Sudan University of Science & Technology

College of Graduate Studies

Detecting Fraudulent Transactions in Automobile Insurance Industry Using Ensemble Models

كشف المعاملات الاحتيالية في صناعة تأمين السيارات باستخدام النماذج التجمعية

Submitted for the Degree of Doctorate of Philosophy in Computer Science

By

Amira Kamil Ibrahim Hassan

Supervisor

Professor Dr. Ajith Abraham

November 2015

Dedication

By the name of Allah, Most Gracious, Most Merciful

I dedicate this thesis to the soul of my loving father

Professor ***Kamil Ibrahim Hassan***

my adviser, supporter and backbone and to whom the finishing my doctoral studies would made happy and proud, since he was always encouraging me to go forward and never give up or stop.

Acknowledgements

My journey toward fulfilment of the doctoral studies at the Sudan University of Science & Technology at Faculty of Computer Science and Information Technology was one of the most valuable experiences in my life. It would not have been possible without all those who encouraged and supported me during this process. I would like to gratefully acknowledge them here.

First I wish to express my sincere gratitude to my supervisor; Professor Dr. Ajith Abraham, who provided me with the needed guidance, encouragements and motivation in a friendly way. It was truly a pleasure for me to work under his supervision.

Thanks are extended to the generous staff of SUST, particularly Faculty of Computer Science and Information Technology staff. I would like to express my sincere gratitude to Prof Izzeldin Mohammed Osman, Dr. Mohammed El hafiz and Dr. Talaat M. Wahby for their support and unlimited help.

I would like to extend my appreciation to my friends Wafaa Fisal, Limia H. Rahamatalla and Eiman M. El Hassan for their endless help, support and encouragement.

My special thanks to my mother for her unconditional support and love throughout my life. I thank my husband and kids, who have supported me like nobody else, despite the disturbance that this work had on our family life. I thank them for their tolerance, and their smiles.

Abstract

Detection of fraudulent claims has become high priority and technology-laden problem for insurers. This was not always so. Until the early 1980s, the polite way to discuss under-writing and claims settlement fraud was to include it with other potential adverse actions by policy holders and claimants under the rubric of moral hazard.

Fraud in insurance companies is not rare incident, it occurs frequently causing many losses. Companies are often reluctant to admit the scale of fraud problems to their shareholders and policy-holders. Moreover, increasing consumer awareness means that one badly-handled claim can undo millions of pounds worth of advertising. Their main aim is to distinguish a fraudster from a genuine claimant. Data mining can minimise some of these losses by making use of the massive collections of customer data. All insurance companies' main aim is to minimize fraud incident so as to prevent losses, gain the customers' confidence by taking the correct decisions about the validity of an accident, and this can be done by using fraud detection models using data mining base-classifiers and ensemble combining classifiers.

Fraud detection is faced with some problems; limitations or poor quality in the data itself, highly imbalance data and making understandable predictions. The limitation of data was solved by used benchmark data that comprises the information regarding the various automobile insurance claims during the period 1994–1996. It consists of 15,420 samples of which 14,497 are non-fraudulent and 923 are fraudulent, which means that it is imbalance dataset with 94% genuine samples and 6% fraudulent samples.

This research proposes a solution to the imbalance dataset problem which is faced by most fraud detection modelling; by applying six novel proposed partitioning-undersampling techniques. To evaluate those six techniques the resulting subsamples and the original dataset were used to design IFD (Insurance Fraud Detection) models using decision tree classifier. Based on the results, partitioning-undersampling technique using 50:50 fraud: legal ratio and sampling with replacement, is chosen as the best resampling technique; to be used throughout the whole research.

Another two base-classifiers were used to design IDF models. The best models designed used DT, SVM and ANN were used to form ensemble models. Then three ensemble combining classifiers were used to form IFD ensemble models. For building IFD ensemble model, base-classifier models and different combinations of base-classifiers models, were used.

This research developed a novel insurance fraud detection model using classification algorithms for detecting fraudulent insurance claims, using base-classifiers and ensemble combining classification techniques. This research also investigates the effect of using a novel resampling technique (partitioning-undersampling) on in imbalanced fraud detection dataset. A novel comparison was conducted between base-classifier models and the ensemble-combining-classifiers models, which is useful for a better understanding of the performance of various models and can be used to create effective models for insurance fraud detection model.

المستخلص

اصبح اكتشاف الاحتيال في مطالبات التامين من المشاكل تقنيه ذا اولويه عاليه بالنسبه لشركات التامين. وهذا لم يكن دائما كذلك. حتى اوئل 1980 ، الطريقة التبعه لمناقشة الادعائات و المطالبات المكتوبه بغرض الاحتيال كانت تتم تسويتها بادراجها ضمن الاجراءات حملة الوثائق متعسره و المدعي تحت اطار اخلاقي.

الاحتيال في شركات التأمين ليست حادثة نادرة، ويحدث في كثير من الأحيان مما تسبب في العديد من الخسائر. الشركات غالبا ما يترددون في الاعتراف بمشاكل الاحتيال لمساهميها وأصحاب شهادة التامين. زيادة وعي المستهلك بحالات الاحتيال يعني أن حالة احتيال واحدة تم التعامل معها بصورة سيئة يمكن ان تلغي تاثير اعلانات بالملايين من الجنيهات. الهدف الرئيسي منها هو التمييز بين المحتال و المطالب حقيقي. استخدم التنقيب البيانات (Data Mining) يمكن أن تقلل بعض من هذه الخسائر من خلال الاستفادة من مجموعات ضخمة من بيانات العملاء. الهدف الرئيسي لجميع شركات التأمين هو تقليل الحوادث الاحتيال وذلك لمنع الخسائرو كسب الثقة العملاء من خلال اتخاذ القرارات الصحيحة حول صحة وقوع حادث، وذلك عن طريق استخدام نماذdج الكشف عن الغش باستخدام المصنفات و تصنيفات موحده (ensemble combining classifiers).

يواجه الكشف عن الغش بعض المشاكل؛ اولا سوء نوعية البيانات و توفرها، مشكلة البيانات غير متوازنة وجعل التوقعات مفهومة. تم حل مشكلة سوء نوعية البيانات و توفرها بستخدام قاعدة البيانات القياسي والتي تضم المعلومات المتعلقة بمختلف مطالبات التأمين على السيارات خلال الفترة 1994-1996. وهي يتألف من 15420 عينات منها 14497 مطالبات حقيقيه و 923 هي مزورة، وهي قاعدة بيانات غير المتوازنة بعينات حقيقية 94٪ و 6٪ عينات مزورة.

يقترح هذا البحث عن حل لمشكلة عدم التوازن بيانات التي تواجهها معظم نماذج الكشف عن الغش؛ من خلال تطبيق ستة رواية تقنيات partitioning-undersampling المقترحة. لتقييم تلك التقنيات استخدمت العينات الجزئيه الناتجة عنها و مجموعة البيانات الأصلية لتصميم IFD (Insurance Fraud Detection) نماذج باستخدام المصنف شجرة القرارات(Decision Tree) . وبناء على نتائج تقييم هذه النماذج تم اختيار undersampling باستخدام 50:50 نسبة حالات الاحتيال: حالات قانونية وأخذ العينات مع استبدال، هي أفضل تقنية لاستخدامها في باقي البحث.

بعد ذلك استخدمت اثنين من المصنفات ، آلة المتجه الداعم (support vector machine) والشبكات العصبانية الاصطناعية (Artificial Neural Network) واستخدمت لتصميم نماذج IFD، وذلك باستخدام تقنية تقسيم البيانات الي عينات التي أعطت أفضل النتائج.

استخدمت ثلاثة تصنيفات موحده (ensemble combining classifiers) لتشكيل نماذج IFD. لبناء نموذج IFD موحده ، استخدمت نماذج المكونه باستخدم المصنف الاساسيه وتركيبات مختلفة من هذه النماذج.

المساهمة الرئيسية لهذا البحث هو تطوير نموذج الكشف عن الاحتيال في مجال التأمين رواية باستخدام الخوارزميات تصنيف للكشف عن مطالبات التأمين الاحتيالية، والتحقيق في أثر استخدام تقنية اختزال رواية (تقسيم undersampling) على في غير متوازن بيانات الكشف عن الغش، ddesign رواية وقد أجريت نموذج IFD باستخدام فرقة الجمع بين تقنيات تصنيف ومقارنة جديدة بين نماذج قاعدة المصنف ونماذج المصنفات-الجمع بين الفرقة.

List of Publications

- Hassan, Amira Kamil Ibrahim, and Ajith Abraham. "Modeling consumer loan default prediction using ensemble neural networks." In Computing, Electrical and Electronics Engineering (ICCEEE), 2013 International Conference on, pp. 719-724. IEEE, 2013.
- Hassan, Amira Kamil Ibrahim, and Ajith Abraham. "Computational Intelligence Models for Insurance Fraud Detection: A Review of a Decade of Research." *Journal of Network and Innovative Computing* 1, no. 2013: 341-347.
- Hassan, Amira Kamil Ibrahim, and Ajith Abraham. "Modeling Insurance Fraud Detection Using Imbalanced Data Classification." In *Advances in Nature and Biologically Inspired Computing*, pp. 117-127. Springer International Publishing, 2016.
- Modeling Insurance Fraud Detection Using Ensemble Combining Classification, Hassan, Amira Kamil Ibrahim, and Ajith Abraham, International Journal of Computer Information Systems and Industrial Management Applications. ISSN 2150-7988 Volume 8 (2016) pp. 257-265 © MIR Labs.

TABLE OF CONTAINS

LIST OF TABLES

LIST OF FIGURES

List of Abbreviations

AI	Automobile insurance
AIFD	automobile insurance fraud detection
ANN	Artificial Neural Network
API	application programming interface
BFM	Bayesian style frequency matrix
CI	Crop insurance
CT	Consolidated Trees
DM	Data mining
DT	Decision Tree
FN	false negative (fraud been classified as non-fraud)
FP	false positive *(non-fraud been classified as fraud)*
GUI	graphical user interface
HI	Healthcare insurance
IFD	Insurance Fraud detection
IFDANN	insurance fraud detection artificial neural network
IFDDT	insurance fraud detection decision tree
IFDSVM	insurance fraud detection support vector machine
LR	Logistic Regression
MCLP	Multiple Criteria Linear Programming
MLP	Multilayer perceptron
NB	Naïve Bayes
ROS	Random Oversampling
RUS	Random Undersampling
SVM	Support Vector Machine
SVM-RFE	Support Vector Machine Recursive Feature Elimination
TN	true negative *(fraud been classified as fraud*
TP	true positive *(non-fraud been classified as non-fraud)*
UDM	Ubiquitous Data Mining

1 CHAPTER ONE
INTRODUCTION

1.1 Background

"Insurance Industry, as one of the most important instruments of the financial development in advanced and developing countries, has continually attracted the attention of economical theorists, since the Insurance Influence Coefficient is counted as an important index for indicating economic development"[1] .

The Oxford English dictionary defines fraud as "criminal deception", "the use of false representations to obtain an unjust advantage" and "to injure the rights and interests of another". Fraud takes many diverse forms and is extremely costly to society. It can be classified into three main types, namely against organizations, government and individuals. This study focuses on fraud against organizations.

Fraud in insurance companies is not a rare incident, it occurs frequently causing the company many losses. All insurance companies' main aim is to minimize fraud incident for preventing losses and gaining the customers' confidence by taking the correct decisions about the validity of an accident. Fraud involves one or more persons who intentionally act secretly to deceive the company for their own benefit. Fraud is as old as humanity itself and can take an unlimited variety of different forms.

Traditional ways of data analysis have been in use since long time as a method of detecting fraud. They require complex and time-consuming investigations that deal with different domains of knowledge like financial, economics, business practices and law. Fraud management is a knowledge-intensive activity.

The main Artificial Intelligence techniques used for fraud management includes data mining to classify, cluster, and segment the data and automatically find associations and rules in the data that may signify interesting patterns, including those related to fraud [2].

Data mining (DM) has emerged as one of the most important research areas in recent decades. DM aims to extract useful knowledge from data and identify significant patterns. Today, the size of the data in different domains is continuously growing due to advanced computational technology and the reduced cost of storage. The real data are frequently affected by outliers, uncertain labels, and uneven distribution of classes (imbalanced data). We call this type of data imperfect data. This work will help to address the unavoidable difficulty of data uncertainty that occurs in many real world problems.

1.2 Aims and Significance

Fraud deception is a costly problem for many profit organisations. Fraud in insurance companies is not rare incident, it occurs frequently causing many losses [1]. Insurance companies began to realize the importance of fraud in the beginning of the 1980's. They began to fight insurance fraud through the forming of experts teams of skilled claims adjusters with expert skills and training in claims investigation [3]. While this originally happened in the 1990's in the United States, the design quickly spread though out the world [4]. Insurance companies had created very inclusive measures to try to prevent fraud, by the end of 1990's. In the late 1980's and early 1990's studies were undertaken to carry out a complete review of fraud and the development of measures for categorizing fraud. These studies led to the development of a number of measures, which were indicators of fraud, these became known universally as "red flags". These red flags could be used by claims adjusters to help recognize or point out potential fraudulent claims.

It became possible to find more and more data regarding claims due to the computerization of claim systems. Business intelligence systems and reporting techniques could be used to produce a list of potential fraudulent claims. This led to the development of a two stage method for dealing with fraudulent claims. During the first stage the claim is tested by a claims adjuster to analyse the amount of responsibility the insurance company may have for the payment of the claim. Claims, which are for small amounts or appear to be non-fraudulent, are settled in a normal manner. Those that are irregular or are for substantial amounts of money are referred onto a team of (greatly experienced claim handlers) experts for further review.

However as always with such manual systems the problem is the .examination, analysis and creating of potential fraudulent claims lists takes time. Early detection of these potentially fraudulent claims and non fraudulent claims by data mining classification algorithms would help to solve this problem. This could save an insurance company time by having to use less human experts (claims adjusters) to investigate claims. Also by increasing the numbers of claims that are checked, the insurance company would be able to discover more fraudulent claims. The use of a machine learning element for classification of possible fraudulent and non fraudulent claims by a binary classification system would help the experts to scanning for possible fraud and processing the claim [5].

Insurance Service Office (ISO), a US based insurance research consultancy (ISO 2012), has found that the most common approaches to fraud detection depend on human experts. While 80 percent of insurance companies used red flags or indicator cards to carry out fraud checks, less than 25 percent use data mining approaches to detect fraud. This shows that the use of data mining is growing in this area. Also, a data mining approach to fraud detection would allow the use of fewer resources. Large volumes of claims could be automatically screened by classifiers to label them as potentially fraudulent or non-fraudulent. Data mining can minimise some of these losses caused by fraudulent claims, by making use of the massive collections of customer data. All insurance companies' main aim is to minimize fraud incident so as to prevent losses and gain the customers' confidence by taking the correct decisions about the validity of an accident.

1.3 Problem Statement

Fraud detection is faced with several problems. In fraud detection datasets the minority class is the fraud-class, as a result of this the dataset is imbalance dataset. All previous studies using credit transactional and insurance data were faced with data imbalance problem[6].

Fraud detection using data mining solutions is faced with another problem that is the unavailability of data, since this type of data the majority of companies don't like to give; on the other hand there are very few open source datasets available. It was found that 80% of the published papers in the area of fraud detection used imbalanced data set [7]. Finally as fraudsters are adept at developing new methods for perpetrating fraud, any model used to counter fraud must be highly adaptable.

To overcome those problems some researchers attempt to generate artificial data which is as similar to the real data as possible. However to generate such dataset a real dataset is needed even if a small one, so this can't be considered as the best solution.

To summarize the above, fraud detection poses some technical and practical problems for data mining; the most significant technical problem is due to limitations or poor quality of the data itself. Another problem is highly imbalanced (skewed) data in fraud detection datasets.

Trying to distinguish a fraudster from a genuine claimant based upon personal and social characteristics alone is, however, problematic. Research categorizing a sample of fraudulent claims has suggested that fraudsters show characteristics that make them for the most part indistinguishable from the genuine claimant [8]. The fact that professional fraudsters typically do not use - genuine identities compounds this difficulty. Consequently, companies are often reluctant to admit the scale of fraud problems to their shareholders and policy-holders. Moreover, increasing consumer awareness means that one badly-handled claim can undo millions of pounds worth of advertising. Their main aim is to distinguish a fraudster from a genuine claimant.

Data mining techniques can detect anomalies between client-supplied data and existing datasets while remaining sensitive to minor mismatches that are likely to generate false positives, and allow the detection of patterns of fraudulent activity (e.g., patterns of repeated claim activity) among complex data sets.

1.4 Research Questions

The main question behind this research is how to design a novel ensemble insurance fraud detection model using different base classification algorithms for detecting fraudulent insurance claims? There are also other important questions behind this research such as:

Q (1): Which set of sampling technique can better solve the problem of imbalance data resulting in better model performance?

Q (2): How to design a model using data mining base classifier algorithms that can predict fraudulent claims?

Q (3): How to choose the best combination of base classification algorithms model to design an ensemble model in hope to enhance the performance of the base classifiers models?

1.5 Research Objectives

1.5.1 General Objective

To design a novel ensemble insurance fraud detection model using classification algorithms for detecting fraudulent insurance claims, in order to provide effective and efficient services for customers and keeping the company stable enough in the market environment.

1.5.2 Specific Objectives

- To solve the problem of ***imbalance*** data by using proposed Partitioning-undersampling techniques using
    - Sampling-with-replacement.
    - Sampling-without-replacement.
- To develop detection models that will help to detect fraudulent insurance claims using base classifiers.
- To build an ensemble model to in hope to enhance the performance of the base classifiers models.

1.6 Methodology

Methodology is considered as a backbone of the thesis because it presents the approaches and techniques that were used for the proposed insurance fraud detection models and also attempts to answer the research questions mentioned (section 1.4).

In this research, WEKA (open source software) was employed to implement most of the technical aspects of the data mining methodology that will be adopted.

The first step in every research is a properly defined problem domain. This was done based on the information obtained from the dataset and overall fraud detection of the insurance company was described.

This research methodology consists of six phases the first one is problem domain identification and the second phase is literature review. The experiment start with phase three under-sampling techniques evaluation, phase four Insurance Fraud detection (IFD) Base-Classifier Models constriction, phase five proposed IFD ensemble models design and final phase six evaluation.

Most of the researches in the insurance fraud detection literature focused on constructing a new model using one of the sampling techniques (undersampling, oversampling and combined technique of both techniques) to solve the problem of imbalance dataset. Most of these techniques results in one subsample with the disadvantage of data lose in case of undersampling and data over-fitting of the model. This research proposed a novel partitioning-undersampling technique. Several ratios of fraud and non-fraud were tested once using sampling with replacement and once sampling without replacement. The results of these different partitioning-undersampling techniques were evaluated.

In the fourth phase IFD Base-Classifier Models were constricted, consists of modelling the data by allowing the algorithm to search automatically for a combination of data that reliably predicts a desired outcome. After the data has been accessed and modified, analysts can use data modelling techniques to construct models that explain patterns in the data.

The use of data mining to gain knowledge discovery regularities in the data and not just detection is common. Gaining knowledge for detecting fraud patterns within the insurance company customers' dataset in particular, is certainly the purpose of this research. Based on the identified goals and the assessment of the available data, the classification technique is adopted on the prepared data to build a insurance fraud detection model using artificial neural network algorithm, support vector machines and decision tree to create model for detecting fraud patterns.

In the fifth phase the proposed IFD ensemble models were designed, after carefully construct the detection models using artificial neural network, support vector machine and decision tree, an ensemble models were build using different combinations of base classifier models. For the ensemble model designing, ensemble combining classifiers were used; Voting, Grading and Stacking.

Final on the sixth phase the problem understanding, data understanding, data pre-processing, and selection of modelling technique, model designing and ensemble model designing undertaken in this research was evaluated thus designing the insurance fraud detection models.

1.7 Thesis Structure and Organization

The rest of this thesis is organized as follows; chapter two (Literature Review), it contains a brief research background about insurance fraud detection. Also contains a brief overview of data mining techniques including classification, clustering, prediction, outlier detection, regression and visualization, concentrating on DM classifiers applied in research artificial neural network, decision tree and support vector machine. This is followed by a brief review of the techniques used for solving imbalance data problem. It describes an exhaustive overview of the existing literature, with a focus on insurance fraud detection for imbalanced data using DM techniques.

Chapter three (Research Methodology); it contains a detailed discussion of data mining methods for insurance fraud detection of imbalanced data used in this research, explaining the methodology used to reach the research objectives. This chapter is considered as a backbone of the thesis.

Chapter Four (Insurance Fraud Detection (IFD) Models Using Base Classifiers); it describes the approaches used to overcome the class imbalance problem in the automobile insurance fraud dataset. The data sampling method was utilized to overcome the class imbalance problem in the dataset. A novel partitioning-undersampling technique was proposed. To guarantee building a successful DM model for classification of fraud and non-fraud insurance claims, practical steps for selecting the best sampling technique were taken. The single classifier models were modelled using the stated base classifiers (DM algorithms) the comparison between these algorithms is presented based on Recall, Precision and Recall-Precision carve to find out the best sampling technique.

Chapter Five (Designing IFD Ensemble Models); in a trial to improve the results of single base classifier models that was explored in the Chapter four, several types of ensemble models were illustrated in this Chapter. Then the proposed IFD models are applied on another dataset for comparison. It present a comparison between the proposed novel IFD models applied on different datasets.

Chapter Six (Conclusion and Future Research); it presents the summary of the thesis, contributions of the research, and identified future research directions.

2 CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction:

The literature review is divided into two main topics. The first topic is Insurance fraud detection with especial interest in automobile insurance fraud detection. The second topic deals with techniques used for solving imbalance dataset problem or shortly imbalance dataset problem. Section 2 contains a review of insurance fraud detection literature. Section 3 contains the data mining classifiers used in this research the justification for using them and some related work. Section 4 contains techniques for solving imbalance dataset. Section 5 contains performance measure for imbalanced classifiers. Section 6 contains automobile insurance fraud detection using imbalance dataset related work.

2.2 Insurance Fraud Detection:

2.2.1 Introduction

Insurance fraud is a significant and costly problem for both policyholders and insurance companies in all sectors of the insurance industry. In recent years, fraud detection has attracted a great deal of concern and attention. The Oxford English Dictionary [9] defines fraud as “wrongful or criminal deception intended to result in financial or personal gain”.

Fraud occurs in a wide variety of forms and is ever changing as new technologies and new economic and social systems provide new opportunities for fraudulent activity. The total extent of business losses due to fraudulent activities is difficult to define. Phua et al. [10] described fraud as leading to the abuse of a profit organization's system without necessarily leading to direct legal consequences. Although there is no universally accepted definition of financial fraud, Wang et al. [11], defined it as “a deliberate act that is contrary to law, rule, or policy with intent to obtain unauthorized financial benefit”. Economically, insurance fraud is becoming an increasingly serious problem.

Insurance fraud detection (IFD) is important for preventing the disturbing results of insurance fraud. IFD involves distinguishing fraudulent claims from genuine claims, thereby disclosing fraudulent behaviour or activities and enabling decision makers to develop appropriate strategies to decrease the impact of fraud.

Data mining has a significant role in IDF, as it is often applied to extract and uncover the hidden truths behind very large quantities of data. Data mining is about finding insights which are statistically reliable, unknown previously, and actionable from data [12]. This data must be available, relevant, adequate, and clean. Also, the data mining problem must be well-defined, cannot be solved by query and reporting tools, and guided by a data mining process model [13].

Bose and Mahapatra [14] defined data mining as a process of identifying interesting patterns in databases that can then be used in decision making. Turban et al. [15] defined data mining as a process that uses statistical, mathematical, artificial intelligence, and machine learning techniques to extract and identify useful information and subsequently gain knowledge from a large database. Frawley et al. stated that the objective of data mining is to obtain useful, non-explicit information from data stored in large repositories [16]. Kou et al. highlighted that an important advantage of data mining is that it can be used to develop a new class of models to identify new attacks before they can be detected by human experts [17]. Phua et al. pointed out that fraud detection has become one of the best established applications of data mining in both industry and government [10]. Various data mining techniques have been applied in IFD, such as artificial neural networks, logistic regression models, Naïve Bayes, support vector machine method and decision trees, among others [10].

The data mining techniques used for insurance fraud detection in published academic papers was identified; those papers were classified according to the used data mining techniques. All published papers on insurance fraud detection (IFD) using data mining technique in the period between 1997 and 2015 were classified. The search phrase used was “insurance fraud detection data mining”, the search was done in the time period between 1997 and 2013 first a published in a review paper, and then the same search was done for years 2013 to 2015.

2.2.2 Data Mining Techniques

The data mining techniques used in insurance fraud detection are classified into six data mining application classes of classification, clustering, prediction, outlier detection, regression, and visualization [18]. A brief description of the six data mining application classes is provided.

Classification:

Classification is defined as the act or process of putting things into groups based on ways that they are alike. In data mining classification is the process of finding a model (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known) [19]. Zhang and Zhou stated that classification and prediction is the process of identifying a set of common features and models that describe and distinguish data classes or concepts [20]. Common classification techniques include artificial neural networks, the naïve Bayes technique, decision trees and support vector machines. Such classification tasks are used in the detection of credit card, healthcare insurance, automobile insurance, and corporate fraud, among other types of fraud, and classification is one of the most common learning models in the application of data mining in IFD.

Clustering:

Clustering is used to divide objects into conceptually meaningful groups (clusters), with the objects in a group being similar to one another but very dissimilar to the objects in other groups. Clustering is also known as data segmentation or partitioning and is regarded as an alternative of unsupervised classification [19]. According to Yue et al., "clustering analysis concerns the problem of decomposing or partitioning a data set (usually multivariate) into groups so that the points in one group are similar to each other and are as different as possible from the points in other groups" [21]. Further, Zhang and Zhou argue that each cluster is a collection of data objects which are similar to one another within the same cluster but dissimilar to those in other clusters [20]. The most common clustering techniques are the K-nearest neighbour, the Naïve Bayes technique and self-organizing map techniques.

Prediction:

Prediction is a statement about the way things will happen in the future based on experience or knowledge. P*rediction* is a statement that some outcome is expected. Prediction estimates numeric and ordered future values based on the patterns of a data set [22]. Han et al. noted that for prediction, the attribute for which the values are being predicted are continuous-valued (ordered) rather than categorical (discrete-valued and unordered) [19]. This attribute can be referred to simply as the predicted attribute. Neural networks and logistic model prediction are the most commonly used prediction techniques.

Outlier detection:

Often there exist data objects that do not comply with the general behaviour or model of the data. Such data objects, which are grossly different from or inconsistent with the remaining set of data, are called outliers. Many data mining algorithms try to minimize the influence of outliers or eliminate them all together. This, however, could result in the loss of important hidden information. In other words, the outliers may be of particular interest, such as in the case of fraud detection, where outliers may indicate fraudulent activity. Thus, outlier detection and analysis is an interesting data mining task, referred to as outlier mining. Outlier detection is employed to measure the “distance” between data objects to detect those objects that are grossly different from or inconsistent with the remaining data set [19], “Data that appear to have different characteristics than the rest of the population are called outliers” [23]. Yamanishi et al. pointed out that the problem of outlier/anomaly detection is one of the most fundamental issues in data mining [24]. A commonly used technique in outlier detection is the discounting learning algorithm.

Regression:

Regression is a defensive reaction to some unaccepted impulses. It is statistical methodology used to reveal the relationship between one or more independent variables and a dependent variable [19]. Many papers have used logistic regression. The regression technique is typically undertaken using such mathematical methods as logistic regression and linear regression, and it is used in the detection of different types of fraud detection.

Visualization:

Data visualization is the creation and study of the visual representation of data, meaning, information that has been abstracted in some graphical form, including attributes or variables for the units of information. Visualization refers to the easily understandable presentation of the complex patterns or relationships uncovered in the data mining process [19]. Eick and Fyock reported that researchers at Bell and AT&T Laboratories have developed the pattern detection capability of the human visual system by building a suite of tools and applications that flexibly encode data using colour, position, size and other visual characteristics [25] .

2.2.3 Distribution of Data Mining Techniques

The Data mining techniques used in the reviewed papers were sorted and counted to show used DM techniques distribution. The insurance fraud has three application areas: automobile insurance (AI), crop insurance (CI) and healthcare insurance (HI). Table (2-1) lists the reviewed papers according to type of insurance fraud detection and the data mining application classes.

Some of the selected applications in the review address more than one IFD problem, and thus these applications were categorized by the dominant problem addressed.

Table 2-1: Sorting the published papers according to DM application class

Type of Fraud Detection	Data Mining Application Class	Number of Papers	Percentage
Automobile insurance	Classification	18	41%
	Prediction	3	7%
	Regression	4	9%
	Total	25	57%
Crop insurance	Regression	2	5%
	Total	2	5%
Health care insurance	Classification	7	16%
	Clustering	4	9%
	Outlier detection	4	9%
	Prediction	1	2%
	Visualization	1	2%
	Total	17	39%
total		44	

Then more categorization is done using data mining algorithmic used (e.g., neural networks). Table (2-2) shows the classification of the 44 papers according to data mining techniques. Table (2-2) illustrates that most of the papers are on automobile insurance fraud (57%) then healthcare insurance fraud (39%). It also shows that classification is the most frequently used data mining application class 57% (41% + 16%) of the total (25 of the 44 papers), and that visualization is the least common, accounting for only 2.0% each (1 out of 44 each). To determine the main algorithms used for IFD, we present a simple analysis of IFD and the data mining techniques used in the reviewed papers in Table (2-3). Table (2-3) shows forty four data mining techniques used for insurance fraud detection. The most often used techniques are logistic models, the Naïve Bayes, Decision tree, support vector machine, and artificial neural network all of which fall into the "classification" class. The majority of these papers are in automobile insurance fraud detection, it is believed that it has to do with the data collection. It very difficult to collect data for insurance fraud detection in general but there is punch-data available for automobile fraud detection.

This review has explored almost all published insurance fraud detection papers in the five online databases that were searched. The search was done using several keywords to search online databases for papers published between 1997 and 2013, latter after publishing the review paper the search was done for years 2014 and 2015.

The following is a review of past work in the area of insurance fraud detection in general. Few recent papers were added to the past papers already reviewed in the review paper [26].

Table 2-2: Sorting the published papers according to DM techniques

Type of fraud detection	Data mining application class	Data mining technique	Reference
Automobile insurance	Classification	Naïve Bayes	[27]
		Artificial neural network, naïve Bayesian, decision trees	[28]
		Logistic model	[29]
		artificial neural networks, support vector machine, K-nearest neighbor, Naïve Bayes, Bayesian belief network, decision trees, Logistic model	[5]
		support vector machine, Genetic programming	[30]
		decision tree , Naive Bayes tree , SVM-RFE (recursive feature elimination), SVM	[31]
		Consolidated Trees, decision trees	[32]
		Naïve Bayesian, Decision Tree	[33]
		Artificial neural network, ensemble artificial neural network	[34]
		Principal component analysis of RIDIT(PRIDIT)	[35]
		Logistic model	[36]
		Logistic model	[37]
		Logistic model	[38]
		Fuzzy logic	[39]
		Bayesian belief network, Logistic model	[40]
		Self-organizing map	[41]
		Fuzzy DEMATEL, Intuitionist fuzzy number, ELECTRE-TRI	[42]
		Gradient Boosting	[43]
	Prediction	Evolutionary algorithms, Cultural algorithms	[44]
		Social network analysis, Iterative Assessment Algorithm (IAA)	[45]
		Logistic model	[46]
	Regression	Probit model	[47]
		Logistic model	[48]
		Probit model	[49]
		Logistic model	[50]
Crop insurance		Yield-switching model	[51]
		Logistic model, probit model	[52]
Health care insurance	Classification	support vector machine, Gaussian (nonlinear), Linear kernels	[53]
		Naïve Bayes (NB), decision tree, Multiple Criteria Linear Programming (MCLP)	[54]
		Polymorphous (M-of-N) logic	[55]
		Self-organizing map	[56]
		Association rule	[57]
		decision tree	[58]
		support vector machine	[59]
	Clustering	SAS EM, CLUTO	[60]
		nonnegative matrix factorization	[61]
		regression analysis, Distances analysis	[62]
		decision trees	[63]
	Outlier detection	distance analysis, density estimation	[64]
		risk	[65]
		R&DB-algorithm, RB-resolution algorithm	[66]
		Discounting learning algorithm	[24]
	prediction	social network analysis, temporal analysis, higher order feature construction	[67]
	visualization	Visualization	[68]

Table 2-3 : Sorting the published papers according to DM techniques

	Data Mining Techniques	Automobile insurance	Crop insurance	Health care insurance	Total
1	Logistic model	9	1		10
2	Decision Tree	5		3	8
3	Naïve Bayes	5		1	6
4	support vector machine	3		2	5
5	Probit model	2	1		3
6	Bayesian belief network	2			2
7	distance analysis			2	2
8	Self-organizing map	1		1	2
9	Social network analysis	1		1	2
10	Artificial neural network	2			2
11	Association rule			1	1
12	CLUTO			1	1
13	Consolidated Trees	1			1
14	Cultural algorithms	1			1
15	density estimation			1	1
16	Discounting learning algorithm			1	1
17	ELECTRE-TRI	1			1
18	Evolutionary algorithms	1			1
19	Fuzzy DEMATEL	1			1
20	Fuzzy logic	1			1
21	Gaussian (nonlinear)			1	1
22	Genetic programming	1			1
23	Gradient Boosting	1			1
24	higher order feature construction			1	1
26	Intuitionistic fuzzy number	1			1
27	Iterative Assessment Algorithm (IAA)	1			1
28	K-nearest neighbor	1			1
29	Linear kernels			1	1
30	Multiple Criteria Linear Programming			1	1
31	nonnegative matrix factorization			1	1
32	Polymorphous (M-of-N) logic			1	1
33	Principal component analysis of RIDIT	1			1
34	R&DB-algorithm			1	1
35	ensemble neural network	1			1
36	RB-resolution algorithm			1	1
37	regression analysis			1	1
38	risk			1	1
39	SAS EM			1	1
40	SVM (recursive feature elimination)	1			1
41	temporal analysis			1	1
42	text mining			1	1
43	Visualization			1	1
44	Yield-switching model		1		1

2.3 Data Mining Classifiers

In this research the base classifiers used are artificial neural network, decision tree and support vector machine. This choice was done randomly from the result of the review of the past papers. The most used classifier Logistic model and Naïve Bayes were not chosen because it would be a repeating for previous work. The classifiers were chosen randomly from the ten most used classifiers to insure good results. Another criteria for choosing those classifiers was their advantages with consideration to their disadvantages; shown in the (table 2-4) [69].

Table 2-4: Advantages and disadvantages of the classifiers.

Methods	Advantage	Disadvantage
Decision Tree	1. There is no need for domain knowledge in the creation of decision tree. 2. It reduces the uncertainty of complex decisions and assigns exact values to outcomes of different actions. 3. It can easily process the data with high dimension. 4. It is easy to analyze. 5. Both numerical and categorical data are handled by Decision tree.	1. It is limited to one output attribute. 2. It produces categorical output. 3. It is a not-fixed classifier i.e. performance of classifier depends upon the type of dataset. 4. A complex decision tree is produced if the type of dataset is numeric.
Support Vector Machine	1. Better Accuracy as compare to other classifier. 2. Easily handle complex nonlinear data points. 3. Over fitting problem is not as much as other methods.	1. Computationally expensive. 2. The main problem is the selection of right kernel function. For every dataset different kernel function shows different results. 3. As compare to other methods training process take more time. 4. SVM was designed to solve the problem of binary class. It solves the problem of multi class by breaking it into pair of two classes such as one-against-one and one-against-all.
Artificial neural network	1. Easily identify complex relationships between dependent and independent variables. 2. Able to handle noisy data.	1. Local minima. 2. Over-fitting. 3. The processing of ANN network is difficult to interpret and require high processing time if there are large artificial neural networks.

2.3.1 Artificial Neural Network

It is an algorithm for classification that uses gradient descent method and based on biological nervous system having multiple interrelated processing elements known as neurons, functioning in unity to solve specific problem. Rules are extracted from the trained Artificial Neural Network (ANN) help to improve interoperability of the learned network [70].

To solve a particular problem ANN used neurons, which are organized processing elements. Artificial neural network is used for classification and pattern recognition [71].

An ANN is adaptive in nature because it changes its structure and adjusts its weight in order to minimize the error. Adjustment of weight is based on the information that flows internally and externally through network during learning phase. In ANN multiclass, problem may be addressed by using multilayer feed forward technique, in which Neurons have been employed in the output layer rather using one neuron.

Wang et al. used neural network to build fraud detection models then Self-organizing map to evaluate this models [56].

Derrig et al. used several state-of-art binary classification techniques were experimentally evaluated in the context of expert automobile insurance claim fraud detection. The predictive power of logistic regression, C4.5 decision tree, k nearest nieghbor, Bayesian learning multilayer perceptron neural network, least square support vector machine, naive Bayes and tree-augmented naive Bayes classification is contrasted. Comparison between those models shows that there is no large difference between those classifiers [5].

Dedene et al. extended on his previous work [5] by exploring the explicative capabilities of neural network classifiers with automatic relevance determination weight regularization. The automatic relevance determination objective function scheme provides us with a way to determine which inputs are most informative to the trained artificial neural network model. The artificial neural network findings were compared to the predictor importance evaluation from popular logistic regression and decision tree classifiers [28].

Wang et al. proposed a random rough subspace based artificial neural network ensemble method for insurance fraud detection. In this method, rough set reduction is firstly employed to generate a set of reductions, which can keep the consistency of data information. Secondly, the reductions are randomly selected to construct a subset of reductions. Thirdly, each of the selected reductions is used to train an artificial neural network classifier based on the insurance data. Finally, the trained artificial neural network classifiers are combined using ensemble strategies. For validation, a real automobile insurance case is used to test the effectiveness and efficiency of our proposed method with two popular evaluation criteria including the percentage correctly classified (PCC) and the receive operating characteristic (ROC) curve [34].

Gepp and Bhattacharya provided a comparative analysis of prediction performance of data mining techniques, ANNs, decision trees, fraud detection, logit model, survival analysis using real-life automotive insurance fraud data [72].

Phillips summarized the use of artificial neural networks as a business tool for analysis of insurance data in order to identify patterns. Using of a artificial neural network to classify vehicles are total-lost or repairable in the UK insurance market, the expansion point of the work carried out was to expand the artificial neural network use into the detection of fraud within the same market [73].

Kathiresan and Faseela used Spectral analysis, Support vector machine, Multilayer neural network (MLP) and compared [74].

2.3.2 Decision Tree

Decision Tree (DT) is similar to the flowchart in which every non-leaf nodes denotes a test on a particular attribute and every branch denotes an outcome of that test and every leaf node have a class label. The node at the top most labels in the tree is called root node. For example a financial institution decision tree, which is used to decide that a person must grant the loan or not. Building a decision for any problem doesn't need any type of domain knowledge. Decision Trees is a classifier that use tree-like graph [69].

Derrig et al. used several state-of-art binary classification techniques including decision tree were experimentally evaluated in the context of expert automobile insurance claim fraud detection [5].

Muguerza et al. presented an analysis of the behaviour of Consolidated Trees, CT (classification trees induced from multiple subsamples but without loss of explaining capacity). Then analysed how CT trees behave when used to solve a fraud detection problem in a car insurance company. In the results presented in the paper CT and C4.5 trees have been compared showing that CT gives better results [32].

Kou et al. used health insurance data with some known suspicious and normal policies. These known policies are used to train the predictive models. Missing values and irrelevant variables are removed before building predictive models. Three predictive models: Naïve Bayes (NB), decision tree, and Multiple Criteria Linear Programming (MCLP), are trained using the claim data [54].

Bhowmik et al studied the existing fraud detection systems. To predict and present fraud we used Naïve Bayesian classifier and Decision Tree-Based algorithms [33].

Ravi et al. proposed approach comprises of three major phases; feature selection using SVM-RFE (recursive feature elimination), then active learning for synthetic data generation and at last rule generation using decision tree (DT) and Naive Bayes tree (NB Tree) [31].

Park et al. used a scoring model that measures the abusiveness of healthcare providers in the medical services claims submitted to a payer. The model is composed of two parts: scoring and segmentation. Through scoring, the model quantifies the degree of abusiveness, and based on these scores, the segmentation is used to group the problematic providers with similar profiles in the billing pattern. The segmentation model, based on decision trees, can be applied with greater efficiency to the resulting scores [63].

Guelman et al., used Gradient Boosting trees is an iterative algorithm that combines simple parameterized functions with ‘‘poor’’ performance (high prediction error) to produce a highly accurate prediction rule. This paper presented the theory of Gradient Boosting trees and its application to the problem of predicting auto ‘‘at-fault’’ accident loss cost using data from a major Canadian insurer [43].

Lin et al., used data to develop the decision tree model taking advantage of data-mining technology to design models and find out cases requiring for manual inspection so as to save time and manpower [58].

Gepp and Bhattacharya, compared several models including decision trees [72].

Kim used a personal credit rating prediction model based on the smart ubiquitous data mining (UDM) and benchmarks their performance against other models which employ logistic regression (LR), Bayesian style frequency matrix (BFM), multilayer perceptron (MLP), classification tree methods (C5.0), and neural network rule extraction (NR) algorithms. UDM is working with 3 steps: (1) constructing different models using datasets Bayesian style frequency matrix (BFM), decision trees (C5.0), and neural network rule extraction (NR) , (2) inducing rules from these models, and (3) consolidating these rules [75].

Joudaki et al. (2015) used Supervised Data Mining Methods for Detecting Health Care Fraud and Abuse the most used data mining are decision tree, support vector and neural network [76].

2.3.3 Support vector machine

Support vector machine (SVM) is given by Vapnik et al. [77, 78], which is based on statistical learning theory. SVMs were initially developed for binary classification but it could be efficiently extended for multiclass problems. The support vector machine classifier creates a hyper plane or multiple hyper planes in high dimensional space that is useful for classification, regression and other efficient tasks [69] .

Derrig et al. evaluated the use of several classifiers showing that support vector machine gave one of the best results [5].

Ravi proposed novel hybrid approach for under-sampling the majority class , several classifiers were used and the results shows that SVM gave second best results [30].

Ravi et al. presented an extended (modified active learning-based approach) (ALBA) to extract rules from the trained SVM model by making use of key concepts of the SVM: the support vectors. ALBA for rule extraction from SVM to mine two unbalance medium scale data mining problems such as; churn prediction in bank credit cards customers and Insurance fraud detection. Feature selection in employed in first phase of the proposed approach and active learning is carried out during second phase, where extra data is generated near support vectors and the predictions are obtained using the trained SVM model. Finally, rule induction algorithms viz., NB Tree and DT are employed separately to generate rules [31].

Kirlidog and Asuk used Data mining tools and techniques to detect fraud in large sets of insurance claim data. Data mining methods such as anomaly detection, clustering and classification can successfully detect anomalies or outliers in large sets of data. Based on a few cases that are known or suspected to be fraudulent, the anomaly detection technique calculates the likelihood or probability of each record to be fraudulent by analyzing the past insurance claims. The analysts can then have a closer investigation for the cases that have been marked by data mining software [53].

Pepper et al. examined the design and efficiency of (Quash), an automated medical bill processing system capable of bill routing and abuse detection, that uses support vector machine for its modelling. Quash is designed to be used in combination with human experts. The primary contribution of Quash is to provide a real world speed up help for medical fraud detection experts in their work [59].

Kirlidog and Asuk used data mining methods, such as anomaly detection, clustering and classification, to detect anomalies or outliers in large sets of data. Once the anomalous claims are detected, several analyses must be made on them in order to conduct a thorough investigation. The main task in these analyses are to narrow the target for detecting frauds. These investigations also reveal some new and unknown patterns [53].

Kathiresan et al. reviewed various approaches for detecting fraudulent behaviour in health insurance claim in several recent papers. Then compared between, the Multi Layer Neural Networks (MLP), Support Vector Machine (SVM) and Spectral Analysis (SA) techniques, was done [74].

Rashidian et al. provided a review paper on health care fraud detection and the most used data mining techniques were neural network, decision tree and support vector machine [76].

2.4 Imbalanced Dataset Problem

When applying conventional machine learning algorithms the mining of imbalanced datasets can result in models that are strongly predictive for the larger class, while delivering performance which is poorly predictive for the minority class [79] [60]. This is due to the fact that conventional classifiers will attempt to return the most correct predictions based upon the entire dataset, this results in them categorizing all data as belonging to the larger class. This class is usually the class, which is of least interest to the data-mining problem. In the case of insurance fraud data-mining, the majority class is the class where no fraud has occurred and the minority class being fraud. When minority class is very small a learner can deliver very high predictive accuracy even though it has classified none of the minority class correctly. Taking the area of interest of this thesis (fraud) the minority class would be 'possibly fraudulent claims' while the majority class would be 'non fraudulent'.

Several classification techniques have been proposed and applied in the literature for imbalanced classification problems. These techniques can be classified in two major categories: resampling then classification and cost-sensitive learning [60]. However, using ensemble algorithms; which build an ingratiation of classifiers; can solve imbalance dataset problem. Typically, these algorithms are ensemble of cost-sensitive learning or resampling-then-classification algorithms. The objective of using ensemble learning is to improve the classification performance.

2.4.1 Resampling

The class imbalance problem can also lead to other problems due to the small size of the minority class. These are often caused by within class differences.

When considering the class imbalance problem were the minority class is very small, the ability of a learner to discover patterns in the minority class data cannot be relied upon. However as the minority class size increases the learner becomes better at being able to recognize minority class samples from the majority [80]. Sampling is the most popular means for overcoming this problem. Sampling is used as a means of altering the distribution of the minority class so that it is not under represented when training the learner.

There are three basic approaches to overcome the class imbalance problem. These are oversampling of the minority class, under sampling of the majority class or the use of a hybrid approach based on both [72]. Ling and Li, have suggested the use of a combined approach which uses both the oversampling of the class with a smaller number of instances and under sampling of the class with the larger number of instances [81]. These different sampling methods can be explained using the following example. In the case of under sampling, take for example a case where there are 1100 samples in the training set, distributed as follows; the minority class consists of 100 samples and the majority class consists of a 1000 samples.

When carrying out under sampling the majority class would be reduced in number through the use of a sampling procedure to 100. In the case of minority oversampling a sampling procedure would be used to adjust the sample size to a sample size of 1000. When a hybrid approach is used, the majority class would be under sampled to a sample size of 550 and the minority class would be oversampled to a sample size of 550. When under sampling is used to overcome the problem of class imbalance, the number of majority class examples is reduced until the number of majority samples equals the number of minority samples.

When using this solution to the class imbalance problem certain problems may arise from removing such a large number of the majority class. Due to a number of potentially useful samples from the majority class may be discarded; the majority class is made up of a number of smaller groups, which are often referred to as partitions. These partitions may themselves be imbalanced, which is termed 'within class imbalance' or these partitions may be balanced by using a complicated undersampling approach. This further complicates the data mining exercise and a more complicated under sampling approach must be used which carries out under sampling a number of times so as to produce a number of classifiers [82]. This approach is similar to ensemble methods such as boosting.

Alternatively a focussed undersampling method may be used which provides an educated selection of the majority class. Undersampling does offer a number of benefits to oversampling. The main one being that it results in a smaller training set as compared to oversampling, thus resulting in shorter training times. This smaller training set will also allow the use of more complex learners, like neural networks. Oversampling the minority class creates a number of replicates of the minority class until the number of the minority class examples equals or nearly equals the number of examples in the majority class. This oversampling redistribution functions is similarly to under sampling as it rebalances the distribution of the minority class when compared to the majority class, thus allowing the learner sufficient samples of the minority class to train the model to recognize the minority class; but as mentioned results in very large dataset.

Oversampling can result in a number of other problems, these include over-fitting of a model especially in the cases of noisy data. Also oversampling does not result in more information being included in the training set, which can cause the production of overly complex models. Take for example the case of decision trees which when developed with oversampling, can result in overly complex decision trees. This is due to the larger number of the minority examples causing the learner to not prune a number of sections of the model, which encompass a very small number of training examples [4].

Oversampling is amongst the most popular methods for overcoming the class imbalance problem. This is due its ease of implementation. Oversampling is most commonly implemented using random sampling of the minority class with replacement of the minority class. This is known as random oversampling of the minority class with replacement, or more simply it is known by the acronym ROS. The samples must be replaced otherwise the pot of minority class samples would be exhausted before rebalancing of the minority class with respect to the majority class had taken place.

Japkowicz has carried out studies on methods suitable for oversampling the minority class [83] . Two methods of sampling were considered as follows:

1. Random sampling where the minority class was randomly re-sampled with replacement until the number of objects in the minority class equalled the number in the majority class.

2. Focussed re-sampling was also considered, this involved selective re-sampling of minority class objects, which exist close to a border between the minority and majority class.

Japkowicz founded that either method were useful in overcoming the class imbalance problem however the use of the more advanced focussed sampling method resulted in any significant increase in performance [83].

While undersampling the data has the advantage of not using replicates of pre-existing data, the amount of data available to create a model may be very small unless the dataset is very large [83]. However it does have the advantage of creating a model on data, which has been observed.

When carrying out a comparison of oversampling and under sampling along with recognition based methods, Japkowicz found that recognition based methods were inferior to both undersampling and oversampling [84]. More exotic forms of undersampling have also been investigated. In these methods the undersampling is focussed on the members of the majority class which are most useful in training the model to recognize the majority class. Kubat et al. investigated this focussed undersampling approach. In their study they used a discriminatory undersampling technique [85]. This is applied to the majority class and keeps the number of samples in the majority class the same as the number of samples in the minority class.

2.4.2 Cost-Sensitive Learning

Cost-sensitive learning algorithms assign weights to data examples based on their importance. They are equivalent to resampling technique and combine both undersampling and oversampling. Many popular classification algorithms can be adapted under this framework [86].

In the case of classification data mining projects the cost of misclassification of a particular class can be extremely expensive. In financial fraud this can be calculated in terms of monetary losses or in the domain of medical diagnosis, the non-diagnosis of dangerous diseases which may lead to death. Normal machine learning algorithms are built to deliver maximum accuracy. When these classifiers are used in a binary class imbalanced classification problem, they can result in building a model, which classifies all samples as members of the majority class. One way of addressing this misclassification problem is to train a cost sensitive misclassification learner where the cost misclassifying the minority class is far greater than that of the majority class [4]. There are two general methods for doing this:

1. Use of cost-sensitive learners.
2. Design a general approaches for turning normal classifiers into cost sensitive ones.

An example of the latter is Meta-cost. Domingos stated that Meta-cost can be seen as a wrapper which is used in conjunction with a classifier [87]. A cost matrix can be seen as a method for capturing the cost of misclassification of examples of one class as belonging to another. C(i,j) can be described as the cost of predicting a record that is actually from class i as belonging to class j. Therefore using this notation the cost of generating a false negative error is denoted by C(+,-), while C(-,+) is the cost of generating a false positive. A negative value in the cost matrix represents a reward for making correct classifications. The overall cost of a model is thereby given by:

$$\mathbf{Ct(M) = TPXC(+;+) + FPXC(-;+) + FNXC(+;-) + TNXC(-;-)} \qquad \mathbf{2.1}$$

When a cost sensitive approach is used, the above cost matrix is utilized to generate a model with the lowest cost. This is the approach the Meta-cost learner takes. In explaining the Meta-Cost algorithm it is important to note that the conditional risk R (i | x) is equal to the anticipated price of predicting that example x belongs to class i [87]. As the Bayes optimal prediction for example x is the class I that reduces as much as possible the conditional risk. Therefore for C(i,j) which is the cost of predicting that an example is of class i when actually it belongs to class j and the probability of x belonging to class j. The Bayes best prediction for example x is the class that keeps to a minimum the conditional risk equation, this is shown below.

$$\mathbf{R(j|i) = \textstyle\sum_j P(j|x)\, C(i,j)} \qquad \mathbf{2.2}$$

This results in the cause of a division of the example space x into j number of divisions in space so that for class j; class j is the lowest cost prediction in the j division in space. The aim then for cost-sensitive learning is to locate the borders between these divisions. If the samples used to train the model were now named according to their most accurate class as provided by the cost matrix, a error based learner could be utilized to find the location of these idealized borders of the divisions of the sample space. This is the approach the Meta-cost procedure is built on. However to rename the samples in the training set with their optimal classes, a means of deriving their class probabilities must be found. Meta-cost accomplishes this by training multiple models and then utilizing each class's percentage of the total vote as an approximate of its probability [4].

2.5 Ensemble Learning

The main idea of ensemble methodology is to combine a set of models, each of which solves the same original task, in order to obtain a better composite global model, with more accurate and reliable estimates or decisions than can be obtained from using a single model [88].

2.5.1 Sequential Methodology

In sequential approaches for learning ensembles, there is an interaction between the learning runs. Thus it is possible to take advantage of knowledge generated in previous iterations to guide the learning in the next iterations. We distinguish between two main approaches for sequential learning.

2.5.1.1 Model-guided Instance Selection

In this sequential approach, the classifiers that were constructed in previous iterations are used for manipulating the training set for the following iteration. One can embed this process within the basic learning algorithm. These methods, which are also known as constructive or conservative methods, usually ignore all data instances on which their initial classifier is correct and only learn from misclassified instances.

Boosting:

Boosting is a general method for improving the performance of any learning algorithm. The method works by repeatedly running a weak learner, on various distributed training data. The classifiers produced by the weak learners are then combined into a single composite strong classifier in order to achieve a higher accuracy than the weak learner's classifiers would have had. Schapire introduced the first boosting algorithm in 1990. Freund and Schapire introduced the AdaBoost algorithm [89]. The main idea of this algorithm is to assign a weight in each example in the training set. In the beginning, all weights are equal, but in every round, the weights of all misclassified instances are increased while the weights of correctly classified instances are decreased. As a consequence, the weak learner is forced to focus on the difficult instances of the training set. This procedure provides a series of classifiers that complement one another. The AdaBoost algorithm is described in (figure 2.2). The algorithm assumes that the training set consists of *m* instances, labelled as -1 or +1. The classification of a new instance is made by voting on all classifiers, each having a weight of α_t. Mathematically, it can be written as:

$$\mathbf{H\,(x) = \ sign\left(\textstyle\sum_{t=1}^{T} \alpha_t \,.\, C_t(x)\right)} \qquad \mathbf{2.3}$$

```
Input: I (a weak inducer), T (the number of iterations), S (training set)

Output: $C_t$, $\alpha_t$ ; t= 1, ....,T

        t ← 1
        $D_1(i)$ ← 1/m; i=1, ..., m
        Repeat
            Build Classifier $C_t$ using I and distribution $D_t$
                        $\varepsilon_t \leftarrow \sum_{i:C_t(x_i)\neq y_i} D_t(i)$
            If $\varepsilon_t > 0.5$ then
                T← t ← 1
                exit Loop.
            end if
            $\alpha_t \leftarrow \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
            $D_{t+1}(i) = D_t(i).\,e^{-\alpha_t y_t C_t(x_i)}$
            Normalize $D_{t+1}$ to be a proper distribution.
            t ++
        Until t > T.
```

Figure 2-1: The AdaBoost Algorithm

Boosting seems to improve performances for two main reasons:

1. It generates a final classifier whose error on the training set is small by combining many hypotheses whose error may be large.

2. It produces a combined classifier whose variance is significantly lower than those produced by the weak learner.

On the other hand, boosting sometimes leads to deterioration in generalization performance. The main reason for boosting failure is overfitting.

The objective of boosting is to construct a composite classifier that performs well on the data, but a large number of iterations may create a very complex composite classifier that is significantly less accurate than a single classifier. A possible way to avoid overfitting is by keeping the number of iterations as small as possible.

Another important drawback of boosting is that it is difficult to understand. The resulted ensemble is considered to be less comprehensible since the user is required to capture several classifiers instead of a single classifier.

Uncertainty Sampling:

This method is useful in scenarios where unlabelled data is plentiful and the labelling process is expensive. We can define uncertainty sampling as an iterative process of manual labelling of examples, classifier fitting from those examples, and the use of the classifier to select new examples whose class membership is unclear. A teacher or an expert is asked to label unlabelled instances whose class membership is uncertain [88].

Windowing:

Windowing is performed by using a sub-sampling procedure. The method may be summarized as follows: a random subset of the training instances is selected (a window). The subset is used for training a classifier, which is tested on the remaining training data. If the accuracy of the induced classifier is insufficient, the misclassified test instances are removed from the test set and added to the training set of the next iteration [88].

2.5.1.2 Incremental Batch Learning

In this method the classifier produced in the first iteration is given as “prior knowledge” to the learning algorithm in the following iteration (along with the subsample of that iteration). The learning algorithm uses the current subsample to evaluate the former classifier, and uses the former one for building the next classifier. The classifier constructed at the last iteration is chosen as the final classifier [88].

2.5.2 Concurrent Methodology

In the concurrent ensemble methodology, the original dataset is partitioned into several subsets from which multiple classifiers are induced concurrently. The subsets created from the original training set may be disjoint (mutually exclusive) or overlapping. A combining procedure is then applied in order to produce a single classification for a given instance. Since the method for combining the results of induced classifiers is usually independent of the induction algorithms, it can be used with different inducers at each subset. These concurrent methods aim either at improving the predictive power of classifiers or decreasing the total execution time.

Bagging

The most familiar method that process samples concurrently is bagging. The method aims to improve the accuracy by creating an improved compound classifier, I^*, by joining the various outputs of trained classifiers into a single prediction.

The bagging algorithm is shown bellow. Each classifier is trained on a subsample randomly selected with replacement from the training dataset. Usually each subsample size is equal to the size of the original training dataset.

```
Input: $I$ (an inducer), $T$ (the number of iterations), $S$ (the training set), $N$ (the subsample size).

Output: $C_t$; $t = 1, \dots, T$

    t ← 1
    repeat
        $S_t$ ← Sample $N$ instances from $S$ with replacement.
        Build classifier $C_t$ using $I$ on $S_t$
        t + +
    until $t > T$
```

Figure 2-2: Bagging algorithm.

Since sampling with replacement is used, some of the original instances of S may appear more than once in S_t and some may not be included at all. As a result of this the training subsets S_t are different from each other, but are certainly not independent. To classify a new instance, each classifier returns the class prediction for the unknown instance. The composite bagged classifier, I^*, returns the class that has been predicted most often (voting method). The result is that bagging produces a combined model that often performs better than the single model built from the original single data.

Bagging, like boosting, is a technique for improving the accuracy of a classifier by producing different classifiers and combining multiple models. They both use a kind of voting for classification in order to combine the outputs of the different classifiers of the same type. In boosting, unlike bagging, each classifier is influenced by the performance of those built before, so the new classifier tries to pay more attention to errors that were made in the previous ones and to their performances. In bagging, each instance is chosen with equal probability, while in boosting, instances are chosen with probability proportional to their weight.

Cross-Validated Committees:

This procedure creates k classifiers by partitioning the training set into k-equal-sized sets and in turn, training on all but the i^{th} set.

2.5.3 Combining Classifiers

The way of combining the classifiers may be divided into two main groups: simple multiple classifier combinations and meta-combiners. The simple combining methods are best suited for problems where the individual classifiers perform the same task and have comparable success. However, such combiners are more vulnerable to outliers and to unevenly performing classifiers. On the other hand, the meta-combiners are theoretically more powerful but are susceptible to all the problems associated with the added learning (such as over-fitting, long training time).

2.5.3.1 Simple Combining Methods

There are several Combining methods that are considered simple e.g. Averaging, Uniform Voting and Order Statistics. All of them have simple function equations.

Averaging:

Averaging is the most popular and basic combination method for numeric outputs. Suppose we are given a set of T individual learners $(h_1, \ldots, h_T)$ and the output of h_i for the instance x is $h_i(x) \in \mathbb{R}$, our task is to combine h_i's to attain the final classification [90].

Simple Averaging

Simple averaging obtains the combined output by averaging the outputs of individual learners directly [91]. Specifically, simple averaging gives the combined output $H(x)$ as

$$\mathbf{H(x)} = \frac{1}{T}\sum_{i=1}^{T} \mathbf{h_i(x)} \qquad \mathbf{2.4}$$

Suppose the underlying true function we try to learn is $f(x)$, and x is sampled according to a distribution $p(x)$. The output of each learner can be written as the true value plus an error item, i.e,

$$\mathbf{h_i(x) = f(x) + \varepsilon_i(x), \quad i = 1, \ldots, T} \qquad \mathbf{2.5}$$

Owing to its simplicity and effectiveness, simple averaging is among the most popularly used methods and represents the first choice in many real applications [90] [92].

Voting:

In this combining schema, each classifier has the same weight. A classification of an unlabelled instance is performed according to the class that obtains the highest number of votes. Mathematically it can be written as:

$$Class(x) = \underset{c_i \in dom(y)}{\operatorname{argmax}} \sum_{\forall k c_i = \underset{c_j \in dom(y)}{\operatorname{argmax}} \hat{P}_{M_k}(y=c_j|x)} 1 \qquad 2.6$$

Where M_k denotes classifier k and $\widehat{P_{M_k}}(y = c \mid x)$ denotes the probability of y obtaining the value c given an instance x.

2.5.3.2 Meta Combining Methods

Meta-learning means learning from the classifiers produced by the inducers and from the classifications of these classifiers on training data [88].

Stacking:

Stacking is a technique whose purpose is to achieve the highest generalization accuracy. This method tries to distinguish between reliable classifiers and not reliable. It is used to combine models built by different inducers. The idea is to create a new dataset containing a tuple for each tuple in the original dataset. However, instead of using the original input attributes, it uses the predicted classification of the classifiers as the input attributes. The target attributes remains as in the original training set.

Test instance is first classified by each of the base classifiers. These classifications are fed into a meta-level training set from which a meta-classifier is produced. This classifier combines the different predictions into a final one. It is recommended that the original dataset will be partitioned into two subsets.

The first subset is reserved to form the meta-dataset and the second subset is used to build the base-level classifiers. Consequently the meta-classifier predications reflect the true performance of base-level learning algorithms. Stacking performances could be improved by using output probabilities for every class label from the base-level classifiers. In such cases, the number of input attributes in the meta-dataset is multiplied by the number of classes [88].

Grading

This technique uses “graded” classifications as metalevel classes [93]. The term graded is used in the sense of classifications that have been marked as correct or incorrect. The method transforms the classification made by the *k* different classifiers into *k* training sets by using the instances *k* times and attaching them to a new binary class in each occurrence. This class indicates whether the *k*–th classifier yielded a correct or incorrect classification, compared to the real class of the instance.

For each base classifier, one meta-classifier is learned whose task is to classify when the base classifier will misclassify. At classification time, each base classifier classifies the unlabeled instance. The final classification is derived from the classifications of those base classifiers that are classified to be correct by the meta-classification schemes. In case several base classifiers with different classification results are classified as correct, voting, or a combination considering the confidence estimates of the base classifiers, is performed. Grading may be considered as a generalization of cross-validation selection [94], which divides the training data into *k* subsets, builds *k-1* classifiers by dropping one subset at a time and then using it to find a misclassification rate. Finally, the procedure simply chooses the classifier corresponding to the subset with the smallest misclassification. Grading make this decision separately for each and every instance; by using only those classifiers that are predicted to classify that instance correctly. The main difference between grading and combiners (or stacking), are that the former does not change the instance attributes by replacing them with class predictions or class probabilities (or adding them to it). Instead it modifies the class values. Furthermore, in grading several sets of meta-data are created, one for each base classifier.

Several meta-level classifiers are learned from those sets. The main difference between grading and arbiters is that arbiters use information about the disagreements of classifiers for selecting a training set, while grading uses disagreement with the target function to produce a new training set.

2.6 Imbalance Classifiers Performance Measures

Classification performance measures can be obtained, directly or indirectly, from the confusion matrix. For a classification problem with k classes, the confusion matrix is a square matrix $C \in R^{kXk}$, with each of its entries C_{ij} , denoting the percentage of the samples that belong to the class i and classified to the class j. For the special case of binary classification (positive and negative), the confusion matrix is as follows:

Table 2-5: Confusion matrix for binary classification

	Predicted Fraud	Predicted Non-Fraud
Actual Fraud	True Positive (TP)	False Negative (FN)
Actual Non-Fraud	False Positive (FP)	True Negative (TN)

In this matrix, diagonal elements represent accurately classified examples and the off-diagonal elements the misclassified data for each class.

A typical performance measure for classification is the so-called accuracy, which is calculated as the correctly classified samples over the total number of training samples. However, for imbalanced classification problems this might not be a good performance indicator, since the majority class dominates the behaviour of this metric. More specifically, naive decision rules can yield high classification accuracy. Alternatively, recall and precision can be used. They are defined as

$$\textbf{Recall} = \frac{\textbf{TP}}{\textbf{TP+FN}} \qquad \textbf{2.7}$$

$$\textbf{Precision} = \frac{\textbf{TP}}{\textbf{TN+FP}} \qquad \textbf{2.8}$$

Still, precision is manipulated by the majority (negative) class. However, the recall is not and therefore, it is a more appropriate measure for this purpose. The space spanned by recall and precision is termed recall (x-axis)/Precision (y-axis) curve (PRC area). The PRC area provides a good visual representation.

There are other metrics used in the literature, including specificity and sensitivity or hit rate, which is the ratio of true positive to the sum of true positive and false positive and lift which is highly related to accuracy, but it is well used in marketing practice.

In this research, recall, precision and PRC are used as performance measures.

2.7 Automobile Insurance Fraud detection Using Class Imbalanced Dataset

In this section special concentration is given to published papers in the area of automobile insurance fraud detection (AIFD), which used imbalanced dataset. The main problems facing IFD are imbalanced data and the choosing of data mining classifiers that give the best results. Analysing AIFD past papers, mention in previous review paper [26] and few recent papers, reviewing the methods used to treat the imbalance data problem, DM classifier used and the evaluation methods used. The aim of analysing the DM technique and imbalance dataset problem solving techniques is to show the unique of the proposed methods in this research.

In the following the techniques used in those papers, to solve the imbalance dataset problem are mentioned. Sternberg and Reynolds solved the problem by search manually for the features that cause type 1error (false positive) and type 2 error (false negative), and use these features to design the model [44]. Brockett et al., and Tennyson and Salsas Forn; improved the method a little by sorting the data into categories by an expert, this way the whole dataset is used [41], [46]. Caudill et al. and Artís et al. in two papers, used an oversampling of fraud claims in order to obtain a good representation for this group [29] [37] [38]. Pérez et al. also used oversampling of the fraud claims but testing with different percentage [32]. Other technique used to solve the imbalance data problem is random sampling with all its types. Belhadji et al. used simple random sampling [47]. Pinquet et al. divide the dataset randomly into two Holdout sample (Random auditing) and Working sample (Usual auditing strategy) [49]. Derrig et al. and Farquad et al. randomly resample the dataset using a stratified sampling using blocked ten-fold cross-validation [5] [31].

Another method used in few papers is partitioning the dataset into several subsamples Viaene et al. partitioning-sample, repeated 100 times, each time using a different randomization selection of the data [27]. Then Viaene et al. improved the above method by resampling the dataset by randomly partitioning the data into k disjoint sets of approximately equal size, and then use k fold cross validation [28]. Xu et al. generated multiple training subsets in terms of the reductions produced by rough set reduction technique [34]. Vasu and Ravi, proposed a hybrid undersampling approach that employs k-reverse nearest neighbour (kRNN) method to detect the outliers from majority class then using K-means clustering to further reduce the influence of the majority class [30]. Sundarkumar and Ravi further improved the proposed sampling method by including One-class support vector machine to reduce the majority class even more [95].

The search for the papers relevant to area of study of this research already shown in previously published paper [26], was update and few recent papers were added. Those papers were reviewed for the data mining algorithms used.

Sternberg and Reynolds, designed fraud detection expert system using the Cultural Algorithms (CA) that provides self-adaptive capabilities, which can generate the information necessary for the expert system to respond dynamically and provide an automated response to environmental changes [44]. Brockett et al. apply Kolionen's used self-organizing feature map to classify automobile bodily injury claims by degree of fraud suspicion, neural network and a back propagation were used to investigate the validity of feature map approach and showed that it performed better than previous methods [41]. Several researches used logit model [29], [37], [46], [5], [38], [36]. Neural network was used by [5] [28]. Xu et al. (2011) used Neural network classifier then improved the performance of the classifier by designing an ensemble neural network [34]. Support vector machine is used in several papers [31] [5] [95] [30] and decision tree is also used in a lot of papers [5] [32] [33] [30] [31] [95]. Belhadji et al. and Pinquet et al. used Probit model to design expert system, then improved it more by using a Probit model, a two equation model for audit and fraud (a Bivariate Probit model with censoring) was estimated on a sample of suspicious claims for which the experts were left to take the audit decision. Results were rather close to those obtained with a random auditing strategy, at the expense of some instability with respect to the regression components set [47] [49] .

Brockett et al. used another statistical technique principal component analysis of RIDIT (PRIDIT) [35]. Pathak et al. (2005) developed a fuzzy logic based expert system that can identify and evaluate whether elements of fraud are involved in insurance claims settlement thus reduce the need for human experts [39].

Viaene et al. compared between several data mining classifiers, Neural networks, support vector machine (SVM), K-nearest nieghbor, Naïve Bayes (NB), Bayesian belief network, decision trees and Logistic model (LM), the results show that there is no great difference between the different classifiers, except SVM, LM and NB result were a little better [5]. Viaene et al. used (smoothed) naive Bayes (NB), AdaBoosted naive Bayes (AB), and AdaBoosted weights of evidence

(ABWOE) comparing them on this real-life data set, the boosted weight of evidence algorithm showed comparable (slightly better) discriminatory and ranking ability to (smoothed) naive Bayes with and without boosting, but clearly improved on the calibration of probability estimates [27]. Pérez et al. compared between two decision tree induction algorithms C4.5 and CTC [32] . Bhowmik et al., compared between Naïve Bayesian classification and decision tree-based classification using two decision tree algorithms (C4.5 algorithms and Consolidated Trees) [33]. Farquad et al. used SVM-RFE for feature selection, and for rule generation used decision tree (DT) and Naive Bayes tree (NB Tree) [31]. Vasu and Ravi compared between several classifiers, support vector machine (SVM), logistic regression (LR), multi layer perceptron (MLP), radial basis function network (RBF), group method of data handling (GMDH), genetic programming (GP) and decision tree (J48) [30]. Sundarkumar and Ravi, improved the previous propped undersampling method tested its efficiency by comparing the new results with previous result, again using several classifier algorithms support vector machine (SVM), logistic regression (LR), multi layer perceptron (MLP), radial basis function network (RBF), group method of data handling (GMDH), genetic programming (GP) and decision tree (J48); it was shown that the new proposed method gave better results; DT and SVM produced the best results [95].

Table 2-6 summarizes the above mentioned analysis of the past papers in the area of automobile insurance fraud detection using imbalance dataset. It also shows that the proposed technique for solving imbalanced dataset (partitioning-undersampling) has not been used the literature and also using (ensemble combining classifiers to combine three different base-classifiers) also have not been used.

Table 2-6: Literature review on automobile insurance fraud detection

Author / year	Imbalance data problem Techniques	Classification Techniques	Evaluation Methods
Sternberg and Reynolds (1997) [44]	Manually feature selection by understanding of what causes a type 1 (false positive) or type 2 (false negative) error.	Designed fraud detection expert system using the Cultural algorithms (CA).	TP, FP, TN and FN (confusion matrix).
Brockett et al. (1998) [41]	Manually sort the data into categories by an expert.	Used Self-organizing feature map, neural network and a back propagation were used to investigate the validity of feature map approach.	Accuracy.
Artís et al. (1999) [29]	Oversampling of fraud claims in order to obtain a good representation for this group.	The nested multinomial logit model.	T-test, chi-square, p-value, accuracy.
Belhadji et al. (2000) [47]	Resampling using random sampling.	Probit model.	Probability, rate of detection%, rate of accuracy%.
Viaene et al. (2002) [5]	Resampling using a stratified, blocked ten-fold cross-validation.	Neural networks, support vector machine, K-nearest nieghbor, Naïve Bayes, Bayesian belief network, decision trees, Logistic model.	Two-way ANOVA, Duncan's multiple range tests, mean PCC and mean AUROC.
Brockett et al. (2002) [35]	No specific technique used to solve the imbalance problem.	Principal component analysis of RIDIT (PRIDIT).	Correlation and confusion matrix.
Artís et al. (2002) [37]	Oversampling of fraudulent claims.	Logistic model.	Mean, standard deviation, p-value and confusion matrix.
Tennyson and Salsas-Forn (2002) [46]	Stratify the data manually into categories by an expert.	Logistic model.	Chi-square test and probability.
Viaene et al. (2004) [27]	Resample by split-sampling, repeated 100 times, each time using a different randomization of the data.	Naive Bayes (NB), **AdaBoosted** naive Bayes (AB), and **AdaBoosted** weights of evidence (ABWOE).	Mean PCC (Percentage Correctly Classified), mean AUROC (Area under the ROC), mean L, mean B(Brier inaccuracy), and mean BSc(Brier imprecision).
Viaene et al. (2005) [28]	Resampling using k-Fold cross-validation technique that randomly splits the data into k disjoint sets of approximately equal size, named folds.	MacKay's neural network models, a practical Bayesian learning approach, logistic regression and decision tree classifiers.	Percentage correctly classified (PCC) and the area under the receiver operating characteristic curve (AUROC).

Pérez et al. (2005) [32]	Over-sampling of the minority class.	Decision tree algorithms C4.5 and CTC.	Precision and Recall.
Pathak et al. (2005) [39]	No specific technique used to solve the imbalance problem.	Fuzzy logic.	Membership Functions of Inputs and Outputs Functions, Surface Plot of the Rule Bases.
Caudill et al. (2005) [38]	Over-sampling of fraudulent claims.	Logistic model.	Coefficients and probability.
Viaene et al. (2007) [36]	No specific technique used to solve the imbalance problem.	Logistic model.	Value of cost-sensitive.
Pinquet et al. (2007) [49]	Resampling by dividing the dataset into two Holdout sample and Working sample.	Probit model.	Correlation coefficient.
Vasu and Ravi (2011) [30]	Hybrid under-sampling approach. Detect the outliers from majority class using k-reverse nearest neighbour (kRNN).Using K-means clustering to further reduce the influence of the majority class.	support vector machine (SVM), logistic regression (LR),multi layer perceptron (MLP), radial basis function network (RBF), group method of data handling (GMDH), genetic programming (GP) and decision tree (J48)	Sensitivity, Specificity, Accuracy and under ROC curve.
Xu, Wang et al. (2011) [34]	Resampling using rough set reduction technique into subsamples.	Neural network and ensemble neural network.	Percentage correctly classified (PCC) and the receive operating characteristic (ROC) curve.
Bhowmik (2011) [33]	No specific technique used to solve the imbalance problem.	Naïve Bayesian Classification, Decision Tree (C4.5) and Consolidated tree.	Confusion matrix and ROC graph.
Farquad et al. (2012) [31]	Resampling using a stratified, blocked ten-fold cross-validation	SVM-RFE (recursive feature elimination, decision tree (DT) and Naive Bayes tree (NB Tree).	Sensitivity, Specificity, Accuracy, under ROC curve and t-test.

2.8 Summary

This chapter has two purposes; the first purpose is providing an overview of the existing literature for the research area. This research contains two major areas of interest; insurance fraud detection and solving imbalance dataset problem, highlighting the data mining techniques used for this purpose.

The second purpose is to give brief explanation of the data mining techniques; artificial neural network, decision tree and support vector machine used in this research. Showing the advantages and disadvantages of each then showing the criteria for choosing them. Also brief explanation is giving for the techniques used for solving imbalance dataset problems; resampling, cost-sensitive learning and ensemble used in this research.

Then the performance measures used to evaluate the different models designed in this research, briefly explaining each.

At the end of this chapter a focus is made on published papers in the area of automobile insurance fraud detection (AIFD), which used imbalanced dataset, highlighting the data mining techniques used and techniques used for solving the imbalance dataset, showing that uniqueness of the proposed solution in this research.

3 CHAPTER THREE
RESEARCH METHODOLOGY

3.1 Introduction

This chapter is considered as a backbone of the thesis because it presents the approaches and techniques that were used for the proposed insurance fraud detection models and also attempts to answer the research questions:

Q (1): Which technique of sampling technique can better solve the problem of imbalance data resulting in better model performance?

Q (2): How to design a model using data mining base classifier algorithms that can predict fraudulent claims?

Q (3): How to choose the best combination of base classification algorithms model to design an ensemble model in hope to enhance the performance of the base classifiers models?

This research methodology was divided into six phases. After briefly explaining those phases, some data preparation is explained; such as dataset description and methods used for the data pre-processing. Then the software tools that were used in this research were displayed. Followed by brief illustrating of techniques used for solving imbalance data problem, then a brief discussion of the data mining algorithms include two main types, base-classifiers models and ensemble classification models. Finally the Chapter ends with evaluation measures, which evaluates and measure the performance of the designed models.

3.2 The Research Phases

The phases of this research are illustrated in Figure 3-1.

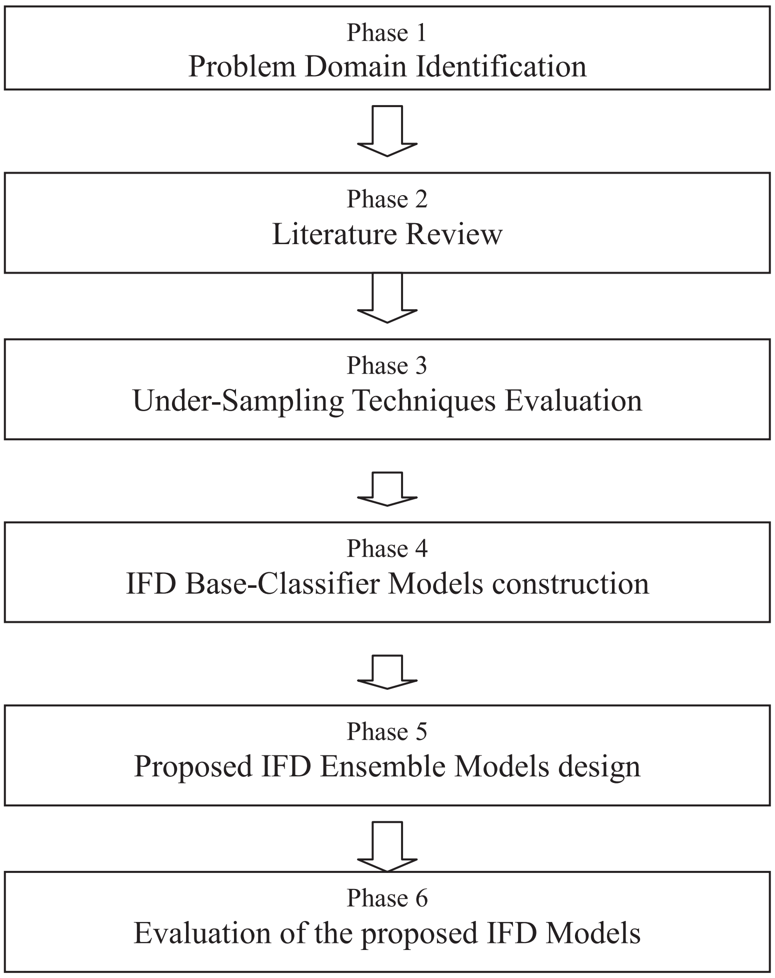

Phase 1
Problem Domain Identification
Phase 2
Literature Review
Phase 3
Under-Sampling Techniques Evaluation
Phase 4
IFD Base-Classifier Models construction
Phase 5
Proposed IFD Ensemble Models design
Phase 6
Evaluation of the proposed IFD Models

Figure 3-1: Research phases.

3.2.1 Phase One: Problem Domain Identification

Fraud detection poses some technical and practical problems for data mining; the most significant technical problem is due to limitations or poor quality of the data itself. Another problem is that datasets used for fraud detection are highly skewed data in fraud detection. Also, finding the best ways that makes prediction more understandable to data analysts is a problem.

Trying to distinguish a fraudster from a genuine claimant based upon personal and social characteristics alone is, however, problematic. Research categorizing a sample of fraudulent claims has suggested that fraudsters show characteristics that make them for the most part indistinguishable from the genuine claimant [8]. The fact that professional fraudsters typically do not use genuine identities compounds this difficulty. Consequently, companies are often reluctant to admit the scale of fraud problems to their shareholders and policy-holders. Moreover, increasing consumer awareness means that one badly-handled claim can undo millions of pounds worth of advertising.

Data mining techniques can detect anomalies between client-supplied data and existing datasets while remaining sensitive to minor mismatches that are likely to generate false positives, and allow the detection of patterns of fraudulent activity (e.g., patterns of repeated claim activity) among complex data sets.

The research activities described in this thesis aim at investigating and proposing different techniques for fraud detection applied to imbalanced data of insurance company. Collecting dataset to be used for fraud detection is the first problem facing any researcher in this field. Companies are often reluctant to admit the scale of fraud problems to their shareholders and policy-holders. Moreover, increasing consumer awareness means that one badly-handled claim can undo millions of pounds worth of advertising. Their main aim is to distinguish a fraudster from a genuine claimant.

3.2.2 Phase Two: Literature Review

In this phase, we analyzed all related scientific papers to give a solid background of the insurance fraud detection methods. This phase contains three stages; the first one is concerned with reviewing insurance fraud in general. Concentrating on data mining techniques and classifiers used for insurance fraud detection and the distribution of the DM techniques and classifier usage in the reviewed past papers. The second stage focuses on methods used for solving the imbalanced data problem, which is a major problem in most insurance fraud detection datasets. The third stage is concerned on data mining techniques and the methods used for solving imbalance data problem in automobile insurance fraud detection.

3.2.3 Phase Three: Sampling Techniques Evaluation

The proposed sampling technique is partitioning-under-sampling technique which is applied on the dataset; producing six groups each containing ten subsamples. An insurance fraud detection decision tree (IFDDT) model is designed for each group of the six groups of subsample. Based on the results of the comparison between all six IFDDT models the best partitioning-undersampling is chosen, which will be used throughout the whole experiment.

3.2.4 Phase Four: IFD Base-Classifier Models construction

Having already constructed IFDDT for the purpose of evaluating the proposed six partitioning-undersampling techniques, now other two IFD models are constructed using support vector machine and artificial neural network. The parameters of both models are changed by trial and error to choose the best and thus designing IFDSVM and IFDANN models.

3.2.5 Phase Five: Proposed IFD Ensemble Models construction

In hope to improve the proposed models ensemble combination methods were used the combine different base-classifiers IFD models. Stacking, grading and voting; the ensemble combining classifiers were used in this research to combine all possible combination of the base-classifiers IFD models to form several ensemble modes.

3.2.6 Phase Six: Evaluation

In the last phase of this research, all the results are analysed to get to conclusions and to make sure that all research question were answered thus pointing out the contributions of this research. Since the dataset was imbalanced the best evaluation and validation measurements are Recall, Precision and the area under the Precision-Recall (PR) curve. Those evaluation and validation measures are explained in more details in Section 3.7 in this Chapter, explaining the justification of their usage.

3.3 Dataset Description

In this research, Ravi [30] provided the insurance dataset, which was taken from Phua et al. [7]. This dataset mainly comprises the information regarding the various automobile insurance claims during the period 1994–1996. The dataset, described in Table 3-1, comprise of 32 variables, with 31 predictor variables and one class variable. It consists of 15,420 samples of which 14,497 are non-fraudulent and 923 are fraudulent, which means there are 94% genuine samples and 6% fraudulent samples.

Table 3-1: Attribute information of the insurance data used

no.	Attribute name Description
1	Month in which accident took place
2	Accident week of month
3	Accident day of week
4	Claim month
5	Claim week of month
6	Claim day of week
7	Year 1994, 1995 and 1996
8	Make Manufacturer of the car (19 companies)
9	Accident area rural or urban
10	Gender male or female
11	Marital status (Single, married, widow and divorced)
12	Age of policy holder
13	Fault Policy holder or third party
14	Policy type of the policy (one to nine)
15	Vehicle category (Sedan, sport or utility)
16	Vehicle price (six categories)
17	Rep. number ID of the person who process the claim (16 ID's)
18	Deductible Amount to be deducted before claim disbursement
19	Driver rating Driving experience (four categories)
20	Days: policy accident Days left in policy when accident happened
21	Days: policy claim Days left in policy when claim was filed
22	Past number of claims
23	Age of vehicle (eight categories)
24	Age of policy holder (nine categories)
25	Policy report filed (Yes or no)
26	Witness presented (Yes or no)
27	Agent type (Internal or external)
28	Number of supplements
29	Address change claim (No of times change of address requested)
30	Number of cars
31	Base policy (BP) (All perils, collision or liability)
32	Class Fraud found (yes or no)

3.4 Data Preprocessing

It is observed that the *age* attribute in the dataset appeared twice in numerical and categorical form as well. Hence, the age attribute with numerical values is removed from the data to reduce the complexity caused by too many unique values it possesses. Further, the attributes *year*, *month*, *week of the month* and *day of week* represent the date of the accident and the attributes *month claimed*, *week of the month claimed* and *day of week claimed* represent the date of the insurance claim. Thus, a new attribute *gap* is derived from seven attributes such as *year*, *month*, *week of the month*, *day of week*, *month claimed*, *week of the month claimed* and *day of week claimed*. The attribute *gap* represents the time difference between the accident occurrence and insurance claim. Thus 24 variables which included some derived variables are selected for further research. Hence, we have 15,420 samples with 24 predictor variables and one class variable. The attributes of the pre-processed dataset are presented in Table 3-2.

Table 3-2: Attribute information of the pre-processed insurance data

no.	Attribute name Description
1	Gap Time difference of accident and insurance claim
2	Make Manufacturer of the car (19 companies)
3	Accident area Rural or urban
4	Gender Male or female
5	Marital status Single, married, widow and divorced
6	Fault Policy holder or third party
7	Policy type of the policy (one to nine)
8	Vehicle category Sedan, sport or utility
9	Vehicle price of the vehicle with six categories
10	Rep. number ID of the person who process the claim (16 ID's)
11	Deductible Amount to be deducted before claim disbursement
12	Driver rating Driving experience with four categories
13	Days: policy accident Days left in policy when accident happened
14	Days: policy claim Days left in policy when claim was filed
15	Past number of claims Past number of claims
16	Age of vehicle Vehicle's age with eight categories
17	Age of policy holder Policy holder's age with nine categories
18	Policy report filed Yes or no
19	Witness presented Yes or no
20	Agent type Internal or external
21	Number of supplements
22	Address change claim No of times change of address requested
23	Number of cars
24	Base policy (BP) All perils, collision or liability
25	Class Fraud found (yes or no)

3.5 Software Tools

WEKA [96] is an open source tool produced by the University of Waikato (New Zealand). The software is written in the Java™ language and contains a GUI for interacting with data files and producing visual results. It also has a general API, so it is easy to insert WEKA in any applications [97]. WEKA is very constant and influential; it is able to implements machine learning algorithms and data mining tasks. Examples of this algorithms are almost all feature selection, classification, clustering and associated rule algorithms [98]. One of weka advantage is the easy of data entry since it accept Attribute-Relation File Format

(ARFF) or CSV; which can be easily converted to excel, making it convenient in many cases. The program has three separate interactive interfaces which another advantage. The primary one is the Explorer, which gives access to all of Weka's facilities using menu selection and form filling. The others are the Knowledge Flow interface, which allows you to design configurations for continues data processing, and the Experimenter, with which the experiment can designed to run automatically, the machine learning algorithms must be specified setting all parameter beforehand and the datasets specified, then all the experiment results are reported automatically [99].

3.6 Data Mining Algorithms

There are six data mining techniques; classification, clustering, prediction, outlier detection, regression and visualization; used for insurance fraud detection [18]; as explained in details in chapter 2. This research uses classification since it the most used and suitable as shown in (table 2-1). There are numerous classification data mining algorithms, this research concentrates on three only; Decision Tree, Support Vector Machine and Artificial Neural Network. These choose was done based on the literary reviewed, the three classifiers were chosen randomly from the best 10 most used classifiers in the literary review shown in (table 2-3). The models designed using those algorithms is called Base-classifiers models. The ensemble models are designed by applying several techniques and meta functions on those base-classifiers models [90].

3.6.1 Base-Classifier Models

3.6.1.1 Decision Tree

A Decision Tree is a flowchart-like tree structure, where each inside node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label. The topmost node in a tree is the root node [100].

Decision trees are used for classification as follows, each tuple, ***X***, which is unlabeled; the attribute values of the tuple are tested against the decision tree. A path is traced from the root to a leaf node, which holds the class predicted label for that tuple. Classification rules can be converted from Decision trees [101].

Major advantage of decision tree classifiers is that its construction does not require any research area deep knowledge or constraint setting, and therefore is appropriate for investigative knowledge discovery. Another advantage of decision trees is that they can handle high dimensional data [102]. Their representation of acquired knowledge in tree form is easy to understand by humans. The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy [103]. However, the final results depend on the data at hand. Decision tree algorithms have been used for classification in many application areas [104].

Decision Tree Induction

During the late 1970s and early 1980s, J. Ross Quinlan, a researcher in machine learning, developed a decision tree algorithm known as ID3 (Iterative Dichotomiser) [105]. Quinlan later presented C4.5 (a successor of ID3), which became a benchmark to which newer supervised learning algorithms are often compared [106]. In 1984, a group of statisticians (L. Breiman, J. Friedman, R. Olshen, and C. Stone) published the book *Classification and Regression Trees*

(CART), which described the generation of binary decision trees [107]. ID3 and CART were invented independently of one another at around the same time, yet follow a similar approach for learning decision trees from training tuple [108].

ID3, C4.5, and CART algorithms construct decision trees uses a top-down recursive divide-and-conquer manner [109]. A top-down approach starts with a training set of tuple and their associated class labels. The training set is recursively partitioned into smaller subsets as the tree is being built. A basic decision tree algorithm is shown in Figure 3.2 [110].

Figure 3-2: Basic algorithm for inducing a decision tree from training tuple.

```
Algorithm: Generate decision tree. Generate a decision tree from the training tuple of data partition D.
Input: Data partition, D, which is a set of training tuple and their associated class labels; attribute list, the set of candidate attributes;
Attribute selection method, a procedure to determine the splitting criterion that "best" partitions the data tuple into individual classes. This criterion consists of a splitting attribute and, possibly, either a split point or splitting subset.
Output: A decision tree.
Method:
(1) create a node N;
(2) if tuples in D are all of the same class, C then
(3) return N as a leaf node labeled with the class C;
(4) if attribute list is empty then
(5) return N as a leaf node labeled with the majority class in D; // majority voting
(6) apply Attribute selection method(D, attribute list) to find the "best" splitting criterion;
(7) label node N with splitting criterion;
(8) if splitting attribute is discrete-valued and multi-way splits allowed then // not restricted to binary trees
(9) attribute-list ← attribute-list ← splitting-attribute; // remove splitting-attribute
(10) for each outcome j of splitting-criterion // partition the tuple and grow sub-trees for each partition
(11) let Dj be the set of data tuple in D satisfying outcome j; // a partition
(12) if Dj is empty then
(13) attach a leaf labelled with the majority class in D to node N;
(14) else attach the node returned by Generate decision tree(Dj, attribute list) to node N;
End for
(15) return N;
```

3.6.1.2 Artificial Neural Network

The field of artificial neural networks was initially started by psychologists and neurobiologists who wanted to develop and test computational analogues of neurons [111]. The neural network is a group of joined input/output neurons in which each bond has a weight related with it. The network learns by adjusting the weights so as to be able to classify the correct class label of the input tuple. Neural network learning is also named connectionist learning because of the connections between neurons [112].

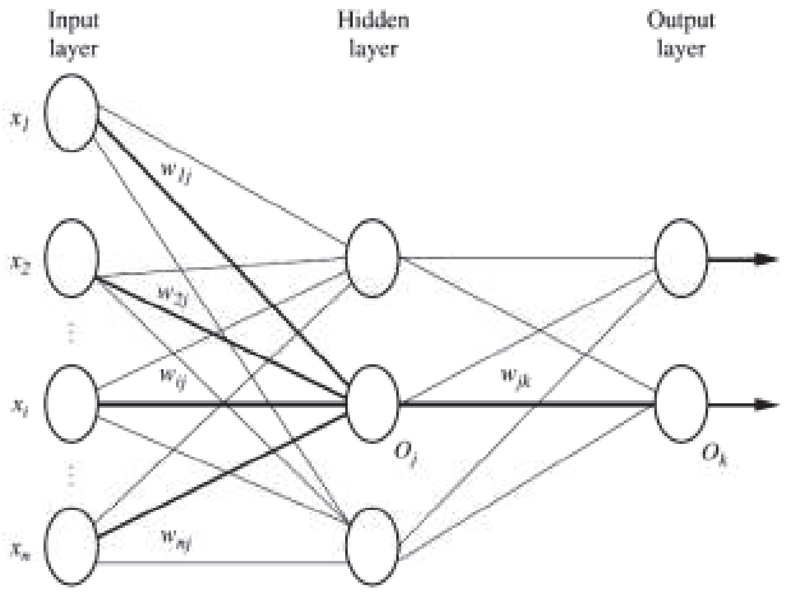
The neural networks have long training times and therefore are more appropriate for applications where this is possible. They need a number of parameters that are usually best determined empirically, such as the network topology or "structure." Neural networks have been criticized for their poor interpretability [113]. It is difficult for humans to understand the symbolic meaning behind the learned weights and of "hidden neurons" in the network. These features initially made neural networks less desirable for data mining. High tolerance of noisy data and ability to classify patterns on which they have not been trained, are two main advantages of neural networks. They are compatible for continuous-valued inputs *and outputs*, unlike most decision tree algorithms [114]. They have been successful on a wide array of real-world data, including handwritten character recognition, pathology and laboratory medicine, and training a computer to pronounce English text. Neural network algorithms are naturally parallel; parallelization techniques can be used to speed up the computation process [115].

Several techniques have been recently developed for the extraction of rules from trained neural networks. These factors add to the usefulness of neural networks for classification and prediction in data mining. There are many different kinds of neural networks and neural network algorithms [116].

A Multilayer Feed-Forward Neural Network

The backpropagation algorithm conducts learning on a *multilayer feed-forward* neural network. Prediction of the class label of tuple is done by iteratively learning of weights [117]. A multilayer feed-forward neural network consists of an *input layer*, one or more *hidden layers*, and an *output layer*. An example of a multilayer feed-forward network is shown in Figure 3.3.

Each layer is made up of neurons. The inputs to the network match the attributes measured for each training tuple. The inputs are fed concurrently into the neurons making up the input layer [118]. These inputs pass through the input layer and are then weighted and fed concurrently to a second layer of "neuron like" neurons, known as a hidden layer. The outputs of the hidden layer neurons can be input to another hidden layer, and so on [119]. The number of hidden layers is subjective, although usually only one is used. The weighted outputs of the last hidden layer are input to neurons making up the output layer, which produce the network's prediction for given tuple [120].



The multilayer neural network (Figure 3.3) has two layers of output neurons. Therefore, we say that it is a two-layer neural network. (The input layer is not counted because it serves only to pass the input values to the next layer.) Similarly, a network containing two hidden layers is called a *three-layer* neural network, and so on. The network is feed-forward in that none of the weights cycles back to an input neuron or to an output neuron of a previous layer [121].

Each output neuron takes, as input, a weighted sum of the outputs from neurons in the previous layer. It applies a nonlinear (activation) function to the weighted input. Multilayer feed-forward neural networks are able to model the class prediction as a nonlinear combination of the inputs. From a statistical point of view, they perform nonlinear regression [122].

Backpropagation

Backpropagation learns by iteratively processing a data set of training tuple, comparing the network's prediction for each tuple with the actual known class label. For each training tuple, the weights are adapted so as to minimize the mean squared error between the predicted value and the actual value. The weight adjusting is done using the "backwards" direction. The backward method starts from the output layer, through each hidden layer down to the first hidden layer (hence the name *backpropagation*). In general the weights will eventually converge, and the learning process stops [108]. The steps of the algorithm are shown in Figure 3-4.

Neural network in this research

The neural network used in this research is a multilayer perceptron with one hidden layer, which is trained using optimization class to minimize the square error. All attributes are standardized. The classifier has a lot of parameters to be adjusted which had been adjusted by trial and error. The ridge parameter is used to determine the penalty on the size of the weights. The number of hidden units has been specified, putting in mind that the larger is the number of units the longer is the training time. The criteria for choosing the number of hidden unit, was the recall of fraud. It is possible to use conjugate gradient descent or BFGS update, again it decided by trial and error. To improve speed an approximate version of logistic function is used as the activation function. The delta value in the backpropagation step is within the user specified tolerance [19].

Figure 3-3: Backpropagation algorithm.

```
Algorithm: (Backpropagation), neural network learning for classification or
        prediction, using the backpropagation algorithm.
Input:
D, a data set consisting of the training tuple and their associated target values;
l, the learning rate;
network, a multilayer feed-forward network.
Output: A trained neural network.
Method:
(1) Initialize all weights and biases in network;
(2) while terminating condition is not satisfied (
(3)     for each training tuple X in D (
(4)      // Propagate the inputs forward:
(5)             for each input layer unit j (
(6)                     $O_j = I_j$; // output of an input unit is its actual input value
(7)             for each hidden or output layer unit j (
(8)             $I_j = \sum_i w_{i\,j} O_i + \theta_j$; //compute the net input of unit j with respect to the
        previous layer, i
(9)                     $O_j = \frac{1}{1\_+e^{-1j}}$ ; ) // compute the output of each unit j
(10)            // Backpropagate the errors:
(11)            for each unit j in the output layer
(12)                    $Err_j = O_j(1 - O_j)(T_j - O_j)$; // compute the error
(13)            for each unit j in the hidden layers, from the last to the first hidden layer
(14)             $Err\,j = O_j(1-O_j)\sum_k Err_k w_{jk}$; // compute the error with respect to the next
        higher layer, k
(15)            for each weight $w_{i\,j}$ in network (
(16)                    $\Delta w_{i\,j} = (l) Err_j O_i$; // weight increment
(17)                    $wi\,j = wi\,j + \Delta wi\,j$; ) // weight update
(18)             for each bias $\theta j$ in network f
(19)                    $\Delta\theta j = (l) Err\,j$; // bias increment
(20)                    $\theta j = \theta j + \Delta\theta j$; ) // bias update
(21)            ))
```

3.6.1.3 Support Vector Machines

Support Vector Machines is a method for the classification of both linear and nonlinear data. It uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane, which separate the tuple of one class from another. With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane. The first paper on support vector machines was presented in 1992 by Vapnik et al. [123], although the groundwork for SVMs has been around since the 1960s (including early work by Vapnik and Chervonenkis on statistical learning theory). Although the SVM training time is general long, it is highly accurate and that is because of its ability to model complex nonlinear decision boundaries. It is much less prone to overfitting than other methods. The support vector found, provide a detailed description of the learned model. SVMs can be used for prediction as well as classification. They have been applied to a number of areas, including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests [124].

The Case When the Data Are Linearly Separable

A two-class problem where the classes are linearly separable is the simplest case of SVM [125]. Let the data set *D* be given as (X_1, y_1), (X_2, y_2)... $(X_{|D|},y_{|D|})$, where X_i is the set of training tuple with associated class labels, y_i. Each y_i can take one of two values, either +1 or -1 (i.e., $y_i \in (+1,-1)$, each one corresponding to one of the two classes. The 2-D data are linearly separable because a straight line can be drawn to separate all of the tuple of class+1 from all of the tuple of class -1 shown in (Figure 3-4). There are an infinite number of separating lines that could be drawn. If the data were 3-D the best separating *plane* should be found. Generalizing to *n* dimensions the best *hyperplane* is found. We will use the term “hyperplane” to refer to the decision boundary that we are seeking, regardless of the number of input attributes [19].

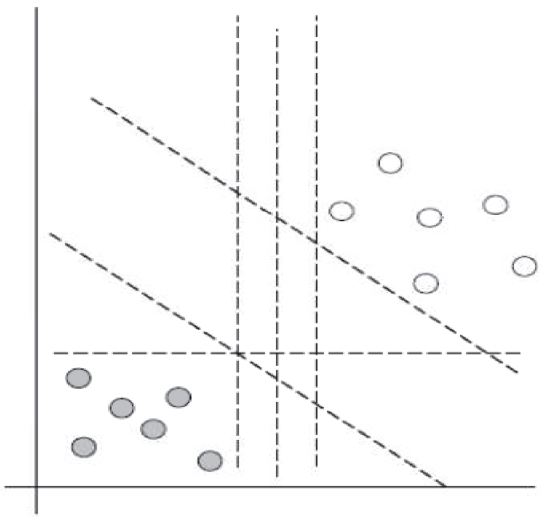


Figure 3-4: 2-D training data which are linearly separable and infinite number of separating hyper-planes

An SVM find the best hyperplane by searching for the maximum marginal hyperplane. Consider (Figure 3-5), which shows two possible separating hyper-planes, however it is expected that the hyperplane with the larger margin to be more accurate at classifying future data tuple than the hyperplane with the smaller margin and that is, the *maximum marginal hyper-plane* (MMH). The associated margin gives the largest separation between classes [126]. Getting to an informal definition of margin, we can say that the shortest distance from a hyperplane to one side of its margin is equal to the shortest distance from the hyperplane to the other side of its margin, where the "sides" of the margin are parallel to the hyperplane. When dealing with the MMH, this distance is, in fact, the shortest distance from the MMH to the closest training tuple of either class [127].

A separating hyperplane can be written as

$$\mathrm{W.X + b = 0} \qquad 3.1$$

Where $\boldsymbol{W}$ is a weight vector, namely, $\boldsymbol{W} = (w_1, w_2, \ldots, w_n)$; n is the number of attributes; and b is a scalar, often referred to as a bias. To aid in visualization, let's consider two input attributes, A_1 and A_2, as in (Figure 3-5). Training tuple are 2-D, e.g., $\boldsymbol{X} = (x_1, x_2)$, where x_1 and x_2 are the values of attributes A_1 and A_2, respectively, for $\boldsymbol{X}$. If we think of b as an additional weight, w_0, we can rewrite the above separating hyperplane as

$$w0+w1x1+w2x2 = 0 \qquad 3.2$$

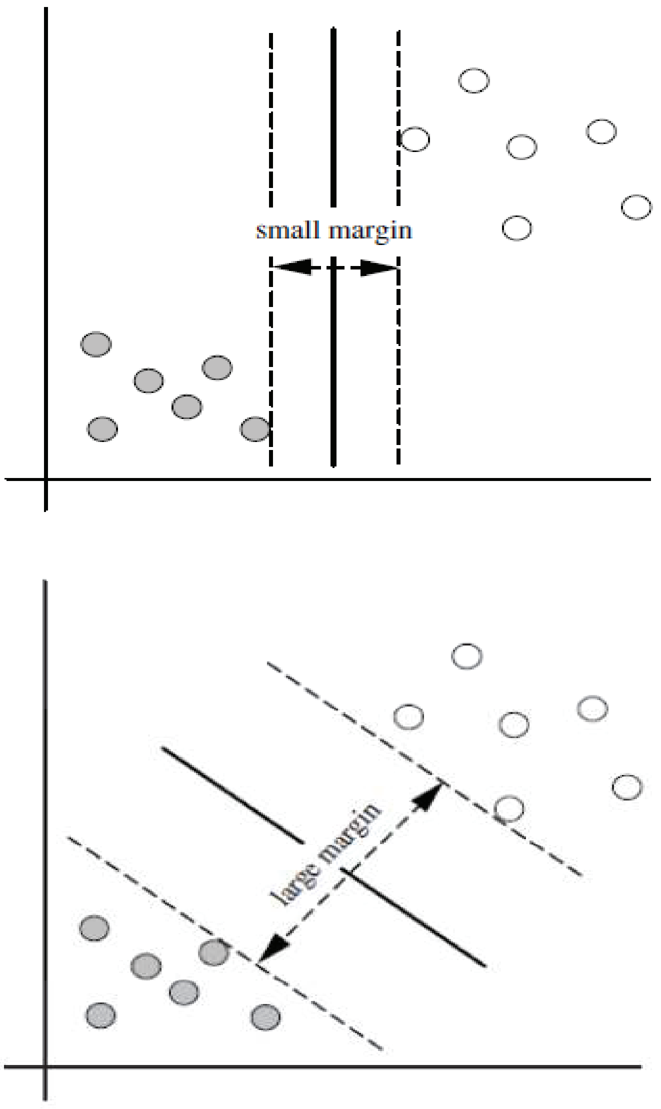


Figure 3-5: Two possible separating hyperplane and their associated margins.

Thus, any point that lies above the separating hyperplane satisfies

$$w0+w1x1+w2x2 > 0 \qquad 3.3$$

Similarly, any point that lies below the separating hyperplane satisfies

$$w0+w1x1+w2x2 < 0 \qquad 3.4$$

The weights can be adjusted so that the hyperplanes defining the "sides" of the margin can be written as

$$H1 : w0+w1x1+w2x2 \_ 1 \text{ for } yi = +1, \text{ and} \qquad 3.5$$

$$H2 : w0+w1x1+w2x2 \_ -1 \text{ for } yi = -1 \qquad 3.6$$

That is, any tuple that falls on or above *H*1 belongs to class +1, and any tuple that falls on or below *H*2 belongs to class -1. Combining the two inequalities of Equations (3.5) and (3.6),

$$y\,(w_0 + w_1x_1 + w_2x_2) \geq 1, \forall i \qquad 3.7$$

Equation (3.7) are called support vectors. In Figure 3.7, the support vectors are shown encircled with a thicker border.

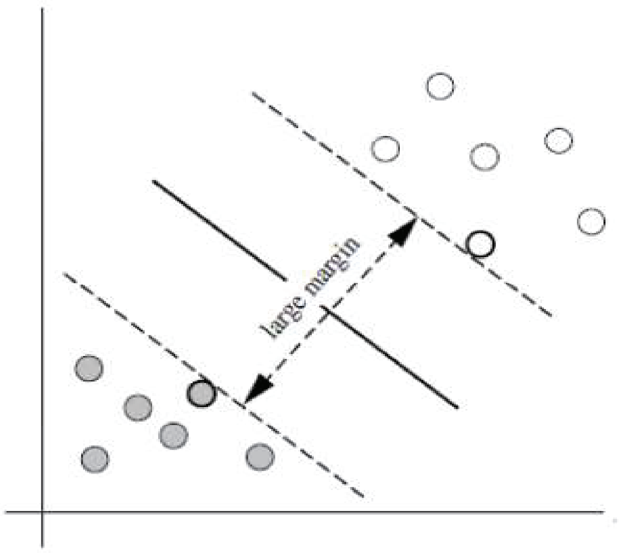


Figure 3-6: The support vectors are shown with a thicker border.

From the above, we can obtain formulae for the size of the maximal margin. The distance from the separating hyperplane to any point on *H*1 is $\frac{1}{||W||}$, by definition, this is equal to the distance from any point on H_2 to the separating hyperplane. Rewriting (Equation 3.7) using a Lagrangian formulation and then solving for the solution using Karush-Kuhn-Tucker (KKT) conditions. The MMH is a linear class boundary, and so the corresponding SVM can be used to classify linearly separable data. We refer to such a trained SVM as a *linear SVM*.

Based on the Lagrangian formulation the MMH can be rewritten as the decision boundary. The α_i are Lagrangian multipliers. SVMs tend to be less prone to overfitting than some other methods [19] [128].

$$\mathbf{d\,(XT\,) = \sum_{i=1}^{l} y_i \propto_i X_i X_T + b_o} \qquad \mathbf{3.8}$$

The Case When the Data are linearly inseparable

The approach explained above for linear SVMs can be extended to create *nonlinear SVMs* for the classification of *linearly inseparable data* (*nonlinearly separable data* or *nonlinear data*) (figure 3.8). Such SVMs are capable of finding nonlinear decision boundaries (i.e., nonlinear hyper-surfaces) in input space [129] [130].

A nonlinear SVM can be found by extending the approach for linear SVMs as follows. The first step is transforming the original input data into a higher dimensional space using a nonlinear mapping. There are several common nonlinear mappings to be use. Once the data have been transformed into the new higher space, the second step is to search for a linear separating hyperplane in the new space. It would end up with a quadratic optimization problem which is solved using the linear SVM formulation. The maximal marginal hyperplane found in the new space corresponds to a nonlinear separating hyper-surface in the original space [19] [131].

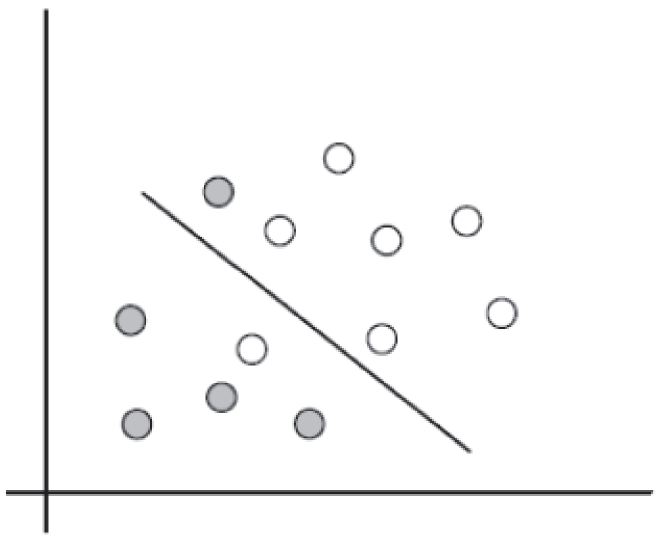


Figure 3-7: A simple 2-D case showing linearly inseparable data.

3.6.2 Ensemble Classification Models

The ensemble learning consists of three methodologies; sequential, concurrent and combining methodologies. The ensemble models in this research were designed using ensemble combining classifiers to combine the base-classifiers models. The combining methodologies used in this research are Averaging, Voting, Stacking and Grading [132].

3.6.2.1 Averaging

Averaging is the most popular and basic combination method for numeric outputs. Suppose we are given a set of T individual learners $(h_1, \ldots, h_T)$ and the output of h_i for the instance x is $h_i(x) \in \mathrm{R}$, our task is to combine h_i's to attain the final classification [90].

Simple Averaging

Simple averaging obtains the combined output by averaging the outputs of individual learners directly [91]. Specifically, simple averaging gives the combined output $H(x)$ as

$$\mathbf{H}(\mathbf{x}) = \frac{1}{T}\sum_{i=1}^{T} \mathbf{h}_i(\mathbf{x}) \quad 3.9$$

Suppose the underlying true function we try to learn is $f(x)$, and x is sampled according to a distribution $p(x)$. The output of each learner can be written as the true value plus an error item, i.e.,

$$\mathbf{h}_i(\mathbf{x}) = \mathbf{f}(\mathbf{x}) + \boldsymbol{\varepsilon}_i(\mathbf{x}), \quad i = 1, \ldots, T \qquad 3.10$$

Owing to its simplicity and effectiveness, simple averaging is among the most popularly used methods and represents the first choice in many real applications [90] [92].

3.6.2.2 Voting

Voting is the most popular and basic combination method for nominal outputs. Given a set of T individual classifiers ($h_1, \ldots, h_T$) and our task is to combine h_i's to predict the class label from a set of l possible class labels $(c_1, \ldots, c_l)$ [133]. It is usually assumed that for an instance $\boldsymbol{x}$, the outputs of the classifier h_i are given as an l-dimensional label vector $(h^1_i(\boldsymbol{x}), \ldots, h^l_i(\boldsymbol{x}))^T$, where $h_{ji}(\boldsymbol{x})$ is the output of h_i for the class label c_j [90]. The $h_{ji}(\boldsymbol{x})$ can take different types of values according to the information provided by the individual classifiers, e.g.,

- *Class label*: $h_{ji}(\boldsymbol{x}) \in (0, 1)$, which takes value one if h_i predicts c_j as the class label and zero otherwise.

- *Class probability*: $h_{ji}(\boldsymbol{x}) \in [0, 1]$, which can be regarded as an estimate of the conditional probability $P(c_j | \boldsymbol{x})$.

There are several types of voting Majority Voting, Plurality Voting, Weighted Voting and Soft Voting [134].

Plurality Voting

On the contrary to majority voting which requires the final class label to take at least half of votes, plurality voting takes the class label which receives the largest number of votes as the final winner [135]. That is, the output class label of the ensemble is

$$\mathbf{H(x) = c_{\arg_j \max \sum_{i=1}^{T} h_i^j(x)}} \qquad \mathbf{3.11}$$

Ties are broken at random. It is obvious that plurality voting does not have a reject option, since it can always find a class label receiving the largest number of votes. Moreover, in the case of binary classification, which is the case in this research, plurality voting is similar to majority voting [136] [137].

3.6.2.3 Stacking

Stacking is a general procedure where a learner is trained to combine the individual learners [138]. Here, the individual learners are called the *first-level learners*, while the combiner is called the *second-level learner*, or meta-learner [139].

The basic idea is to train the first-level learners using the original training data set, and then generate a new data set for training the second-level learner, where the outputs of the first-level learners are regarded as input features while the original labels are still regarded as labels of the new training data [140]. The first-level learners are often generated by applying different learning algorithms, and so, stacked ensembles are often heterogeneous, though it is also possible to construct homogeneous stacked ensembles.

```
Input: Data set $D = ((x_1, y_1), (x_2, y_2) . . . (x_m, y_m))$;
First-level learning algorithms $£_1, . . . , £_T$ ;
Second-level learning algorithm L.
Process:
        for $t = 1, . . . , T$ :   % Train a first-level learner by applying the
            $h_t = £_t(D)$;          % first-level learning algorithm $£_t$
        end
        $D' = \emptyset$;                % Generate a new data set
        for i = 1, . . .,m:
            for t = 1, . . . , T :
                $z_{it} = h_t(x_i)$;
            end
            $D' = D' \cup ((z_{i1}, . . . , zi_T), y_i)$;
        end
        $h' = £(D')$;                  % Train the second-level learner $h'$ by
    applying
                                 % the second-level learning algorithm £ to the
                                 % new data set $D'$.
Output: $H(x) = h'(h_1(x) . . . h_T(x))$
```

Figure 3-8: General Stacking Procedure.

In the training phase of stacking, a new data set needs to be generated from the first-level classifiers. If the exact data that are used to train the first-level learner are also used to generate the new data set for training the second-level learner, there will be a high risk of overfitting [141]. Hence, it is suggested that the instances used for generating the new data set are excluded from the training examples for the first-level learners, and a cross validation or leave-one-out procedure is often recommended.

Take k-fold cross-validation for example. Here, the original training data set D is randomly split into k almost equal parts $D_1, \ldots, D_k$. Define Dj and $D_{(-j)} = D \setminus D_j$ to be the test and training sets for the jth fold. Given T learning algorithms, a first-level learner $h_t^{(-j)}$ t is obtained by invoking the tth learning algorithm on $D_{(-j)}$. For each x_i in D_j, the test set of the jth fold, let z_{it} denote the output of the learner $h_t^{(-j)}$ on x_i. Then, at the end of the entire cross-validation process, the new data set is generated from the T individual learners as

$$\mathbf{D'} = (\mathbf{z_{i1}}, \ldots, \mathbf{z_{iT}}, \mathbf{y_i})_{\mathbf{i=1}}^{\mathbf{m}} \quad \mathbf{3.12}$$

On which the second-level learning algorithm will be applied, and the resulting learner h' is a function of $(z_1, \ldots, z_T)$ for y. After generating the new data set, generally, the final first-level learners are re-generated by training on the whole training data.

For stacked classification it is important to consider the types of features for the new training data, and the types of learning algorithms for the second-level learner [140].

3.6.2.4 Grading

This technique uses “graded” classifications as metalevel classes. The term graded is used in the meaning of classifications that have been labelled as correct or incorrect [93]. The method transforms the classification made by the *k* different classifiers into *k* training sets by using the examples *k* times and attaching them to a new binary class in each occurrence. This class shows whether the *k*–th classifier is highly accurate classifier capable of correctly classifying, compared to the real class of the example.

When the base classifier misclassifies, each base classifier, one meta-classifier is learned whose task is to classify. The base classifier classifies the unlabeled instance. The final classification is reached by classifications of those base classifiers that are classified by meta-classification schemes correctly [142]. Grading may be considered as a generalization of cross-validation selection, since it divides the training data into *k* subsets and builds *k*-1 classifiers by dropping one subset at a time and then using it to find a misclassification rate. The classifier with the smallest misclassification rate is chosen [94]. Grading tries to make the final classification for each example is independent from the other examples. This is done by using only those classifiers that are expected to classify those examples correct. The main difference between grading and other combining classifiers is that the former does not change the example attributes by replacing them with class predictions or class probabilities (or adding them to it). Instead it modifies the class values. Furthermore, in grading several sets of meta-data are created, one for each base classifier. Grading uses disagreement with the target function to produce a new training set [88].

3.7 Evaluation and Validation measures

There are several measures used for evaluation and validation of the designed models. Recall, precision and Precision-Recall (PR) curve had been used in this research since they are the most suitable for the type of data used in this research, which is mentioned in more detail in Section 2.5.

Recall measures how many positive cases are correctly classified as fraud, while **Precision** measures how many cases classified as fraud are really fraud. That is,

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \qquad 3.13$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \qquad 3.14$$

By these definitions, the precision does not contain any information about *FN (fraud not been classified as fraud)*, and the recall does not contain any information about *FP (non-fraud been classified as fraud)*. Therefore they are complementary to each other. Though a high precision and a high recall are both desired, there are often conflicts to achieve the two goals together since *FP* usually becomes larger when *TP* increases [143].

To evaluate a model in various situations, e.g., with different class distributions, the **Precision-Recall (PR) curve** can be used. It plots recall on the *x*-axis and precision on the *y*-axis, as illustrated in Figure 3-10. A classifier corresponds to a point in the PR space. If it classifies all examples correctly, *Precision* = 1 and *Recall* = 1; if it classifies all examples as fraud, *Recall* = 1 and *Precision* = $(TP + FN) / (TP + FN + TN + FP)$; if it classifies all examples as non-fraud, *Recall* = 0 and *Precision* = 1. If two classifiers are compared, the one located on the upper right is better. A functional hypothesis $h: X \times Y \rightarrow R$ corresponds to a curve on which a series of (*Precision, Recall*) points can be generated by applying different thresholds on the outputs of *h* to separate different classes [90].

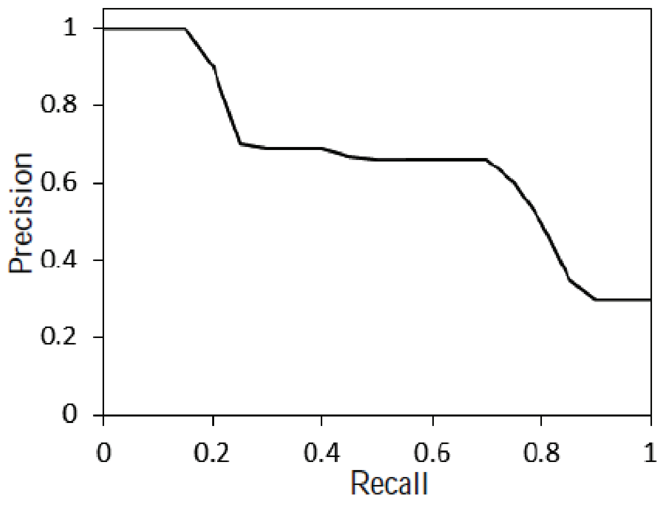


Figure 3-9: Precision-Recall (PR) curve [90]

3.8 Summary

This chapter first presents in some details the six research phases; Problem Domain Identification, Literature Review, Partitioning-Undersampling Techniques Evaluation, IFD Base-Classifier Models construction, Proposed IFD Ensemble Models construction and Evaluation.

This is followed by an overview of the dataset description; all the dataset attributes are explained. Data preprocessing methods used on the dataset are explained and the software tool used in producing this research.

Base-classifier and ensemble classifier learning algorithms used this research are described which were used to design the fraud detection model. Finally the validation measures used for evaluation of the research process and models are also presented.

4 CHAPTER FOUR
Insurance Fraud Detection (IFD) Models (Using Base-Classifiers)

4.1 Introduction

This Chapter deals with modelling the fraud detection of insurance claims. Three data mining classifiers were applied, decision tree, artificial neural network and support vector machine; to design fraud detection model. But first the problem of the imbalance dataset was solved by using the proposed partitioning–undersampling techniques.

The results in this chapter are divided into two parts. The first part is resampling the dataset using six different proposed partitioning-undersampling methods. The second part shows the result of designing the insurance fraud detection (IFD) models using data mining base-classifiers as explained in Chapter three (section 3.6.1).

This research proposed six different techniques of Partitioning-Undersampling for solving the imbalance data problem, as previously explained in Chapter two (Section 2.4) and Chapter three (section 3.2.3).

To compare between the six techniques, the subsamples resulting from these techniques are used to design IFD models using decision tree, for the reasons explained in Chapter three. The partitioning-undersampling techniques, which gave the best results with decision tree is used throughout the whole experiment.

In the second stage of this part of the experiment; support vector machine and artificial neural network are used to design other two IFD models.

The experiments used to solved the imbalance data problem and design insurance fraud detection (IFD) models using base-classifiers, which are phases 3 and 4 in this research methodology, as mentioned in Chapter three (Section 3.2) and are divided into three stages, as shown in Figure 4.1.

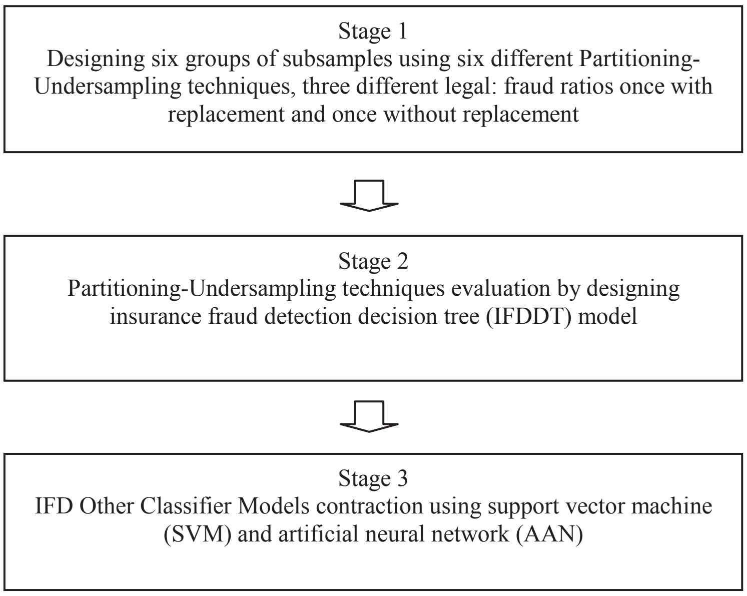

Stage 1
Designing six groups of subsamples using six different Partitioning-Undersampling techniques, three different legal: fraud ratios once with replacement and once without replacement
Stage 2
Partitioning-Undersampling techniques evaluation by designing insurance fraud detection decision tree (IFDDT) model
Stage 3
IFD Other Classifier Models contraction using support vector machine (SVM) and artificial neural network (AAN)

Figure 4-1: Stages of phases 3 and 4.

4.2 Partitioning-Undersampling Technique

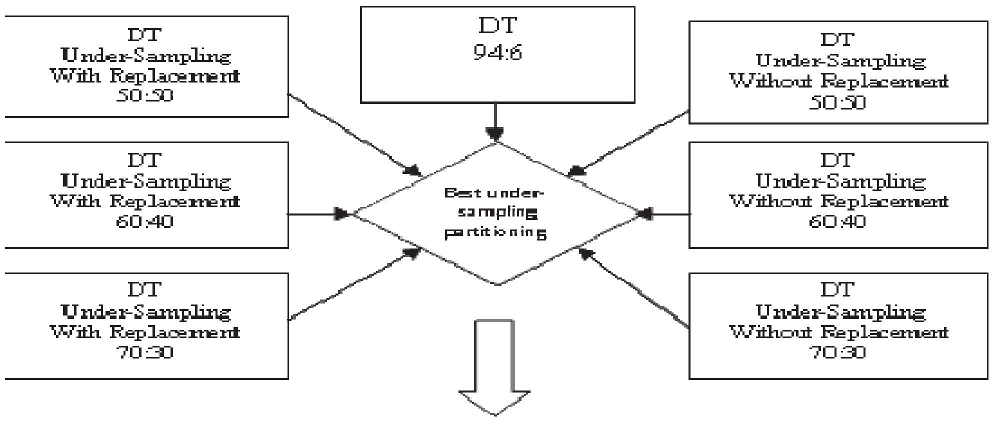
The dataset was divided into majority class and minority class. The minority class was untouched. The majority class was divided into ten subsamples (partitions), to choose the cases of each subsample (partition), two sampling methods were used, sampling with replacement and sampling without replacement. With each sampling techniques three different (legal: fraud) ratios, were used, i.e. 50:50, 60:40 and 70:30. This means that the majority class set is sampled using six techniques; sampling with replacement and 50:50 ratio, with replacement 60:40, with replacement 70:30, without replacement 50:50, without replacement 60:40 and without replacement 70:30. Ten subsamples were selected using each one of the six sampling techniques. As a result there were 60 subsamples plus the original dataset.

After finishing the resampling of the dataset there is no way to evaluate it except by using all the subsamples to design models then compare between these models. That was done in the second stage of this experiment.

4.3 IFD Decision Tree Models Designing

The DT algorithm is extremely time-efficient so it is used to compare the difference between partitioning-undersampling techniques and the original dataset. The specific DT classifier used in this research is a classifier for building a best-first decision tree classifier, that uses binary split for both nominal and numeric attributes. For more information see [144] [145] [146].

Insurance Fraud Detection Decision Tree (IFDDT) models were designed. All 60 subsamples and the original dataset were used to design IFDDT models. The 60 models designed using the 60 subsamples were compared with the model of the original dataset, to show the benefit of using sampling techniques to solve the imbalance problem. The partitioning-undersampling technique that produces the IFDDT model with the highest Recall is used through the rest of the experiment, as shown in (figure 4-2).



DT Under-Sampling With Replacement 50:50
DT 94:6
DT Under-Sampling Without Replacement 50:50
DT Under-Sampling With Replacement 60:40
Best under-sampling partitioning
DT Under-Sampling Without Replacement 60:40
DT Under-Sampling With Replacement 70:30
DT Under-Sampling Without Replacement 70:30

Figure 4-2: Choosing the best Under-Sampling Technique

The proposed six techniques of partitioning-undersampling techniques resample the dataset into six groups of subsamples each containing ten partitions (or subsample). The training dataset is divided into 10 subsets of equal size using sampling technique explained above. For each group decision tree models are designed and evaluated. For each model of the 60 models the validation measures (recall, precision and PRC Area) are calculated and then the average of each group of the ten partitions is calculated, as shown in Tables 4.2 - 4.4. Averaging is used here as an ensemble combining classifier, as explained in Chapter three (Section 3.6.2.1). Also a decision tree model is designed using the original data. Ten cross validation is use throughout the whole experiment. Finally a comparison between the average validation measures of the six groups of partitions and the validation measures of the model designed from the original dataset, in shown in Table 4.5.

The DT classifier's parameters are stated below, shown in Table 4.1. The debug was set to (off) since after testing with both (on) and (off) there was no improvement in the results. "Capability Pre-check of classifier" was set (on); for the same reason. "Heuristic Search" was set to (on) since it improves the results. "Min instances of terminal nodes" is set to 2 and "Num folds pruning" is set to 9; using trial and error those were the values that gave us the best results. "Pruning Strategy" "Random number Seed" "Training set %" "Split criteria" "Use one SE rule"; for each of these parameters several chooses were tested and those that gave best result were recorded.

Table 4.1: DT classifier parameters

Parameters	Valves
Debug	OFF
Capability Pre-check of classifier	ON
Heuristic Search	ON
Min instances of terminal nodes	2
Num folds pruning	9
Pruning Strategy	post pruning
Random number Seed	1
Training set %	1
error estimate	error rate
Split criteria	Gini index
Use One-SE rule	OFF

Table 4.2: Recall of IFD models using decision trees

Sub-Sample	SpSm 50-50	woR 50-50	SpSm 60-40	woR 60-40	SpSm 70-30	woR 70-30
S1	96.8%	94.3%	88.5%	95.9%	48.4%	51.2%
S2	96.8%	95.9%	90.6%	91.2%	49.5%	51.4%
S3	96.7%	95.8%	91.2%	88.2%	51.0%	51.9%
S4	95.0%	94.0%	92.8%	89.5%	51.9%	52.0%
S5	94.5%	93.5%	89.9%	92.1%	41.7%	55.8%
S6	94.1%	91.9%	89.7%	78.8%	44.7%	49.1%
S7	96.8%	95.8%	90.5%	82.8%	51.5%	51.2%
S8	96.8%	95.8%	88.2%	93.3%	51.2%	51.2%
S9	94.6%	93.0%	88.2%	95.8%	32.7%	47.2%
S10	95.8%	95.8%	93.2%	86.5%	51.2%	50.3%
	95.8%	94.6%	90.3%	89.4%	47.4%	51.1%

Where SpSm = Spread Subsample (with replacement); woR = without replacement

Table 4.3 : precision of IFD models using decision trees

Sub-Sample	SpSm 50-50	woR 50-50	SpSm 60-40	woR 60-40	SpSm 70-30	woR 70-30
S1	69.4%	68.9%	60.3%	59.0%	60.5%	56.6%
S2	68.2%	67.9%	59.1%	59.8%	60.5%	57.7%
S3	68.6%	69.1%	61.3%	59.2%	60.4%	57.0%
S4	68.9%	68.9%	61.7%	60.1%	57.6%	59.1%
S5	68.4%	68.4%	58.9%	60.8%	57.1%	57.5%
S6	67.9%	69.6%	59.5%	61.1%	58.2%	58.4%
S7	69.8%	69.8%	61.5%	63.0%	58.3%	59.0%
S8	68.7%	69.4%	59.9%	61.2%	58.3%	58.0%
S9	69.2%	69.2%	60.7%	59.8%	62.0%	57.5%
S10	69.2%	69.3%	60.3%	60.8%	60.0%	56.5%
	68.8%	69.0%	60.3%	60.5%	59.3%	57.7%

Where SpSm = Spread Subsample (with replacement); woR = without replacement

Table 4.4 : PRC Area of IFD models using decision trees

Sub-Sample	SpSm 50-50	woR 50-50	SpSm 60-40	woR 60-40	SpSm 70-30	woR 70-30
S1	73.0%	74.9%	73.1%	77.3%	81.2%	79.4%
S2	72.3%	72.6%	72.5%	75.4%	80.8%	77.6%
S3	75.3%	75.3%	74.5%	73.2%	80.3%	75.2%
S4	72.9%	72.9%	75.0%	74.7%	77.9%	79.1%
S5	73.0%	73.0%	73.8%	74.0%	79.7%	77.6%
S6	71.9%	73.9%	73.7%	72.0%	80.7%	78.3%
S7	74.9%	74.7%	75.4%	74.9%	78.4%	80.6%
S8	73.3%	73.9%	75.0%	76.9%	80.6%	80.5%
S9	74.7%	73.7%	73.3%	74.5%	79.6%	78.4%
S10	73.8%	73.7%	75.4%	72.9%	80.8%	75.5%
	73.5%	73.9%	74.2%	74.6%	80.0%	78.2%

Where SpSm = Spread Subsample (with replacement); woR = without replacement

Table 4.5: IFD models using decision trees

	recall	precision	PRC Area
SpSm50-50	95.8%	69.4%	73.0%
woR 50-50	94.6%	69.0%	73.9%
SpSm60-40	90.3%	60.3%	74.2%
woR 60-40	89.4%	60.5%	74.6%
SpSm70-30	47.4%	59.3%	80.0%
woR 70-30	51.1%	57.7%	78.2%
Original dataset	3.6%	86.8%	93.6%

Where SpSm = Spread Subsample (with replacement); woR = without replacement.

Looking at (Table 4.5) the partitioning-undersampling technique that was used to form the IFDDT model with the highest recall is; partitioning-undersampling using sampling with replacement and with (legit: fraud) ratio of 50: 50. This technique will be used though out the whole experiment.

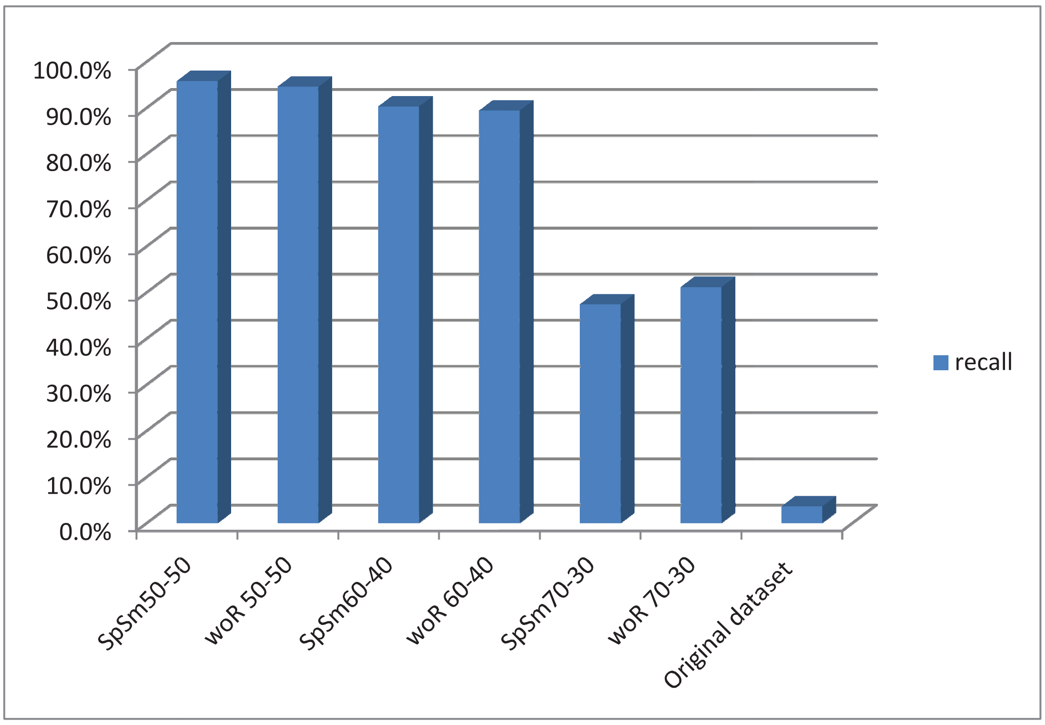


Figure 4-3: IFD models using decision trees

Looking at graph in Figure 4.3 it is easy to see the previously mention result that partitioning-undersampling (SpSm50-50) is the best and the big different between using the proposed technique and not using it as was the case with the original data.

4.4 IFD Support Vector Machine Models Design

IFDSVM (Insurance Fraud Detection Support Vector Machine) models are designed. The different parameters of the classifiers are adjusted using trial and error, shown on (Table 4.6).

The support vector machine classifier was repeated ten times each time with different Kernel; Radial Basis Function kernel, Polynomial Kernel, Person Universal Kernel and Normalized Polynomial Kernel.

The SVM models are evaluated and the model which yields the highest recall is chosen, also the precision and PRC Area are recorded to evaluate the models and chose between them in the case of equal recall. To obtain statistically reliable results, 10-fold cross validation method is used throughout this study. The 9 subsamples used as training data, and the remaining subsample as testing. Then the results on test fold are averaged. The recorded validation measures are shown in (Tables 4.7 - 4.9). Finally a comparison between the average validation measures of the best model for each kernel type, in shown in (Table 4.10).

The SVM classifier used in this research implements (John Platt)'s sequential minimal optimization algorithm, for training a support vector classifier. This implementation transforms nominal attributes into binary ones and normalizes all attributes by default [124] [147] [148] [149]. The classifier has several parameters which are shown in (Table 4.6). Several values where tested for those parameters and only those that gave the models with highest recall were used in the experiment, in case the recall are equal the models are chosen according to precision and PRC area.

Four Kernel types were used in the experiment and the result recorded.

Polynomial Kernel:
K(x , y) = <x , y> ^p
K(x , y) =(<x , y>+1) ^p

Normalize Polynomial Kernel:
K(x, y) = <x, y>/ sqrt(<x, x> <y, y>)
Where
<x, y> = Poly Kernel (x, y)

PuK:
The Pearson VII Function-Based Universal Kernel, (PuK) [150].

The RBK kernel (RBK):
K(x, y) = e ^ - (gamma ^ <x-y, x-y> ^ 2).

Table 4.6: SVM classifier parameters

Parameters	Valves
Build logistic models	OFF
Complexity parameter	1
Checked Turned	ON
Debug	OFF
Do not check capabilities	OFF
Epsilon	1.00E-12
Filter type	Normalize training data
Kernel	Normalized Polynomial
Num folds for cross-validation	5
Random seed	5
Tolerance Parameter	0.001

Table 4.7 : Recall of IFD models using support vector machine

Sub-	Kernel type			
sample	(RBF)	(PuK)	(PolyK)	(NPolyK)
S1	92.2%	94.9%	94.5%	95.6%
S2	92.1%	92.7%	92.4%	92.8%
S3	92.1%	93.9%	91.2%	92.4%
S4	92.2%	93.9%	91.7%	93.2%
S5	92.2%	93.8%	92.2%	92.8%
S6	92.2%	93.3%	93.4%	94.5%
S7	92.2%	94.5%	93.8%	95.4%
S8	92.2%	94.6%	93.9%	94.7%
S9	92.2%	95.0%	93.4%	95.0%
S10	92.2%	94.4%	93.9%	94.6%
	92.2%	94.1%	93.0%	94.1%

Table 4.8 : precision of IFD models using support vector machine

	Kernel type			
Subsample	(RBF)	(PuK)	(PolyK)	(NPolyK)
S1	68.6%	68.8%	69.0%	69.2%
S2	68.2%	68.3%	68.1%	68.6%
S3	68.1%	68.5%	69.4%	68.8%
S4	70.3%	70.6%	71.1%	70.8%
S5	67.5%	67.8%	67.9%	67.6%
S6	67.5%	67.6%	67.8%	67.8%
S7	69.1%	69.3%	69.1%	69.3%
S8	68.6%	68.7%	68.8%	69.1%
S9	68.4%	68.6%	68.7%	69.0%
S10	68.4%	68.4%	68.9%	69.2%
	68.5%	68.7%	68.9%	68.9%

Table 4.9 : PRC Area of IFD models using support vector machine

	Kernel type			
Subsample	(RBF)	(PuK)	(PolyK)	(NPolyK)
S1	69.6%	70.8%	70.9%	71.5%
S2	69.1%	69.5%	69.2%	69.8%
S3	69.1%	70.2%	70.0%	69.8%
S4	71.1%	72.1%	71.7%	72.0%
S5	68.5%	69.4%	68.9%	68.9%
S6	68.6%	69.0%	69.2%	69.9%
S7	70.0%	71.1%	70.6%	71.5%
S8	69.6%	70.6%	70.4%	71.0%
S9	69.4%	70.7%	70.2%	71.1%
S10	69.4%	70.2%	70.5%	71.1%
	69.4%	70.4%	70.2%	70.7%

Table 4.10 : best models for each kernel type

Kernel Type		Recall	Precision	PRC Area
Radial Basis Function	(RBF)	92.2%	68.5%	69.4%
Person VII function-based universal	(PuK)	93.0%	68.9%	70.2%
Polynomial	(PolyK)	94.1%	68.7%	70.4%
Normalized Polynomial	(NPolyK)	94.1%	68.9%	70.7%

The difference between the four insurance fraud detection support vector machine models is shown in the graph bellow (figure 4-4). There is no difference between the model using polynomial kernel and normalized polynomial. But comparing between using precision the model using normalized polynomial gave better results as it is shown in (table 4.10).

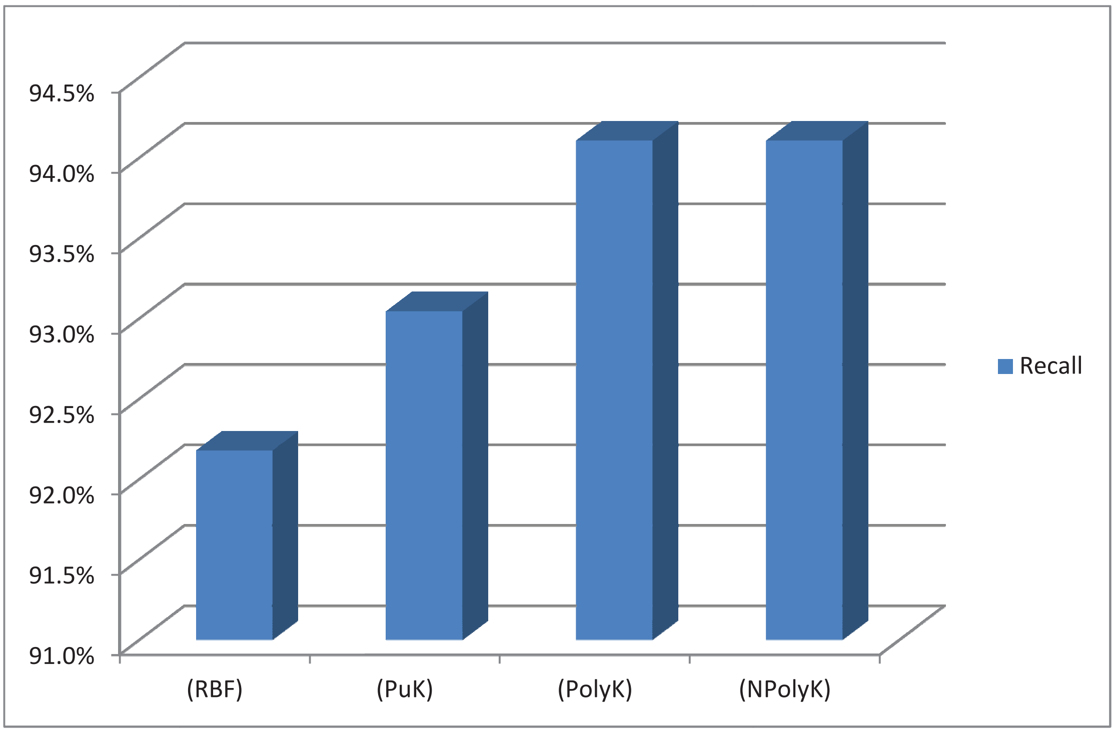


Figure 4-4 : IFD models using support vector machine

4.5 IFD Artificial Neural Network Models Design

Then an artificial neural network (ANN) model is trained by a back propagation algorithm.

Several parameters of the ANN model were tested by trial and error to get the best results that were recorded as shown in (Table 4.11). The parameters of the network (number of hidden layer, no of units in each layer, learning rate and momentum) were adjusted using trial and error method. The best result was recorded. First the number of the hidden layer is changed from the default 1 to 2 and when increase more the results got very low so only 1 and 2 were recorded. Also different values for the unit number in each layer were tested and again the best results were recorded. The same thing was done for learning rate and momentum. These classifier parameters are shown in (Table 4.11).

Recall, precision and PRC area are recorded for each partition model is shown in (Tables 4.13 - 4.15) and the comparison between the averaged models is shown in (Table 4.16).

The models designed using the above ANN classifier were compared with the models designed using another ANN that trains a multilayer perceptron using Optimization class, by minimizing the squared error plus a quadratic penalty with BFGS method shown in (Table 4.12). The attributes are standardized. The ridge parameter is used to determine the penalty on the size of the weights. The number of the hidden units can also be specified. Paralleled calculation of squared error and gradient is possible when multiple CPU cores are present. Data is split into batches and processed in separate threads in this case. Nominal attributes are processed using unsupervised Nominal-To-Binary filter.

Table 4.11: ANN classifier parameters.

Parameters	Valves
GUI	F
Auto build	T
Debug	F
Decay	F
Do-Not-Check-Capabilities	F
Hidden layers	a
Learning rate	0.3
Momentum	0.2
Nominal to Binary Filter	T
Normalize Attributes	T
Normalize Numeric class	T
Reset	T
Seed	0
Training time	500
Validation Set Size	0
Validation threshold	20

Table 4.12: ANN classifier using BFGS parameters.

Parameters	Valves
Debug	F
Do-Not-Check-Capabilities	F
Num-Functions	2
Num-Threads	1
Pool size	1
Ridge	0.01
Seed	1
Tolerance	1.0 E-6
use CGD	F

Table 4.13: Recall of IFD models using artificial neural network

Hidden Layer	1	1	2	2	1	1	1	1	1
Layer Units	2	3	2 ; 2	2 ; 3	2	2	2	2	2
learning Rate	0.3	0.3	0.3	0.3	0.25	0.35	0.3	0.3	BFGS Optimi zation
Momentum	0.2	0.2	0.2	0.2	0.2	0.2	0.25	0.15	
S1	91.9%	83.4%	90.7%	90.5%	92.6%	86.3%	86.5%	91.9%	92.1%
S2	93.4%	87.1%	88.0%	86.5%	93.6%	92.3%	87.9%	89.9%	89.7%
S3	87.2%	81.7%	90.6%	82.8%	84.4%	87.5%	87.0%	86.8%	87.6%
S4	90.2%	84.3%	88.6%	90.8%	64.0%	90.8%	90.6%	90.7%	90.4%
S5	91.8%	84.0%	91.8%	91.7%	92.8%	93.6%	92.8%	91.5%	89.2%
S6	87.3%	85.5%	83.0%	87.6%	86.3%	86.3%	88.1%	86.8%	90.9%
S7	91.9%	90.7%	90.8%	91.3%	91.9%	92.5%	91.9%	91.7%	91.3%
S8	86.3%	83.4%	82.0%	91.1%	84.9%	86.5%	86.6%	84.8%	88.2%
S9	85.2%	86.3%	90.7%	92.0%	85.6%	84.9%	84.9%	91.2%	89.1%
S10	92.0%	89.8%	89.5%	92.1%	92.2%	91.9%	92.4%	91.0%	89.7%
	89.7%	85.6%	88.6%	89.6%	86.8%	89.3%	88.9%	89.6%	89.8%

Looking at the recall of the designed IFD models using artificial neural network the best result was given by ANN using (BFGS optimization). Although the differences are not large but still the research stick to the rule stated in the beginning of the research the highest recall is chosen. This model will be used throughout the rest of the experiment.

Table 4.14 : precision of IFD models using artificial neural network

Hidden Layer Layer Units learning Rate Momentum	1 2 0.3 0.2	1 3 0.3 0.2	2 2 ; 2 0.3 0.2	2 2 ; 3 0.3 0.2	1 2 0.25 0.2	1 2 0.35 0.2	1 2 0.3 0.25	1 2 0.3 0.15	1 2 BFGS Optimization
S1	69.1%	69.6%	69.1%	69.2%	69.0%	69.4%	69.2%	68.7%	69.7%
S2	68.6%	68.8%	68.8%	68.0%	68.6%	68.7%	69.3%	68.7%	68.4%
S3	68.6%	69.9%	68.7%	68.7%	69.1%	68.8%	68.8%	68.7%	69.3%
S4	70.6%	70.9%	71.4%	70.8%	83.6%	71.0%	70.7%	70.8%	71.3%
S5	67.8%	68.5%	67.6%	67.4%	67.9%	67.7%	67.5%	67.2%	68.0%
S6	68.5%	68.1%	68.4%	68.0%	68.3%	68.5%	68.4%	68.4%	68.3%
S7	69.5%	69.0%	69.7%	69.2%	69.7%	69.6%	69.8%	69.4%	70.1%
S8	69.5%	70.3%	69.4%	69.0%	69.8%	69.7%	69.8%	69.7%	69.8%
S9	68.9%	68.9%	68.9%	68.4%	69.0%	68.8%	69.0%	68.9%	70.0%
S10	68.6%	69.3%	69.2%	68.4%	68.5%	68.7%	68.5%	67.9%	69.3%
	69.0%	69.3%	69.1%	68.7%	70.3%	69.1%	69.1%	68.8%	69.4%

Table 4.15 : PRC Area of IFD models using artificial neural network

Hidden Layer Layer Units learning Rate Momentum	1 2 0.3 0.2	1 3 0.3 0.2	2 2 ; 2 0.3 0.2	2 2 ; 3 0.3 0.2	1 2 0.25 0.2	1 2 0.35 0.2	1 2 0.3 0.25	1 2 0.3 0.15	1 2 BFGS Optimi zation
S1	77.3%	77.4%	76.5%	76.5%	77.2%	77.4%	77.2%	77.3%	78.1%
S2	76.8%	75.6%	74.2%	75.8%	76.8%	76.8%	76.6%	76.6%	77.8%
S3	77.3%	77.3%	76.4%	75.9%	77.6%	77.6%	77.7%	77.1%	78.9%
S4	77.6%	77.1%	78.0%	78.0%	78.9%	78.1%	78.0%	77.9%	78.7%
S5	75.3%	75.7%	75.3%	74.6%	75.3%	75.3%	75.1%	76.0%	75.7%
S6	76.5%	77.0%	75.8%	76.8%	76.2%	76.2%	75.6%	76.7%	78.1%
S7	75.8%	76.3%	75.8%	75.8%	76.6%	75.9%	76.4%	75.6%	78.0%
S8	76.5%	77.9%	76.5%	76.6%	76.6%	76.9%	76.9%	76.6%	78.9%
S9	75.6%	77.2%	74.2%	76.2%	75.6%	75.8%	75.6%	75.9%	79.1%
S10	77.8%	76.7%	74.8%	75.7%	77.5%	77.9%	77.7%	77.5%	78.7%
	76.7%	76.8%	75.8%	76.2%	76.8%	76.8%	76.7%	76.7%	78.2%

Table 4.16 : IFD models using artificial neural network

case no.	No of Hidden Layer	No. of Units in Each Layer	learning Rate	Momentum	recall	precision	PRC Area
1	1	2	0.3	0.2	89.7%	69.0%	76.7%
2	1	3	0.3	0.2	85.6%	69.3%	76.8%
3	2	2 ; 2	0.3	0.2	88.6%	69.1%	75.8%
4	2	2 ; 3	0.3	0.2	89.6%	68.7%	76.2%
5	1	2	0.25	0.2	86.8%	70.3%	76.8%
6	1	2	0.35	0.2	89.3%	69.1%	76.8%
7	1	2	0.3	0.25	88.9%	69.1%	76.7%
8	1	2	0.3	0.15	89.6%	68.8%	76.7%
9	1	2	Optimization using BFGS		89.8%	69.4%	78.2%

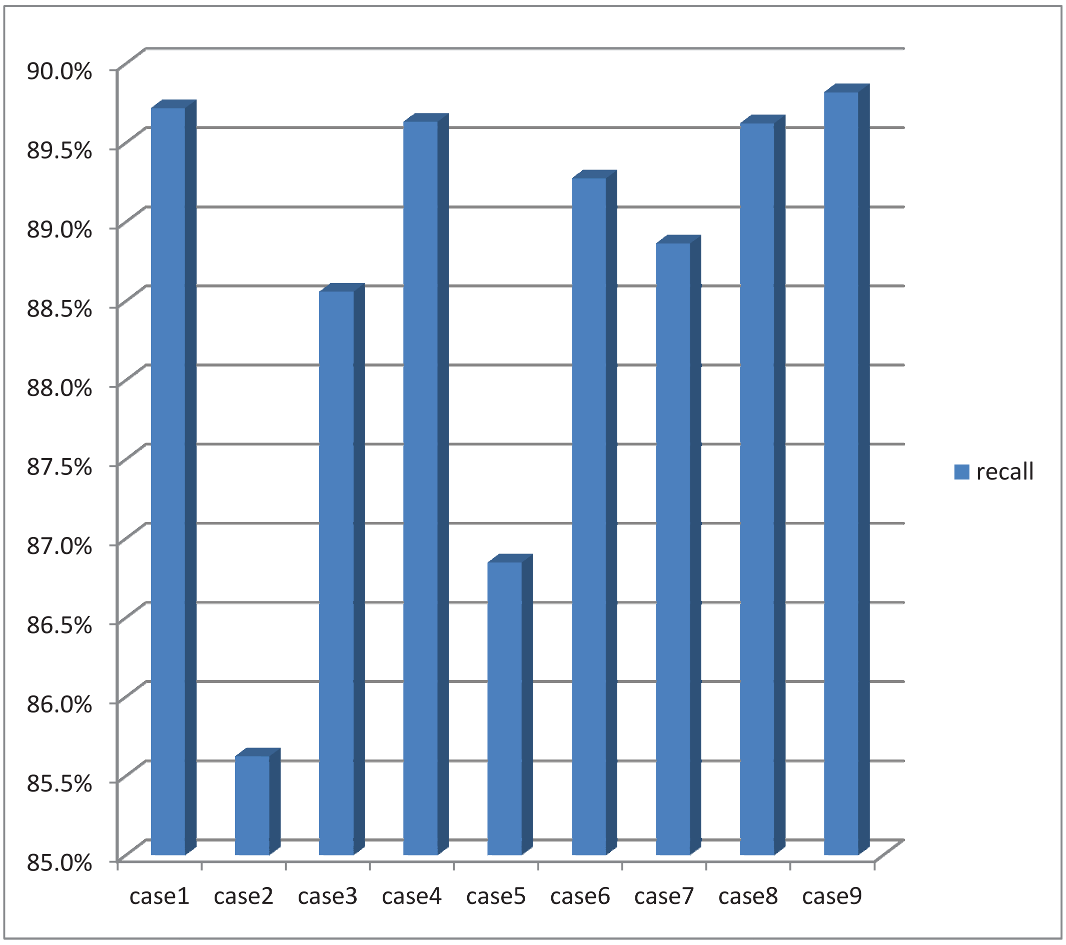


Figure 4-5 : IFD models using artificial neural network

For testing the proposed mode it was applied on another imbalanced dataset (German dataset), the results are reported in (Table 4.17).

Table 4.17: German dataset results

Classifier Type	Recall	Precision	PRC Area
Artificial Neural Network	76.8	75.8	80.5
Support Vector Machine	77.0	76.1	68.1
Decision Tree	73.5	72.3	68.2

Those results were compared with the result published in previous paper [151]. The highest recall (Default% as referred to in the previous paper) is 68.0, which is much less than 76.8, which is the recall of the proposed model using NN reported in (Table 4.18).

Table 4.18 : Results comparison with other works in the literature

Classifier Type	Previous model using NN	Proposed model using NN
Artificial Neural Network	68.0	76.8

4.6 Summary

The first step in this experiment is solving the imbalance dataset problem by applying six proposed partitioning - undersampling techniques. To evaluate those six techniques the resulting subsamples were used to design IFD models using decision tree classifier. Then another two base-classifier, support vector machine and artificial neural network, were used to design IFD models using, the resampling technique that gave the best results. The proposed IFD models are applied on another imbalanced dataset for comparison. At the end the resulting IFD artificial neural network model is compared to other artificial neural network in previous work.

5 CHAPTER FIVE
Designing IFD Ensemble Models

5.1 Introduction

In this part of the experiment phase five of the research is applied, that is ensemble combining classifiers are applied on base-classifier to design proposed ensemble models. The experiment is designed to use the best base-classifiers from the previous chapter: IFDDT, IFDSVM and IFDANN. Three ensemble combining classifiers are used namely grading, stacking and voting.

Experiments were done using the previously explained ensemble combining classifiers as discussed in Chapter 3 (Section 3.6.2). Various comparisons are done between different IFD base-classifiers combined models and finally with the results an ensemble model from previous study. The results of using ensemble combining classifiers, in hope to enhance the performance of the IFD base-classifiers models, are show here.

5.2 IFD Models Using Grading

Grading is a meta combining ensemble classifier, that grade the base classifiers [93], as explained in Chapter 3. All possible combinations of the three best classifiers from the previous Chapter, are formed by using grading also single classifiers were used since it could be applied on single classifier. As mentioned previously the models are evaluated using recall, precision and PRC area, which were recorded. The models are chosen according to the recall value if two or more have the same recall value then they are chose according to precision or both.

The Grading classifier has several parameters, shown in Table 5.1. Those parameters were adjusted by trial and error choosing the best. The meta-classifier used for the grading is OneR, which was chosen from several chooses. OneR is a classifier that uses the minimum-error attribute for prediction of numeric attributes [152].

The ten partitions of the subsample are used to design an ensemble model and then an average is calculated for the ten partitions. The recall of the ensemble models using grading of base-classifier models combinations is shown in Table 5.2. The precision of these models is shown in Table 5.3 and the PRC area of these models is shown in Table 5.4.

Table 5.1: Grading meta-classifier parameters

Parameters	Valves
Base classifiers used	all possible combination of the 3 base-classifiers
Debug	OFF
Do not check capabilities	OFF
Meta classifier used	OneR
Num ensemble execution slots	1
Num cross-validation folds	10
Random number seed	1

Table 5.2 : Recall of Grading of base-classifier models combination

	Grading SVM	Grading ANN	Grading DT	Grading SVM-ANN	Grading SVM-DT	Grading ANN-SVM	Grading SVM-ANN-DT
1	95.3%	90.2%	93.3%	94.0%	95.7%	92.1%	95.1%
2	93.0%	89.6%	95.1%	91.0%	95.0%	92.3%	93.5%
3	91.9%	86.8%	92.2%	93.0%	95.7%	92.5%	92.7%
4	92.9%	90.9%	93.8%	94.3%	93.9%	93.4%	93.0%
5	91.5%	88.2%	95.4%	93.7%	95.6%	93.6%	93.8%
6	94.0%	89.4%	95.8%	94.0%	95.8%	94.0%	94.7%
7	95.3%	90.5%	93.2%	94.5%	95.8%	92.3%	95.2%
8	93.9%	89.9%	92.8%	93.9%	95.6%	91.5%	94.8%
9	94.5%	90.0%	95.6%	93.5%	95.8%	93.5%	95.2%
10	93.9%	89.9%	95.4%	93.9%	95.8%	93.7%	94.7%
	93.6%	89.5%	94.3%	93.6%	95.4%	92.9%	94.3%

Table 5.3 : Precision of Grading of base-classifier models combination

	Grading SVM	Grading ANN	Grading DT	Grading SVM-ANN	Grading SVM-DT	Grading ANN-SVM	Grading SVM-ANN-DT
1	69.2%	69.0%	68.9%	68.9%	69.3%	68.9%	69.4%
2	68.5%	68.2%	68.6%	68.7%	68.8%	68.3%	68.3%
3	68.7%	68.5%	68.5%	68.5%	68.7%	68.5%	68.6%
4	71.4%	71.5%	71.2%	70.8%	71.0%	71.0%	71.1%
5	67.7%	67.7%	68.0%	67.9%	68.0%	67.8%	67.9%
6	67.8%	69.1%	68.4%	68.8%	68.4%	68.6%	68.2%
7	69.3%	69.9%	70.1%	69.8%	69.7%	70.2%	69.9%
8	69.2%	70.1%	69.3%	69.3%	69.1%	69.3%	69.5%
9	68.9%	70.0%	69.1%	69.1%	69.2%	69.2%	69.3%
10	69.1%	69.1%	69.2%	69.2%	69.2%	69.4%	69.3%
	69.0%	69.3%	69.1%	69.1%	69.1%	69.1%	69.2%

Table 5.4 : PRC Area of Grading of base-classifier models combination

	Grading SVM	Grading ANN	Grading DT	Grading SVM-ANN	Grading SVM-DT	Grading ANN-SVM	Grading SVM-ANN-DT
1	71.4%	69.2%	70.3%	70.6%	71.6%	69.9%	71.5%
2	69.8%	68.3%	70.8%	69.3%	70.9%	69.4%	69.8%
3	69.6%	67.6%	69.5%	69.8%	71.0%	69.6%	69.8%
4	71.7%	71.8%	72.7%	72.5%	72.5%	72.3%	72.2%
5	68.2%	67.3%	70.3%	69.5%	70.3%	69.4%	69.5%
6	69.5%	69.0%	70.8%	70.2%	70.8%	70.3%	70.2%
7	71.5%	70.1%	71.4%	71.6%	72.0%	71.1%	72.0%
8	70.8%	70.1%	70.5%	70.9%	71.4%	70.0%	71.5%
9	70.8%	70.1%	71.4%	70.5%	71.5%	70.7%	71.4%
10	70.8%	69.2%	71.4%	70.9%	71.5%	70.9%	71.2%
	70.4%	69.3%	70.9%	70.6%	71.4%	70.4%	70.9%

The models of the ten partitions of the subsample are combined in one model using averaging. The seven models that were designed using grading are compared using recall, precision and PRC area as shown in Table 5.5.

Table 5.5 : Grading of base-classifier models combination

Grading	Recall	Precision	PRC Area
SVM	93.6%	69.0%	70.4%
ANN	89.5%	69.3%	69.3%
DT	94.3%	69.1%	70.9%
SVM-ANN	93.6%	69.1%	70.6%
SVM-DT	95.4%	69.1%	71.4%
ANN-DT	92.9%	69.1%	70.4%
SVM-ANN-DT	94.3%	69.2%	70.9%

Grading did not improve any base-classifier model as their recall value decreased; IFDANN model using grading to 89.5% from 89.8% IFDSVM model using grading, recall decreased to 93.4% from 94.1%. IFDDT model using grading; recall decrease to 94.3% from 95.8%. The best model designed using grading is the ensemble combination of IFDSVM and IFDDT; which recall is 95.4%.

The comparison between those seven models using recall is shown in Figure 5.1.

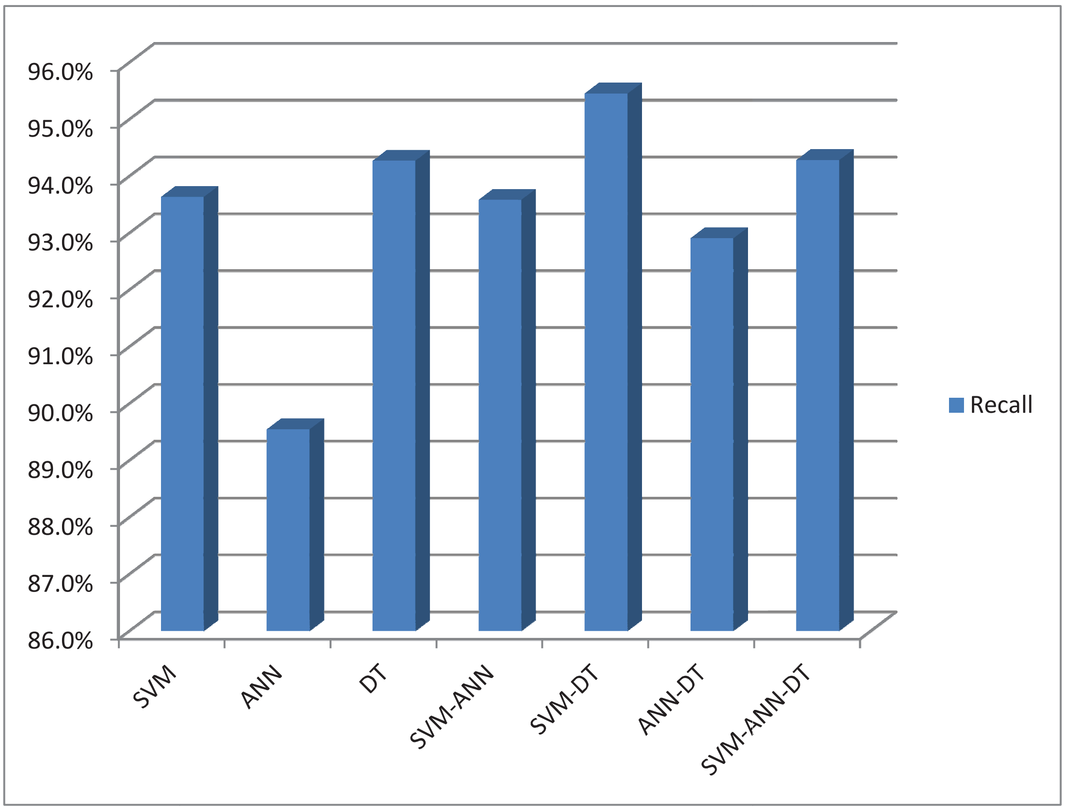


Figure 5-1 : Grading of base-classifier models combination

5.3 IFD Models Using Stacking

In this Section, another ensemble IFD models were designed using stacking, which is a meta combining ensemble classifier [153]. Stacking meta-classifier has several parameters; shown in Table 5.6; whose values were chosen by trial and error. This classifier requires a meta-classifier; to be a numeric prediction scheme. The meta-classifier chose is Linear Regression, which was chosen after testing a lot of meta-classifiers. This classifier learns a Simple Linear Regression model. It picks the attribute that result in the lowest squared error. Missing values are not allowed. It can only deal with numeric attributes.

The best base-classifiers models and all possible combinations of those models were used to design the ensemble IFD models. The recall, precision and PRC area of those models are shown in Tables 5.7 - 5.9.

Table 5.6 : Stacking meta-classifier parameters.

Parameters	Valves
Base classifiers used	All possible combination of the 3 base-classifiers
Debug	OFF
Do not check capabilities	OFF
Meta classifier used	Linear Regression
Num ensemble execution slots	1
Num cross-validation folds	10
Random number seed	1

Table 5.7: Recall of stacking of base-classifier models combination

	Stacking SVM	Stacking ANN	Stacking DT	Stacking SVM-ANN	Stacking SVM-DT	Stacking ANN-DT	Stacking SVM-ANN-DT
1	95.3%	94.1%	93.3%	94.8%	93.8%	95.1%	95.3%
2	92.8%	93.8%	95.0%	93.6%	94.4%	94.5%	93.9%
3	92.1%	93.5%	93.5%	93.2%	91.7%	93.5%	90.8%
4	92.6%	94.3%	93.9%	93.4%	93.1%	93.7%	93.5%
5	91.5%	93.8%	95.6%	93.7%	95.6%	95.4%	95.4%
6	94.0%	94.6%	95.8%	93.5%	95.8%	95.7%	95.2%
7	95.3%	95.1%	93.2%	95.2%	93.5%	94.3%	93.7%
8	94.3%	94.4%	93.2%	95.2%	92.5%	93.2%	92.4%
9	94.5%	94.8%	95.6%	94.1%	95.1%	95.4%	94.9%
10	93.7%	94.7%	95.4%	93.9%	95.1%	95.6%	95.0%
	93.6%	94.3%	94.4%	94.1%	94.1%	94.6%	94.0%

Table 5.8: Precision of stacking of base-classifier models combination

	Stacking SVM	Stacking ANN	Stacking DT	Stacking SVM-ANN	Stacking SVM-DT	Stacking ANN-DT	Stacking SVM-ANN-DT
1	69.2%	69.2%	69.0%	69.6%	69.2%	69.1%	69.4%
2	69.0%	68.3%	68.8%	69.0%	69.0%	68.8%	68.9%
3	68.8%	68.3%	68.7%	68.7%	68.6%	68.6%	68.6%
4	70.9%	71.1%	71.3%	71.0%	71.3%	71.2%	71.2%
5	67.8%	67.5%	68.0%	67.4%	68.0%	68.0%	68.0%
6	67.8%	68.3%	68.4%	68.1%	68.3%	68.4%	68.4%
7	69.3%	70.1%	70.1%	69.6%	69.9%	69.7%	70.2%
8	69.1%	69.2%	69.3%	69.5%	69.4%	69.1%	69.6%
9	69.0%	68.7%	69.1%	69.1%	68.8%	69.0%	69.0%
10	69.1%	68.9%	69.2%	69.4%	69.0%	69.1%	69.2%
	69.0%	68.9%	69.2%	69.1%	69.1%	69.1%	69.2%

Table 5.9: PRC area of stacking of base-classifier models combination

	Stacking SVM	Stacking ANN	Stacking DT	Stacking SVM-ANN	Stacking SVM-DT	Stacking ANN-DT	Stacking SVM-ANN-DT
1	71.9%	78.0%	74.4%	78.0%	75.4%	78.1%	78.3%
2	70.3%	76.8%	77.8%	77.0%	78.1%	79.5%	79.6%
3	72.2%	77.5%	74.3%	77.4%	75.7%	78.0%	77.7%
4	73.4%	79.2%	77.9%	79.3%	77.8%	80.1%	79.8%
5	70.4%	75.8%	74.3%	75.8%	74.3%	76.9%	76.7%
6	70.9%	77.8%	74.9%	78.4%	75.0%	79.4%	80.1%
7	72.4%	78.0%	77.0%	78.5%	77.6%	78.9%	79.1%
8	71.9%	78.5%	74.4%	78.9%	77.7%	77.9%	76.1%
9	72.5%	78.7%	77.2%	78.8%	77.7%	79.5%	79.2%
10	71.7%	77.0%	77.1%	77.1%	77.4%	78.9%	79.4%
	71.8%	77.7%	75.9%	77.9%	76.7%	78.7%	78.6%

After combing the ten partitions of the subsample using averaging the resulting models are compared according to recall, precision ad PRC area as shown in Table 5.10. Applying stacking on the single base-classifier model improved the IFDANN model from 89.8% to 94.3% but the other two models did not improve. IFDSVM model using stacking, recall decreased to 93.6% from 94.1%. IFDDT model using stacking; recall decrease to 94.4% from 95.8%. The best model designed using stacking is the ensemble combination of IFDANN and IFDDT; which recall is 94.6%.

The recall of IFD models, designed using stacking, is shown in Figure 5.2.

Table 5.10 : Stacking of base-classifier models combination

STACKING	Recall	Precision	PRC Area
SVM	93.6%	69.0%	71.8%
ANN	94.3%	68.9%	77.7%
DT	94.4%	69.2%	75.9%
SVM-ANN	94.1%	69.1%	77.9%
SVM-DT	94.1%	69.1%	76.7%
ANN-DT	94.6%	69.1%	78.7%
SVM-ANN-DT	94.0%	69.2%	78.6%

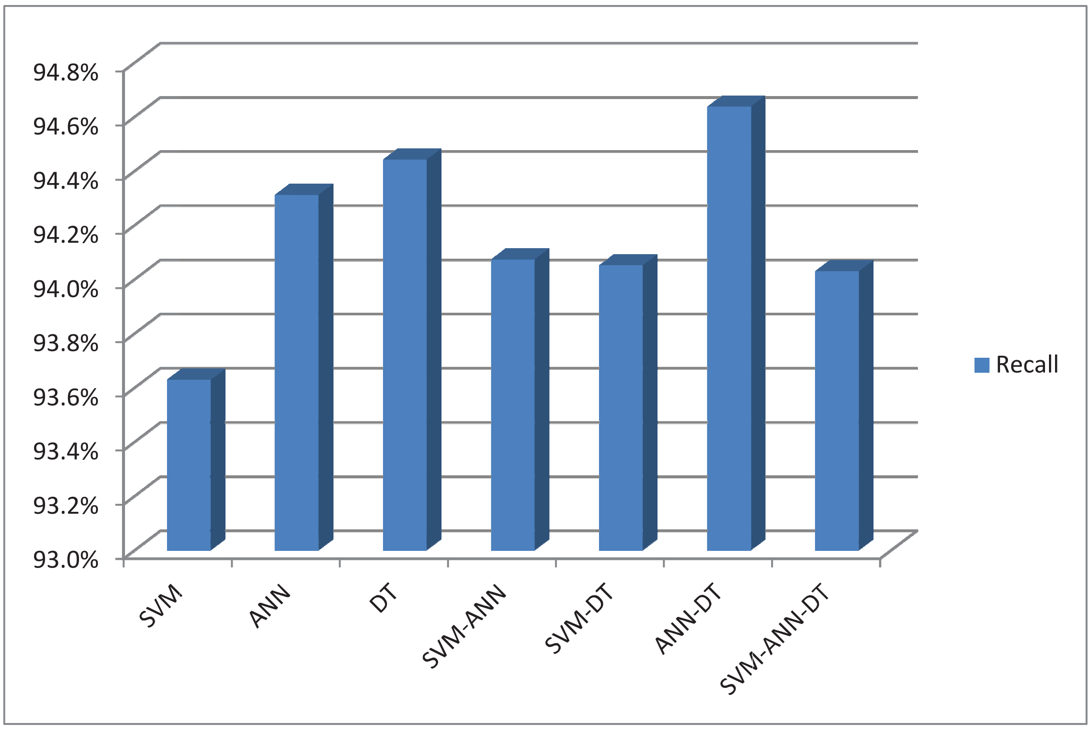


Figure 5-2 : Stacking of base-classifier models combination

5.4 IFD models using Voting

In this Section, another IFD models were designed using voting, which is a simple combining ensemble classifier. It is a class for combining classifiers; different combinations of probability estimates for classification are available [154, 155] . The classifier has several parameters whose best values were found by testing several values by trial and error, shown in Table (5.11). The best base-classifiers models and all possible combinations of those models were used to design the ensemble IFD models. The recall, precision and PRC area of those models are shown in Tables 5.12 - 5.14.

Table 5.11: voting meta-classifier parameters.

Parameters	Valves
Base classifiers used	All possible combination of the 3 base-classifiers
Combination Rule	Average of probabilities
Debug	OFF
Do not check capabilities	OFF
Random number seed	1

Table 5.12: recall of vote of base-classifier models combination

	Vote SVM	Vote ANN	Vote DT	Vote SVM-ANN	Vote SVM-DT	Vote ANN-DT	Vote SVM-ANN-DT
1	95.3%	92.1%	95.6%	95.3%	95.3%	95.2%	95.3%
2	92.8%	88.7%	93.3%	92.8%	93.3%	91.9%	93.2%
3	92.1%	88.3%	92.5%	92.1%	92.2%	91.4%	92.2%
4	92.6%	90.1%	94.8%	92.6%	92.8%	94.3%	92.7%
5	91.5%	95.6%	95.6%	91.5%	91.5%	94.5%	92.0%
6	94.0%	95.8%	95.8%	94.0%	94.6%	94.9%	94.5%
7	95.2%	95.0%	95.0%	95.2%	95.3%	94.9%	95.7%
8	94.3%	92.8%	92.8%	94.3%	94.7%	92.6%	94.5%
9	94.5%	95.0%	95.0%	94.5%	94.5%	94.4%	94.7%
10	93.7%	92.4%	92.4%	93.7%	93.8%	91.3%	93.7%
	93.6%	92.6%	94.3%	93.6%	93.8%	93.5%	93.8%

Table 5.13: precision of vote of base-classifier models combination

	Vote SVM	Vote ANN	Vote DT	Vote SVM-ANN	Vote SVM-DT	Vote ANN-DT	Vote SVM-ANN-DT
1	69.2%	69.7%	69.4%	69.2%	69.2%	69.5%	69.6%
2	69.0%	68.4%	68.9%	69.0%	69.1%	69.2%	69.1%
3	68.8%	69.3%	68.5%	68.8%	68.8%	68.8%	69.0%
4	70.9%	71.3%	71.1%	70.9%	70.9%	71.0%	71.0%
5	67.8%	68.0%	68.0%	67.8%	67.8%	67.8%	67.8%
6	67.8%	68.4%	68.4%	67.8%	67.9%	68.4%	68.2%
7	69.3%	69.8%	69.8%	69.3%	69.3%	69.7%	69.7%
8	69.1%	69.1%	69.1%	69.1%	69.1%	69.2%	69.6%
9	69.0%	68.9%	68.9%	69.0%	69.0%	69.1%	69.4%
10	69.1%	69.0%	69.0%	69.1%	69.1%	69.0%	69.6%
	69.0%	69.2%	69.1%	69.0%	69.0%	69.2%	69.3%

Table 5.14 : PRC area of vote of base-classifier models combination

	Vote SVM	Vote ANN	Vote DT	Vote SVM-ANN	Vote SVM-DT	Vote ANN-DT	Vote SVM-ANN-DT
1	71.4%	78.1%	75.2%	78.3%	75.6%	79.9%	79.9%
2	70.2%	77.8%	75.2%	78.1%	77.0%	79.6%	79.8%
3	69.8%	79.0%	74.5%	78.9%	76.4%	79.5%	79.5%
4	71.9%	78.7%	77.1%	79.1%	77.5%	80.2%	80.6%
5	68.6%	74.0%	74.0%	75.9%	74.0%	77.1%	77.2%
6	69.5%	75.2%	75.2%	77.6%	74.9%	79.5%	79.2%
7	71.5%	76.8%	76.8%	78.6%	76.9%	79.4%	79.3%
8	70.9%	73.4%	73.4%	78.5%	75.6%	78.7%	78.7%
9	70.8%	76.5%	76.5%	78.8%	76.9%	79.7%	79.6%
10	70.7%	72.7%	72.7%	78.9%	75.6%	78.4%	78.5%
	70.5%	76.2%	75.1%	78.3%	76.0%	79.2%	79.2%

After combing the ten partitions of the subsample using averaging the resulting models are compared according to recall, precision ad PRC area as shown in Table (5.12). Applying vote on the single base-classifier model improved the IFDANN model a lot from 89.8% to 92.6% but the other two models didn’t improve. IFDSVM model using voting, recall decreased to 93.6% from 94.1%. IFDDT model using voting; recall decrease to 94.3% from 95.8%. The best model designed using voting is the IFDDT; which recall is 94.3%.

Table 5.15 : Vote of base-classifier models combination

VOTE	Recall	Precision	PRC Area
SVM	93.6%	69.0%	70.5%
ANN	92.6%	69.2%	76.2%
DT	94.3%	69.1%	75.1%
SVM-ANN	93.6%	69.0%	78.3%
SVM-DT	93.8%	69.0%	76.0%
ANN-DT	93.5%	69.2%	79.2%
SVM-ANN-DT	93.8%	69.3%	79.2%

The recall of IFD models, designed using vote, is shown in Figure 5.3.

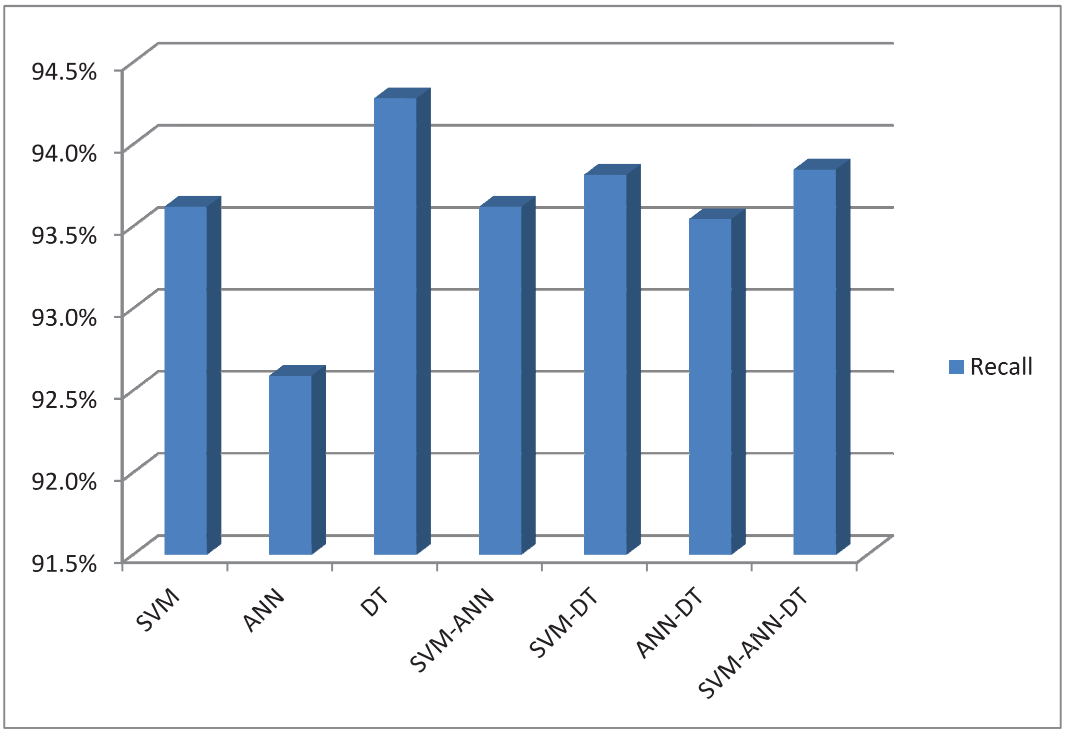


Figure 5-3 : Vote of base-classifier models combination

5.5 Evaluating the IFD Ensemble Models

The following tables contain several comparisons between IFD ensemble models. In Table 5.16 the averages of the partitions of the subsample are stated for all the twenty one models. Then in Table 5.17, the models are sorted according to their recall, if the recall is the same then according to precision and if their precision is the same then according to their PRC area. In Table 5.18 the IFD base-classifiers were added to the list and sorted. Table 5.19 shows the p-values that were calculated, using t-test with nil-hypothesis that the mean of samples are equal. All models are compared with the best model DT. The models have p-value less than 0.05; thus the nil-hypothesis is reject; meaning that the difference between the models is a significant difference [156].

The proposed models were applied on German dataset to use for evaluation of the proposed models and the results are illustrated in Table 5.20. A comparing between averaged partitions and German dataset models is shown in Table 5.21.

Table 5.16 : IFD ensemble models

	Recall	Precision	PRC Area
Vote (SVM)	93.6%	69.0%	70.5%
Vote (ANN)	92.6%	69.2%	76.2%
Vote (DT)	94.3%	69.1%	75.1%
Vote (SVM-ANN)	93.6%	69.0%	78.3%
Vote (SVM-DT)	93.8%	69.0%	76.0%
Vote (ANN-DT)	93.5%	69.2%	79.2%
Vote (SVM-ANN-DT)	93.8%	69.3%	79.2%
Stacking (SVM)	93.6%	69.0%	71.8%
Stacking (ANN)	94.3%	68.9%	77.7%
Stacking (DT)	94.4%	69.2%	75.9%
Stacking(SVM-ANN)	94.1%	69.1%	77.9%
Stacking (SVM-DT)	94.1%	69.1%	76.7%
Stacking (ANN-DT)	94.6%	69.1%	78.7%
Stacking (SVM-ANN-DT)	94.0%	69.2%	78.6%
Grading (SVM)	93.6%	69.0%	70.4%
Grading (ANN)	89.5%	69.3%	69.3%
Grading (DT)	94.3%	69.1%	70.9%
Grading (SVM-ANN)	93.6%	69.1%	70.6%
Grading (SVM-DT)	95.4%	69.1%	71.4%
Grading (ANN-DT)	92.9%	69.1%	70.4%
Grading(SVM-ANN-DT)	94.3%	69.2%	70.9%

Table 5.17 : IFD ensemble models sorted according to the validation measures

	Recall	Precision	PRC Area
Grading(SVM-DT)	95.4%	69.1%	71.4%
Stacking(ANN-DT)	94.6%	69.1%	78.7%
Stacking (DT)	94.4%	69.2%	75.9%
Grading (SVM-ANN-DT)	94.3%	69.2%	70.9%
Grading (DT)	94.3%	69.1%	70.9%
Vote (DT)	94.3%	69.1%	75.1%
Stacking (ANN)	94.3%	68.9%	77.7%
Stacking (SVM-ANN)	94.1%	69.1%	77.9%
Stacking (SVM-DT)	94.1%	69.1%	76.7%
Stacking (SVM-ANN-DT)	94.0%	69.2%	78.6%
Vote (SVM-ANN-DT)	93.8%	69.3%	79.2%
Vote (SVM-DT)	93.8%	69.0%	76.0%
Grading (SVM-ANN)	93.6%	69.1%	70.6%
Vote (SVM-ANN)	93.6%	69.0%	78.3%
Stacking (SVM)	93.6%	69.0%	71.8%
Vote (SVM)	93.6%	69.0%	70.5%
Grading (SVM)	93.6%	69.0%	70.4%
Vote (ANN-DT)	93.5%	69.2%	79.2%
Grading (ANN-DT)	92.9%	69.1%	70.4%
Vote (ANN)	92.6%	69.2%	76.2%
Grading (ANN)	89.5%	69.3%	69.3%

As reported in Table 5.17 Grading (SVM-DT) gave the best results. All best six classifiers contained DT base-classifier, showing that DT gave best results.

Table 5.18 : IFD base-classifiers and ensemble models sorted

	Recall	Precision	PRC Area
DT	95.8%	69.4%	73.0%
Grading (SVM-DT)	95.4%	69.1%	71.4%
Stacking (ANN-DT)	94.6%	69.1%	78.7%
Stacking (DT)	94.4%	69.2%	75.9%
Grading (SVM-ANN-DT)	94.3%	69.2%	70.9%
Vote (DT)	94.3%	69.1%	75.1%
Grading (DT)	94.3%	69.1%	70.9%
Stacking (ANN)	94.3%	68.9%	77.7%
Stacking (SVM-ANN)	94.1%	69.1%	77.9%
Stacking (SVM-DT)	94.1%	69.1%	76.7%
SVM	94.1%	68.9%	70.7%
Stacking (SVM-ANN-DT)	94.0%	69.2%	78.6%
Vote (SVM-ANN-DT)	93.8%	69.3%	79.2%
Vote (SVM-DT)	93.8%	69.0%	76.0%
Grading (SVM-ANN)	93.6%	69.1%	70.6%
Vote (SVM-ANN)	93.6%	69.0%	78.3%
Stacking (SVM)	93.6%	69.0%	71.8%
Vote (SVM)	93.6%	69.0%	70.5%
Grading (SVM)	93.6%	69.0%	70.4%
Vote (ANN-DT)	93.5%	69.2%	79.2%
Grading (ANN-DT)	92.9%	69.1%	70.4%
Vote (ANN)	92.6%	69.2%	76.2%
ANN	89.8%	69.0%	76.7%
Grading (ANN)	89.5%	69.3%	69.3%

When the results of the base-classifiers were added to the results of the ensemble classifiers, DT classifier gave the best results with recall of 95.8% as shown in Table 5.18. The next five best were ensemble classifiers, Grading (SVM-DT), Stacking (ANN-DT), Stacking (DT), Grading (SVM-ANN-DT) and Vote (DT). This results show that although the ensemble classifiers didn't give the best results still it gives good results.

The p-values were very small values as reported in table 5.19 and this indicated that there is significant difference between the models.

Table 5.19: IFD base-classifiers and ensemble models p-value.

	Recall	Precision	PRC Area	p-value
DT	95.8%	69.4%	73.0%	
Grading(SVM-DT)	95.4%	69.1%	71.4%	1.325E-11
Stacking(ANN-DT)	94.6%	69.1%	78.7%	6.787E-22
Stacking(DT)	94.4%	69.2%	75.9%	1.555E-30
grading(SVM-ANN-DT)	94.3%	69.2%	70.9%	2.342E-09
vote(DT)	94.3%	69.1%	75.1%	1.000E+00
grading(DT)	94.3%	69.1%	70.9%	1.043E-15
stacking(ANN)	94.3%	68.9%	77.7%	7.866E-82
stacking(SVM-ANN)	94.1%	69.1%	77.9%	2.408E-28
stacking(SVM-DT)	94.1%	69.1%	76.7%	1.487E-19
SVM	94.1%	68.9%	70.7%	1.023E-21
stacking(SVM-ANN-DT)	94.0%	69.2%	78.6%	1.364E-18
vote(SVM-ANN-DT)	93.8%	69.3%	79.2%	3.418E-06
vote(SVM-DT)	93.8%	69.0%	76.0%	1.425E-08
grading(SVM-ANN)	93.6%	69.1%	70.6%	2.531E-09
vote(SVM-ANN)	93.6%	69.0%	78.3%	1.372E-10
stacking(SVM)	93.6%	69.0%	71.8%	5.047E-42
vote(SVM)	93.6%	69.0%	70.5%	3.267E-12
grading(SVM)	93.6%	69.0%	70.4%	2.473E-10
vote(ANN-DT)	93.5%	69.2%	79.2%	8.324E-25
grading(ANN-DT)	92.9%	69.1%	70.4%	2.352E-14
vote(ANN)	92.6%	69.2%	76.2%	6.543E-32
ANN	89.8%	69.0%	76.7%	6.788E-33
grading(ANN)	89.5%	69.3%	69.3%	2.134E-13

Table 5.20 : German dataset base-classifier and ensemble models

Algorithm	Recall	Precision	PRC Area
DT	73.5%	72.3%	68.2%
SVM	77.0%	76.1%	68.1%
ANN	76.8%	75.8%	80.5%
Grading (SVM)	60.2%	38.8%	62.5%
Grading (ANN)	75.5%	74.2%	68.2%
Grading (DT)	74.8%	73.1%	66.4%
Grading (SVM-ANN)	76.0%	75.6%	66.3%
Grading (SVM-DT)	75.8%	76.4%	65.6%
Grading (ANN-DT)	76.9%	76.7%	67.3%
Grading (SVM-ANN-DT)	**78.4%**	77.7%	69.8%
Stacking (SVM)	77.0%	76.1%	68.2%
Stacking (ANN)	74.4%	74.4%	74.4%
Stacking (DT)	74.7%	73.1%	69.1%
Stacking (SVM-ANN)	77.0%	76.3%	74.3%
Stacking (SVM-DT)	76.4%	75.2%	74.0%
Stacking (ANN-DT)	74.7%	73.4%	74.3%
Stacking (SVM-ANN-DT)	77.0%	76.0%	76.0%
Vote (SVM)	77.0%	76.1%	68.1%
Vote (ANN)	75.3%	74.3%	78.6%
Vote (DT)	73.5%	72.3%	68.2%
Vote (SVM-ANN)	77.0%	76.1%	80.6%
Vote (SVM-DT)	75.0%	73.6%	75.0%
Vote (ANN-DT)	74.3%	72.9%	78.1%
Vote (SVM-ANN-DT)	77.3%	76.2%	80.1%

The proposed models were applied on German dataset to use for evaluation of the proposed models. A comparing between averaged partitions and German dataset models is shown in Table 5.21. The best two models that gave best results are the model using Grading (SVM-ANN-DT) with recall 78.4% and the model using Vote (SVM-ANN-DT) with recall 77.3%. This results show that the ensemble combing classifiers give enhance the IFD models that were designed at first using base-classifiers.

Table 5.21 : comparing between averaged partitions and German dataset models

	Averaged Partitions Recall	German Dataset Recall
DT	95.8%	73.5%
SVM	94.1%	77.0%
ANN	89.8%	76.8%
Vote (SVM)	93.6%	77.0%
Vote (ANN)	92.6%	75.3%
Vote (DT)	94.3%	73.5%
Vote (SVM-ANN)	93.6%	77.0%
Vote (SVM-DT)	93.8%	75.0%
Vote (ANN-DT)	93.5%	74.3%
Vote (SVM-ANN-DT)	93.8%	77.3%
Stacking (SVM)	93.6%	77.0%
Stacking (ANN)	94.3%	74.4%
Stacking (DT)	94.4%	74.7%
Stacking (SVM-ANN)	94.1%	77.0%
Stacking (SVM-DT)	94.1%	76.4%
Stacking (ANN-DT)	94.6%	74.7%
Stacking (SVM-ANN-DT)	94.0%	77.0%
Grading (SVM)	93.6%	60.2%
Grading (ANN)	89.5%	75.5%
Grading (DT)	94.3%	74.8%
Grading (SVM-ANN)	93.6%	76.0%
Grading (SVM-DT)	95.4%	75.8%
Grading (ANN-DT)	92.9%	76.9%
Grading (SVM-ANN-DT)	94.3%	78.4%

5.6 Summary

Three ensemble combining classifiers were used to form IFD ensemble models. For building IFD ensemble model, base-classifier models and different combinations of base-classifiers models, were used.

The proposed models were evaluated according to their recall. The model with the highest recall is IFDDT which was designed using decision tree base classifier, but the second, third and fourth best classifiers are; model using grading (SVM-DT) with recall 95.4%, model using stacking (ANN-DT) with recall 94.6% and the model using grading (SVM-ANN-DT) with recall 94.3%; which all were designed using ensemble classifiers.

These novel proposed models were applied on another imbalance dataset again the models with the highest recall were the ensemble combination of the three base-classifiers models. The best two models are the model using Grading (SVM-ANN-DT) with recall 78.4% and the model using Vote (SVM-ANN-DT) with recall 77.3%, this proves that ensemble combining classifiers produce powerful IFD models. The difference between those models was shown to be statistically significant difference by calculating their p-value.

6 CHAPTER SIX
Conclusions and Future Research

6.1 Introduction

This Chapter provides a summary of the main achievement and major findings of this research. It also identifies the main contributions of the novel proposed partitioning-undersampling techniques developed in this thesis. Finally, future research topics are identified that might be of interest for further investigation.

6.2 Summary of the Thesis

This thesis main objective is, designing a novel ensemble insurance fraud detection model using classification algorithms for detecting fraudulent insurance claims, in order to provide effective and efficient services for customers and keeping the company stable enough in the market environment. This was done by fulfilling the specific objectives of this research

1) Solve the problem of imbalance data by using proposed Partitioning-undersampling techniques using sampling-with-replacement and sampling-without-replacement.

2) Develop detection models that will help to detect fraudulent insurance claims using base classifiers.

3) Build an ensemble model to enhance the performance of the previous models.

Before trying to fulfil those objects, an extensive literature review was conducted. This review was conducted from three different points; Insurance

Fraud Detection (IFD) in general concentrating on the data mining classifiers used in this area, the other point is the techniques used for solving the imbalanced data problem concentrating on the performance measure used for evaluation of classifiers of imbalance dataset, then the last point is Automobile Insurance Fraud Detection that are using imbalance dataset (presented in chapter 2).

The first problem facing designing any fraud detection model is the imbalance dataset, since most dataset used for fraud detection are imbalance.

When applying conventional machine learning algorithms the mining of imbalanced datasets can result in models that are strongly predictive for the larger class, while delivering performance, which is poorly predictive for the minority class. This is due to the fact that conventional classifiers will attempt to return the most correct predictions based upon the entire dataset, this results in them categorizing all data as belonging to the larger class. This class is usually the class, which is of least interest to the data-mining problem. In the case of fraud detection data-mining, the majority class is the class where no fraud has occurred and the minority class being fraud. When minority class is very small a learner can deliver very high predictive accuracy even though it has classified none of the minority class correctly. Taking the area of interest of this thesis (fraud) the minority class would be ’possibly fraudulent claims’ while the majority class would be ’non fraudulent’. Several classification techniques have been proposed and applied in the literature for imbalanced classification problems. This research uses resampling-then-classification. However, there are ensemble algorithms, which build an ingratiation of classifiers. The objective of using ensemble learning is to improve the classification performance.

A novel technique was proposed in this research, as it is clear from the literature review that a similar technique was not used before, this technique is partitioning-undersampling.

The dataset was divided into majority class and minority class. The minority class was untouched. The majority class was divided into ten subsamples (partitions), to choose the cases of each subsample (partition), two sampling methods were used, sampling with replacement and sampling without replacement. With each sampling techniques three different (legal: fraud) ratios, were used, i.e. 50:50, 60:40 and 70:30. This means that the majority class set is sampled using six techniques; sampling with replacement and 50:50 ratio, with replacement 60:40, with replacement 70:30, without replacement 50:50, without replacement 60:40 and without replacement 70:30. Ten subsamples were selected using each one of the six sampling techniques. As a result there were 60 subsamples plus the original dataset. After finishing the resampling of the dataset there is no way to evaluate it except by using all the subsamples to design models then compare between this models. The best proposed partitioning-undersampling technique is (sampling with replacement and 50:50 ratio), and by this the first objective (Solve the problem of imbalance data by using proposed Partitioning-undersampling techniques) was accomplished.

To compare between the six techniques, the subsamples resulting from these techniques are used to design IFD models using decision tree. The partitioning-undersampling techniques, which gave the best results with decision tree are used throughout the whole experiment.

Having already constructed IFDDT for the purpose of evaluating the proposed six partitioning-under-sampling techniques, now other two IFD models are constructed using support vector machine and artificial neural network. The parameters of both models are changed by trial and error to choose the best and thus designing IFDSVM and IFDANN models.

Then IFDSVM (Insurance Fraud Detection Support Vector Machine) models are designed. The different parameters of the classifiers are adjusted using trial and error. The support vector machine classifier was repeated ten times each time with different Kernel; Radial Basis Function kernel, Polynomial Kernel, Person Universal Kernel and Normalized Polynomial Kernel. The SVM models are evaluated and the model, which yields the highest recall is chosen. Then an artificial neural network (ANN) model is trained by a back propagation algorithm. Several parameters of the ANN model were tested by trial and error to get the best results that were recorded. To obtain statistically reliable results, 10-fold cross validation method is used throughout this study. The results of the ten subsamples are averaged. By choosing the best IFDDT, IFDSVM and IFDANN models; the second objective (develop detection models that will help to detect fraudulent insurance claims using base classifiers) was accomplished.

Then ensemble combining classifiers are applied on the designed base-classifier models to design the proposed IFD ensemble models. The experiment is designed to use the best base-classifiers from the previous chapter: IFDDT, IFDSVM and IFDANN. Three different ensemble combining classifiers are used namely grading, stacking and vote. The ten partitions of the subsample are used to design an ensemble model and then an average is calculated for the ten partitions.

Several comparisons were performed between all twenty one IFD ensemble models. Then the p-values were calculated, using t-test with nil-hypothesis that the mean of samples are equal. All models have p-value less than 0.05; thus the nil-hypothesis is reject; meaning that the difference between the models is a significant difference. The proposed models were applied on German dataset to use for evaluation of the proposed models and the results, those results were compared to previous study showing an improvement in the proposed models results. By applying the ensemble classifiers of the best models; the third objective (Build an ensemble model to enhance the performance of the previous models) was accomplished.

6.3 Contributions of the Research

- The main contribution of this research is the development of a novel insurance fraud detection model using classification algorithms for detecting fraudulent insurance claims, in order to provide effective and efficient services for customers and keeping the insurance company stable enough in the market environment.

- Investigation of the effect of using a novel resampling technique (partitioning-undersampling), on fraud detection dataset, which mostly is imbalanced; showing the improvement of the IFD models recall.
- The experimental evidence and illustrations showing the effectiveness of the proposed IFD model, using ensemble combining classifiers; although the best model with the highest recall is IFDDT which was designed with base-classifier decision tree; but all other models on the list of best designed models are designed using ensemble combining classifiers.
- A novel comparison was conducted between base-classifier models and the ensemble-combining-classifiers models in order to investigate the appropriate algorithms for insurance fraud detection problem. This comparison is useful for a better understanding of the performance of various models and can be used to create effective models for insurance fraud detection model.

6.4 Future Research

1. The very first problem, which was faced by this research, was find a dataset, even publicly available datasets in fraud detection area is very rare and difficult to find. Designing a Sudanese fraud detection dataset is suggested for future research although there are several companies that have a fraud detection department already established but general refuse to cooperate.

2. In this research feature selection algorithms were not used so a future research experiment could be applying feature selection on dataset then apply the proposed resampling techniques and apply the proposed resampling techniques first and after that apply feature selection techniques on each subsample separately. Use those two different subsamples-forming methods and design IFD models and compare between them and the method applied in this research.

3. In order to evaluate these IFD models proposed in this research, those models should be implemented in real life dataset and compare their results with the insurance fraud detection technique used, if found.

7 REFERENCES

[1] E. Taghiloo, A. Azar, E. Nasiri, and H. Ghaedrahmati, "A New Model for Insurance Fraud Detection in Car Accidents Using a Combined Fuzzy DEMATEL and ELECTRE-TRI Approach," *International Journal of Business and Social Science,* vol. 3, 2012.

[2] M. Daniel, "Application of Data Mining Technology to Support Fraud Protection: The Case of Ethiopian Revenue and Custom Authority," Addis Ababa University, 2013.

[3] S. G. Ghezzi, "A private network of social control: Insurance investigation units," *Social Problems,* pp. 521-531, 1983.

[4] P. Brennan, "A comprehensive survey of methods for overcoming the class imbalance problem in fraud detection," *Institute of technology Blanchardstown Dublin, Ireland,* 2012.

[5] S. Viaene, R. A. Derrig, B. Baesens, and G. Dedene, "A Comparison of State-of-the-Art Classification Techniques for Expert Automobile Insurance Claim Fraud Detection," *Journal of Risk and Insurance,* vol. 69, pp. 373-421, 2002.

[6] P. Clifton, L. Vincent, S. Kate, and G. Ross, "A Comprehensive Survey of Data Mining-based Fraud Detection Research," *Artificial Intelligence Review,* vol. 2, p. 9, 2005.

[7] C. Phua, D. Alahakoon, and V. Lee, "Minority report in fraud detection: classification of skewed data," *ACM SIGKDD Explorations Newsletter,* vol. 6, pp. 50-59, 2004.

[8] N. Dodd, "Insurance claims fraud: Applying psychology to the reduction of insurance claims fraud," *Insurance Trends,* vol. 18, pp. 11-16, 1998.

[9] J. Pearsall, *The concise Oxford dictionary-/ed. by Judy Pearsall*: Oxford [etc.]: Oxford University Press, 1999.

[10] C. Phua, V. Lee, K. Smith, and R. Gayler, "A comprehensive survey of data mining-based fraud detection research," *arXiv preprint arXiv:1009.6119,* 2010.

[11] J.-H. Wang, Y.-L. Liao, T.-m. Tsai, and G. Hung, "Technology-based Financial Frauds in Taiwan: Issues and Approaches," in *SMC*, 2006, pp. 1120-1124.

[12] C. Elkan, "Magical thinking in data mining: lessons from CoIL challenge 2000," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 426-431.

[13] N. Lavrač, H. Motoda, T. Fawcett, R. Holte, P. Langley, and P. Adriaans, "Introduction: Lessons learned from data mining applications and collaborative problem solving," *Machine learning,* vol. 57, pp. 13-34, 2004.

[14] I. Bose and R. K. Mahapatra, "Business data mining—a machine learning perspective," *Information & management,* vol. 39, pp. 211-225, 2001.

[15] E. Turban, R. Sharda, D. Delen, and T. Efraim, *Decision support and business intelligence systems*: Pearson Education India, 2007.

[16] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus, "Knowledge discovery in databases: An overview," *AI magazine,* vol. 13, p. 57, 1992.

[17] Y. Kou, C.-T. Lu, S. Sirwongwattana, and Y.-P. Huang, "Survey of fraud detection techniques," in *Networking, sensing and control, 2004 IEEE international conference on*, 2004, pp. 749-754.

[18] E. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decision Support Systems,* vol. 50, pp. 559-569, 2011.

[19] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*: Morgan kaufmann, 2006.

[20] D. Zhang and L. Zhou, "Discovering golden nuggets: data mining in financial application," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 34, pp. 513-522, 2004.

[21] D. Yue, X. Wu, Y. Wang, Y. Li, and C.-H. Chu, "A review of data mining-based financial fraud detection research," in *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, 2007, pp. 5519-5522.

[22] S. R. Ahmed, "Applications of data mining in retail business," in *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, 2004, pp. 455-459.

[23] M. Agyemang, K. Barker, and R. Alhajj, "A comprehensive survey of numeric and symbolic outlier mining techniques," *Intelligent Data Analysis,* vol. 10, pp. 521-538, 2006.

[24] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne, "On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp. 320-324.

[25] S. G. Eick and D. E. Fyock, "Visualizing corporate data," *Potentials, IEEE,* vol. 15, pp. 6-11, 1996.

[26] A. K. I. Hassan and A. Abraham, "Computational Intelligence Models for Insurance Fraud Detection: A Review of a Decade of Research," *Journal of Network and Innovative Computing,* vol. 1, pp. 341-347, 2013.

[27] S. Viaene, R. A. Derrig, and G. Dedene, "A case study of applying boosting Naive Bayes to claim fraud diagnosis," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 16, pp. 612-620, 2004.

[28] S. Viaene, G. Dedene, and R. A. Derrig, "Auto claim fraud detection using Bayesian learning neural networks," *Expert Systems with Applications,* vol. 29, pp. 653-666, 2005.

[29] M. Artís, M. Ayuso, and M. Guillén, "Modelling different types of automobile insurance fraud behaviour in the Spanish market," *Insurance: Mathematics and Economics,* vol. 24, pp. 67-81, 1999.

[30] M. Vasu and V. Ravi, "A hybrid under-sampling approach for mining unbalanced datasets: applications to banking and insurance," *International Journal of Data Mining, Modelling and Management,* vol. 3, pp. 75-105, 2011.

[31] M. Farquad, V. Ravi, and S. B. Raju, "Analytical CRM in banking and finance using SVM: a modified active learning–based rule extraction approach," *International Journal of Electronic Customer Relationship Management,* vol. 6, pp. 48-73, 2012.

[32] J. M. Pérez, J. Muguerza, O. Arbelaitz, I. Gurrutxaga, and J. I. Martín, "Consolidated tree classifier learning in a car insurance fraud detection domain with class imbalance," in *Pattern Recognition and Data Mining*, ed: Springer, 2005, pp. 381-389.

[33] R. Bhowmik, "Detecting auto insurance fraud by data mining techniques," *Journal of Emerging Trends in Computing and Information Sciences,* vol. 2, pp. 156-162, 2011.

[34] W. Xu, S. Wang, D. Zhang, and B. Yang, "Random Rough Subspace Based Neural Network Ensemble for Insurance Fraud Detection," in *Computational Sciences and Optimization (CSO), 2011 Fourth International Joint Conference on*, 2011, pp. 1276-1280.

[35] P. L. Brockett, R. A. Derrig, L. L. Golden, A. Levine, and M. Alpert, "Fraud classification using principal component analysis of RIDITs," *Journal of Risk and Insurance,* vol. 69, pp. 341-371, 2002.

[36] S. Viaene, M. Ayuso, M. Guillen, D. Van Gheel, and G. Dedene, "Strategies for detecting fraudulent claims in the automobile insurance industry," *European Journal of Operational Research,* vol. 176, pp. 565-583, 2007.

[37] M. Artís, M. Ayuso, and M. Guillén, "Detection of automobile insurance fraud with discrete choice models and misclassified claims," *Journal of Risk and Insurance,* vol. 69, pp. 325-340, 2002.

[38] S. B. Caudill, M. Ayuso, and M. Guillen, "Fraud detection using a multinomial logit model with missing information," *Journal of Risk and Insurance,* vol. 72, pp. 539-550, 2005.

[39] J. Pathak, N. Vidyarthi, and S. L. Summers, "A fuzzy-based algorithm for auditors to detect elements of fraud in settled insurance claims," *Managerial Auditing Journal,* vol. 20, pp. 632-644, 2005.

[40] L. Bermúdez, J. Pérez, M. Ayuso, E. Gómez, and F. Vázquez, "A Bayesian dichotomous model with asymmetric link for fraud in insurance," *Insurance: Mathematics and Economics,* vol. 42, pp. 779-786, 2008.

[41] P. L. Brockett, X. Xia, and R. A. Derrig, "Using Kohonen's self-organizing feature map to uncover automobile bodily injury claims fraud," *Journal of Risk and Insurance,* pp. 245-274, 1998.

[42] E. Taghiloo, A. Azar, E. Nasiri, and H. Ghaedrahmati, "A New Model for Insurance Fraud Detection in Car Accidents Using a Combined Fuzzy DEMATEL and ELECTRE-TRI Approach."

[43] L. Guelman, "Gradient boosting trees for auto insurance loss cost modeling and prediction," *Expert Systems with Applications,* vol. 39, pp. 3659-3667, 2012.

[44] M. Sternberg and R. G. Reynolds, "Using cultural algorithms to support re-engineering of rule-based expert systems in dynamic performance environments: a case study in fraud detection," *Evolutionary Computation, IEEE Transactions on,* vol. 1, pp. 225-243, 1997.

[45] L. Šubelj, Š. Furlan, and M. Bajec, "An expert system for detecting automobile insurance fraud using social network analysis," *Expert Systems with Applications,* vol. 38, pp. 1039-1052, 2011.

[46] S. Tennyson and P. Salsas-Forn, "Claims auditing in automobile insurance: fraud detection and deterrence objectives," *Journal of Risk and Insurance,* vol. 69, pp. 289-308, 2002.

[47] E. B. Belhadji, G. Dionne, and F. Tarkhani, "A model for the detection of insurance fraud," *Geneva Papers on Risk and Insurance. Issues and Practice,* pp. 517-538, 2000.

[48] K. J. Crocker and S. Tennyson, "Insurance fraud and optimal claims settlement strategies," *JL & Econ.,* vol. 45, p. 469, 2002.

[49] J. Pinquet, M. Ayuso, and M. Guillen, "Selection bias and auditing policies for insurance claims," *Journal of Risk and Insurance,* vol. 74, pp. 425-440, 2007.

[50] H. I. Weisberg and R. A. Derrig, "Quantitative methods for detecting fraudulent automobile bodily injury claims," *Risques,* vol. 35, pp. 75-101, 1998.

[51] J. A. Atwood, J. F. Robison-Cox, and S. Shaik, "Estimating the prevalence and cost of yield-switching fraud in the federal crop insurance program," *American journal of agricultural economics,* vol. 88, pp. 365-381, 2006.

[52] Y. Jin, R. M. Rejesus*, and B. B. Little, "Binary choice models for rare events data: a crop insurance fraud application," *Applied Economics,* vol. 37, pp. 841-848, 2005.

[53] M. Kirlidog and C. Asuk, "A Fraud Detection Approach with Data Mining in Health Insurance," *Procedia-Social and Behavioral Sciences,* vol. 62, pp. 989-994, 2012.

[54] Y. Peng, G. Kou, A. Sabatka, J. Matza, Z. Chen, D. Khazanchi, and Y. Shi, "Application of classification methods to individual disability income insurance fraud detection," in *Computational Science–ICCS 2007*, ed: Springer, 2007, pp. 852-858.

[55] J. A. Major and D. R. Riedinger, "EFD: A Hybrid Knowledge/Statistical-Based System for the Detection of Fraud," *Journal of Risk and Insurance,* vol. 69, pp. 309-324, 2002.

[56] H. He, J. Wang, W. Graco, and S. Hawkins, "Application of neural networks to detection of medical fraud," *Expert Systems with Applications,* vol. 13, pp. 329-336, 1997.

[57] W.-S. Yang and S.-Y. Hwang, "A process-mining framework for the detection of healthcare fraud and abuse," *Expert Systems with Applications,* vol. 31, pp. 56-68, 2006.

[58] K. C. Lin, C. L. Yeh, and S. Y. Huang, "Use of Data Mining Techniques to Detect Medical Fraud in Health Insurance," *Applied Mechanics and Materials,* vol. 284, pp. 1574-1578, 2013.

[59] C. Francis, N. Pepper, and H. Strong, "Using support vector machines to detect medical fraud and abuse," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, 2011, pp. 8291-8294.

[60] Y. Peng, G. Kou, A. Sabatka, Z. Chen, D. Khazanchi, and Y. Shi, "Application of Clustering Methods to Health Insurance Fraud Detection," in *Service Systems and Service Management, 2006 International Conference on*, 2006, pp. 116-120.

[61] S. Zhu, Y. Wang, and Y. Wu, "Health care fraud detection using nonnegative matrix factorization," in *Computer Science & Education (ICCSE), 2011 6th International Conference on*, 2011, pp. 499-503.

[62] R. M. Musal, "Two models to investigate Medicare fraud within unsupervised databases," *Expert Systems with Applications,* vol. 37, pp. 8628-8633, 2010.

[63] H. Shin, H. Park, J. Lee, and W. C. Jhee, "A scoring model to detect abusive billing patterns in health insurance claims," *Expert Systems with Applications,* vol. 39, pp. 7441-7450, 2012.

[64] R. M. Konijn and W. Kowalczyk, "Finding fraud in health insurance data with two-layer outlier detection approach," in *Data Warehousing and Knowledge Discovery*, ed: Springer, 2011, pp. 394-405.

[65] K. D. Aral, H. A. Güvenir, İ. Sabuncuoğlu, and A. R. Akar, "A prescription fraud detection model," *Computer methods and programs in biomedicine,* vol. 106, pp. 37-46, 2012.

[66] W. Xiaoyun and L. Danyue, "Hybrid outlier mining algorithm based evaluation of client moral risk in insurance company," in *Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on*, 2010, pp. 585-589.

[67] V. Chandola, S. R. Sukumar, and J. C. Schryver, "Knowledge discovery from massive healthcare claims data," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 1312-1320.

[68] L. Sokol, B. Garcia, M. West, J. Rodriguez, and K. Johnson, "Precursory steps to mining HCFA health care claims," in *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, 2001, p. 10 pp.

[69] D. Tomar and S. Agarwal, "A survey on Data Mining approaches for Healthcare," *International Journal of Bio-Science and Bio-Technology,* vol. 5, pp. 241-266, 2013.

[70] M. Silver, T. Sakata, H.-C. Su, C. Herman, S. B. Dolins, and M. J. O Shea, "Case study: how to apply data mining techniques in a healthcare data warehouse," *Journal of healthcare information management,* vol. 15, pp. 155-164, 2001.

[71] H. D. Margaret, "Data mining introductory and advanced topics," *Pearsons Education Inc,* 2003.

[72] J. H. W. Adrian Gepp, Kuldeep Kumar and and S. Bhattacharya, "A Comparative Analysis of Decision Trees Vis-a-vis Other Computational Data Mining Techniques in Automotive Insurance Fraud Detection," *Journal of Data Science,* vol. 10, pp. 537-561, 2012.

[73] E.-J. Phillips, Peter Phillips, and Mark Hurrell. , "Optimising Insurance Fraud Detection and Classification of Vehicle Accident Damange by Using Neural Networks to Identify Patterns in Behaviour, Linked Cases and Vehicle Recognition.," presented at the The Third International Conference on Digital Information Processing and Communications (ICDIPC2013), 2013.

[74] S. G. V. Kathiresan, Faseela V. S, "Health Insurance Claim Fraud Detection: A Survey," *International Journal of Latest Trends in Engineering and Technology (IJLTET),* vol. 5, pp. 43-51, 2015.

[75] J. B. a. JinhwaKim, "A Personal Credit Rating Prediction Model Using Data Mining in Smart Ubiquitous Environments," *International Journal of Distributed Sensor Networks,* 2015.

[76] H. Joudaki, A. Rashidian, B. Minaei-Bidgoli, M. Mahmoodi, B. Geraili, M. Nasiri, and M. Arab, "Using data mining to detect health care fraud and abuse: a review of literature," *Glob J Health Sci,* vol. 7, pp. 194-202, Jan 2015.

[77] V. N. Vapnik and V. Vapnik, *Statistical learning theory* vol. 1: Wiley New York, 1998.

[78] V. Vapnik, "The support vector method of function estimation," in *Nonlinear Modeling*, ed: Springer, 1998, pp. 55-85.

[79] L. Xu and M.-Y. Chow, "A classification approach for power distribution systems fault cause identification," *Power Systems, IEEE Transactions on,* vol. 21, pp. 53-60, 2006.

[80] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis,* vol. 6, pp. 429-449, 2002.

[81] C. X. Ling and C. Li, "Data Mining for Direct Marketing: Problems and Solutions," in *KDD*, 1998, pp. 73-79.

[82] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 18, pp. 63-77, 2006.

[83] N. Japkowicz, "Learning from imbalanced data sets: a comparison of various strategies," in *AAAI workshop on learning from imbalanced data sets*, 2000, pp. 10-15.

[84] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. of the Int'l Conf. on Artificial Intelligence*, 2000.

[85] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine learning,* vol. 30, pp. 195-215, 1998.

[86] T. Razzaghi, "Cost-Sensitive Learning-Based Methods for Imbalanced Classification Problems with Applications," University of Central Florida Orlando, Florida, 2014.

[87] P. Domingos, "Metacost: A general method for making classifiers cost-sensitive," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1999, pp. 155-164.

[88] L. Rokach, "Ensemble Methods for Classifiers," in *Data Mining and Knowledge Discovery Handbook*, ed: Springer US, 2010, pp. 957-980.

[89] P. K. Chan and S. J. Stolfo, "A comparative evaluation of voting and meta-learning on partitioned data," in *ICML*, 1995, pp. 90-98.

[90] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*: CRC Press, 2012.

[91] B. Clarke, "Comparing Bayes model averaging and stacking when model approximation error cannot be ignored," *The Journal of Machine Learning Research,* vol. 4, pp. 683-712, 2003.

[92] J. J. Rodríguez, J. F. Díez-Pastor, Á. Arnaiz-González, and C. García-Osorio, "An Experimental Study on Combining Binarization Techniques and Ensemble Methods of Decision Trees," in *Multiple Classifier Systems*, ed: Springer, 2015, pp. 181-193.

[93] A. K. Seewald and J. Fürnkranz, "An evaluation of grading classifiers," in *Advances in Intelligent Data Analysis*, ed: Springer, 2001, pp. 115-124.

[94] C. Schaffer, "Selecting a classification method by cross-validation," *Machine learning,* vol. 13, pp. 135-143, 1993.

[95] G. G. Sundarkumar and V. Ravi, "A novel hybrid undersampling method for mining unbalanced datasets in banking and insurance," *Engineering Applications of Artificial Intelligence,* vol. 37, pp. 368-377, 2015.

[96] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter,* vol. 11, pp. 10-18, 2009.

[97] G. Holmes, A. Donkin, and I. H. Witten, "Weka: A machine learning workbench," in *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, 1994, pp. 357-361.

[98] A. Jovic, K. Brkic, and N. Bogunovic, "An overview of free software tools for general data mining," in *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on*, 2014, pp. 1112-1117.

[99] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann, 2005.

[100] P. E. Utgoff, N. C. Berkman, and J. A. Clouse, "Decision tree induction based on efficient tree restructuring," *Machine learning,* vol. 29, pp. 5-44, 1997.

[101] M. Ankerst, C. Elsen, M. Ester, and H.-P. Kriegel, "Visual classification: an interactive approach to decision tree construction," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 1999, pp. 392-396.

[102] M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han, "Generalization and decision tree induction: efficient classification in data mining," in *Research Issues in Data Engineering, 1997. Proceedings. Seventh International Workshop on*, 1997, pp. 111-120.

[103] J. K. Martin and D. S. Hirschberg, *The time complexity of decision tree induction*: Citeseer, 1995.

[104] D. E. Brown, V. Corruble, and C. L. Pittard, "A comparison of decision tree classifiers with backpropagation neural networks for multimodal classification problems," *Pattern Recognition,* vol. 26, pp. 953-961, 1993.

[105] J. R. Quinlan, "Learning logical definitions from relations," *Machine learning,* vol. 5, pp. 239-266, 1990.

[106] E. B. Hunt, J. Marin, and P. J. Stone, "Experiments in induction," 1966.

[107] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*: CRC press, 1984.

[108] G. G. Towell and J. W. Shavlik, "Extracting refined rules from knowledge-based neural networks," *Machine learning,* vol. 13, pp. 71-101, 1993.

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

[110] J. Gehrke, V. Ganti, R. Ramakrishnan, and W.-Y. Loh, "BOAT—optimistic decision tree construction," in *ACM SIGMOD Record*, 1999, pp. 169-180.

[111] M. W. Craven and J. W. Shavlik, "Using neural networks for data mining," *Future generation computer systems,* vol. 13, pp. 211-229, 1997.

[112] S. I. Gallant, *Neural network learning and expert systems*: MIT press, 1993.

[113] S. Haykin and N. Network, "A comprehensive foundation," *Neural networks,* vol. 2, 2004.

[114] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the theory of neural computation* vol. 1: Basic Books, 1991.

[115] D. S. Touretzky, *Advances in Neural Information Processing Systems 8: Proceedings of the 1995 Conference* vol. 8: Mit Press, 1996.

[116] D.-Y. Yeung, "A neural network approach to constructive induction," in *Machine Learning Proc. of the 8th Int. Workshop*, 2014, p. 228.

[117] S. Haykin, "Multilayer perceptrons," *Neural Networks: A Comprehensive Foundation,* vol. 2, pp. 156-255, 1999.

[118] M. W. Craven and J. W. Shavlik, "Learning symbolic rules using artificial neural networks," in *Proceedings of the Tenth International Conference on Machine Learning*, 2014, pp. 73-80.

[119] R. R. Yager and L. A. Zadeh, *Fuzzy sets, neural networks and soft computing*: John Wiley & Sons, Inc., 1994.

[120] R. Patidar and L. Sharma, "Credit card fraud detection using neural network," *International Journal of Soft Computing and Engineering (IJSCE),* vol. 1, 2011.

[121] B. Widrow, D. E. Rumelhart, and M. A. Lehr, "Neural networks: Applications in industry, business and science," *Communications of the ACM,* vol. 37, pp. 93-105, 1994.

[122] E. Aleskerov, B. Freisleben, and B. Rao, "Cardwatch: A neural network based database mining system for credit card fraud detection," in *Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997*, 1997, pp. 220-226.

[123] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144-152.

[124] J. Platt, "Fast training of support vector machines using sequential minimal optimization," *Advances in kernel methods—support vector learning,* vol. 3, 1999.

[125] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning,* vol. 20, pp. 273-297, 1995.

[126] A. Kulkarni, V. K. Jayaraman, and B. D. Kulkarni, "Support vector classification with parameter tuning assisted by agent-based technique," *Computers & chemical engineering,* vol. 28, pp. 311-318, 2004.

[127] S. R. Gunn, "Support vector machines for classification and regression," *ISIS technical report,* vol. 14, 1998.

[128] B. Schölkopf and A. J. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*: MIT press, 2002.

[129] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing,* vol. 14, pp. 199-222, 2004.

[130] C. Campbell and Y. Ying, "Learning with support vector machines," *Synthesis Lectures on Artificial Intelligence and Machine Learning,* vol. 5, pp. 1-95, 2011.

[131] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 2, p. 27, 2011.

[132] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recognition,* vol. 44, pp. 1761-1776, 2011.

[133] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Machine learning,* vol. 36, pp. 105-139, 1999.

[134] J. Cao, S. Kwong, R. Wang, X. Li, K. Li, and X. Kong, "Class-specific soft voting based multiple extreme learning machines ensemble," *Neurocomputing,* vol. 149, pp. 275-284, 2015.

[135] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective voting of heterogeneous classifiers," in *Machine Learning: ECML 2004*, ed: Springer, 2004, pp. 465-476.

[136] D. Carroll, J. Dolmas, and E. Young, "Majority Voting: A Quantitative Investigation," 2014.

[137] F. Markatopoulou, G. Tsoumakas, and I. Vlahavas, "Dynamic ensemble pruning based on multi-label classification," *Neurocomputing,* vol. 150, pp. 501-512, 2015.

[138] L. Breiman, "Bagging predictors," *Machine learning,* vol. 24, pp. 123-140, 1996.

[139] P. Smyth and D. Wolpert, "Stacked density estimation," *Advances in Neural Information Processing Systems,* pp. 668-674, 1998.

[140] D. H. Wolpert, "Stacked generalization," *Neural networks,* vol. 5, pp. 241-259, 1992.

[141] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, "Machine learning, neural and statistical classification," 1994.

[142] H. F. Jelinek, J. H. Abawajy, A. V. Kelarev, M. U. Chowdhury, and A. Stranieri, "Decision trees and multi-level ensemble classifiers for neurological diagnostics," *AIMS Medical Science,* vol. 1, pp. 1-12, 2014.

[143] C. Van Rijsbergen, "Information Retrieval. Dept. of Computer Science, University of Glasgow," *URL: citeseer. ist. psu. edu/vanrijsbergen79information. html,* 1979.

[144] H. Shi, "Best-first decision tree learning," Citeseer, 2007.

[145] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics,* vol. 28, pp. 337-407, 2000.

[146] P. M. Caldwell, C. S. Bretherton, M. D. Zelinka, S. A. Klein, B. D. Santer, and B. M. Sanderson, "Statistical significance of climate sensitivity predictors obtained by data mining," *Geophysical Research Letters,* vol. 41, pp. 1803-1808, 2014.

[147] B. Schölkopf, C. J. Burges, and A. J. Smola, *Advances in kernel methods: support vector learning*: MIT press, 1999.

[148] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," *Neural Computation,* vol. 13, pp. 637-649, 2001.

[149] T. Hastie and R. Tibshirani, "Classification by pairwise coupling," *The annals of statistics,* vol. 26, pp. 451-471, 1998.

[150] B. Üstün, W. J. Melssen, and L. M. Buydens, "Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel," *Chemometrics and Intelligent Laboratory Systems,* vol. 81, pp. 29-40, 2006.

[151] A. K. I. Hassan and A. Abraham, "Modeling consumer loan default prediction using neural netware," in *Computing, Electrical and Electronics Engineering (ICCEEE), 2013 International Conference on*, 2013, pp. 239-243.

[152] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine learning,* vol. 11, pp. 63-90, 1993.

[153] A. K. Seewald, "How to make stacking better and faster while also taking care of an unknown weakness," in *Proceedings of the nineteenth international conference on machine learning*, 2002, pp. 554-561.

[154] L. I. Kuncheva, *Combining pattern classifiers: methods and algorithms*: John Wiley & Sons, 2004.

[155] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 20, pp. 226-239, 1998.

[156] P. J. Bickel and K. A. Doksum, *Mathematical Statistics: Basic Ideas and Selected Topics, volume I* vol. 117: CRC Press, 2015.