REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ MATHEMATIQUES ET INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité : [Génie Logiciel]

Par :

[BACHA Soufiane]

Sur le thème

Imbalanced datasets :Towards a better classification using boosting methods

---

Soutenu publiquement le 20/ 08 / 2021 à Tiaret devant le jury composé de :

Mr BOUDAA Boudjemaa	Grade	MCB Université Ibn Khaldoun -Tiaret -	Président
Mr DJAFRI Laouni	Grade	MCB Université Ibn Khaldoun -Tiaret -	Encadreur
Mr KOUADRIA Abderrahmane	Grade	MAA Université Ibn Khaldoun -Tiaret -	Examinateur

2021-2022

Acknowledgements

I would like to thank my supervisor **Dr. DJAFRI Laouni**. for his consistent support and guidance during the running of this project. I also wish to thank all the people whose assistance was a milestone in the completion of this project. I wish to acknowledge the support and great love of my family; especially my mother. They kept me going on and this work would not have been possible without their input.

Abstract

Imbalanced datasets classification is inherently difficult. This situation becomes a challenge when amounts of data are processed to extract knowledge because traditional learning models fail to generate required results due to imbalanced nature of data. In this thesis, we will address the problem of imbalanced datasets whether at the class level, or at the classifier level. In our work, we are interested in binary classification. To do this, we present a set of techniques used to solve this problem in particular boosting methods and machine learning algorithms. Our goal is therefore to rebalance the dataset at the class level and to find an optimal classifier to handle these datasets after balancing. Through the obtained results, it was observed that the boosting methods are well suitable to rebalance the data and thus give a very satisfactory classification result.

Keywords: Imbalanced Datasets; Supervised Classification; Boosting approach;Data Sampling Approach; SMOTEBoost Algorithm; RUSBoost Algorithm.

Resumé

La classification des ensembles de données déséquilibrés est intrinsèquement difficile. Cette situation devient un défi lorsque des quantités de données sont traitées pour extraire des connaissances, car les modèles d'apprentissage traditionnels ne parviennent pas à générer les résultats requis en raison de la nature déséquilibrée des données. Dans cet article, nous aborderons le problème des jeux de données déséquilibrés que ce soit au niveau de la classe, ou au niveau du classificateur. Dans notre travail, nous nous intéressons à la classification binaire. Pour ce faire, nous présentons un ensemble de techniques utilisées pour résoudre ce problème en particulier des méthodes de boosting et des algorithmes d'apprentissage automatique. Notre objectif est donc de rééquilibrer l'ensemble de données au niveau de la classe et de trouver un classifieur optimal pour gérer ces ensembles de données après équilibrage. A travers les résultats obtenus, il a été observé que les méthodes de boosting sont bien adaptées pour rééquilibrer les données et ainsi donner un résultat de classification très satisfaisant.

Mots clés: Ensembles de données déséquilibrés ; Classification supervisée ; Methodes de Boosting ; Echantillonnage de données ; Algorithme de SMOTBoost ; Algorithme de RUSBoost.

Contents

List of Figures

List of Tables

Nomenclature

ADASYN	Algorithm of Adaptive Synthetic Sampling Approach.
AUC	Area under the ROC curve.
CNN	Algorithm of Condensed Nearest Neighbor.
CPM	Algorithm of Class Purity Maximization.
ENN	Algorithm of Edited Nearest Neighbors Rule.
NCL	Algorithm of Neighborhood Cleaning Rule.
OSS	Algorithm of One-Sided Selection.
ROC	Receiver operating characteristic curve.
ROS	Technique of random Overdersampling.
RUS	Technique of random Undersampling.
SMOTE	Algorithm of Synthetic Minority Oversampling Technique.
TL	Algorithm of Tomek Links.
US-CNN	Algorithm of Condensed Nearest Neighbor.
US-CNN + TL	The combination of algorithm Condensed Nearest Neighbor Rule and Tomek Link.

Chapter 1

General Introduction

1.1 Introduction

In Machine Learning, the construction of predictive classification models is a very useful and recursive tool for countless application scenarios, in which you want to predict certain behaviors or classify a series of inputs made up of different variables.

One of the procedures to provide knowledge to a system is through supervised learning algorithms, where already know data records are exploited to guide the training of the classifier. Specifically, when we talk about predictive classification, we have data for which we already know its class or output target, and it is based on this information that the system is able to obtain said knowledge to classify new data inputs.

The construction of a good classification model implies, after training, evaluating the system for new entries, and observing the result obtained with respect to the true class that was recorded in the original training set. Based on this information, the performance of the system is calculated, with which we can be sure that we have obtained an effective and accurate classification.

The characteristics of the training set used to learn the classifier are decisive to correctly guide the learning. This can be altered if there are certain anomalies regarding the structure of the training data. In fact, real-world data collection used in machine learning prediction can suffer from many anomalies that restrict the data structure and thus degrade leaner performance. The most common anomalies in data when distribution of the classes is not balanced or as well-know ”Imbalanced data problem ”.

The class imbalance occurs when the number of examples representing one class is much lower than the ones of the other classes. Hence, one or more classes may be underrepresented in the dataset.

1.2 Motivation

As mentioned in the previous part, imbalanced data(datasets) is data with unequal class distribution. This definition has attracted a lot of attention from researchers and practitioners due to the number of real-world applications. For instance, applications known to suffer from this problem are bug diagnosis(Yang, 2009), anomaly detection (Khreich, 2010), medical diagnosis (Mazurowski, 2008)(Cancer prevalence is particularly low among screening populations which results in class imbalance in the collected set of examples), e-mail foldering(Bermejo, 2011), face recognition(Liu, (2005, October)) or detection of oil spills (Kubat, 1998), among others. This bring us to some very importants induction related to our search and motivated us to study and address largely this issues "imblanced datasets" :

1. any data scientist cannot be ignore the problem that the data collection for the first time from real world used during the data training model suffers from unequal distribution of data.

2. Any beginner in data science he think that the achievment of higher accuracy(99%) after apply machine learning model on testing data is the purpose, and this result can lead us to say this model is work fine. In contrast, these are erroneous conclusions, and a clear example of this, in medical field: where we should differentiate between benign(negative class) and malign tumours(positive class) of a specific type of cancer using two different features that have been measured after a biopsy/. In this case, it is much more important to correctly identify malign tumours (positive class)than benign ones, since the consequences of undetected malign tumours can be fatal, whereas a false positive when the tumour is benign would not be as harmful. Actually, Standard classifier learning algorithms are usually biased towards the Majority class, for example, in case of rare disease, as can be expected, we have much more information and cases about healthy people(negative class) than sick people(positive class). In this context, if we train the classification model without taking into account these characteristics of the data that we have at our disposal, it is most likely that we will obtain a robust and effective system when it comes to detecting that a person does not suffer from the rare disease(negative class). However, the model will not be so good and will make more mistakes when diagnosing positive cases towards said pathology, when this is, really, what is usually most interested in detecting.

1.3 A brief about imbalanced datasets

Most of the well-known traditional machine learning techniques are designed to solve classification problems showing reasonably balanced class distributions. However, this assumption does not always hold in reality. Occasionally, skewed class distributions of dataset is present in many real-world classification datasets. Obviously, any dataset with an unequal class distribution is technically called imbalanced datasets(Ortigosa-Hernández, 2016). Class imbalance occurs when the class distributions are not balanced. If the positive class has significantly lower samples than the negative class the classifier accuracy for positive test samples decreases. This problem with imbalance in the class distribution became more pronounced with the applications of the machine learning algorithms to the real world. These applications range from telecommunications management(Ezawa, 1996), bioinformatics(Radivojac, 2004), text classification (Grobelnik, 1999)(Lewis, 1994), speech recognition, to detection of oil spills in satellite images (Kubat, 1998). The problem of class imbalanced has received attention from machine learning and data dining community in form of Workshops (Japkowicz, 2000)and Special Issues(Chawla, 2004). Several research papers in literature discussed the differents approaches that used to deal with the issue of class imbalanced that faced by the data mining community. Sampling /Undersampling methodologies „ oversampling with synthetic generation of new samples continue to be popular in the research work. The research continues to evolve with different applications, as each application provides a compelling problem. The traditionally evaluation criteria for machine learning models become unuseful anymore. The ROC curves soon emerged as a popular choice.

In addition, the imbalanced datasets can be described as more than unequal of the disribution of classes in Real-world. Obviously, it can be related by anothers characteristics, that we will presented in details in chapter Three such as the distribution of data within each class small disjuncts, ovelapping datasets shift...etc.

1.4 Objectives

The aim of this study is to check whether the SMOTEBoost and RUSBoost algorithms that we have developed (parameter adjustment) are able to obtain good performance and provide an effective solution to problems with imbalanced data sets, where we have studied these algorithms independently to verify the real contribution of each of them to solve our problem. Finally, the results obtained will be compared with those obtained with other models already known in literature in order to solve imbalanced datasets problems with sets of imbalanced data.

1.5 Research Questions

The goal of the our research is to answer the following question: Does solving imbalanced dataset problem (at the class level or at the classifer level) improve classification results ?

This question investigates if solving the class imbalance problem will improve the classification results.

1.6 Approach

This section describes the approach that was taken to answer the research questions. The approach for each research question is then converted into an implementation to arrive at the results.

Does solving class imbalance problem improve classification results ?

To solve the classification imbalanced probelm we applied SMOTEBoost [Chawla, et. al., 2003] and RUSBoost[Liu,2005] algorithms to the KEEL datasets.

1.7 Research Scope

Nowadays, researcher's interest in the field of imbalanced data has increased more and more. The scope of our study relates to the work carried out in recent years that focuses on the problem of class imbalance and the resulting solutions developed by researchers. Moreover, in the interest of our focus on binary class, we consider related work that analyze class imbalance with several datasets with defferents IR(Imbalance ratio) and unequal instances between this datasets. In addition to analyzing the servey papers, we also provide our own insights into proposed methods in current research in the area and discuss avenues for future work for the community. To the best of our knowledge, we have included all published articles that fall within our master thesis study's scope. We believe such a large-scale survey of works addressing, and developing solutions for, class imbalance problems in datasets is unique in the data mining and machine learning domain.

1.8 Thesis structure

We have structured our master thesis according to the plan described below:

Chapitre 1: general introduction

The first chapter describes in general, our proposed subject related to imbalance dataset problems, as well as the solutions that will be discussed in this context.

From an organizational point of view, the rest of the masterś thesis is structured around three chapters.

Chapter 2: Data mining and Machine learning.

This chapter describe two famous concepts in data science domains,data mining and machine learning concept including the data mining steps and his application domains. Finally we end this chapter with a relation between this two concepts.

Chapter 3: Classification approaches and techniques for imbalanced datasets.

In this chapter, we will take an in-depth look at the field of imbalanced datasets problem, we will largely cover the approaches that used to deal with this problem.

Chapter 4: Realization and Implementation.

In this chapter, we will present the work that we have done as well as the results obtained. We end this chapter with a synthesis of comparison of the approaches implemented.

Finaly, will end our master thesis with a general conclusion and perspectives opened up by the work presented in this project.

Chapter 2

Data Mining and Machine Learning

2.1 Introduction

Recent technological advances imply that the capacities to generate and store data are increased everyday. Among the factors that influence this reality is the automation of all type of transactions (commercial, business, economic, scientific). In other hand, the Internet has rapid access to information. Besides, the evolution of mass storage devices (in relation to price – storage capacity), such as hard disks that can store gigabytes of information at a reduced price, has led to companies and organizations to store all kinds of information(data of customers, patients, markets data). The amount of data that was stored began to grow and, although the tools to perform the data management was suitable, the significant relationships existing between them, began to surpass the human capacities for analysis. At the same time, database systems had begun to be decentralized, hence the decisions lacked credibility, inefficiency and lack of productivity.

The emergence of a new field of research called KDD(Mining, 2006) (Piatetsky-Shapiro, 1996) denoted to the process of discovering the knowledge from data. In fact, other names have been used in literature for this same concept such as: Knowledge Discovery, Data Discovery, Discovery Information, Knowledge Extraction, Data Extraction, Patter Discovery, DM, Data Science. This KDD field contains a accomplished process(phases) to extract information from storage data and to optimize the results called"data mining(KDD) process".

In order to better understand the nature of Data Science, this chapter is organized as follows. Sections 2.2) Data Science Problems. Section 2.3) Data Mining. Section 2.4) Kind of Data Mining. Section 2.5) Data mining perspectives. Section 2.6) Kind of applications data mining targeted. Section 2.7) Phases of Data Mining. Section 2.8) Data Classification Algorithms. Section 2.9) Machine Learning.

2.2 Data Science Problems

The modeling step is the central stage of the discovery process in which knowledge extraction algorithms are applied to previously preprocessed data. In fact, this step is inseparable from the next step in the chain of results analysis. In fact, often the analysis of the results obtained causes that it goes back again to the preprocessing phase in order to obtain more data or more attributes. In order for the process to be correct, it is necessary that the analyst has the preprocessed data set, the corresponding metadata and all the data information that has been previously extracted in the previous steps of the analysis. Although it is difficult to establish a classification of the possible complications we can find in Data Science, it is even more difficult to find a procedure that establishes the algorithm suitable for each type of problem. The chosen approach depends on the problem to be solved (goal) and the type of data that we are dealing with in each moment.

Generally, we can find two groups of problems in data science literature: **Standard Data Science Problems** and **non-Standard Data Science Problems**. Each group contains techniques to deal with it as shown in Figure 2.1 In this chapter, we will address on the first type of problem. In chapter three, we will take the second type of problem.

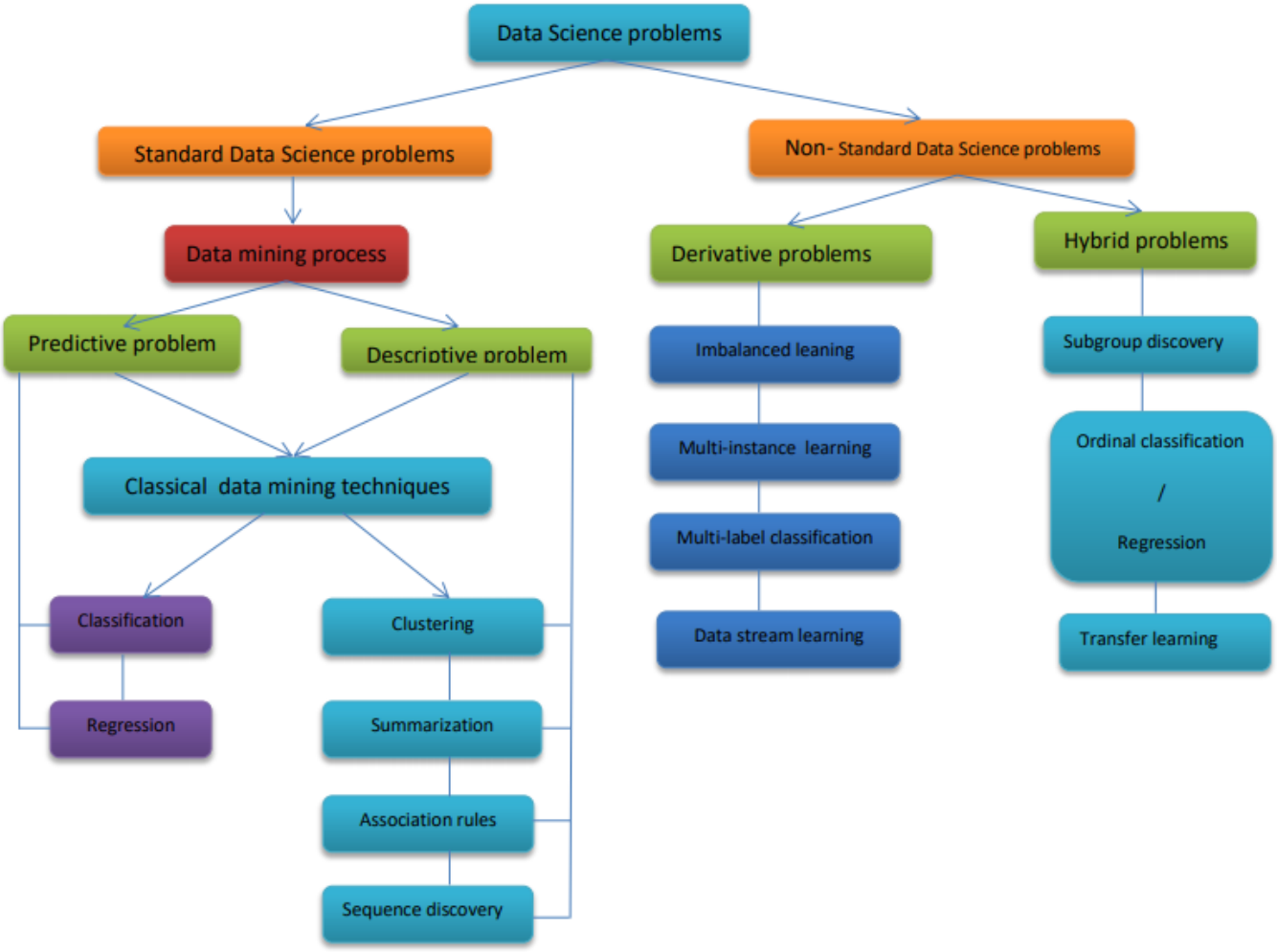

FIGURE 2.1: Data science problems.

Derivative problems

1. **Imbalanced learning**

   The presence of this problem in many real-world applications has increased researcher's interest. This problem occurs when one class(minority) is smaller than the ones from other classes(majority). This topic represent our thesis and we will address it in details.

2. **Multi-instance learning**

   This problem assumes an extension based on constraints imposed on models in which each instance is a bag of instances rather than an instance alone. The assignment of labels is done at the stock exchange and at the individual level. There are two main ways to solve this problem, whether converting multiple instances to single instances by data transformations or by updating single case algorithms.

3. **Multi-label classification**

   It is a generalization of the traditional classification, in which each processed instance is is not associated with a single class, but at the same time a subset of classes(Herrera, 2016). During the last years, various techniques have appeared that try, by transforming data or adapting classical algorithms, to provide a solution to this problem.

4. **Data stream learning**

   In some circumstances, not all data is available at a particular time, so it is necessary to develop learning algorithms that treat the input as a continuous data stream (Gama, 2010). The basis of this problem is to assume that each instance can be inspected only once and then must be discarded to make room for incoming instances. This problem is an extension in the way of processing the data in online mode and is oriented for both descriptive and predictive problems.

Hybrid Problems

1. **Semi-supervised learning**

   This approach emerges as a hybrid between the task of predictive classification and descriptive analysis based on clustering (Zhu, 2007). Here, the model design is performed considering both labeled and unlabeled data. Mainly, developments in this field use unlabeled samples to alter or rechange the formulation obtained from the labeled samples. Both the semi-supervised classification and the semi-supervised clustering emerged from the traditional perspectives including unlabeled or supervised examples, respectively.

2. **Subgroup Discovery**

   It is formed as the result of another hybridization between predictive and descriptive tasks, namely between classification and association of

patterns (Herrera, 2011). A method of subgroup discovery aims to extract interesting rules focused on the examples that belong to a particular class. Other terms denoting this problem are Contrast Set Mining and Emergent Pattern Mining.

3. **Ordinal classification/Regression**

   In this context, labels present an ordering relation according to the variable meaning (Gutiérrez, 2015). For example, financial trading could be assisted by ordinal classification techniques to predicting the amount of investment according to several categories such as "no investment", "low investment", "medium investment" and "huge investment". Thus, the sense of ordinal classification is to accomplish this kind of problems by exploiting the ordinal relations among the label values and forcing this constraint in the modeling.

4. **Transfer learning**

   It aims to extract the knowledge from one or more origin sources and to apply the knowledge obtained to a different destination task (Pan, 2009). Traditional learning algorithms assume that training data and test data are extracted from the same source and they more or less maintain the same distribution and feature space. But if this distribution changes, these methods need to rebuild or adapt the model in order to work well. The so called data set shift problem is closely related to transfer learning.

2.3 Data Mining

- Data mining (DM)

There are several definitions of data mining. For example, according to (Rajaraman, 2011) knowledge discovery and characterizing the process that finding patterns and extract the rules in data sets. The model, the heart of the data mining process. Analogously, data mining should have been more appropriately named "knowledge mining from data".

- Knowledge discovery(KD)

As a process that seeks new knowledge about an application domain. It consists of many steps, one of them being DM, each aimed at completion of a particular discovery task, and accomplished by the application of a discovery method (Fayyad, 2001).

- Knowledge discovery in databases(KDD)

Concerns the knowledge discovery process applied to databases.

2.4 Kinds of data mining

DM can be applied to any kind of data. The most forms of data for mining applications are databases, data warehouse, transactional and Other Data.

- Databases

Like as described in(Garcia-Molina, 2008)Databases is a collection of data(collection of interrelated tables)that is managed by specialized software called DBMS(database management system). Database management system (DBMS) ,is set of software programs for defining database structures, manage and access the data, also allowing it to persist over long periods of time, safely. This databases available and richest information repositories, they are a major data form in the study of data mining(Devedzic, 2001)(Chen, 1996). For example, customer data can be analyzed to predict the credit risk of new customers based on their age by using data mining system.

- Data Warehouses

Data warehouse or as well-know OLAP(Silver, 2001) for denote online analytical processing is a repository of information collected of subject-oriented, integrated, non-volatile data from multiple sources, stored under a unified schema(e.g star schema, Snowflake schema), that supports the management decision process(data cleaning, data integration, data transformation, data loading, and periodic data refreshing(Breault, 2002)). Usually, datawarehouses modeled modeled by a multidimensional data structure (construct from fact table and dimensions), called a data cube, that provides a multidimensional view of data by supported several operations include drill-down and roll-up.

- Transactional data

Data mining can find the interconnections of the association between the data items in a data transaction(Brilliant, 2017). Generally, each record in a transactional database captures a transaction, such as a customer'is purchase, a flight booking, or a user's clicks on a web page. A transaction typically includes a unique transaction identity number (trans ID) and a list of the items making up the transaction, such as the items purchased in the transaction. A transactional database may have additional tables, which contain other information related to the transactions, such as item description, information about the salesperson or the branch, and so on(Adomavicius, 2001).

- Other kinds of data

In fact, except of what mentioned there are many other kinds of data that have versatile forms and structures and rather different semantic meanings can be mined. Such kinds of data: networked data to detect intrusions based

on the anomaly of message flows, spatial data (e.g., maps), hypertext and multimedia data (text, image, video, and audio) to identify objects and classify them by assigning semantic labels(image), Web data (e.g,distributed information repository made available by the Internet).

2.5 Data Mining perspectives

Data mining has incorporated many techniques from other domains. Here we give examples of several disciplines that strongly influence the development of data mining methods.

- Statistics

Statistics studies the collection, analysis, interpretation or explanation, and presentation of data. Statistical models are a set of mathematical functions used to model data and describe the behavior of the objects in a target class (Kuonen, 2004). Data mining tasks will be able to integrate statistical models(Hand, 1999). For example, we can use statistics to model noise and missing data values. Also, statistical methods can be used to verify data mining results by statistical hypothesis testing. A statistical hypothesis test makes statistical decisions using experimental data.

- Machine learning

Subfield of artificial intelligence will be describe in the section 2.9.

- Database Systems and data warehouses

As mentioned earlier in section 2.4 databases and data warehouse are repositories to store small and large data. However, data warehouse provides a multidimensional view of data. So, we will be able to exploit this two data spaces to processing them using data mining technology.

FIGURE 2.2: Data mining confluence of multiple disciplines.

2.6 Kinds of applications Data Mining targeted

Data mining has seen great successes in many applications domains such as: financial forecasting, weather forecasting, predicting share of television audiences, medical diagnosis...etc. We briefly discuss two popular application examples of data mining: **Business Intelligence** and **Search engines.**

- Business intelligence

Data mining is the core of business intelligence domain. It can help the business to be more successful, for example customer feedback on similar products, make smart business decisions, understanding customer behavior(Kasemsap, 2018)(Wang, 2008).

- Search engines

Web search engine is a computer server that searches for information on the web, operate algorithmically or by a mixture of algorithmic and human input. Various data mining techniques are used in all aspects of search engines. for example web crawling (spider or spider bot), indexing, and searching. Search engines pose grand challenges to data mining one of them concerned about how to deal with amount of data. Firstly, in this case search engines need to use computer clouds. Secondly, how to deal with online data and incrementally updating a model on fast growing data streams. Thirdly, the dealing with queries that are asked only a very small number of times(Han, 2002)(France, 2002).

2.7 Phases of Data-Mining

The knowledge discovery (KD) process usually involves clearly steps from start to finish. The input to this process is the data, and the output is the useful information desired by the users. The overall data mining solution process can be segmented into three phases: defining the problem, data mining, and taking action, Deploying the solution.

2.7.1 Problem identification

The data mining process begins not with data but with a problem to be solved. Stating the problem, translating it into one or more questions that data mining can address, and understand how the data mining results will be used to solving the problem and that enable us to determine the best data mining approach. It important to mention that data mining is not always the right tool to solve the problems. For example, make the sales force more productive by reducing the time required to gather certain information. The

right suggestion is might be a new reporting system based on real-time, multidimensional queries rather than data mining.

Understanding the problem lead us to the appropriate analytical approach to identify the best mining method to use. In fact, there are two categories approaches of data mining: discovery(descriptive) methods and predictive methods.

1. predictive methods

According to(Siraj, 2007), makes a prediction about values of data using known results found from different data (classification , Regression).

2. discovery(descriptive) methods

According to(Siraj, 2007), Discovery methods are data mining techniques that find patterns in the data. The objective is to discover the relationships that are inherent in the data.(clustering, summarization associations rules, and sequences discovery).

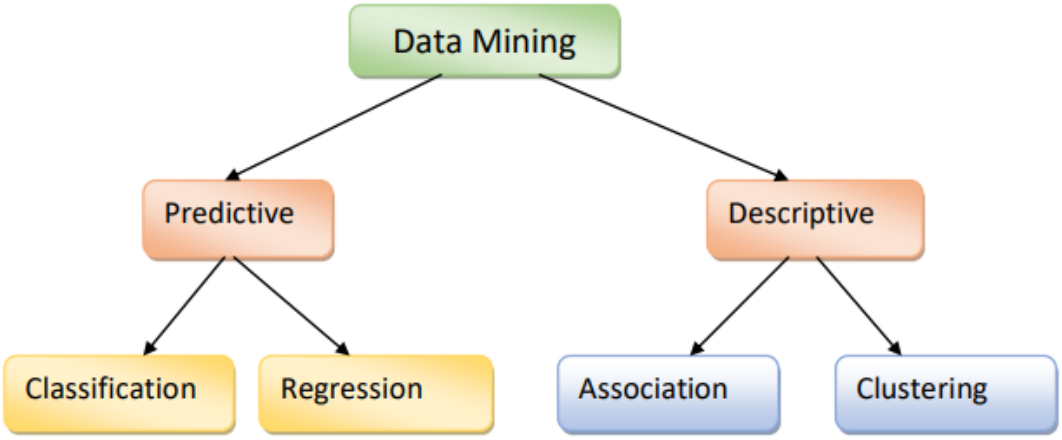

FIGURE 2.3: Data mining tasks.

3. Combining methods

In some cases, a combination of data mining techniques is most appropriate. For exemple, issue is to improve the effectiveness of a marketing campaign for an upcoming back-to-school promotion, then you may decide to perform customer segmentation using the clustering technique to identify a target group of customers, followed by a market basket analysis using the associations technique with the transactions only for the target group to find item affinities on which to base the promotion, or medical diagnostics (Shamsollahi, 2019)

2.7.2 Build and deploy data mining

This phase itself can be highly iterative it consists of nine major steps:

Phase 1 - Data preparation

Inconsistencies, null values, extreme values and noise are properties of all data sets and relations in data bases. In this way, data cleansing routines will help fill in the null values, identifying outliers and solving inconsistencies. Uncleaned data creates confusion for the scanning procedures and although some algorithms include routines for cleaning these mechanisms they are often not robust, so it is preferable to clean them beforehand(Alasadi, 2017).

Phase 2 – Data integration(Collection)

The merging of data from multiple data sources. Careful integration can help reduce and avoid redundancies and inconsistencies in the resulting data set(Alasadi, 2017). This can help improve the accuracy and speed of the subsequent data mining process.

Phase 3 – Data selection(Exploration of data)

Identification of the available data sources and the extraction and select useful data for analysis, by checking it quantity and quality to build robust models in data mining step, this data consist of a series of samples that will be described by means of a series of variables (Agarwal, 2013). This stage can be achieve by series of steps for example analyze the metadata (data about the data) associated with variable of sample to provide information about data types potential values, original source, format and other characteristics. There are several types of variables of samples :

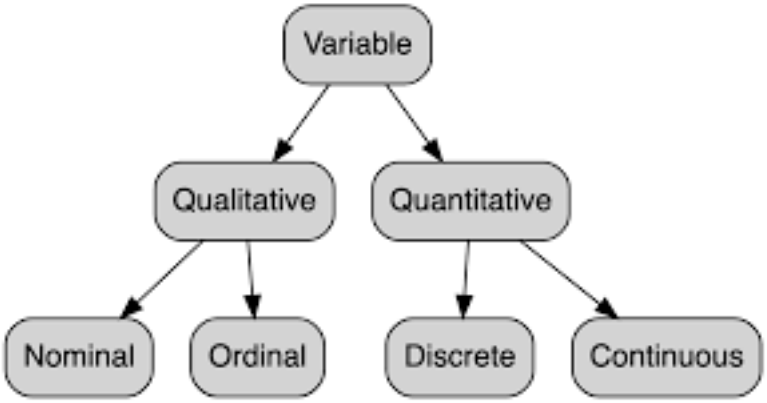

FIGURE 2.4: Variable types.

Quantitative: Can be include:

- **Discrete :** For example: number of persons, number of vehicles...etc.
- **Continues :** For example: salary, length, benefits ...etc.

Qualitative : Can be include:

- **Nominal :** Name the object to which they refer without being able to establish an order (civil status, gender, colour, race...etc).
- **Ordinal :** An order can be established in its values (high, medium, low).

Phase 4 – Data cleaning(data cleansing)

Data can be suffer during the previous phases from several problems, can be incomplete, noisy, consists from from missing values, outliers values(Alasadi, 2017). This stage take the role to handle all these problems and make the data very clean.

Variation of data cleaning problems:

Missing values : it appear when many tuples have no recorded value for several attributes. There are many techniques to handling this problem(Alasadi, 2017).

Ignore the tuple : This method is not very effective when given several attributes with missing values(Alasadi, 2017).

Fill in the missing value manually : This method may not be feasible given

a large data set with many missing values(Alasadi, 2017).

Use a global constant to fill in the missing value : This method is not foolproof, because sometime the mining program may mistakenly think that they (global constant) form an interesting concept(Alasadi, 2017).

Use a measure of central tendency for the attribute : Use the main for normal (symmetric) data distributions, and the median for skewed data distribution.

Noisy data : Is a random error or variance in a variable(corrupt data user system cannot understand and interpret correctly), can adversely affect the results of any data analysis(Alasadi, 2017).

Phase 5 – Data transformation

At this point the data transformed transformed (consolidated) into suitable representation and form (structure) appropriate for mining(Fu, 1997) (Alasadi, 2017). For example, if the algorithm to be used requires numerical input and the data selected are categorical, it very important to convert these data into appropriate format(into numerical data)

Strategies for data transformation

- **Smoothing**

  Remove noise from the data(Alasadi, 2017).

- **Attribute construction**

  add new attributes to help the mining process.

- **Aggregation**

  summary or aggregation operations are applied to the data(e.g., data warehouse aggrega)(Alasadi, 2017).

- **Normalization**

  The principal idea is to adjusting the data value into a specific range such as between [0,1] or [-1,1], techniques can be used to speed the learning stage, Minimum-Maximum, Z-score and decimal scaling are popular forms of normalization(Alasadi, 2017).

Phase 6 – Data mining

At this stage, we choose the data analyst (create), and we apply mining techniques (models) to discover patterns (rules)(Erdoğan, 2005).Different machine learning algorithms are used to build a range of prediction models from which the best model will be selected for deployment.

Phase 7 – Presentation

The resulting model(the discovered knowledge) is created, stored, and will be viewed using the appropriate visualization that the analyst uses to assess model quality and validate the appropriateness of model. In this step the data analyst will need to understand how setting the various parameters affects the resulting model and how to interpret the data model(Lakshmi, 2011).

Phase 8 – Pattern evaluation

Before models can be deployed and used to make decisions within an organization, it is important that they are fully evaluated and proved to be fit for the purpose(Lakshmi, 2011). This phase covers all the evaluation tasks required showing that a prediction model will be able to make accurate predictions after being deployed and that it does not suffer from overfitting or underfitting.

Phase 9 –Deploying the solution(Knowledge discovery)

Machine learning models are built to serve a purpose within an organization. Data mining results can be deployed in various ways to diverse systems:

1. **Ad hoc insight**

   Use data mining on an ad hoc basis to address a specific, non-recurring question. For example, a pharmaceutical researcher may use data mining techniques to discover a relationship between gene counts and disease state for a cancer research project(Iannaccone, 2006).

2. **Interactive insight**

   Integrating data mining into a BI application for interactive analysis(Behal, 2007). For example, a business analyst may regularly use a targeted marketing application to segment the customer base, select a segment suitable for the next promotion, and perform market basket analysis specific to that segment. The item affinities and customer profile then feed analytical/reporting functions that identify the highest-value item relationships for the promotion and the best promotional channel given the characteristics of the target group.

3. **Scoring:**

   At this point, We need to apply a data mining model to generate some sort of prediction for each record, depending on the type of model. For a clustering model, the score is the best-fit cluster for each individual. For an associations model, the score is the highest-affinity item,

given other items. For a sequences model, the score is the most likely action to occur next. For a predictive model, the score is the predicted value or response.

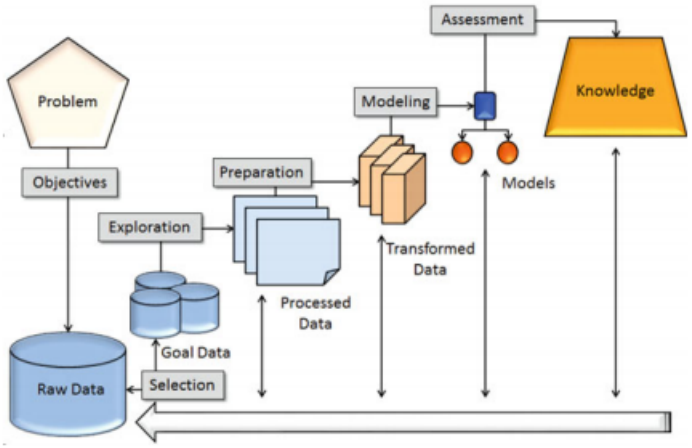

FIGURE 2.5: data mining(KDD) process.

2.8 Data Classification Algorithms

Classification is a task that extracts models called classifiers from given datasets, assign(predict) categorical (discrete, unordered) class labels to samples. For example classification of emails to spam or non-spam emails, medical researcher (e.g., cancer prediction), text classification, images classification...etc.

2.8.1 General approach for classification

The classification process can be divided into two main steps(Aggarwal, 2015), consisting of a learning step and a classification step, these two steps are described briefly below.

1. Learning step(Training data)

The classifier is designed to describe a predetermined set of training data. This step viewed as the learning of a mapping or function, y = f (X) .This mapping is represented in the form of classification rules decision trees, svm, association rules. . . etc(Aggarwal, 2015).

2. Classification step (Test data)

The model is used to classify a given set called test set to estimate the accuracy of this classifier (samples are correctly classified by the classifier)(Aggarwal, 2015). Several methods used for estimating classifier accuracy (accuracy, precision, f-score).

2.8.2 Common Techniques in Data Classification

It is difficult to comprehensively discuss all the methods used for data classification in this section, but we will discuss most common techniques used for data classification, that can be divided into some groupes: ***Probabilistic methods, Tree-based methods, Rule-based methods, Instance-based methods, Support vector machine(SVM) methods, Artificial neural networks (ANN).***

2.8.3 Probabilistic Methods

probabilistic methods is subclass from classification rules. These methods use statistical inference to find the best class for a given instance. The main idea behind these methods is to output a corresponding probability (posterior probability) of the instance being a member of each of the possible classes, the class with the highest probability is selected as the best class for this example. Posterior probability is the probability after observing the specific characteristics of the test instance. On the other hand, the prior probability is fraction of training records belonging to each particular class, with no knowledge of the test sample. For new data point we will use decision theory to to determine class label of this instance. There are two ways to estimate the posterior probabilities. Firstly, use class conditional probability and the prior class separately. Secondly, is to directly model the posterior probability.

Common probabilistic methods

There are several fundamental algorithms for probabilistic classification, including: Naive Bayes Classifier, Logistic Regression, Probabilistic Graphical Models for Classification(Bayesian Networks, Hidden Markov Model, Conditional Random Fields). In this section we will briefly review these techniques.

1. **Naive Bayes Classifier**

   Naive Bayes is a probabilistic classifier. It is especially suitable when dimensionality of the inputs(features) is high,uses simple rules of probability know as Bayes theorem to train a classifier, used in several classification problem such as document classification, spam detection, text classification, for more discuss you can review this papers (McCallum, 1998)(Metsis, 2006.), medical diagnosis, etc.

2. **Logistic Regression(LR)**

   LR is statistical classification algorithm, is an approach to learning p(Y | X) directly from the training data,where X=(x1,x2,x3,. . . ,xn) is vector containing discrete or continuous variables and Y is discrete-valued. LR is a technique used in many classification problem such as patient patient, for more informations on this type of problem you can look to (Bagley, 2001)(Liao, 2007). Also, used in American voter (vote Democratic or Republican) , you can review(Antonakis, 2009).

3. **Probabilistic Graphical Models for Classification**

   Graphical model(Jordan, 2004), is a probabilistic formalism provide a way to visualize the structure of a probabilistic model, by formulating probabilistic models of complex phenomena and control over the computational cost associated with these models in applied fields. Constructed from nodes and links(arc). The nodes in the graph are identified with random variables, and the links express probabilistic relationships between these variables. There two common forms of graphical model; directed graph and undirected graph.

   (a) **Bayesian Networks(BNs)**

   BNs also known as belief networks or belief networks is directed graphical models. In fact, it is a veryideal algorithm to predict the causes of a event occurred. The nodes represent random variables and the edges represent conditional dependencies, and the arcs have a particular directionality indicated by arrows.

2.8.4 Tree-based methods

The famous intuitive classification(regression) algorithm called decision trees is a map of the possible outcomes of a series of related choices. It used in several real application such as business(to making high-value investments), scientific, and health care(diagnosing illnesses and making treatment decisions)(Chen, 2011). It consists from root node(unique node), internal node and leaf nodes. The internal node corresponds to a partitioning decision (or edge of the tree can be translated to a Boolean expression (e.g., x <15), and leaf nodes is mapped to a class label prediction(outcomes prediction). The path from root to leaf generates the conjunction rule (logical AND) of all the decisions from parent to child.

2.8.5 Rule-based classification

Famous classification approach called rule-based classification presented in literature is valuable due to the following advantages.

1. The people can understand and interpret them easily.

2. Once rules are learned and stored into a rule database, we can subsequently use them to classify new instances rapidly.

In rule-based approach the rules can represent information (knowledge)in a very simple way. They provide a very good data model that human beings can understand very well. Rules are represented in the logic form as **IF**:Represented the condition (Antecedent).
THEN: Represented the consequent(Conclusion).
IF-THEN: Statements,a commonly used rule can be expressed as follows:
IF:Condition **THEN**:Conclusion (Also,can write **Condition** $\rightarrow$ **Conclusion)**.

It basically means that if the condition of the rule is satisfied, we can infer or deduce the conclusion. The condition consists of one or more feature tests (e.g. feature 1 > value2 , feature 5 = value 3) connected using logic operators (i.e., ”and”, ”or”, ”not”).

Example:

If Sex=”femal” **And** (35 < Age <45) **And** (Salary=”high” **Or** credit Limit=”high”), **THEN** Potential Customer =”yes”.

This approach is based on storage knowledge and logic inference.Thus, it is applied in various expert systems such as fault diagnosis, medical diagnosis, conversational QA systems.

Common rule-based techniques

There are two main groups techniques for rule-based classification, here we will address these two kinds briefly :

1. **Rule induction(sequential covering)**

   Rule induction consists of a list list of discovered rules called a decision list, the ordering of the rules is very important to decides which rule should be first used for classification. The key strategy used here as follows:constructed the rules(e.g,**Condition** $\rightarrow$ **conclusion**) on data instance (training or test instance). after each rule is learned, the training examples covered by the rule are removed from the training data, and the remaining training data are used only to learn data are used to learn subsequent rules. These key learning steps are performed repeatedly until the remaining training set becomes empty or no new rule can be learned from the training data.

   2. Classification based on association rule mining

   This kinds of learning use the association rule mining techniques to mine association rules by applying the discovered rules. The key strategy used here as follows: It aims to discover all co-occurrence strong correlations(associations) between the frequent item sets and class labels based on association rule mining techniques(e.g,**Apriori algorithm**) from very large data sets. This kinds of learning applied in

several real-world domains such as business, finance, economy, manufacturing, aerospace, biology, and medicine, Web and text documents, etc.

The classic application of association rule mining is the market basket data analysis, aiming to determine how items purchased by customers in a supermarket (or a store/shop) are associated or co-occurring together.

For example, an association rule mined from market basket:
Bread $\rightarrow$ $Milk[support = 10\%, confidence = 80\%]$.

The rule basically means that we can use Bread to infer Milk or those customers who buy Bread also frequently buy Milk. However, it should be read together with two important quality metrics, namely support and confidence. Particularly, the support of 10% for this rule means that 10% of customers buy Bread and Milk together, or 10% of all the transactions under analysis show that Bread and Milk are purchased together. In addition, a confidence of 80% means that those who buy Bread also buy Milk 80% of the time. This rule indicates that item Bread and item Milk are closely associated.

Typically, association rules are useful if they satisfy two constraints, if [their support]is larger than[a minimum support threshold.] and [their confidence]is larger than [a minimum confidence threshold].Both thresholds are typically provided by users and good thresholds may need users to investigate the mining results and vary the threholds multiple times.

2.8.6 Support Vector Machine classifiers (SVM)

SVM is a powerful classification tool for controlling model complexity. It is constructed to build the separating surface between data instances. It used to binary classification and also, perform multi class classifications. It search to obtain an optimal separating hyperplane, which maximizes the margins between classes to get less over fit, it can also handle non linear data separable by kernel tricks to construct non linear separating surfaces(mapping the sample points into a high-dimensional feature space)(Huang, 2018).

2.8.7 Artificial Neural Networks (ANN)

ANN is mathematical models that mimic the functioning of the human nervous system. In 1943 McCulloch and Pitts in 1943 published model of the neuron (McCulloch, 1943)

2.9 Machine Learning(ML)

ML is field from artificial intelligence(AI) investigates how computers can learn based on data. That means the ability to generalization, or more generally speaking, creating a classifier that can be used to generalize from new instances such as described by Kotsiantis in(Kotsiantis, 2007). formal definition of machine learning is given by Mitchel (Mitchell, 1997): A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

2.9.1 Types of machine learning

Machine learning algorithms are broadly classified into three categories: ***supervised learning***, ***unsupervised learning*** and ***reinforcement learning***. Each learning method has its meaning and its dimensionality. The figure below shows the common types of machine learning.

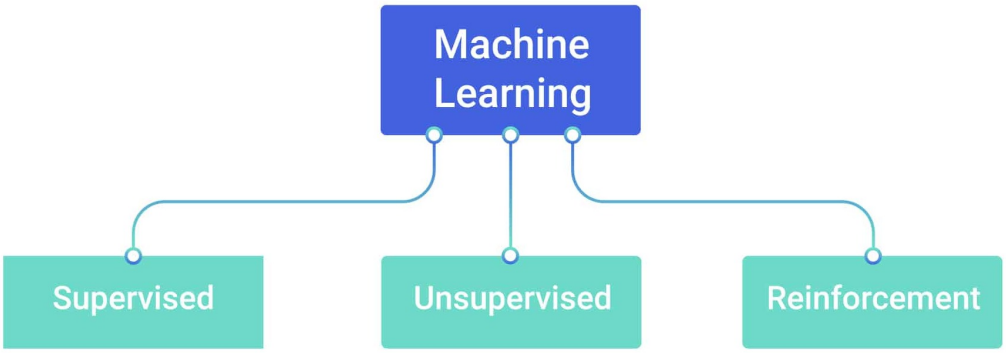

FIGURE 2.6: Type of machine learning.

1. Supervised learning

From the literature litterature papers (Hastie, 2009)(Liu, 2011) the supervision in the learning comes from the labeled instances(predicted classes are known beforehand) in the training datasets. This approach has been broadly classified into two types regression(include some of regression algorithms such as Linear regression,..,etc) and classification(include some of regression algorithms such as**SVM**, **logistic regression(LR),...,etc)**. There are several problems in real word used this kind of learning for example: BioInformatics, Spam Detection, Speech Recognition.

2. Unsupervised learning (clustering)

As mentioned in the literaturein literature (Berry, 2019) the input examples (data collected)are not class labeled. Typically, we may use clustering to

discover classes(groups) within the data. This approach contains many clustering models such as, **K-means,...etc**. There are several problems used this learning for example, anomaly detection.

3. Reinforcement learning

Reinforcement learning (agents) mimics how humans learn by interacting with environment, repeating actions for which the reward that is received is higher, and avoiding risky moves for which there is a low or negative reward as an outcome of their actions(Ghahramani, 2003). The system has no prior knowledge about the behavior of the environment and the only way to find out is through trial and failure (error).

2.9.2 Relationship between machine learning and data mining

We can see there are many similarities between data mining and machine learning. For classification and clustering tasks, machine learning research often focuses on the accuracy of the model. In addition to accuracy, data mining research places strong emphasis on the efficiency and scalability of mining methods on large data sets, as well as on ways to handle complex types of data and explore new, alternative methods. According to **Kaggle** web page [1] and **javatpoint** web page [2] we describe the difference of both such as:

- **Machine Learning**

  ML relates to the study, design, and development of the algorithms that give computers the capability to learn without being explicitly programmed.

- **Data Mining**

  DM is the process in which the unstructured data tries to extract knowledge or unknown interesting patterns. During the process of Data Mining, Machine Learning algorithms are used.

[1] https://www.kaggle.com/getting-started/206108
[2] https://www.javatpoint.com/data-mining-vs-machine-learning

Conclusion

This chapter covers an overview of phases of data mining from selection of goals (problem identification) to the deployment. Moreover, we talked about data classification algorithms, we will talk about data classification algorithms. Finally, we conclude this chapter by describing the relationship between two important domains data mining(DM) and machine learning(ML). In the next chapter, we will discuss the classification approaches and techniques for imbalanced data sets, this is the core of our proposed subject.

Chapter 3

Classification approaches and techniques for imbalanced datasets

3.1 Introduction

Generally, we can describe The imbalanced dataset problem when the number of instances which represents one class is smaller than the ones from other classes. The minority classes are usually the most important concepts to be learnt. Additionally, other data characteristics that are related to this concept(imbalanced data) may include: Small sample size, overlapping between classes, lack of representative data, small disjuncts, dateset shift data distribution (imbalance)(Guan, 2009).

3.2 Characteristics related to imbalanced data

1. Small sample size(the severe lack of information)

When you do't have enough minority class examples. Also, in (Japkowicz, 2002), the authors report that the error rate caused by imbalanced class distribution decreases when the number of examples of the minority class is representative, because few size of data sets(e.g., instances in minority class) avoid the classifiers to learn better (small domain down the sample of minority class this directly increase error rate classification and large domain rise the samples of minority class and decrease error rate.

2. Overlapping (class separability)

The instances of the minority class share a region with the majority class, where all the instances are intertwined. Also, as mentioned in (Prati, 2004), the authors reported that the overlap affects more than imbalance.

3. Small disjuncts

This problem occurs when the concept represented by the minority class is formed of subconcepts (Holte, 1989). In most of the problems, small disjuncts are implicit and their existence usually increases the complexity of the problem. the sub-concepts lead to the creation of small disjuncts.

4. Dataset shift(data fracture or changing environments)

Dataset shift or data fracture have already been described in several literature papers(Alaiz-Rodríguez, 2008)(Cieslak, 2009)a common problem in predictive modeling that occurs when the joint distribution of inputs and outputs differs between training and test stages or as Abdallah & al. (Abdallah, 2016) reported concept drift as the phenomenon where the underlying model (or concept) is changing over time for example an example is email spam filtering, which may fail to recognize spam that differs in form from the spam the automatic filter has been built on.
There are three types of dateset shift:

- **Prior probability shift**

  According to what the authors described (Storkey, 2009), prior probability shift refers to a situation where when one considers the distribution of both of the outputs(target) (distribution P (y) changes between training and test situations), the distribution is different.
  Ptraining $(x|y) = Ptest\,(x|y)\,and Ptraining\,(y) \neq Ptest\,(y)$

- **Covariate shift**

  It is about attribute values that have different distributions between the training and test sets.

- **Concept shift (concept drift or functional relation change)**

  The relationship between the input and class variable changes over time, or like the authors described in (Yamazaki, 2007)(Lane, 1998), where the p(y | x) (observed patterns of usage) changes between the training and test phases .

  Example

  Context of weather data. Seasons may not be clearly defined in temperature data, but ultimately influence the temperature data.

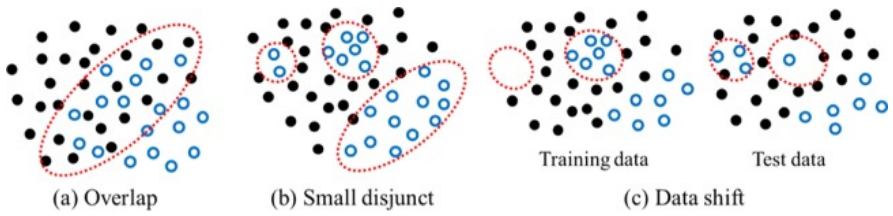

FIGURE 3.1: Approach to solve imbalanced problem.

5. Data distribution (imbalance)

Class imbalance is often presented as an obstacle to the induction of good classifiers by machine learning algorithms. furthermore, also related to the degree of overlapping among the classes(Prati, 2004). An imbalanced class is represented when the number of instances which represents one class (minority class) is smaller than the ones from other classes(majority class).

6. Noisy data

Real world data generally present many inconsistencies that negatively affects data quality. These inconsistencies are generally denominated as noise, and can decrease the performance of learning algorithms(Zhu, 2004). Two kinds of noise are distinguished in the literature: feature (or attribute) and class noise.

- **Attribute noise**

  Feature noise is represented represented by errors that are introduced to attribute values. Examples of those external errors include **(1)** erroneous attribute values, **(2)** missing or don't know attribute values, **(3)** incomplete attributes or don't care values.

- **Class noise**

  Two possible sources for class noise; **(1)** Contradictory examples: The same examples appear more than once and are labeled with different classifications. **(2)** Misclassifications: Instances are labeled with wrong classes. This type of errors is common in situations that different classes have similar symptoms.

3.3 Popular approaches and techniques for handling imbalanced datasets

Over the past decade, several methods have been proposed to address class imbalance. Although no global solution has yet been found, many approaches have been introduced during the past decade. We discuss some of these approaches in this section.

3.3.1 Tackling imbalanced data

A large amount of techniques have been developed in order to achieve the objective of correctly distinguishing the minority class. These techniques can be categorized into four main groups: **Data-level-techniques**, **Algorithm-level-techniques**, **Cost-sensitive-level(CSL)-techniques**, **Ensemble-based-level-techniques**, depending on how they deal with the problem. such as

mentioned in this papers(Krawczyk, 2016)(Fotouhi, 2019)(Fernández, 2018) (Amin, 2016)(López, 2013). the figure below describe this approach:

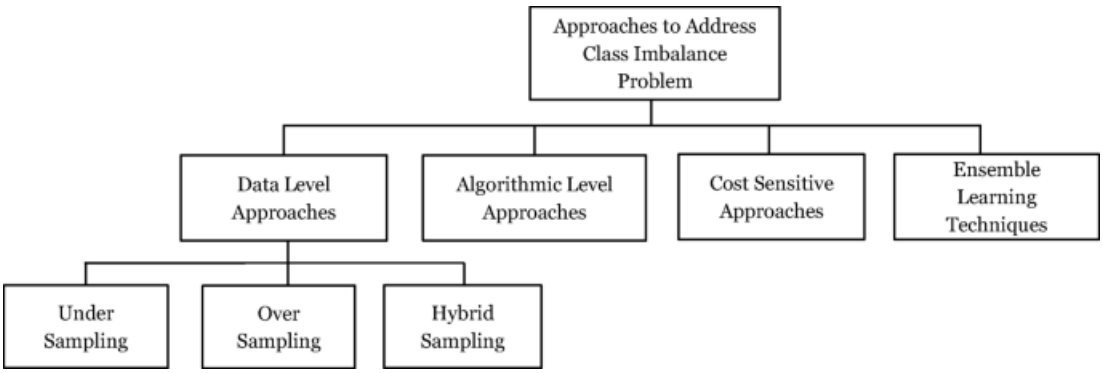

FIGURE 3.2: Approach to solve imbalanced problem.

3.3.2 Data-level methods(external methods or Data sampling)

That modify the collection of examples to balance distributions (add or remove) difficult samples. To balance the imbalanced data, various sampling methods have been proposed in the literature and are used to change imbalanced data into balanced data.

There are several **data-level techniques**, we will describe the most common techniques below:

A- Under-sampling and Oversampling Basics

Resampling or data preprocessing level techniques can be categorized into three groups or families(Fernández, 2018) (Batista, 2004):

1. **Undersampling (RUS):**

   RUS is random which creates a subset of the original dateset by eliminating (removing)instances (usually majority class instances). Undersampling of the majority (normal) class has been proposed as a good means of increasing the sensitivity of a classifier to the minority class (abnormal).

2. **Random overdersampling (ROS)**

   ROS which create a subset of the original datasets by replicating(adding) some instances or creating new instances from existing ones.

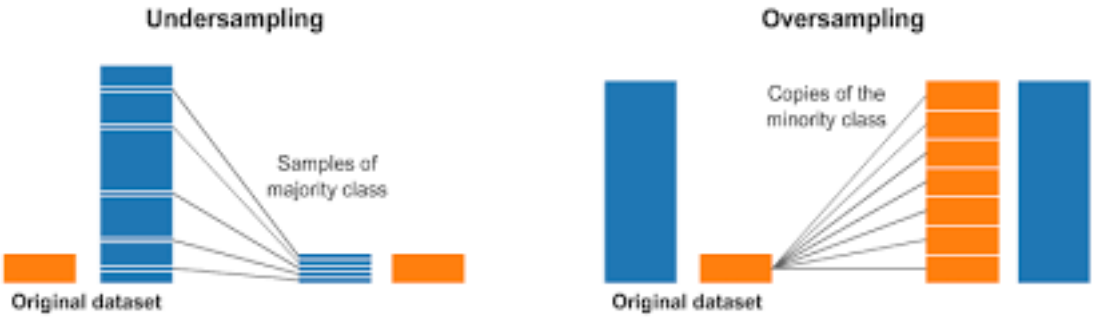

FIGURE 3.3: Oversampling and Undersampling approaches for class imbalanced.

3. **Hybrids methods:**

Hybrids methods which combine the two sampling approaches.

B-Common under-sampling methods

1. Tomek Links (TL):

TL (Tomek, 1976) algorithm can be defined:Two instances (observation)$E_i = (x_i, y_i)$ and $E_j = (x_j, y_j)$ where$y_i ! = y_j$and $d(E_i, E_j)$:denotes the Euclidean distance between E_i and E_j, where E_i denotes sample that belongs to the minority class and E_j denotes sample that belongs to the majority class. If there is no samplex_k satisfies the following condition:
1.$d(x_i, x_k) < d(x_i, x_j)$, or
2.$d(x_j, x_k) < d(x_i, x_j)$
then the pair of (x_i, x_j) is a**Tomek Link.**

The learning base obtained after removing links from Tomek is organized into a set of clusters. Figures **left** represented the original dataset before **Tomek Link**, and the **Middle** represented application of **Tomek Link**, and the **right** represented the original dataset after **Tomek Link** below illustrate the identification of Tomek links and the state of the database after removing these links. Knowing that the little blue circles represent the majority instances and the orange rectangle represent the minority instances.

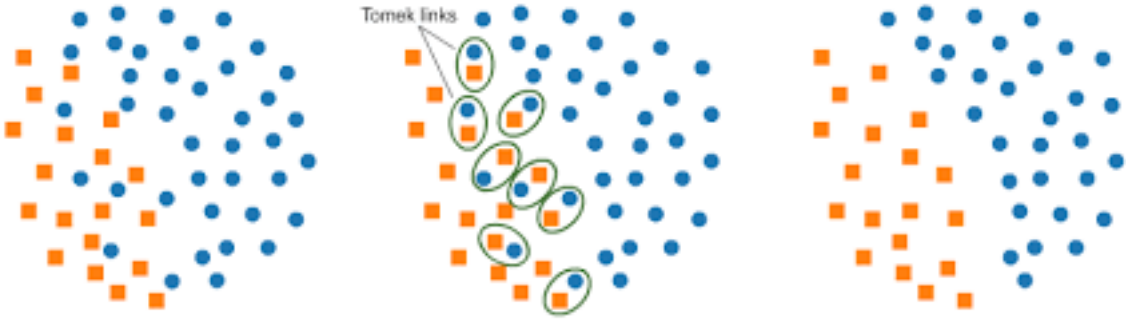

FIGURE 3.4: Represented Tomek Links algorithm.

2. Condensed Nearest Neighbor Rule (US-CNN):

Condensed Nearest Neighbor (CNN)(Tomek, 1976) is an algorithm designed to reduce the data set for k-NN classification proposed by(Hart, 1968). It constructs a subset Ê of examples which are able to correctly classifythe original data set.

A subset Ê ⊆ E is consistent with E if using a 1-nearest neighbor. Ê correctly classifies the examples in E. The **US-CNN** rule uses the following algorithm to determine a consistent subset of the original sample set.

Algorithm 1 Condensed Nearest Neighbor Rule (US-CNN):

Input: Original training set E.

Output: consistent subset Ê.

1: **procedure** US-CNN
2: Randomly draws one majority class example and all examples from the minority class and put these examples in Ê and.
3: Use a 1-NN over the examples in Ê to classify the examples in E.
4: **while** $x^{'}$ misclassified example from E **do**
5: $x^{'}$ is moved to Ê.
6: **end while**
7: **end procedure**

3. One-Sided Selection (OSS)

Undersampling method applies Tomek links followed by the application of US-CNN.

Tomek links are used as an undersampling method and removes noisy and borderline majority class examples and US-CNN aims to remove examples from the majority class that are distant from the decision border. Algorithm for the one-sided selection of examples according to(Kubat, 1997)can be such as:

Algorithm 2 One-Sided Selection (OSS)

Input: Original training set S.

Output: Resulting set is referred to as T

procedure OSS
    Initially, C contains all positive examples from S and one randomly selected negative example.
    Classify S with the 1-NN rule using the examples in C, and compare the assigned concept labels with the original ones.

    for all misclassied examples from S **do**
        Move all misclassied examples into C that is now consistent with S while being smaller.
        Remove from C all negative examples participating in **Tomek links.**(This removes those negative examples that are believed borderline and/or noisy).
        All positive examples are retained.
    end for
end procedure

4. US-CNN + TL

According to(Batista, 2004) it is similar to the one-sided selection, but the method to find the consistent subset is applied before the Tomek links. As finding Tomek links is computationally demanding, it would be computationally cheaper if it was performed on a reduced data set. The resulting set is referred to as T.

5. Neighborhood Cleaning Rule (NCL):

According to (Batista, 2004), NCL modifies the ENN in order to increase the data cleaning. For a two class problem the algorithm can be described in the following way: for each example E_i in the training set, its three nearest neighbors are found.If E_i belongs to the majority class and the classification given by its three nearest neighbors contradicts the original class of E_i, then E_i is removed. If E_i belongs to the minority class and its three nearest neighbors misclassify E_i, then the nearest neighbors that belong to the majority class are removed.

6. Edited Nearest Neighbors Rule(ENN)

A new approach of resampling and classification was proposed Dennis Wilson in 1972 paper titled "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data." (Wilson, 1972)(Penrod, 1976)(Guan, 2009) (Alejo, 2010).

According to (Penrod, 1976) (Guan, 2009)(Alejo, 2010) we can resume this algorithmically, and it can be expressed as follows:

Algorithm 3 Edited Nearest Neighbors Rule(ENN)

Input: Original training set S.

Output: Resulting set is referred to as T

1: **procedure** ENN
2: let S=X
3: **for** xi in X **do**
4: Discard xi from S if is missclassified using the K-NN rule with prototypes in X-xi
5: **end for**
6: **end procedure**

7. Class Purity Maximization (CPM) :

Acorrding to(Yoon, 2005) **CMP** can define such as:

Algorithm 4 Class Purity Maximization (CPM)

Input: Imp : cluster impurity of parent cluster.
parent : parent cluster ID.

Output: Subclusters C i rooted at parent.

```
impurity ←∞
procedure CPM(IMP,PARENT)
    while Imp impurity do
        If all the instance pairs in parent were tested then return
        Pick a pair of majority and minority class instances as centers
        Partition all instances into 2 clusters C_1 and C_2 according to nearest center
        impurity min(impurity (C_1 ),impurity (C_2))
    end while
end procedure
CPM (impurity (C_1),C_1 )
CPM (impurity (C_2 ),C_2)
```

C- Advanced Under-sampling Techniques:

In this section, we will review the advanced undersampling mechanisms to address the data sampling, such as **(1)** Evolutionary Undersampling algorithms textbf(2)Undersampling by Cleaning Data **(3)**Ensemble Based Undersampling **(4)** Clustering Based Undersampling **(5)**Synthetic Minority Oversampling Technique(SMOTE). The description of each groups will be done briefly below.

Here we will enough to explain SMOTE and some of his extension that we will need it to combine it with boosting methods in chapter four to address imbalanced problem.

1. **Synthetic Minority Oversampling Technique (SMOTE): Introdution**

   Previous research such as (Ling, 1998) has discussed over-sampling techniques (with replacement), he noted that it doesn't significantly improve minority class recognition. To make this more significantly a new specific over-sampling approach proposed in literature called "SMOTE" technique. According to (Chawla, 2003), SMOTE algorithm is over-sampling approach to rebalance the original training set. The

objective of SMOTE is to generate examples of the minority class through the linear interpolations of instances (along the line segment joining a minority class sample and its nearest neighbor)of this same class that are close in the dimensional space of the problem. These new examples will be known as 'synthetic' examples such as described in the figure below. The complet pseudo-code represent SMOTE algorithm mentioned in chapter four.

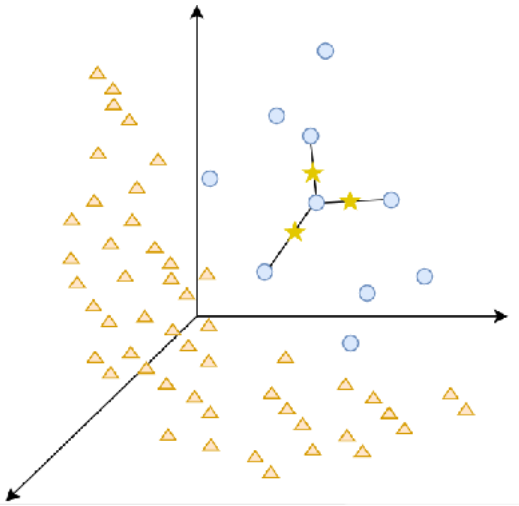

FIGURE 3.5: Synthetic Minority Oversampling Technique.

Extensions of SMOTE

There are several SMOTE-based extensions proposed in the specialized literature. Currently, there are more than 90 extensions published in scientific journals and conferences. Here, we will take example from this extension called ADASYN: Adaptive Synthetic Sampling Approach.

ADASYN: Adaptive Synthetic Sampling Approach

According to(He, 2008) is an extension of smote algorithm, based on the idea of adaptively generating minority examples according to their distributions. The idea behind this algorithm that: more synthetic data are generated for samples of minority classes that are more difficult to learn compared to samples of minorities that are easier to learn.

3.3.3 Algorithm-level methods (internal methods or Algorithmic modification)

Algorithm approach is alternative approach to data preprocessing level for dealing imbalanced datasets. This techniques directly modify existing learning procedure(learning algorithm) or creating a specific algorithms instead on altering the supplied training set to modify the learning cost(accuracy)

and alleviate the bias towards majority class and adapt them to mining data. (Yusof, 2017)(Kotsiantis, 2006).

Also, this is requires a good insights and a good understanding (e.g a good understating of the parameters) into the modified learning algorithm to identify the reasons for failure in mining skewed distributions. Generally, this level divided into two branches cost-sensitive approaches and one-class learning such as described in the figure bellow.

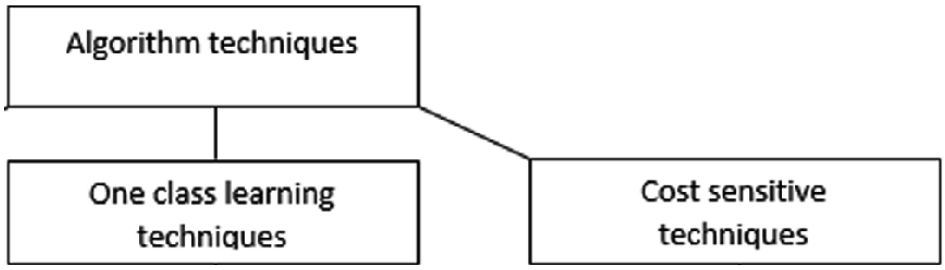

FIGURE 3.6: Types of algorithmic approaches consists of Cost Sensitive Learning and One Class Classification.

- Cost sensitive learning :

One of the classification metric is confusions metric, it allow to different types of errors including false negative and false positive. Cost sensitive learning or also called as reweighting or adjusting the cost of various imbalanced classes is an approach allow to assigned a different cost to classification errors such as false negative and false positive patterns(Nanni, 2015). According to (Weiss, 2007), the best metric for evaluating the classifier performance is the total cost. Total cost formula: Total cost$= (FN \times CFN) + (FP \times CFP)$. Categories of cost sensitive learning :
1) Misclassification cost to form data space weighting,
2) cost minimizing techniques to combine schemes of ensemble method .
3) combining the cost sensitive function with classifiers.

- One class classification approach(OCC) :

OCC also called learning in the absence of counterexamples, is an approach care about learning and capturing characteristics of training instances from one class called (target class) and ignore remaine classes called (outliers classes). In handling the imbalance problem, this approach normally focuses on identification of the minority class samples as the target class, and the majority class will be outliers inverse to when the minority class when the minority class lack any structure, being such as small disjuncts or noisy instances. However, it needs some specialized methods to use one-class learners for more complex problems. Categories of one class classification :

1) Training a one-class classifier on the majority class.
2) Training a well-tuned one-class classifier on the minority class.
3) Training one-class classifiers on both classes and combining their outputs.

- Algorithm level examples

In the following sections, we will discuss selected algorithm-level modifications of popular classification algorithms such as **Decsion tree and support vector machine(SVM**.

Decision Trees(DT)

DTs is an efficient algorithm. However, they are fail in learning from imbalanced data. From algorithm-level point of view to improving decision trees lies in modifying their key component. Actually, with the balanced datasets the splitting criterion of features will be entropy (gain ratio). However with imbalanced data the traditional decision tree splitting criteria (i.e., entropy and Gini) will be inefficient and weak.

Solution: As as result, Cieslak et al. (Cieslak, 2012). Proposed a new criteria ''Hellinger Distance" for creating splits. This will refer to Hellinger distance and Hellinger distance based decision trees as HDDT(Hellinger Distance Decision Tree).

Formula of Hellinger distance for binary imbalanced data.

$$d_H(TPR, FPR) = \sqrt{(\sqrt{TPR} - \sqrt{FPR})^2 + (\sqrt{1 - TPR} - \sqrt{1 - FPR})^2} \quad (3.1)$$

1. **Support Vector Machines:**

   SVMs are powerfull algorithm to handling various learning difficulties. However, in their original form they are highly prone to imbalanced class distributions. This derived a number of solutions that modified the underlying mechanisms of SVMs in order to make them skew-insensitive. Generally, the modification of SVM algorithm divided into four main categories:
   1) Using specific kernel modifications.
   2) Weighting training.
   3) Instances according to their importance.
   4) Using active learning paradigm to select a subset of training instances.
   5) Other approaches.

3.3.4 Cost-Sensitive Learning :

Hybrid approach methods combine both algorithm and data level approaches to incorporate different misclassification costs for each class in the learning phase.

Cost Sensitive Learning (CSL)(Ling, 2008) is subfield of learning in data mining and machine learning. It is one of the most important research areas in machine learning, and it plays an important role in real-world data mining applications. It has been mostly studied in recent years. It training a models on data that have uneven costs(penaltie) when making predictions. The idea behind this type of learning is that takes the misclassification costs (and possibly other types of cost) into consideration. The goal of this type of learning is to minimize the total cost.

The imbalanced datasets occurs in many real-world applications where the class distributions of data are not equals. CSL is one from the recent approaches introduce to deal with class imbalanced.

Theory

Most machine learning models(supervied learning) based in assumption that the class distribution is balanced and most machine learning algorithms also assume that the prediction errors made by a classifier are the same error(all incorrect (miss-classifications)prediction,are equally costly). Howover, in most real-world applications, this assumption is not true,the differences between different misclassification errors can be costly (loss monetary, loss time, threat humain life...etc). For example:

1. **Bank Loan Problem**

   According to **Machine Learning Mastery** web page [1], the bank wants to determine whether to give a loan to a customer or not. if the(bad cutomer=Denying a loan) is regarded as the positive class(Minority class), and (good cutomer=giving a loan) as negative class(majority), then missing a loan (the customer is actually positive but it is classified as negative; thus it is also called "false negative") is much more serious (thus expensive=vey costly) than the false-positive error. The bank could lose all it money because the bad cutomer that may never repay it.

2. **Cancer Diagnosis Problem**

   According to **Machine Learning Mastery** web page [2], the medical diagnosis of a cancer, if the cancer is regarded as the positive class (Minority class), and non-cancer (healthy) as negative, then missing a cancer (the patient is actually positive(not healthy) but is classified as negative(healthy); thus it is also called "false negative") is much more expensive(cotly) than the false-positive error(patient healthy classified as not healthy). The patient could lose his/her life because of the delay in the correct diagnosis and treatment.

[1] https://machinelearningmastery.com/cost-sensitive-learning-for-imbalanced-classification/

[2] https://machinelearningmastery.com/cost-sensitive-learning-for-imbalanced-classification/

Cost sensitive learning(CSL) introduced mostly in(Elkan, 2001)(Zadrozny, 2001) (Domingos, 1999) to describe how much the misclassification cost plays its essential role in various cost-sensitive learning algorithms.

1-Cost-Sensitive Imbalanced Classification

Cost-sensitive learning for imbalanced classification solution
is focused on first assigning different costs to the types of misclassification errors that can be made. Maloof(Maloof, 2003)indicated that, learning from imbalanced data sets and learning with unequal costs can be handled in a similar manner, then using specialized methods to take those costs into account, because the cost(penalty) of misclassifying a minor class example is usually more expensive than that of misclassifying a major class(Weiss, 2003).

This different cost assigned to the types of missclassification errors can be lead to a new concept in models evaluation called cost matrix derived from traditionally confusion matrix, the both matrix described below.

Confusion matrix is table of the predictions made by a model on classification tasks. Typically, for binary classification contains two classes negative(majority class)(assigned with value of 0) and positive class(minority class)(assigned with value of 1). The columns represent the actual class to which examples belong, and the rows represent the predicted class. A cell in the table is the count of the number of examples that meet the conditions of the row and column.

	Predicted Positive	Predicted Negative
Actual Positive class	TP(True Positive)	FN (Fals Negative)
Actual Negative class	FP(False Positive)	TN (True Negative)

TABLE 3.1: Confusion matrix for binary classification.

From the confusion matrix we can see there are two most kind of errors predicted, First called False Positives and second called False Negatives. In imbalanced classification process False Negatives(FN) is more intersted because respresented the worse classification case(the positive class(minority class), also called rare class classified as negative class (majority class)).

The cost matrix by assigned different costs into each class. with interchange between columns and rows described as follows.

	Actual Positive class	Actual Negative class
Predicted Positive	C(0,0)=c_{00}	C(0,1)=c_{01}
Predicted Negative	C(1,0)=c_{10}	C(1,1)=c_{11}

TABLE 3.2: Cost matrix for a two-class problem

Cost-Sensitive views

The are two distinct views on cost-sensitive classifiers exist in the literature. On the one hand **cost associated with features** and, on the other hand **cost associated with classes**.

1. **Cost associated with features(Feature cost)**

   In contrast to the traditional concept of features selection that based on select a small subset of informative features that best discriminate the data objects of different classes. Feature cost(estimation cost for featurs extraction) or test costs, literally meaning the cost consumed in acquiring a features value, is a special case of various cost types in machine learning and data mining thus, FC select a small subset of informative features that has select both low-cost and informative features(Zhou, 2016)(Zhang, 2005).

   This type of learning has multi-objective, we try to strike a balance between performance and cost of used features(Turney, 1994)(Penar, 2010). Because in many cases using costly features offer higher predictive power than several cheaper features (few more expensive ones)(Jackowski, 2012).

2. **Cost associated with classes**

   Standard machine learning algorithms use so-called (0/1) loss function, value 0 indicate to a correctly classified instance and value 1 to an incorrectly classified instances. Loss function(LF) 0/1 uses the same cost associated with a missclasication classification for all classes, it is highly susceptible to skewed class distributions(Landgrebe, 2004). In imbalanced classes can be easily minimized loss function by focusing on majority class and ignoring minority class. Cost sensitive solved this problem by adapting a different loss function, with different costs associated with each class. increasing the importance of difficult (e.g., minority) classes. By stronger cost of errors on a given class, we force the classifier training procedure to focus on instances coming from this distribution, aiming to minimize the overall cost, consider the German credit dataset that was published by (Michie, 1994). As the standard machine learning models the objective is to minimize then missclassification error, when predict an instance to specific class the same purpose here, an example should be predicted to have the class that leads to the lowest expected cost.According to (Michie, 1994) the cost matrix given with this dataset is as follows:

	Actual Positive(good)	Actual Negative(bad)
Predict Positive(good)	0	1
predict Negative(bad)	5	0

TABLE 3.3: led German credit datset for credit cart.

3-Type Cost-sensitive learning

Cost sensitive learning can be divided into two groups(Ling, 2008):

Direct approaches

Direct methods are based on directly introducing the misclassification cost into the training procedure of a classifier thus building the underlying training algorithm utilize misclassification costs. It design classifiers that are cost-sensitive in themselves(Ling, 2008).

Meta-learning approaches

Meta-learning approaches convert existing cost-insensitive classifiers into cost-sensitive ones without modifying them by modifying either the training data or the outputs of a classifier. Meta-learning can be applied during two different steps of the classification process(Ling, 2008).

1. **Preprocessing**

   Preprocessing is similar to data-level discussed in previously and aim to modifying the training set by weighting the instances according to a provided cost matrix.

2. **Postprocessing**

   Post-Processing is intended to modify the outputs of a classifier. It does not involve any modification before or during training and the entire effort is moved to introducing the cost factor when a decision about a new instance is being made.

Obtaining the Cost Matrix

Cost-sensitive learning(CSL) relies strongly on cost matrix, incorrectly initialized costs can impair the learning process(Zhang, 2014). In case of class imbalance wrongly set costs can actually mirror a bias towards the majority class into a bias towards minority class while we should aim to get a balanced performance on both of them. There are two scenarios to solve this issue:

1. **Cost matrix provided by an expert**

   Data is accompanied by the cost matrix that comes directly from the nature of a problem(access to a domain expert). For example on given cost matrix credit card fraud detection (Moepya, 2014). A given cost represent two cases an average monetary loss in a case of accepting a fraudulent transaction (this is C(i, j)) and in case of losing a customer after rejecting a valid transaction (this is C(j, i)).

2. **Cost matrix estimated using training data**

   In this approach, we do not have an access to a domain expert, so we do not have a cost matrix. Actually, this a common scenario when we

use CSL to solve imbalanced problems below there are some heuristic setting of cost values or learning them from training data:

Heuristic approach lies in utilizing the IR as a direct way to estimate the cost.For C(i, j) = IR and C(j, i) = 1.

3.3.5 Ensemble methods

Such as to human natural behavior before making any important decision ;he look to several several experts opinions and to consider all of them in order to take the most suitable decision. Ensemble methods (ensemble-based solutions)(Galar, 2011) are technique to improve the performance of single classifiers fellows the same behavior, by constructing several classifiers from the original data and then combining (aggregating)their predictions. The key key point in classifier ensembles, that is, classifier combination or classifier fusion.

1. **Classifier fusion(combination)**

   Classifier fusion is powerfull method for increasing classification rate(Gader, 2996). In this case, all the classifiers of the ensemble are aggregated to obtain the final decision. All classifiers are supposed to be competent in the whole feature space and expected to misclassify different examples (Ponti Jr, 2011). According to(Sajedin, 2010)two different types of classifier fusion (combination) can be considered: non-trainable and trainable.

   (a) **Non-trainable combiners**

   Where the way to combine the classifiers is prefixed, for example, the majority or weighted majority voting(Folino, 2016)(Sarlak, 2011). In this cases, each classifier gives a vote (or a weighted vote) for the predicted class, all the votes are summed up and the class achieving the largest number of votes is predicted.

   (b) **Trainable combiners**

   Where the combiner itself is trained aiming at improving accuracy for instance.

2. **Classifier/ensemble selection**

   These methods perform the combination in a different way. Instead of combining all the classifiers of the ensemble, classifier selection tries to select only those (or that) leading to the best classification performance. Hence, classifier selection assumes that the classifiers are complementary, being experts on classifying the instances from a part of the input space; therefore, when a new instance is submitted, the most competent classifiers (one or more) are selected, which are the ones used to perform the classification. Two types of selections are commonly distinguished(Kuncheva, 2002):

(a) **Static**

A region of competence is defined in training time for each classifier. Then, before classifying a new example, its region is found and the corresponding classifier is used to decide its class(Kuncheva, 2002)(Britto Jr, 2014).

(b) **Dynamic**

In this case, the classifier that classify the new instance is decided during the classification(Kuncheva, 2002). First, the competence (accuracy) of each classifier for classifying the instance is estimated, and only the winner's output will be used to label the instance(Britto Jr, 2014).

The diffetent methods of ensemble methods

1. Bagging methods

Breiman in his paper (Breiman, 1996) introduced the concept of bootstrap aggregating to construct ensembles. The acronym bootstrap is a technique proposed to reduce the variance associated with the prediction, the idea behind the concept bagging consists of training several classifiers with different data subset (bootstrap) replica of the original training dateset. This bootstrap formed by randomly drawing (with replacement) instances from the original dateset like as showing in figure below.

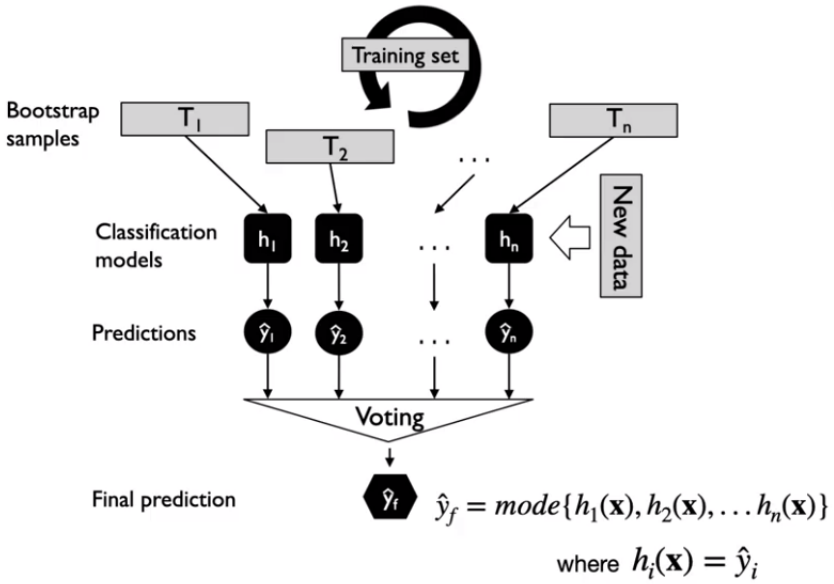

FIGURE 3.7: Description of bagging algorithm.

Algorithm steps

Step1

Create multiple data sets(bootstrap) through random sampling(use specific techniques to create) with replacement.

Step2

Build multiple learners(classifiers) each learner training on bootstrap (subset) in parallel.

Step3

Combine all learners using an averaging or majority vote strategy. The complete pseudo-code of Bagging:

2. Boosting methods

Boosting or as know "Adaptive Resampling and Combining " introduced by Schapire in **1990** proved that a weak learner can be turned into a strong, This idea is the main concept of boosting technique. Actuality, Boosting is a technique proposed to boost the accuracy of a weak learning algorithm and reduce the bias associated with the prediction learner by encourages new models to become experts for instances handled incorrectly by previous model, by combines (running) weak models of the same type (e.g decision trees) many times iteratively on the same original datasets(non-subsets).

Algorithm steps

Step1

Create multiple data sets through random sampling with replacement over weighted data.

Step2

Build learners sequentially.

Step3

Combine all learners using a weighted average strategy.

Several family of boosting techniques in literature, AdaBoost (Freund, 1997) is the most representative algorithm in this family and one of the top ten DM algorithms(Wu, 2008), and its modifications: **AdaB.M1** and **AdaB.M2.**

Adaboost algorithm

Adaboost algorithm is part of the boosting family algorithms. It consists of training each of the base classifiers from the same training set, but in each iteration the weights are updated according to the examples that have failed

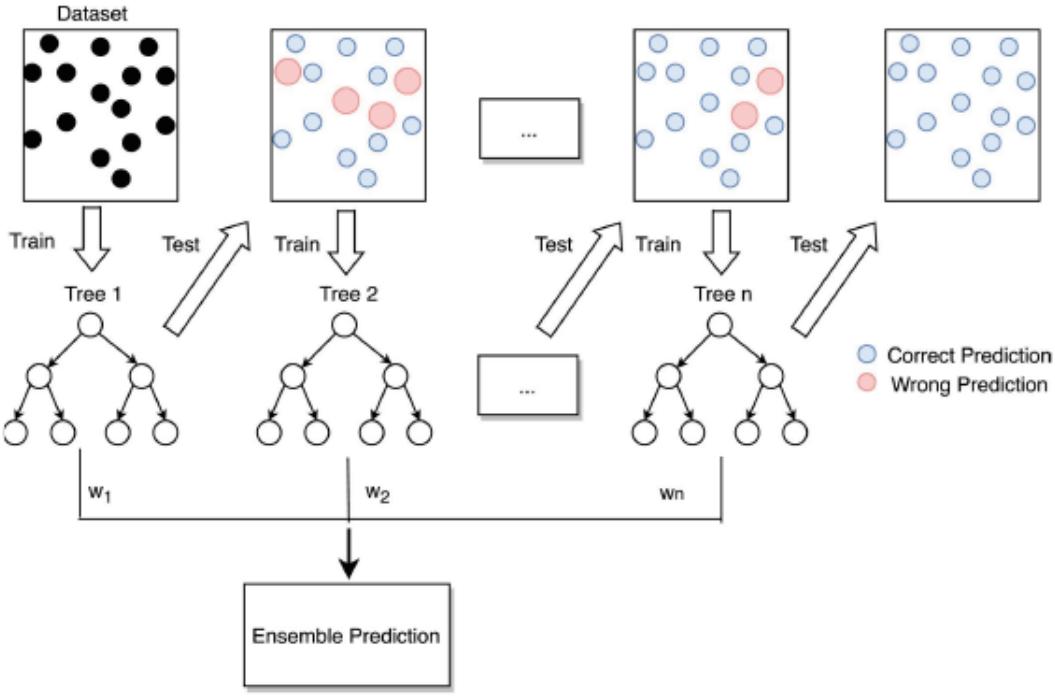

FIGURE 3.8: Description of boosting algorithm.

with the previously trained classifier, with more emphasis on those instances misclassified. The complete pseudo-code is mentioned in chapter four.

3.4 Ensemble Learning for Addressing the Class Imbalance Problem

Actually looking to the previous ensemble algorithms we will be able to deduce that neither of these algorithms by itself deals with the imbalance problem directly. Because they focus their attention on difficult examples (instances that are misclassified) without differentiating the class they belong to.

3.4.1 Different ensemble learning for imbalance data

We distinguish two different families(Cost-Sensitive Ensemble,) among ensemble approaches for imbalanced learning such as showing in figure below.

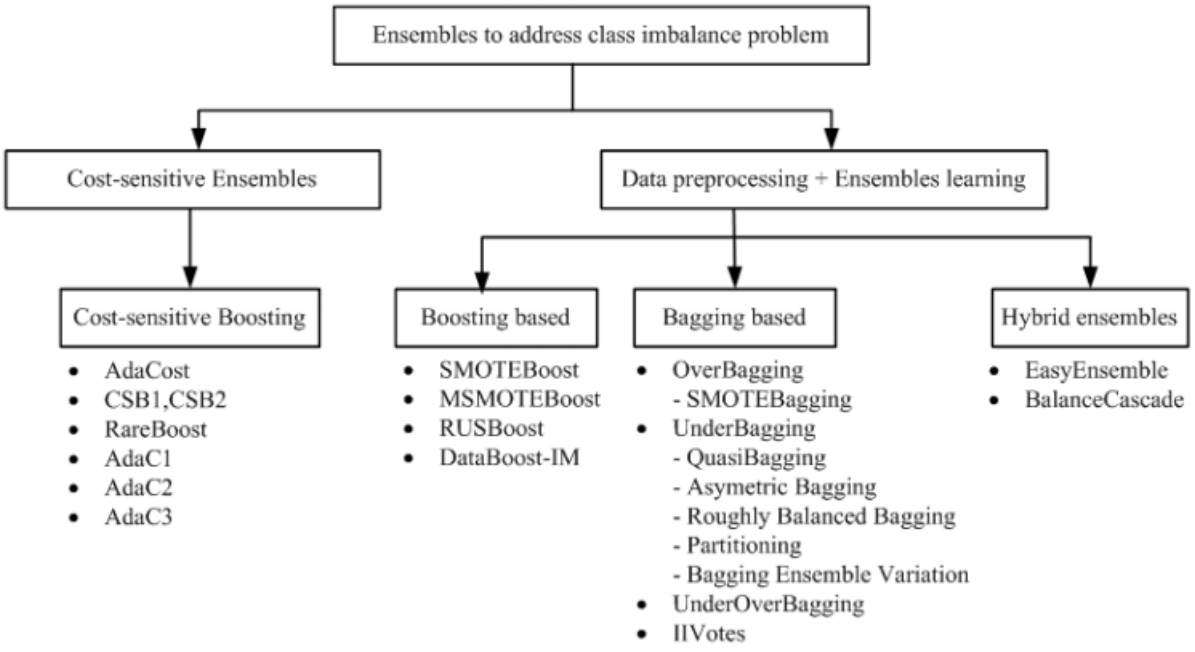

FIGURE 3.9: Ensemble methods to address calss imbalance problem.

1. **Cost-Sensitive Ensemble**

   In this type of ensembles, combinations of classifiers are learned with the usage of costs for each classes in the problem are considered. As we mentioned before adaboost algorithm assigne the different weight into missclassification in each round actually, this strategy biases the learning (the weights) toward the majority class for this reason, there are several proposals modifying the weight update of AdaBoost, so that examples from the different classes are unequally treated for example, cost-sensitive boosting approaches solved this problem by introduce cost items into the weight. Bellow, there are some categories of this approach that are differ in the way that they modify the weight update rule AdaCost (Fan, 1999), CSB1, CSB2 (Ting, 2000), RareBoost (Joshi, 2001), AdaC1, AdaC2 and AdaC3 (Sun, 2007)

   Including two ways:

   (a) **Cost-Sensitive Boosting**

   These approaches are similar to cost-sensitive methods, but instead of introducing the management of the costs into the classifier learning algorithm the costs minimization is guided by the boosting algorithm.

   (b) **Ensembles with Cost-Sensitive Base Classifiers:**

   Different from cost-sensitive boosting approaches, these ensemble models make use of cost-sensitive classifiers in order to make the ensemble capable of dealing with imbalanced classes.

2. **Data Preprocessing and Ensemble Learning**

   Ensembles in this category share a common characteristic. All of them consist of embedding a data preprocessing technique in an ensemble learning algorithm. The main difference lies in the base ensemble learning algorithm considered and the preprocessing method selected.

   (a) **Boosting-based**

   These methods use AdaBoost or any of its variants as ensemble method where a data level technique is introduced. These methods alter the weight distribution used to train the next classifier toward the minority class every iteration. This familly include: SMOTEBoost (Chawla, 2003), RUSBoost (Seiffert, 2009), MSMOTEBoost (Hu, 2009), DataBoost-IM (Guo, 2004), GESuperPBoost (García-Pedrajas, 2003), BalancedBoost (Wei, 2014), RB-Boost (Díez-Pastor, 2015), Balanced-St-GrBoost (Lusa, 2017) and RHBoost (Gong, 2017) algorithms.

   In chapter four, we will discuss in details some of these methods to deal with imbalanced problem.

   (b) **Bagging-based**

   In this case, Bagging is the ensemble method considered to be combined with a data level technique.

   (c) **Hybrid**

   In hybrid approaches both Bagging and Boosting are considered together and in combination with preprocessing mechanisms.

3.5 Overview on imbalanced classification with Multiple Classes

Imbalanced classification problem has been traditionally related to binary datasets such as (Branco, 2016)(He, 2009)(**272**)(Prati, 2015), where it contains a two classes negative class(majority class) and positive class(minority class).

However, in many real applications practitioners must deal with more than two classes for examples. For example, protein classification(Zhao, 2008), medical diagnosis(Chen, 2016), video mining(Gao, 2014). Actually, the presence of multi-minority and multi-majority classes will be a challenge we must to solve it.

The fact that multiple class imbalanced is represent a real problem we must be properly addressed(Haixiang, 2018). On the one hand, data level

solutions (preprocessing) are not directly applicable as the search space is increased, i.e. to determine the proper sampling rate for each class(Fernández-Navarro, 2011). On the other hand, algorithmic level solutions become more complicated since there can be more than one minority class(Zhou, 2005).

To solve the problem of multiple class imbalanced, we must stress the simple ways and the effective ways to maintain traditional binary-class imbalanced approaches to be applied in multi-class problems(Prachuabsupakij, 2014). There are two strategies in literature to achieve this. The first called **decomposition strategies**, The second called **ad-hoc strategies**.

Conclusion

In this chapter, we saw an overview on class imbalance problem in details. Furthermore, we discussed and summarized the most common approaches introduced in literature to solve the imbalanced data problem. Some of these approaches presented here will be verified and compared to prove their effectiveness in the next chapter.

Chapter 4

Experimental Studies

4.1 Introduction

In this last chapter, we have developed a experimental study comparing several boosting ensemble methods that we have carefully selected. In fact, our objective is two-fold: **(1)** We want to show the effectiveness of boosting ensemble methods by comparing them against the usage of preprocessing (without considering ensembles); **(2)** We present a comparison among boosting ensemble methods which show the usefulness of boosting ensemble methods to enhance their results.

we will approach the practical part of our project which consists of describing the technicalities of our proposed study, and we will mention the different tools and datasets used in our application.

4.1.1 Experimentation

4.1.2 Experimental Environment

All experiments were carried under the Linux Ubuntu operating system 20.04 ,with the tools and libraries located in the table below.

N^0	Tools	Type	Version
1	Python	Programming language	3.8.8
2	Jupyter NoteBook	open-source web application	6.3.0
3	Python 3.8.8 using Scikit-learn	Python library	0.22

TABLE 4.1: Tools and libraries used in experimentation.

4.1.3 Methods and Materials

A-Methods

A-1) Data preprocessing anf Ensemble learning Boosting-based ensembles(Data-level and Boosting algorithms)

We have selected **Boosting-based ensembles** the first kind of **Data preprocessing and Ensemble learning** approach to realize this experimental because they are the most widely used and are highly suitable for dealing with imbalanced datasets.

From this algorithmic structure, resampling techniques are integrated within the loop and prior to the training of the base classifier, which aim to improve learning for classification problems with unbalanced data sets.

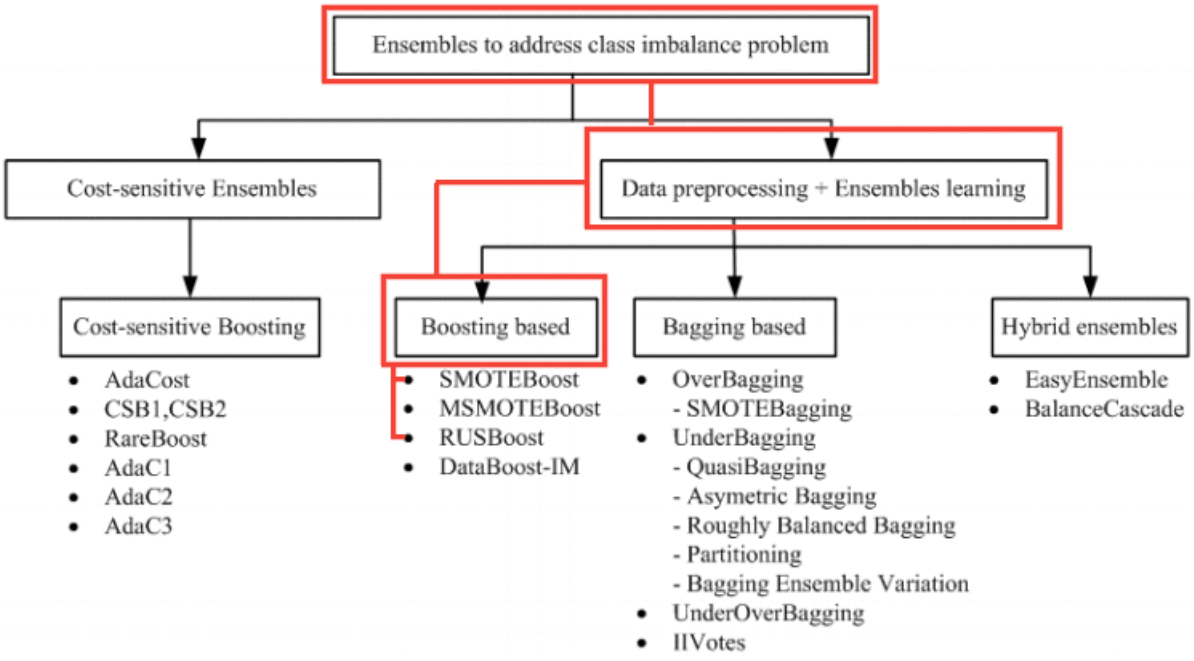

FIGURE 4.1: Ensemble methods to address calss imbalance problem.

Algorithm 1 The base: SMOTE

Input: Training set:$S = (x_i, y_i)$, where $i \in \{1, ..., N\}$ and $y_i \in \{-1, +1\}$; T:Number of iterations.

Output: $S_{synthetic}$.

procedure SMOTE
  $S_{syn} = \{\} = \varnothing$: Initialize the set of synthetic examples(empty set).
  for $x_i \in S$ where y_i =1 **do**
    for t=1 to n_{syn} **do**
      n_a = random_select(n_1,...,n_k).#Randomly select one of the k plus neighbors of x_i.
      for for j=1 to m do : **do** #For each attribute of x_i
        distance(j) = n_{aj}– x_{ij}.#Calculate the distance between x ij and its neighbor.
        x_Syn_j =x_{ij} + rand([0,1])*dist j. #Create value for attribute of synthetic example.
        $S_{synthetic}$ = $S_{synthetic} \cup \{x\_synthetic\}$#Add the new sample to the synthetic sample set.
      end for
    end for
  end for
end procedure

Algorithm 2 The base: AdaBoost

Input: Training set: $S = (x_i, y_i)$, where $i \in \{1, ..., N\}$ and $y_i \in \{-1, +1\}$; T:Number of iterations. I: base classifier.

Output: $H_{final} = \sin(f(x)) = \sin(\sum_{t=1}^{T}(\alpha_t h_t(x))$.
#Final classifier composed of multiples weak classifiers.

procedure ADABOOST
  $D_1(i) = \frac{1}{N}, \forall i \in \{1, ..., N\}$ Initial weight.

  for For i=1 to T: **do**
    $h_t = I(D_t, S)$ #Train weak classifier with weights.
    $E_t = \sum_{i, y \neq h_t(x_i)} D(i)$Calculate the error.
    $\alpha_t = \frac{1}{2}\ln(\frac{1-E_t}{E_t})$ Calculate the weight of the weak classifier.
    $D_{t+1^{(i)}} = D(i).\exp^{(-\alpha.h_t(x_i).y_i)} \forall i \in \{1, ..., N\}$ #Update weights.
    $D_{t+1^{(i)}} = \frac{D_{t+1}(i)}{\sum_{i=1}^{n} D_{i+1}(i)}$ #Normalize the weights.
  end for
end procedure

4.1.4 SMOTEBoost(Chawla)

SMOTEBoost, proposed by Chawla et al.(Bermejo, 2011), is a data-level method to deal with the imbalanced data problem, which combines the Synthetic Minority Oversampling Technique (SMOTE) and the standard boosting procedure. According to (Pratama, 2018).The pseudo code described below:

Algorithm 3 SMOTEBoost

Input: Training set $S = (x_i, y_i)$; T:Number of iterations ; I: base classifier.
$D_1(i) = \frac{1}{N}, \forall i \in \{1, ..., N\}$

Output: $H_{final} = \sin(f(x)) = \sin(\sum_{t=1}^{T}(\alpha_t h_t(x))$.
Final classifier composed of multiples weak classifiers.

1: **procedure** SMOTEBOOST
2: **for** i=1 to T **do**
3: $S_{syn} = SMOTE(S)$ Generate set of synthetic examples.
4: $S^{'} = S \cup S_{syn}$
5: $D_t^{'} = \left\{ d_i \in D_t : (x_i, y_i) \in S^{'} \right\}$ Weight distribution for selected examples.
6: $h_t = I(D_t^{'}, S^{'}) Trainweakclassifierwithweights.$
7: $E_t = \sum_{i, y \neq h_t(x_i)} D(i) Calculatetheerror.$
8: $\alpha_t = \frac{1}{2} \ln(\frac{1-E_t}{E_t}) Calculatetheweightoftheweakclassifier.$
9: $D_{t+1^{(i)}} = D(i).\exp^{(-\alpha.h_t(x_i).y_i)} \forall i \in \{1, ..., N\}$ Update weights.
10: $D_{t+1^{(i)}} = \frac{D_{t+1}(i)}{\sum_{i=1}^{n} D_{i+1}(i)}$ Normalize the weights.
11: **end for**
12: **end procedure**

4.1.5 RUSBoost

RUSBoost (Liu, (2005, October)), As its name suggests, it integrates within each iteration or boosting a random subsampling(remove instances) of the instances of the majority class. The pseudo code shown below:

Algorithm 4 RUSBoost

Input: Training set $S = (x_i, y_i)$; T:Number of iterations ; I: base classifier.
$D_1(i) = \frac{1}{N}, \forall i \in \{1, ..., N\}$

Output: $H_{final} = \sin(f(x)) = \sin(\sum_{t=1}^{T}(\alpha_t h_t(x))$
#Final classifier composed of multiples weak classifiers.

procedure RUSBOOST
  for i=1 to T **do**
    $S_{syn} = RUS(S)$Random undersampling of class instances majority.
    $S^{'} = S \cup S_{syn}$
    $D_t^{'} = \left\{ d_i \in D_t : (x_i, y_i) \in S^{'} \right\}$ Weight distribution for selected examples.
    $h_t = I(D_t^{'}, S^{'})$ Train weak classifier with weights.
    $E_t = \sum_{i, y \neq h_t(x_i)} D(i)$ Calculate the error as the sum of the weights of the wrong examples classified (from original data set).
    $\alpha_t = \frac{1}{2} \ln(\frac{1-E_t}{E_t})$ Calculate the weight of the weak classifier.
    $D_{t+1^{(i)}} = D(i).\exp^{(-\alpha.h_t(x_i).y_i)} \forall i \in \{1, ..., N\}$ Update weights.
    $D_{t+1^{(i)}} = \frac{D_{t+1}(i)}{\sum_{i=1}^{n} D_{i+1}(i)}$ Normalize the weights.
  end for
end procedure

4.1.6 Base classifier and Parameters

In first place, the baseline classifier that will be used in all the ensembles needs to be defined. With this goal, we will use **Decision Tree**generating algorithm. were proposed using **decision tree** as base classifier. Moreover, it has been widely used to deal with imbalanced datasets(García, 2009), and decisi tree on has also been included as one of the top-ten data-mining algorithms(Wu, 2008). Therefore, from our point of view it is the most appropriate base learning for these experiments. In addition, we use algorithm from the same family called **Radom Forest**.

Regarding the number of classifiers considered for the ensembles, all the methods should have the same opportunities to achieve their best performance, but avoiding the fine-tuning of their parameters depending on the dataset. In(Galar, 2011), it was shown that Boosting-based ensembles work better with only **10** classifiers. In our experimental we use our special deduction to estimate the number of estimators **T** and the number of synthetic samples of SMOTE algorithm **N** :
Trade-off between Majority class and Minority class(two classes equal) :
Majority class= Minority class +**T:number of estimators**$\times$**Synthetic samples N.**

B-Materials
B-1) Datasets

Table 4.2 summarizes all the datasets used in this study along with their respective properties. For each of the datasets, we show Dataset name, Total number of instances, attributes (Real(R)/Integer(I)/Nominal valued(N)) and imbalance ratio value. between **1.8** and **129**. In this study, we consider **IR** as the ratio of percentages of Majority to Minority instances, i.e. I R = $\frac{\%Maj}{\%Min}$. All the datasets are real-world coming from different domains, they were prepared for binary classification tasks in mind and they are publicly available on KEEL **KEEL(Knowledge Extraction based on Evolutionary Learning)** data-set repository, which is publicly available on the corresponding web-page[1] and (Alcalá-Fdez, 2011).

Imbalance ratio(IR): According to this web-page[2] indecate the degree of imbalance of a two-class problems.

$$\textbf{Imbalance ratio(IR)} = \frac{N^0\ of\ instances\ of\ the\ majority\ class}{N^0\ of\ instances\ of\ the\ minority\ class} \tag{4.1}$$

1. **1.5** $\leq$**IR**$\leq$**3**: The imbalance ratio (IR) is lower between the minority classes(positive) and majority classes(negative).

[1]https://sci2s.ugr.es/keel/imbalanced.php
[2]https://www.intechopen.com/chapters/70393#B19

2. **3<IR≤9**:The imbalance ratio (IR) is middle between the minority classes(positive) and majority classes(negative).

3. **IR>9**:The imbalance ratio (IR) is higher between the minority classes(positive) and majority classes(negative).

2.Imbalanced data sets available on KEEL web site In our experimental we will take different datasets with defferents **IR**.

Imbalanceratio(IR): 1.5≤ IR ≤ 3					
Datasets name	Attributes (R/I/N)	Instances	Origin	Missing values	IR
vehicle1	18 (0/18/0)	846	Real world	No	2.9
Imbalnace ratio(IR): 3<IR≤9					
segment0	10 (19/0/0)	2308	Real world	No	6.02
Imbalnace ratio(IR):IR>9					
shuttle-6_vs_2-3	9 (0/9/0)	230	Real world	No	22

TABLE 4.2: Summary of real-world binary imbalanced datasets sorted by IR

2.Evaluation metrics

Such as mentioned in literature paper(Huang, 2005), the accuracy is better metric to evaluate machine learning approaches in balanced datasets, it is no longer a proper measure in the imbalance scenario. because it will be liar in his evaluation and do not give us the truth about the performance of models. It does not distinguish between the numbers of correctly classified examples of different classes. The accuracy formula is as follows:

$$Accuracy = \frac{Number\ of\ correct\ prediction}{Total\ number\ of\ prediction} \tag{4.2}$$

Image the two scenario below for binary classification problem,sice:

N:Totale sample(observation):**1000** samples.

Negative class: 700 Majority classs.

Positive class:300 Minority class.

$$\textbf{Imbalance ratio(IR):} = \frac{700}{300} = 2.33 \quad (\textbf{Maj} \quad 2 : \textbf{Min} \quad 1) \tag{4.3}$$

	Negative	Positive
Negative	600	100
Positive	100	200

TABLE 4.3: Honeste accuracy.

	Negative	Positive
Negative	900	0
Positive	100	0

TABLE 4.4: Liar accuracy.

Selon the formula(1),the accuracy of two table will be such as:

$$\textbf{Accuracy}\,(tab1) = \frac{600+200}{1000} = 80\%. \quad \textbf{Accuracy}\,(tab2) = \frac{900+0}{1000} = 90\%.$$

Interpretation of accuracy result

In fact, the accuracy of left table**(tab1)** decrease and the accuracy of right table **(tab2)** increase, but clearly the model in table one can strongly distinguish between the majority class**(600)** and the minority class **(200)** and he learn from the both classes, so it accuracy is honeste. In contrast, the model in table one can not be distinguish between the majority class and the minority class. In the reality, the model learn only from the majority class **(900)** and ignore the minority class and this is big trick because all of us we think that the higher accuracy is best and ignore the truth that model learn only from one class (majority class) and in most cases there are usefull information in minority class can aid us to increase prediction.

Metrics for imbalanced classification

In order to avoid the accuracy trick and measure the performance of each approach and to assess the quality of the models we consider the different metrics in the litterateur, which requires both classes to be properly distinguished without favoring any of them. The most common metrics to evaluate imbalanced datasets are mentioned below:
ROC/AUC curve,**Precsion-recall(PR) Curve**, **G_means (geometric-mean)**, $F_1 - Score$, **precision**,**recall**.

(a) $F_1 - measure$**:** Is a measure that focus on the analysis of positive class, and is widely used in information retrieval. Its objective is to analyze the trade-offs between correctness and coverage in classifying positive instances. To this end, the measure uses a weighed harmonic mean between positive predictive value and true positive rate.

$$F_1 - measure = 2\frac{precision \times recall}{precision + recall} \tag{4.4}$$

(b) **Precision:** Evaluates the fraction of correct classified instances among the ones classified as positive.

$$\textbf{Precision} = \frac{True \quad positives(TP)}{True \quad positives(TP) + False \quad positives(TP)} \tag{4.5}$$

(c) **Recall:** Is the fraction of total positive instances correctly classified as positive.

$$\textbf{Recall} = \frac{True \quad positives(TP)}{True \quad positives(TP) + False \quad negatives(FN)} \tag{4.6}$$

(d) **G_means:** Geometric mean, uses information from both classes, it is the geometric mean of true positive and true negative rates (G-mean). This measure aims at a balance between classification performances on the majority and minority classes. A poor performance in prediction of the positive examples will lead to a low G-mean value, even if the negative examples are correctly classified.

$$\textbf{G\_means} = \sqrt{TPR.TNR} \tag{4.7}$$

(e) **ROC/Auc curve:** As mentioned in literature (Branco, 2015) Roc/Auc curve is a graphical evaluation method that is not dependent of a specific threshold. A ROC graph is a plot of False Positive Rate (FPR) on the x-axis, and True Positive Rate (TPR) on the y-axis. Given the ranked list of instances according to scores, to draw the graph, the threshold is varied from the most restrictive (highest score) to the most lenient (lowest score). For each possible value of the threshold, there is a point in the ROC space based on the values of FPR and T P R for that threshold. The curve is drawn by linear interpolation among these points .

- Good classifier should reach as close to the top left corner as possible.
- The upward diagonal indicates random performance.
- All points in the ROC curve should lie above this diagonal.

- points below the diagonal indicates performance worse than random.
- The lower left corner corresponds to a classifier which always predict the wrong class.
- The lower left corner (origin) corresponds to always predicting the negative class.
- The top right corner corresponds to always predicting the positive class.

(f) **Precision-Recall(PR) curve:** A curve is more adequate for imbalance context, this curve uses in the x-axis the Recall (also known as TPR), and P recision (also known as PPV) in the y-axis and focus in the positive class. The interpretation of PR graph is slightly different from the ROC graph.

Good classifiers should be as close as possible to the top right, as this corner represents the best trade-off between precision and recall (Saito, 2015)(Boyd, 2013).

4.1.7 Experimental Design

In this work, the selected technique that will be carried out for the evaluation of models is called **'k-fold cross validation (KFold-CV)'**. It

is a cross-validation technique that divides the original data set into **k** subsets. The number of partitions that will be taken into account in our experimental will be **k = 10**, a value that tends to be very common for this parameter when using this evaluation method. Subsets generated from the original dataset must have the same number of samples, or at least less similar. This partition will be made randomly and there must be an empty intersection between the generated subsets, that is, each instance of the original dataset will be assigned to only one subset. It is important that the distribution of the classes in the subsets for **cross-validation (CV)** is similar to that of the original data set, so the partition, in addition to being random, will be created using a stratification technique(use a specific kind called **'StratifiedKFold'**), which is a type of technique that pursues this last objective.

Once the **k** subsets of the original dataset have been obtained, we proceed to carry out **k** trainings of the model to be learned. Each training will be carried out using **k-1** subsets of those generated and the rest will be left to perform the test of that model. After performing all the tests. For each of the k models, the performance of the classifier will be given by the average of the results obtained in each of the tests. Each of these experiments are done with the range of **[10,100] iteration** independent with **10-fold** cross validation and acquires the average results in terms of**Roc/AUC curve**, **Precsion-recall curve**, **G-means**, **Recall**, **Precsion**, **F-measure** respectively.

Through this procedure, we obtain a performance for the learned classification model that guarantees that the results obtained are independent of the data with which said model has been trained, that is, that it reflects the generalizability of the classifier in the face of new data.

The following figure shows the **10-fold** cross validation procedure graphically:

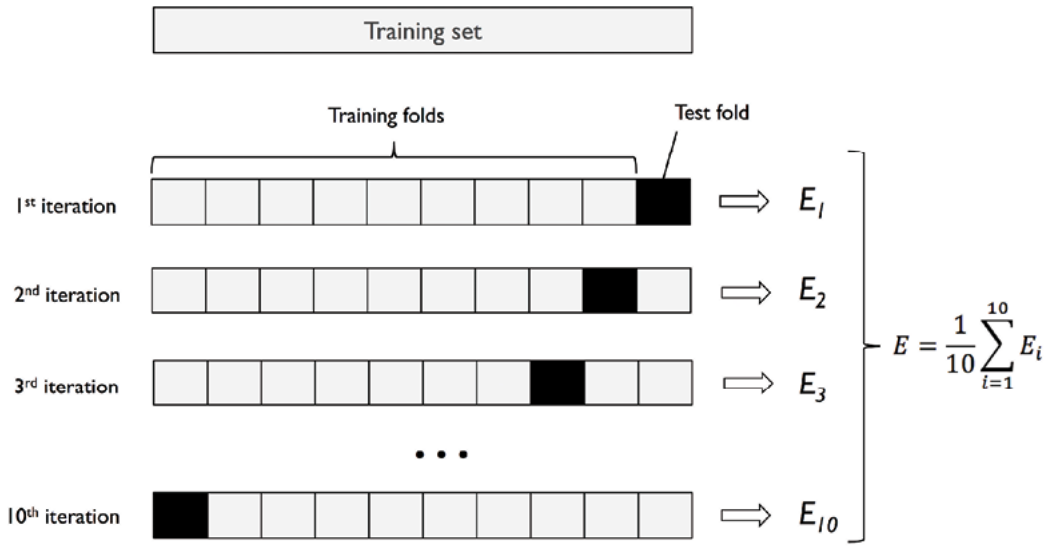

FIGURE 4.2: Workflow for the **10-fold** cross validation technique.

In this case, the performance of the model is evaluated based on the geometric mean, for which the average of those obtained in the ten training and tests that have been carried out is calculat.

4.2 Experimental Results and Discussion

4.2.1 Experimental results with: 1.5<=IR<=3 on Vehicle1 dataset

-Vehicle1 dataset and SMOTE algorithm

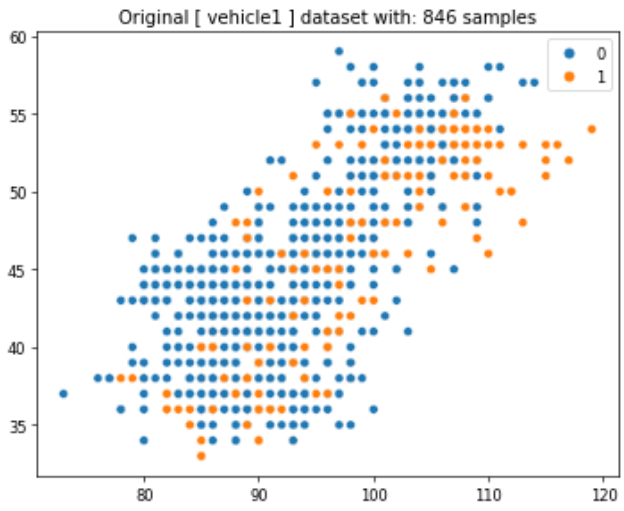

FIGURE 4.3: Original **vehicle1** dataset with 846 samples.

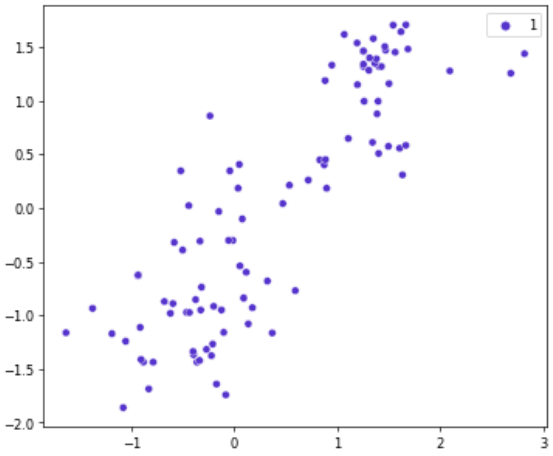

FIGURE 4.4: Round **(1)** Smote Created **100** instances.

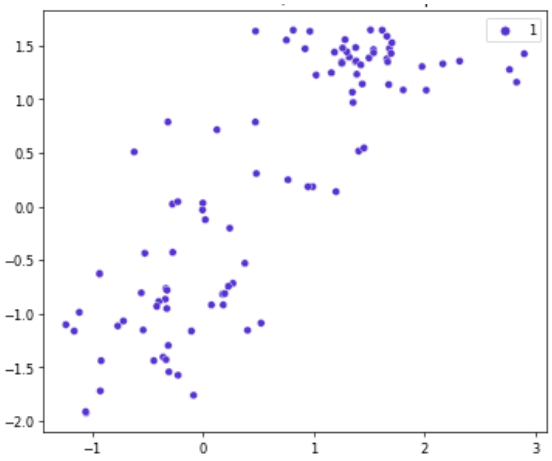

FIGURE 4.5: Round **(2)** Smote Created **100** instances.

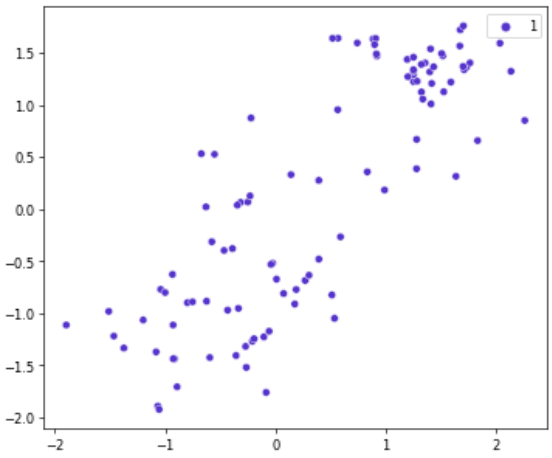

FIGURE 4.6: Round **(3)** Smote Created **100** instances.

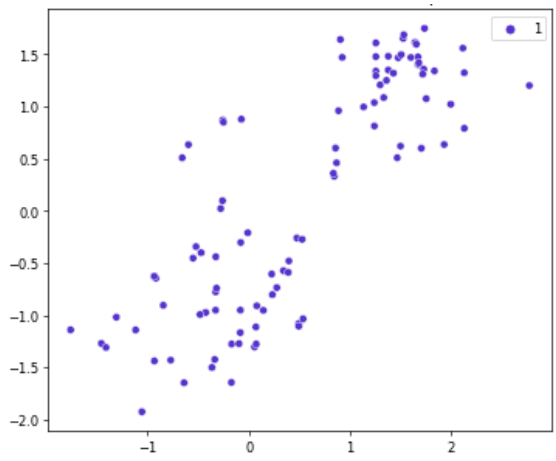

FIGURE 4.7: Round **(4)** Smote Created **100** instances.

The tables below show the results for performance metrics: for geometric mean (GM), Recall, Precsion, and f1-score(F1) with 10-fold(CV) each thes **10-fold** done **n** iterations on **vehcile1 Dataset**.

K-iteration (10-CV)	Decsion-Tree							
	(N=100,T=4)				(N=200,T=2)			
	F1-score	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*
Avg-[10]	0.237	0.571	0.369	0.181	0.430	0.638	0.616	0.363
Avg-[20]	0.239	0.572	0.372	0.183	0.436	0.641	0.621	0.368
Avg-[30]	0.238	0.572	0.370	0.182	0.434	0.640	0.620	0.366
Avg-[40]	0.238	0.572	0.370	0.182	0.437	0.641	0.622	0.369
Avg-[50]	0.239	0.572	0.371	0.182	0.436	0.641	0.620	0.368
Avg-[60]	0.239	0.572	0.371	0.182	0.435	0.640	0.620	0.367
Avg-[70]	0.239	0.572	0.371	0.182	0.437	0.641	0.622	0.369
Avg-[80]	0.238	0.572	0.369	0.182	0.435	0.640	0.620	0.368
Avg-[90]	0.239	0.572	0.371	0.182	0.435	0.640	0.620	0.367
Avg-[100]	0.239	0.572	0.371	0.182	0.435	0.640	0.620	0.367
Average	0.2385	0.5719	0.3705	0.182	0.435	0.6402	0.6201	0.3674

TABLE 4.5: Summary average of of **Decision-Tree** Result on **Vehicle1** dataset.

K-iteration (10-CV)	DT-SMOTESBoost							
	(N=100,T=4)				(N=200,T=1)			
	F1-score	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*
Avg-[10]	0.560	0.712	0.479	0.682	0.571	0.723	0.475	0.729
Avg-[20]	0.572	0.722	0.487	0.700	0.577	0.728	0.482	0.732
Avg-[30]	0.565	0.716	0.484	0.688	0.573	0.726	0.475	0.735
Avg-[40]	0.562	0.715	0.484	0.683	0.574	0.726	0.480	0.726
Avg-[50]	0.566	0.717	0.483	0.693	0.573	0.725	0.478	0.726
Avg-[60]	0.561	0.713	0.481	0.683	0.574	0.725	0.478	0.728
Avg-[70]	0.562	0.714	0.481	0.686	0.574	0.726	0.479	0.727
Avg-[80]	0.564	0.716	0.482	0.690	0.575	0.726	0.479	0.731
Avg-[90]	0.562	0.715	0.479	0.690	0.573	0.725	0.477	0.731
Avg-[100]	0.563	0.715	0.481	0.689	0.574	0.726	0.478	0.730
Average	0.5637	0.7155	0.4821	0.6874	0.5738	0.7256	0.4781	0.7293

TABLE 4.6: Summary average of of **SMOTEBoost** Result on **Vehicle1** dataset.

K-iteration (10-CV)	DT-RUSBoost							
	(N=100,T=4)				(N=200,T=2)			
	F1-score	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*
Avg-[10]	0.567	0.722	0.451	0.772	0.573	0.727	0.462	0.774
Avg-[20]	0.565	0.722	0.451	0.770	0.569	0.725	0.451	0.784
Avg-[30]	0.567	0.723	0.451	0.776	0.566	0.722	0.452	0.774
Avg-[40]	0.567	0.722	0.456	0.762	0.568	0.723	0.452	0.777
Avg-[50]	0.564	0.719	0.452	0.761	0.572	0.727	0.454	0.786
Avg-[60]	0.564	0.719	0.453	0.759	0.569	0.725	0.453	0.780
Avg-[70]	0.563	0.718	0.448	0.768	0.567	0.723	0.452	0.775
Avg-[80]	0.563	0.719	0.449	0.766	0.566	0.723	0.449	0.779
Avg-[90]	0.568	0.723	0.454	0.771	0.565	0.721	0.450	0.773
Avg-[100]	0.565	0.720	0.452	0.765	0.567	0.723	0.451	0.776
Average	0.5653	0.7207	0.4517	0.767	0.5682	0.7239	0.4526	0.7778

TABLE 4.7: Summary average of of **RUSBoost** Result on **Vehicle1** dataset.

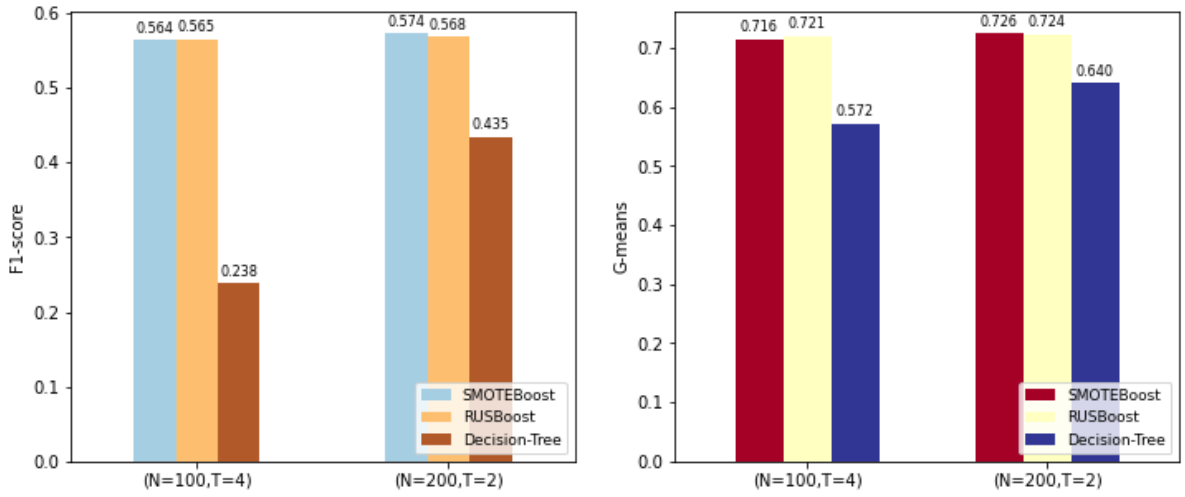

FIGURE 4.4: F1-score and G-means on the **Vehicle1** dataset.

Discussion

Tables 4.5, 4.6 and 4.7 and from Figure 4.8 show the classification performance in terms of F-measure, G-means, Rcall, Precsion, It was obtained using Decion Tree with ensemble boosting methods techniques. As shown in the results, The proposed SMOTE-Boost and RUSBoost methods using Decision-Tree algorithm showed satisfactory results (best performance). Also DT-SMOTEBoost has a F1-score (0.5637) better than single DT-SMOTEBoost (without SMOTEBoost). Also DT-RUSBoost has better F1-score**(0.5653)** better than single Decsion-Tree (without SMOTEBoost)**(0.2385)**. In addition for the rest metrics Boosting ensembles methods(DT-SMOTEBOOST,DT-RUSBoost) performed better than single Decsion-Tree (without boosting ensmble methods).

1. Roc-Auc Curve:Result Visualization with(N=100,T=4) on vehicle1 dataset

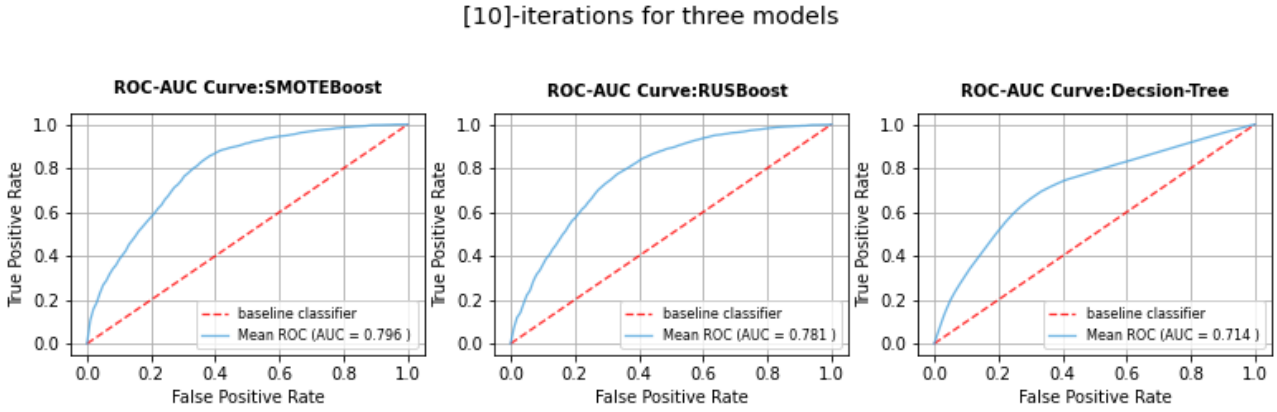

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

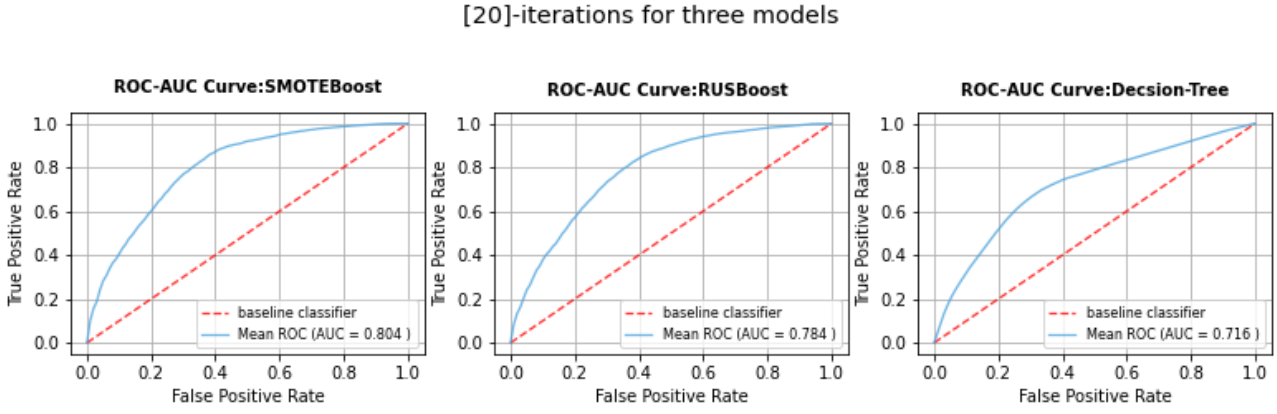

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

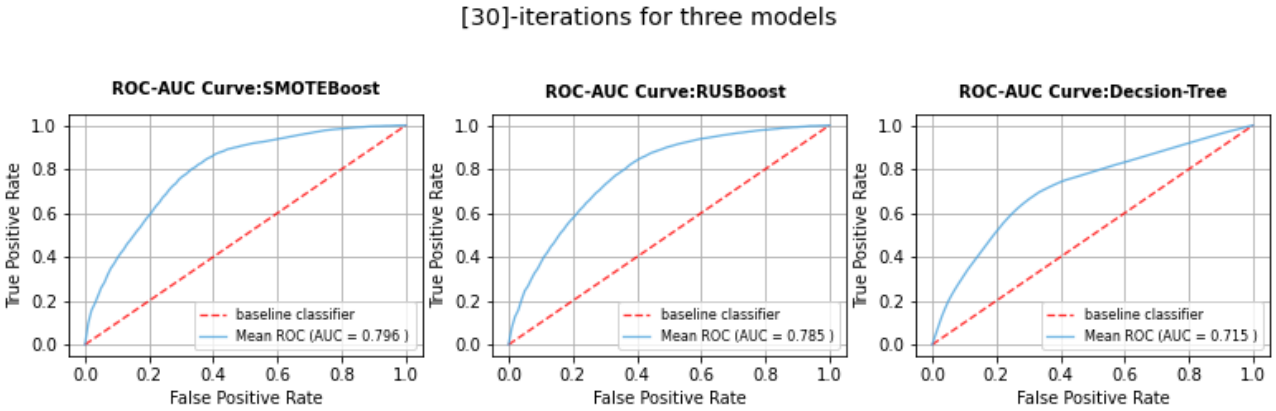

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

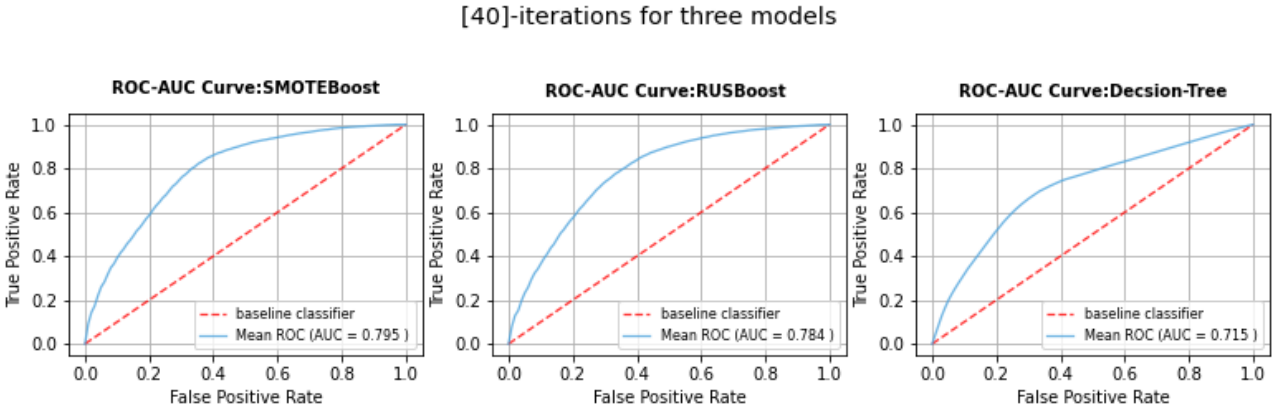

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

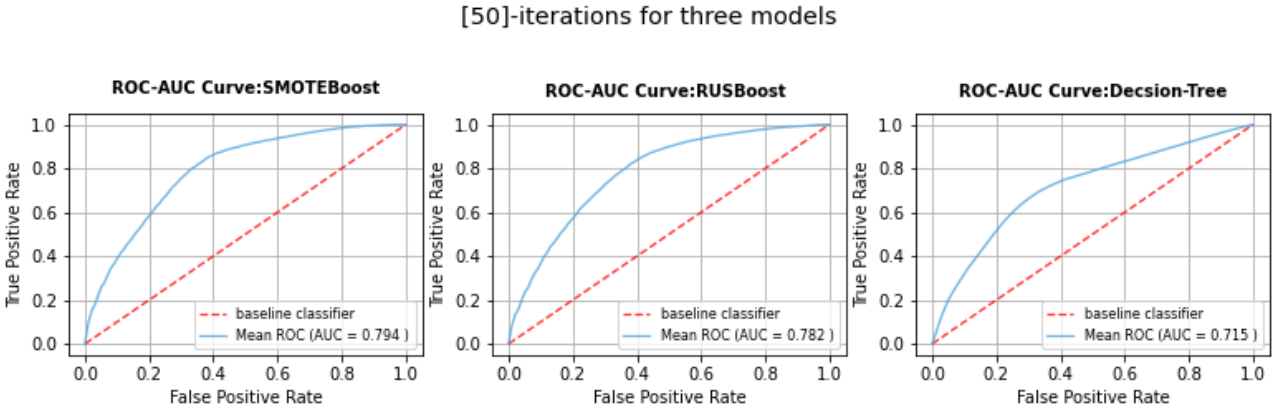

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

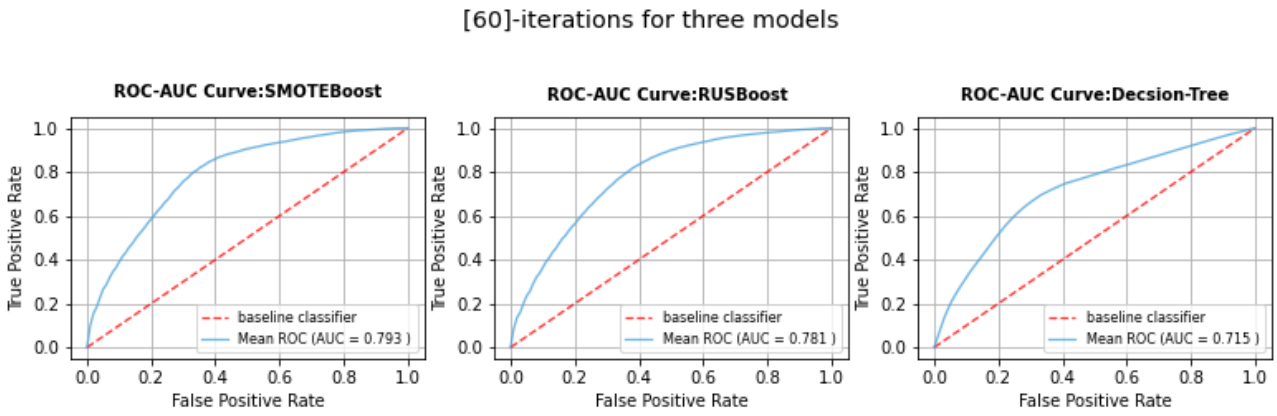

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

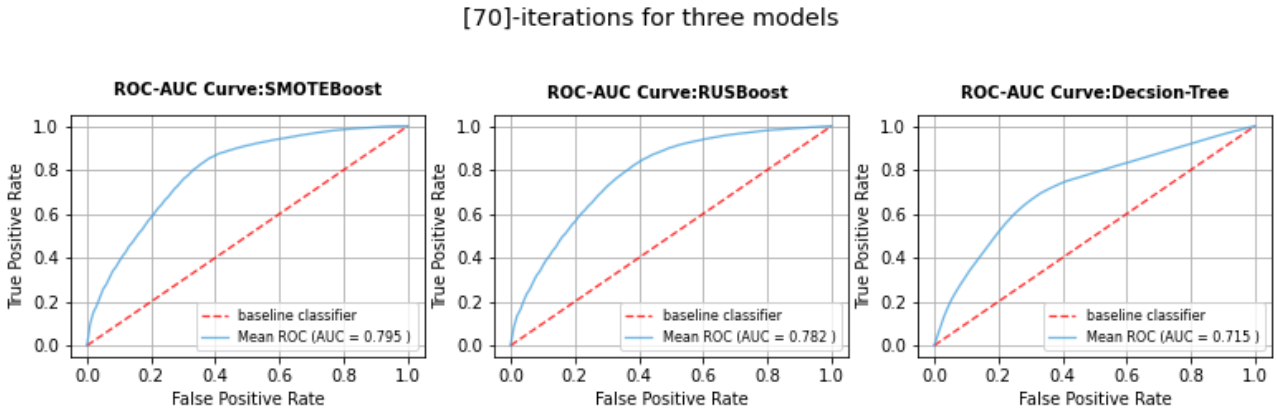

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

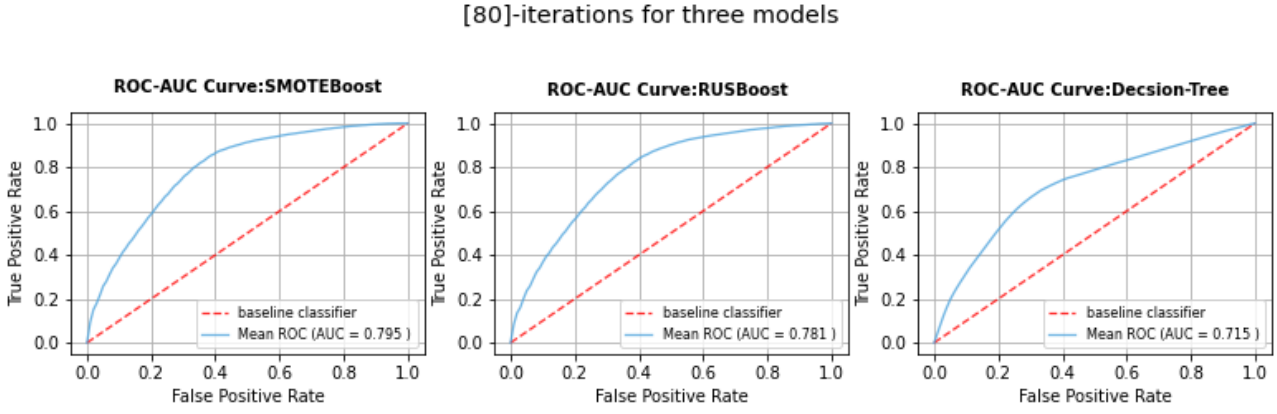

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

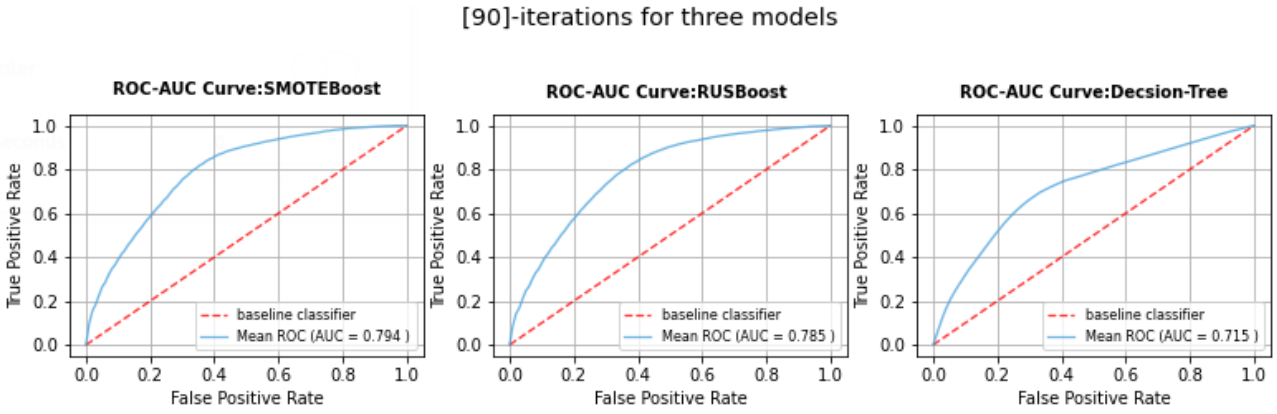

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

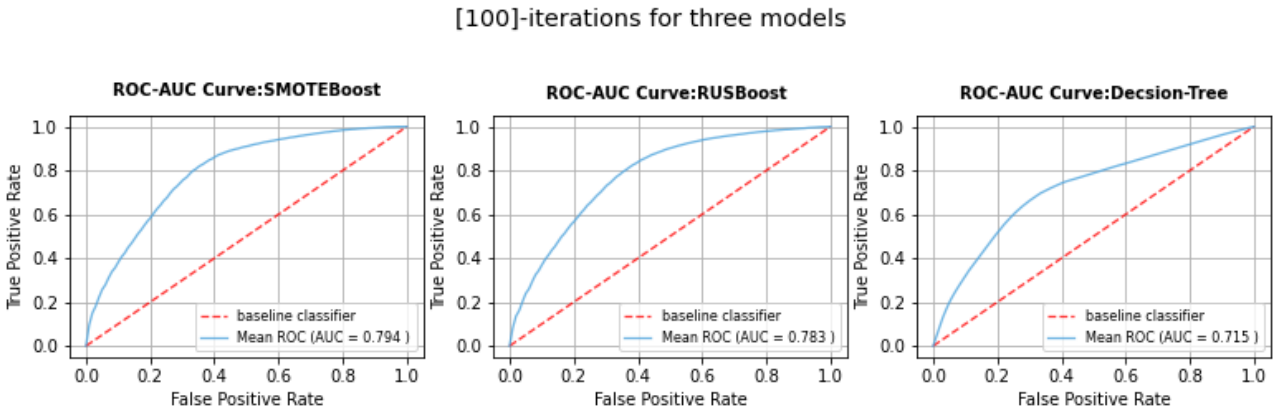

FIGURE 4.4: Roc/Auc Curve with **(N=100,T=4)** parameters.

Discussion

The Roc/Auc graph above represents the models' ability to differentiate between classes of data sets (between the majority class and the minrity class), the high value of Auc (Area Under the ROC Curve) indicates that models can differ strongly between classes. Our curve clearly shows that the auc value of DT-SMOTEBoost and DT-RUSBoost is higher that the individual decision tree models (effectiveness of boosting methods).

2. Roc-Auc Curve:Results Visualization with(N=200,T=2) on vehicle1 datasets

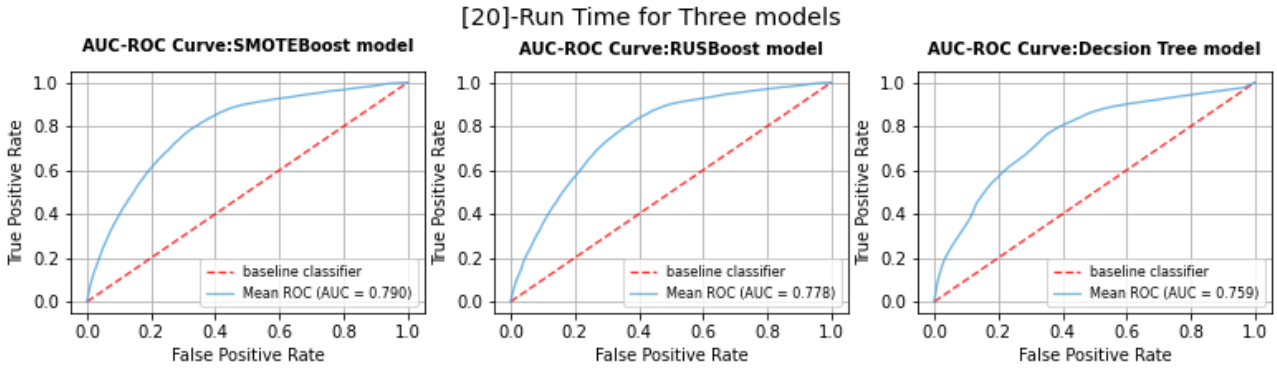

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

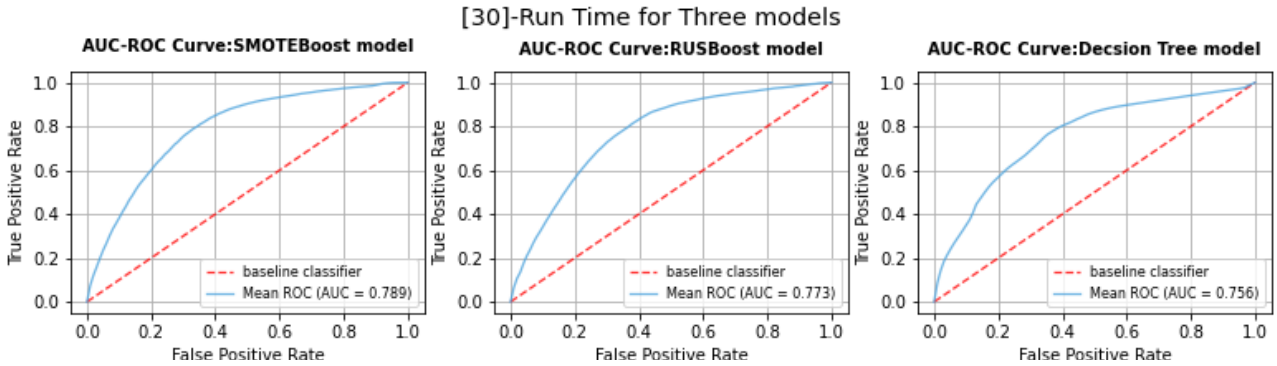

FIGURE 4.4: Roc/Auc Curve with**(N=200,T=2)** parameters.

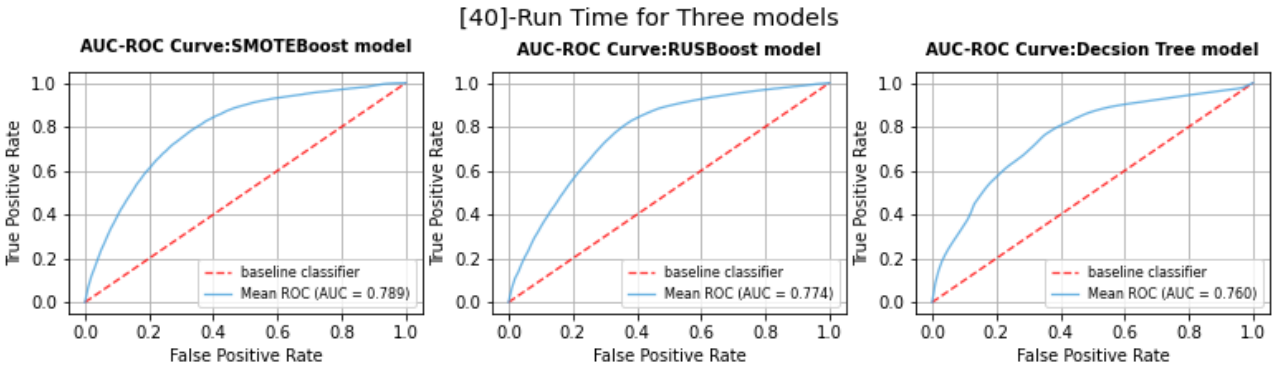

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

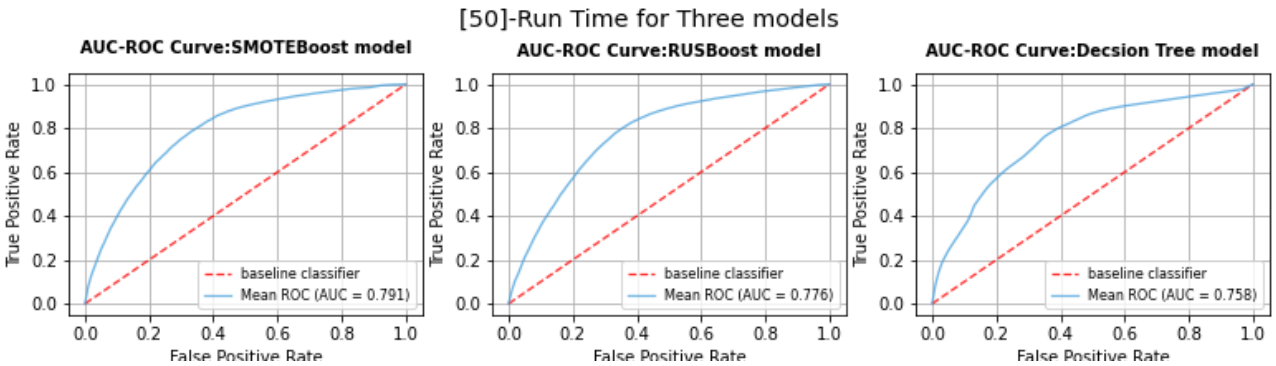

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

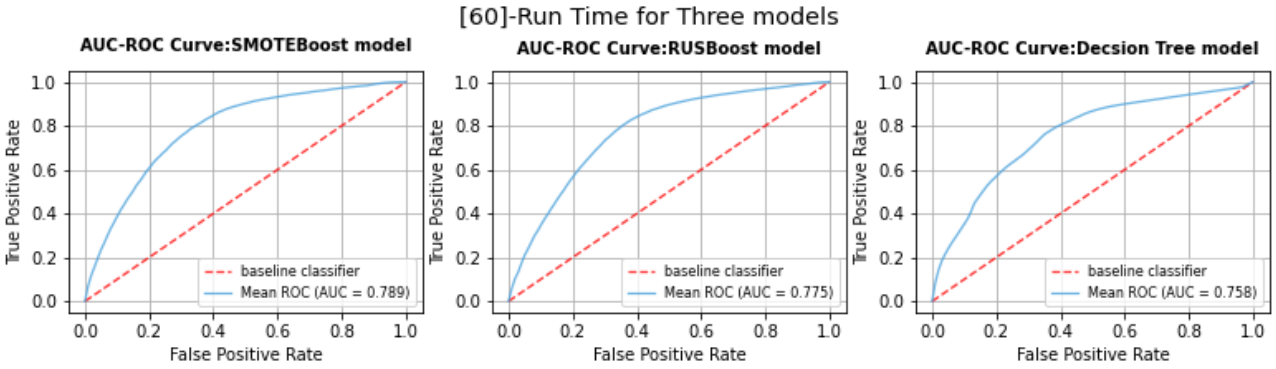

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

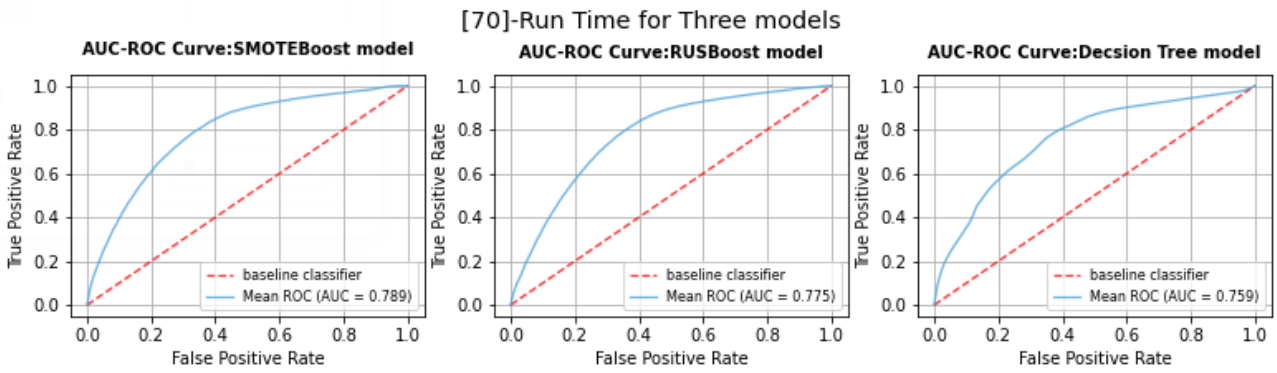

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

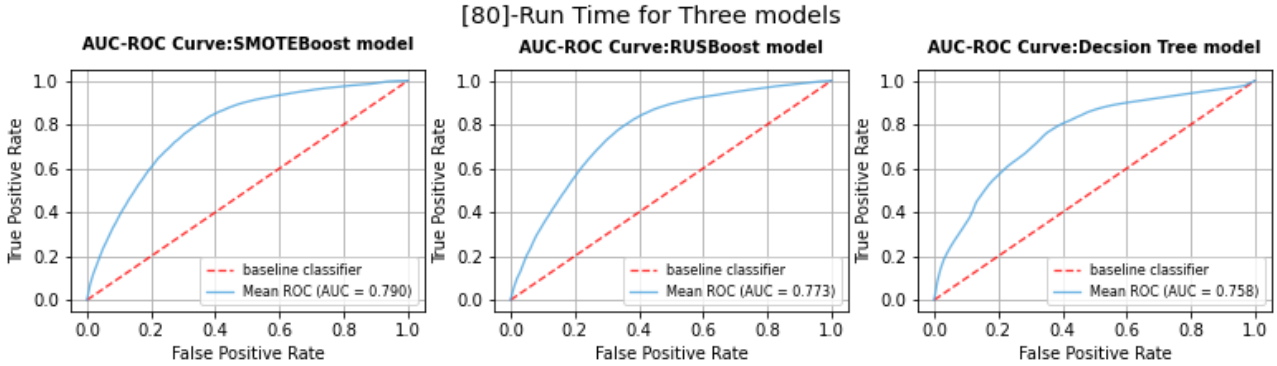

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

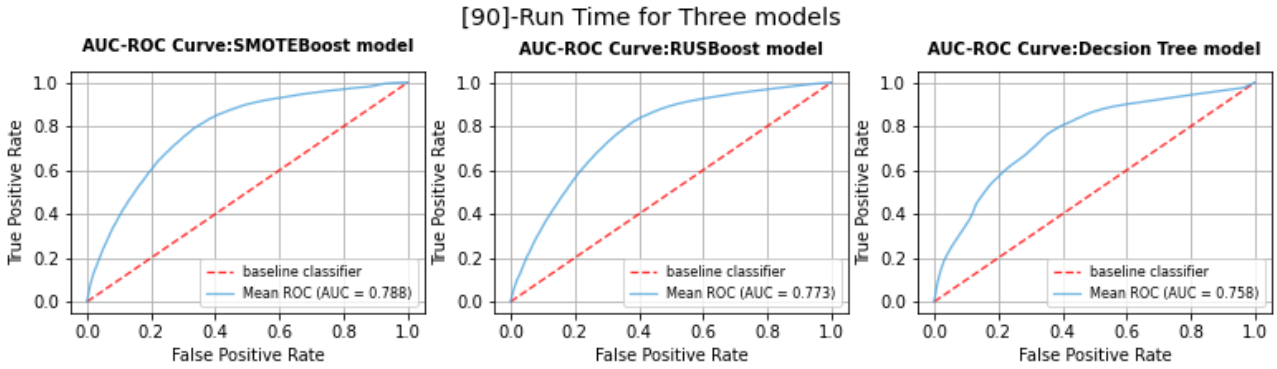

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

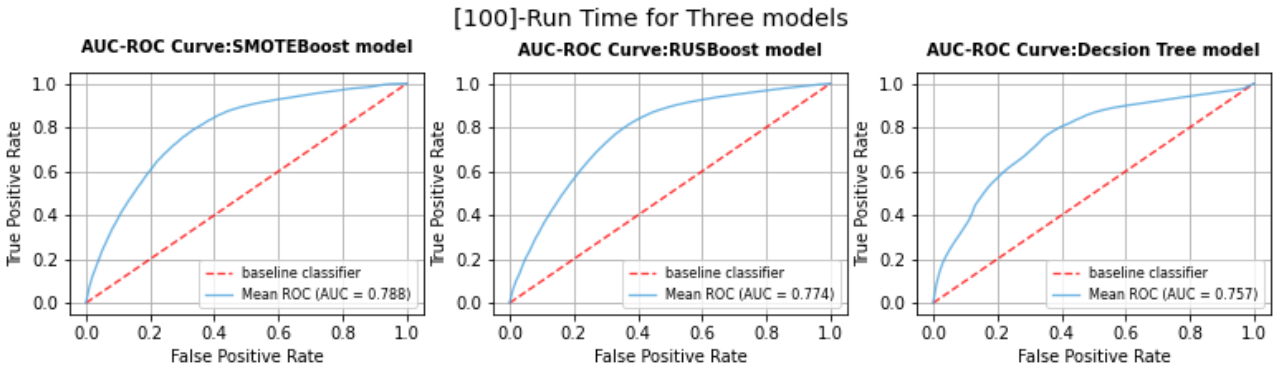

FIGURE 4.4: Roc/Auc Curve with **(N=200,T=2)** parameters.

3. Precion-Recall Curve:Results Visualization with(N=100,T=4) on vehicle1 datasets

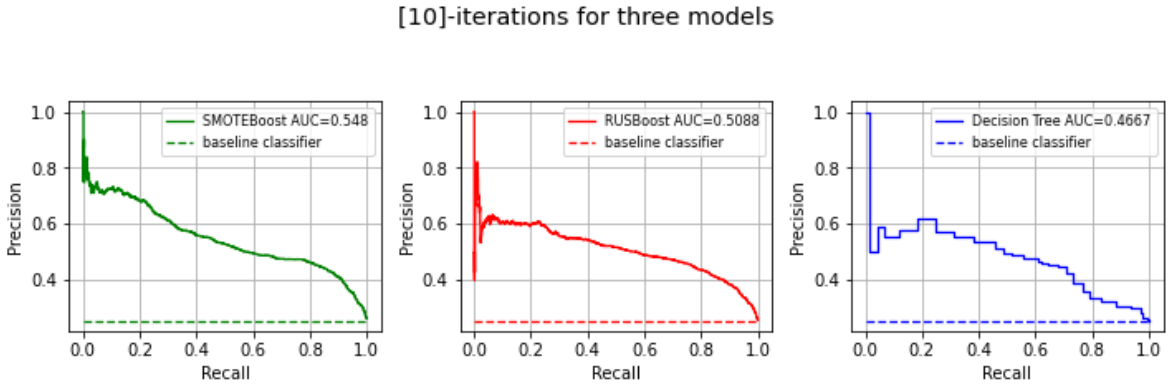

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

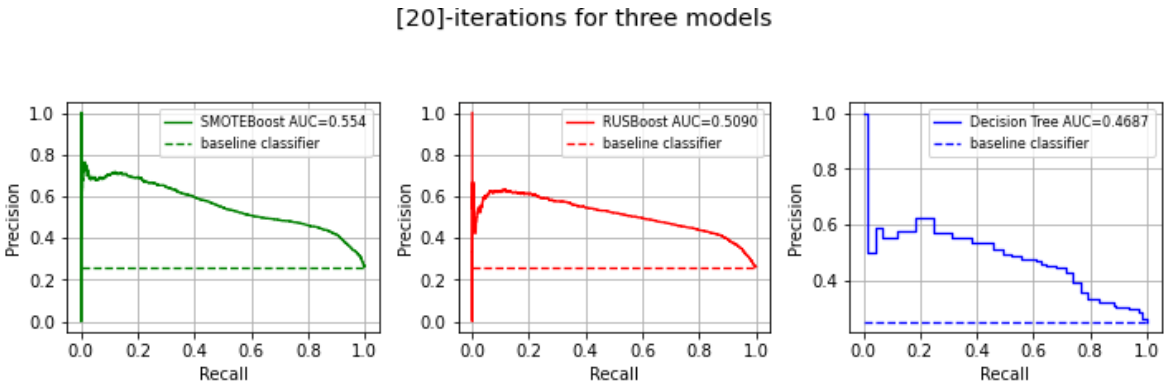

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

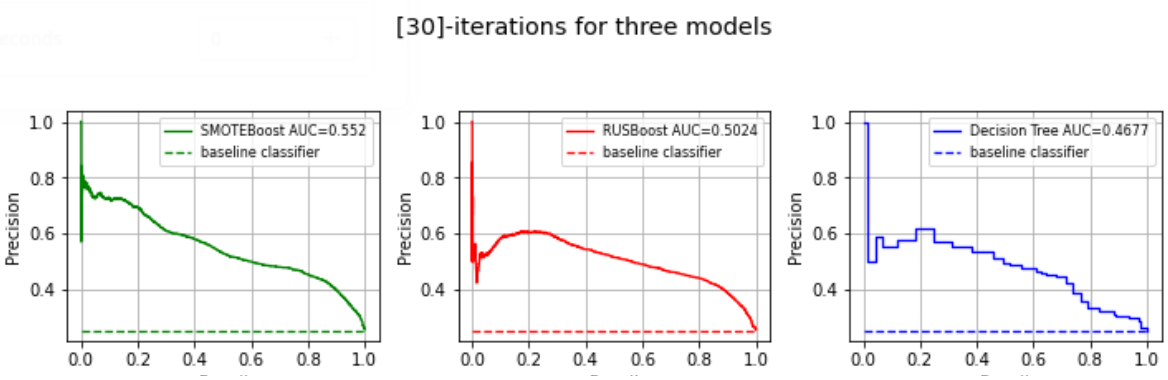

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

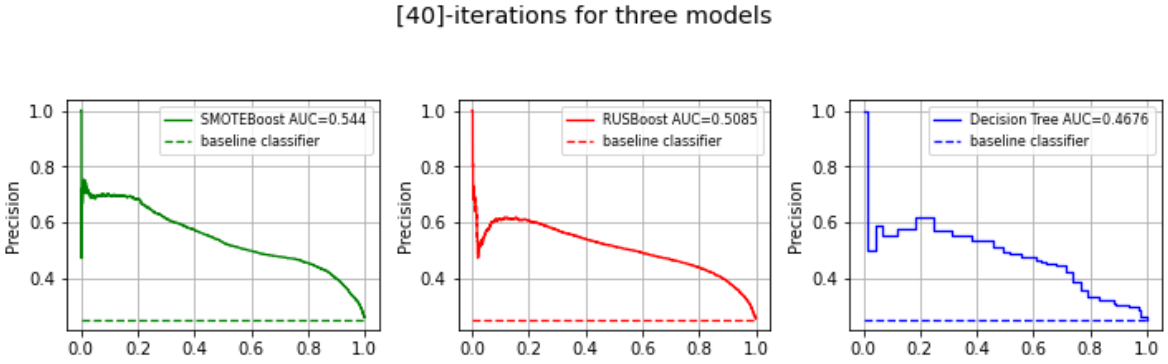

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

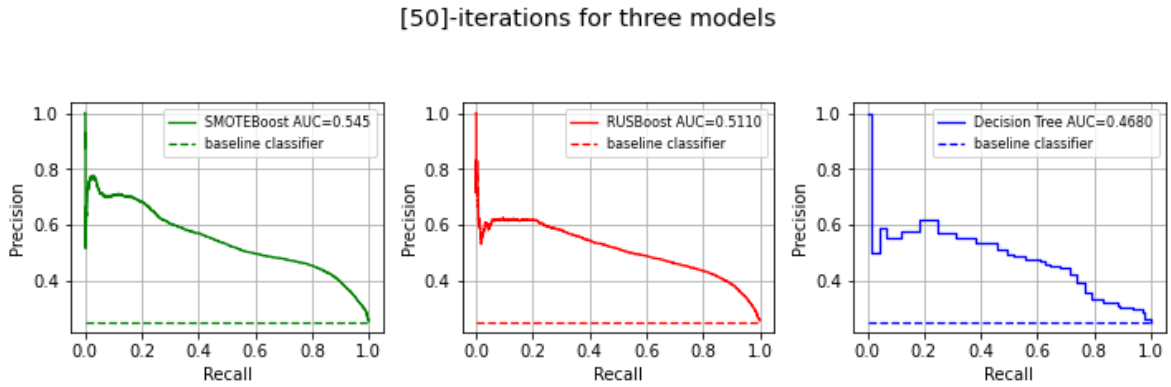

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

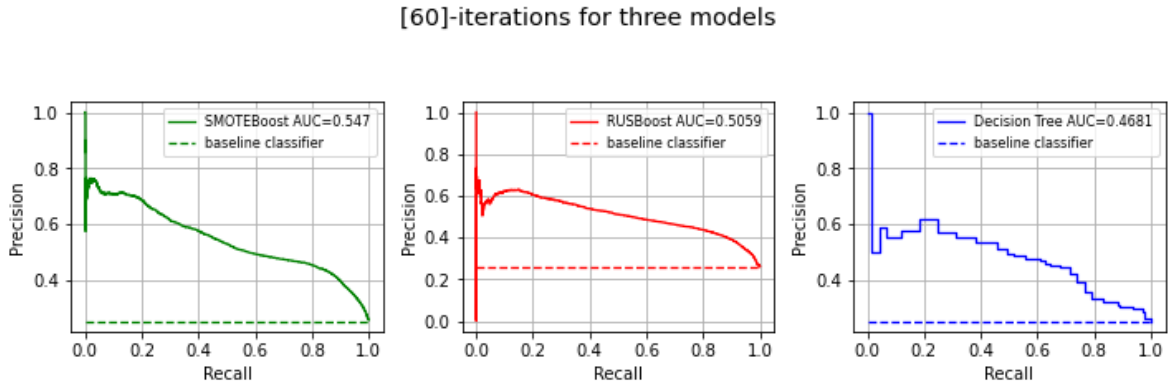

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

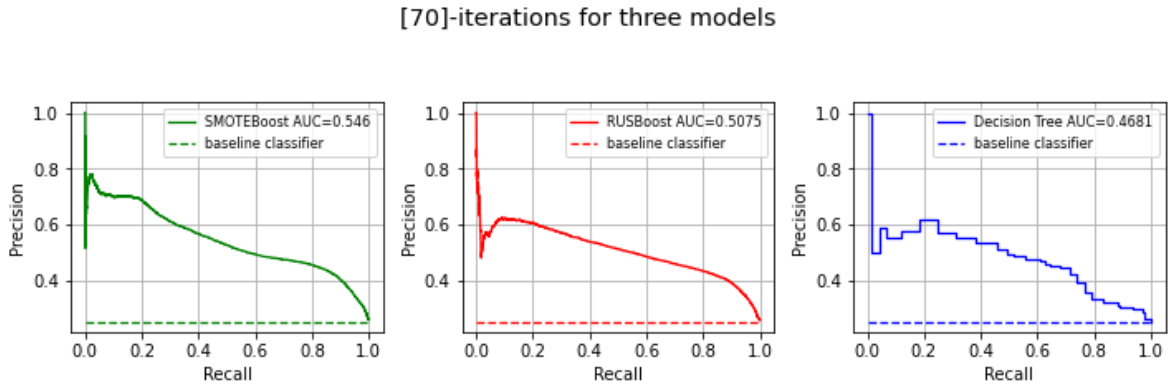

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

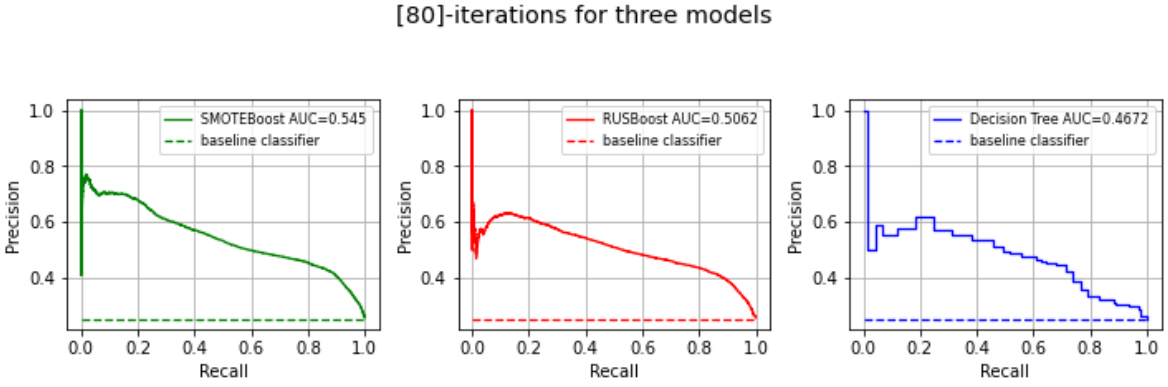

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

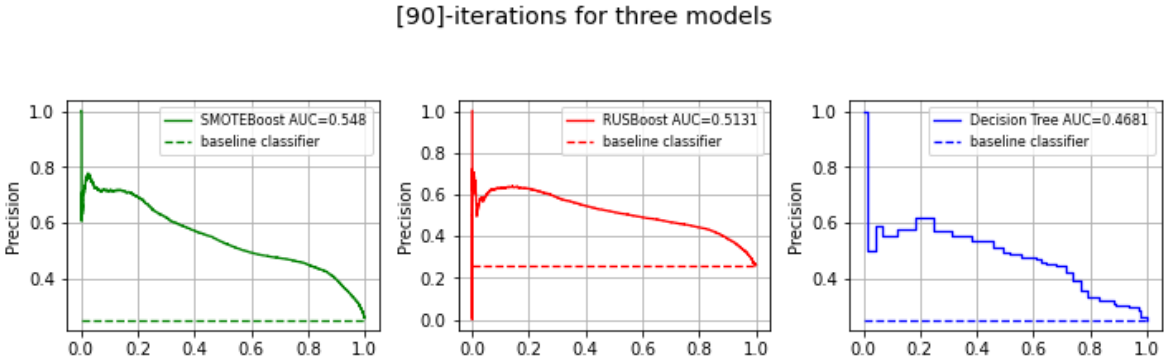

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

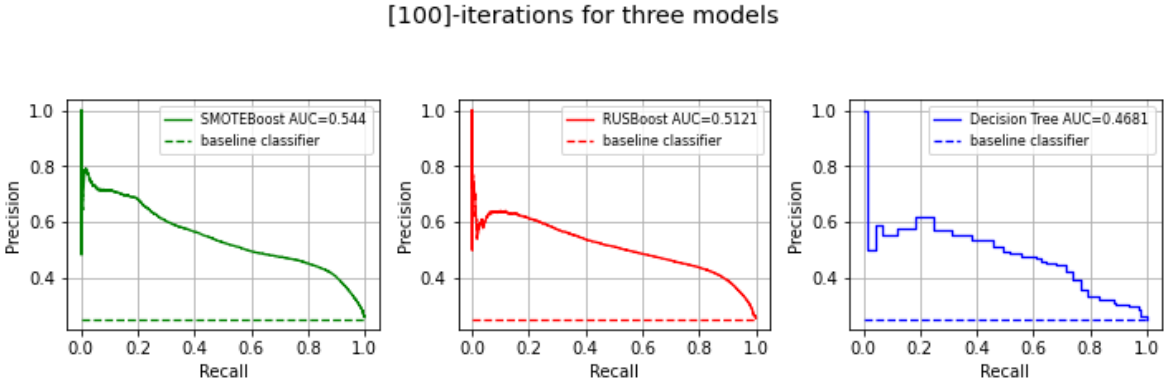

FIGURE 4.4: Precion-Recall Curve with **(N=100,T=4)** parameters.

4.2.2 Experimental result with: 3 < IR <= 9 on Segment0 datasets for: SMOTEBoost, RUSBoost, Decsion-Tree models

	(N=100,T=14)											
	DT-SMOTEBoost				DT-RUSBoost				Decsion-Tree(DT)			
K-iteration (10-CV)	*F1-score*	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*
Avg-[10]	0.979	0.989	0.976	0.982	0.968	0.992	0.943	0.995	0.714	0.780	0.985	0.562
Avg-[20]	0.979	0.989	0.977	0.983	0.969	0.993	0.946	0.995	0.714	0.781	0.987	0.562
Avg-[30]	0.979	0.989	0.977	0.982	0.969	0.993	0.944	0.996	0.714	0.780	0.984	0.562
Avg-[40]	0.979	0.989	0.976	0.983	0.969	0.993	0.945	0.995	0.714	0.780	0.984	0.562
Avg-[50]	0.979	0.989	0.976	0.982	0.968	0.993	0.943	0.996	0.714	0.781	0.986	0.562
Avg-[60]	0.979	0.989	0.976	0.983	0.969	0.993	0.945	0.995	0.714	0.780	0.985	0.562
Avg-[70]	0.979	0.989	0.976	0.982	0.968	0.993	0.943	0.995	0.714	0.780	0.984	0.562
Avg-[80]	0.980	0.990	0.977	0.984	0.968	0.992	0.944	0.995	0.714	0.780	0.985	0.562
Avg-[90]	0.979	0.990	0.976	0.983	0.969	0.993	0.945	0.996	0.714	0.780	0.985	0.562
Avg-[100]	0.979	0.990	0.976	0.983	0.968	0.992	0.943	0.995	0.714	0.780	0.985	0.562
Average	0.9791	0.9893	0.9763	0.9827	0.9685	0.9927	0.9441	0.9953	0.714	0.7802	0.985	0.562

TABLE 4.8: Summary average of of **three Models** on **Segment0** dataset.

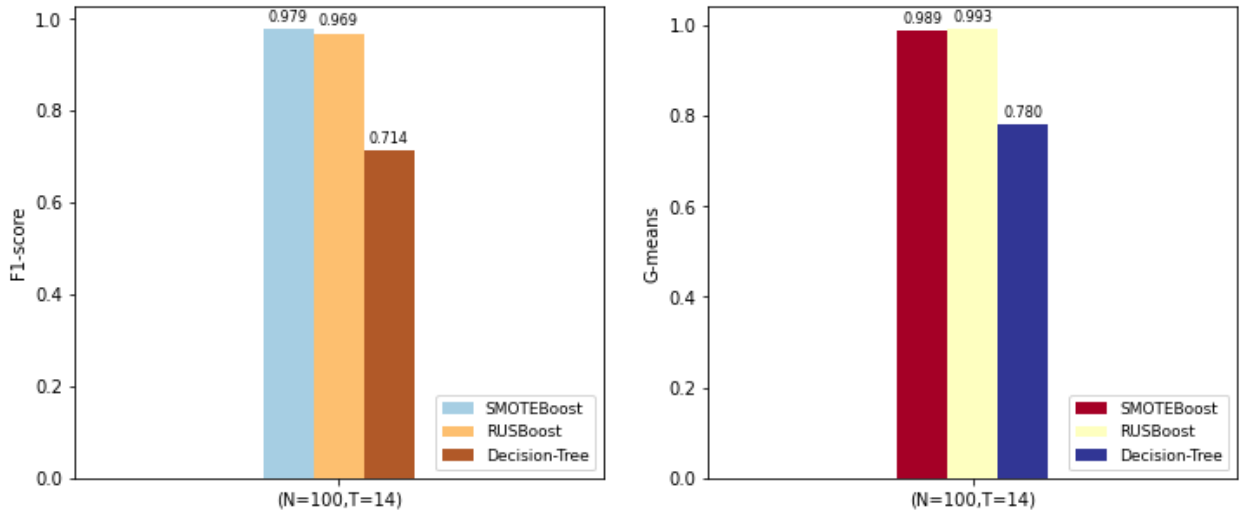

FIGURE 4.8: F1-score and G-means on the **Segments0** dataset.

Discussion

The graphe of f1-score, G-means and Table 4.8 show the classification performance in terms of F-measure ,G-means, Rcall, Precsion. We have obtained very satisfactory results usingusing random forest with ensemble boosting methods techniques.

As shown in the results, the proposed SMOTEBoost and RUSBoost methods using Decision-Tree algorithm have shown good results. So that, DT-SMOTEBoost got a F1-score (0.9791) better than Decsion-Tree alone (without SMOTE-Boost), also DT-RUSBoost has a F1-score (0.9685) that is better

than a Decsion-Tree alone (without SMOTEBoost) (0.714). In addition, for the rest metrics Boosting ensembles methods(SMOTEBOOST, RF-RUSBoost) gave better results than Decsion-Tree alone (without boosting ensmble methods). As indicated by the results, the proposed SMOTEBoost and RUSBoost methods with Decion-Tree algorithm demonstrated the best performance DT-SMOTEBoost has F1-score **(0.9791)** better than single DT-SMOTEBoost (without SMOTEBoost), also DT-RUSBoost has better F1-score**(0.9685)**better than single Decsion-Tree (without SMOTEBoost)**(0.714)**. In addition for the rest metrics Boosting ensembles methods(Dt-SMOTEBOOST, DT-RUSBoost) performed better than single Decsion-Tree (without boosting ensmble methods). Results Visualization with(N=200,T=2) on vehicle1 datasets

4. Roc-Auc Curve: Results Visualization with(N=100,T=14) on segment0 datasets

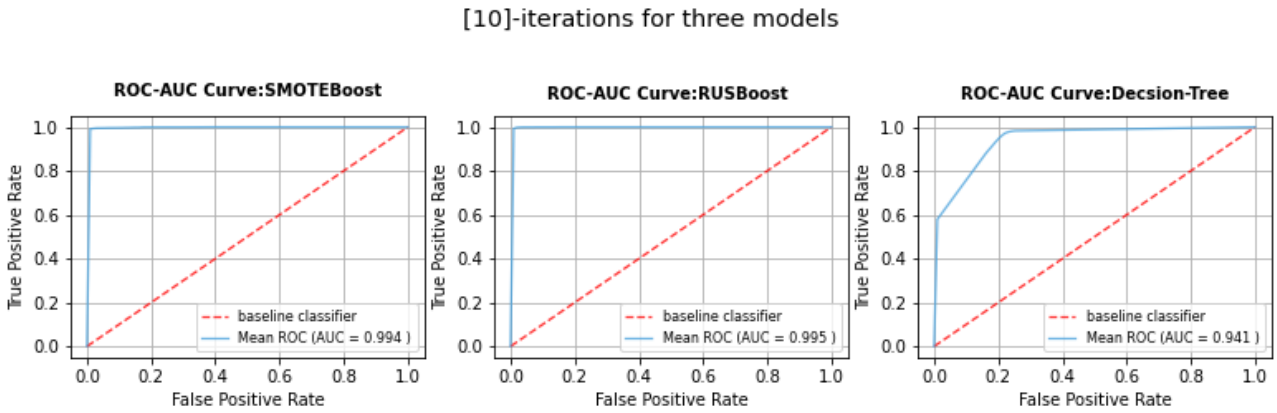

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

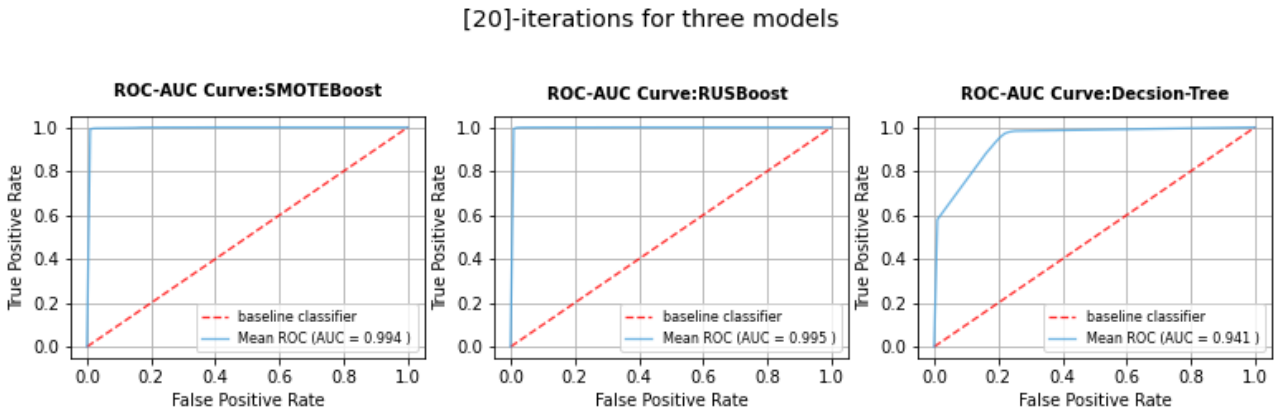

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

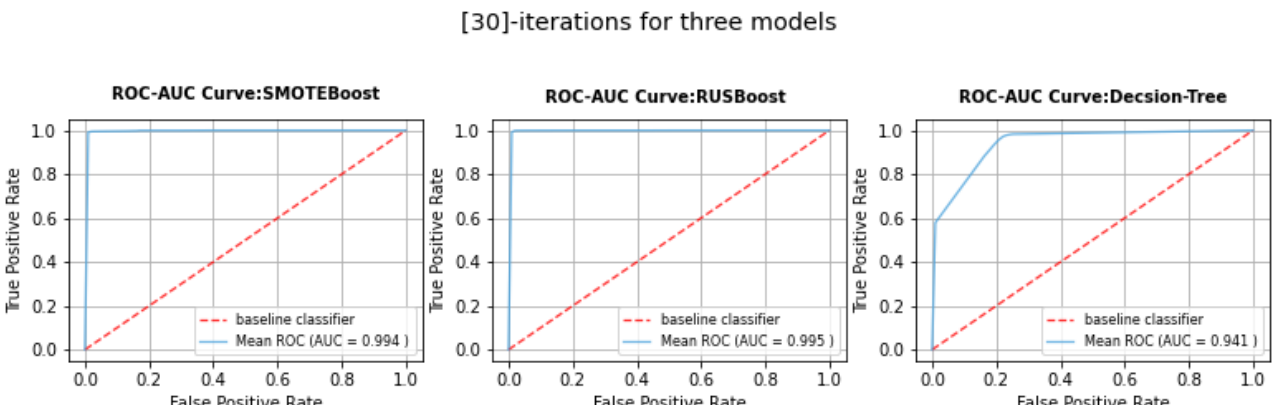

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

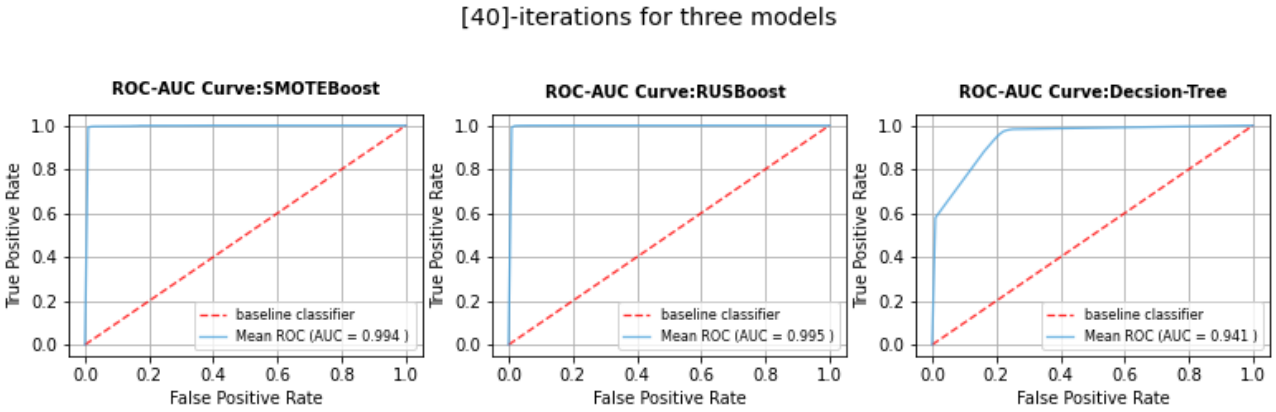

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

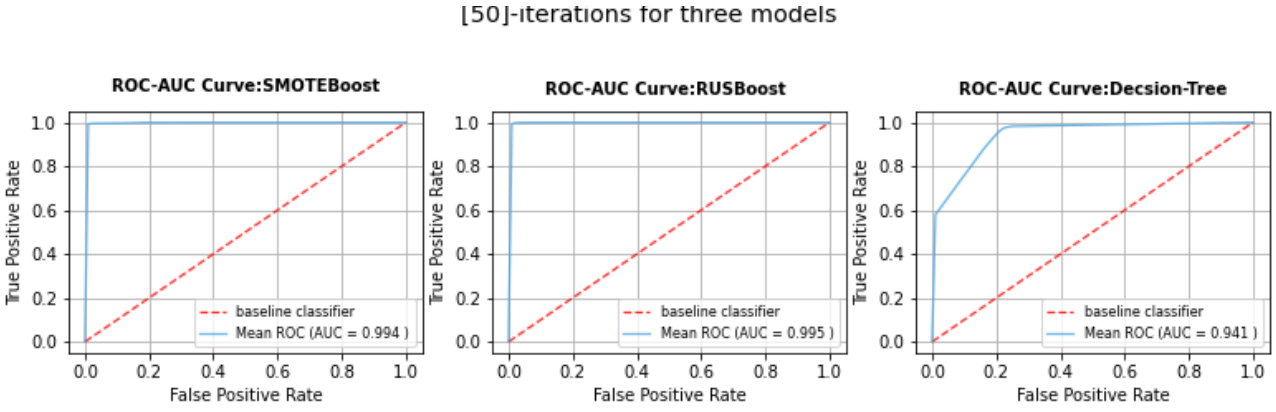

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

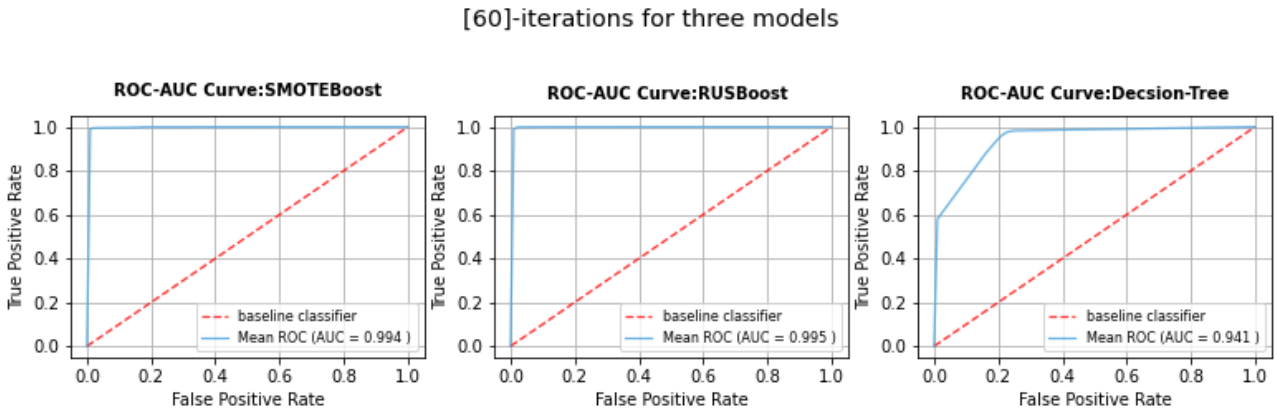

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

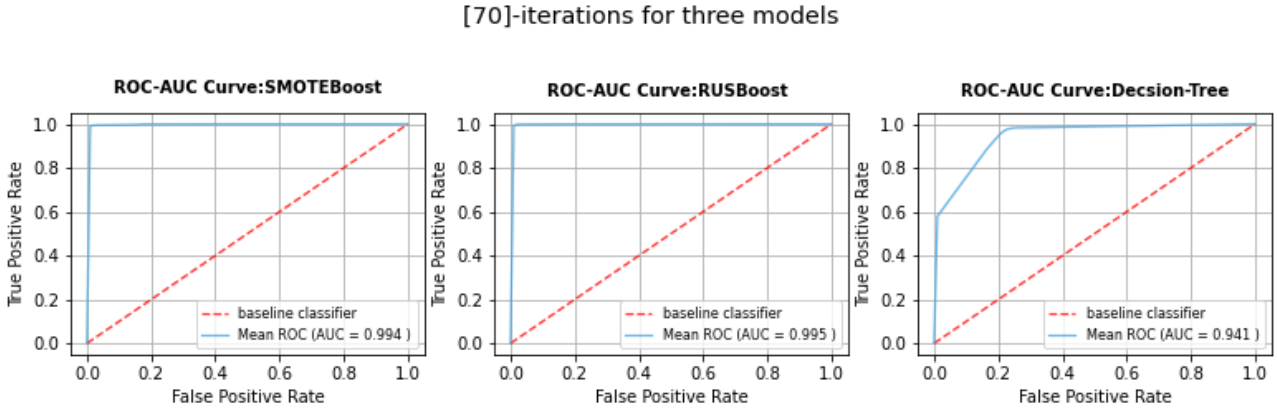

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

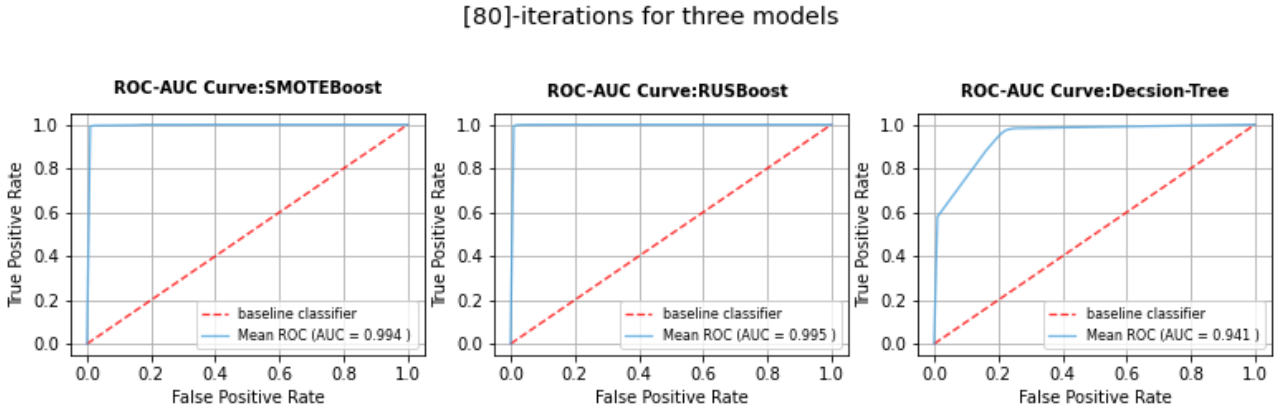

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

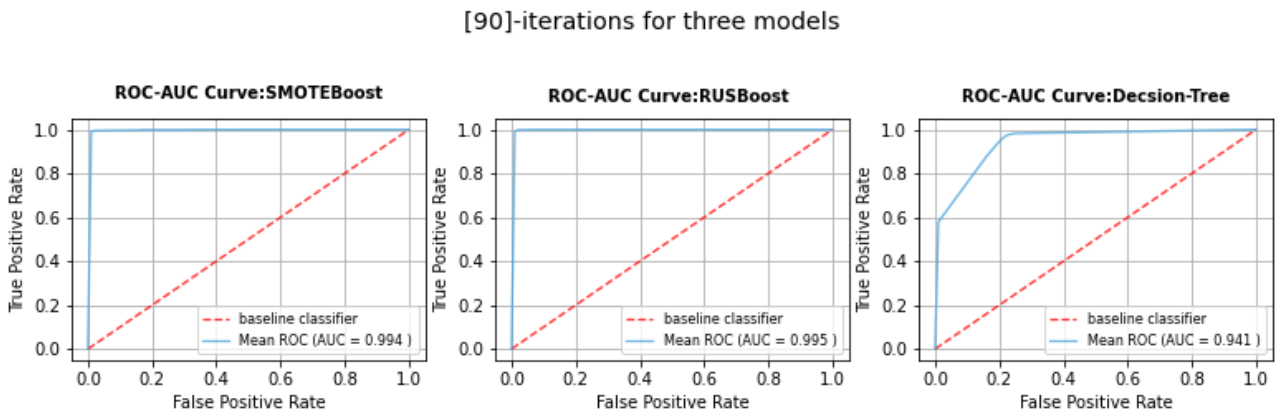

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

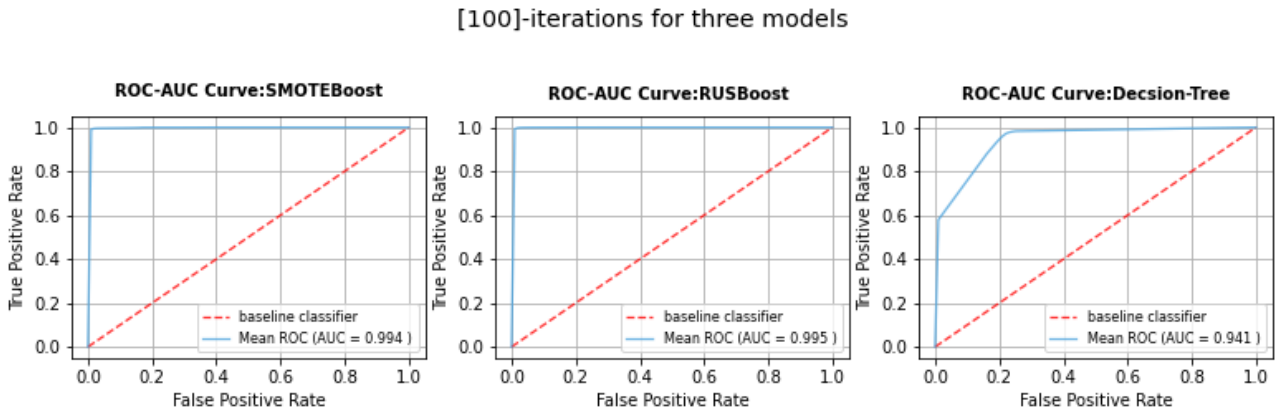

FIGURE 4.8: Roc/Auc Curve with **(N=100,T=14)** parameters.

5. Precion-Recall Curve:Result Visualization with(N=100,T=14) on Segment0 datasets

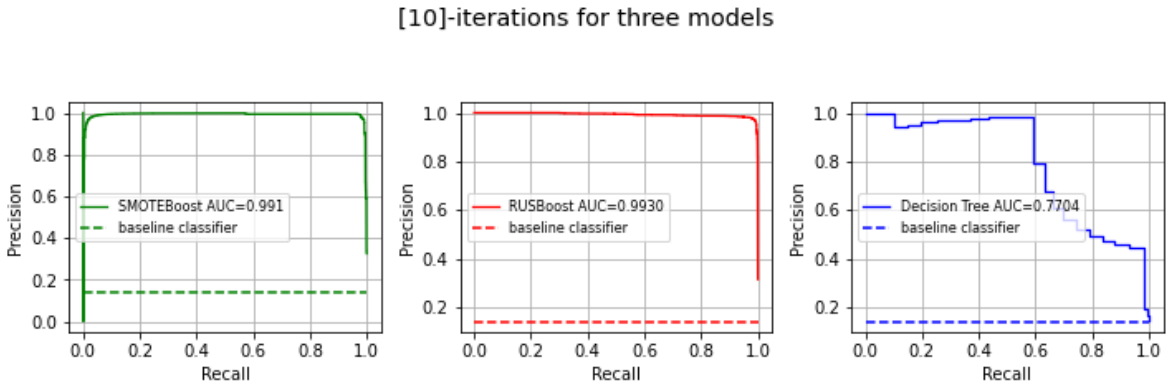

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

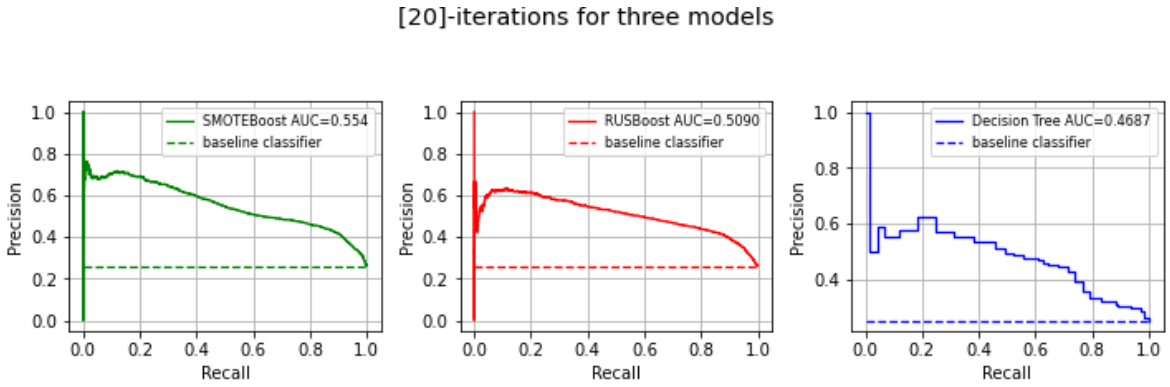

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

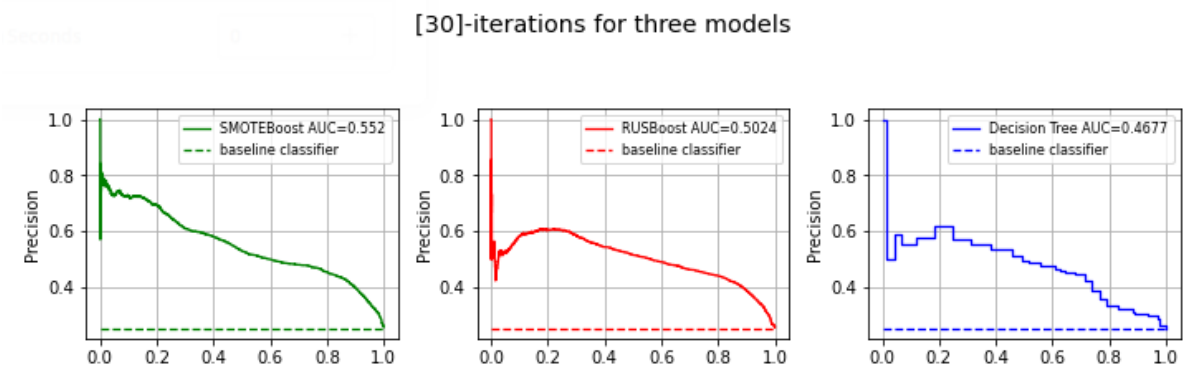

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

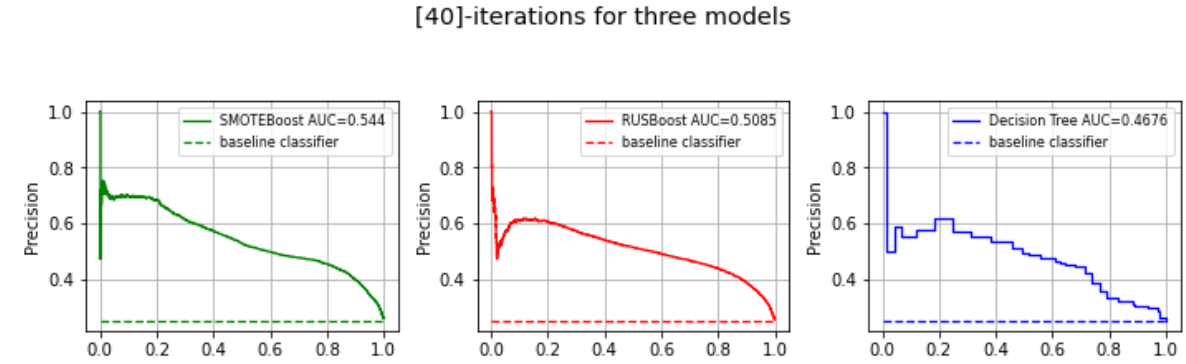

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

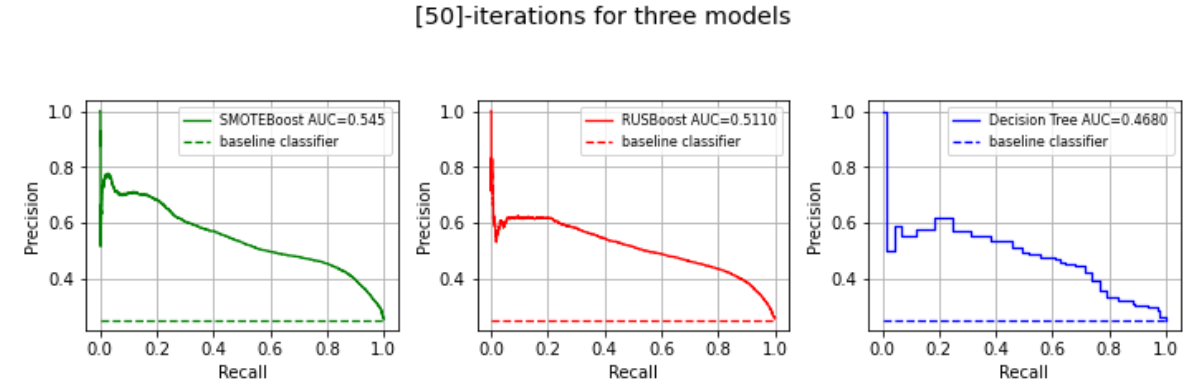

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

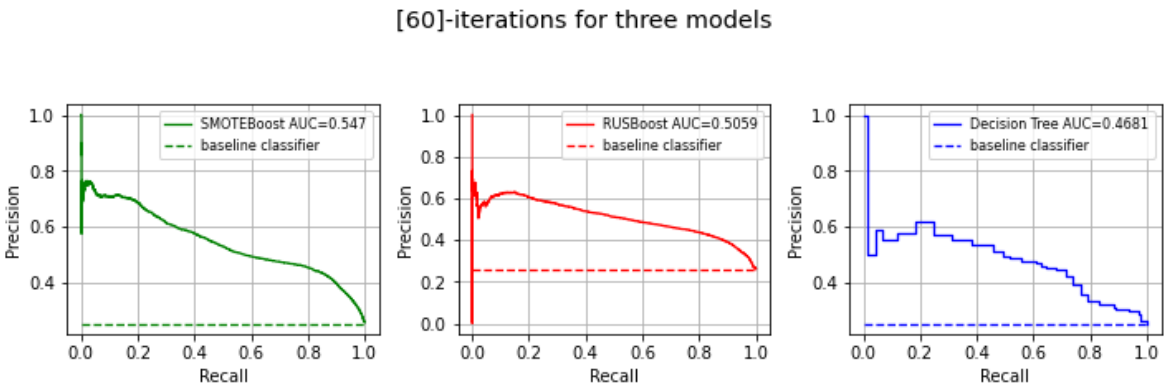

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

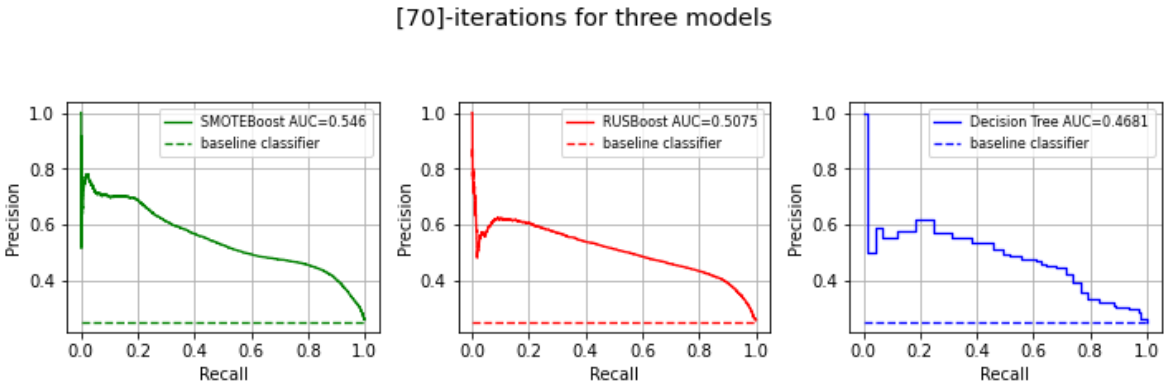

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

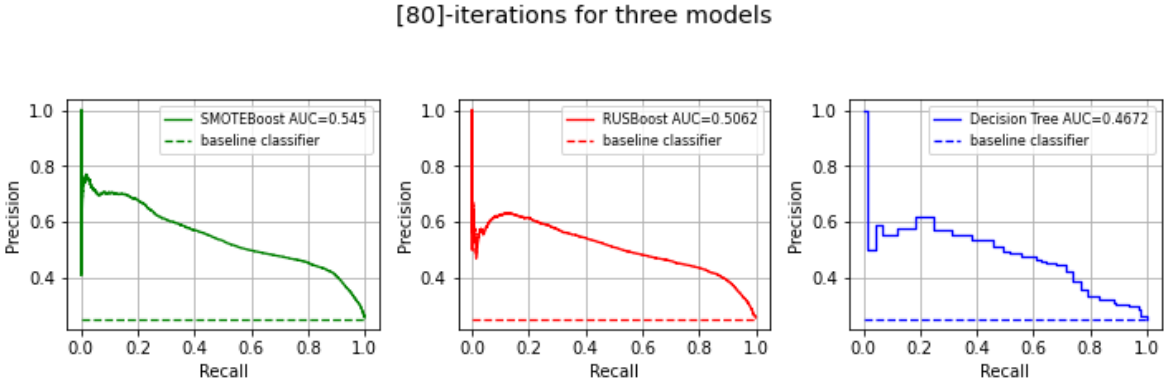

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

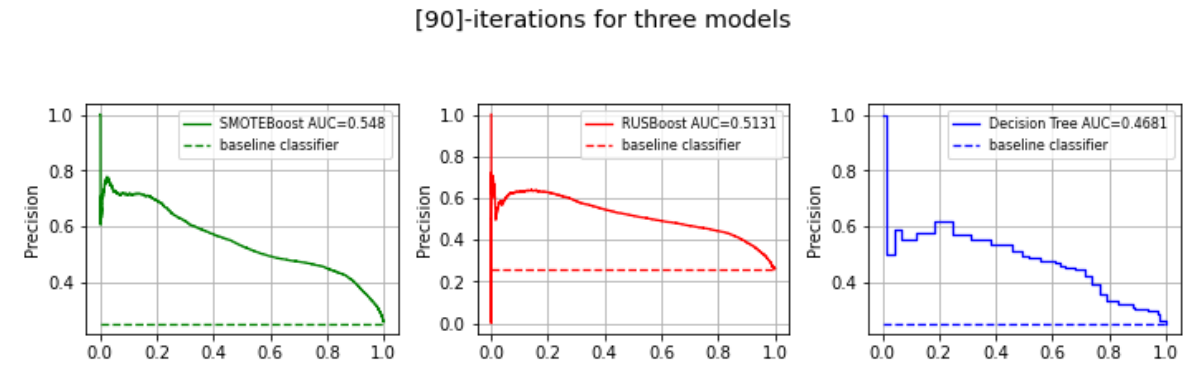

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

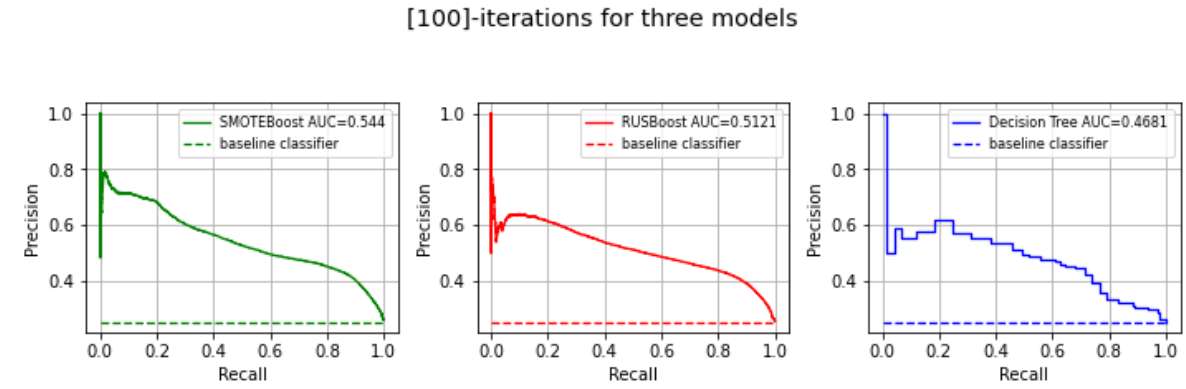

FIGURE 4.8: Precision-Recall Curve with **(N=100,T=14)** parameters.

4.2.3 Experimental result with: IR>9 on shuttle-6_vs_2-3 datasets for:SMOTEBoost, RusBoost and Random-Forest models.

	(N=100,T=2)											
	RF-SMOTEBoost				RF-RUSBoost				Random-Forest			
k-iteration (10-CV)	*F1-score*	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*	*F1-score*	*G-means*	*Precision*	*Recall*
Avg-[10]	0.862	0.947	0.850	0.91	0.803	0.944	0.775	0.93	0.627	0.813	0.625	0.63
Avg-[20]	0.844	0.938	0.830	0.885	0.874	0.969	0.851	0.975	0.601	0.804	0.597	0.61
Avg-[30]	0.844	0.960	0.817	0.963	0.844	0.960	0.817	0.963	0.602	0.802	0.60	0.607
Avg-[40]	0.837	0.937	0.821	0.888	0.807	0.952	0.774	0.96	0.602	0.804	0.598	0.612
Avg-[50]	0.852	0.938	0.842	0.89	0.830	0.956	0.802	0.96	0.620	0.813	0.616	0.628
Avg-[60]	0.855	0.941	0.843	0.895	0.831	0.950	0.809	0.953	0.592	0.799	0.589	0.6
Avg-[70]	0.859	0.943	0.847	0.899	0.818	0.953	0.789	0.963	0.588	0.796	0.585	0.596
Avg-[80]	0.857	0.948	0.840	0.912	0.834	0.956	0.809	0.962	0.615	0.810	0.612	0.622
Avg-[90]	0.877	0.955	0.862	0.923	0.823	0.956	0.794	0.969	0.585	0.795	0.581	0.593
Avg-[100]	0.847	0.939	0.834	0.893	0.843	0.959	0.818	0.969	0.591	0.798	0.587	0.599
Average	0.8534	0.9446	0.8386	0.9058	0.8306	0.9555	0.8038	0.9604	0.6023	0.8034	0.599	0.6097

TABLE 4.9: Summary average of of **three Models** on **shuttle-6_vs_2-3** dataset.

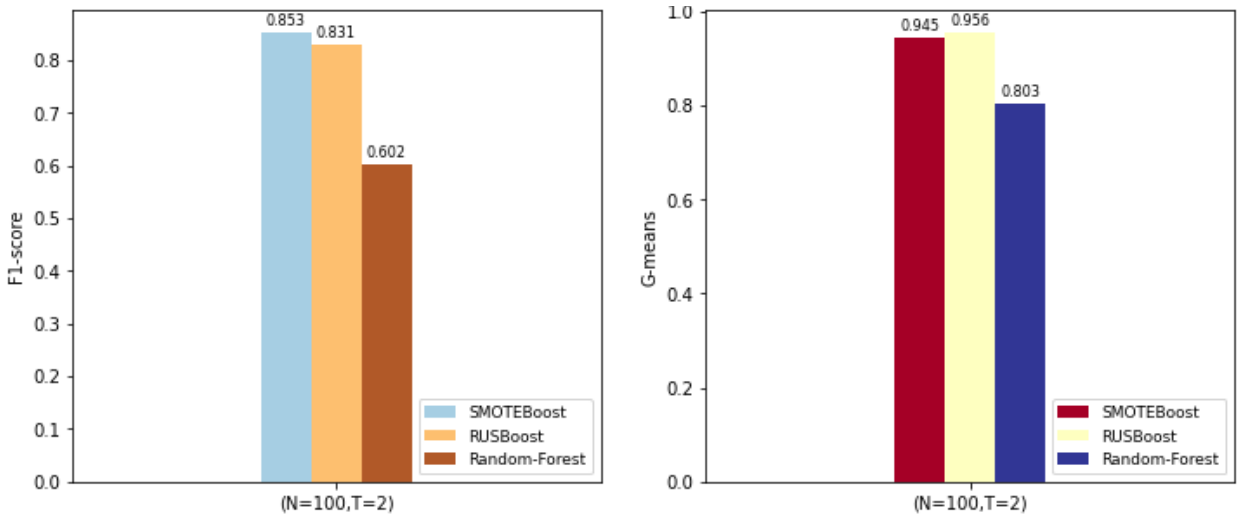

FIGURE 5.2: F1-score and G-means on the **Shuttle-6_vs_2-3** dataset.

Discussion

Table 4.9 shows the classification performance in terms of F-measure, G-means, Rcall, Precsion, that was obtained using random forest with ensemble boosting methods techniques. As indicated by the results, the proposed SMOTEBoost and RUSBoost methods with Random Forest algorithm have shown very satisfactory results. So that, RF-SMOTEBoost has F1-score (0.8534) is better than random Forest alone (without SMOTEBoost), also RF-RUSBoost has better F1-score(0.8306) is better than random Forest alone (without SMOTEBoost). In addition, for the rest metrics Boosting ensembles

methods(RF-SMOTEBOOST, RF-RUSBoost) performed better than random forest alone (without boosting ensemble methods).

6. Roc/Auc Curve:Result Visualization with(N=100,T=2) on shuttle-6_vs_2-3 dataset

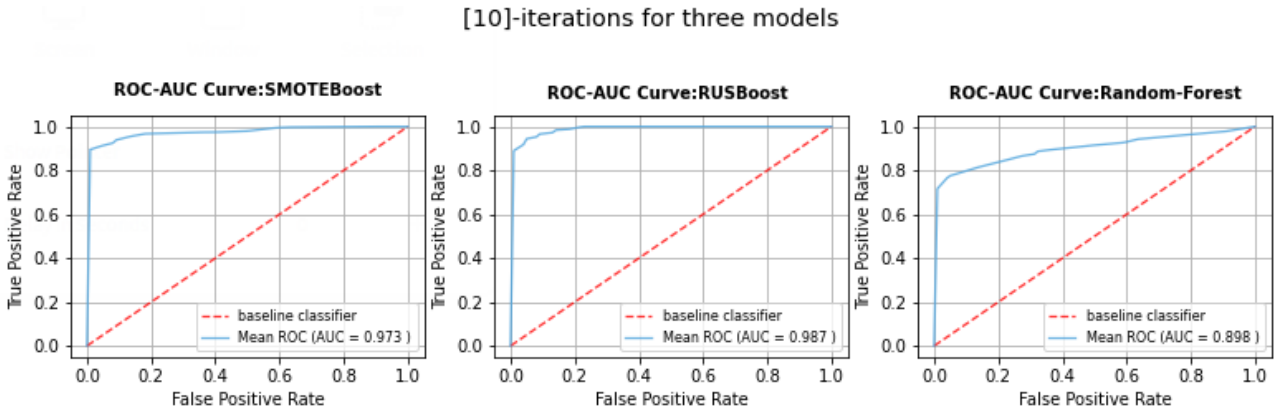

FIGURE 5.2: Precion-Recall Curve with **(N=100,T=2)** parameters.

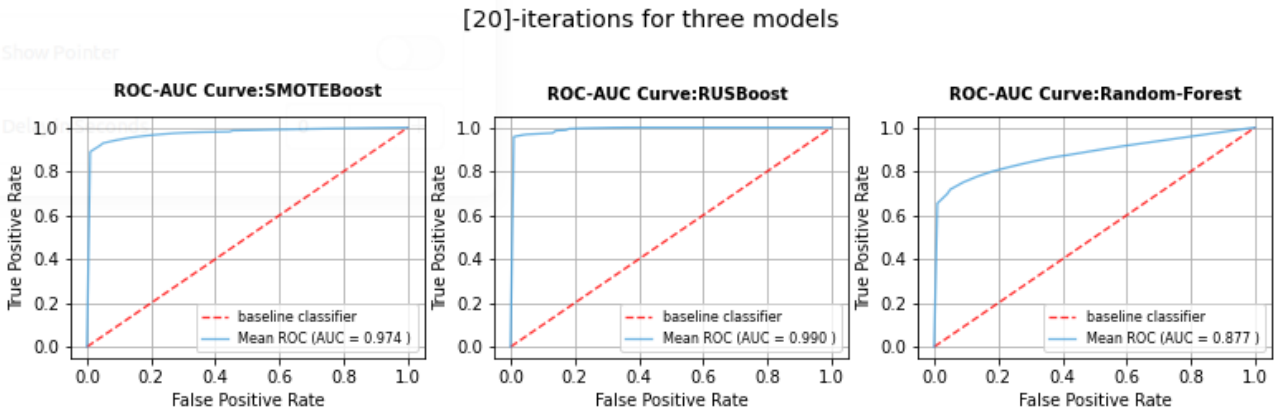

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

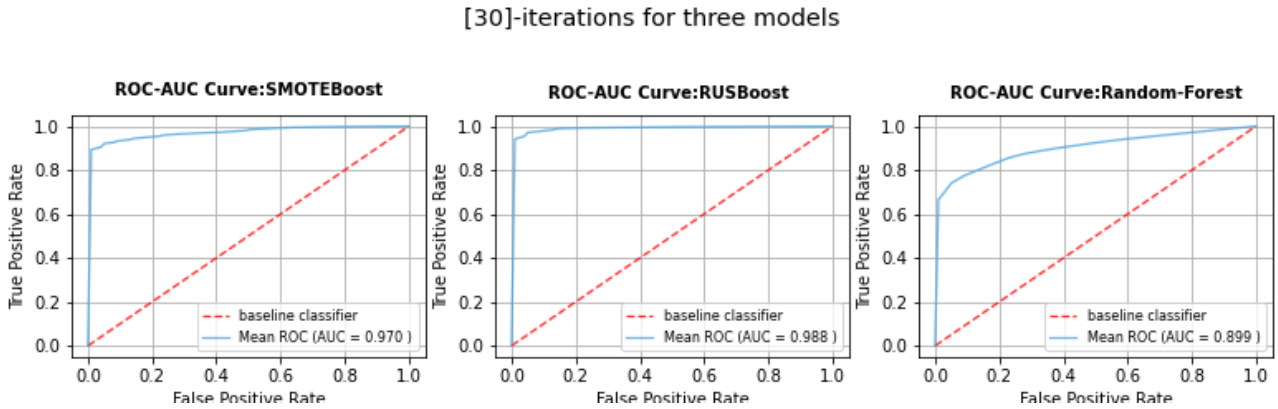

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

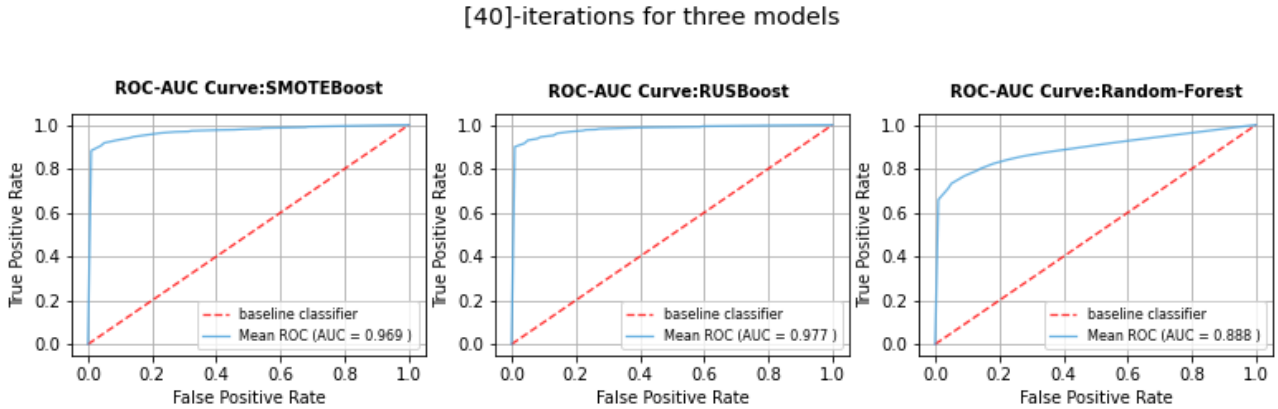

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

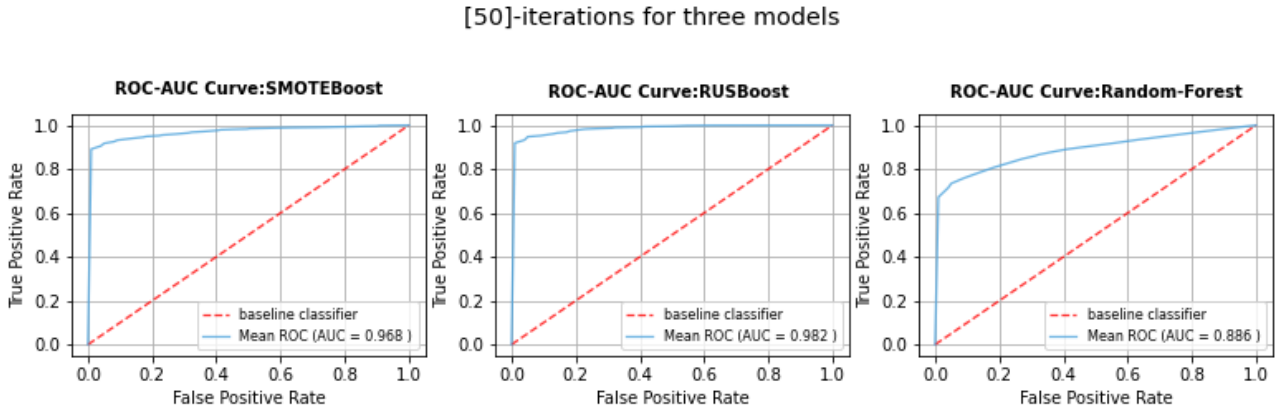

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

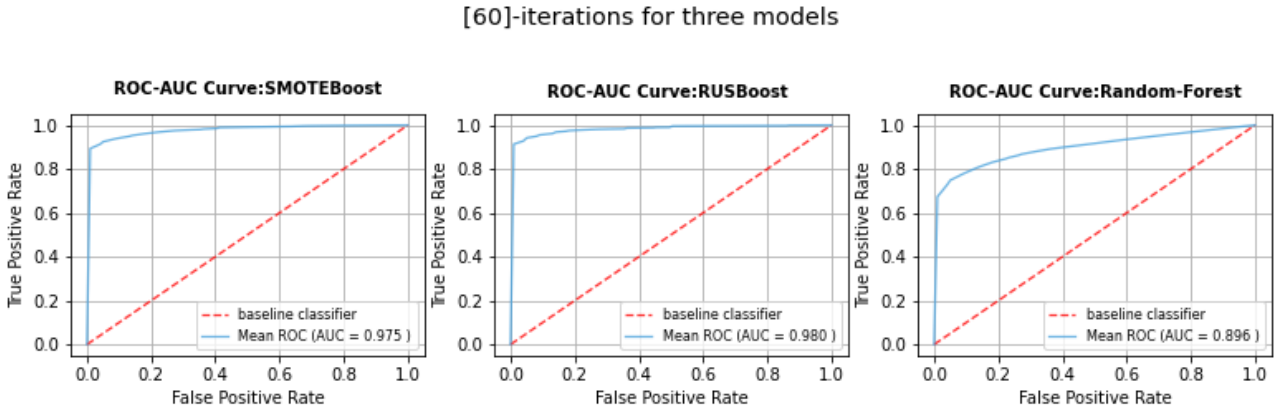

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

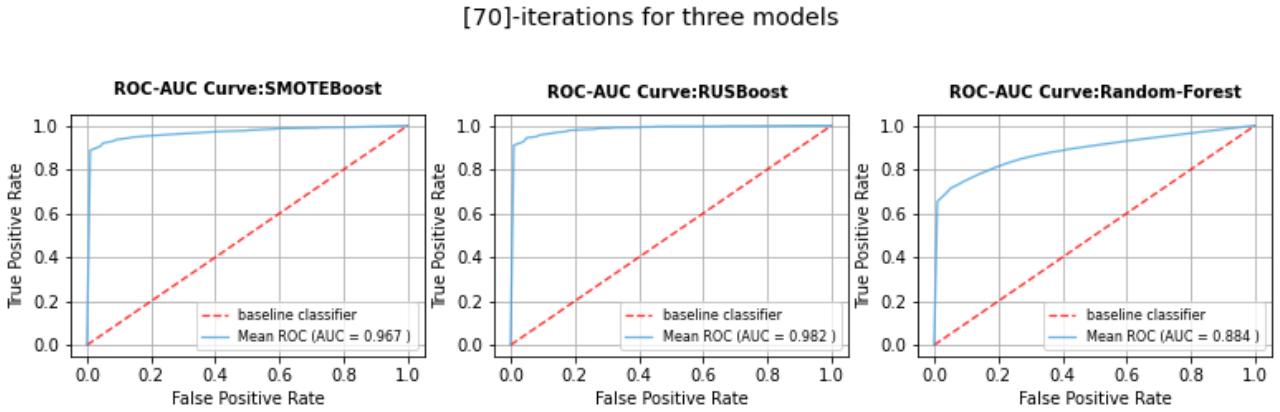

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

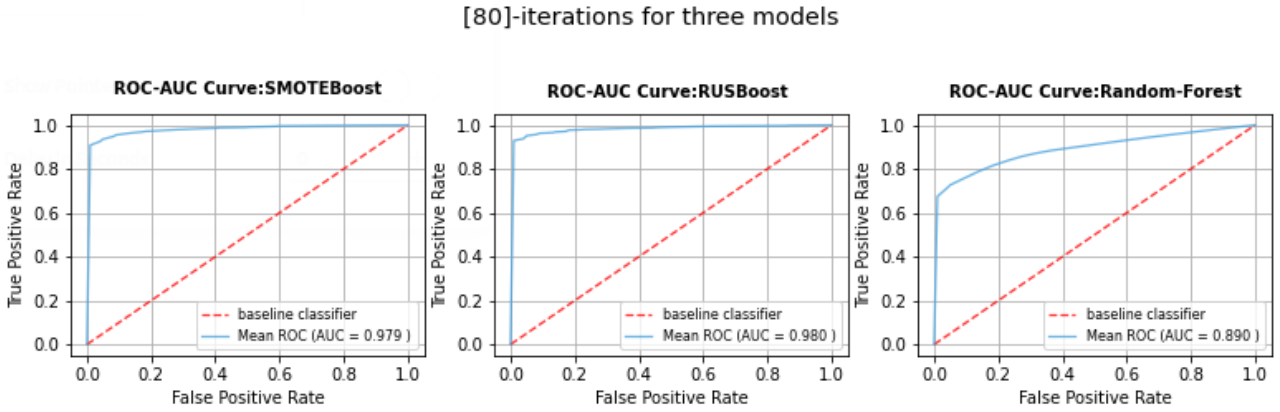

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

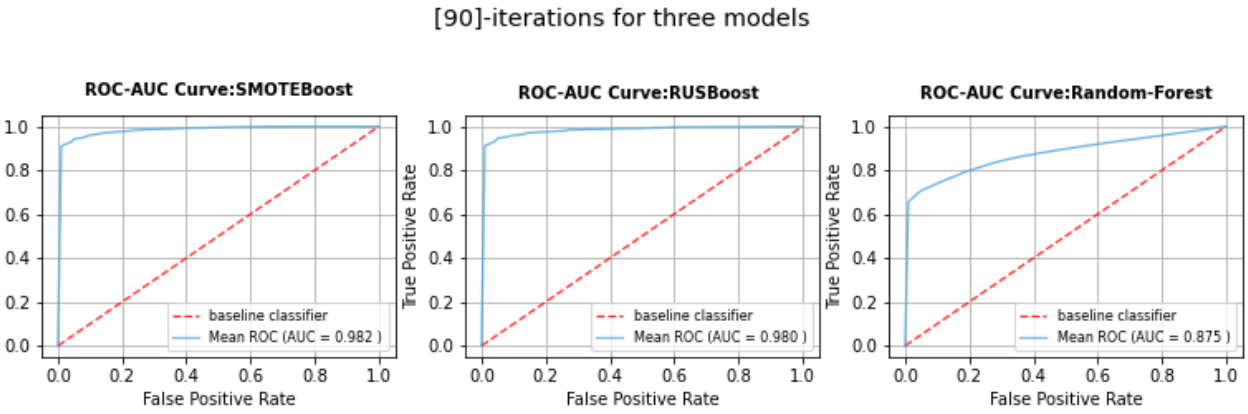

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

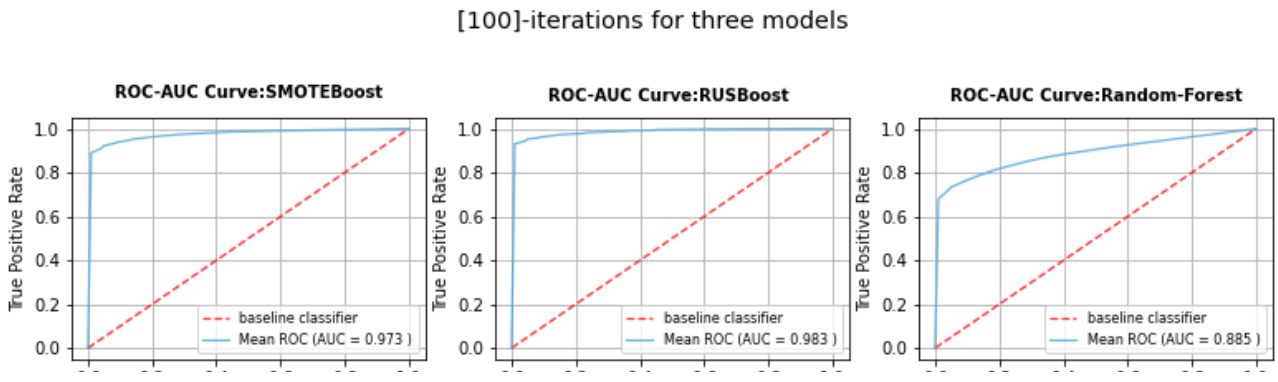

FIGURE 5.2: Roc/Auc Curve with **(N=100,T=2)** parameters.

7. Precion-Recall Curve:Result Visualization with(N=100,T=2) on shuttle-6_vs_2-3 dataset

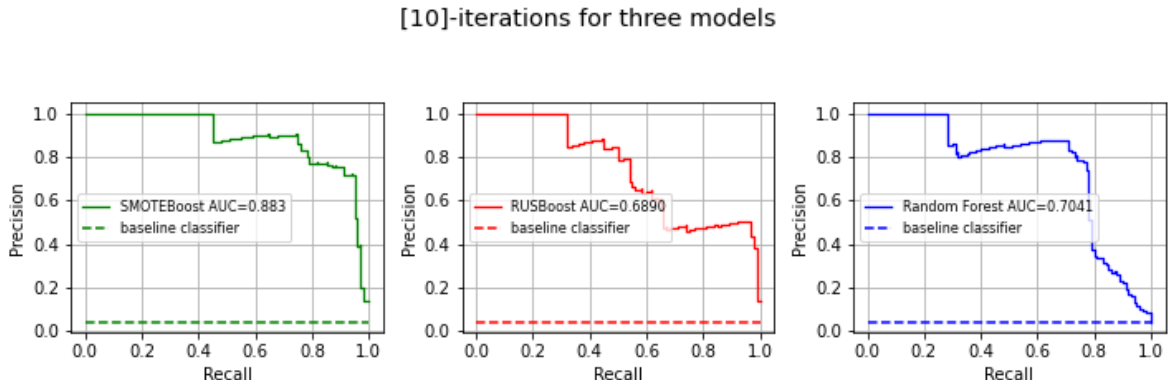

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

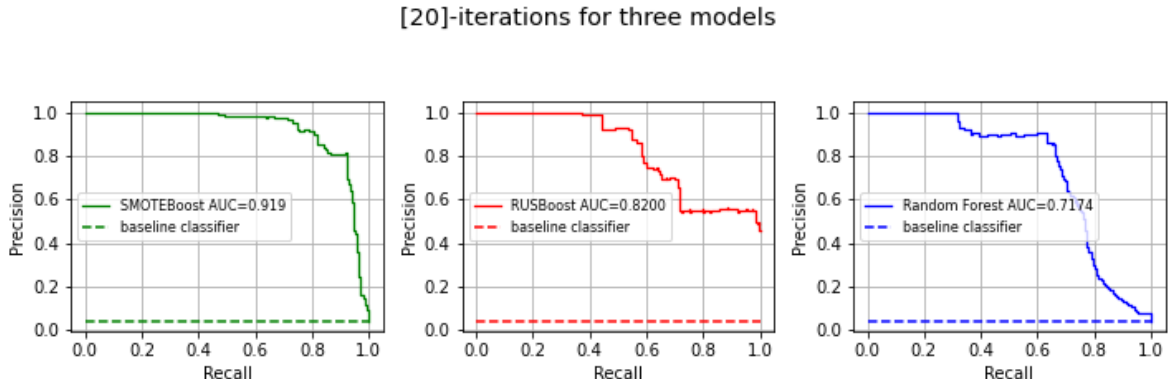

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

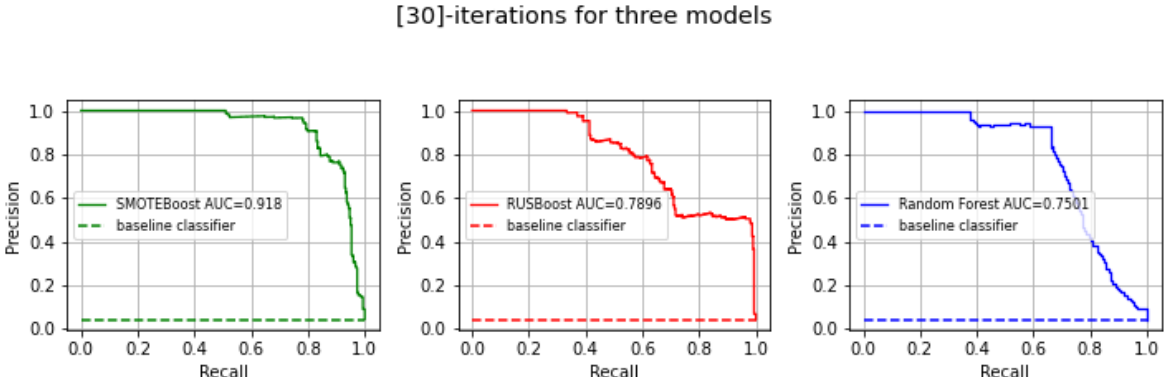

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

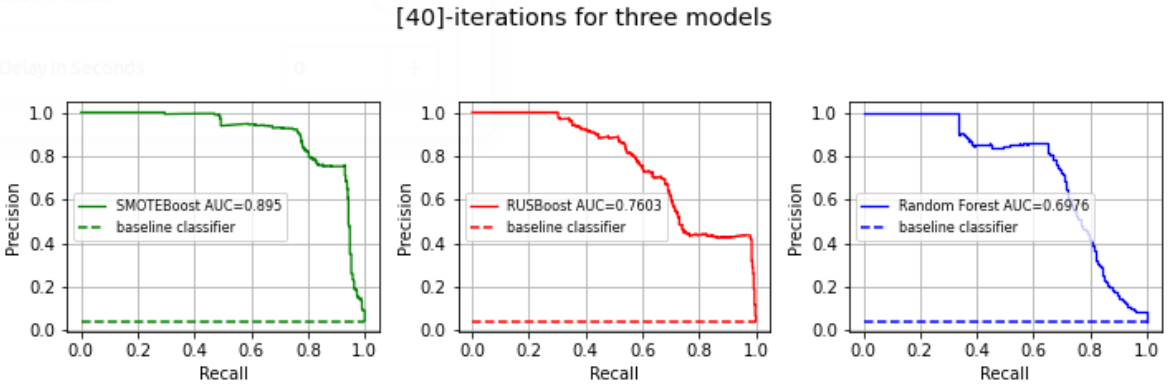

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

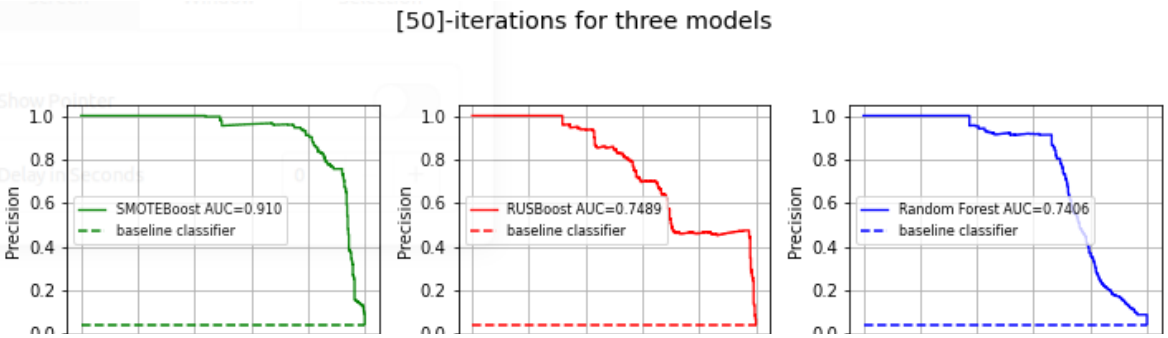

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

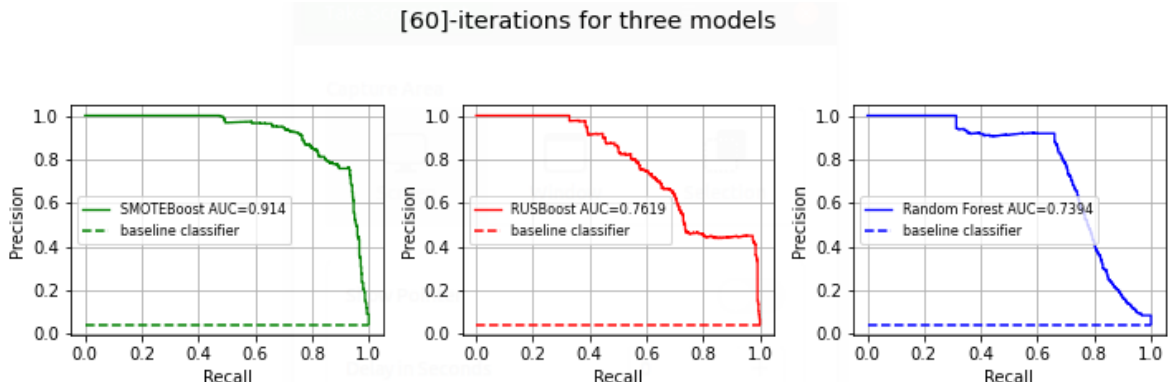

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

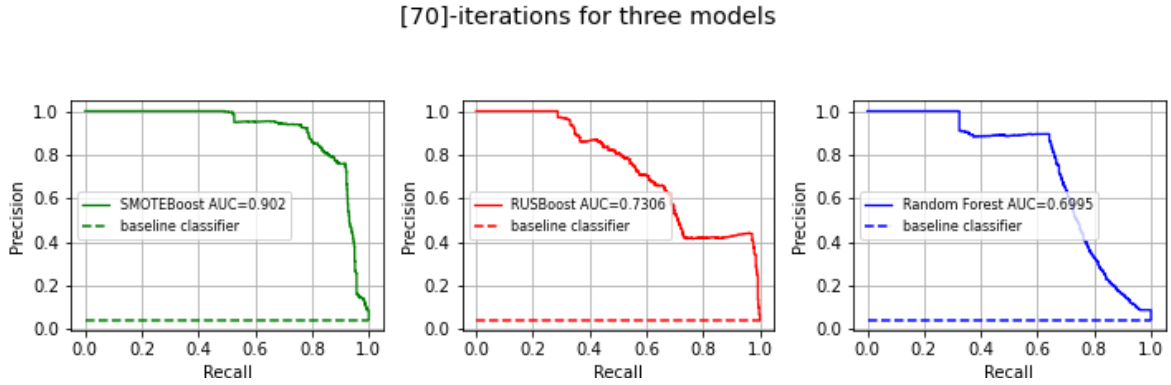

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

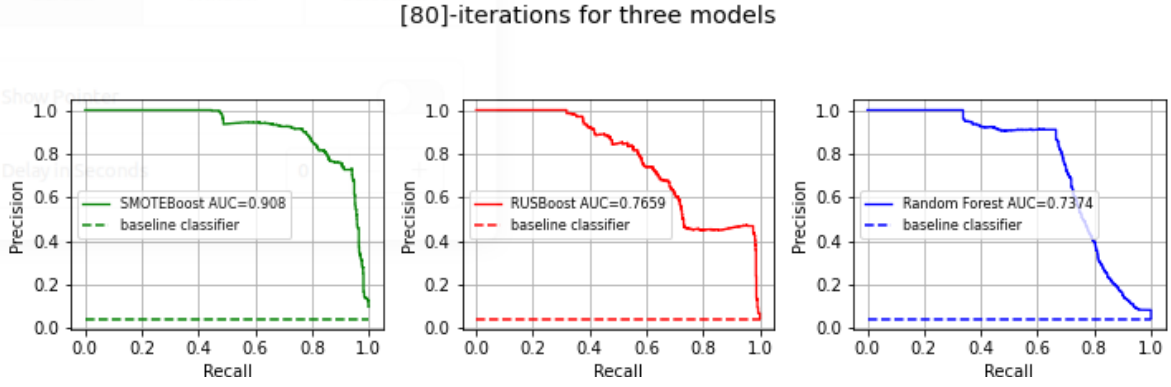

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

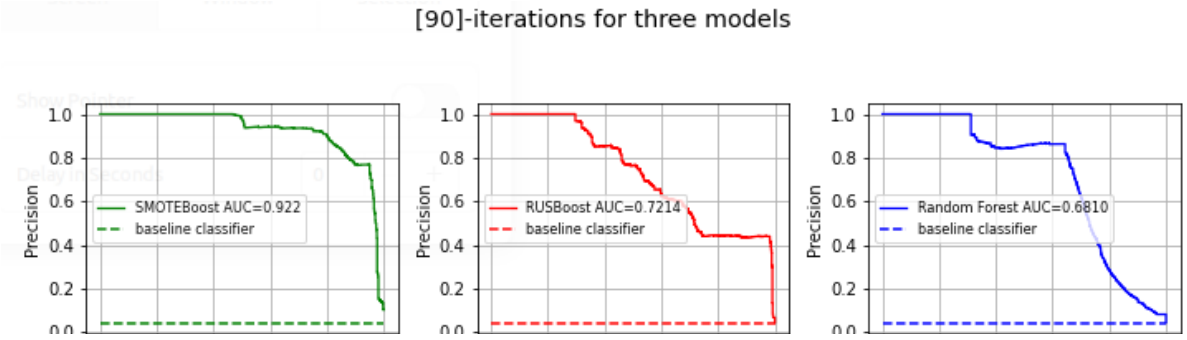

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

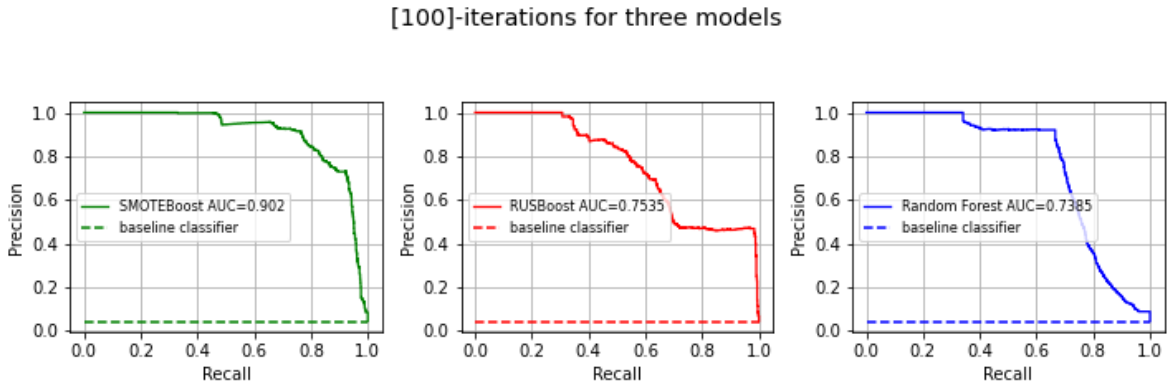

FIGURE 5.2: Precision-Recall Curve with **(N=100,T=2)** parameters.

Conclusion

Existing classification algorithms generally focus on majority class instances and ignore the minority class instances. So, it is a challenging task to construct an effective classifier that can correctly classify the instances of the minority class. This is ever so pertinent as class imbalance problems affect a vast range of domains. Recently, computational intelligence researchers have proposed several hybrid techniques

by combining sampling with ensemble classifiers for dealing with class imbalance problems. The purpose of this thesis is to present the effectiveness boosting algorithms, in order to alleviate the problem of class imbalance. We have compared the performance of SMOTEBoost and RUSboost with Decision tree(DT) and Random Forest as base classifier algorithms. Based on experimental results, we have found that proposed methods performed favourably.

Both RUSBoost and SMOTEBoost perform significantly better than Decision Tree and Rndom Forest. In other words, the application of hybrid sampling/boosting is better than Decion Tree and Random Forest alone (without Boosting mehods).

General conclusion

The problem of classification in imbalanced datasets is an active research topic in artificial intelligence, machine learning and data mining. This is due to two factors. First, the rare cases appear in several areas such as fraud detection and diagnosis medical. Second, the approaches and methods of complete classification ignore minority instances or see them as noisy instances and they tend to extract rules specific to majority instances. Therefore, most models have poor performance for the basics of imbalanced data because they are designed to minimize the overall error rate.

Several methods and approaches have been proposed to deal with the problem of classification in imbalanced databases. These methods have been divided in four levels: **the data level(sampling approach)**, **the algorithm level**, **the hybrid level(Cost-sensitive learning)** and **ensemble methods**.

At **the data level(sampling approach)**, the distribution of instances is altered by under-sampling which removes majority instances or oversampling which duplicates minority instances or by hybrid sampling.

At **the algorithm level**, the proposed solutions are divided into two categories: modification of the algorithms of the complete classification to adapt them to the partial classification and the proposal(create) of specific algorithms.

At **the hybrid level(Cost-sensitive learning)**, recent popular research areas in data science field that take the misclassification costs and other types of cost into consideration to minimize the total cost by introduced unequal cost to misclassification errors.

At **ensemble methods**, it fellows the human behaviour, it decides to take a decision by constructing several classifiers from data and combining their prediction to enhance the performance and the outcome of classifier.

Perspectives

Many different adaptations, tests and experiments have been left for the future due to lack of time. In addition, the following ideas could be tested:

- Exploration the effectiveness of others level unlike ensemble level (boosting methods) that mentioned in previous chapter, such as Cost sensitive learning, Algorithm level,...etc.
- Address the problem of imbalanced classification in Big Data Problems for binary and multi class using parallelism.
- Address the problem of imbalanced classification with multiple classes
- Deal with the problem of classification in imbalanced datasets by studying other factors such as the presence of small disjunctions of minority instances and the overlap between classes.
- Address the problem of Imbalanced Data Streams.

Bibliography

Abdallah A., Maarof M. A. Zainal A. (2016). "Fraud detection system: A survey." In: *Journal of Network and Computer Applications* 68, pp. 90–113.

Adomavicius G., Tuzhilin A. (2001). "Using data mining methods to build customer profiles." In: 2.34, pp. 74–82.

Agarwal, S. (2013). "Data mining: Data mining concepts and techniques." In: pp. 203–207.

Aggarwal, C. C. (2015). "Data classification." In: pp. 285–344.

Alaiz-Rodríguez R., Japkowicz N. (2008). "Assessing the impact of changing environments on classifier performance." In: *Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 13–24.

Alasadi S. A., Bhaya W. S. (2017). "Review of data preprocessing techniques in data mining." In: 16.12, pp. 4102–4107.

Alcalá-Fdez J., Fernández A. Luengo J. Derrac J. García S. Sánchez L. & Herrera F. (2011). "Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework". In: 17, pp. 451–466.

Alejo R., Sotoca J. M. Valdovinos R. M. Toribio P (2010). "Edited nearest neighbor rule for improving neural networks classifications." In: *International Symposium on Neural Networks*, pp. 303–310.

Amin A., Anwar S. Adnan A. Nawaz M. Howard N. Qadir J. ... Hussain A. (2016). "Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study." In: *IEEE Access* 4, pp. 7940–7957.

Antonakis J., Dalgas O. (2009). "Predicting elections: Child's play!" In: *Science* 323.5918, pp. 1183–1183.

Bagley S. C., White H. Golomb B. A. (2001). "Logistic regression in the medical literature:: Standards for use and reporting, with particular attention to one medical domain." In: *Journal of clinical epidemiology* 54.10, pp. 979–985.

Batista G. E., Prati R. C. Monard M. C. (2004). "A study of the behavior of several methods for balancing machine learning training data." In: *ACM SIGKDD explorations newsletter* 6.1, pp. 20–29.

Behal A., Chen Y. Kieliszewski C. Lelescu A. He B. Cui J. ... Spangler W. S. (2007). "Business Insights Workbench–An Interactive Insights Discovery Solution." In: pp. 834–843.

Bermejo P., Gámez J. A. Puerta J. M. (2011). "Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets." In: *Expert Systems with Applications* 38.3, pp. 2072–2080.

Berry M. W., Mohamed A. Yap B. W. (Eds.). (2019). "Supervised and unsupervised learning for data science". In:

Boyd K., Eng K. H. & Page C. D. (2013). "Area under the precision-recall curve: point estimates and confidence intervals." In: pp. 451–466.

Branco P., Torgo L. Ribeiro R. (2015). "A survey of predictive modelling under imbalanced distributions." In:

Branco P., Torgo L. Ribeiro R. P. (2016). "A survey of predictive modeling on imbalanced domains." In: 49.2, pp. 1–50.

Breault J. L., Goodall C. R. Fos P. J. (2002). "Data mining a diabetic data warehouse." In: 1-2.26, pp. 37–54.

Breiman, L. (1996). "Bagging predictors. Machine learning". In: *Machine learning* 24.2, pp. 123–140.

Brilliant M., Handoko D. (2017). "Implementation of Data Mining Using Association Rules for Transactional Data Analysis." In: pp. 177–180.

Britto Jr A. S., Sabourin R. Oliveira L. E. (2014). "Dynamic selection of classifiers—a comprehensive review." In: 11.47, pp. 3665–3680.

Chawla N. V., Japkowicz N. Kotcz A. (2004). "Special issue on learning from imbalanced data sets." In: *ACM SIGKDD explorations newsletter* 1.6, pp. 1–6.

Chawla N. V., Lazarevic A. Hall L. O. Bowyer K. W. (2003). "SMOTEBoost: Improving prediction of the minority class in boosting." In: pp. 107–119.

Chen M. S., Han J. Yu P. S. (1996). "Data mining: an overview from a database perspective." In: 6.8, pp. 866–883.

Chen X., Wang M. Zhang H. (2011). "The use of classification trees for bioinformatics." In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery.* 1.1, pp. 55–63.

Chen, Y. S. (2016). "An empirical study of a hybrid imbalanced-class DT-RST classification procedure to elucidate therapeutic effects in uremia patients." In: 54.6, pp. 983–1001.

Cieslak D. A., Chawla N. V. (2009). "A framework for monitoring classifiers' performance: when and why failure occurs?." In: *Knowledge and Information Systems* 18.1, pp. 83–108.

Cieslak D. A., Hoens T. R. Chawla N. V. Kegelmeyer W. P. (2012). "Hellinger distance decision trees are robust and skew-insensitive." In: *Data Mining and Knowledge Discovery* 24.1, pp. 136–158.

Devedzic, V. (2001). "Knowledge discovery and data mining in databases." In: pp. 615–637.

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

Díez-Pastor J. F., Rodríguez J. J. Garcia-Osorio C. Kuncheva L. I. (2015). "Random balance: ensembles of variable priors classifiers for imbalanced data." In: 85, pp. 96–111.

Elkan, C. (2001). "The foundations of cost-sensitive learning." In: *International joint conference on artificial intelligence* 17.1, pp. 973–978.

Erdoğan Ş. Z., Timor M. (2005). "A data mining application in a student database". In: 2.2, pp. 53–57.

Ezawa K. J., Singh M. Norton S. W. (1996). "Learning goal oriented Bayesian networks for telecommunications risk management." In: *ICML*, pp. 139–147.

Fan W., Stolfo S. J. Zhang J. Chan P. K. (1999). "AdaCost: misclassification cost-sensitive boosting." In: 99, pp. 97–105.

Fayyad, U. (2001). "Knowledge discovery in databases". In: pp. 28–47.

Fernández A., García S. Galar M. Prati R. C. Krawczyk B. Herrera F. (2018). "Learning from imbalanced data sets". In: *Berlin: Springer.* 11.

Fernández-Navarro F., Hervás-Martínez C. Gutiérrez P. A. (2011). "EA dynamic over-sampling procedure based on sensitivity for multi-class problems." In: 44.8, pp. 1821–1833.

Folino G., Pisani F. S. Sabatino P. (2016). "An incremental ensemble evolved by using genetic programming to efficiently detect drifts in cyber security datasets." In: pp. 1103–1110.

Fotouhi S., Asadi S. Kattan M. W. (2019). "A comprehensive data level analysis for cancer diagnosis on imbalanced data." In: *Journal of biomedical informatics* 90, p. 103089.

France T., Yen D. Wang J. C. Chang C. M. (2002). "Integrating search engines with data mining for customer-oriented information search." In: 10.5, pp. 242–254.

Freund Y., Schapire R. E. (1997). "A decision-theoretic generalization of online learning and an application to boosting". In: *Journal of computer and system sciences* 55.1, pp. 119–139.

Fu, Y. (1997). "Data mining". In: 4.16, pp. 18–20.

Gader P. D., Mohamed M. A. Keller J. M. (2996). "Fusion of handwritten word classifiers." In: 6.17, pp. 577–584.

Galar M., Fernandez A. Barrenechea E. Bustince H. Herrera F. (2011). "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches." In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.4, pp. 463–484.

Gama, J. (2010). "Knowledge discovery from data streams." In:

Gao Z., Zhang L. F. Chen M. Y. Hauptmann A. Zhang H. Cai A. N. (2014). "Enhanced and hierarchical structure algorithm for data imbalance problem in semantic extraction under massive video dataset." In: 68.3, pp. 641–657.

Garcia-Molina, H. (2008). "Database systems: the complete book." In: *Pearson Education India.*

García S., Fernández A. Herrera F. (2009). "Enhancing the effectiveness and interpretability of decision tree and rule induction classifiers with evolutionary training set selection over imbalanced problems." In: 4.9, pp. 1304–1314.

García-Pedrajas N., García-Osorio C. (2003). "Boosting for class-imbalanced datasets using genetically evolved supervised non-linear projections." In: 2.1, pp. 29–44.

Ghahramani, Z. (2003). "Unsupervised learning". In: pp. 72–112.

Gong J., Kim H. (2017). "RHSBoost: Improving classification performance in imbalance data." In: 111, pp. 1–13.

Grobelnik, M. (1999). "Feature selection for unbalanced class distribution and naive bayes." In: *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 258–267.

Guan D., Yuan W. Lee-Y. K. Lee S. (2009). "Nearest neighbor editing aided by unlabeled data." In: *Information Sciences* 179.13, pp. 2273–2282.

Guo H., Viktor H. L. (2004). "Learning from imbalanced data sets with boosting and data generation: the databoost-im approach." In: 6.1, pp. 30–39.

Gutiérrez P. A., Perez-Ortiz M. Sanchez-Monedero J. Fernandez-Navarro F. Hervas-Martinez C. (2015). "Ordinal regression methods: survey and experimental study." In: 28.1, pp. 127–146.

Haixiang G., Yijing L.-Shang J. Mingyun-G. Yuanyue H.- Bing G. (2018). "Learning from class-imbalanced data: Review of methods and applications." In: 73, pp. 220–239.

Han J., Chang-K. C. (2002). "Data mining for web intelligence." In: 11.35, pp. 64–70.

Hand, D. J. (1999). "Statistics and data mining: intersecting disciplines." In: 1.1, pp. 16–19.

Hart, P. (1968). "The condensed nearest neighbor rule (corresp.)" In: *IEEE transactions on information theory* 14.3, pp. 515–516.

Hastie T., Tibshirani R.- Friedman J. (2009). "Overview of supervised learning". In:

He H., Garcia-E. A. (2009). "Learning from imbalanced data." In: 21.9, pp. 1263–1284.

He H., Bai Y.-Garcia E. A.- Li S. (2008). "ADASYN: Adaptive synthetic sampling approach for imbalanced learning." In: pp. 1322–1328.

Herrera F., Carmona C. J. González P.- Del Jesus-M. J. (2011). "An overview on subgroup discovery: foundations and applications." In: 29.3, pp. 495–525.

Herrera F., Charte F. Rivera A. J.- Del Jesus-M. J. (2016). "Multilabel classification." In: pp. 17–31.

Holte R. C., Acker L. Porter-B. W. (1989). "Concept Learning and the Problem of Small Disjuncts." In: 89, pp. 813–818.

Hu S., Liang Y. Ma L. He Y. (2009). "MSMOTE: Improving classification performance when training data is imbalanced." In: pp. 13–17.

Huang J., Ling C. X. (2005). "Using AUC and accuracy in evaluating learning algorithms". In: 3.17, pp. 299–310.

Huang S., Cai N. Pacheco P. P. Narrandes S. Wang-Y. Xu W. (2018). "Applications of support vector machine (SVM) learning in cancer genomics". In: 1.15, pp. 41–51.

Iannaccone, G. (2006). "Fast prototyping of network data mining applications." In: pp. 41–50.

Jackowski K., Krawczyk B. Woźniak M. (2012). "Cost-sensitive splitting and selection method for medical decision support system." In: *International Conference on Intelligent Data Engineering and Automated Learning .*, pp. 850–857.

Japkowicz N., Stephen S. (2002). "The class imbalance problem: A systematic study. " In: *Intelligent data analysis* 6.5, pp. 429–449.

Japkowicz, N. (2000). "Learning from imbalanced data sets: a comparison of various strategies." In: *AAAI workshop on learning from imbalanced data sets* 68, pp. 10–15.

Jordan, M. I. (2004). "Graphical models. Statistical science". In: *Science* 19.1, pp. 140–155.

Joshi M. V., Kumar V. Agarwal R. C. (2001). "Evaluating boosting algorithms to classify rare classes: Comparison and improvements." In: pp. 257–264.

Kasemsap, K. (2018). "Multifaceted applications of data mining, business intelligence, and knowledge management." In: pp. 810–825.

Khreich W., Granger E. Miri A. Sabourin R. (2010). "Iterative Boolean combination of classifiers in the ROC space: An application to anomaly detection with HMMs". In: *Pattern Recognition* 43.8, pp. 2732–2752.

Kotsiantis S., Kanellopoulos D. Pintelas P. (2006). "Handling imbalanced datasets: A review." In: *GESTS International Transactions on Computer Science and Engineering* 30.1, pp. 25–36.

Kotsiantis S. B., Zaharakis I. Pintelas P. (2007). "Supervised machine learning: A review of classification techniques." In: *Emerging artificial intelligence applications in computer engineering.* 160.1, pp. 3–24.

Krawczyk, B. (2016). "Learning from imbalanced data: open challenges and future directions". In: *Progress in Artificial Intelligence* 5.4, pp. 221–232.

Kubat M., Matwin S. (1997). "Addressing the curse of imbalanced training sets: one-sided selection." In: *Icml* 97, pp. 179–186.

Kubat M., Holte R. C. Matwin S. (1998). "Machine learning for the detection of oil spills in satellite radar images." In: *Machine learning.* 30.2, pp. 195–215.

Kuncheva, L. I. (2002). "Switching between selection and fusion in combining classifiers: An experiment." In: 2.32, pp. 146–156.

Kuonen, D. (2004). "Data mining and Statistics: What is the connection?." In: 30, pp. 1–6.

Lakshmi B. N., Raghunandhan G. H. (2011). "A conceptual overview of data mining". In: pp. 27–32.

Landgrebe T., Paclík P. Tax D. M. Verzakov S. Duin R. P. (2004). "Cost-based classifier evaluation for imbalanced problems." In: pp. 762–770.

Lane T., Brodley C. E. (1998). "Approaches to Online Learning and Concept Drift for User Identification in Computer Security." In: *KDD*, pp. 259–263.

Lewis D. D., Catlett J. (1994). "Heterogeneous uncertainty sampling for supervised learning." In: *Machine learning proceedings*, pp. 148–156.

Liao J. G., Chin K. V. (2007). "Logistic regression for disease classification using microarray data: model selection in a large p and small n case." In: *Bioinformatics* 23.15, pp. 1945–1951.

Ling C. X., Li C. (1998). "Data mining for direct marketing: Problems and solutions." In: 98, pp. 73–79.

Ling C. X., Sheng V. S. (2008). "Cost-sensitive learning and the class imbalance problem." In: *Encyclopedia of machine learning.* 2011, pp. 231–235.

Liu Y. H., Chen Y. T. ((2005, October)). "Total margin based adaptive fuzzy support vector machines for multiview face recognition". In: *IEEE International Conference on Systems, Man and Cybernetics* 2, pp. 1704–1711.

Liu, B. (2011). "Supervised learning". In: pp. 63–132.

Lusa, L. (2017). "Gradient boosting for high-dimensional prediction of rare events." In: 113, pp. 19–37.

López V., Fernández A. García S. Palade V. Herrera F. (2013). "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics." In: *Information sciences* 250, pp. 113–141.

Maloof, M. A. (2003). "Learning when data sets are imbalanced and when costs are unequal and unknown." In: 2, pp. 2–1.

Mazurowski M. A., Habas P. A. Zurada J. M. Lo J. Y. Baker J. A. Tourassi G. D. (2008). "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance." In: *ibn* 21.2-3, pp. 427–436.

McCallum A., Nigam K. (1998). "A comparison of event models for naive bayes text classification." In: *AAAI-98 workshop on learning for text categorization* 752,1, pp. 41–48.

McCulloch W. S., Pitts W. (1943). "A logical calculus of the ideas immanent in nervous activity." In: *The bulletin of mathematical biophysics* 4, pp. 115–133.

Metsis V., Androutsopoulos I. Paliouras G. (2006.). "Spam filtering with naive bayes-which naive bayes?" In: *CEAS* 17, pp. 28–69.

Michie D., Spiegelhalter D. J. Taylor C. C. (1994). "Machine learning, neural and statistical classification". In:

Mining, W. I. D. (2006). "Data mining: Concepts and techniques." In: 10, pp. 559–569.

Mitchell, T. M. (1997). "Machine learning." In: *Emerging artificial intelligence applications in computer engineering.*, pp. 3–24.

Moepya S. O., Akhoury S. S. Nelwamondo F. V. (2014). "Applying cost-sensitive classification for financial fraud detection under high class-imbalance." In: pp. 183–192.

Nanni L., Fantozzi C. Lazzarini N. (2015). "Coupling different methods for overcoming the class imbalance problem." In: *Neurocomputing* 158, pp. 48–61.

Ortigosa-Hernández J., Inza I. Lozano J. A. (2016). "Towards competitive classifiers for unbalanced classification problems: A study on the performance scores." In:

Pan S. J., Yang Q. (2009). "A survey on transfer learning." In: 22.10, pp. 1345–1359.

Penar W., Wozniak M. (2010). "Cost-sensitive methods of constructing hierarchical classifiers." In: *Expert Systems.* 27.3, pp. 146–155.

Penrod C. S., Wagner T. J. (1976). "Another look at the edited nearest neighbor rule." In: *. TEXAS UNIV AT AUSTIN DEPT OF ELECTRICAL ENGINEERING.*

Piatetsky-Shapiro G., Brachman R. J. Khabaza T. Kloesgen W. Simoudis E. (1996). "An overview of issues in developing industrial data mining and knowledge discovery applications." In: 96, pp. 89–95.

Ponti Jr, M. P. (2011). "Combining classifiers: from the creation of ensembles to the decision fusion." In: pp. 1–10.

Prachuabsupakij W., & Kulchan S. (2014). "Performance comparison of decomposition methods in multiclass imbalanced datasets." In: pp. 28–30.

Pratama R. F. W., Purnami S. W. Rahayu S. P. (2018). "Boosting support vector machines for imbalanced microarray data." In: 144, pp. 174–183.

Prati R. C., Batista G. E. Monard M. C. (2004). "Class imbalances versus class overlapping: an analysis of a learning system behavior." In: *Mexican international conference on artificial intelligence*, pp. 312–321.

Prati R. C., Batista G. E. Silva D. F. (2015). "Class imbalance revisited: a new experimental setup to assess the performance of treatment methods." In: 45.1, pp. 247–270.

Radivojac P., Chawla N. V. Dunker A. K. Obradovic Z. (2004). "Classification and knowledge discovery in protein databases." In: *Journal of Biomedical Informatics* 37.4, pp. 224–239.

Rajaraman A., Ullman J. D. (2011). "Mining of massive datasets." In: *Cambridge University Press.*

Saito T., Rehmsmeier M. (2015). "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets." In: 3.10.

Sajedin A., Zakernejad S. Faridi S. Javadi M. Ebrahimpour R. (2010). "High impedance fault detection using combination of multi-layer perceptron neural networks based on multi-resolution morphological gradient features of current waveform." In: 69.5.

Sarlak M., Shahrtash S. M. (2011). "High impedance fault detection using combination of multi-layer perceptron neural networks based on multi-resolution morphological gradient features of current waveform." In: 5.5, pp. 588–595.

Seiffert C., Khoshgoftaar T. M. Van Hulse J. Napolitano A. (2009). "RUSBoost: A hybrid approach to alleviating class imbalance." In: 40.1, pp. 185–197.

Shamsollahi M., Badiee A. Ghazanfari M. (2019). "Using combined descriptive and predictive methods of data mining for coronary artery disease prediction: a case study approach." In: 1.7, pp. 47–58.

Silver M., Sakata T. Su H. C. Herman C. Dolins S. B. O Shea M. J. (2001). "Case study: how to apply data mining techniques in a healthcare data warehouse." In: 2.15, pp. 155–164.

Siraj F., Abdoulha M. A. (2007). "Mining enrolment data using predictive and descriptive approaches." In: pp. 53–72.

Storkey, A. (2009). "When training and test sets are different: characterizing learning transfer. " In: *Dataset shift in machine learning* 30, pp. 3–28.

Sun Y., Kamel M. S. Wong A. K. Wang Y. (2007). "Cost-sensitive boosting for classification of imbalanced data." In: 40.12, pp. 3358–3378.

Ting, K. M. (2000). "A comparative study of cost-sensitive boosting algorithms." In: 99, pp. 97–105.

Tomek, I. (1976). "Two modifications of CNN." In: 11.6, 769–772. URL: doi:10.1109/tsmc.1976.4309452.

Turney, P. D. (1994). "Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm." In: *Journal of artificial intelligence research.* 2.12, pp. 369–409.

Wang H., Wang S. (2008). "A knowledge management approach to data mining process for business intelligence." In:

Wei H., Sun B. Jing M. (2014). "BalancedBoost: A hybrid approach for real-time network traffic classification." In: pp. 1–6.

Weiss G. M., Provost F. (2003). "Learning when training data are costly: The effect of class distribution on tree induction." In: 19, pp. 315–354.

Weiss G. M., McCarthy K. Zabar B. (2007). "Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?." In: *Dmin* 7.35-41, p. 24.

Wilson, D. L. (1972). "Asymptotic properties of nearest neighbor rules using edited data." In: *IEEE Transactions on Systems, Man, and Cybernetics* 3, pp. 408–421.

Wu X., Kumar V. Quinlan J. R. Ghosh J. Yang Q. Motoda H. ... Steinberg D. (2008). "Top 10 algorithms in data mining." In: *Knowledge and information systems* 14.1, pp. 1–37.

Yamazaki K., Kawanabe M. Watanabe S. Sugiyama M. Müller K. R. (2007). "Asymptotic bayesian generalization error when training and test distributions are different." In: *Proceedings of the 24th international conference on Machine learning*, pp. 1079–1086.

Yang Z., Tang W. H. Shintemirov A. Wu Q. H. (2009). "Association rule mining-based dissolved gas analysis for fault diagnosis of power transformers". In: *IEEE Transactions on Systems, Man, and Cybernetics* 39.6, pp. 597–610.

Yoon K., Kwek S. (2005). "An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics." In: *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, 6–pp.

Yusof R., Kasmiran K. A. Mustapha A. Mustapha N. Mohd Zin N. A. (2017). "echniques for handling imbalanced datasets when producing classifier models." In: *Journal of Theoretical and Applied Information Technology*, 95.7, pp. 1425–1440.

Zadrozny B., Elkan C. (2001). "Learning and making decisions when costs and probabilities are both unknown." In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 204–213.

Zhang X., Hu B. G. (2014). "A new strategy of cost-free learning in the class imbalance problem." In: 26.12, pp. 2872–2885.

Zhang S., Qin Z. Ling C. X. Sheng S. (2005). "Missing is useful": Missing values in cost-sensitive decision trees." In: *IEEE transactions on knowledge and data engineering.* 17.12, pp. 1689–1693.

Zhao X. M., Li X. Chen L. Aihara K. (2008). "Protein classification with imbalanced data. Proteins: Structure, function, and bioinformatics". In: 70.4, pp. 1125–1132.

Zhou Z. H., Liu X. Y. (2005). "Training cost-sensitive neural networks with methods addressing the class imbalance problem." In: 18.1, engineering.

Zhou Q., Zhou H. Li T. (2016). "Cost-sensitive feature selection using random forest: Selecting low-cost subsets of informative features." In: *Knowledge-based systems.* 95, pp. 1–11.

Zhu X., Wu X. (2004). "Class noise vs. attribute noise: A quantitative study." In: 22.3, pp. 813–818.

Zhu, X. (2007). "Semi-supervised learning tutorial." In: pp. 1–135.