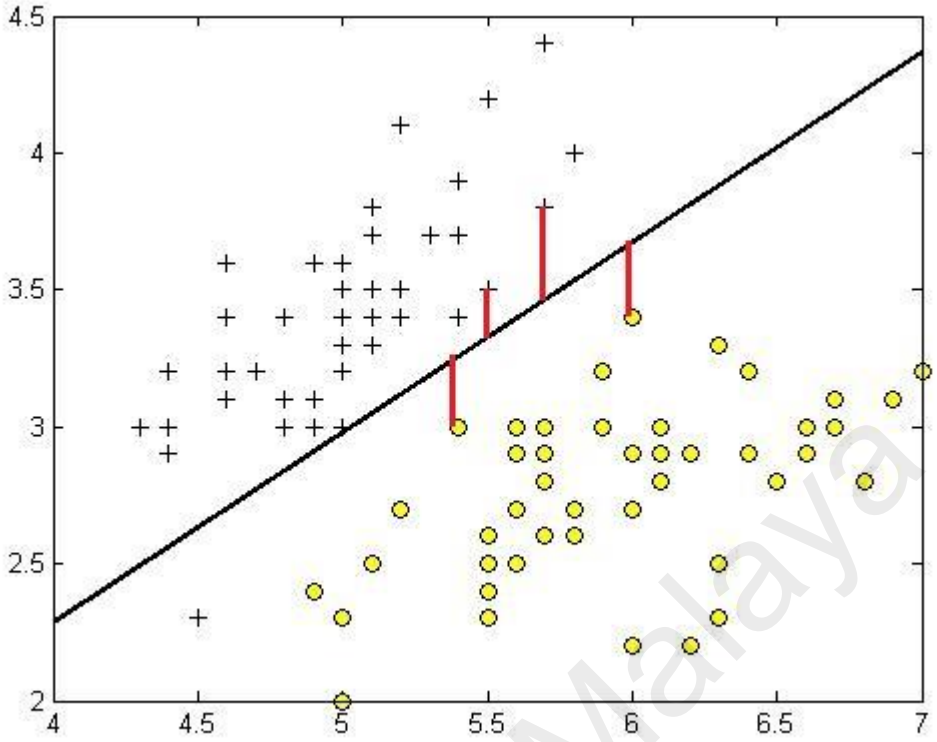


Figure 2.2: Least square error

If we rewrite Equation 2.6 in numeric form, we have:

$$J(\theta) = (y - \theta X)^T (y - \theta X) \tag{2.7}$$

After obtaining the derivation with respect to θ, the equation becomes:

$$X^T (y - X\theta) = 0 \tag{2.8}$$

Solving this equation leads us to:

$$\theta = (X^T X)^{-1} (X^T y) \tag{2.9}$$

This result is only accurate if $X^T X$ is not a singular matrix. In the classification context, linear regression works by defining a threshold. For instance, if we assume that the learning function is identified, we will have the membership function:

$$G = \begin{cases} 1 & if\ y > 0.5 \\ 0 & if\ y < 0.5 \end{cases} \tag{2.10}$$

Generally, the threshold is calculated by minimizing the sum of the root square error (James et al., 2013). In this example, the threshold is 0.5. If the result of the learning function is larger than 0.5, it belongs to class "1"; otherwise, it belongs to class "0". The sample of linear regression is shown graphically in Figure 2.3. The points above the line belong to one class, while those below belong to another (James et al., 2013).

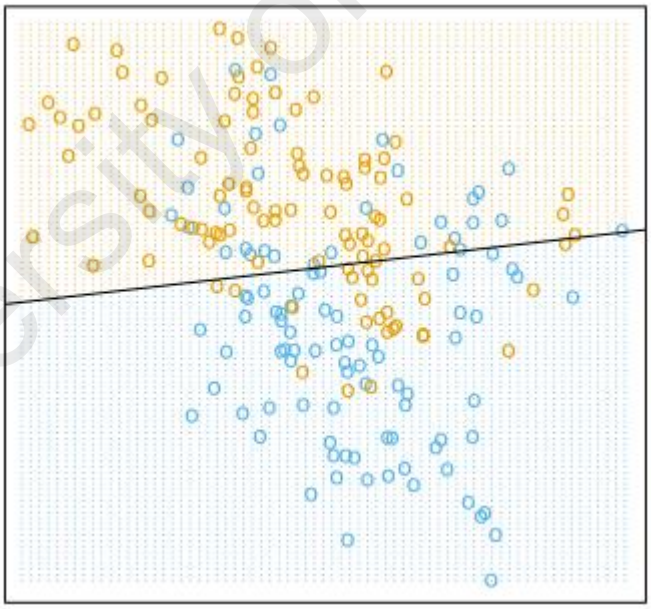


Figure 2.3: Linear Regression

2.5.4 Logistic Regression

Although linear regression is a prevalent tool in the world of statistics, it is not strong enough in classification applications. For this reason, the evolved form of this

algorithm, logistic regression, is used. In binary classification, we want to predict to which group the new case belongs (labeled with 0 or 1). However, in linear regression, the output span is unlimited (James et al., 2013). To limit *g(x)* between 0 and 1, the Sigmoid or Logistic function is used, as defined in Equation (2.11):

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.11}$$

The range of this function lies between 0 and 1, as presented in Figure 2.4:

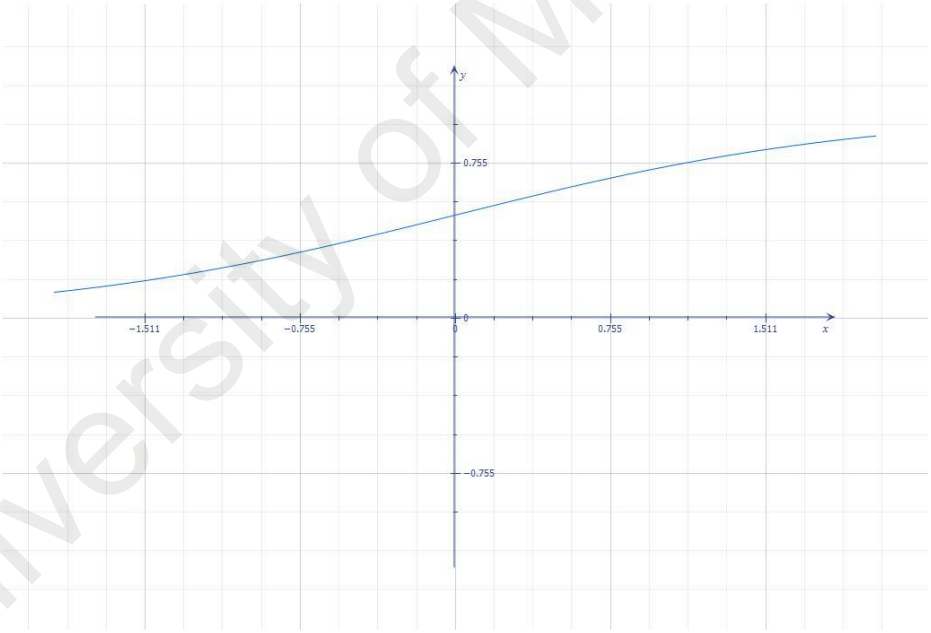


Figure 2.4: Shape of g(z) function

By replacing power z with the equation of linear regression $h_\theta(X) = -\theta^T X$, the equation becomes

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T X}} \tag{2.12}$$

If $h_\theta(x) > 0.5$ and equivalently $\theta^T X > 0$, y would belong to class "1"

If $h_\theta(x) < 0.5$ and equivalently $\theta^T X < 0$, y would belong to class "0"

For example, if $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$ and $\theta = [-2, 0, 0, 1, 1]^T$ for class y=1, the prediction formula converts to $x_1^2 + x_2^2 \geq 2$ (Figure 2.5).

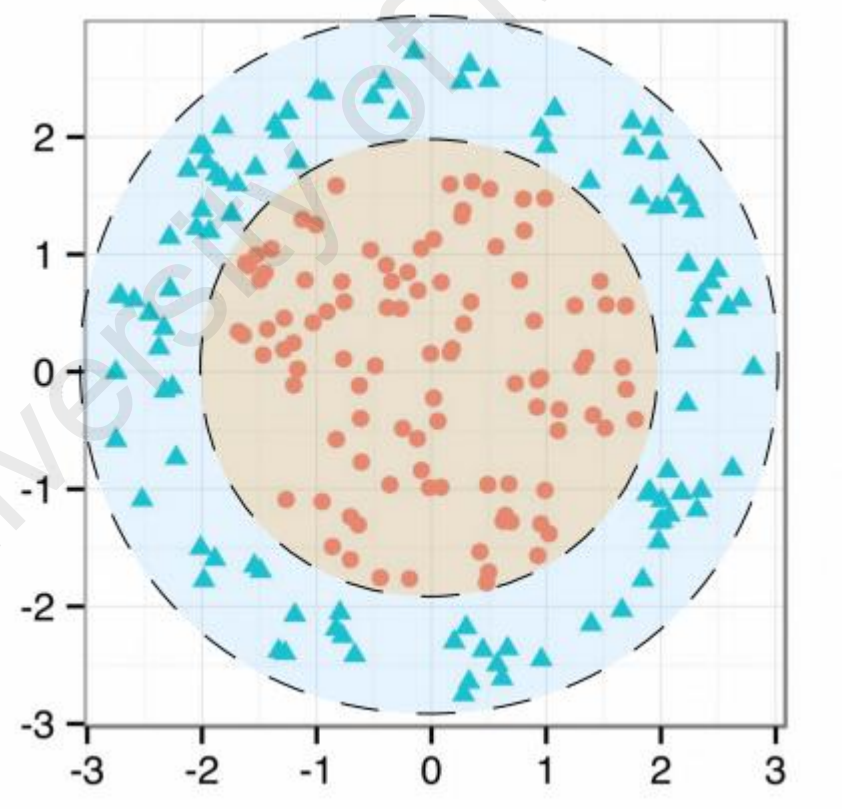


Figure 2.5: Logistic Regression prediction for nonlinear area

To fit the logistic regression, we use the cost function. The cost function will let us figure out how to fit the best possible straight line to our data.

$$J(\theta) = -\frac{1}{m}(\sum_{i=1}^{m}(y^{(i)}\log(h_\theta(x^i) + (1 - y^i)\log(1 - h_\theta(x^i))) \quad (2.13)$$

2.5.5 Support Vector Machine (SVM)

Vapnik & Lerner (1963) introduced the SVM method as a subset of the linear supervised learning classification method. Since that time, many researchers have tested, implemented and developed this algorithm in different applications (Liu et al., 2011; Orrù et al., 2012; K. Kim & Lee, 2014). This method tries to classify sample cases in each training set by calculating the optimal hyperplane between two different classes. The best hyperplane is one that has the biggest distance between two samples. In Figure 2.6, two different hyperplanes are shown. The one with a wider margin would be the choice of SVM. The cases on the border of the margin are called support vectors.

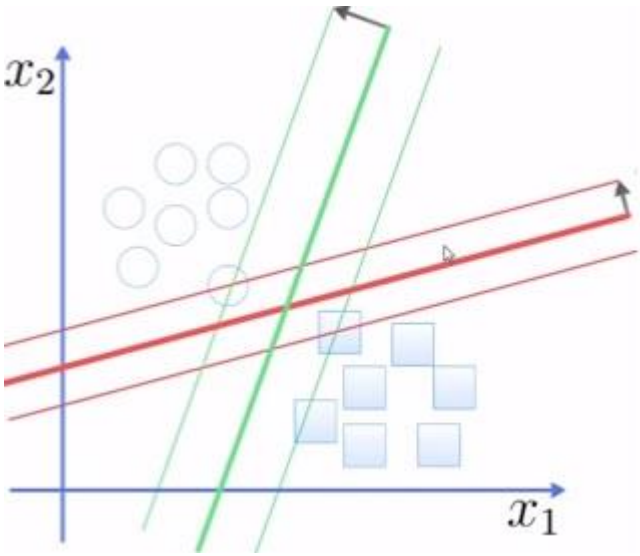


Figure 2.6: Comparison between different margins in SVM

Binary SVM has two classes (*y=1 and y=-1).* After finding the decision boundary on the best hyperplane ($g(x) = x_i w^T + b$), if $g(x) \geq 1$, x belongs to class *y=1* and if $g(x) \leq -1$, it belongs to class *y=-1*. The value of *w* is a function of $\vec{w} = \sum_{i=0}^{N} \alpha_i x_i y_i$ with the condition of $\sum_{i=0}^{N} \alpha_i y_i = 0$.

As an example for simple data set ({(1,1), (2,3)}) with given $\vec{w} = (1,2)$ which is illustrated in Figure 2.7, The optimal decision surface is between these two points with this equation:

$$y = x_1 + 2x_2 - 5.5$$

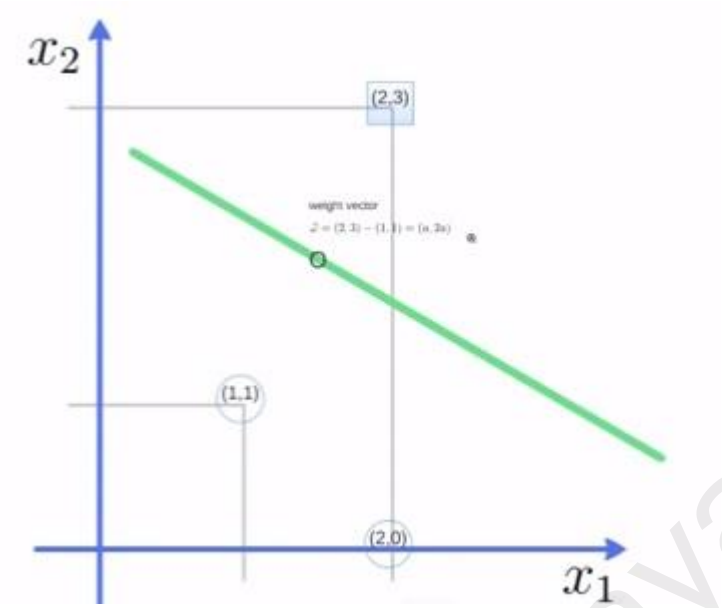


Figure 2.7: Simple SVM example

To find the optimal *w*, we know that $\vec{w} = (a, 2a)$; therefore, it should work in this equation $sign\big(y_i(\vec{w}^T x_i + b)\big) > 1$ with the given points. Therefore, we will have:

$$a + 2a + b = -1$$

$$2a + 6a + b = 1$$

Therefore, a=2/5 and b=-11/5, so the optimal $\vec{w} = (\frac{2}{5}, \frac{4}{5})$ and $b = \frac{-11}{5}$.

In 1992, Vapnik proposed a non-linear SVM model (Boser, Guyon & Vapnik 1992), and, subsequently, a SVM model using a soft margin algorithm (Vapnik & Cortes 1995). Based on Meyer's work, the SVM model has demonstrated acceptable performance in comparison to other methods, especially when used for classification (Meyer, Leisch & Hornik 2003).

2.5.6 Decision Tree

The decision tree is similar to the human decision system. Most people can understand and interpret it very easily (James et al., 2013). This technique has applications in both classification and regression problems. The state space of the problem is divided into rectangular regions in the training set, based on certain conditions concerning the selected features. The tree grows in this way until we obtain the terminal nodes or leaves, which determine the class probability. Figure 2.8 is a visualized decision tree for text classification (Apté et al., 1994).

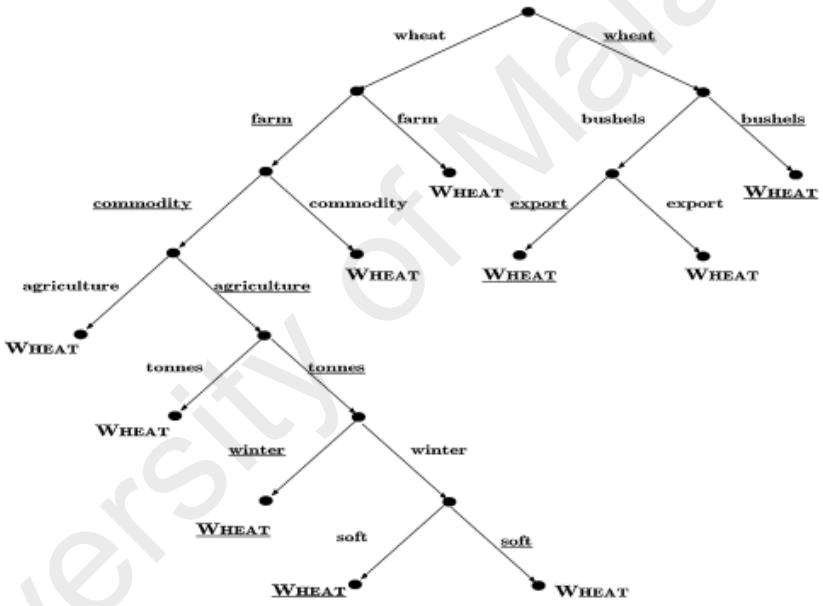


Figure 2.8: Decision tree application for text classification

In the text-mining area, internal nodes show the terms that have appeared in the document, while the leaves represent the class of the document. In creating the tree, the priority for selecting a given feature is important and can make the tree longer or shorter. For solving this problem and detecting the priority of expanding which node, the Gini index and Information Gain are commonly used. Equation 2.14 describes the Gini index. p_{mk} is the probability that class *k* has appeared in leaf *m*.

$$G = 1 - \sum_{k=1}^{K} {p_{mk}}^2 \tag{2.14}$$

Equation 2.15 shows Information Gain, which is another technique for deciding which feature is more informative for further processing.

$$D = -\sum_{k=1}^{K} p_{mk} log\ p_{mk} \tag{2.15}$$

Figure 2.9 depicted another application of the decision tree approach for the classification of different kinds of Iris according to different specifications of collected samples. Decision trees are ideal for visualization because they are similar to the human decision-making process. Figure 2.10 shows how the decision tree breaks down the Iris dataset to classify different samples into five different classes and simplify this process based on some simple rules (James et al., 2013).

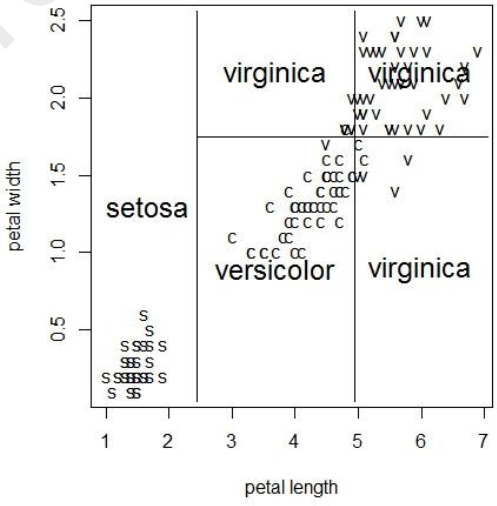


Figure 2.9: Dividing the space of the Iris database using a decision tree

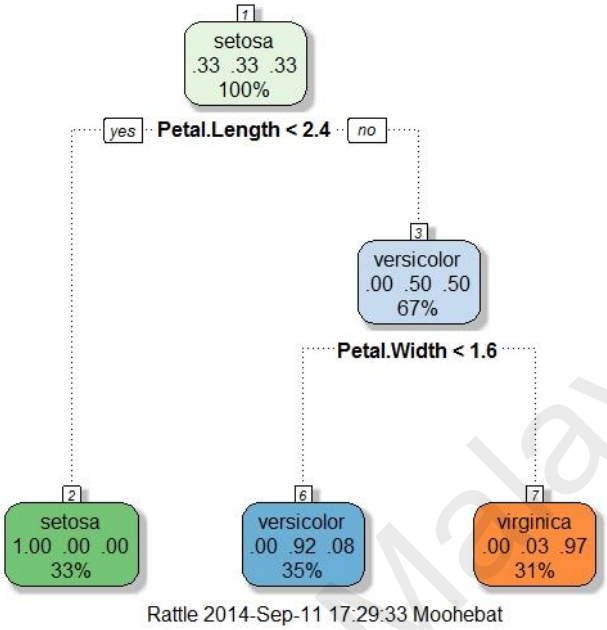


Figure 2.10: Decision tree rules for the Iris dataset

Despite the flexibility of the decision tree, it suffers from various problems. The most serious one is overfitting. Overfitting is defined as following the noise (James et al., 2013). In other words, the algorithm tries to become too specific and becomes dependent on the data of the dataset, instead of real existing patterns. In decision trees, overfitting can occur if we do not reduces the size of decision trees by removing sections of the tree that provide little power to classify instances (It is called Prunning). The estimation of uncertainty is difficult and the results can be variable based on the data and features that are selected. That is why most scholars prefer using other types of classification methods that are created based on the tree, such as random forest and boosting.

2.6 Ensemble classifiers

"*Unity is strength*" is the fundamental concept for ensemble classifiers. The main idea of ensemble classifiers is to combine a few weak classifiers and make a strong one. Schapire (1990), who is considered the father of ensemble classifiers, proved that combining various weak classifiers through boosting makes a strong classifier. Since their introduction, ensemble classifiers have received considerable attention from different researchers. Ensemble classifiers are referred to by various names, such as Multiple Classifier System (MSC), consensus aggregation, and committees of classifiers, etc. One of the reasons for the popularity of ensemble classifiers is that they have the potential to work with either a large or small amount of data. In the case of data scarcity, bagging and bootstrapping can also be useful, whereas when there is a large volume of data, ensemble classifiers can partition the data and merge the decision for each partition at the end. Moreover, it has been proven analytically that ensemble classifiers can outperform individual classifiers under certain conditions (Tumer & Ghosh, 1996).

Various types of ensemble classifiers have been created by concentrating on at least one of the following issues: the way that individual classifiers can interconnect with each other (system topology), the method for choosing the most valuable classifier (Ensemble design) and how to combine the result of the selected classifiers to obtain the best outcome (Fuser design) (Woźniak, Graña & Corchado, 2014). According to Polikar (2006), the main approach for creating an ensemble classifier is to bring more diversity to the selected classifiers. The following introduces some of most common ensemble classifiers.

2.6.1 Bagging

Bagging stands for bootstrap aggregating. Bagging is one of the most primitive ensemble-based algorithms. However, it is very intuitive and simple to implement. At the same time, surprisingly, it also performs well. Diversity in bagging is gained by using bootstrapped replicas of the training data: various training data subsets are randomly drawn – with replacements – from the entirety of the training data. Each training data subset is applied to train a different classifier of the same type. Finally, individual classifiers are then mixed by taking a majority vote of their decisions. For any new instance, the predicted class chosen by the majority of classifiers is the ensemble decision. Bagging is especially successful when the available data has a limited size. To ensure that there are adequate training samples in each subset, relatively large chunks of the cases (75% to 100%) are drawn into each subset. This leads individual training subsets to overlap automatically, with many of the similar instances appearing in most subsets, and some instances occurring multiple times in a given subset. In order to ensure diversity under this procedure, an approximately unstable model is used, so that adequately different decision boundaries can be retrieved for small disturbances in various training datasets. As stated before, neural networks and decision trees are good options to achieve this goal, as their uncertainty can be controlled by the selection of their free parameters. Algorithm 1 shows how bagging works.

Algorithm 1: Bagging

Training phase

1. Initialize the parameters
    - $D = \emptyset$, the ensemble.
    - L, The number of classifiers

2. For k=1,…,L
    - Take bootstrap sample S_k from Z.
    - Build a classifier D_k using S_k as the trainer set.
    - Add the classifier to the current ensemble, $D = D \cup D_k$
3. Return D.

Classification Phase

4. Run $D_1, D_2, \ldots, D_L$ on the input x.
5. The class with the maximum number of votes is the label for x

2.6.2 Boosting

Schapire (1990) proved that a weak learner, an algorithm that produces classifiers that barely work better than random guessing, can be converted into a strong learner that creates a classifier that is able to correctly classify all but a randomly small fraction of the instances. Boosting is one of the most important developments in the recent history of machine learning. Similar to bagging, boosting also produces an ensemble of classifiers by resampling the data, which is then incorporated by majority voting. However, the similarity ends here. In boosting, resampling is conducted to provide the most informative training data for each successive classifier. In essence, boosting combines three weak classifiers: the first classifier C1 is trained with a random subset of the existing training data. The training data subset for the second classifier C2 is chosen as the most informative subset given by C1. That is, C2 is trained on training data for which only half is correctly classified by C1, while the other half is misclassified. The third classifier, C3, is trained with instances in which C1 and C2 disagree. The three classifiers are incorporated through a three-way majority vote. The algorithm is shown in detail in the following code (Algorithm 2). Schapire (1990) proved that the error of this three-classifier ensemble above is limited, and that it is less than the error of the

best classifier in the ensemble, based on each classifier having an error rate of less than 0.5. For a two-class problem, an error rate of 0.5 is the least one can expect from a classifier, as an error of 0.5 amounts to random guessing. Hence, a stronger classifier is generated from three weaker classifiers. A strong classifier in the strict PAC learning sense can then be created by recursive applications of boosting.

Algorithm 2: Boosting

Input:

- Training data S of size N with correct label $w_i \in \Omega = \{w_1, w_2\}$
- Weak learning algorithm

Training

1. Select $N_1 < N$ patterns without replacement from S to create data subset S_1
2. Call weak learner and train with S_1 to create classifier C_1
3. Create dataset S_2as the most informative dataset, given C_1, such that half of S_2 is correctly classified by C_2, and the other half is misclassified. So:
    - Flip a coin. If Heads, select samples from S and present them to C_1until the first instance is misclassified. Add this instance to S_2.
    - If Tails, select samples from S and present them to C_1until the first one is correctly classified. Add this instance to S_2.
    - Continue flipping the coin until no new pattern can be added to S_2
4. Train the second Classifier $C_2$with S_2
5. Create S_3 by selecting those instances that C_1 and C_2 disagree on. Train the third classifier C_3 on the S_3 dataset.

Test, given the test instances X

1. Classify X with C_1, C_2. If they agree on the result, this is the final result.
2. Otherwise, classify X with C_3 and consider it to be the final result

2.6.3 AdaBoost

AdaBoost was introduced by Schapire and Freund in 1997 (Freund & Schapire, 1997). We can consider AdaBoost as the general version of the Boosting algorithm. Since it was first introduced, different versions of AdaBoost have been proposed for dealing with multiple classification and regression problems, such as AdaBoost.M1 and AdaBoost.R. In this section, we discuss the AdaBoost.M1 mechanism, because it is less complicated and attaches equal importance to each training example (Cameron-Jones, 2001). Adaboost combines weak classifiers through weighted majority voting. At the beginning of the process, all of the instances have an equal chance for selection. The distribution is updated during the iteration in order to ensure that misclassified cases have a higher chance of being reselected as the training case of the next iteration. In this way, AdaBoost focuses on difficult cases. The algorithm of AdaBoost is depicted in Algorithm 3. As shown, in the initial stage, all the instances have an equal chance for selection. However, in each iteration, the error is calculated based on the summation of the misclassified cases. If this rate exceeds 0.5, the classification fails. Otherwise, the normalized error (β_t) is calculated according to Algorithm 3. In the next step, and with the help of the normalized error, the distribution weight is recalculated and AdaBoost prepares for cases that are more difficult. This process repeats until the assigned iteration number (T) finishes. In contrast to bagging or boosting, AdaBoost uses a weighted voting system. The idea is simple, the classifier with better performance during the training gains more weight. Schapire and Freund (1997) chose $1/\beta_t$ as the measurement. Because this number could be a large number, they decided to apply $\log 1/\beta_t$. Figure 2.11 depicts how AdaBoost works (R. Polikar 2006).

During years, varieties of AdaBoost techniques have been created. For instance, AdaBoost.M2 which is proposed by Freund and Schapire (1997), does not limit the algorithm to maintain a weighted error less than half, while AdaBoost.R extends the boosting approach to regression-type problems.

.

There are more heuristic varieties that change either the distribution update rule or the combination rule of the classifiers. For instance, AveBoost averages the distribution weights to make the errors of each hypothesis as uncorrelated as possible with those of the previous ones (Dietterich 2000), whereas Learn++ makes the distribution update rule contingent on the ensemble error (instead of the previous hypothesis' error), to allow for efficient incremental learning of new data that may introduce new classes (Polikar et al., 2001).

Algorithm 3: AdaBoost.M1

Initialize $D_1(i) = \frac{1}{N}$, i=1,…,N

Do for t=1,2,…,T:

1. Select a training data subset S_t, drawn from the distribution D_t
2. Train WeakLearn with S_t, receive hypothesis h_t

Calculate the error of

$$h_t: \varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i)$$

If $\varepsilon_t > \frac{1}{2}$, Abort

Set $\beta_t = \varepsilon_t/(1-\varepsilon_t)$

3. Update distribution

$$D_t: D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & if\ h_t(x_i) = y_i \\ 1, & otherwise \end{cases}$$

Where $Z_t = \sum_i D_t(i)$ is a normalization constant chosen so that D_{t+1}becomes a proper distribution function.

Test- Weighted Majority Voting, given an unlabeled instance x,

4. Obtain total vote received by each class

$$V_j = \sum_{t:h_t(x)=\omega_j} \log\frac{1}{\beta_t}, j = 1, \dots, C.$$

5. Choose the class that receives the highest total vote as the final classification.

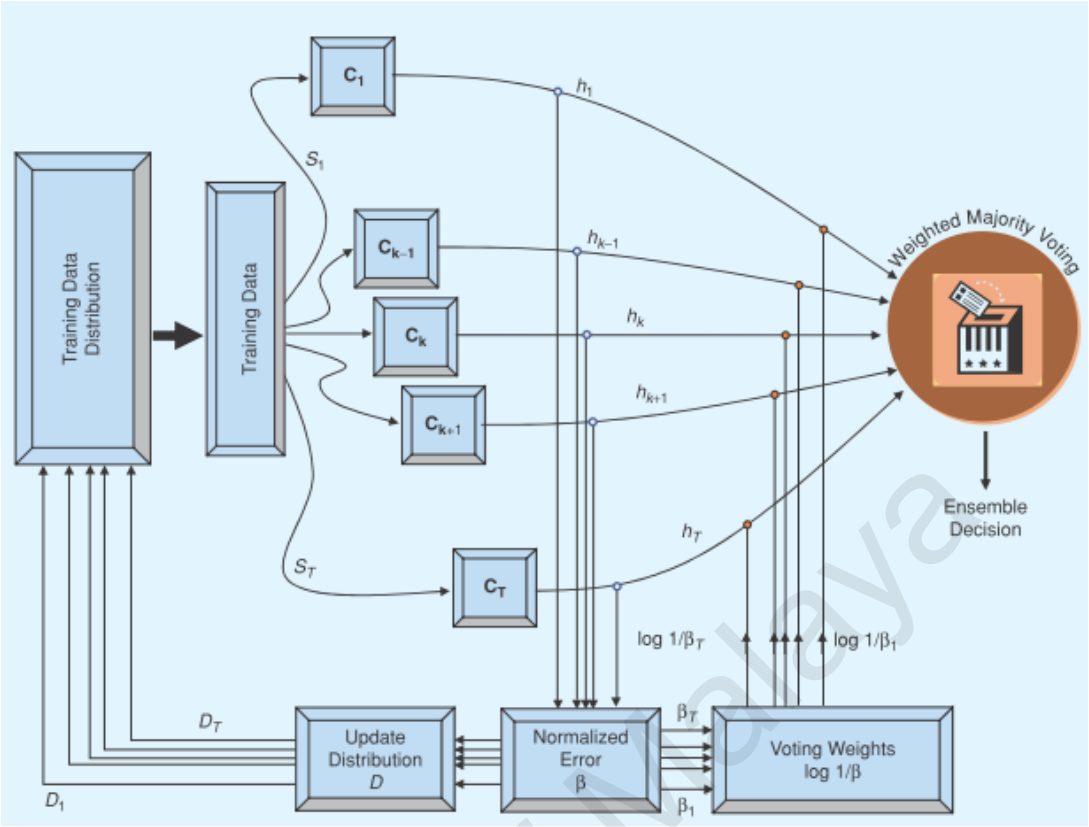


Figure 2.11: AdaBoost

2.6.4 Random Forests

When speaking about Random Forest, practically, we should categorize it as a special kind of Bagging algorithm. The main idea behind Bagging as an ensemble classifier is the way in which it brings more diversity to the classification. The fundamental feature of the forest is the dissimilarity of the trees used therein. This de-correlation among different trees improves the robustness of the forest. Breiman (1996) achieved this goal by sampling with a replacement (Bootstrapping), training a weak classifier with this random data and making the prediction by aggregating the results. He discovered that decision trees could bring even more diversity to the bagging. For this purpose, in each tree node random numbers of the attributes were selected (usually $\sqrt{F}$ or *log(F)+1* number of the features in each node). He implemented the Random Forest with Classification and Regression Trees (CART) for solving both classification

and regression problems (refer to section 2.5.3). To find the splitting point among attributes, he used the Gini Index (Breiman, 2001). The mechanism of the Random Forest is shown in Algorithm 4.

Algorithm 4: Random Forest

1. For b=1 to B:
    a. Draw a Bootstrap sample Z of size N from the training data
    b. Grow a random forest tree T_b to the bootstrap data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $n_{min}$is reached.
        I. Select m variable at random from the p variable
        II. Pick the best variable/split-point among the m
        III. Split the node into two daughter nodes.
2. Output the ensemble of Trees $\{T_b\}_1^B$

B is the number of the different weak learners (Trees). *N* is the training data and *Z* is the random subset of this data in each weak learner; *p* is the number of the features and *m* is the random number of the features.

In Figure 2.12, a Random Forest with three random trees is depicted. The new case *v*, as shown, has a different probability for being a member of the red, green or blue classes in each tree. The Random Forest, instead of looking at one of them, considers all of them together and averages the different probabilities according to the following formula:

$$p(c|v) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|v) \qquad 2.16$$

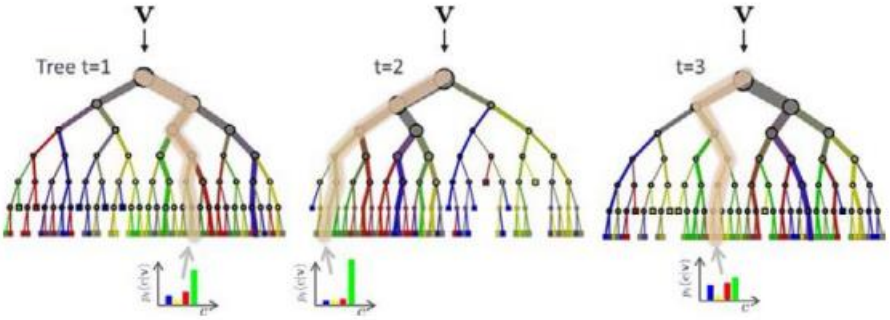


Figure 2.12: Random Forests

The Random Forest model also has some parameters and aspects that can affect the accuracy of the algorithm, such as:

- Depth of the trees
- The amount of randomness
- The size of the forest
- The weak learner type

Figure 2.13 shows the effect of the depth of the trees. Three different random forests test the same data with equal forest size (*T=200*); the only difference is the depth of the Random Forest. As shown, when *D=3*, the classifier does not assign many cases to correct classes that causes underfitting, and, with too much depth (*D=15*), the overfitting problem arises. *D=6* is a reasonable depth in this example.

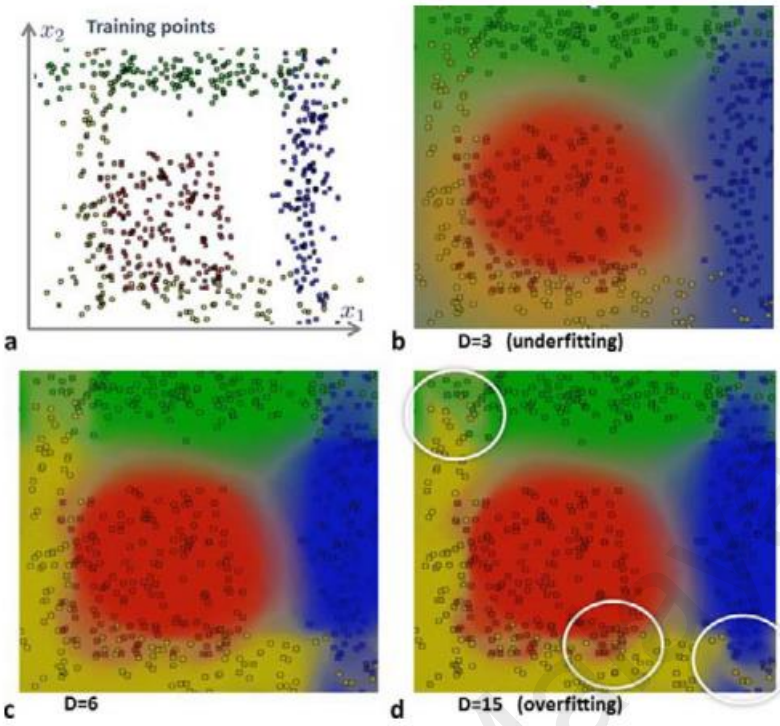


Figure 2.13: Depth of the tree

Larger randomness produces more rounded decision boundaries; on the one hand, it decreases the confidence on the other cases. Figure 2.14 has low randomness, while, in contrast, the randomness is high in Figure 2.15.

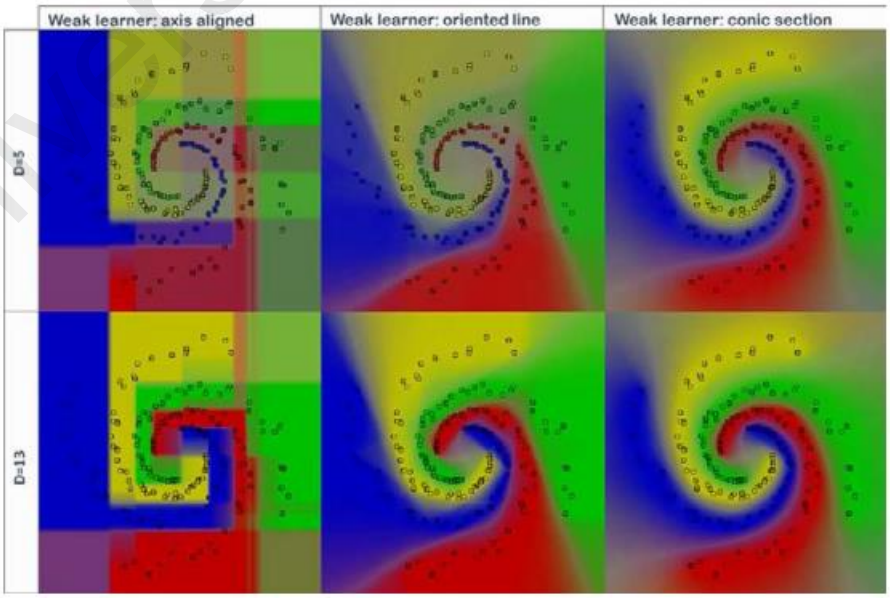


Figure 2.14: Random Forest with p=500

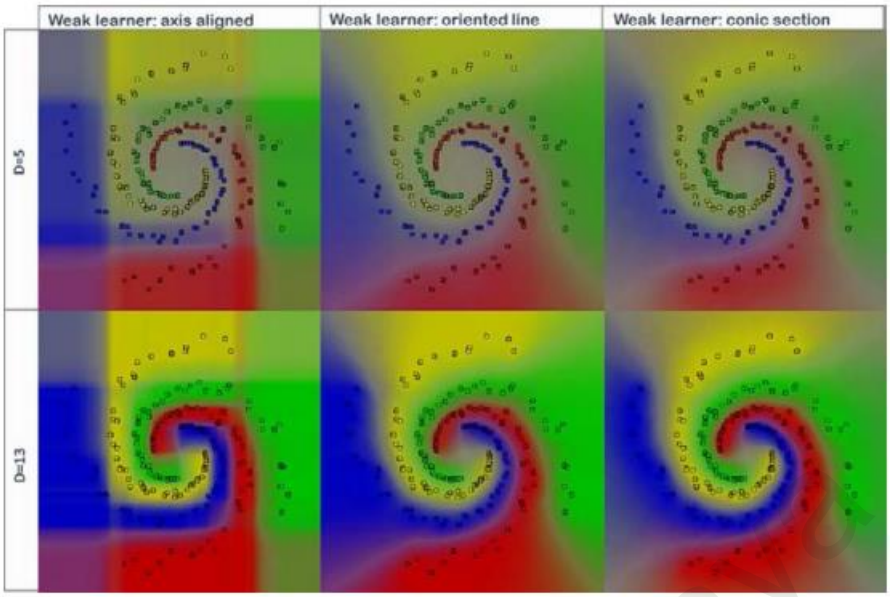


Figure 2.15: Random Forest with p=5

In Figure 2.16, the effect of the Forest's size (*T*) is depicted on the stamps with a shallow depth (*D=2*). The Weak learner is an axis-based separator. Each generated tree is slightly different from the others. Therefore, when the size of the forest increases, the confidence about the decision also increases. As shown in C_1, which only uses one tree, the decision does not have that much flexibility and, by only moving one inch, the detected class will change. However, as the number of stamps increases, we obtain better and more reasonable knowledge about the data and, finally, with 200 different trees in C_3, we obtain a realistic view from the data.

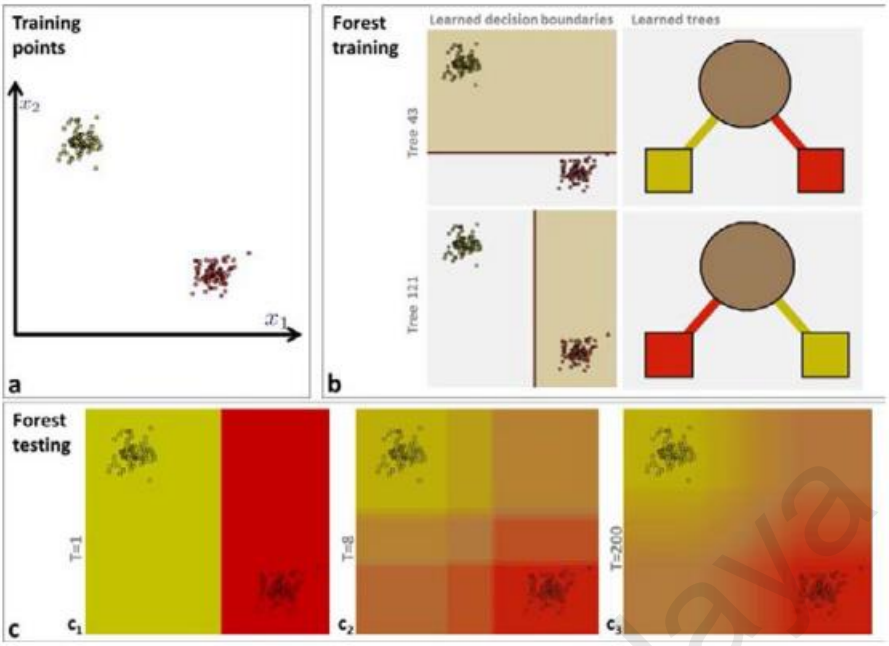


Figure 2.16: Effect of the Forest's size

The type of the Weak learner also affects the classification quality in the Random Forest. In Figure 2-17, three Random Forests are applied on the same dataset (a) with various weak classifiers that are slightly stronger than a random classifier (Weak learners). Axis-aligned separators are used as the Weak learner in (b), the oriented line is used in (c), and the conic section in (d). The size of the forests and their depth are kept equal to remove any side effects (*D=3*, *T=200*). The answer to which of these Weak learners is superior depends on the application. For instance, in the axis-aligned Weak learner, the corner of the shape has high confidence. However, in the conic Weak learner, the same corner has lower confidence.

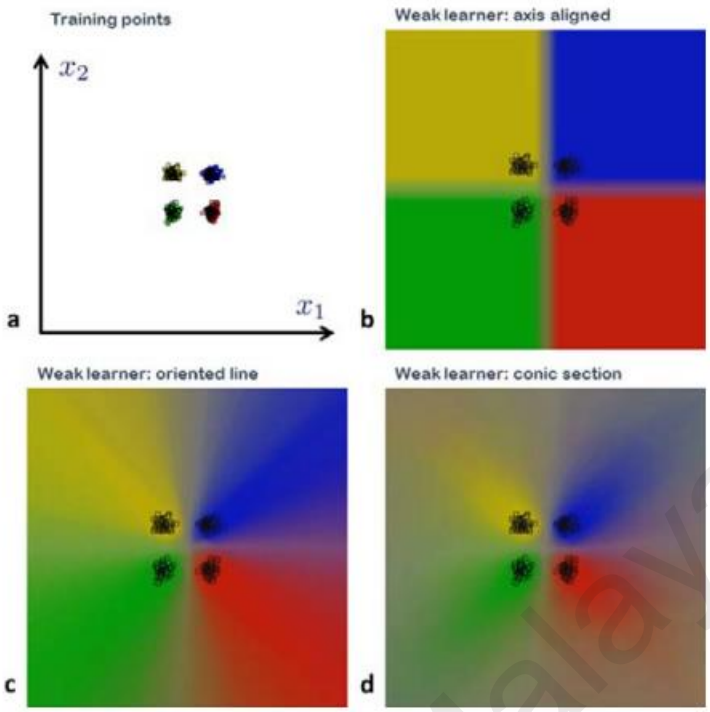


Figure 2.17: Type of WeakLearner effect

It has been more than a decade since Breiman (2001) introduced the Random Forest as a successful ensemble classifier. Since then, a number of researchers have endeavored to improve the random forest approach with respect to different aspects and applications. For instance, Robnik-Sikonja (2004) conducted a research study to improve Random Forests, in which he conducted two experiments to validate his two hypotheses. He argued that the Gini Index, which was used by Breiman, might not be the best option, as it cannot detect strong conditional dependencies among the features. He applied ReliefF instead of the Gini Index in his experiment. Another innovation that he made was upgrading the voting part, which slightly improved the precision of Random Forest.

Rodríguez, Kuncheva and Alonso (2006) invented a new method for creating ensemble classifiers. They chose the Random Forest because it was more sensitive to

rotation of the feature axis. They broke the *n* number of features into *K* random subsets, for which the eigenvector was calculated through Principal Component Analysis. PCA is a technique that reduces an data to its most important components by removing correlated characteristics (Islam, 2014). With these eigenvectors, they created the coefficient rotation matrix. A scalar λ is called an eigenvalue of the n × n matrix A. There is a nontrivial solution x of Ax= λ. Such an x is called an eigenvector corresponding to the eigenvalue λ. They sorted this matrix according to the order of the original features in the main dataset. Finally, by multiplying the dataset by this coefficient rotation matrix, they calculate the Random Forest training set. They tested their method on 33 different datasets and, for most, the Rotation Forest was significantly better than C4.5, Bagging C4.5 and Boosting C4.5.

Do et al. (2010) focused on improving the learning function of the Random Forest. Their intention was to increase the performance of the Random Forest for high-dimensional datasets, such as text. Instead of using the Gini Index of the original Random Forest, they used the Support Vector Machine (SVM) as an oblique learning function to select the best split in each subset of the selected random attributes. Their proposed technique worked well on the 25 selected datasets. Ye et al. (2013) also tried to adopt Random Forests for high-dimensional data, in which they divided the features into strong informative and weak informative groups. The subfeatures were chosen proportionally from each group in the process of creating the random trees.

2.7 Genetic Algorithm

Genetic Algorithm (GA) is an important optimization technique that simulates the evolution theory. In Genetic Algorithm, each generation consists of a population of

character strings that are analogous to the chromosomes that we see in our DNA. Each individual represents a point in a search space, as well as a possible solution. The individuals in the population are then made to go through a process of evolution.

GA is based on an analogy to the genetic structure and behavior of chromosomes within a population of individuals using the following foundations:

- Individuals in a population compete for resources and mates.
- Successful individuals in each "competition" will produce more offspring than those individuals that perform poorly.
- Genes from "good" individuals propagate throughout the population so that two good parents will sometimes produce offspring that are better than either parent.
- Thus, each successive generation will become more suited to their environment.

A population of individuals is maintained within a search space for a GA, each representing a possible solution to a given problem. Each individual is coded as a finite length vector of components, or variables. To continue the genetic analogy, these individuals are likened to chromosomes, while the variables are analogous to genes. Thus, a chromosome (solution) is composed of several genes (variables). A fitness score is assigned to each solution, representing the abilities of an individual to compete. The individual with the optimal (ornear optimal) fitness score is sought. The GA aims to use selective "breeding" of the solutions to produce better "offspring" than the parents by combining information from the "chromosomes".

The GA maintains a population of ***n*** chromosomes (solutions) with associated fitness values. Parents are selected to mate on the basis of their fitness, producing offspring via a reproductive plan. Consequently, highly fit solutions are given more opportunities to reproduce, so that offspring inherit characteristics from each parent. As parents mate and produce offspring, room must be made for the new arrivals, since the population is kept at a static size. Individuals in the population die and are replaced by new solutions; eventually, this creates a new generation once all the mating opportunities in the old population have been exhausted. In this way, it is hoped that over successive generations, better solutions will thrive, while the least fit solutions die out.

New generations of solutions are produced containing, on average, better genes than a typical solution in the previous generation. Each successive generation will contain better "partial solutions" than previous generations. Eventually, once the population has converged and is not producing offspring that is noticeably different from those in previous generations, the algorithm itself is said to have converged on a set of solutions for the problem at hand.

After an initial population is randomly generated, the algorithm evolves through three operators:

1. **Selection**: This operator selects chromosomes in the population for reproduction. The fitter the chromosome, the more times it is likely to be selected to reproduce.
2. **Crossover**: This operator randomly chooses a locus and exchanges the subsequences before and after the locus between two chromosomes to create two offspring. For example, this string 10000100 and 11111111 could be crossed over after the third locus in each to produce two offspring: 10011111and

11100100. The crossover operator roughly mimics biological recombination between two single-chromosome (haploid) organisms.

3. **Mutation**: This operator randomly flips some of the bits in chromosomes. For example, the string 00000100 might be mutated in its second position to yield 01000100. Mutation can occur at each bit position in a string with some probability, usually very small (e.g., 0.001).

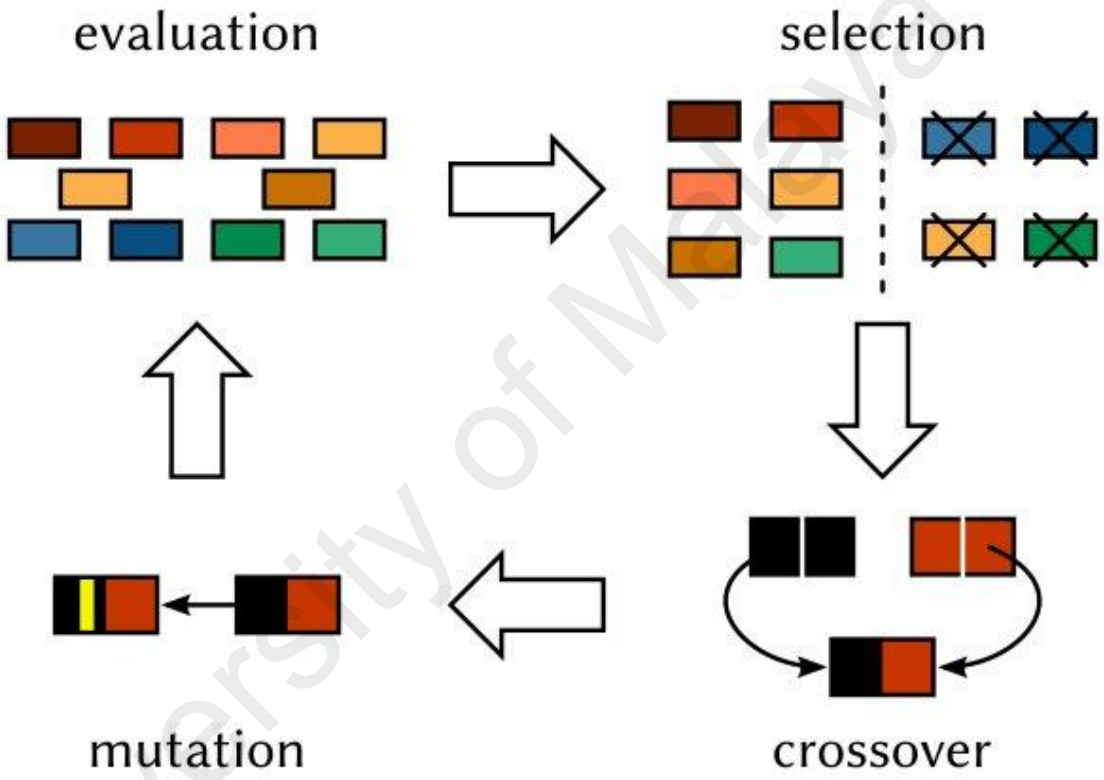


Figure 2.18: Mutation and Crossover (Jade, 2016)

Figure 2.18 depicts how mutation and crossover is doing in GA. Chapter 5 explains in detail how this research uses Genetic Algorithm.

2.8 Cross-validation

As discussed earlier, in supervised learning, the dataset is divided into training and test sets. To measure the error rate, the predictor is applied on the test dataset. However, there is a concern that the predictor attunes itself with the test dataset and does not show

the correct error rate. Cross-validation is the technique for solving this problem. Cross-validation focuses on splitting the training data into a new test and training data. Different types of validation techniques are discussed in the following subsections (Panik, 2005).

2.8.1 K-fold Validation

K is the number of folds. For instance, if we select *k*=3, we keep one of the folds of the training set as the test set. This action is repeated *k* times, and each time, the learning function is applied to the selected test set, so finally, the error rate can be estimated. We should consider that with a bigger *K*, the bias would decrease, while the variance would increase, and vice versa (Panik, 2005).

Figure 2.19: K-fold cross-validation

2.8.2 Random validation

In random validation, the testing set is randomly selected from the training set. This process can be repeated several times with or without a replacement (Bootstrap). A random estimate with replacement or Bootstrapping causes an underestimate in the error rate. To solve this issue, scholars usually use 0.632 Bootstrap algorithms, as this technique helps that training data contain approximately 63 percent of the instances (Panik, 2005). Figure 2.20 describes the random validation process.

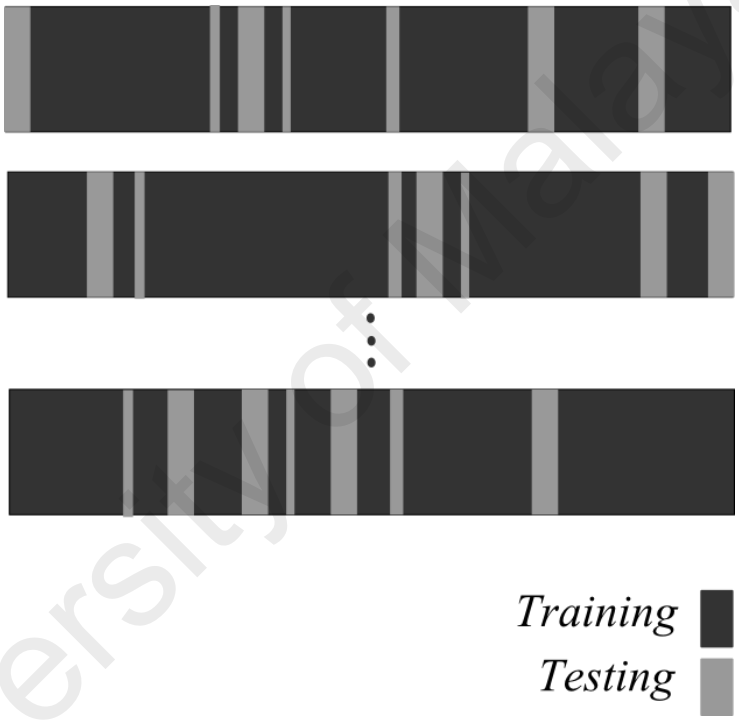


Figure 2.20: Random cross-validation

2.8.3 Leave one out

In this method, we leave one sample out each time, and make the prediction on the rest of the training set, and predict the sample with the learning function (Panik, 2005). Figure 2.21 shows how this method works.

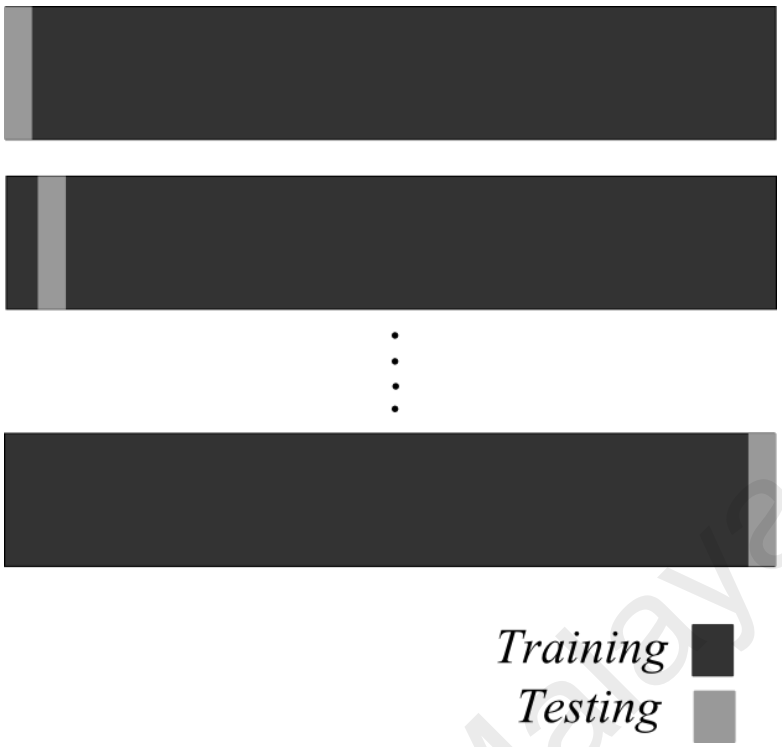


Figure 2.21: Leave one out

2.9 Text Mining

In recent years, a tremendous amount of information has been produced and spread over the Internet. According to Eric Schmidt, Google's CEO, we create as much information as human beings created from the dawn of civilization up until 2003, in only two days (Siegler, 2010). Each minute, users produce 100,000 tweets, post 684,478 pieces of content and send more than 2 million emails around the globe (Tepper, 2012). This ocean of information enables us to conduct different types of research on a wide range of issues, such as how people feel, how they describe themselves, what their political views are, what the different opinions from different countries are on a common issue etc..

The required tool to discover and extract knowledge from the text is called Text Mining or Text Analysis. According to Montes-y-Gómez et al. (2002), text mining is knowledge discovery in large text collections. Text mining has an interdisciplinary nature, as it uses different techniques depending on the field where it's being applied, such as information retrieval, information extraction, Natural Language Processing (NLP), machine learning, data mining and statistics.

2.9.1 Text Encoding

In order to analyze plain text in a simpler way, it is necessary to put it in a structured data format. There are several main approaches, such as the vector space model, the probabilistic model and the logical model. The primitive text analysis approach uses the absence or presence of the word in the context. Using the Bag-of-Words (BoW) or the collection of unordered words is also a very simple and straightforward method. For each word, a value is assigned. This value could be the frequency of the word, a weight (tf-IDF refer 3.4), a Boolean value or a normalized frequency value (Hu et al., 2009). Predominantly, researchers employ the BoW and vector space model together (Aggarwal & Zhai, 2012; Aphinyanaphongs et al., 2014; Corney, 2003; Kim et al., 2006; Sebastiani, 2002; L. Zhang, 2012). In the following section, we discuss various other related techniques to make a text file ready for more processing.

2.10 Text preprocessing

The first phase in most text-mining approaches is text preprocessing. This step is essential for two reasons; firstly, proper text pre-processing minimizes the size of the

term-frequency matrix, which models various terms in our documents; and secondly, it is important because the quality of the result depends on the input terms.

2.10.1 Tokenizing

The first step for processing text and scripts is *tokenization* (Gries & Schneider, 2010). Tokenizing is the process of converting the stream of text into split words. During this process, punctuation and superfluous symbols are removed. One of the common problems in text mining is the size reduction of the term matrix. There are various other preprocessing techniques, such as the removal stopwords, stemming, filtering and lemmatization, all of which are explained in the following subsections. In this research, tokenizing is done and then followed by stopword removal and stemming.

2.10.2 Stopword removal

According to Liu (2006), stopwords are "frequent words occurring in any context which do not represent any content", such as articles, conjunctions, prepositions, pronouns, etc. Typically, the writing style is affected by the proportion of these stopwords to the total number of words. A writer who writes in a very wordy style commonly uses more stopwords than one who is more succinct (Judd & Kalita, 2013). Furthermore, one way to assess the writing style of writers is to use the existence or frequency of stopwords as a gauge. Appendix A consists lists all the words that are considered as stopwords in this research and has removed from tokens.

2.10.3 Stemming

The removal of stopwords is usually followed by stemming, in order to convert words (usually verbs) into their stems (root forms). In many languages, there are many different forms of the root form for various grammatical uses. For instance, in English, different word shapes are created by the verb root, plural or singular nouns, adjectives

or adverbs. Even verbs come in different forms, such as gerunds, past and past participle. For example, consider "succeed" as both the verb and the root. Other forms of this root are succession (noun), successive (adj.), successful (adj.) and successfully (adv.). Stemming is achieved by removing the suffix of the words in English. One of the most popular stemming algorithms for the English language is the Porter algorithm (Porter, 1980), which is also used in this research.

2.10.4 Filtering

Filtering refers to discarding unwanted tokens and symbols. Usually, digits are removed in text mining, unless it is decided to use them in a specific application. Regarding the hyphen, there are two strategies: removing the hyphen or replacing it with a white space. For instance, based on each of the above methods, "state-of-the-art", can be converted to "state of the art" or "thestateoftheart". For solving the case letter problem, all the words are usually converted to either lower- or uppercase. Filtering and the removal of stopwords are applied together in this research.

2.11 Scientific Writing and text analysis

The style of scientific writing is very different from ordinary English, as it uses particular structures, lexicons and semantics that are devised for developing and creating scientific knowledge, such as planning research, making hypotheses, analyzing data, interpreting diagrams, and forming scientific conclusions (Fang, 2005). A recent study involved the application of text mining on scientific texts dealing with technologies by proposing a set of knowledge-based and semantic text-mining parameters (Thorleuchter & den Poel, 2013a). Based on these parameters, scientific texts are assigned to technological areas (Thorleuchter & den Poel, 2013c), the espionage risk of technological texts can be estimated (Thorleuchter & den Poel, 2013b), and textual patterns representing technological weak signals from the Internet

are identified (Thorleuchter & den Poel, 2013d). Braam et al. (1991) mixed co-citation and word analysis together to enhance the accuracy of co-citation analysis. It was revealed that this method leads to more precise results in comparison with pure co-citation analysis tools. In 2007, Tseng (2007) successfully applied text-mining techniques in his research with the aim of creating an automated patent analysis system. Ahlgren and Colliander (2009) conducted a study to determine the similarities between 43 papers from the journal 'Information Retrieval'. They implemented five different approaches, of which two were text-based and the others used bibliographic coupling or a combination of both. They found that the first-order similarity of a mixed/hybrid approach was better than the other approaches, while the second-order similarity of a pure text-based approach obtained the best performance (Ahlgren & Colliander, 2009). Argamon et al. (2008) tried to identify the possible variations of the linguistic styles of various journals in different fields using machine-learning techniques. To achieve this goal, they applied classification techniques on six fields of experimental and historical science. Their results showed that the writing styles in historical science and experimental science are clearly different.

In 2012, North (2012) demonstrated by means of an experiment how classification techniques can successfully and efficiently detect, and classify the writing of three American authors based on their writing structure and vocabulary usage (North, 2012b). In the same year, another research study was conducted pertaining to fraud detection through machine-learning techniques; the results were above 96% precision for detecting fraudulent documents from regular documents (Afroz, Brennan & Greenstadt, 2012).

In 1988, Santos (1988) conducted a study to determine the feedback concerning the writing of skilled professors of non-English speaking students (namely Chinese and Korean Students). According to the views of 178 different professors who examined the students' papers, and based on the quality of content and language, it was reported that the writing suffered from broad lexical mistakes and was considered to be academically un-publishable.

Eggins (1994) suggested a metric for gauging the lexicon density of documents. According to his definition, lexical density is measured by dividing the number of content words (nouns, base verbs, adjectives and adverbs) by the running words (prepositions, congestion, auxiliary verbs, pronouns and determinants). Based on Eggins (1994), lexical density in academic manuscripts is significantly higher than in other scripts. In another study, Halliday and his colleagues alleged that lexical density in every clause of an academic manuscript is two or three times greater than the density of a normal manuscript (Halliday, Michael Alexander Kirkwood & Martin, 1993).

Ghanem et al. (2002) conducted research on automated scientific classification and ranking. They used a feature selection technique with Bag-of-Word and lexical pattern approaches. For classification, they chose SVM. Their results reached up to 80% accuracy (Ghanem et al., 2002).

Due to name abbreviations, similar names, and name misspellings in publications or bibliographies (citations), an author may have multiple names and multiple authors may share the same name. Such name ambiguity affects the performance of document retrieval, Web searches and database integration. To solve this problem, two different classifiers were used (SVM and KNN). Features that were selected for this classification

were co-author names, journal name and article title. This study reported that Naïve Bayesian (73%) outperformed SVM (65%) (Han et al., 2004).

Detecting the reason for citing scientific papers by authors is also important. Some of the citation is done to demonstrate friendship or show respect. Several categories were detected manually, such as weak (weakness of cited approach) and CoCoGM, by Teufel and his colleagues (Table 2-4).

Table 2.4: Teufel's categories for citation reasons

Category	Description
Weak	Weakness of cited approach
CoCoGM	Contrast/Comparison in Goals or Methods(neutral)
CoCo-	Author's work is stated to be superior to cited work
CoCoR0	Contrast/Comparison in Results (neutral)
CoCoXY	Contrast between 2 cited methods
PBas	Author uses cited work as basis or starting point
PUse	Author uses tools/algorithms/data/definitions
PModi	Author adapts or modifies tools/algorithms/data
PMot	This citation is positive about approach used or problem addressed (used to motivate work in current paper)
PSim	Author's work and cited work are similar
PSup	Author's work and cited work are compatible/provide support for each other
Neut	Neutral description of cited work, or not enough textual evidence for above categories, or unlisted citation function

Teufel proposed a new method to solve this problem. Later authors focused on features for classification. Cue phrases were identified by adding notes to the text (annotation), as well as some other features, such as verb tense. The term cue phrase refers to meta-discourse, the set of expressions that talk about the act of presenting research in a paper, rather than the research itself. In the classification phase, a Vector Space Model is built over 116 scientific papers with 2829 citations. In the next phase,

10-fold cross-validation was applied with the IBK algorithm (*k=3*). Weak citation was recognized with 80% accuracy (Teufel, Siddharthan & Tidhar, 2006).

Existing uncertainty is a major issue in scientific scripts. Szarvas (2008) tried to propose a solution for this problem through a classification technique, using biomedical data in his research (radiology records and gene extraction information). A Vector Space Model was created for both corpora. Two- and three-neighbor chunks of tokens were also considered, which are called bi-grams and tri-grams, respectively. He applied the Maximum Entropy as a classifier. In the second phase of the research, feature reductions were used separately to improve the result. Maximum Entropy yielded an F-score=76.61 for biomedical reports, while adding feature reduction improved the results to 78.95. For the medical report, he achieved an F-score=64.4 without feature reduction and an F-score=79.73 with feature reduction. In both cases, using bi-grams and tri-grams improved the final results' accuracy.

In another endeavor, researchers tried to discover similarities among scientific articles. For solving the problem, a new approach was proposed – Keyword extraction. In their study, they considered abstracts, keywords and the body of articles. They found that their proposed method worked better than the link-based approach, which finds similarities between certain features, such as Bibliographic coupling and Co-citation (Yoon et al., 2011).

Uccelli et al. (2012) analyzed 51 scientific essays from high school students in the northeastern United States. They determined that the quality of academic writing depends significantly on the ways that terms and grammar are used.

Akritidis and Bozanis (2013) conducted research on automatically assigning scientific papers into one or two fields. They used various features of scientific articles,

such as keywords, authors, co-authorship and publishing journals for the classification process. For the classification algorithm, AdaBoost.MH, SVM and the new proposed algorithm were applied to 1.5 million of scientific articles. As a result, they proved that the proposed classifier outperformed the other classic classifiers.

Giannakopoulos et al. (2015) tried to classify figures of scientific papers. To achieve this aim, they focused on image features, such as Color, Edges, Lines, Histogram of oriented gradient, local binary pattern, Face-related attributes and Text-related attributes. They detected five categories (intro chart, diagram, geometric shapes, maps and continuous 2D representation, and Photoshop) for scientific figures that they classified manually among 1500 figures. Three algorithms were tested on their dataset KNN, SVM and Deep Belief Network. F-score was used for result assessment (3.8.1). They discovered that Deep Belief Network is the best.

There are millions of biological articles and knowledge discovery from them can be very tough for researchers. Zheng and Blake (2015) proposed the text extraction technique. In their research, they used supervised learning to extract sub-cellular localization information. The goal of their research was to identify a knowledge base system that contains target relations, detect and preprocess a large collection of full-text articles, identify candidate sentences by aligning the knowledge base with the text corpus, extract features from the candidate sentences, build an SVM classifier based on the features extracted during the research, and apply the classifier on unseen text from the previous step.

In another study done by Al-Daihani and Abrahams (2016), the researchers tried to investigate how academic libraries use social media. They collected tweets from 10 public universities in the US to answer these questions:

• How often do academic libraries use Twitter?

• What type of content is posted by academic libraries on Twitter?

• What are the themes associated with academic libraries' tweets?

In the preprocessing of the tweets, they removed stepwords, abbreviations, punctuation, numbers and user names. Using SVM as the classification technique for tweets, they achieved the detection of various classes with 0.85 percent accuracy.

2.12 Summary

In this chapter, we reviewed supervised learning, which was divided into two main sections: individual and ensemble classifiers. For each part, some of the most popular algorithms were introduced. In addition, text mining was explained and some of the preliminary processes for text analysis were described. Finally, some of the previous studies about text analysis and scientific writing were reviewed. Table 2.5 sums up how scientific scripts can benefit from supervised learning.

Table 2.5: Supervised learning applications in scientific area

Author	Goal	Applied Algorithm
Argamon et al. (2008)	Identify possible variations of the linguistic styles of various journals in different fields using machine-learning techniques	SVM
Afroz, Brennan & Greenstadt (2012)	Fraud detection in academic script	SVM
North (2012)	Author detection	Naïve Bayesian
Ghanem et al. (2002)	Automated scientific classification and ranking	SVM
Han et al. (2004)	Solving name ambiguity effect	KNN and SVM
Teufel, Siddharthan, and Tidhar (2006)	Detecting the reason for citing scientific papers	Instance Base KNN (IBK)
Szarvas (2008)	Detecting hedgeing in scientific text	Maximum Entropy
Akritidis &	Paper classification	AdaBoost.MH, SVM and

Bozanis (2013)		a New approach was suggested
Giannakopoulos et al. (2015)	Classifying figures of scientific articles	KNN, SVM and Deep Belief Network
Zheng & Blake (2015)	Text extraction from biological articles	SVM
Al-Daihani & Abrahams (2016)	Studying the usage of social media by academic institute libraries	SVM

As Table 2.5 depicts, supervised learning is widely used in academic and scientific writings. Some research studies only applied one classifier in order to conduct categorization; however, researchers who are interested in investigating the accuracy of classifiers or aim to improve the accuracy of the classification experiment usually compare their proposed methods with classic classifiers that are widely used, known about and considered reliable by other researchers, such as SVM, Naïve Bayesian, etc.

CHAPTER 3: RESEARCH DESIGN

3.1 Introduction

The goal of this section is to discuss the multiple steps that have been taken in this research to achieve the final results. Research design contains the plan, structure and strategy of investigation that are conceived to obtain answers to research questions or problems. The plan is the complete scheme or program of the research (Kerlinger, 1986). According to Selltiz et al. (1962), "a research design is the arrangement of conditions for collection and analysis of data in a manner that aims to combine relevance to the research purpose with economy in procedure". Research design or methodology is a process to systematically solve the research problem; in other words, it is the science of studying how research is done scientifically. Figure 3.1 explains various ways to get to the final results. In the first phase, one formulates research problems and objectives. Required datasets are created in the second phase and it is proven that classification techniques are an appropriate method to differentiate ISI and non-ISI articles from each other. Also, syntactical analysis between ISI and non-ISI articles is done in this phase. Finally, in the final step, a new classification method is proposed for improving classification accuracy amongst the ISI and non-ISI datasets.

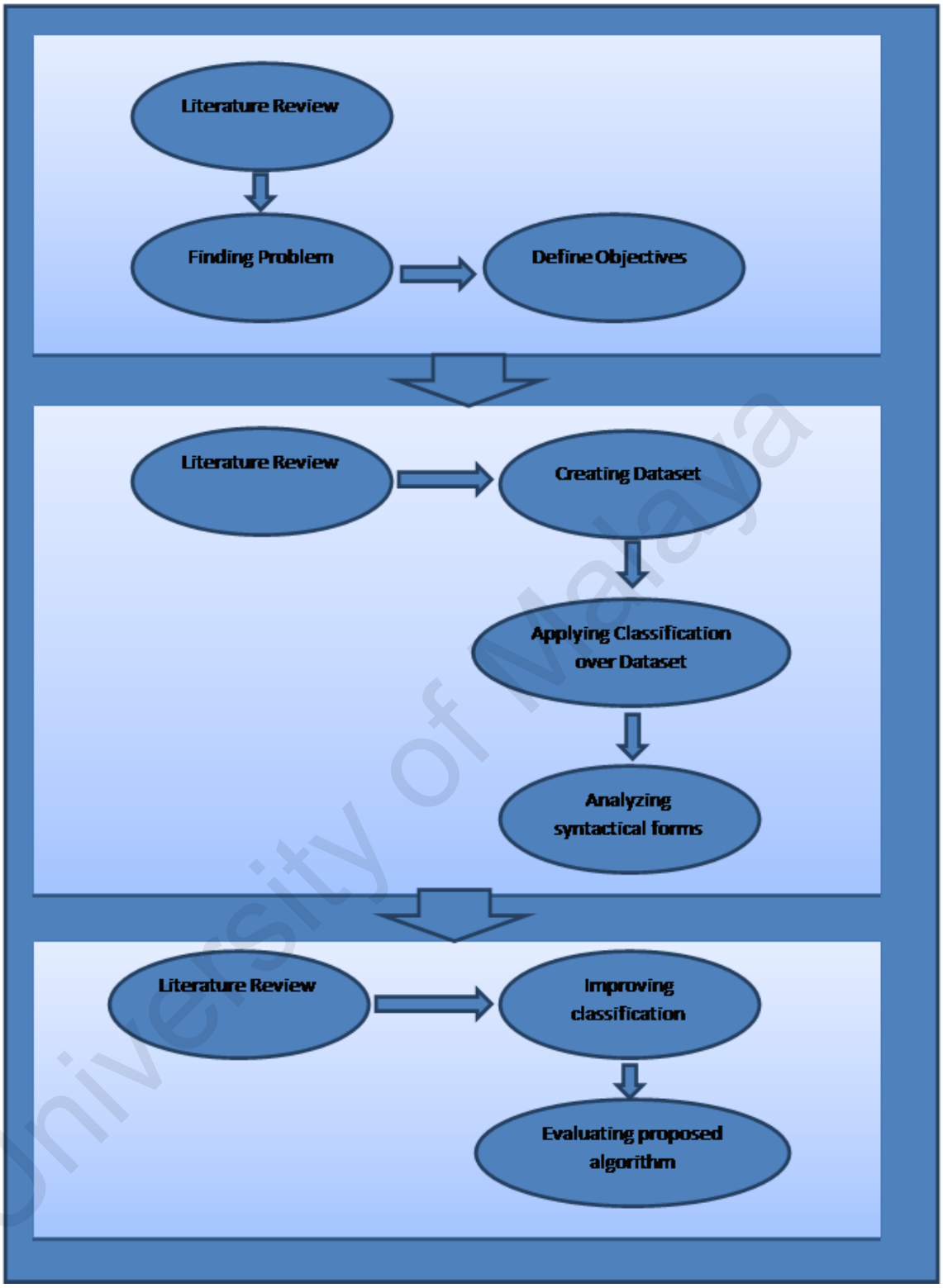


Figure 3.1: Research Methodology

3.2 Data Collection

Accessing data is an essential and primary requirement for studying the categorization of scientific scripts. Various terms were used in the Google search engine to find related datasets in 2012 (Table 3.1).

Table 3.1: Various terms used for finding related datasets

high quality scientific papers dataset
high quality scientific articles dataset
high quality academic articles dataset
high quality academic papers dataset
ISI articles dataset
ISI papers dataset

However, a proper dataset was not available at that date. Therefore, it was necessary to create a unique dataset for this research. It is challenging to decide which scientific texts contain high-quality writing. For more than 300 years, scientific journals have been using a peer-review technique for determining the quality of scientific articles (Elsevier, 2016). It is assumed that prominent scientific journals use meticulous and sophisticated examiners for peer review. Therefore, we decided to use the quality of the scientific journals as a metric for detecting the quality of scientific writings. As mentioned earlier (Chapter 1), the Institute for Scientific Information (ISI) indexes high-quality journals and annually publishes a ranking list of academic journals based on their influence in their related fields.

Due to the credibility of these journals and the belief in the precise monitoring process of the Institute for Scientific Information concerning academic journals, ISI-indexed articles are selected as samples of high-quality academic writing.

On the other hand, we needed papers that were believed to have lower quality than ISI-indexed articles in terms of writing style. Some less popular conferences seemed like a good option for this goal, due to the low standards of conference organizers. However, the difference between the two types of articles became too great, so it was not possible to generalize the result. Therefore, we decided to choose from journals that are indexed in a reliable scientific database, but without ISI-indexed metrics.

To reduce the possible divergences among collected articles and to have a homogenous dataset, all of the articles were collected from the same discipline (computer science). Moreover, articles with similar subjects were chosen because the field of computer science has a broad scope, so it could not be guaranteed that articles in that discipline shared a common vocabulary domain. Nevertheless, using particular keywords isolated papers from a small technological area and decreased bias. It is assumed that the lexical domain within a limited area of the scientific world is identical. Therefore, allocating a paper to the ISI-indexed journal class or to the non-ISI-indexed journal class only considers the various vocabulary usages, rather than the writing style of the technological area. These papers were chosen through the random keyword "wireless network".

To authenticate the accuracy of the model and decrease the chance of independency of the data, another dataset was designed from a very different scientific domain (business). Each data group included 100 articles, including both ISI and non-ISI papers, which were selected based on a random sampling from scientific databases (50

cases each). For business articles, the chosen articles were selected by using the random term “ERP implementation”.

All the ISI papers were chosen from the Web of Science database, whereas the non-ISI papers were extracted from the Emerald database. Table 3.1 shows the source of the chosen articles for each category in the Computer Science and Business domains. Due to the strict security policy of these scientific databases, the use of crawlers and robots to download the articles was not possible, so the entire data collection process was done manually.

Table 3.2: ISI and non-ISI indexed selected journals

	ISI papers	**Non-ISI papers**
	Journals	*Journals*
Computer Science	Ad Hoc Networks Annals of telecommunications Applied Soft Computing Computer Networks Expert Systems and Applications Internet Research Journal of Network and Computer Applications Journal of Network and Systems Management Journal of Signal Processing Systems Kybernetes Mobile Networks and Applications Telematics and Informatics	Campus-Wide Information Systems Info International Journal of Intelligent Computing and Cybernetics International Journal of Pervasive Computing and Communications
	Journals	*Journals*
Business	Information And Organization International Journal Of Production Research Information Technology And Management Scandinavian Journal Of Management Industrial Management & Data Systems International Journal Of Operations & Production Management Decision Support Systems Journal Of The Chinese Institute Of Engineers Expert Systems With Applications Management Decision Production Planning & Control Service Industries Journal Total Quality Management & Business Excellence	Journal of Enterprise Information Management Benchmarking: An International Journal Information Technology & People Business Process Management Journal Journal of Manufacturing Technology Management Journal of Management in Medicine International Journal of Managing Projects in Business Information Management & Computer Security Logistics Information Management Journal of Information, Communication and Ethics in Society International Journal of Physical Distribution & Logistics Management Management Research Review

Information & Management
IEEE Transactions On Engineering Management
International Journal Of Production Economics
International Journal Of Human-Computer Studies
Public Management Review Service Business
Business Strategy Series

3.3 Preprocessing

As mentioned in Chapter 2, preprocessing is the first step of text processing and most data mining projects. Several steps are necessary before any further processing can occur. These steps are discussed in this section. Collected data is not pure and has been polluted with HTML and Java Script codes. Moreover, all parts of scientific articles are not useful for analysis, such as figures, tables and article references. All of this redundant data is removed in the first step of preprocessing.

Tokenizing becomes essential here to break down the sentences into split words. During this process, punctuation and superfluous symbols are removed. The next phase in preprocessing is the removal of stopwords, a basic technique in preprocessing, as discussed in Chapter 2. According to Liu (2006), stopwords are frequent words that occur in any context that do not represent any content. However, this research is trying to quantify the authors' scientific writing style, so the removal of stopwords was necessary. The chosen stopword list was collected from the *Rank L* website (Rank NL, 2015).

The stopword removal is usually followed by stemming to convert words (usually verbs) into their stems (root forms). However, in this research, it was chosen not to apply stemming, because applying the different syntactical forms of words can change the writing quality and, in the English language, every term has different syntactical

forms based on its role and usage in various contexts. For instance, the use of noun phrases is more common in scientific manuscripts than in others (Biber & Gray, 2010; Cortes, 2004; Fang, 2005). Because applying stemming on a scientific text affects the perceived scientific writing style, stemming was not implemented in order to keep the original forms of words in both datasets.

Table 3.3 describes how tokenizing and stopword removal works through the use of an example.

Table 3.3: Tokenizing and stopword removal example

Original text
The automated categorization (or classification) of texts into predefined categories has witnessed a booming interest in the last ten years, due to the increased availability of documents in digital form and the ensuing need to organize them
Tokenized text
'The', 'automated', 'categorization', 'or', 'classification', 'of', 'texts', 'into', 'predefined', 'categories', 'has', 'witnessed', 'a', 'booming', 'interest', 'in', 'the', 'last', 'ten', 'years,', 'due', 'to', 'the', 'increased', 'availability', 'of', 'documents', 'in', 'digital', 'form', 'and', 'the', 'ensuing', 'need', 'to', 'organize', 'them'
Removing stopwords
'automated', 'categorization', 'classification', 'texts', 'predefined', 'categories', 'witnessed', 'booming', 'interest', 'last', 'ten', 'years,', 'due', 'increased', 'availability', 'documents', 'digital', 'ensuing', 'need', 'organize'

3.4 Term-document matrix

The next step after preprocessing is to build a term vector based on the vector space model. To build a term vector for each paper, we used the Term Frequency-Inverse Document Frequency (TF-IDF) weighting scheme, which is defined in Formula 3-1 (Ahlgren & Colliander, 2009).

$$tf - idf_{t,d} = tf_{t,d} \times log \frac{N}{df_t} \quad 3\text{-}1$$

$tf_{t,d}$ represents the term frequency of term *t* in document *d*, and *N* shows the total number of documents, while df_t represents the number of documents that contain term *t*. The benefit of using TF-IDF is that it harmonizes moderate frequency terms that are repeated in many papers by assigning them an acceptable score, while assigning a higher score to those terms with large frequencies in a few papers (Manning, Raghavan & Schütze, 2008). The term vectors can then be used for creating a term-document matrix. The rows and columns represent terms and papers, respectively, and each element of the matrix is described by the weighted frequency of terms in the papers, as calculated by TF-IDF.

Consider a document containing 100 words, wherein the word *technology* appears 7 times. The term frequency (i.e., tf) for *technology* is then (7 / 100) = 0.07. Now, assume we have 1000 documents and the word *technology* appears in 600 of them. Then, the inverse document frequency (i.e., idf) is calculated as log(1,000 / 600) = 1.66. Thus, the Tf-idf weight is the product of these quantities: 0.07 * 1.66 = 0.116.

3.5 Applying Basedline Classifiers

This stage of the research clarifies how practical it would be to use a classification technique to distinguish ISI articles from non-ISI articles. For this purpose, the output of the preprocessing step is used for classification in the term-document matrix format. As mentioned in section 3.2, two datasets are available. For each of them, data labeling is done for ISI and non-ISI indexed articles.

To validate this model and discover how well the created model functions, it was necessary to divide the dataset into training and test sets. However, due to the manual process of collecting data, our dataset was not very large and the size of the test and training sets could be unsatisfactory. The solution to this problem was to use cross-validation, which was introduced in Section 2.8. 10-fold cross-validation is used in this research. Cross-validation divides data into 10 equal parts and chooses one part as the test set and the rest as training data each time. If you have a single holdout set, where 90% of the data is used for training and 10% is used for testing, the test set is very small, so there will be a lot of variation in the performance estimate for different samples of data, or for the various partitions of the data to form the training and test sets. 10-fold validation reduces this variance by averaging more than 10 different partitions, so the performance estimate is less sensitive to the partitioning of the data.

As basedline classifier, three popular machine-learning algorithms were chosen: Naïve Bayesian, K-nearest neighbor and Support Vector Machine. The reason for choosing these algorithms was their popularity among researchers (Aggarwal & Zhai, 2012). For text classification, the threshold used to classify a document into ISI versus non-ISI is a 50% probability in the case of Naïve Bayesian. In the case of k-NN, *k* as the number of neighbors is set to three, after experimenting various numbers of *k* , it concluded that the best result belongs to three neighbours. In SVM, the calculated distance of each document to the hyperplane is transformed to a class probability using Platt's method (Platt, 1999). Eventually, the documents are assigned to a class based on a 50% threshold.

For SVM, a linear discriminant function was used (linear kernel). We are aware that non-linear functions possibly perform better, but the linear kernel was chosen for two reasons: first, the use of a linear kernel avoids the higher complexity of a non-linear

kernel and makes the results more transparent; second, according to experiments that were done with the collected datasets, K-NN and NB, the linear kernel works fine.

Figure 3.2 explained the classification process from preprocessing to the application of the baseline algorithms.

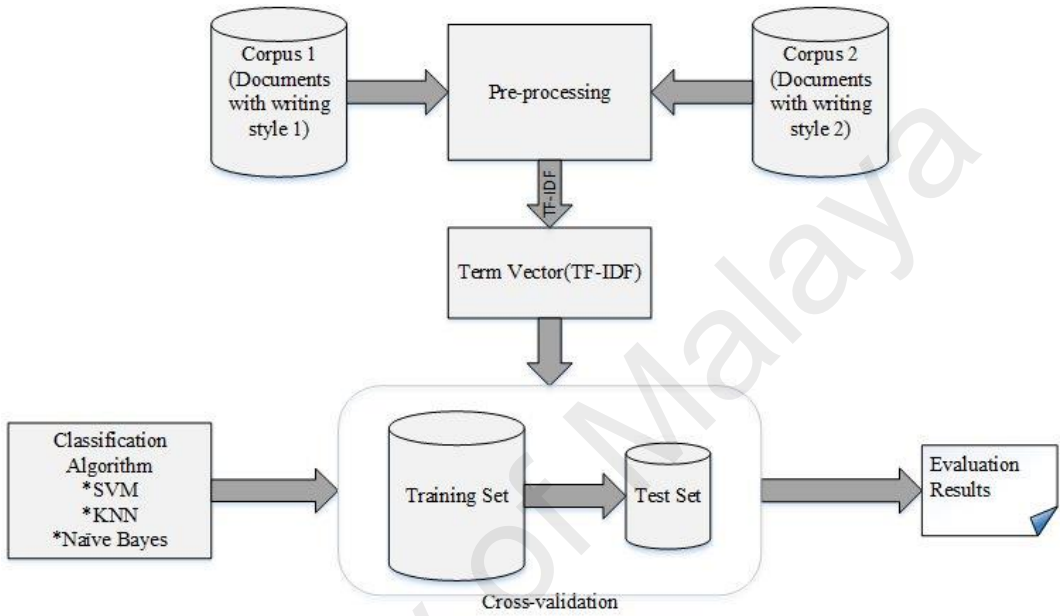


Figure 3.2: Classification Framework

3.6 Proposing a novel classifier

In order to improve the accuracy of the existing classifiers, we developed a hybrid classifier by upgrading one of the existing ensemble classifiers (Random Forest) and optimizing it with Genetic Algorithm. The reason for choosing ensemble classifiers was for their acceptable performance and we chose Random Forest for its high accuracy in different situations (Verikas, Gelzinis & Bacauskiene, 2011). We called this novel algorithm Hybrid Genetic Random Forests (HGRF). Chapter 5 introduces HGRF in greater detail. To ensure that HGRF is a stable and reliable algorithm, we applied it on 20 different UCI datasets. The UCI Machine Learning Repository is a collection of

databases, domain theories and data generators that are used by the machine learning community for the empirical analysis of machine-learning algorithms. This archive was created in 1987 by David Aha and his fellow graduate students at UC Irvine. Since then, it has been widely used by students, educators and researchers all over the world as a primary source of machine-learning datasets (Lichman, 2013).

In the following, HGRF is tested on the ISI and non-ISI indexed article dataset. The evaluation section in Chapter 5 compares the results of HGRF and its competitors.

3.7 Discovering common syntactical forms

Based on the classification models created using the three algorithms, the scientific vocabulary is analyzed by the algorithms to distinguish between ISI-indexed journals and non-ISI-indexed journals. A part-of-speech (POS) tagger is applied to identify the different syntactical forms of words in the document collection. This research uses the Pen Threebank POS (Pennsylvania, 1999).

Table 3.4: POS tagging example

They refuse to permit us to obtain the refuse permit	
('They', 'PRP'), ('refuse', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'), ('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')	
CC	Coordinating conjunction
PRP	Personal pronoun
VBP	Verb, non-3rd person singular present
VB	Verb, base form
DT	Determiner
NN	Noun, singular or mass

We selected these words in a specific syntactical form and for which the corresponding term weights were above a specific threshold. A sensitivity analysis based on the three basedline classification models was conducted to identify the impact

of the selected words on the classification decision. The result of this analysis is reported in Chapter 4.

3.8 Evaluation

In supervised learning, we measure the accuracy of the error rate. In classification, we encounter two types of errors*: in sample* and *out sample*. *In-sample* errors refer to the misclassified samples in the dataset on which the predictor is built. *Out-sample* errors *(Generalization)* refer to the errors that happen when we apply the predictor on new data. The out-sample error rate is higher than the in-sample error rate. Because the learning function attunes itself with the training data (*overfitting*), it decreases the sample error. On the other hand, the performance of the learning function on unseen data is lower (James et al., 2013).

In this research, the accuracy of the classification technique is calculated. To understand the concept of accuracy, it is necessary to be familiar with the confusion matrix.

3.8.1 Confusion matrix

If we consider that we are doing binary classification, we can categorize our predictions into four different sets: *True positive*, *False positive*, *True negative* and *False negative.*

True positive refers to the cases that are predicted positive, and are actually positive in the dataset.

False positive refers to the cases that are mistakenly predicted as positive, but do not actually belong to the positive set.

True negative refers to cases that are predicted negative and are actually negative.

False negative refers to cases that are positive in the real world, but the predictor categorizes them as negative.

These four different sets form a *Confusion Matrix* (Figure 3.3). Using a confusion matrix is a very common way to measure the accuracy of classification.

	Positive ***(Disease)***	**Negative** ***(Healthy)***
Predicted Positive *(Predicted Disease)*	*True positive (TP)*	*False Positive (FP)*
Predicted Negative *(Predicted*	*False negative (FN)*	*True negative (TN)*

Figure 3.3: Confusion Matrix

Based on the information in the confusion matrix, some other useful measurements can be calculated.

Sensitivity (recall) is the probability that a case will be predicted positive. For example, the probability of being predicted sick when you are genuinely sick. This item is calculated based on the formula:

$$Sensitivity = \frac{TP}{TP + FN}$$

Specificity is the probability that a case will be predicted negative when it is actually negative.

$$Specifity = \frac{TN}{TN + FP}$$

Positive predicted value (precision) is defined as:

$$Positive\ predicted\ value = \frac{TP}{TP + FP}$$

Negative predicated value is defined as

$$Negative\ predicted\ value = \frac{TN}{TN + FN}$$

And the ***Accuracy,*** which is the probability of predicted correctly, is defined as

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

F-score (F1-Score) is the harmonic mean of *precision* and *recall*. In many research studies, this number is reported to show the accuracy of the prediction.

$$F - score = 2.\frac{Precision.recall}{Precision + recall}$$

3.9 Summary

To summarize, this chapter is a road map of this research study and it explained how the study was conducted and how we managed to answer the research questions and

meet the objectives. We will elaborate on the results of applying the baseline classifier in Chapter 4 and will discuss the prevalent syntactical terms in ISI and non-ISI articles. Later, the HGRF will be introduced in Chapter 5. Chapter 5 also advances some evidence of why HGRF is such a successful method.

CHAPTER 4: CLASSIFICATION FOR DISTINGUISHING ISI AND NON-ISI ARTICLES

4.1 Introduction

This chapter attempts to answer the question of whether machine learning and specifically supervised learning is a proper method for distinguishing high-quality scientific articles from low-quality ones. To answer this question, some of the classic and popular classification algorithms have been chosen to apply on ISI and non-ISI article datasets. These datasets were introduced in Chapter 3. Chapter 4 presents the results of these classification algorithms on ISI and non-ISI datasets. Moreover, it discusses common grammatical syntax and compares the results between these two groups.

4.2 Classification Experiment

To find out whether classification is an appropriate technique for the determination of the quality of academic writing, three basic and popular classic classifiers were chosen. As explained in Chapter 2, preprocessing removes redundant data, such as stopwords, tables, figures and punctuation symbols. TF-IDF also converts the tokenized text into a matrix, where each row represents a document and each column represents a token of scientific articles. Figure 4.1 is a snapshot of a TF-IDF matrix (3.4).

ISI	abandon	abandoned	abandonm...	abc	abdinnour	abi	abilities	ability	able	abnormal	absence
1	0	0	0	0	0	0	0	0.005	0.007	0	0.013
1	0	0	0	0	0	0	0.006	0.001	0.005	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0.009	0	0	0	0	0	0.005	0.003	0	0
1	0	0	0	0	0	0	0	0.002	0	0	0
1	0	0	0	0	0	0	0	0.010	0	0	0
1	0	0	0	0	0	0	0.016	0.007	0.001	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0.008	0	0
1	0	0.026	0.034	0	0	0	0	0.011	0.015	0	0
1	0	0	0	0	0	0	0	0.002	0.002	0	0
1	0	0	0	0	0	0	0	0.008	0.006	0	0
1	0	0	0	0	0	0	0	0.003	0.004	0	0
1	0	0	0	0	0	0	0	0	0.008	0	0
1	0	0	0	0	0.021	0	0	0	0.005	0	0
1	0	0	0	0	0.054	0	0	0.006	0.004	0	0
1	0.014	0.015	0	0	0	0	0	0	0.005	0	0
1	0	0	0	0	0	0	0	0.003	0.003	0	0

Figure 4.1: TF-IDF matrix

In the next step, the target column will be added to this matrix as a classification label for each sample. As described in Chapter 2, a supervised learning algorithm needs some data to learn from. Labeling helps the classification algorithm assign an existing pattern in the data to one of the classes and predict the unseen data (test set) with the help of a learning function that is created during training.

One of the common dilemmas in supervised learning is the problem of overfitting. Overfitting refers to learning from incorrect or fake patterns in the data. In this way, the learning function cannot detect real patterns. One of the solutions to overfitting is cross-validation. Cross-validation can be used to simply estimate the generalization error of a given model, or it can be used for model selection by choosing one of several models that has the smallest estimated generalization error. For example, you might use cross-validation to choose the number of hidden units, or you could use cross-validation to choose a subset of the inputs (subset selection). A subset that contains all relevant inputs will be called a "good" subset, while the subset that contains all relevant inputs but no relevant input will be called the "best" subset. Note that subsets are "good" and "best" in

an asymptotic sense (as the number of training cases extends to infinity). With a small training set, it is possible that a subset that is smaller than the "best" subset may provide a better generalization error.

To determine which cross-validation method works better with the collected data, an experiment is run to choose the best method. In this research, we tried three different cross-validation settings. *One leave out*, *k-fold* cross-validation with testing *k=5* and *k=10,* which are called 5-fold and 10-fold cross-validation, respectively. The *One leave out* case trains itself with 99 cases and keeps one as a test to evaluate the performance with the test. This process will repeat over the 100 cases and an average of the outcome will be reached. In 5-fold cross-validation, 20 cases are kept for training and 80 cases for training and this process repeats four more times. The rest of the process is similar to One leave out. 10-fold cross-validation is just like 5-fold, but it breaks down the data into 10 pieces and the process repeats 10 times, rather than 5.

In this research, KNN classifier, Naïve Bayesian and SVM were chosen. The three trained classification models are applied on the selected test documents. The results are an assignment probability to a class (in the case of Naïve Bayesian), an assignment function to a class that depends on the number of K (in the case of KNN), and a distance to the hyperplane (in the case of SVM). In a manual process, the selected documents are modified by adding words at randomly selected positions or by changing the syntactical forms of words. Then, the three classification models are applied to the modified documents. Changes in the assignment probability (Naïve Bayesian), the assignment function (KNN) and the distance (SVM) are used to estimate the impact of the document changes on the classification decision.

4.3 Cross-validation experiment over KNN

KNN follows a straightforward and effective idea in classification by testing each sample in a given vector space with the majority class of its *K*-nearest neighbors. The number of neighbors can play an important role in the accuracy of the KNN algorithm. Usually, the number of neighbors is calculated by a number of features. As a rule of thumb, *K* sets to $\sqrt{n}$ of features. However, in a high-dimensional dataset, using such a rule is not useful. Hence, it is decided to use trial and error instead. Here, several different numbers of *K* (between 1 and 7) are tested over the dataset to confirm which one has the best outcome. These numbers are chosen because many researchers have used this amount for their experiments (Guo, Shao & Hua 2010; Tseng, Lin & Lin, 2007; Woods, Kegelmeyer & Bowyer 1997; Zhu et al., 2010).

The experiment is repeated for 10-fold, 5-fold and one leave out cross-validation for both computer and business datasets. In each experiment, for each *K*, precision, recall and accuracy are calculated. Table 4.1 depicts the result for 10-fold cross-validation and its respective precision and recall for both Business and Computer Science articles. P and R stand for Precision and Recall, respectively.

Table 4.1: Computer and Business precision and recall results for KNN algorithm with 10-fold cross-validation

		Business			Computer	
		ISI	Non-ISI		ISI	Non-ISI
K=1	P	62.11	62.71	R	58	68.9
	R	55	74	P	72.09	66.67
K=2	P	65.75	74.36	R	82	54
	R	63.4	58	P	85.29	68.18
K=3	P	68.29	62.71	R	58	90
	R	56	65.4	P	77.2	80.43
K=4	P	66.7	66.4	R	74.8	74
	R	63.8	72	P	72.4	75.41
K=5	P	85.71	69.23	R	72.6	67.6
	R	60	90	P	66.1	74.9
k=6	P	65.9	72.41	R	60.2	84
	R	68	66.1	P	76.9	73
K=7	P	63.5	66.67	R	69.1	94
	R	56	65.7	P	62.9	68.18

Figure 4.2 presents the accuracy of different numbers of neighbors in 10-fold cross-validation. As Figure 4.2 shows, in both of the cases, *k=3* gives the best result for 10-fold cross-validation. However, it is necessary to repeat the experiment for 5-fold and one leave out cross-validation.

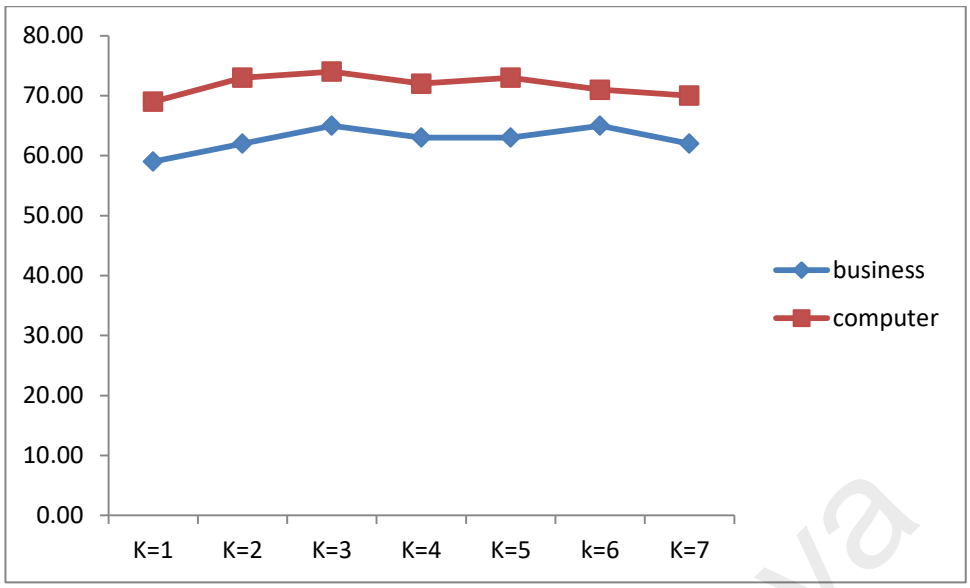


Figure 4.2: KNN accuracy with 10-fold cross-validations for computer and business datasets

The experiment is also implemented for 5-fold cross-validation. Table 4.2 shows the precision and recall data for the Business dataset.

Table 4.2: 5-fold cross-validation KNN Business and Computer dataset

		Business		Computer	
		ISI	Non-ISI	ISI	Non-ISI
K=1	P	60.1	61	74	68.18
	R	63.24	57.6	66.2	60.8
K=2	P	63.11	74.36	72.09	66.67
	R	66.6	58	70.7	54
K=3	P	64.32	62.71	72.8	65.1
	R	57.54	65.7	73.4	74.9
K=4	P	67.1	67.5	76.5	80.43
	R	63.8	72	71.9	71.4
K=5	P	70.26	69.23	64	68.5
	R	62.8	90	63.8	66.4
k=6	P	60.3	72.41	63.8	66.5
	R	63.7	62.9	67.6	69.8
K=7	P	63.2	64.5	62.8	69.1
	R	66.6	67.3	77.7	68.5

Figure 4.3 presents the accuracy of the business and computer science datasets with 5-fold cross-validation. For business articles, *k=5* has the best result and for computer science articles, *k=3* has the best.

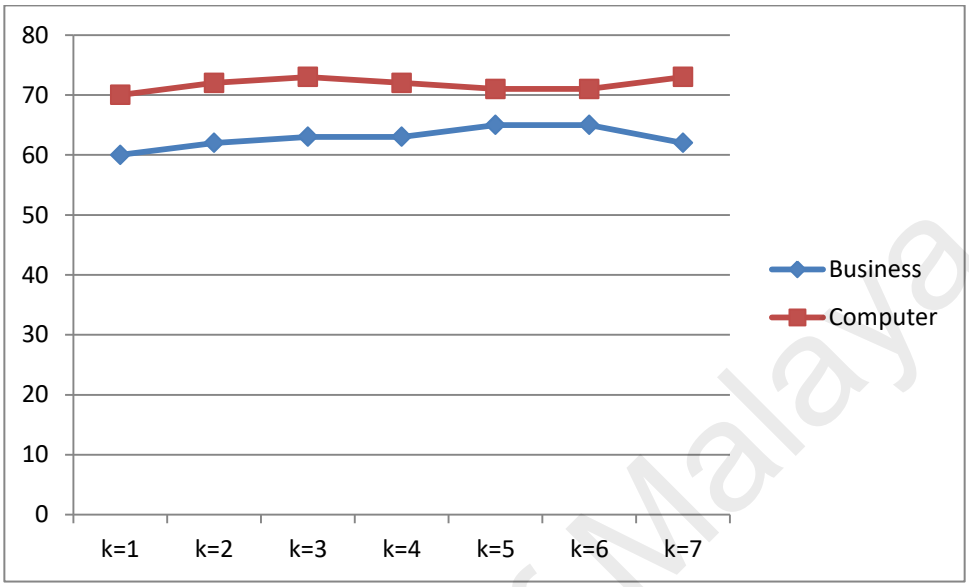


Figure 4.3: 5-fold KNN Business and Computer dataset

Finally, the experiment is done for one leave out cross-validation for the computer science and business datasets. Table 4.3 shows the precision and recall for the business data set and Figure 4.4 depicts the accuracy diagram. *K=3* is the best number for this configuration for either business or computer science.

Table 4.3: Business and Computer one leave out KNN

		Business		Computer	
		ISI	Non-ISI	ISI	Non-ISI
K=1	P	61.6	60.9	63	68.5
	R	61.2	62.6	61.5	62.5
K=2	P	60.6	61.6	61.6	75.2
	R	62.7	60	70.1	76.8
K=3	P	68.29	62.71	85.29	68.18
	R	56	65.7	58	90
K=4	P	65.3	66	61.5	74.7
	R	60.7	65.2	70.9	72.1
K=5	P	65.3	64.3	66.6	75.8
	R	67.2	66.4	71	70.5
k=6	P	62.6	61.1	61.2	66.7
	R	62.2	65.1	66.6	71.4
K=7	P	62.5	62	75.4	69.3
	R	60.6	64.9	63.3	72.4

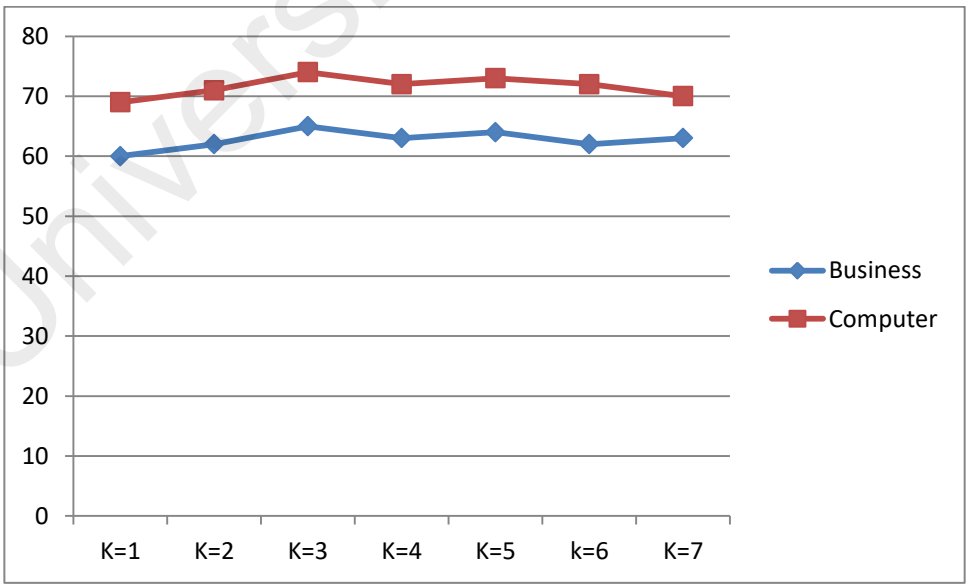


Figure 4.4: One leave out KNN Business and Computer datasets

4.4 Cross-validation experiment over Naïve Bayesian

The idea behind a Bayesian classifier is that, if an agent knows the class, it can predict the values of the other features. If it does not know the class, Bayesian' rule can be used to predict the class, given some of the feature values. In a Bayesian classifier, the learning agent builds a probabilistic model of the features and uses that model to predict the classification of a new example.

This section presents the result of applying a naïve Bayesian classifier for the business and computer science datasets with various settings for cross-validation. Table 4.4 shows the precision and recall for naïve Bayesian for both datasets with three different cross-validation settings.

Table 4.4: Precision and recall for Naïve Bayesian classifier

		5-fold		10-fold		one leave out	
		ISI	Non ISI	ISI	non ISI	ISI	Non ISI
Business	P	67.19	67.98	73.33	69.09	67.19	68.97
	R	64	80	66	76	64	80
Computer	P	73.18	66.1	71.43	65.52	71.88	88.89
	R	60	78	60	76	92	64

Table 4.4 shows the final accuracy for each of the datasets under different settings. In the business dataset, the weakest result belonged to 5-fold cross-validation and in the computer dataset, 10-fold cross-validation had the best record.

Table 4.5: Accuracy for Naïve Bayesian classifier

	5-fold	10-fold	one leave out
Business	71	72	72
Computer	69	68	62

4.5 Cross-validation experiment over Support Vector Machine

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates a set of objects with different class memberships (Chapter 2). To determine which kind of cross-validation is more consistent with the existing dataset while running SVM, an experiment is done with a different cross-validation with SVM. Table 4.6 documents the precision and recall of the business and computer science datasets with SVM.

Table 4.6: SVM results

		5-fold		10-fold		one leave out	
		ISI	Non ISI	ISI	non ISI	ISI	Non ISI
Business	P	57.14	87.5	58.02	84.21	55.56	73.68
	R	96	28	94	32	90	28
Computer	P	82.86	67.69	81.08	68.02	85.29	67.69
	R	58	87	60	86	58	88

Table 4.7 presents the result of the same experiment, this time with calculating the accuracy of SVM over our dataset. The results also showed that 10-fold cross-validation is more promising, as compared to 5-fold and one leave out methods.

Table 4.7: SVM accuracy

	5-fold	10-fold	one leave out
Business	63.01	63.03	59.14
Computer	73.33	73.21	74.11

4.6 Dataset Size effect

It is challenging to know how much data is enough to run an experiment related to machine learning. In this sense, during the data collection process, the basic classifiers

are applied over different sizes of datasets to discover how much data is enough for this research.

Table 4.8 presents the experiment that is done for the computer science dataset with various classifiers and different data size. As shown in Figure 4.5 and Table 4.5, when the size of the computer dataset is 20 (10 ISI and 10 non-ISI articles), all the algorithms predict extremely well. This happens because of the small size of the dataset. When the dataset size grows, the accuracy drops to the size of 60 cases and then gradually rises and stabilizes for 80 and 100 cases.

Table 4.8: Computer different data set size

Size	10-10	20-20	30-30	40-40	50-50
SVM	95	75	73	75	73
KNN	95	77	63	72	74
Naïve Bayesian	95	65	60	61.25	68

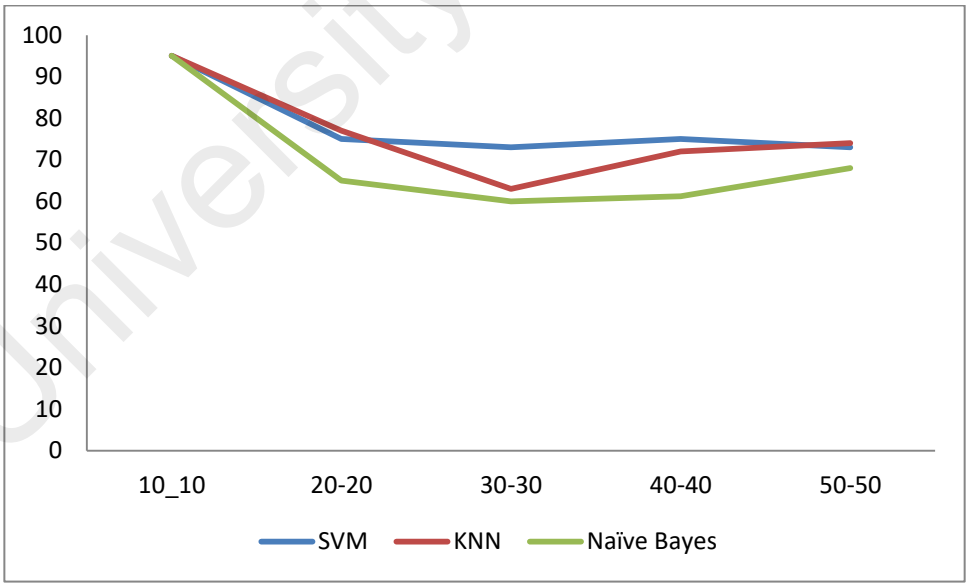


Figure 4.5: Effect of size of dataset on computer articles 10-fold cross-validation

The same experiment is repeated for the business dataset, which is presented in Table 4.9. For the business dataset, initially, the accuracy for various algorithms is very different. This is because of the small size of the dataset. For instance, the high accuracy of KNN is due to overfitting. Later, when the size grows, the trend becomes positive and stabilized for 80 and 100 cases.

Table 4.9: Business different data set size

Size	10-10	20-20	30-30	40-40	50-50
SVM	50	52	61.67	61.25	63
KNN	73	55	64	62	65
Naïve Bayesian	40	52.5	66.67	73.75	76

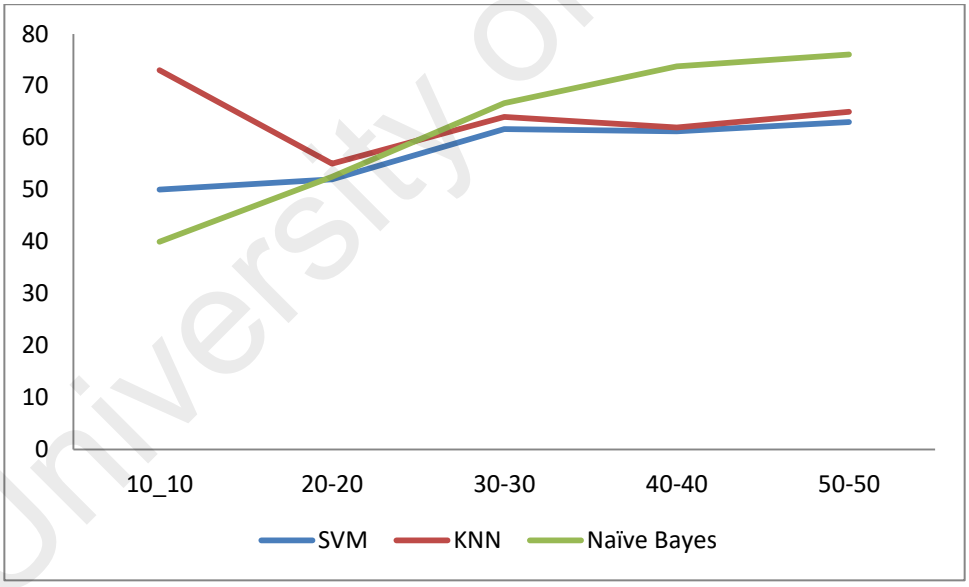


Figure 4.6: Effect of size of data set on 10-fold cross-validation Business papers

At the end, it was decided to stop at 50 cases for each category in both datasets and continue the final experiment with this amount of data.

4.7 Final Evaluation

Based on our experiments, which are presented in Sections 4.3 to 4.6, 100 cases were chosen for each dataset (50 ISI and 50 non-ISI articles). In the case of the KNN algorithm, *k=3* is used as the best number of neighbors for the final experiment. The evaluation results obtained by applying three different classification algorithms on our two datasets are summarized in Table 4.10. The best recall is obtained using the SVM algorithm, with a 94% recall for the computer science ISI-indexed journals, and interestingly, the lowest recall is obtained by SVM in the classification of non-ISI business papers. However, the SVM outcome for the classification of ISI-indexed papers in both computer science and business is quite good (above 70% accuracy). Surprisingly, the Naïve Bayesian algorithm performs better than the other algorithms in the classification of non-ISI papers. It even shows acceptable performance in detecting ISI-indexed articles. The KNN algorithm (with three neighbors) also shows moderate performance and is able to successfully differentiate between ISI and non-ISI papers.

The results for Accuracy are also presented in Table 4.10. As Table 4.10 depicts, SVM has the best outcome over business articles and KNN for computer science.

Table 4.10: Performance of SVM, KNN and Naive Bayesian on ISI and non-ISI datasets

Algorithms	Area	Precision	Recall / Sensitivity	Accuracy
SVM	Business	58.02	94	71.75
	Computer	81.08	60	78.96
KNN	Business	68.29	56	61.53
	Computer	85.29	58	69.04
Naïve Bayes	Business	73.33	66	69.47
	Computer	63.16	72	67.29

4.8 Investigating Syntactical role in scientific writings

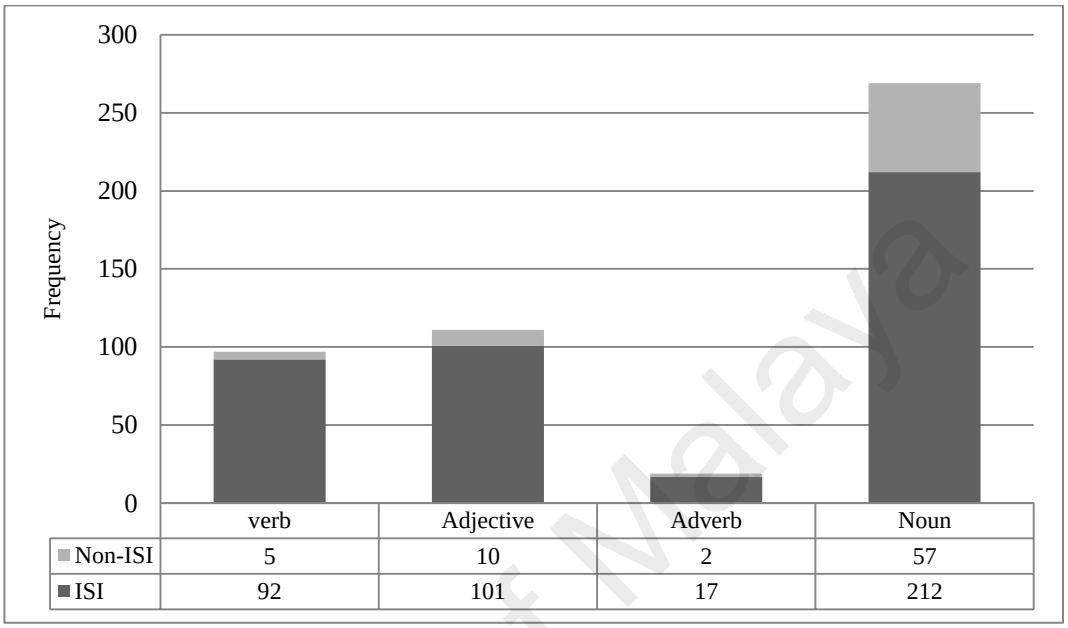
The impact of the syntactical form of a word can be illustrated by considering the word 'compromise', which often occurs in several variations in the document collection, especially in computer science journals (see Table 4.11). "Compromise" has three syntactical forms, and its impact on the two categories is estimated based on the three forms. For each form, term co-occurrences are grouped together. These terms often occur together with the syntactical form of the term. As a result, the use of "compromise" or "compromises" as a noun in a scientific paper indicates that authors in non-ISI journals have more of a tendency to use the noun form of this term, as compared to authors who write for ISI journals. Furthermore, the use of the term as an adjective or a verb is more indicative of ISI journals than non-ISI journals.

Table 4.11: Different forms of "compromise" in the papers considered and corresponding data characteristics

Term	*Form*	*Frequency*	*Number of Documents*	*Term weight*	*Impact*
compromised	Adj.	25	5	*0.81*	ISI journal
compromise	Verb	12	6	0.53	ISI journal
compromise	Noun	6	5	0.58	non-ISI journal
compromised	Verb	24	9	0.53	ISI journal
compromises	Verb	3	3	0.53	ISI journal
compromises	Noun	2	2	0.58	non-ISI journal
compromising	Verb	1	1	0.53	ISI journal

Error! Reference source not found. compares the frequencies of different grammatical forms that have been repeated at least 10 times in various articles in each category (ISI and non-ISI). Our findings show that there are some terms that are more commonly used in ISI papers. For instance, the word "complementary" as an adjective is used 46 times in 12 ISI articles. As shown in **Error! Reference source not found.**,

there is a meaningful discrepancy between ISI and non-ISI papers from using grammatical forms. For more information, some of the most popular terms in each of the grammatical groups for the collected dataset are presented in Appendix C.



	verb	Adjective	Adverb	Noun
Non-ISI	5	10	2	57
ISI	92	101	17	212

Figure 4.7: Comparing various grammatical forms' frequencies in ISI and non-ISI (Number of Document>10).

Based on the three classification models, the impact of terms on the two categories is estimated. Terms that have the greatest impact on the ISI category are listed in Table 4.12. Terms that have the most impact on the non-ISI category are listed in Table 4.13. The terms are ordered by their discriminative power from the other category.

Table 4.12: Terms that are representative of ISI papers

Term	*Form*	*Frequency*	*Number of Documents*	*Term weight*
attack	Noun	180	17	*0.56*
threat	Noun	70	12	0.68
round	Noun	163	22	0.59
border	Noun	116	16	0.65
intrusion	Noun	68	14	0.65
integrity	Noun	61	11	0.60
convex	Adj.	75	11	0.67
count	Noun	202	23	0.56
boundary	Noun	58	16	0.64
reactive	Adj.	57	14	0.57
intelligence	Noun	28	11	0.61
validity	Noun	40	13	0.60
engineering	Noun	28	11	0.63
protection	Noun	71	11	0.59
terminal	Noun	119	17	0.63
segment	Noun	259	23	0.64
multicast	Noun	645	16	0.68
competition	Noun	108	15	0.57
mutual	Adj.	33	11	0.56
combined	Adj.	41	13	0.60
surveillance	Noun	105	14	0.60
digital	Adj.	121	23	0.62
angle	Noun	46	12	0.59
shape	Noun	56	14	0.63
pick	Verb	36	12	0.57
population	Noun	105	13	0.66
multihop	Noun	70	12	0.58
immediate	Adj.	32	11	0.58
market	Noun	383	25	0.57
classical	Adj.	47	15	0.60
complementary	Adj.	46	12	0.61
cable	Noun	85	11	0.73
spread	Noun	26	11	0.56
individual	Noun	53	13	0.60
subscriber	Noun	179	15	0.59
replacement	Noun	37	11	0.69
government	Noun	51	11	0.63
broadcasting	Adj.	47	12	0.58
ground	Noun	33	12	0.64
positioning	Noun	34	12	0.63

penetration	Noun	54	11	0.65
dominant	Adj.	40	11	0.58
regulation	Noun	65	11	0.56
route	Noun	268	40	0.34
tier	Noun	23	3	0.90
protect	Verb	74	24	0.41

Table 4.13: Terms that are representative of non-ISI papers

Term	*Form*	*Frequency*	*Number of Document*	*Term weight*
content	Noun	67	13	*0.60*
participant	Noun	63	11	0.65
European	Adj.	66	12	0.60
country	Noun	116	11	0.59
regulation	Noun	65	11	0.56
subscription	Noun	223	12	0.75
industry	Noun	79	16	0.57
organizer	Noun	2	2	0.85
government	Noun	51	11	0.63
penetration	Noun	54	11	0.65
party	Noun	64	13	0.64
privacy	Noun	128	15	0.60
market	Noun	383	25	0.57
competition	Noun	108	15	0.57
subscriber	Noun	179	15	0.59
language	Noun	76	13	0.61
road	Noun	33	11	0.59
person	Noun	93	17	0.60
protection	Noun	71	11	0.59

The results show that the word form of characteristic terms for the non-ISI category is often a noun. Nouns also have an impact on the ISI category, but adjectives and verbs have a large impact on the ISI category, but not on the non-ISI category. This shows that different formulations are used in both categories.

CHAPTER 5: HYBRID GENETIC RANDOM FORESTS

5.1 Introduction

This chapter introduces a novel classification technique called Hybrid Genetic Random Forests (HGRF). HGRF's roots are in classic random forests. The next section covers how other scientists have tried to evolve the classic random forests algorithm and improve it. Section 5.2 and 5.3 explain what HGRF is and how it works. Section 5.5 evaluates the performance of HGRF in comparison to standard and well-known classifiers. Finally, HGRF will be applied on ISI and non-ISI index articles dataset and the results will be compared with the baseline algorithms in Chapter 4.

5.2 Hybrid Random Forest

In normal Random Forests (refer to Section 2.4.4), there is only one type of random tree. Xu et al. proposed a hybrid RF with three different types of decision trees: C4.5, CART, and Chi-square Automatic Interaction Detector (CHAID) (Xu, Huang, Williams, Li et al. 2012). The biggest difference between these trees is the splitting criterion of the features. C4.5 uses normalized information gain, while CART splits them based on the attribute value test, and CHAID relies on the Chi-square test. In our research, we modify Xu et al.'s method for creating the hybrid tree. Instead of using the CHAID tree algorithm, we apply the REPTree algorithm. REPTree is a random tree based on the ID3 algorithm. It uses plain information gain for splitting the features in the tree. The reason for selecting REPTree is its speed during processing and its acceptable accuracy.

After doing the sampling with replacement, the machine is trained on each of the in-of-bag samples (samples that are selected with replacement during the bootstrapping

process) by three different decision tree classifiers: C4.5, CART and REPTree. As we know, RF does not use almost one-third of the data in each bootstrap. This part is called out-of-bag, which is used to find errors in classification (Breiman, 1996). For each bootstrap, the out-of-bag error is calculated. The classifier with the lowest out-of-bag error is selected for that bootstrap. This process is iterated for all bootstrap partitions. The procedure is depicted in Figure 5-1.

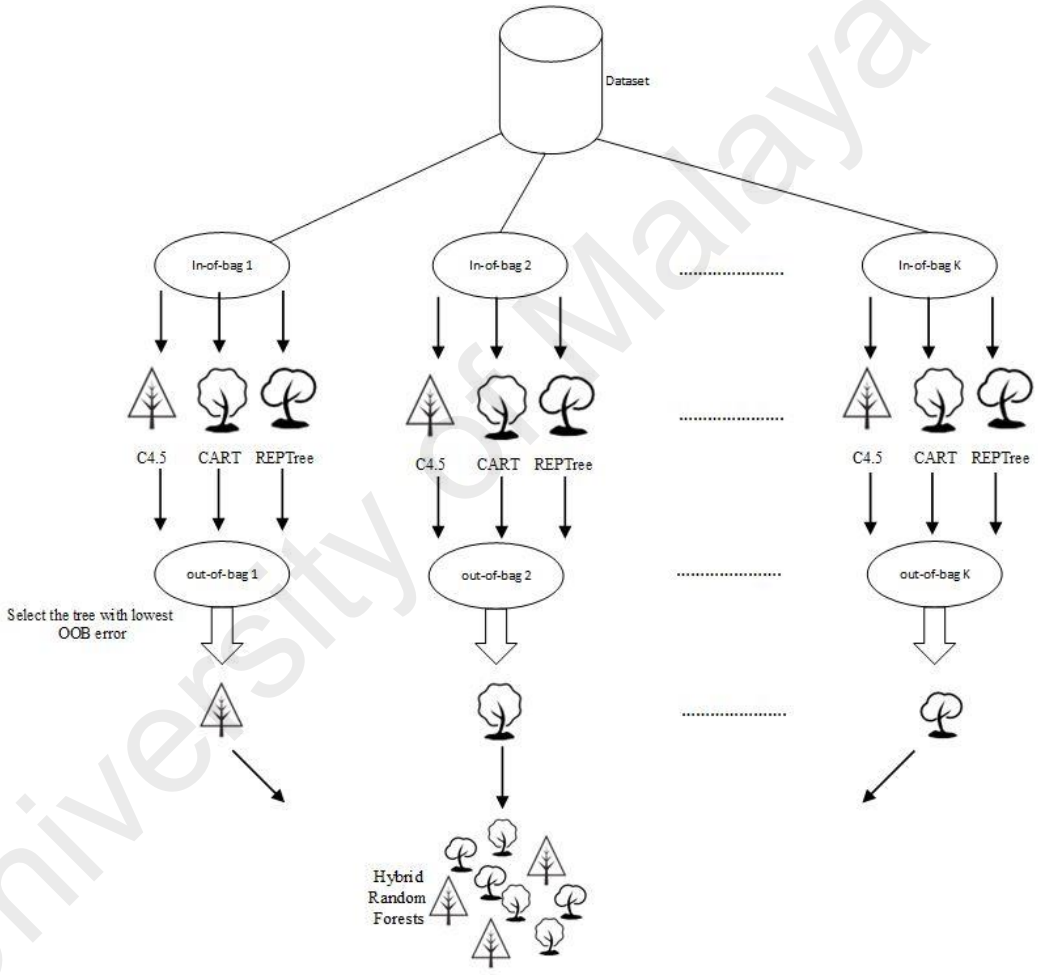


Figure 5.1: Creating RF by combining three different types of trees

5.3 Applying Genetic Algorithm

Figure 5.2 describes the GA operation on the hybrid RF. Here, the produced hybrid RF is used as a pool of genes for the new algorithm (HGRF). The initial population for GA is built by selecting a random number of genes (trees) from the hybrid RF to create random chromosomes. As depicted in Figure 5.2, GA evolves the initial population in several generations. Finally, we select a chromosome with the highest fitness value as the best forest.

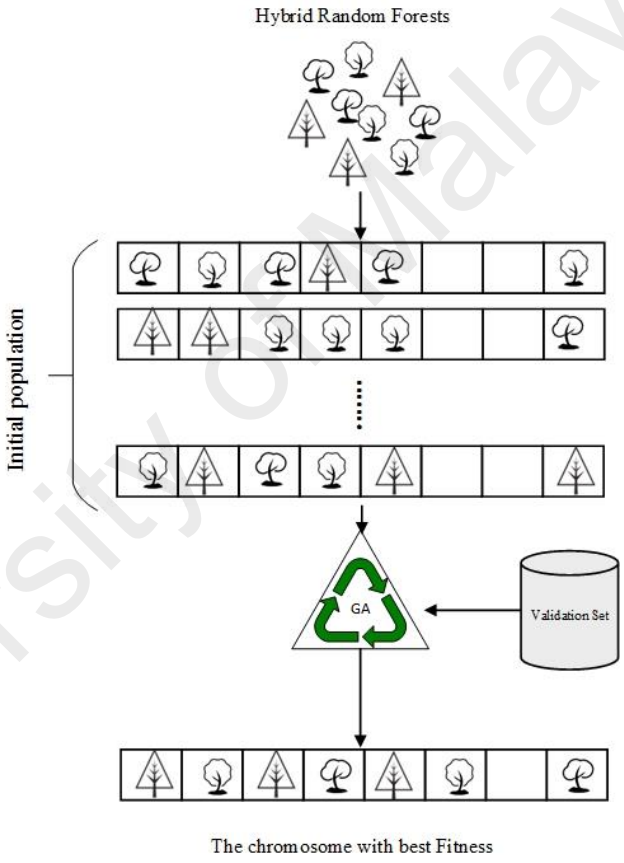


Figure 5.2: Genetic algorithm operation on the hybrid RF

During the implementation of HGRF, we are not sensitive for selecting repeated genes in the chromosome. Therefore, it is possible that the same gene/tree repeats in one chromosome. Standard uniform mutation is employed for the mutation process.

Randomly, we replace two of the genes in the chromosome with each other. Mutation helps GA not to be in local maximum and instead try to find the global maximum. By replacing some genes with others, we guarantee that one-point crossover spreads this randomness to the next generation.

In data partitioning, we applied 10-fold cross-validation with a small change. As mentioned, HGRF needs a validation set as well. We divided the test set into validation and test sets during the cross-validation process. The training set is used for creating the hybrid RF in the first phase and validation will be used in the second phase with GA. At the end, the accuracy of the algorithm can be calculated with the test set. For comparing HGRF with other well-known classifiers, the same process follows without the validation set. In this way, we keep the experimental conditions equal among various classifiers.

i. Algorithm 1. HGRF Algorithm

{User Settings}
Input N, //number of trees
M, //number of features
T, //number of different types of RandomForests
S, //number of forests/chromosomes
NG //number of generations
RF[0]=call C45RandomForest(N,M)
RF[1]=call CARTRandomForest(N,M)
RF[2]=call RepTreeRandomForest(N,M)
For i=1->N **do**
bestTree= RF[0][i]
OOB_Err= RF[0][i].out_of_bag()
For j=1->T **do**
If(RF[j][i].out_of_bag()<OOB_Err
OOB_Err= RF[j][i].out_of_bag()
bestTree= RF[j][i]
end if
pool[j]=bestTree
End for
End for

For i=1->S **do**
For k=1->n **do**
x=Random(1->N)
Add pool[x] into Forest/Chromosome i and gene number k in P0
End for
End for
Evaluate each forest in the initial population P0

For j=1->NG **do**
{Generate a new population by applying GA: operators mutation and crossover}
Pnew=GAOperators(P)
Evaluate each forest in P
bestForest<-copy of the best P
P=Pnew
End For
{output}
A vector of trees bestForest

ii.

GA has some configurations, such as chromosome size, population size, number of generations and, more importantly, fitness function. Chromosome size refers to the number of the genes (trees here) in each chromosome (forest). The population size is the number of chromosomes in each population. The maximum number of generations can also be important for us. The core of the GA is the fitness function. It measures how well our current generation is and evolves the next generation in the correct direction. In our case, we are looking for the best forest or chromosome that classifies the validation set with higher accuracy.

According to our assumption, an instance in the validation set is considered as correctly classified if the majority of the genes/trees are correctly assigned. On the other hand, the instance is incorrectly classified if the assignment of most of the genes/trees in the chromosome categorization fails. There are some cases where the numbers of trees/genes that are classified correctly and incorrectly are equal. In this situation, a "tie" has occurred.

For calculating the fitness, we apply the following formula:

$$f(v) = \sum_{i}^{K} c(v,i) + \frac{t(v,i)}{K} \quad (5\text{-}2)$$

In formula 5-2, *K* is the number of instances in the validation set. *c(v,i)* returns one if the instance *i* with the majority of the genes/trees is classified correctly and it returns zero otherwise. *t(v,i)* indicates if instance *i* in chromosome *v* is a *tie* or not. In the case of a tie (when classifiers assign equally to both classes), *t(v,i)* returns one and otherwise it returns zero.

5.4 HGRF Evaluation

To be confident that HGRF is working on our dataset, it was necessary to test the proposed algorithm with some of the standard datasets and compare the results with baseline classifiers. We choose traditional RF, Genetic Algorithm Random Forests (GARF) and AdaBoost as the baseline for ensemble classifiers, and C4.5 is chosen as a powerful individual classifier. RF is chosen because HGRF was inspired from RF and it attempts to improve it. In addition, HGRF is an ensemble classifier; it is interesting to compare its results with Adaboost, as a successful ensemble classifier. Although C4.5 is an individual classifier, it is newer than other algorithms, very popular in machine learning, and is referred to by many researchers (Chang, Lin & Wang, 2009; Puuronen, Terziyan & Tsymbal, 1999; Xu, Huang, Williams, Li et al., 2012; Xu, Huang, Williams, Wang et al., 2012).

Weka 3.6 was used for implementing AdaBoost and C4.5 where exists in Weka package as J48 algorithm. GARF is implemented in the Java environment, along with using some of the Weka classes to make this experience easier. Moreover, for simulating the genetic algorithm, the Genetic Algorithm Package (JGAP) is used and customized for use in HGRF. However, it was necessary for this experiment to add new classes and methods into the basic package.

We tested HGRF with various configurations, such as different numbers of chromosomes, different numbers of the genes in each chromosome, various probabilities of mutation and crossover. These experiments were done on a diabetes dataset from UCI. As shown in Figure 5.3, the best probability for mutation was around 0.7 and for crossover probability, 0.9 provided the best results.

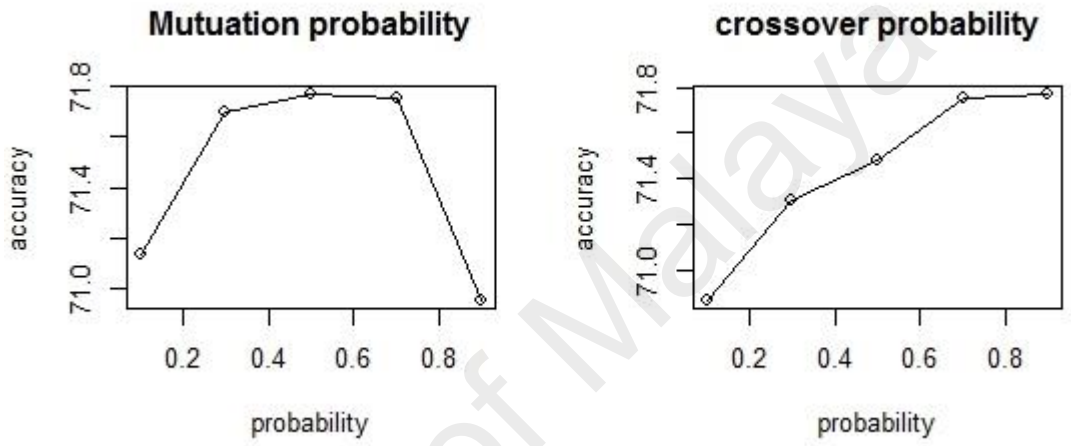


Figure 5.3: Impact of Mutation and Crossover probability over accuracy

We repeated this test to get the best values for the number of the genes (trees) in each chromosome (forest) and the number of generations that GA should optimize for the random forest results. Obtaining the best number of the genes happens when chromosomes are between 50 and 100. We continued the GA for 50 continuous generations. This information is depicted in Figure 5.4. We kept these variables during our experiments with other datasets as well.

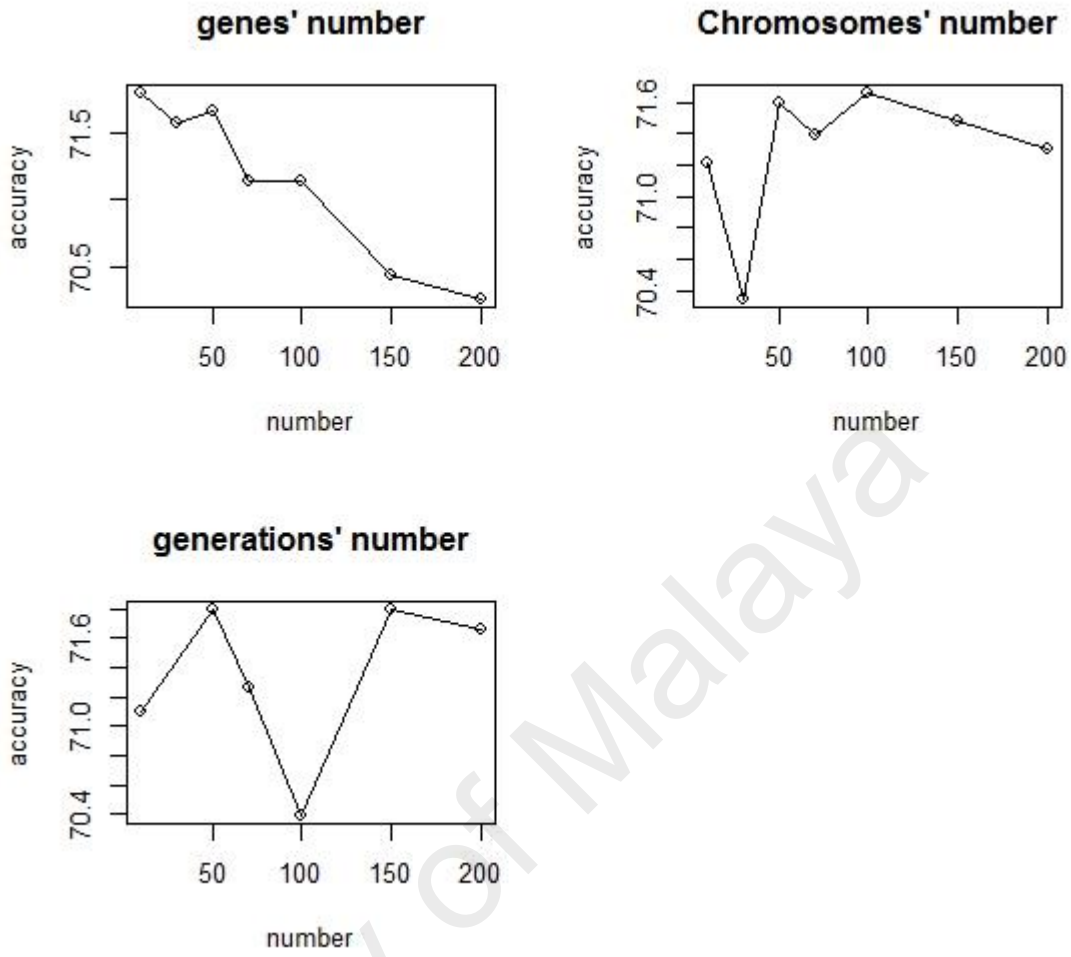


Figure 5.4: Impact of number of the genes, chromosomes and generation over accuracy

In this experiment, we used 20 standard datasets of the UCI repository. The UCI Machine Learning Repository is a collection of databases, domain theories and data generators that are used by the machine-learning community for the empirical analysis of machine-learning algorithms. This archive was created in 1987 by David Aha and his fellow graduate students at UC Irvine (Lichman, 2013). We selected various ranges of datasets with different sizes, target classes, and features.

The audiology dataset was created at Austin University in that school's medical college. It has 24 classes and 70 attributes. Annealing relates to steel annealing data,

with 798 case and 38 features. In Table 5.1, the specifications of each dataset are described. A balance-scale is generated to model the psychological experiments with three classes and five features over 625 cases. Colic or Horse-colic has collected data about the chance of survival among sick horses; 23 elements are measured for each case. Diabetes data was provided by the National Institute of Diabetes and Digestive and Kidney Diseases to UCI, and was represented by eight different attributes and two classes. The Glass dataset consists of six types of glasses' information based on their oxide content. Heart-statlog is a heart disease database with 14 various features. Ionosphere is the classification of radar returns from the ionosphere with 35 various attributes. Labor dataset was collected from the Collective Bargaining Review. The data includes all collective agreements reached in the business and personal services sector for locals with at least 500 reviews in Canada. Character image features were collected in a letter dataset with 26 classes. Lymph consists of 148 cases of lymphography data, provided by University Medical Centre. Targets are categorized into four classes: normal find, metastases, malign lymph, fibrosis. The Mushroom dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the *Agaricus* and *Lepiota* families. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. The Vehicle dataset includes 3D objects within a 2D image through the application of an ensemble of shape feature extractors to the 2D silhouettes of the objects. It has four different classes with 18 attributes. The Vote dataset originates from 1984 United Stated Congressional Voting Records and classifies candidates into Democrats and Republicans.

The Vowel dataset consists of a three-dimensional array: vowel data [speaker, vowel, input]. The speakers are indexed by integers 0-89 (actually, there are fifteen individual speakers, each saying each vowel six times). The vowels are indexed by integers 0-10.

For each utterance, there are ten floating-point input values, with array indices of 0-9. 990 cases exist in the Vowel dataset.

Information on three classes of waves is collected in the Waveform-500 dataset. It consists of 5000 instances with 40 various features. The Yeast dataset predicts the cellular localization sites of proteins in 10 classes. This data set is introduced by Kenta Nakai from the Institute of Molecular and Cellular Biology. Finally, the Zoo dataset categorizes seven animals with 18 attributes. Table 5.1 summarizes the information of the presented datasets.

Table 5.1: Dataset specification

Name	#Instances	#class	#attributes
Anneal	898	6	39
Audiology	226	24	70
Balance-scale	625	3	5
Colic	368	2	23
Credit-g	1000	2	20
Diabetes	768	2	8
Glass	214	6	10
Heart-statlog	270	2	14
Ionosphere	351	2	35
Labor	57	2	17
Letter	20000	26	16
Lymph	148	4	18
Mushroom	8124	23	2
Soybean	683	19	36
Vehicle	846	4	18
Vote	435	2	16
Vowel	990	11	14
Waveform-500	5000	3	40
Yeast	1483	10	9
Zoo	101	7	18

The results of the experiment are summarized in Table 5.2. Each classifier ran with 10-fold cross-validation over UCI data sets. To make it easier to comprehend, Table 5.2 shows the average of accuracy of various folds. The standard deviations of each classifier over UCI datasets are reported in Appendix B. In 11 datasets, HGRF was the winner. In two cases,

HGRF's result is similar to the best classifiers, and in seven other cases, HGRF is not as successful as the best one. From Table 5.2, we can conclude that in 13 cases, HGRF performs better than traditional RF. However, this is not the accurate way for comparing two classification algorithms. it should get proven that the difference between the results is statistically significant.

Table 5.2: Accuracy of different algorithms on various datasets

Dataset	HGRF	Random Forests	AdaBoost	C4.5
Anneal	**99.17**	92.87	84.44	82.77
Audiology	**77.27**	72.54	44.54	71.81
Balance-scale	**84.83**	81.87	73.54	80.64
Colic	83.73	80.77	**84.44**	82.77
Credit-g	76.12	72.8	72.2	72.6
Diabetes	71.57	70.52	**72.89**	71.57
Glass	**79.23**	70.42	51.72	64.45
Heart-statlog	**85.34**	77.76	79.23	80
Ionosphere	92.35	**93.52**	92.35	91.76
Labor	88.33	80.10	**91.66**	81.66
Letter	94.06	**94.28**	6.51	87.97
Lymph	**86.85**	79.03	74.28	77.14
Mushroom	**100**	**100**	96.18	**100**
Soybean	**87.33**	80.56	73.06	84.75
Vehicle	**73.8**	70.14	39.04	70
Vote	**96.27**	**96.27**	95.84	**96.27**
Vowel	**92.87**	82.85	11.83	81.02
Waveform-500	**83.811**	80.34	67.76	76.52
Yeast	**59.75**	58.13	38.64	57.83
Zoo	95.32	91.05	60	**96**

Researchers usually compare the results of two classifiers with a paired T-test. However, according to Demšar (2006), a T-test needs some conditions to to become applicable. For instance, the sample size should be large enough (approximately over 30 data sets) and the differences between two random variables should distribute normally. The alternative method suggested by Demšar is the Wilcoxon signed-ranks test. Wilcoxon is a non-parametric test that can rank the differences of two classifiers without noticing their signs. In order to compare HGRF with other tested classifiers, we used the Wilcoxon signed-ranks test. We set a null hypothesis that there is no difference between the results of the HGRF and RF, and an alternative hypothesis that there is a difference between them and that HGRF is superior. The earned p-value is 0.004, which is less than a α=0.05 significance level, which assured us that we could reject the null hypothesis and that HGRF is superior to RF. After repeating the Wilcoxon test for other classifiers, we found the same result, that HGRF works better.

5.5 Applying HGRF on ISI and non-ISI datasets

The result of the first experiment assured us that HGRF is an authentic ensemble classifier and that it works well on the most commonly selected datasets. Now, it is time to apply it on the created ISI and non-ISI datasets (Chapter 4). Similar to the experiment in Chapter 4, preprocessing is done on the text to put it in the appropriate format for text classification. During preprocessing, the text is tokenized and converted into the text-document matrix. Furthermore, stopwords are removed from the matrix. The list of these stopwords is presented in Appendix A. As before, the 10-fold cross-validation is used for data partitioning. The test dataset is divided in two parts: validation and test sets. The reason is that after every training, the genetic algorithm needs to practice over data to choose the best trees in each chromosome and evolve to reach the optimum solution. Nevertheless, using the training set itself can affect the classification accuracy,

so it is decided to have a separate set as validationset . This validation part is only used in the HGRF algorithm, although the same test set is used for all other algorithms to measure them in similar conditions. This process is depicted in Figure 5.5.

On the other hand, for testing other baseline classifiers, the same structure is kept to assist us to compare the final results with each other.

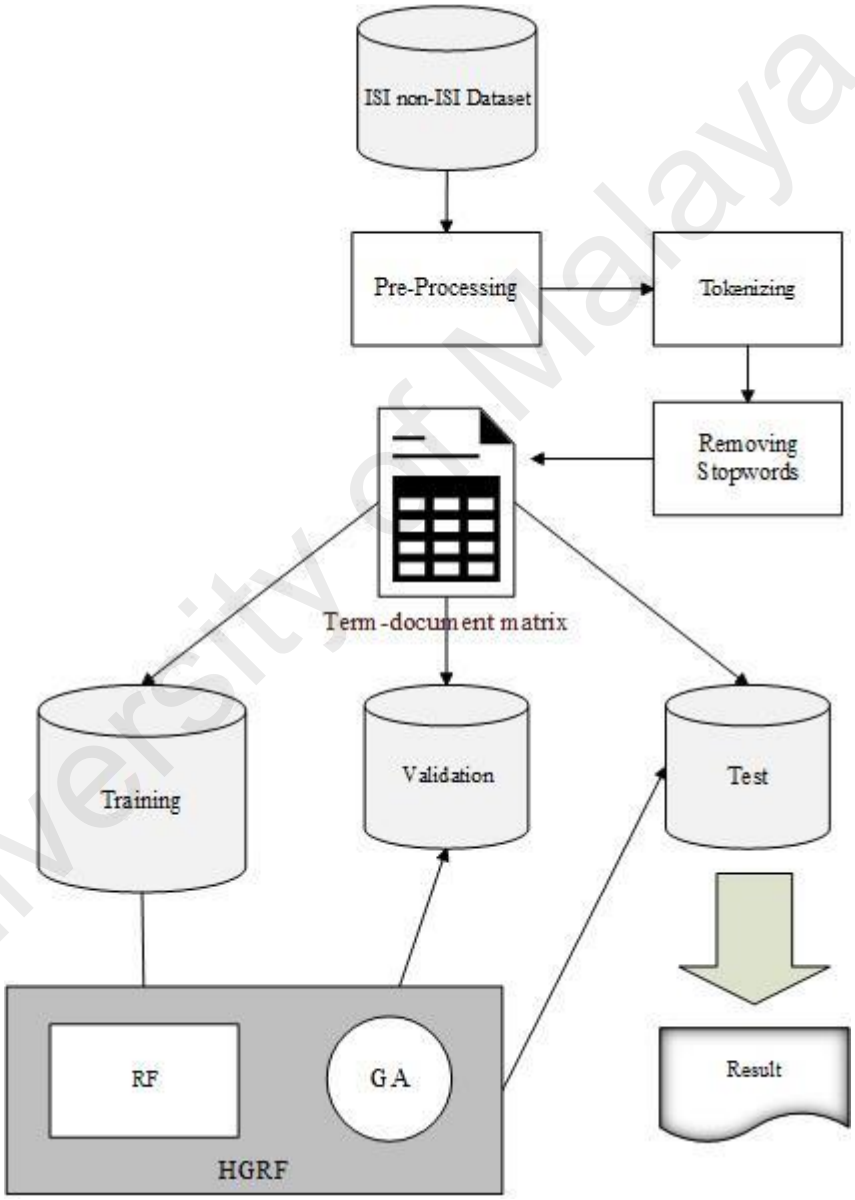


Figure 5.5: Design of the second experiment

The result of this experiment is summarized in Table 5.3. As shown, the best accuracy is reported in the computer science journals by the proposed HGRF algorithm. However, for Business papers, Random Forests performed slightly better in comparison with HGRF.

Table 5.3: Final experiment results

Algorithms	Area	Specificity	Precision	Recall / Sensitivity	Accuracy
SVM	Business	38.11	59.21	90.43	71.42819
	Computer	56.23	68.12	94.44	78.99433
KNN	Business	68.75	63.64	56.87	59.57606
	Computer	54.13	65.15	86.43	74.13695
Naïve Bayes	Business	76.12	73.33	66.33	69.47219
	Computer	57.55	63.16	72.17	67.29091
Random Forest	Business	58.67	68.15	77.87	**73.52145**
	Computer	66.22	80.24	79.32	80.12247
HGRF	Business	60.12	66.52	78.33	72.23879
	Computer	68.12	74.91	90.26	**81.762678**

Based on the collected results, the HGRF algorithm proved that it can differentiate between ISI and non-ISI articles with reliable accuracy. In most cases, HGRF is superior to other baseline algorithms. In the computer science dataset, HGRF had the

best results for differentiating between two various classes. However, RF worked better for business articles (in terms of accuracy measurement). It is not rare in the machine-learning area that the performance of an algorithm does not always lead to the best result. This can be explained by the No Free Lunch Theory, according to this theory a specific classifier is not a solution for every dataset (Wolpert, 2002). The reason for this is intrinsic patterns that exist in each dataset that react to each algorithm in a different way.

5.6 Summary

In this section, the accuracy of the RF algorithm was improved by applying the genetic algorithm on hybrid RF. Using different kinds of trees at the same time ensures an increased diversity in forests and the genetic algorithm. It brings the highest level of diversity through randomness to the RF. By conducting an experiment and comparing its learning accuracy with other ensemble classifiers, it has been shown that HGRF can be a good alternative for standard Random Forests. In order to evaluate the performance of HGRF, the proposed algorithm was tested on 21 different UCI datasets and the results were compared with some baseline algorithms, such as RF, AdaBoost and C4.5. The Wilcoxon signed-ranks test was used to be sure that the result was statistically meaningful. At the end, the final experiment was run and HGRF was applied to the ISI and non-ISI datasets. As expected, the result was promising and HGRF outperformed that classification of Computer Science articles by SVM, KNN and Naïve Bayesian classifiers.

CHAPTER 6: CONCLUSION

6.1 Introduction

In this chapter, we try to briefly summarize the main points of this research, investigate whether this research has answered the questions of the thesis, and determine whether the defined objectives have been satisfied or not. In addition, contributions of this research are discussed and limitations of this research are examined. Finally, possible future work to complement or complete this research will be proposed.

6.2 Discussion

The main goal of this research was to propose a method for differentiating ISI and non-ISI articles from each other and help authors and researchers discover whether their style is similar to ISI journals or not. To answer this question, the problem statement breaks down into several sub-problems.

It is understood that lexical domains of scientific writings are different. However, it was interesting to determine whether this difference can help us differentiate such scripts from each other. Many researchers use classification techniques in the categorization of various scientific scripts. However, to the best of our knowledge, it is rarely used for investigating the quality of text and scripts. Another question that has come up is that, if a classification technique is able to differentiate low- and high-quality academic writing from each other, which technique is more accurate? According to the No Free Lunch Theory, there is not an absolute answer for such questions. So, we had to try a number of experiments to find out.

In order to answer the problem statement, a complete literature review was done (Chapter 2). After getting some insights from previous research studies, our method was proposed in Chapter 3. As discussed earlier (Chapter 3), ISI articles were chosen as high-quality samples of scientific writing. 200 articles were selected from two distinct domains to increase the randomness and decrease dependency. Non-ISI articles were chosen from scientific conferences (Chapter 3). Preprocessing was done for all the data. All scripts were tokenized and stopwords were discarded.

In the next step, three classifiers (KNN, Naïve Bayes and SVM) were tested on the dataset. Each of them was run with different configurations to get the best results. The KNN algorithm was tested with a different number of neighbors. Cross-validation was implemented to make the results more reliable. The results proved that the classification technique was suitable for detecting the quality of scientific writing. In order to answer the question, "what difference exists between the lexicon and semantics of high-quality or low-quality academic writing?" Section 4.8 investigated the syntactical role in scientific writing; ISI and non-ISI styles were compared with each other.

In order to improve the accuracy of classification, Chapter 5 proposed Hybrid Genetic Random Forests (HGRF) as a new classification algorithm. After proposing HGRF, it was applied to the UCI datasets and compared with some classic ensemble classifiers (Random Forests, AdaBoost) and a novel classifier (C4.5) to prove that this proposed algorithm is effective. In the final stage, it was implemented on the ISI and non-ISI article datasets and the results were compared with baseline algorithms that had been tested earlier. The results were promising and the proposed method categorized ISI and non-ISI articles with better accuracy.

6.3 Contribution

The first contribution of this research is advancing a novel method for identifying ISI articles from non-ISI ones. Previously, other researchers had used AI techniques and ML in scientific writing for various reasons, such as improving grammar or detecting plagiarism. However, this is the first time that classification methods were used to discover whether an article has the ability to be published in ISI journals from a writing style point of view. We proved that most classification algorithms have the ability to categorize ISI papers correctly with the proposed method.

The second contribution was creating the ISI and non-ISI dataset. Unfortunately, during this research, we did not find an appropriate dataset in the area, so we created a dataset that consists of ISI and non-ISI papers. As we mentioned in Chapter 3, in order to decrease the probable bias in this research, we built the dataset in two distinct scientific areas. Computer Science and Business were selected due to my personal familiarity with both areas.

The third contribution was proposing the new ensemble classifier. In this research, we focused on Random Forest as one of the most popular ensemble classifiers. Two innovations were made in developing the new algorithm. First, as we know, the RF uses Tree as a weak classifier. Different types of RF have been created by using various kinds of Trees. For instance, RF based on CART or RF based on C4.5. We created three different kinds of Trees for each in-of-bag part and the best technique was selected by testing on out-of-bag partitions.

The result was a forest with a different range of trees. The second innovation was when we passed this Forest into the genetic algorithm. Each chromosome was a small forest with various kinds of trees. At the end of this stage, the best forest was provided by the GA. This algorithm was tested on different standard datasets for validity. Finally, as presented in Chapter 5, the algorithm was applied to the ISI and non-ISI dataset as well.

Moreover, the syntactical role of high-quality scientific articles (ISI articles) was investigated in Section 4.8, which could be helpful for other researchers in future work.

6.4 Future works

This research can enable other scientists to predict whether a paper follows the ISI journal pattern or otherwise. However, one of the challenges of this study was the lack of a standard dataset. We created two datasets, one that contained ISI articles and another that contained non-ISI articles by considering two distinct scientific areas with a hundred papers from each. Access to most of the ISI journals is limited and crawlers cannot access them. Therefore, we had to collect the data manually. This process was slow and time-consuming. Otherwise, more samples from various areas could have been selected. We believe that more samples could lead to a stronger classifier. In addition, this study only focused on two scientific domains (Computer Science and Business) with random keywords. Expanding this domain could lead to a stronger classifier. On the other hand, it would be interesting if this process were repeated for certain specific journals to discover whether such journals are definable by their patterns.

Another important point is that this study focused mainly on the lexicon of academic scripts. It is assumed that if the proposed method of this research is mixed with grammatical specification of the academic papers, the results would be more robust and reliable. There is the room for researchers to work on this idea in future work.

The performance of HGRF was surprisingly good. However, it is suggested that future studies apply and use more kinds of decision trees in HGRF to see how the results change. For instance, using a chi-square tree, BFTree, ADTree, NBTree, etc.

6.5 Summary

Creating an automated system for the identification of ISI and non-ISI papers is helpful for many students, scholars and scientists that intend to submit manuscripts to ISI journals. There are many automated tools for grammar checking at present, but providing such a unique service can accelerate the publishing process and decrease some of the confusion of novice researchers. In addition, this research introduced a new hybrid classification algorithm with results showing that it could be very successful in comparison to its ancestors.

REFERENCES

Afroz, Sadia, Michael Brennan, and Rachel Greenstadt. 2012. "Detecting Hoaxes, Frauds, and Deception in Writing Style Online." In *2012 IEEE Symposium on Security and Privacy*, IEEE, 461–75. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6234430 (August 12, 2013).

Aggarwal, Charu, and ChengXiang Zhai. 2012. "A SURVEY OF TEXT CLASSIFICATION ALGORITHMS." In *Mining Text Data*, , 163–222.

Ahlgren, Per, and Cristian Colliander. 2009. "Document-Document Similarity Approaches and Science Mapping: Experimental Comparison of Five Approaches." *Journal of Informetrics* 3(1): 49–63. http://www.sciencedirect.com/science/article/pii/S1751157708000680.

Akritidis, Leonidas, and Panayiotis Bozanis. 2013. "A Supervised Machine Learning Classification Algorithm for Research Articles." In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, , 115–20.

Al-Daihani, Sultan M, and Alan Abrahams. 2016. "A Text Mining Analysis of Academic Libraries' Tweets." *The Journal of Academic Librarianship* 42(2): 135–43.

Aphinyanaphongs, Yindalon et al. 2014. "A Comprehensive Empirical Comparison of Modern Supervised Classification and Feature Selection Methods for Text Categorization." *Journal of the Association for Information Science and Technology*: n/a – n/a. http://doi.wiley.com/10.1002/asi.23110 (August 18, 2014).

Apté, Chidanand, Fred Damerau, and Sholom M Weiss. 1994. "Automated Learning of Decision Rules for Text Categorization." *ACM Transactions on Information Systems (TOIS)* 12(3): 233–51.

Argamon, Shlomo, Jeff Dodick, and Paul Chase. 2008. "Language Use Reflects Scientific Methodology: A Corpus-Based Study of Peer-Reviewed Journal Articles." *Scientometrics* 75(2): 203–38. http://dx.doi.org/10.1007/s11192-007-1768-y (February 19, 2013).

Batista, Gustavo E A P A, and Maria Carolina Monard. 2003. "An Analysis of Four Missing Data Treatment Methods for Supervised Learning." *Applied Artificial Intelligence* 17(5-6): 519–33. http://dx.doi.org/10.1080/713827181.

Biau, Gérard. 2012. "Analysis of a Random Forests Model." *Journal of Machine Learning Research* 13: 1063–95.

Biber, Douglas, and Bethany Gray. 2010. "Challenging Stereotypes about Academic Writing: Complexity, Elaboration, Explicitness." *Journal of English for Academic Purposes* 9(1): 2–20. http://linkinghub.elsevier.com/retrieve/pii/S1475158510000020 (February 15, 2013).

Bornmann, Lutz, Christophe Weymuth, and Hans-Dieter Daniel. 2009. "A Content Analysis of Referees' Comments: How Do Comments on Manuscripts Rejected by a High-Impact Journal and Later Published in Either a Low- or High-Impact Journal Differ?" *Scientometrics* 83(2): 493–506. http://www.springerlink.com/index/10.1007/s11192-009-0011-4 (February 19, 2013).

Boser, Bernhard E, Isabelle M Guyon, and Vladimir N Vapnik. 1992. "A Training Algorithm for Optimal Margin Classifiers." In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, Pittsburgh, Pennsylvania, USA: ACM, 144–52. http://doi.acm.org/10.1145/130385.130401.

Braam, Robert R., Henk F. Moed, and Anthony F. J. van Raan. 1991. "Mapping of Science by Combined Co-Citation and Word Analysis. II: Dynamical Aspects." *Journal of the American Society for Information Science* 42(4): 252–66. http://doi.wiley.com/10.1002/(SICI)1097-4571(199105)42:4<252::AID-ASI2>3.0.CO;2-G (February 25, 2013).

Breiman, Leo. 1996. "Bagging Predictors." *Machine Learning* 24(2): 123–40.

———. 2001. "Random Forests." *Machine Learning* 45(1): 5–32. http://link.springer.com/article/10.1023/A:1010933404324.

Cameron-Jones, Mike. 2001. "Gradient Descent Style Leveraging of Decision Trees and Stumps for Misclassification Cost Performance." In *AI 2001: Advances in Artificial Intelligence: 14th Australian Joint Conference on Artificial Intelligence Adelaide, Australia, December 10--14, 2001 Proceedings*, eds. Markus Stumptner, Dan Corbett, and Mike Brooks. Berlin, Heidelberg: Springer Berlin Heidelberg, 107–18. http://dx.doi.org/10.1007/3-540-45656-2_10.

Chang, Che-Wei, Chin-Tsai Lin, and Lian-Qing Wang. 2009. "Mining the Text Information to Optimizing the Customer Relationship Management." *Expert Systems with Applications* 36(2): 1433–43. http://linkinghub.elsevier.com/retrieve/pii/S0957417407005520 (February 25, 2013).

Conrad, Susan M. 1996. "Investigating Academic Texts with Corpus-Based Techniques: An Example from Biology." *Linguistics and Education* 8: 299–326.

Corney, Malcolm Walter. 2003. "Analysing E-Mail Text Authorship for Forensic Purposes by." (March).

Cortes, Viviana. 2004. "Lexical Bundles in Published and Student Disciplinary Writing: Examples from History and Biology." *English for Specific Purposes* 23(4): 397–423. http://linkinghub.elsevier.com/retrieve/pii/S0889490603000851 (February 27, 2013).

Coxhead, a. 2012. "Academic Vocabulary, Writing and English for Academic Purposes: Perspectives from Second Language Learners." *RELC Journal* 43(1): 137–45. http://rel.sagepub.com/cgi/doi/10.1177/0033688212439323 (August 5, 2013).

Demšar, Janez. 2006. “Statistical Comparisons of Classifiers over Multiple Data Sets.” *J. Mach. Learn. Res.* 7: 1–30. http://dl.acm.org/citation.cfm?id=1248547.1248548.

Dietterich, Thomas G. 2000. “Multiple Classifier Systems.” In *Multiple Classifier Systems, ,* 1–15. http://www.springerlink.com/index/10.1007/3-540-45014-9.

Do, Thanh Nghi, Philippe Lenca, Stéphane Lallich, and Nguyen Khang Pham. 2010. “Classifying Very-High-Dimensional Data with Random Forests of Oblique Decision Trees.” *Studies in Computational Intelligence* 292: 39–55.

Donaldson, Ian et al. 2003. “PreBIND and Textomy--Mining the Biomedical Literature for Protein-Protein Interactions Using a Support Vector Machine.” *BMC bioinformatics* 4: 11. http://www.biomedcentral.com/1471-2105/4/11.

Eggins, Suzanne. 1994. *Introduction to Systemic Functional Linguistics*. London: Pinter.

Elsevier. 2016. “What Is Peer Review?” *Elsevier*: what – is – peer – review. https://www.elsevier.com/reviewers/what-is-peer-review.

Fang, Zhihui. 2005. “Scientific Literacy: A Systemic Functional Linguistics Perspective.” *Science Education* 89(2): 335–47. http://doi.wiley.com/10.1002/sce.20050 (February 22, 2013).

Finzen, Jan, Maximilien Kintz, and Stefan Kaufmann. 2012. “Aggregating Web-Based Ideation Platforms.” *International Journal of Technology Intelligence and Planning* 8(1): 32–46. http://www.inderscience.com/link.php?id=47376 (May 13, 2013).

Freund, Y, and R E Schapire. 1997. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.” *Journal of Computing Systems and Science* 55: 119–39.

Gaizauskas, Robert, George Demetriou, and Kevin Humphreys. 2000. “Term Recognition and Classification in Biological Science Journal Articles.” In *In Proc. of the Computional Terminology for Medical and Biological Applications Workshop of the 2 Nd International Conference on NLP, ,* 37–44.

Ghanem, Moustafa M, Yike Guo, Huma Lodhi, and Yong Zhang. 2002. “Automatic Scientific Text Classification Using Local Patterns: KDD Cup 2002 (task 1).” *ACM SIGKDD Explorations Newsletter* 4(2): 95–96.

Giannakopoulos, Theodoros et al. 2015. “Visual-Based Classification of Figures from Scientific Literature.” In *Proceedings of the 24th International Conference on World Wide Web, ,* 1059–60.

Gries, David, and Fred B Schneider. 2010. 41 Media *Fundamentals of Predictive Text Mining*. http://www.springerlink.com/index/10.1007/978-1-84996-226-1.

Guo, Yi, Zhiqing Shao, and Nan Hua. 2010. "Automatic Text Categorization Based on Content Analysis with Cognitive Situation Models." *Information Sciences* 180(5): 613–30. http://www.sciencedirect.com/science/article/pii/S0020025509004824.

Gutowitz, Howard A. 1990. "A Hierarchical Classification of Cellular Automata." *Physica D: Nonlinear Phenomena* 45: 136–56.

Halliday, Michael Alexander Kirkwood and Martin, James Robert. 1993. "Grammatical Problems in Scientific English." In *Writing Science: Literacy and Discursive Power*, ed. James Robert Halliday, Michael Alexander Kirkwood and Martin. London: Routledge, 76–94. http://www.scribd.com/doc/16929396/HALLIDAY-Some-Grammatical-Problems-in-Scientific-English-c-4.

Han, Hui et al. 2004. "Two Supervised Learning Approaches for Name Disambiguation in Author Citations." In *Digital Libraries, 2004. Proceedings of the 2004 Joint ACM/IEEE Conference on, ,* 296–305.

Ho, Y. C., and D. L. Pepyne. 2002. "Simple Explanation of the No-Free-Lunch Theorem and Its Implications." *Journal of Optimization Theory and Applications* 115: 549–70.

Hodge, Victoria J., and Jim Austin. 2004. "A Survey of Outlier Detection Methodologies." *Artificial Intelligence Review* 22(2): 85–126. http://link.springer.com/10.1007/s10462-004-4304-y (July 30, 2014).

Hu, Xiao, Stephen Downie, and Andreas Ehmann. 2009. "Lyric Text Mining in Music Mood Classification." In *10th International Society for Music Information Retrieval Conference,* Kobe, Japan, 411.

Islam, Md Rabiul. 2014. "Feature and Score Fusion Based Multiple Classifier Selection for Iris Recognition." *Computational intelligence and neuroscience* 2014: 380585. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4120484&tool=pmcentrez&rendertype=abstract (September 22, 2014).

Jade, Cheng Yu. 2016. "Numerical Optimization." *Aarhus University Denmark*: optimization.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. 103 *An Introduction to Statistical Learning*. New York, NY: Springer New York. http://link.springer.com/10.1007/978-1-4614-7138-7 (July 11, 2014).

Judd, Joel, and Jugal Kalita. 2013. "Better Twitter Summaries?" In *Proceedings of NAACL-HLT*, Atlanta, Georgia, 445–49.

Kerlinger, Fred Nichols. 1986. *Foundations of Behavioral Research*. 3rd ed. Holt, Rinehart and Winston.

Kim, Kyoungok, and Daewon Lee. 2014. "Inductive Manifold Learning Using Structured Support Vector Machine." *Pattern Recognition* 47(1): 470–79.

http://linkinghub.elsevier.com/retrieve/pii/S0031320313003099 (February 20, 2014).

Kim, Soo-min et al. 2006. "Automatically Assessing Review Helpfulness." (July): 423–30.

Ko, Youngjoong, and Jungyun Seo. 2009a. "Text Classification from Unlabeled Documents with Bootstrapping and Feature Projection Techniques." *Information Processing & Management* 45(1): 70–83. http://linkinghub.elsevier.com/retrieve/pii/S0306457308000812 (April 27, 2013).

———. 2009b. "Text Classification from Unlabeled Documents with Bootstrapping and Feature Projection Techniques." *Information Processing & Management* 45(1): 70–83.

Kwon, Oh-Woog, and Jong-Hyeok Lee. 2003. "Text Categorization Based on K-Nearest Neighbor Approach for Web Site Classification." *Information Processing & Management* 39(1): 25–44. http://linkinghub.elsevier.com/retrieve/pii/S0306457302000225 (August 6, 2013).

Lichman, M. 2013. "{UCI} Machine Learning Repository." http://archive.ics.uci.edu/ml.

Lin, Mu-Hua, and Chao-Fu Hong. 2011. "Opportunities for Crossing the Chasm between Early Adopters and the Early Majority through New Uses of Innovative Products." *The Review of Socionetwork Strategies* 5(2): 27–42. http://link.springer.com/10.1007/s12626-011-0019-0 (May 13, 2013).

Lindsay, David. 2011. 14 Veterinary pathology *Scientific Writing=thinking in Words*.

Liu, Bing. 2006. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

———. 2007. 10 New York *Web Data Mining*. http://cs.famaf.unc.edu.ar/~laura/llibres/wm.pdf.gz.

Liu, Yang, Xiaohui Yu, Jimmy Xiangji Huang, and Aijun An. 2011. "Combining Integrated Sampling with SVM Ensembles for Learning from Imbalanced Datasets." *Information Processing & Management* 47(4): 617–31. http://linkinghub.elsevier.com/retrieve/pii/S030645731000097X (August 6, 2013).

Manning, Christopher D, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press.

Manning, Christopher Raghavan, Prabhakar, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Meinshausen, Nicolai. 2006. "Quantile Regression Forests." *Journal of Machine Learning Research* 7: 983–99.

Meneghini, Rogerio, and Abel L Packer. 2007. “Is There Science beyond English? Initiatives to Increase the Quality and Visibility of Non-English Publications Might Help to Break down Language Barriers in Scientific Communication.” *EMBO reports* 8(2): 112–16.

Meyer, David, Friedrich Leisch, and Kurt Hornik. 2003. “The Support Vector Machine under Test.” *Neurocomputing* 55(1): 169–86.

Michalski, R. 1983. “A Theory and Methodology of Inductive Learning.” *Artificial Intelligence* 20: 111–61. http://www.sciencedirect.com/science/article/pii/0004370283900164.

Montes-y-Gómez, Manuel, Alexander Gelbukh, and Aurelio López-López. 2002. “Text Mining at Detail Level Using Conceptual Graphs.” In *Conceptual Structures: Integration and Interfaces*, Lecture Notes in Computer Science, eds. Uta Priss, Dan Corbett, and Galia Angelova. Springer Berlin Heidelberg, 122–36. http://dx.doi.org/10.1007/3-540-45483-7_10.

North, Matthew. 2012a. *Data Mining for the Masses*. Global Text Project.

———. 2012b. *Data Mining for the Masses*. Athens: Global Text Project. http://dl.dropbox.com/u/31779972/DataMiningForTheMasses.pdf.

Orrù, Graziella et al. 2012. “Using Support Vector Machine to Identify Imaging Biomarkers of Neurological and Psychiatric Disease: A Critical Review.” *Neuroscience and biobehavioral reviews* 36(4): 1140–52. http://www.ncbi.nlm.nih.gov/pubmed/22305994 (August 6, 2013).

Panik, Michael J. 2005. *Advanced Statistics from an Elementary Point of View*. 1st editio. Academic Press.

Pennsylvania, University of. 1999. “The Penn Treebank Project.” : treebank. http://www.cis.upenn.edu/~treebank/.

Platt, J. 1999. “Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods.” *Advances in large margin classifiers* 10(3): 61–74. http://www.researchgate.net/publication/2594015_Probabilistic_Outputs_for_Support_Vector_Machines_and_Comparisons_to_Regularized_Likelihood_Methods/file/504635154cff5262d6.pdf (October 25, 2013).

Polikar, R. 2006. “Ensemble Based Systems in Decision Making.” *IEEE Circuits and Systems Magazine* 6(3): 21–45. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1688199 (September 16, 2014).

Polikar, Robi, Lalita Udpa, Satish S. Udpa, and Vasant Honavar. 2001. “Learn++: An Incremental Learning Algorithm for Supervised Neural Networks.” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 31: 497–508.

Porter, M.F. 1980. “An Algorithm for Suffix Stripping.” *Program: electronic library and information systems* 14(3): 130–37. http://www.emeraldinsight.com/10.1108/eb046814 (July 16, 2014).

Puuronen, Seppo, Vagan Terziyan, and Alexey Tsymbal. 1999. “A Dynamic Integration Algorithm for an Ensemble of Classifiers.” In *Foundations of Intelligent Systems*, , 592–600. http://www.springerlink.com/index/A5125P218760RJ04.pdf.

Rank NL. 2015. “Stopwords.” : stopwords. http://www.ranks.nl/stopwords.

“Ranking Criteria and Weights.” 2013. *Shanghairanking*. http://www.shanghairanking.com/ARWU-FIELD-Methodology-2013.html.

Robnik-Sikonja, Marko. 2004. “Machine Learning: ECML 2004.” In *Machine Learning: ECML 2004*, , 12. http://www.springerlink.com/index/u8bgf0xde638byva.pdf.

Rodríguez, Juan J., Ludmila I. Kuncheva, and Carlos J. Alonso. 2006. “Rotation Forest: A New Classifier Ensemble Method.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28(10): 1619–30.

Santos, Terry. 1988. “Professors’ Reactions to the Academic Writing of Nonnative-Speaking Students.” *TESOL Quarterly* 22(1): 69–90. http://www.jstor.org/stable/3587062?origin=crossref (February 22, 2013).

Schapire, Robert E. 1990. “The Strength of Weak Learnability.” *Machine Learning* 5(2): 197–227.

Sebastiani, Fabrizio. 2002. “Machine Learning in Automated Text Categorization.” *ACM Computing Surveys* 34(1): 1–47. http://portal.acm.org/citation.cfm?doid=505282.505283 (July 11, 2014).

Selltiz, Claire, Lawrence S Wrightsman, and Stuart W Cook. 1962. *Research Methods in Social Relations*. Holt, Rinehart and Winston.

Siegler, MG. 2010. “Eric Schmidt: Every 2 Days We Create As Much Information As We Did Up To 2003.” *TechCrunch*. http://techcrunch.com/2010/08/04/schmidt-data/.

De Souto, Marcilio C P, Pablo A Jaskowiak, and Ivan G Costa. 2015. “Impact of Missing Data Imputation Methods on Gene Expression Clustering and Classification.” *BMC bioinformatics* 16: 64. http://www.biomedcentral.com/1471-2105/16/64.

Sudhamathy, G, and C Jothi Venkateswaran. 2012. “Fuzzy Temporal Clustering Approach for E-Commerce Websites.” *International Journal of Engineering and Technology* 4(3): 119–32.

Szarvas, György. 2008. “Hedge Classification in Biomedical Texts with a Weakly Supervised Selection of Keywords.” In *Proceedings of 46th Meeting of the Association for Computational Linguistics*, , 281–89.

Tepper, Allegra. 2012. “How Much Data Is Created Every Minute? [INFOGRAPHIC].” *mashable*. http://mashable.com/2012/06/22/data-created-every-minute/.

Teufel, Simone, Advaith Siddharthan, and Dan Tidhar. 2006. “Automatic Classification of Citation Function.” In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, , 103–10.

“Thomson Corporation Acquired ISI.” 1991. *Online*. https://www.highbeam.com/doc/1G1-12394745.html.

Thorleuchter, Dirk, and Di Van den Poel. 2013a. “Web Mining Based Extraction of Problem Solution Ideas.” *Expert Systems with Applications* 40(10): 3961–69. http://www.sciencedirect.com/science/article/pii/S095741741300016X.

Thorleuchter, Dirk, and Dirk Van den Poel. 2013b. “Protecting Research and Technology from Espionage.” *Expert Systems with Applications* 40(9): 3432–40. http://www.sciencedirect.com/science/article/pii/S0957417412012924.

———. 2013c. “Technology Classification with Latent Semantic Indexing.” *Expert Systems with Applications* 40(5): 1786–95. http://www.sciencedirect.com/science/article/pii/S0957417412010779.

———. 2013d. “Weak Signal Identification with Semantic Web Mining.” *Expert Systems with Applications* 40(12): 4978–85. http://www.sciencedirect.com/science/article/pii/S0957417413001528.

Thorleuchter, Dirk, Dirk Van den Poel, and Anita Prinzie. 2010. “Mining Ideas from Textual Information.” *Expert Systems with Applications* 37(10): 7182–88. http://linkinghub.elsevier.com/retrieve/pii/S0957417410002848 (May 13, 2013).

Tseng, Yuen-Hsien, Chi-Jen Lin, and Yu-I Lin. 2007. “Text Mining Techniques for Patent Analysis.” *Information Processing & Management* 43(5): 1216–47. http://linkinghub.elsevier.com/retrieve/pii/S0306457306002020 (February 15, 2013).

Tumer, Kagan, and Joydeep Ghosh. 1996. “Analysis of Decision Boundaries in Linearly Combined Neural Classifiers.” *Pattern Recognition* 29(2): 341–48.

Uccelli, P., C. L. Dobbs, and J. Scott. 2012. “Mastering Academic Language: Organization and Stance in the Persuasive Writing of High School Students.” *Written Communication* 30(1): 36–62. http://wcx.sagepub.com/cgi/doi/10.1177/0741088312469013 (February 15, 2013).

Vapnik, Vladimir, and Corinna Cortes. 1995. “Support-Vector Networks.” *Machine Learning* 20(3): 273–97.

Vapnik, Vladimir, and AJ Lerner. 1963. "Generalized Portrait Method for Pattern Recognition." *Automation and Remote Control* 24(6): 774–80.

Verikas, a., a. Gelzinis, and M. Bacauskiene. 2011. "Mining Data with Random Forests: A Survey and Results of New Tests." *Pattern Recognition* 44(2): 330–49. http://dx.doi.org/10.1016/j.patcog.2010.08.011.

Wolpert, David H. 2002. "The Supervised Learning No-Free-Lunch Theorems." In *Soft Computing and Industry*, Springer, 25–42.

Wolpert, David H., and William G. Macready. 1997. "No Free Lunch Theorems for Optimization." *IEEE Transactions on Evolutionary Computation* 1: 67–82.

Woods, K., W.P. Kegelmeyer, and K. Bowyer. 1997. "Combination of Multiple Classifiers Using Local Accuracy Estimates." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(4): 405–10. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=588027 (September 22, 2014).

Woźniak, Michał, Manuel Graña, and Emilio Corchado. 2014. "A Survey of Multiple Classifier Systems as Hybrid Systems." *Information Fusion* 16: 3–17. http://linkinghub.elsevier.com/retrieve/pii/S156625351300047X (July 11, 2014).

Wu, Xindong et al. 2007. 14 Knowledge and Information Systems *Top 10 Algorithms in Data Mining*. http://link.springer.com/10.1007/s10115-007-0114-2 (April 29, 2014).

Wu, Xindong, and Kumar Vipin. 2009. *The Top Ten Algorithms in Data Mining*. illustrate. Taylor & Francis.

Xu, Baoxun, Joshua Zhexue Huang, Graham Williams, Qiang Wang, et al. 2012. "Classifying Very High-Dimensional Data with Random Forests Built from Small Subspaces." *International Journal of Data Warehousing and Mining* 8(June): 44–63.

Xu, Baoxun, Joshua Zhexue Huang, Graham Williams, Mark Junjie Li, et al. 2012. "Hybrid Random Forests: Advantages of Mixed Trees in Classifying Text Data." In *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, , 147–58.

Ye, Yunming et al. 2013. "Stratified Sampling for Feature Subspace Selection in Random Forests for High Dimensional Data." *Pattern Recognition* 46(3): 769–87. http://dx.doi.org/10.1016/j.patcog.2012.09.005.

Yoon, Seok-Ho, Sang-Wook Kim, Ji-Soo Kim, and Won-Seok Hwang. 2011. "On Computing Text-Based Similarity in Scientific Literature." In *Proceedings of the 20th International Conference Companion on World Wide Web*, , 169–70.

Zhang, Lei. 2012. "Aspect and Entity Extraction from Opinion Documents." University of Illinois.

Zhang, Shichao, Chengqi Zhang, and Qiang Yang. 2002. “Data Preparation for Data Mining.” *Applied Artificial Intelligence* 17: 375–81.

Zheng, Wu, and Catherine Blake. 2015. “Using Distant Supervised Learning to Identify Protein Subcellular Localizations from Full-Text Scientific Articles.” *Journal of biomedical informatics* 57: 134–44.

Zhu, Jingbo, Huizhen Wang, Benjamin K Tsou, and Matthew Ma. 2010. “Active Learning With Sampling by Uncertainty and Density for Data Annotations.” *IEEE Transactions on Audio, Speech, and Language Processing* 18(6): 1323–31. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5272205 (January 4, 2014).

LIST OF PUBLICATIONS AND PAPERS PRESENTED

1. M. Moohebat, R.G. Raj, D. Thorleuchter and S. Abdul- Kareem. 2016, *LINGUISTIC FEATURE CLASSIFYING AND TRACING*. Malaysian Journal of Computer Science. (ISI-Indexed)
2. Mohammadreza Moohebat, Ram Gopal Raj, Dirk Thorleuchter and Sameem Binti Abdul Kareem, 2015, *Identifying ISI Indexed Papers by Their Lexical Usage: A Text Analysis Approach*, Journal of the American Society for Information Science and Technology, Vol 66(3): pp.501-511. (Tier 1) (ISI/SCOPUS Indexed Publication)