Wholly owned by UTAR Education Foundation
(Co. No. 578227-M)
DU012(A)

MECM15110
PROJECT

Project Title: Predicting Customer Buying Decisions for Online Shopping with Unbalanced Data Set

NAME	YAP CHAU TEAN
STUDENT ID	2000681
SUPERVISOR	DR. KHOR KOK CHIN

Table of Content

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

●K-Nearest Neighbor (KNN) ●Naïve Bayers ●J48 ●Support Vector Machine (SVM)

●Sequential Minimal Optimization (SMO) ●Multilayer Perceptron (MLP)

ABSTRACT

One of the common e-commerce problems is the low purchase conversion rate. Data mining techniques can help tackle the problem by analysing and predicting the customer purchase intention to give better service and better recommendations to customers. In this project, the real-time online shoppers purchasing intention data set from Sakar *et al.* (2018) was used. The data set is unbalanced as it consists of 15.5% of the positive class and 84.5% of the negative class. Weka, a data mining tool, provides the facility to classify the data set with different machine learning algorithms. Six machine learning algorithms were applied and compared based on the classification evaluation methods. The algorithms involved were K-Nearest Neighbor (KNN), Naïve Bayers, J48, Support Vector Machine (SVM), Sequential Minimal Optimization (SMO) and Multilayer Perceptron (MLP). Data pre-processing on the data set may improve the classification results. The methods used were over-sampling, under-sampling and hybrid sampling, which modified the data set class distribution to achieve a better result. The hybrid sampling method gave comparable classification results compared with Sakar *et al.* (2018). Ensemble learning methods AdaBoost and Bagging were tested but showed no improvement on this online shoppers purchasing intention data set.

CHAPTER 1

1. INTRODUCTION

Online shopping, also known as e-commerce, is a very popular trend in this era and is expected to continue to grow and expand in the future. There was a pandemic outbreak of Covid-19 worldwide in the early of the year 2020. People were therefore encouraged to stay at home during the lockdown. During the lockdown period, most people chose online shopping to get the necessary product, and some of them had never tried buying online before. This has boosted up the online shopping trend worldwide. In April 2020, the global e-commerce retail sales had grown by 207 % compared to last year's same month (ACI Worldwide, 2020).

1.1 Research Background

The common issue in e-commerce is the low conversion rate which the number of customers who completed transactions in online shops is lower than the number of customers who visit the shop. The purchase conversion rate is often dealers' concern as the resources had already invested in the online shop and need to manage it frequently. The study by Chaffey (2020) had stated that the average conversion rate for e-commerce is only 3.3% that could be viewed by the conversion rate funnel.

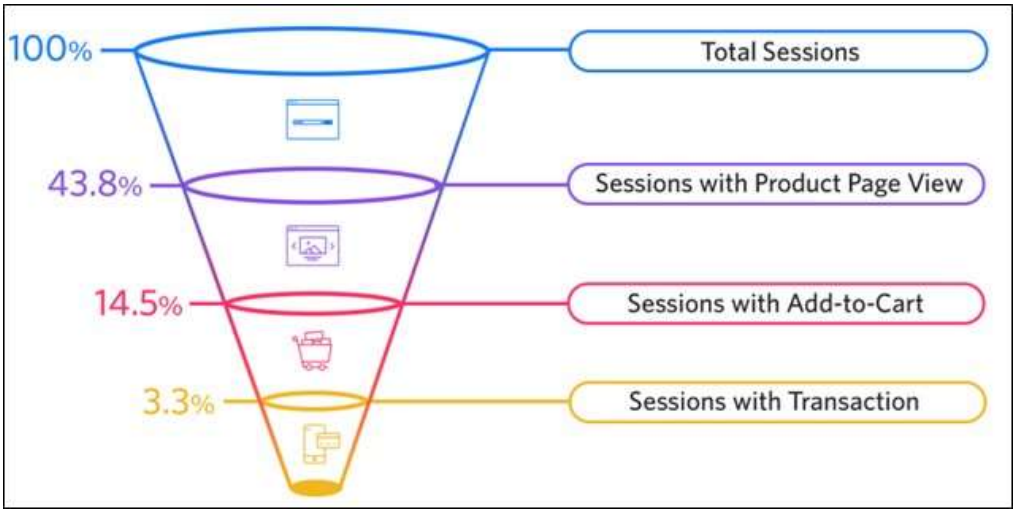


Figure 1.1: E-commerce conversion funnel. (Chaffey, 2020)

The resources and expertise in an SME sector are usually limited. Therefore, efficiency and cost-effectiveness are important when managing an online shop (Cronin-Gilmore, 2012; Grandón et al., 2011). Di Fatta, Patton and Viglia (2018) investigated how e-commerce websites can positively affect the conversion rates based on the management of promotions and quality. Manipulated with promotions and quality are the good starting points for improving the conversion rate.

Unlike physical retail shops and the traditional commercial way in which the promotions usually can be provided by experienced salespersons to different customers based on the observation, these experiences can improve the sales figure and purchase conversion rate in online shops (Moe, 2003). Some e-commerce and IT companies are creating such experiences by acting like a salesperson in online shops via early detection and behaviour predicting systems (Rajamma et al., 2009; Albert and Hartford, 2004). At the same time, researchers also studied this issue from a different perspective with machine learning, according to the navigation pattern or predicting the real-time customer behaviour by taking a corresponding action to minimise the abandonment rate of purchase by customers.

Sakar et al. (2018) developed an analysis system based on real-time user behaviour for online shopping. The system could detect the visitors who have purchasing intentions but may leave the site. It will take corresponding actions to improve the online shopping abandon rate and the purchase conversion rates. The data set used in this research was pre-processed with over-sampling and feature selection methods. The best prediction of customer behaviour is achieved by using a multilayer perceptron network with an accuracy of 87.24% and a true positive rate of 84%; the result was better than the decision tree algorithm or support vector machines.

In short, the online shopping customers' buying decision predicting system is very important to improve the low purchase conversion rate problem, with the combination of corresponding actions taken based on the outcomes from the predicting system.

1.2 Project Overview

In this project, data set from Sakar et al. (2018) was used. Since the data set consists of only 15.5% of the positive class (Buy), the predicting results on the positive class are weak. Therefore, methods such as modifying the unbalanced class distribution data will be carried out to obtain better

predicting results. The pre-processing methods applied in this project are over- sampling and under-sampling.

Six machine learning algorithms were evaluated in this project to compare their performance in predicting buying decisions. They are K-Nearest Neighbor (KNN), Naïve Bayers, J48, Support Vector Machine (SVM), Sequential Minimal Optimization (SMO) and Multilayer Perceptron (MLP).

The results from each algorithm will be visualised in a table to determine the most suitable algorithms for customers' buying decision predicting system.

This project used Weka - a tool for data mining, analysis and visualisation. Weka stands for Waikato Environment for Knowledge Analysis; it contains various visualisation tools and algorithms which can be used for data mining and machine learning. The tool is used in this project to increase the efficiency of applying machine learning and data visualisation. With the Weka tool, the unbalanced data set solution can be visualised to obtain better results and then compare the machine learning algorithms. Finally, the best combination of learning algorithms and the sampling pre-processing method on the unbalanced data set can be determined for predicting customers' buying decisions.

1.3 Problem Statement

The most common problem of online shops is the low purchase conversion rate, which means fewer buyers than the total number of visitors. If the customers' behaviour can be detected and predicted earlier, further actions can be carried out to increase the chances for customers to buy products.

The data set for online customers are usually unbalanced – the majority is the negative class (Not Buy), and the minority is the important positive class (Buy). Such unbalanced class distribution in the data set will greatly affect the prediction for the positive class, as the machine learning algorithms tend to favour the majority class.

1.4 Objectives

To predict the customers' buying decisions by applying various machine learning algorithms.

To improve the predicting performance of the algorithms for the minority yet important class (BUY) in the data set using sampling techniques, ensemble method and multiple classifiers.

1.5 Scope

This project is a data mining project, and it is to study the predictive capability of the selected machine learning algorithms using the online shoppers purchasing intention data set provided by Sakar *et al.* (2018). The experiments are conducted using Weka, the data mining and analysis tool.

CHAPTER 2

2. LITERATURE REVIEW

2.1 Online Purchasing Portal

There are many existing online shopping portals in the current information age, including giants in this e-commerce field such as Amazon, eBay and Alibaba. Amazon was started as a bookseller since the year 1994 and continues growing until now, with over 500 million products sold. The success of Amazon has become a role model for the newly started online business portals. Meanwhile, eBay is another e-commerce portal that includes an auction site. eBay has had over 147 million registered users in over 30 countries since the year 2005, and continue to grow until now. Alibaba was founded in the year 1999, now the biggest e-market portal in China. While in Malaysia, there are also a few famous online shopping portals such as Shopee and Lazada.

More and more competitors are joining to share the pie of this e-commerce world. The world has changed since e-commerce became so popular and affects our lives from the social and economic aspects. Converting physical business to online business may be fraught with challenges like the expensive cost for Internet development and fulfillment, but the problem solves if one joins the existing portals. The existing portals generally allow online presence development and favourable business proposition for retailers (Kennedy and Coughlan, 2006). To compete and stand among all the e-commerce shops, the shop's owner must consider e-commerce strategies. For example, a new perspective of seeing the product with augmented reality or visual reality technology, learning about customer behaviour with artificial intelligence/machine learning, or implementing a mobile shopping portal application. Therefore, the machine learning strategies will be focused to tackle the e-commerce problem as stated in this project.

2.2 Existing Research

Online shopping has brought convenience, but it has some issues that the online shop owner is always concerned about, such as the low purchase conversion rate. There are two categories of research for this issue: (i) to study the navigation path of visitors or (ii) to predict the users' behaviour in real-time. Hu *et al.* (2020) studied the online shopping customer purchase behaviour by analysing the customer online purchase data with a deep forest algorithm and developing a predicting system for online customer purchase intention. This data set used in the study is the behavioural data collected from information from Alibaba's online shopping portal in 2017. The data consists of 16,880 users and 393,798 products. The data attributes include user id, product id, product category id, type of user behaviour on the product and the time of the behaviour. Since the data set has an unbalanced class distribution in which the number of "not purchase" is higher than the "purchase", therefore it is not suitable to use the accuracy to measure the predicting performance of the machine learning algorithms. In the study, they measure the performance by using the F1 value. F1 value is the weighted harmonic average of the recall and accuracy for evaluating the predicted results from the system. The performance of the algorithms is as shown in Table 2.1, and the result showed that the deep forest algorithm achieved a higher F1 value than the other algorithms.

Table 2.1: Comparison of algorithm performance in the study Hu *et al.* (2020).

	Training Time (s)	**F1 Result (%)**
SVM	37	7.21
random forest	18	9.01
Xgboost	29	6.78
deep neural network	1021	8.09
deep forest	41	9.51

Another literature study used the unbalanced data of an e-commerce portal and analyses with cat-boost model to predict whether a customer will purchase a specific product (Dou, 2020). In the study, the accuracy and precision of the model were used to evaluate the predictive performance. The Cat-boost model is good in auto-processing the variables, which decreases the steps of processing the previous data and decreases the loss of information when dealing with unbalanced

data sets. The original data set's information can be fully mined and prevent the problem of over-fitting. The data set analysed consists of 12,316 records. 10,303 records are negative class (no buy), and 1,889 are positive class (end up buy). Due to the unbalanced data, AUC-ROC and F1 were chosen to measure the performance of the model. The result obtained an accuracy of 88.51% and a recall rate of 84.48%.

Mohammed et al. (2018) built models to predict potential customers of a POS machine in a bank system using J48 and Naïve Bayes learning algorithms. In this study, the data set collected from the UK-Bank data repository consist of 5000 users' data. They had compared J48 and Naïve Bayes algorithms. The result showed that J48 had 89.72% classification accuracy, better than Naïve Bayes, which had 89.58% classification accuracy.

In Rusmee and Chumuang (2019) study, they built a prediction system for the consumers' buying decision on the personal car by applying the SMO learning algorithm. The data set was collected from the Toyota center and consisted of 1,110 data. The result obtained from the model created with SMO showed an accuracy of 95.13% and an error rate of 0.05.

Nayyar (2019) had built a predicting model for customers' purchase behaviour based on customers' gender, age and salary data. The author had compared Logistic Regression, KNN, SVM, Decision Tree and Random Forest for the predicting models. The result evaluation in this study was based on the confusion matrix. The result showed that SVM with non-linear kernel support had the best performance among all with an accuracy of 93.0%, a true positive rate of 95.5% and a true negative rate of 87.8%

Xu et al. (2020) proposed a model of analysing customer behaviour data to improve customer satisfaction by utilising the collect and deliver (CDP) location for an online shop. This model predicted the purchase probability to optimise the CDP location. Real customer behaviour data consisting of 257,685 records were used for this study. The records are unbalanced data set and five machine learning algorithms include naïve Bayes, gradient boosting tree, random forest, logistic regression and multilayer perceptron was applied for the model. The results were compared and showed that the gradient boosting trees algorithms had the best performance.

Meanwhile, in the study by Moe (2003), the classification model for buy and no buy classes by visit behaviours was developed and tested. In the study, page-to-page clickstream data from an online store was collected and analysed; the data are categorised as buying, browsing or searching

based on the study from the clickstream data patterns, such as the pages viewed of the product. Each type of visit has represented different purchasing likelihood from customers. Shop owners are also driven by different motivations and would act corresponding to various marketing messages. The data collection was done by a market research firm, NetConversions, which was employed by the store site. NetConversions used cookies downloaded onto the visitor's computer so that the store site can track and record the shopper's behaviour at that webpage. The real-time clickstream data was evaluated and updated with the result of purchasing rate, abandon rate, and promotional response rate of a customer when that customer visited the online shop.

In the study by Mînăstireanu and Meșniță (2020), they analysed the bank user data from fraud detection system with machine learning algorithms. The data set is highly unbalanced because the non-fraudulent case classes are the majority and dominate the fraudulent case class. In the study, they had stated three ways for handling unbalanced data sets. The first way is resampling methods such as under-sampling and over-sampling; the second way is cost-sensitive training and the third way is using tree algorithms such as decision tree and random forest. The best result was obtained using SMOTE, an over-sampling method. The performance of the classification models was evaluated with the AUC PR curve, and high precision of 94% was obtained with a random forest classifier.

C and Ravikumar (2019) studied the suitable model to predict the customers who will do more purchasing in the online shop and take the corresponding action to improve the sales. In this study, the data set was collected from various online shopping portals. Data Mining tools and techniques were implemented to analyse the huge data set. The data mining techniques such as data pre-processing and feature selection was applied. In conclusion, they found the best result was obtained with features selection PCA algorithm, with an accuracy of 89%. The result can be further improved with features extraction techniques which finally obtained an accuracy of 92.18%.

Another research from Dang et al. (2020) had studied and analysed online purchase behaviour for young generations. They proposed a structured model that analysed the effects of four attributes, such as information adoption, personalised service, perceived switching risk and habitual behaviour on the buying intention in online shopping. The data set analysed in this paper was collected from 407 users on Taobao who are in the 90s generation. A structural equation modeling was applied and the result showed that information adoption, personalised service and perceived

switching risk are the key factors that affect the online purchase intention in young generations, while habitual behaviour had a negative influence on online purchase intention. The correlation coefficient was used in the structured model to analyse the influence of each attribute on online purchase intention.

From the literature review above, it can be concluded that the problem of conversion rate existing in e-commerce is always studied to overcome the issue with data mining data techniques. However, when dealing with the issue, the problem of unbalanced data will rise as in reality, the variance of the number of people who buy and the number of people who window shopping is big. Each of the literature papers above used different approaches, and the common learning algorithms applied are decision tree, SVM, KNN, and MLP to obtain better classification results. Hence, machine learning can effectively solve the problem of the e-commerce state in this project. In this project, six common learning algorithms will be applied which include K-Nearest Neighbor (KNN), Naïve Bayers, J48, Support Vector Machine (SVM), Sequential Minimal Optimization (SMO) and Multilayer Perceptron (MLP) as a comparison to deal with the problems in this project. With the aid of the Weka tool, the data set can be processed efficiently without coding the selected algorithms. With this, different combinations of data pre-processing solutions can be experimented to obtain the best solution to the problems.

2.3 Unbalanced Data Set and Its Solutions

With the continuous growth of data mining and machine learning, unbalanced data learning has become a concern. Data-level and algorithm-level methods are continuously being improved to overcome the unbalanced data issue. Recent trends show not only the disproportion between classes but other difficulties that exist in the real-time data. These problems have motivated data researchers to focus more on the efficient, adaptive and real-time data mining method.

In Krawczyk (2016), his research focuses on handling the unbalanced data as real-life data set distribution is usually skewed since classes always appear more frequently compared to others. This has caused difficulty for machine learning algorithms because they will always bias towards the majority group, while the minority class may contain more useful and important knowledge and become more important for the data mining research. When facing the skewed distribution in

the data, an intelligent system that can overcome the bias has to be designed. Meanwhile, in an unbalanced binary classification, the unbalance ratio may not be the only factor of learning difficulties. For example, the classes with high disproportion are well represented and come from non-overlapping distributions. Good classification results can still be obtained using canonical classifiers. Most contemporary works in class unbalanced data concentrate on unbalanced ratios ranging from 1:4 up to 1:100. But in real-life applications, the data sets may have unbalanced ratios ranging from 1:1000 up to 1:5000. This may cause new difficulties to data pre-processing and machine learning algorithms. The algorithms must be prepared for such extreme scenarios. Three general methods were introduced in the study: (i) modifying the train set and making it suit for any standard machine learning algorithm, (ii) modifying existing learners to ease their bias towards majority class or (iii) combining the two to focus their strengths and reduce their weaknesses. Combining data-level solutions with ensemble algorithms results in more efficient and robust learners.

In a study by Lee and Kim (2018), overlapped and unbalanced data sets were studied. Class overlap occurs if the region in the class consists of the same value of data for another class. When class overlapping occurs in an unbalanced data set, the classification process will become harder. If the overlapping issue does not exist in unbalanced data, the classifier rate will be more accurate. Most previous methods considered only the class unbalance problem, but their study aims to improve classification results with data sets that have both unbalanced and overlapped class problems. The study proposed a method based on OSM margin, which allows the user to separate the unbalanced and overlapping data set into soft and hard overlap regions to improve the classification process.

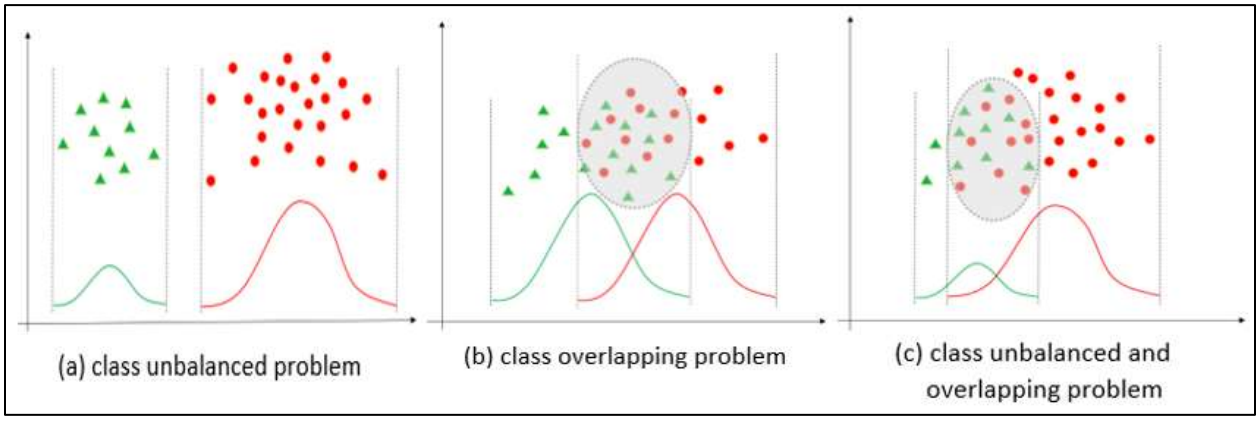


Figure 2.1: Example of unbalanced and overlapped data set adopted from Lee and Kim (2018).

Jacobusse and Veenman (2016) researched unbalanced data from law enforcement and medical screening. In their study, they showed how to resolve the selection bias issue when dealing with unbalanced data. The data sets analysed in the study was a synthetic data set and a real-world law enforcement data set. In their study, they found that applying Positive and Unlabelled (PU) learning to the data sets will improve the final classification performance. This method leaves out the labeled non-targets (negative class) and uses only the positive data and unlabeled data to obtain the best results. They considered class unbalanced with a ratio of 1 in 100 for positive class and negative class.

From the above literature studies, the unbalanced data set has become a more concerning issue. However, not only unbalanced data influent the performance of classification, but the overlapping of classes is also one of the reasons that affect the classification performance. To overcome the issues, researchers implemented modifications on the data set or the algorithms need to be implemented. According to the research papers, some researchers focus on pre-processing the data set with sampling methods to balance the class distribution and make the machine learning algorithms easy to understand. They also attempt to improve the classification performance with ensemble algorithms by combining two algorithms to enhance each strong point and overcome weaknesses.

2.3.1 Sampling Methods

Knowing that unbalanced and overlapped classes will affect the performance of the customer predicting model, one of the objectives in this project is to handle the unbalanced data set and overcome the weak predicting performance caused by the data set. In Ganganwar (2012) paper, the necessity of balancing unbalanced data was elaborated. They had reviewed the different unbalanced data handling solutions in this study, such as random over-sampling, Synthetic Minority Over-sampling Technique(SMOTE), random majority under-sampling, one-sided selection under-sampling and cost-sensitive boosting learning.

The study of Arafat et al. (2019) stated that unbalanced data classification is the most challenging research issue for supervised learning in data mining. Even though there are many data sampling methods introduced by past researchers to handle unbalanced data, learning with unbalanced data is a challenging task and still a focused research interest. Hence, the authors introduced a new

under-sampling model with the support vectors algorithm to balance the unbalanced data sets. The support vector will select instances from the majority class which is equal to the number of minority classes in the data set and form a balanced data set. The result obtained was compared with C4.5, Naïve Bayes, Random Forest and AdaBoost. In conclusion, the proposed model with under-sampling and Support Vector algorithms gave the best result.

Choirunnisa and Lianto (2017) used a combination of under-sampling and over-sampling to solve the unbalanced data problem. Five data sets were collected and each data set was pre-processed with the under-sampling method followed by the over-sampling method. Then, they carried out classification using random forest and decision tree C4.5. The result showed that with the combination of two sampling methods, the accuracy and ROC values had increased around 0.1% - 4.0% as compared to only handling the unbalanced data set with only one of the sampling methods.

Furthermore, McLean and Weaver (2018) stated that the impact of unbalanced data had caused high-cost losses in computing for the important classes. They introduced a new approach for handling an unbalanced data set, which is a hybrid classification method that combines algorithmic adaptations and multi-modal data formats. The evaluation metrics used in the study include accuracy, precision and specificity. The solution they designed is an ensemble learning algorithm that uses a custom over-sampling technique together with K-means and combined with a random under-sampling technique to counter any overfitting issue. The classification result has shown the effectiveness of the new algorithm in dealing with unbalanced commerce data.

Meanwhile, in Li and Zhou (2019) research, they introduced an improved over-sampling method from SMOTE known as TDSMOTE to solve the problem of classification effect on unbalanced data set. TDSMOTE divided minority samples into three regions and applied different over-sampling approaches at each region. In this study, 6 data sets from UCI data sets were selected. The data sets had an unbalanced ratio of 0.064 to 0.428. The over-sampling method SMOTE, BSMOTE, SVM-SMOTE and TDSMOTE were used to pre-process each data set and classification by random forest. The results were compared and the TDSMOTE approach had a better result than others based on G-mean, F-value and AUC. The average G-mean obtained with TDSMOTE on 6 data sets in the study is 0.8456, the average F-value is 0.8622 and the average AUC obtained is 0.8858.

From the studies of handling methods for unbalanced data sets above, sampling techniques for pre-processing unbalanced data sets showed effective outcomes on improving classification performance. In my project, the sampling methods will be utilised to improve the classification performance for the data set used and the result of each sampling method will be compared. The summary of each research was compiled and shown in Table 2.2:

Table 2.2: Summary of Literature Review on Unbalanced Data Set Solution (Sampling)

Authors	Research Title	Problem	Application	Result
Arafat et al. (2019)	An Undersampling Method with Support Vectors in Multi-class Imbalanced Data Classification	13 data sets with unbalanced ratios in between 5.55 and 853.	Under-sampling with support vector.	Most of the results with the proposed model are better than the other four algorithms.
Choirunnisa and Lianto (2017)	Hybrid Method of Undersampling and Oversampling for Handling Imbalanced Data	5 data sets from Keel with an unbalanced ratio in between 3.25 and 8.79	Hybrid with Under-sampling and Over-sampling.	The evaluation metric ROC shows an increase of 0.1 – 4 % with the hybrid method.
McLean and Weaver (2018)	Classification of Imbalanced Data in E-Commerce	5 data sets with minority class at 7.4 ~ 8.7%.	Over-sampling with K-mean and Under sampling	The solution shows better results compared to another common algorithm.
Mînăstireanu and Meșniță (2020)	Methods of Handling Unbalanced Datasets in Credit Card Fraud Detection	Kaggle credit card data set with minority class at 0.172%	Over-sampling and Under-sampling	The Best result was obtained by using SMOTE with a random forest classifier.

Li and Zhou (2019)	Research on Improving SMOTE Algorithms for Unbalanced Data Set Classification	6 UCI data sets with minority class at 6.4% ~ 42.8%	Over-sampling SMOTE, BSMOTE, SVM-SMOTE and TDSMOTE	TDSMOTE showed effective improvement in the classification performance for the unbalanced data set.

2.3.2 Ensemble Learning

Another study from Malhotra and Jain (2020) suggested handling unbalanced data sets with ensemble learning in predicting software defects. They had provided an experimental comparison of software defect prediction models with various boosting-based ensemble methods. Seven ensemble methods with re-sampling techniques were applied, and their performances were evaluated using stable metrics, such as Balance, G-Mean and AUC. The results showed that using re-sampling techniques before ensemble methods had significantly improved the model prediction performance. RUSBoost is the most suitable method among the seven methods, followed by MSMOTEBoost and SMOTEBoost.

Next, in Zhang et al. (2019) paper, an evolutionary-based ensemble under-sampling (EEU) algorithm was proposed to solve the problem of unbalanced data set classification. The EEU algorithm ensemble the under-sampling method with multiple based classifiers to improve the classification of the minority class in the unbalanced data set. In their study, 5 UCI data sets were selected, which had minority class percentages at 0.86% to 9.35%. The evaluation metrics used in this study are sensitivity, Matthews Correlation Coefficient (MCC), AUC and G-Mean. The authors had experimented with the EEU, RUS and KNN algorithms, the results showed EEU had the best result among the selected algorithms.

Xiao et al. (2020) used the ensemble cost-sensitive model with unbalanced customer credit data. Cost-sensitive learning allocates different costs to each wrong classification result and builds a classification model based on the principle of minimising the total wrong classification costs. The results obtained from the six data sets showed that the proposed model has a better classification

performance than the other models used in the study, such as Subagging, Semi-Bagging, CoBag, and Tri-training models.

Jiang and Hu (2014) introduced multiple classifiers known as the Dempster-Shafer fusion model by combining two different classification methods with Dempster-Shafer's rule. The combined classifier fully used the strengths between each classifier, purposed to obtain better classification performance for consumers' credit scoring. In the study, the two combined classification methods used are traditional linear Logistic Regression and nonlinear BP neural network. From the result obtained with the DS fusion model on the test sample, the accuracy is higher than applying each classification method separately. Therefore, using the DS theory of fusion can improve the classification accuracy of personal credit scoring. The second type error rate obtained from the test sample was slightly higher than the two single classification models, but overall the DS fusion model showed the advantage of classifying the personal credit score for the commercial banks.

Ahmed et al. (2018) introduced multiple classifier systems (MCS) by combining the boosting and stacking techniques. In this paper, two data sets were used. The first data set was telecom industries churn data from UCI data set that has 5000 data samples with minority class percentage at 14.3%. The second data set used was data gathered from a telecom operator in South Asia. Pre-processing method such as feature selection and sampling techniques was applied to the two data sets. They compared a few classifiers include K Nearest Neighbor, Artificial Neural Network, Decision Tree, Naïve Bayesian and Logistic Regression. From the result, MCS achieved 97.2% accuracy on the first data set and 86.3% accuracy on the second data set which is the best performance among other selected classifiers.

From the studies of handling methods for unbalanced data sets above, ensemble learning improved the classification performance for unbalanced data sets. The common ensemble learning applied was boosting and bagging, which will be used in this project. The summary of each research was compiled and shown in Table 2.3:

Table 2.3: Summary of Literature Review on Unbalanced Data Set Solution (Ensemble)

Authors	Research Title	Problem	Application	Result
Malhotra and Jain (2020)	Handling Imbalanced Data using Ensemble Learning in Software Defect Prediction	3 Java Software project data sets with minority class less than 22%.	Ensemble AdaBoost AdaBoostNC RUSBoost MSMOTEB	RUSBoost model showed better results compared to others.
Zhang et al. (2019)	Evolutionary-Based Ensemble Under-Sampling for Imbalanced Data	5 UCI data sets with minority class at 0.86% ~ 9.35%	Evolutionary-Based Ensemble Under-Sampling method	EEU algorithm improved the classification result compared to RUS and KNN
Xiao et al. (2020)	Cost-sensitive semi-supervised selective ensemble model for customer credit scoring	5 data sets with an unbalanced ratio between 1.2 and 13.3.	Ensemble cost-sensitive with Semi-supervised learning.	GMDH-based cost-sensitive semi-supervised selective ensemble model gave better results.
Jiang and Hu (2014)	Combining multiple classifiers based on Dempster-Shafer theory for personal credit scoring	Consumer credit data from Shenzhen bank	Fusion model with logistic regression, BP neural network	The combined model had higher accuracy and lower error than the single classifiers.
Ahmed et al. (2018)	MCS: Multiple classifier systems to predict the churners in the telecom industry	UCI data sets with minority class at 14.3%	MCS with combining the boosting and stacking techniques.	The proposed MCS was more accurate than the individual classifier.

2.4 Data Set Overview

The data set analysed in this project is the online shoppers purchasing intention data set from Sakar et al. (2018). The authors C. Okan Sakar and Yomi Kastro. From this data set, each session represents different user data in one year. The data set had avoided any specific campaign or promotion day, and user profile information which may disrupt the result.

This data set consists of 12,330 sessions, with 10,422 rows as the negative class (Not Buy) and 1,908 rows as the positive class (Buy). The percentage of the data distribution is 84.5% for the negative class and only 15.5% for the positive class. Thus, the ratio of the data set for the positive class to the negative class is 1:6, which is unbalanced distribution. The attributes in the data consist of Administrative, Administrative duration, Informational, Informational duration, product related, Product related duration, Bounce rate, Exit rate, Page value, Special day, Month, Operating Systems, Browser, Region, Traffic Type, Visitor Type, Weekend and Revenue. Each attribute is described in Table 3.

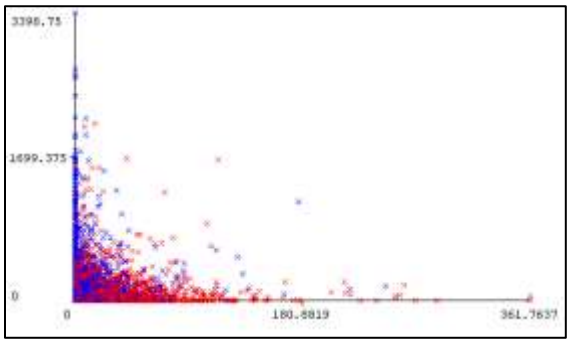
Table 2.4: Data Set Attribute Description Adopted from Sakar et al. (2018)

Attribute Name	Description
Administrative	Represent the number of pages view on user account related page
Administrative_Duration	Represent the time user spent on view the account related page
Informational	Represent the number of pages view on webpage and shop detail
Informational_Duration	Represent the time user spent on view the shop details related page
ProductRelated	Represent the number of pages view on product detail page
ProductRelated_Duration	Represent the time user spent on view the product detail page
BounceRates	Represent the rate of user leave without further view on the website
ExitRates	Represent the rate of user leave after view the website
PageValues	Represent the page visited by user before completed the transaction
SpecialDay	Represent the day when visited website was on holiday or not
Month	Represent the month when visited the website
OperatingSystems	Represent user's operating system
Browser	Represent user's browser
Region	Represent user's location
TrafficType	Represent the sources of user reached the website
VisitorType	Represent the type of user whether first visit or returning visit.
Weekend	Represent the day when visited website was on weekend or not
Revenue	The class, positive (Buy) and negative (Not Buy)

From the data set, the descriptive statistic for each attribute with numeric data was measured, and the overlapping graph was generated using the Weka tool. The detailed information is shown in Table 4.

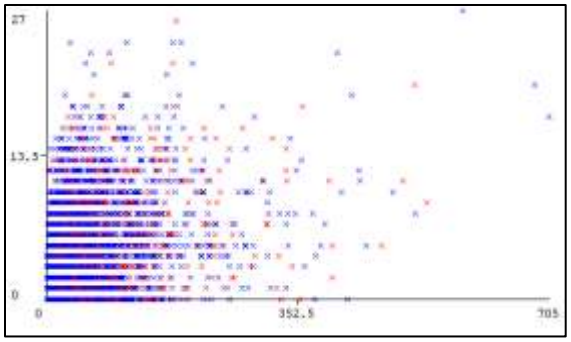
Table 2.5: Descriptive Statistic for each attribute in the data set. (Sakar et al., 2018)

Attribute Name	Minimum	Q1	median	Q3	Maximum	Mean	Std Deviation
Administrative	0	0	1	4	27	2.315	3.322
Administrative_Duration	0	0	7.5	93.2563	3398.75	80.819	176.779
Informational	0	0	0	0	24	0.504	1.27
Informational_Duration	0	0	0	0	2549.375	34.472	140.749
ProductRelated	0	7	18	38	705	31.731	44.476
ProductRelated_Duration	0	184.138	598.937	1464.16	63973.52	1194.75	1913.669
BounceRates	0	0	0.00311	0.01681	0.2	0.022	0.048
ExitRates	0	0.01429	0.02516	0.05	0.2	0.043	0.049
PageValues	0	0	0	0	361.764	5.889	18.568
SpecialDay	0	0	0	0	1	0.061	0.199
OperatingSystems	1	2	2	3	8	2.124	0.911
Browser	1	2	2	2	13	2.357	1.717
Region	1	1	3	4	9	3.147	2.402
TrafficType	1	2	2	4	20	4.07	4.025

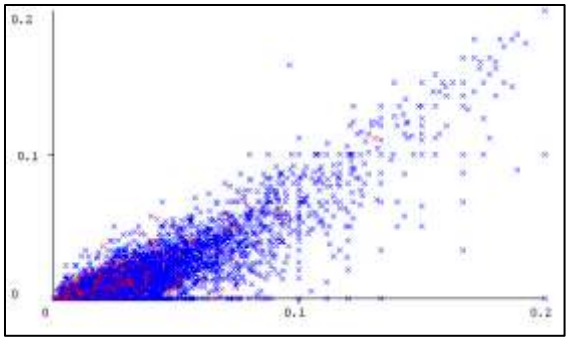


(a) Administrative Duration vs Page Values

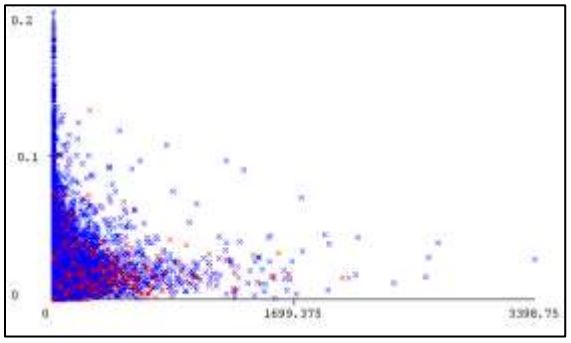
(b) Administrative vs Product Related Duration



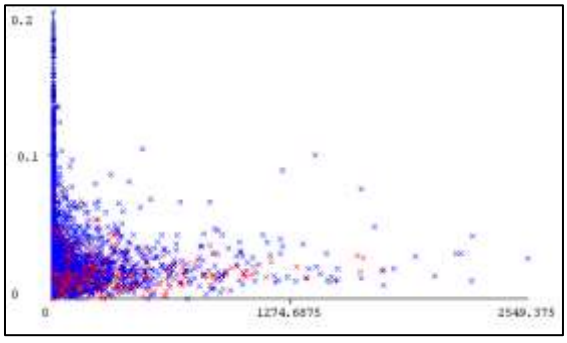
(c) Administrative vs Product Related



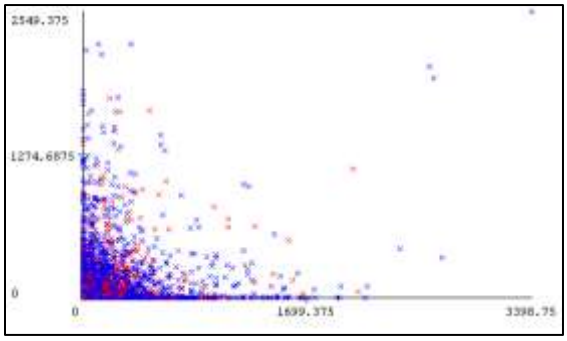
(d) Bounces Rates vs Exits Rates



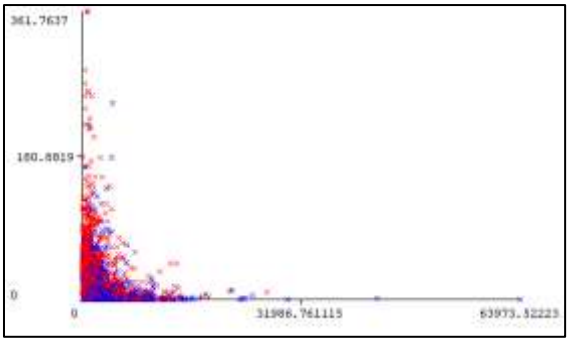
(e) Exits Rates vs Administrative Duration



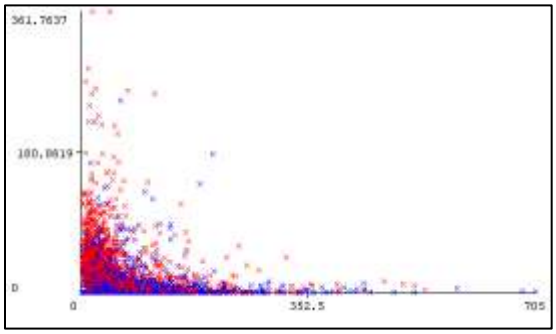
(f) Exits Rates vs Informational Duration



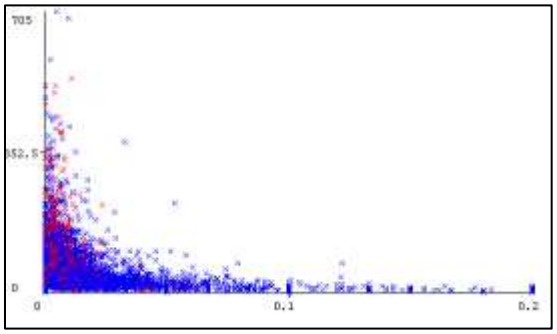
(g) Informational Duration vs Administrative Duration



(h) Page Values vs Product Related Duration



(i) Page Values vs Product Related



(j) Product Related vs Bounces Rates

Figure 2.2: Relationship between two attributes with Weka.

The graphs above show the relationship between each attribute in the data set. From the result shown that the data set is highly overlapped and having unbalanced data distribution as the unbalanced ratio is 1:6. Hence, this highly overlapped and unbalanced data set was selected to be applied in this project to further resolve the problem of the low true positive rate for the positive class.

2.5 Evaluation Methods

In this project, the six classification models will be applied and the classification model evaluation methods were used for evaluated the performance of the classification models.

The Cross-validation method can estimate the generalisation performance of a predictive model. In k-fold cross-validation, the data was sampled into k equal size subsets and each of the subsets will become either the training set or the validation set. All data will be used as the training and validation set and the performance of the predicting model will be evaluated from the average performance from the number of folds created for this cross-validation method. The 10-folds cross-validation is the most common use in machine learning, which splits the data into ten subsets and proceeds with the training and testing for ten iterations.

The holdout method separates the data into two sets, one for training and one for validation, based on a defined ratio, such as 80% to be the training set and 20% for the validation set. This method needs a short time to compute, but the evaluation has high variances, depending on how the data is being separated.

A confusion matrix evaluation method visualise the performance for the classification by separating the result into two groups, the positive and negative groups that include the true positive, true negative, false positive and false negative. A confusion matrix or error matrix is shown in Figure 4. The confusion matrix consists of the actual classification result and the predicted classification result. The true positive and true negative is the correct classification, while the other two are incorrect classification.

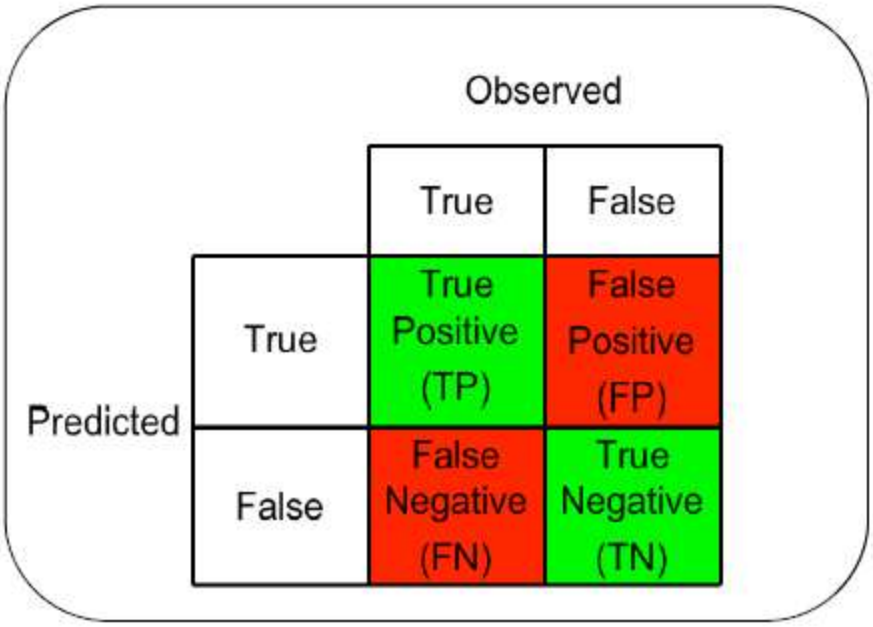


Figure 2.3: Format of a confusion matrix. (Kaur, 2013)

The formula for accuracy, precision and true positive rate (recall) are defined as below.

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

Precision $= \frac{TP}{TP+FP}$

True positive rate (Recall) $= \frac{TP}{TP+FN}$

False Positive Rate $= \frac{FP}{FP+TN}$

F1 Score $= 2 \times \frac{Precision \times Recall}{Presicion + Recall}$

Receiver Operating Curves (ROC) is a two-dimension graph that visually depicts the performance of the classification model. This is due to the confusion matrix may have poor summary when deploying a non-parametric model such as the neural networks or decision trees. Besides that, some of the performance derived using confusion matrix is sensitive to data anomalies such as unbalanced class distribution. ROC shows the same information as a confusion matrix but in a much more robust way. ROC curves were designed to determine the optimal operating points in the first place. There are two new metrics introduced in ROC, which are the true positive rate and the false positive rate.

ROC graphs are plotted by the true positive rate against the false positive rate. The number of regions of interest can be determined from the graph. Random performance is the middle line whereby the model produces the same true positive responses as the false positive response. A conservative performance is the regions with less false positive error. For liberal performance regions, a classifier model has a good true positive response. The bottom right region is the worst performing region as it will have a high false positive, which means a fail classification. Figure 5 shows the example of a ROC graph. (Kaur, 2013)

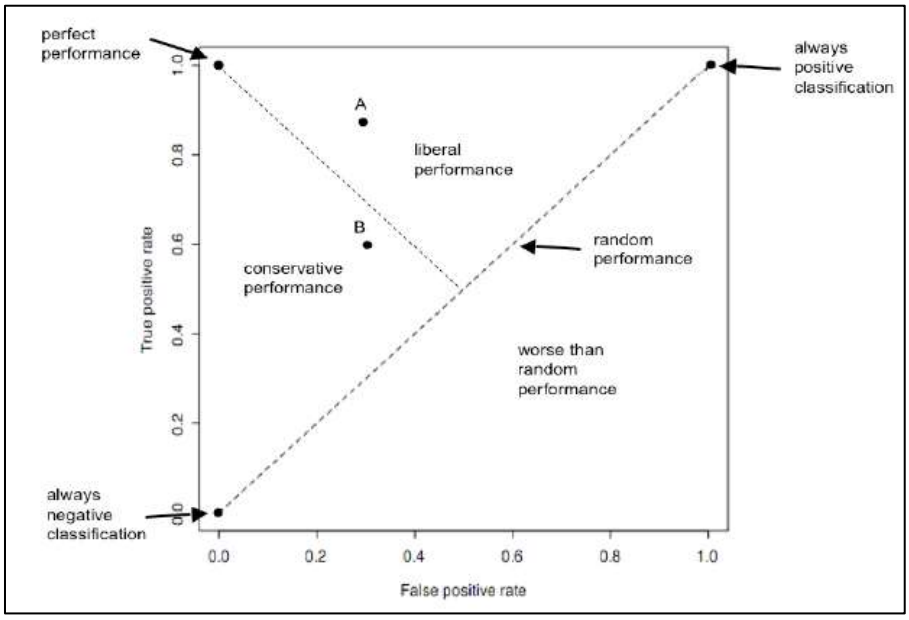


Figure 2.4: ROC graph regions. (Kaur, 2013)

From Kaur (2013) paper, the measurable performance metrics for evaluating a classification model include accuracy, precision, recall, F1 score, confusion matrix and AUC ROC. Accuracy is calculated based on the overall true positive and true negative for the model's predicting outcome. Thus, with the unbalanced data that skew to the negative class, the predicting result will tend to predict more negative class correctly and this may lead to high accuracy. However, the accuracy result is not favourable to the users as the most important is to predict the positive class correctly. The next evaluation metrics is precision. The rate of true positive among the positive result predicted from the system can be measured. But precision is not suitable for this predicting model with unbalanced data, as precision only measures the model's positive results but fails to look at

the false negative prediction by the model. In the case of predicting customers' buying decisions in the online shop, the false negative will cause loss to the users.

For recall or true positive rate, it measures the overall correct positive class prediction from the model. Recall performance is very important in my experimental study because correctly predicting positive class (Buy) is the priority in this work. Next, the F1 score is the combination of both precision and recall evaluating the performance. For Area under the curve (AUC), it is the probability of the system to choose the positive class than the negative class. The higher the value, the better the performance. The curve is the receiver operating curve (ROC) plotted by true positive against the false positive. With ROC, evaluation of results is clearer to be seen on the overall system performance as compared to accuracy or precision.

Therefore, in this work, the recall/true positive rate for the customers' buying decision predicting model will be emphasised. The favourable outcome is to obtain a more correct prediction of the 'Buy' class. This is because the loss from misclassifying the 'Not Buy' class is much lower than misclassifying the 'Buy' class that may potentially increase the sales in an online shop. Hence, the evaluation with recall is better than precision for this work.

CHAPTER 3

3. RESEARCH METHODOLOGY

3.1 Research Methodology

Based on this project objectives, better performance of online customers' buying decision predicting model needs to be achieved using the secondary data from Sakar *et al.* (2018). The authors achieved an overall accuracy of 87.94% and a true positive rate of 0.84 for the 'buy' class using the data set. Therefore, the data set was explored to achieve similar performance with Sakar *et al.* (2018) results by applying common machine learning algorithms

The knowledge discovery (KDD) process is the process that transfers the raw data set into information and knowledge. This project implemented the KDD process to convert the behaviours data set into a customer predicting model to allow online shop owners to prepare for strategies and decision making. There are five steps in the KDD process: data selection, data pre-processing, data transformation, data mining and result evaluation.

For the data selection part, this project used the online shoppers purchasing intention data set from Sakar *et al*. (2018).

The data set was then pre-processed to handle missing values and unwanted data. Since the data set in this project had complete data value and was in CSV format, the data set need not be transformed. During this stage, the data set was split into a train set and a test set (70% and 30%). The train set was pre-processed by sampling methods to balance the unbalanced class distribution of the 'buy' class and 'no buy' class in the data set.

The data mining step involves the classification process. In this stage, the data is analysed by a machine learning algorithm to find any relationship and pattern within the data set. Six algorithms were selected for this project. The algorithms are K-Nearest Neighbor (KNN), Naïve Bayers, J48, Support Vector Machine (SVM), Sequential Minimal Optimization (SMO) and Multilayer Perceptron (MLP).

Lastly, the data mining results are evaluated. In this project, the evaluation focuses on using true positive rate and accuracy so that it can be compared with the Sakar *et al.* (2018) results.

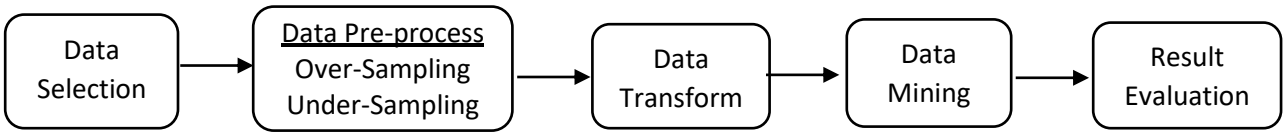


Figure 3.1: Flowchart of the Experiment.

To deal with the average performance of the learning algorithms caused by the unbalanced data set, data pre-processing method was applied as the solution. The data pre-processing techniques used were the sampling technique. Changes in the class distribution will make the data set balanced, and when input into each learning algorithm, its performance is expected to improve.

3.1.1 Data Pre-processing

In this project, the data set was split into 70% train set and 30% test set. The train set was pre-processed by sampling techniques, including over and under-sampling. The over-sampling was applied to the data set to increase the minority class size. The over-sampling technique applied was Synthetic Minority Over-sampling Technique (SMOTE). SMOTE uses the KNN method by selecting k number of nearest neighbours from sample data and joining them to generate synthetic samples in the data set. It takes the difference between the minority class sample and its nearest neighbours. Then it takes a random number between 0 and 1 to multiply with the difference to generate the synthetic samples. Since SMOTE is based only on minority class observation, it may cause the class boundary between the majority and minority class to become differentiable.

Under-sampling will also be applied by removing some of the data in the majority class. A specific percentage of the majority class data will be removed to balance the distribution in a data set. However, removing too much data may cause the predicting model to train with insufficient data and the performances to become weak.

Therefore, in this project, the pre-processing for the unbalanced data set shall be experimented by

- Over-sampling started from 10% to 150% for the minority class

- Under-sampling started from 10% to 80% for the majority class.

- A hybrid method with over-sampling started from 10% plus under-sampling started from 10%, and continuously to increase until over-sampling 100% and under-sampling 80%.

The results for each were recorded and visualised to compare and decide the best data pre-processing with selected machine learning algorithms.

3.1.2 Classification

In this research study, six machine learning algorithms are compared. The algorithms selected include K-Nearest Neighbor (KNN), Naïve Bayers, J48, Support Vector Machine (SVM), Sequential Minimal Optimization (SMO) and Multilayer Perceptron (MLP).

Besides the single learning algorithms, the ensemble learning method was also applied to further improve the classification performance. Ensemble learning combined several base classifiers in the same group to optimise the classification output while multiple classifiers combined several different classifiers to improve the classification output. The ensemble learning methods selected are AdaBoost and Bagging. Adaptive boosting (AdaBoost) is a boosting algorithm that combines weak learners and concludes the result based on the weight of each classifier so that the classification performances can be boosted. Boosting algorithm built the first model by randomly selecting data from supplied data set and test with other non-select data. During the process, those data classified wrongly by the first model will be collected and used to train the next model. The process continues several times to improve the classification performance and reduce the wrong classification result. The weight of misclassified samples is increased in each round and more samples were classified correctly. Bagging is also known as the bootstrap aggregating method. Like its name, bagging placed random and repeated data from the original train set into different "bags" to create multiple training subsets. Each subset will be used to generate multiple learning models with the based learning algorithm. The classification results obtained from each learning model will be voted to obtain the final decision. It improves the classification performance with multiple learning models to reduce the variance and enhance the steadiness of the supplied data set. (Taser, 2021)

The classification results were compiled and visualised into a table for better comparison of each algorithm capability to decide the suitable algorithms for the online shopping customers' buying decision predicting model.

3.1.3 Evaluation

The classification performance was evaluated with Accuracy, True Positive rate and True Negative rate to compare the performance of the algorithms based on the correct prediction for the positive and minor classes in the data set. The best result will then compare with the previous result from Sakar *et al.* (2018) based on accuracy and true positive rate.

However, in this project, the true positive rate was more focused than the accuracy rate as accuracy may not be appropriate for judging the performance for an unbalanced data set. Based on the formula of true positive rate, $\frac{TP}{TP+FN}$ takes the correct guess positive class (buy) and is divided by the total number of positive class data from the classification output. The higher the true positive rate, the better the prediction on the minority class. While accuracy is the overall prediction for both positive and negative classes in the classification output. In this project, the rate of getting a good prediction on the positive class (buy) is more important than getting a good prediction for the negative class (no buy). There is no point if the system predicts all the majority negative classes correctly and obtains high accuracy.

3.1.4 Project Plan

Table 3.1: Project Plan

TASK NAME	DURATION (DAYS)	START DATE (MM/DD/YY)	FINISH DATE (MM/DD/YY)
1. Initiation	**15 days**	**Mon 5/17/21**	**Mon 5/31/21**
1.1 Project Title Register	7 days	Mon 5/17/21	Sun 5/23/21
1.2 Obtain Data Set	4 days	Mon 5/24/21	Thu 5/27/21
1.3 Install WEKA Data Mining Tool	4 days	Fri 5/28/21	Mon 5/31/21
2. Planning	**49 days**	**Tue 6/1/21**	**Sat 7/17/21**
2.1 Project Problem Statement	7 days	Tue 6/1/21	Mon 6/7/21
2.2 Project Objective	7 days	Tue 6/8/21	Mon 6/14/21
2.3 Project Scope	7 days	Tue 6/15/21	Mon 6/21/21
2.4 Selection of Machine Learning Algorithms	14 days	Tue 6/22/21	Sun 7/4/21
2.5 Preliminary Report	14 days	Mon 7/5/21	Sat 7/17/21
3. Execution	**151 days**	**Sun 7/18/21**	**Sat 12/4/21**
3.1 Split Raw Data Set into Train Set and Test Set	4 days	Sun 7/18/21	Wed 7/21/21
3.2 Classification with selected algorithms (KNN, Naïve Bayes, J48, SVM, SMO and MLP)	7 days	Thu 7/22/21	Wed 7/28/21
3.3 Evaluate Result	7 days	Thu 7/29/21	Wed 8/4/21
3.4 Optimize Algorithms with CV Parameter Selection	14 days	Thu 8/5/21	Tue 8/17/21

3.5 Classification with Optimize Algorithms	7 days	Sat 8/14/21	Fri 8/20/21
3.6 Evaluate Result	7 days	Sat 8/21/21	Fri 8/27/21
3.7 Pre-process Train Set with SMOTE Over-Sampling Method	14 days	Sat 8/28/21	Thu 9/9/21
3.8 Classify the Pre-process Data Set	7 days	Fri 9/10/21	Thu 9/16/21
3.9 Pre-process Original Train Set with Under-Sampling Method	14 days	Fri 9/17/21	Wed 9/29/21
3.10 Classify the Pre-process Data Set	7 days	Thu 9/30/21	Wed 10/6/21
3.11 Pre-process Original Train Set with Combine Under-Sampling and Over-Sampling Method	28 days	Thu 10/7/21	Sun 10/31/21
3.12 Classify the Pre-process Data Set	7 days	Mon 11/1/21	Sun 11/7/21
3.13 Classify with ensemble learning method (AdaBoost and Bagging)	14 days	Mon 11/8/21	Sun 11/21/21
3.14 Evaluate Result	7 days	Mon 11/21/21	Sun 11/27/21
3.15 Make Conclusion	7 days	Sun 11/28/21	Sat 12/4/21
4. Monitoring	**14 days**	**Sun 12/5/21**	**Sat 12/18/21**
4.1 Project Limitation	7 days	Sun 12/5/21	Sat 12/11/21
4.2 Project Recommendation	7 days	Sun 12/12/21	Sat 12/18/21
5. Closing	**13 days**	**Sun 12/19/21**	**Fri 12/31/21**
5.1 Project Submission	7 days	Sun 12/19/21	Sat 12/25/21
5.2 Presentation	6 days	Sun 12/26/21	Fri 12/31/21

3.1.5 Project Gantt Chart

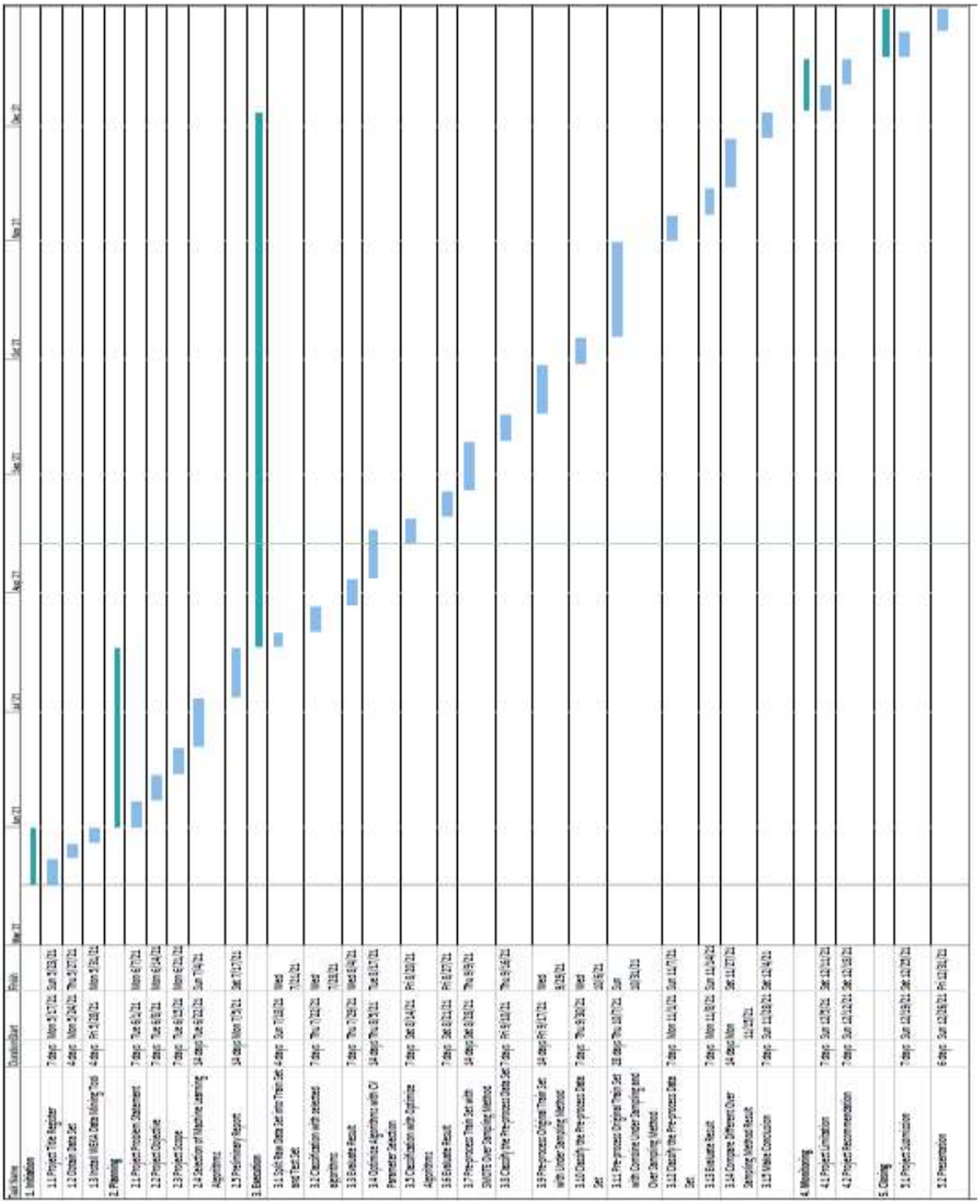


Figure 3.2: Project Gantt Chart.

CHAPTER 4

4. RESULT

4.1 Without Any Sampling

Firstly, the data set without sampling was classified by the selected learning algorithms in this project. The performance of each learning algorithm was evaluated using true positive rate (TPR), true negative rate (TNR) and accuracy. The results were compared with the best result from Sakar et al. (2018) as shown in the table below. The authors' results are highlighted in Table 4.1.

Table 4.1: Classification Results Obtained without Any Pre-Processing and Compared with Sakar *et al.* (2018)

Algorithm Approach	TPR/TNR (Buy / No Buy)	Accuracy	Pre-processing method
MLP (Sakar *et al.*, 2018)	84.0% / 92.0%	87.9%	Feature selection
KNN	36.3% / 89.5%	81.2%	-
Naïve Bayes	67.6% / 85.1%	82.4%	-
J48	58.7% / 95.2%	89.6%	-
LibSVM	99.5% / 0.3%	15.6%	-
SMO	35.6% / 98.2%	88.5%	-
MLP	52.1% / 95.0%	88.4%	-

As shown in Table 4.1, average classification rates were obtained using the selected learning algorithms, and they were uncompetitive with the past result obtained by Sakar et al. (2018). Even though satisfactory accuracy was obtained, the algorithms either performed weakly in the Buy class or the No Buy class. Hence, further experiments need to be conducted to achieve a result better than Sakar et al. (2018).

From the perspective of true positive rate and accuracy, the Naïve Bayes and J48 algorithms gave more balanced classification results in both the Buy and No Buy classes. Therefore, these two

learning algorithms were selected for the later experiments. Before pre-processing the data set in the next phase of experiments, the learning algorithms' parameters of J48 were fine-tuned with CV parameter selection. The confidence factor (c) and minimum object per leaf (m) for J48 had been set to 0.05 and 2, respectively. The small value of the confidence factor means more branches for the classification model built. The fine-tuned parameters are as shown in Table 4.2. The Naïve Bayes did not have any parameters to adjust.

Table 4.2: The fine-tuned Parameters of J48

Algorithm Approach	Parameter	TPR/TNR (Buy / No Buy)	Accuracy
J48	c: 0.25 m:2	58.7% / 95.2%	89.6%
J48	c: 0.05 m:2	61.5% / 95.1%	89.9%

4.2 With Over-Sampling (SMOTE)

Next, data pre-processing with the over-sampling method - SMOTE was applied to the minority class (Buy) of the train data set and the model built was evaluated using the test set. The classification results are detailed tabulated in the Appendix. The results were summarised and plotted in the figures below.

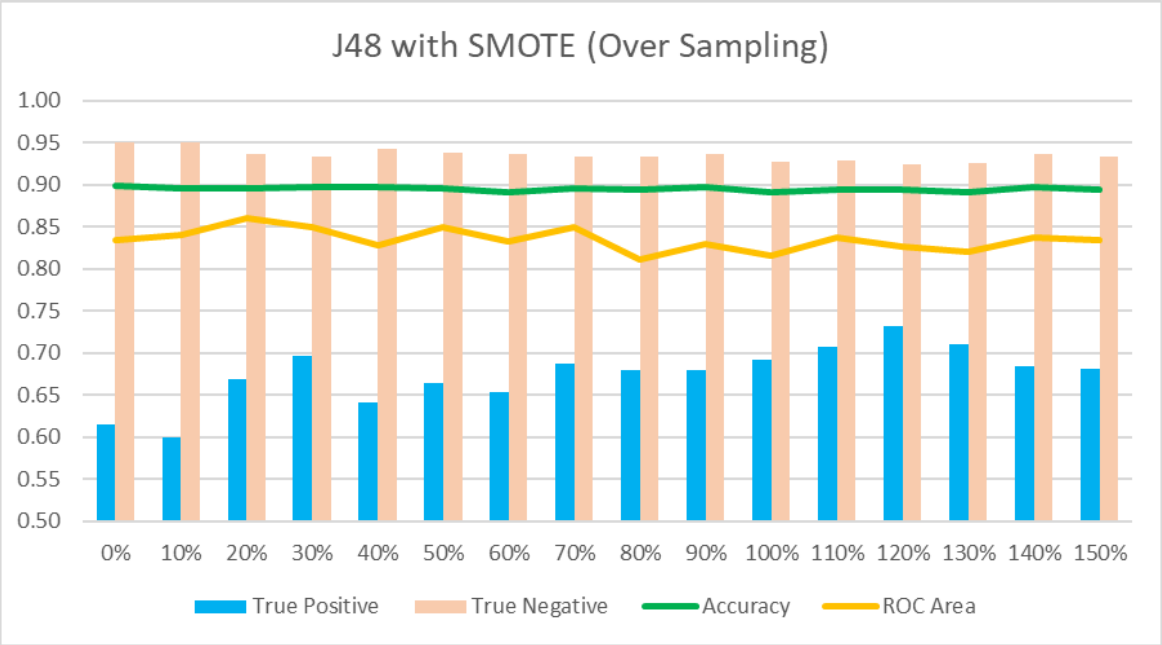


Figure 4.1: The classification results obtained using J48 after applying SMOTE.

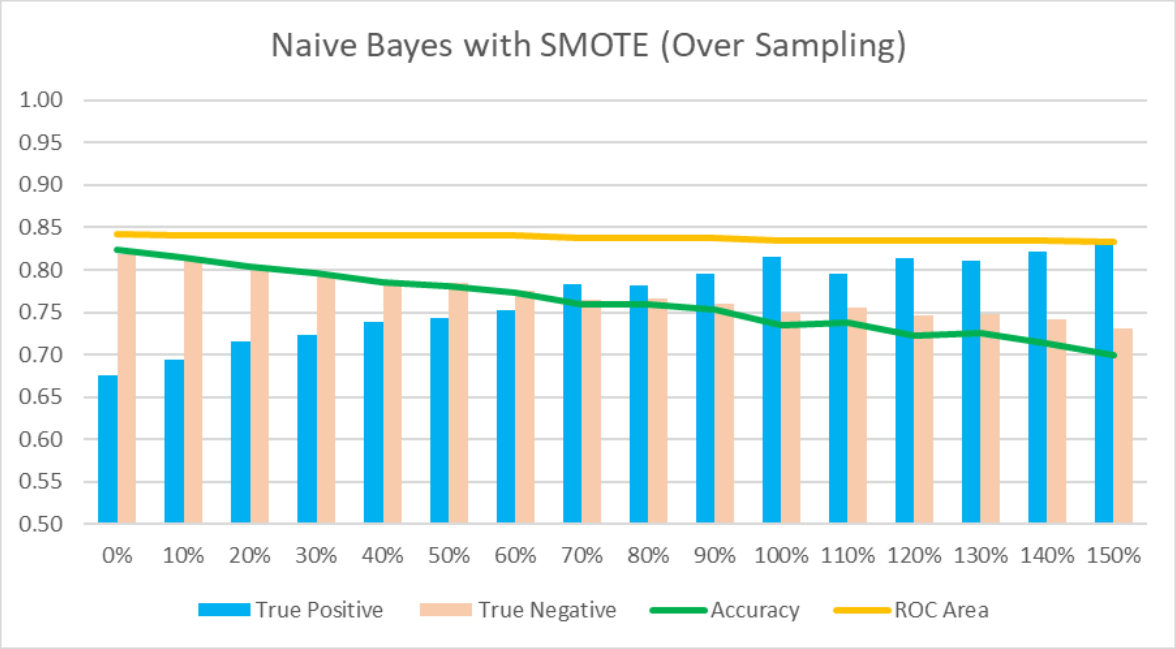


Figure 4.2: The classification results obtained using Naïve Bayer after applying SMOTE.

As shown in Figure 4.1 and Figure 4.2, the TPR of both classification models had improved with SMOTE (over-sampling). With J48, the max TPR achieved with SMOTE was 73.2% (120% over-sampling) and the result stagnated after 120% over-sampling on the train set. For Naïve Bayes, the TPR increased continuously when increasing the over-sampling percentage but the TNR and accuracy drop accordingly. Therefore, the overall performance of Naïve Bayes was unsatisfactory with the application of SMOTE.

4.3 With Under-Sampling

Besides, under-sampling was done with the SpreadSubsample of Weka to reduce the size of the majority class (No Buy). The classification results are detailed tabulated in the Appendix. The results were summarised and plotted in the figures below.

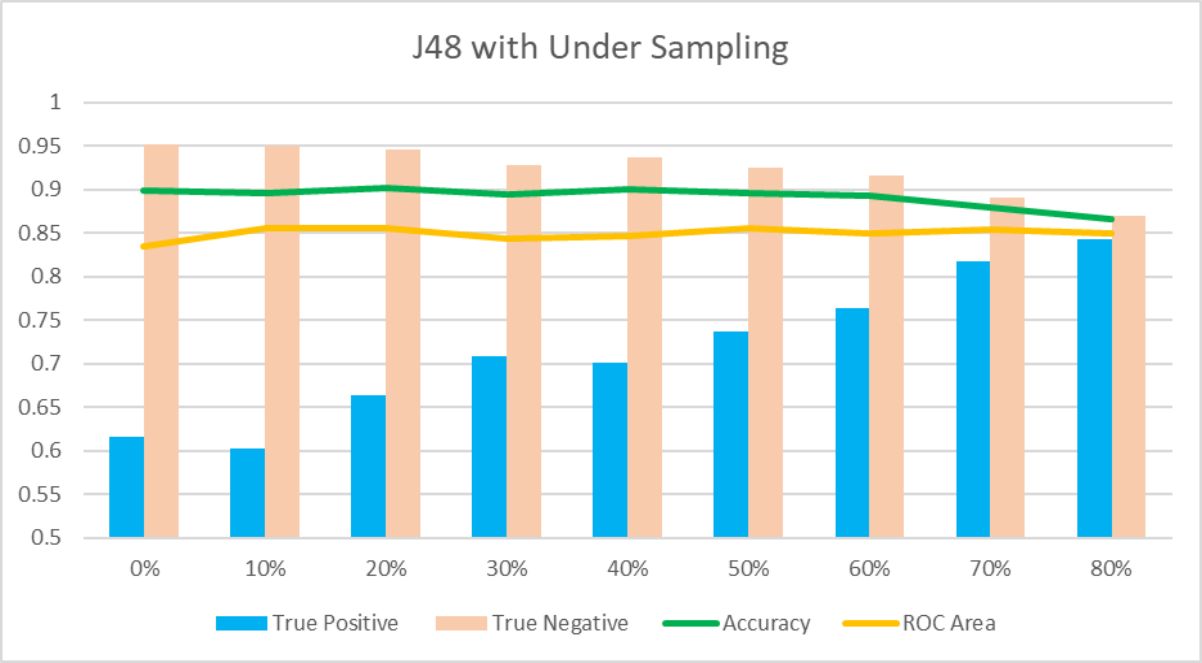


Figure 4.3: The classification results obtained using J48 after applying under-sampling.

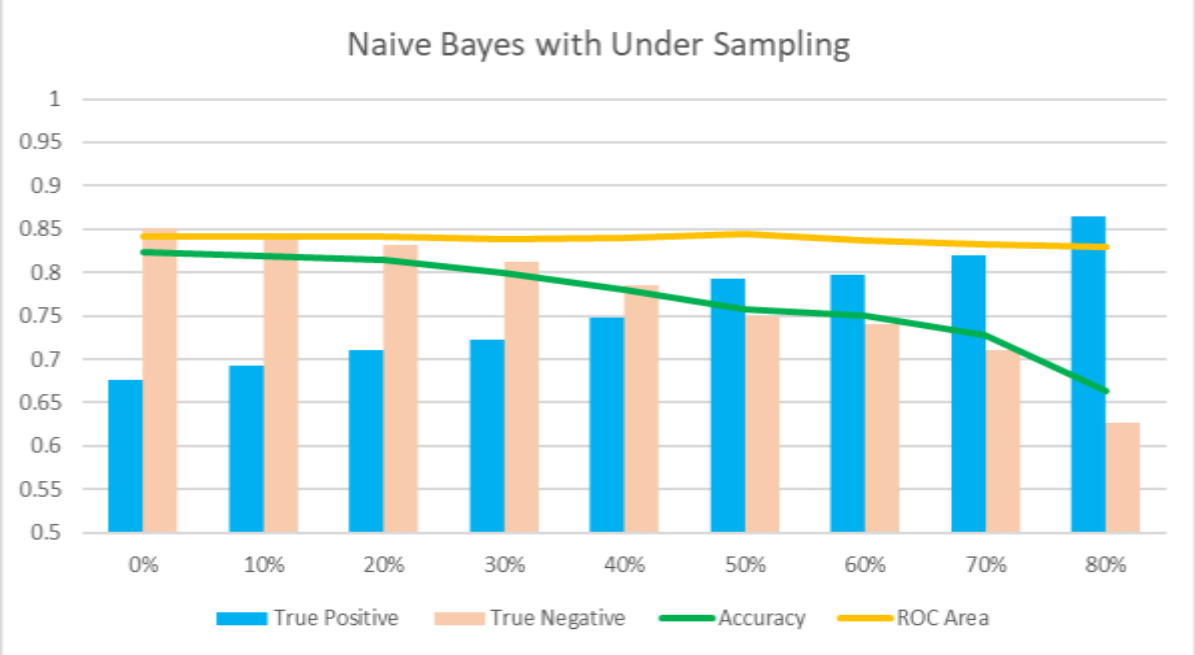


Figure 4.4: The classification results obtained using Naïve Bayes after applying under-sampling.

As shown in Figure 4.3, the under-sampling method achieved a better result than over-sampling with J48 by under sampling 80% of the majority class (No Buy). The TPR was 84.2%, TNR was 87.0% and the accuracy was 86.6%. From Figure 4.4, Naïve Bayes with under-sampling suffered from the same problem as in SMOTE.

4.4 With Hybrid Sampling

For the hybrid sampling method obtained the results as below.

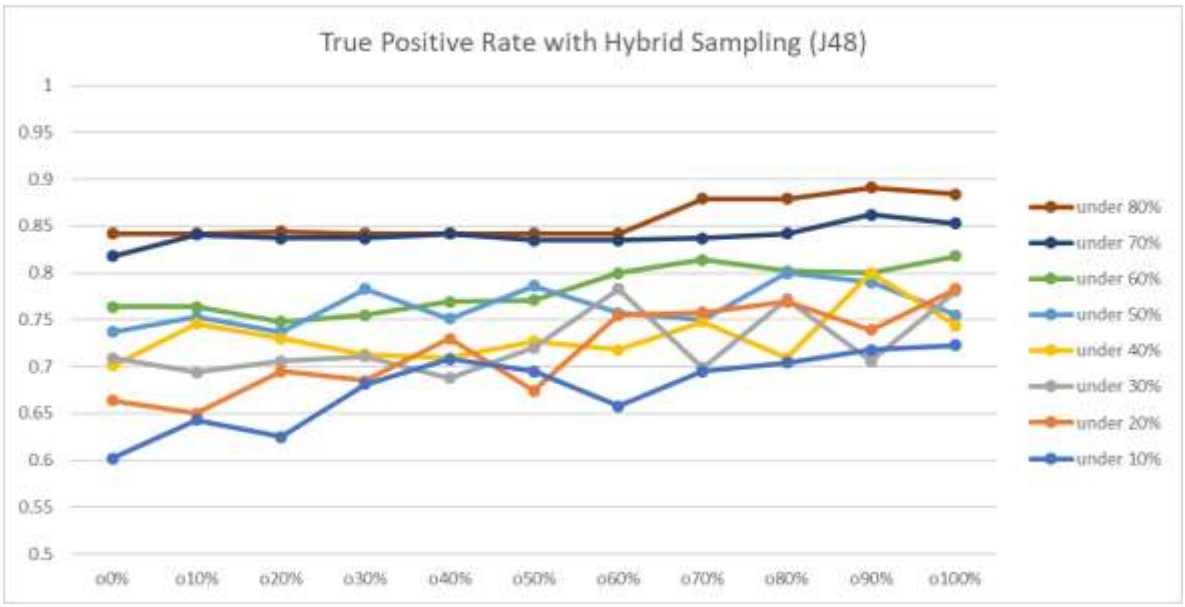


Figure 4.5: The TPR obtained using J48 after applying hybrid sampling.

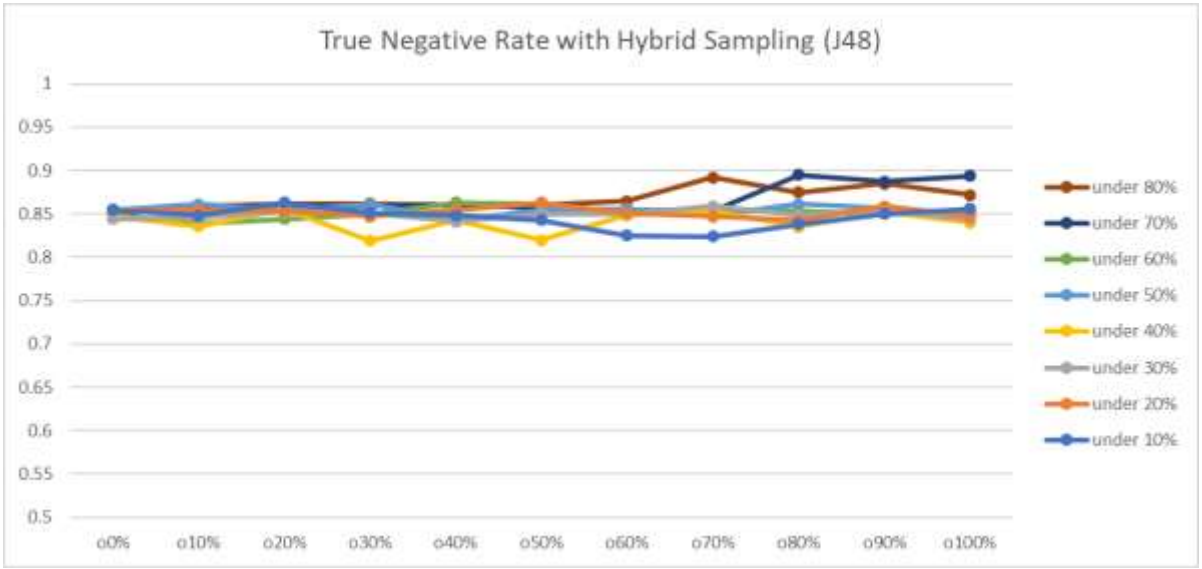


Figure 4.6: The TNR obtained using J48 after applying hybrid sampling.

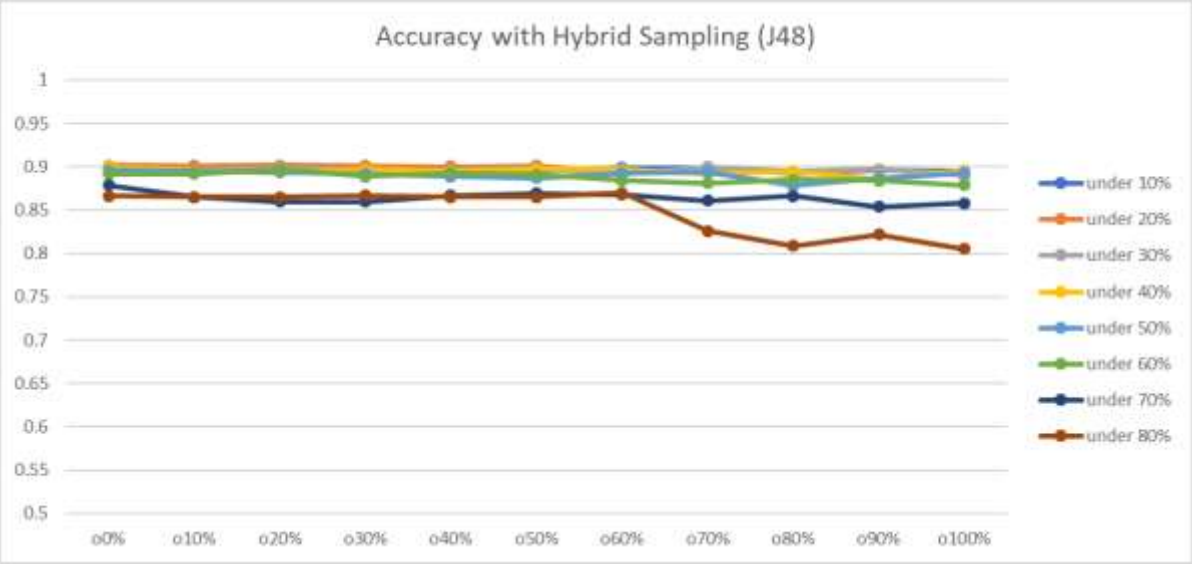


Figure 4.7: The accuracy obtained using J48 after applying hybrid sampling.

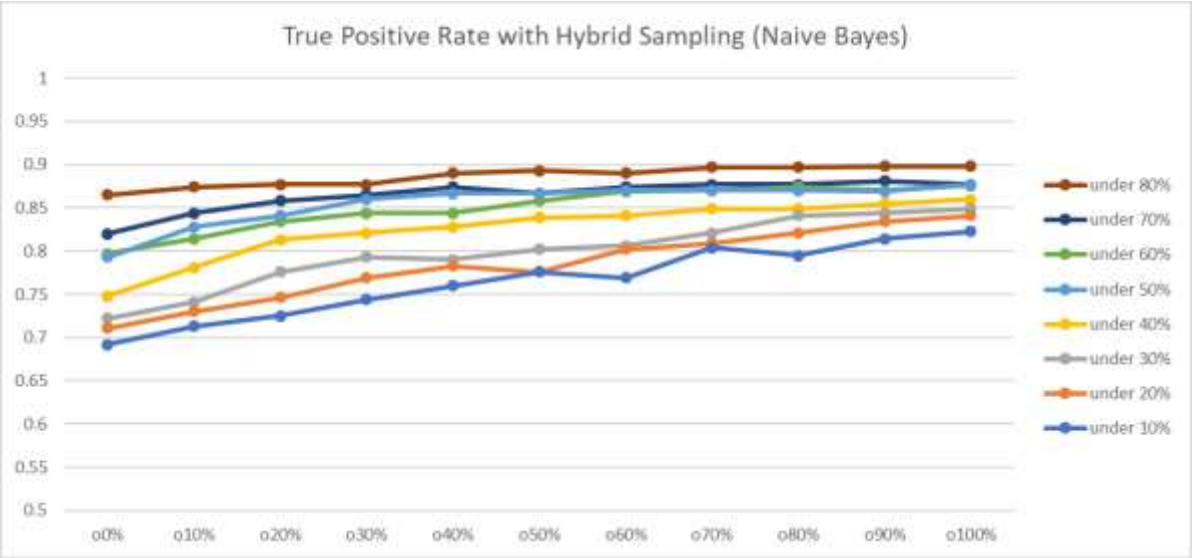


Figure 4.8: The TPR obtained using NB after applying hybrid sampling.

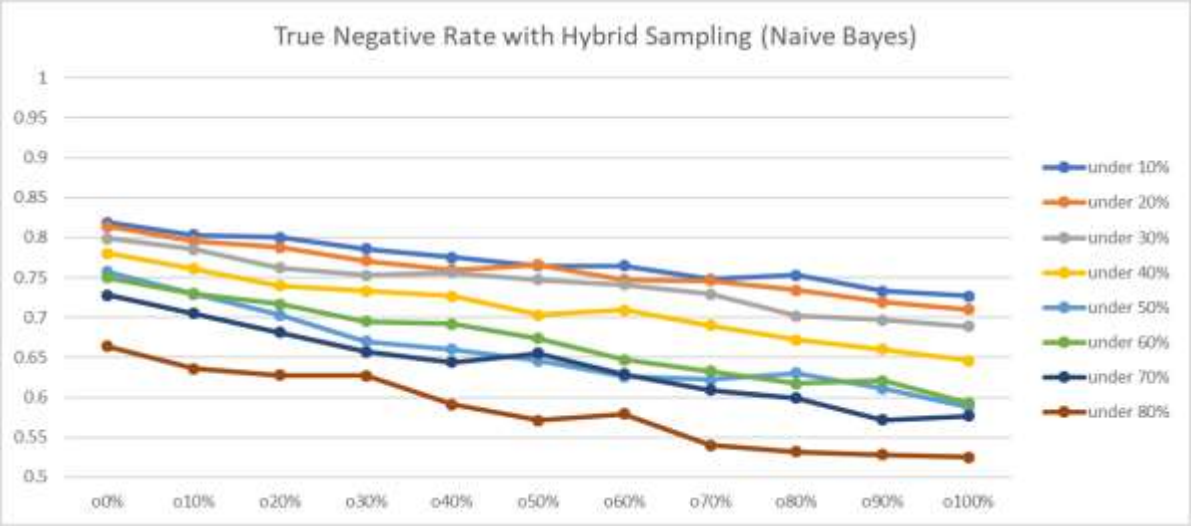


Figure 4.9: The TNR obtained using NB after applying hybrid sampling.

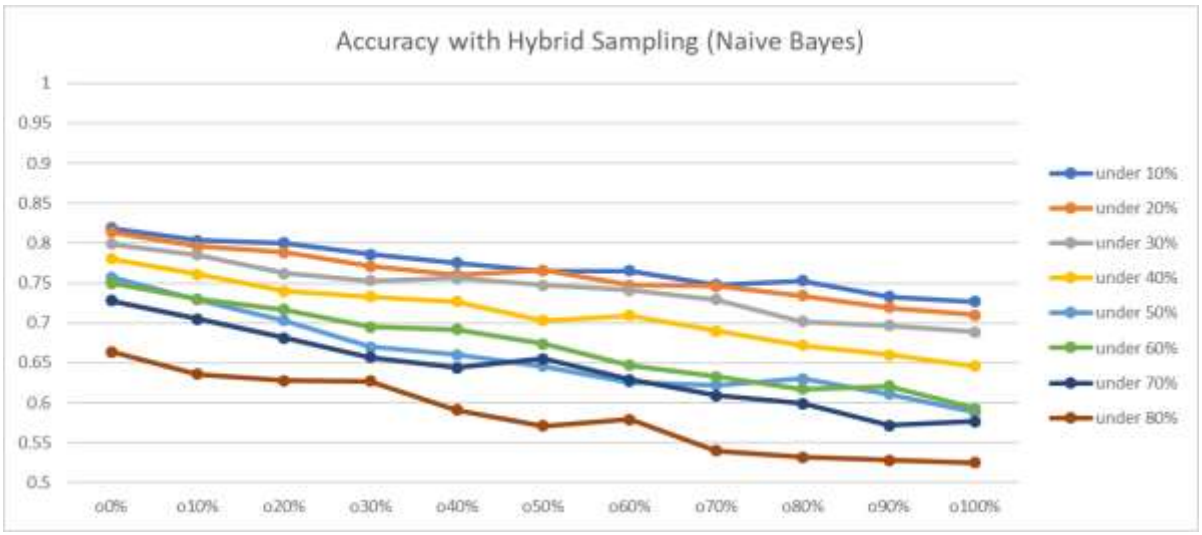


Figure 4.10: The accuracy obtained using NB after applying hybrid sampling.

As shown in the graph from Figure 4.5 to Figure 4.7, the overall performance with J48 using hybrid sampling had slightly improved compared with just applying under-sampling. The best result was obtained with the hybrid method (60% oversampling + 80% under sampling) among the three data pre-processing method. The TPR was 84.2%, TNR was 87.5% and accuracy was 87.0%.

4.5 Ensemble Learning Method

Two ensemble learning methods, AdaBoost and Bagging were implemented. The base learning algorithms applied were J48 and Naïve Bayes. Classification results are shown below.

Table 4.3: Result Obtained with Ensemble Method AdaBoost

Ensemble Learning	Algorithm Approach	TPR/TNR (Buy / No Buy)	Accuracy
AdaBoost	J48	58.5% / 94.3%	88.8%
AdaBoost	Naïve Bayes	50.6% / 94.2%	87.5%

Table 4.4: Result Obtained with Ensemble Method Bagging

Ensemble Learning	Algorithm Approach	TPR/TNR (Buy / No Buy)	Accuracy
Bagging	J48	59.9% / 95.4%	89.9%
Bagging	Naïve Bayes	68.0% / 85.0%	82.4%

From the result above, average classification performance was obtained using AdaBoost and Bagging. Since there is no significant improvement with the ensemble methods applied, further experiments were conducted by combining ensemble learning with the sampling method.

4.5.1 Ensemble Learning with Over Sampling

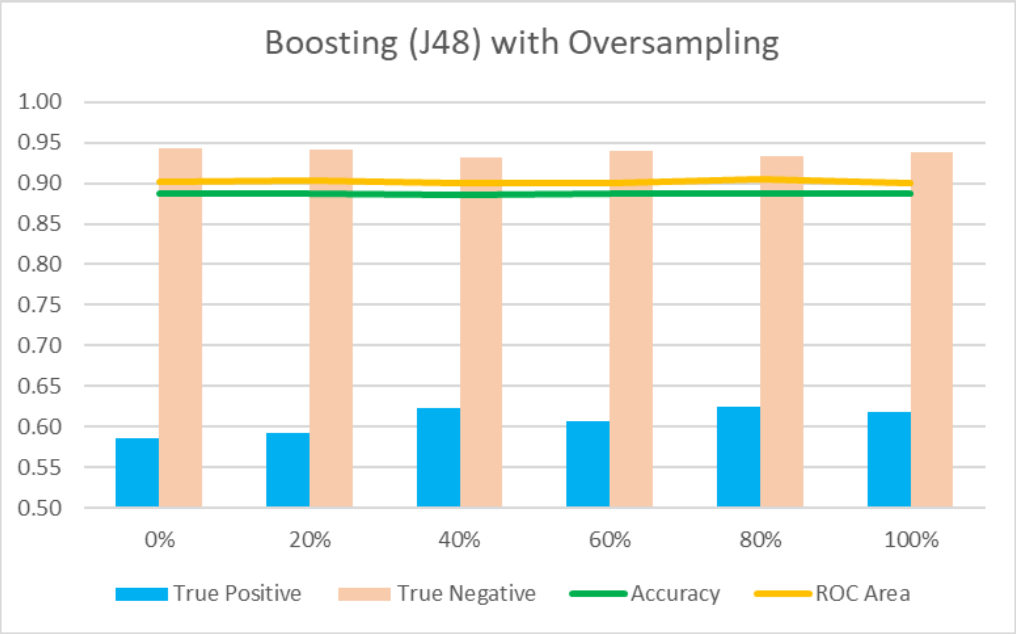


Figure 4.11: The classification results obtained using AdaBoost (J48) after applying SMOTE

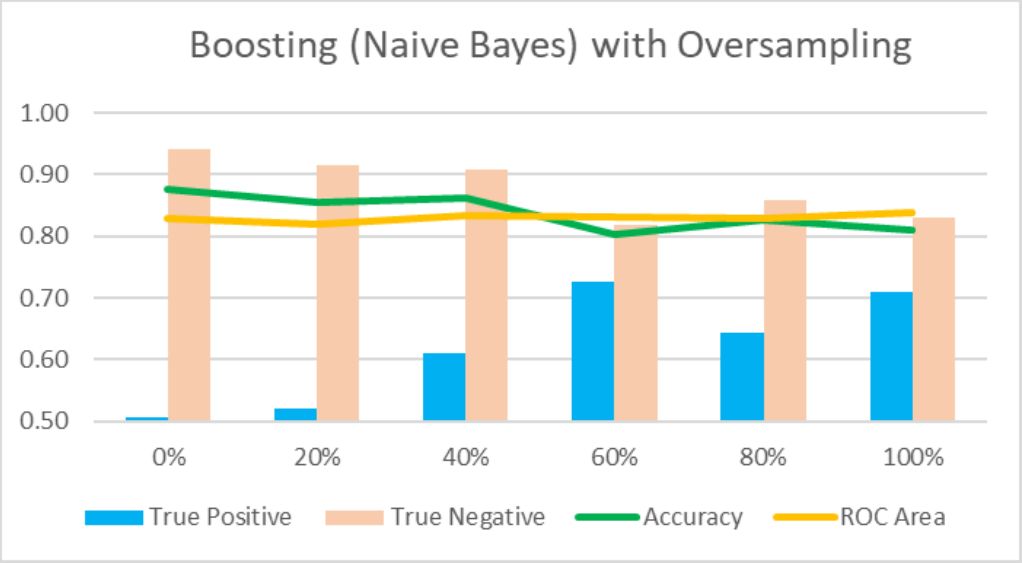


Figure 4.12: The classification results obtained using AdaBoost (Naïve Bayes) after applying SMOTE

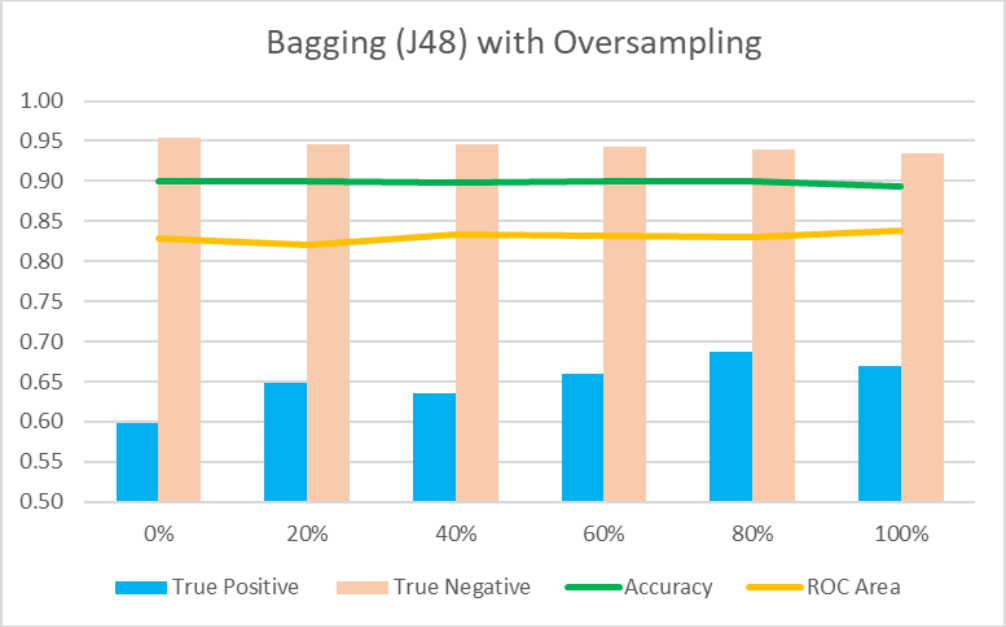


Figure 4.13: The classification results obtained using Bagging (J48) after applying SMOTE

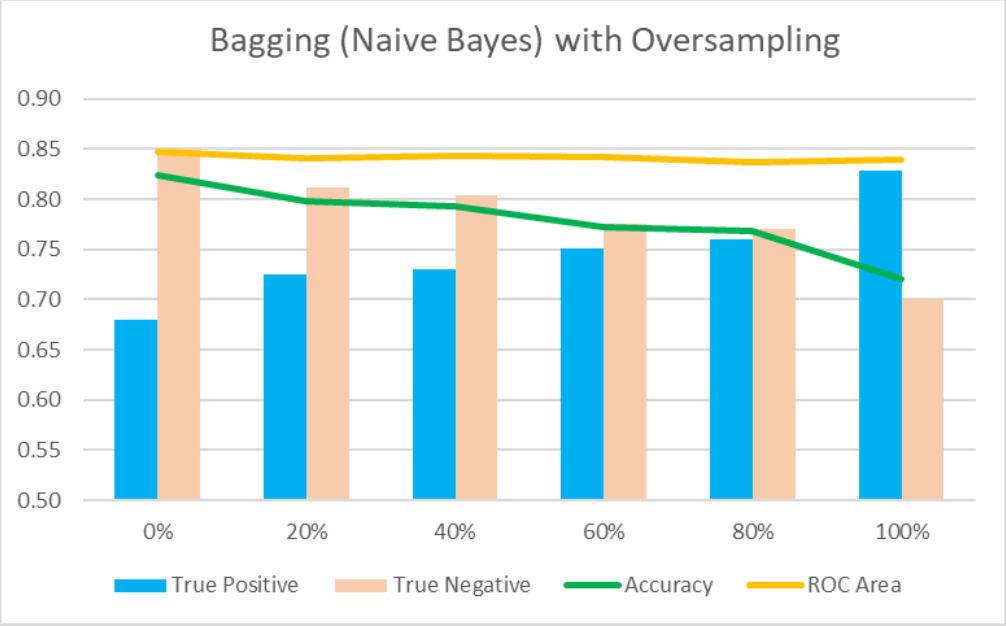


Figure 4.14: The classification results obtained using Bagging (Naïve Bayes) after applying SMOTE

For the ensemble method with over-sampling, the best was bagging with J48 by 80% over-sampling. The TPR was 68.7%, TNR was 93.9% and accuracy was 90.0%.

4.5.2 Ensemble Learning with Under Sampling

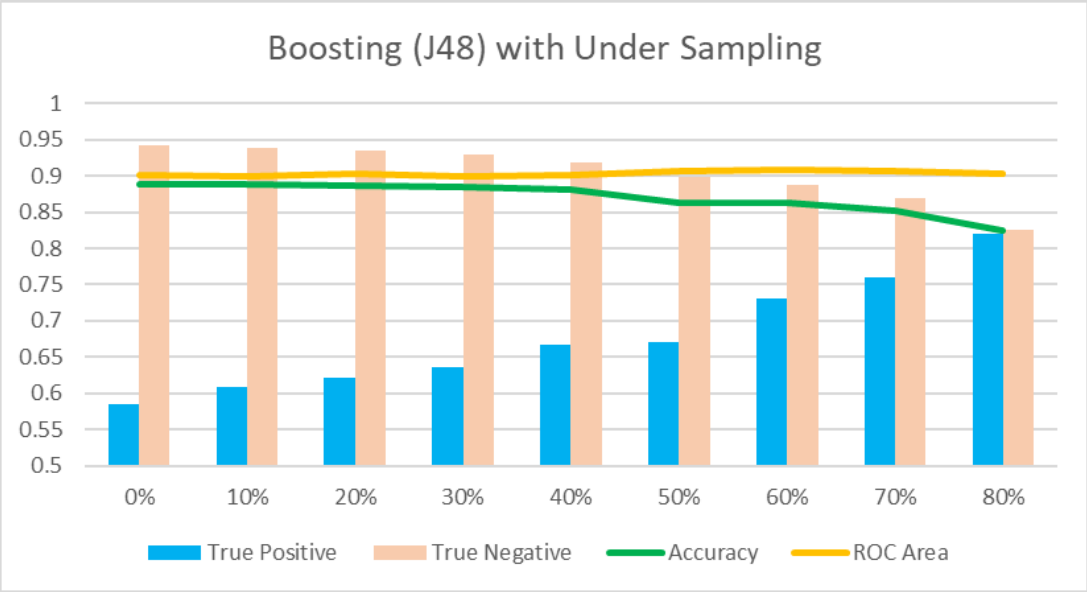


Figure 4.15: The classification results obtained using AdaBoost (J48) after applying under-sampling.

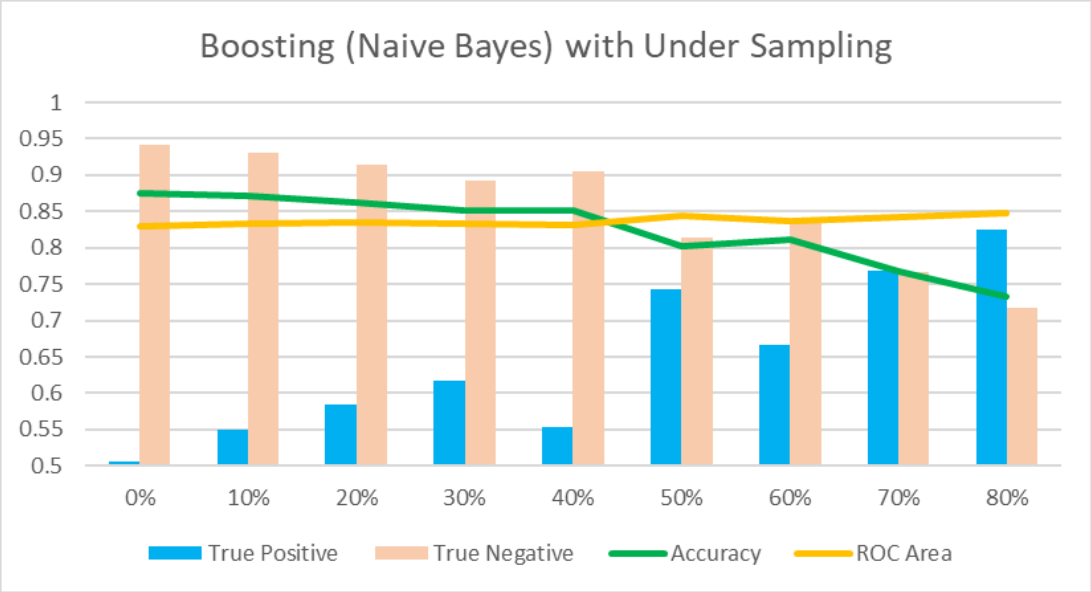


Figure 4.16: The classification results were obtained using AdaBoost (Naïve Bayes) after applying under-sampling.

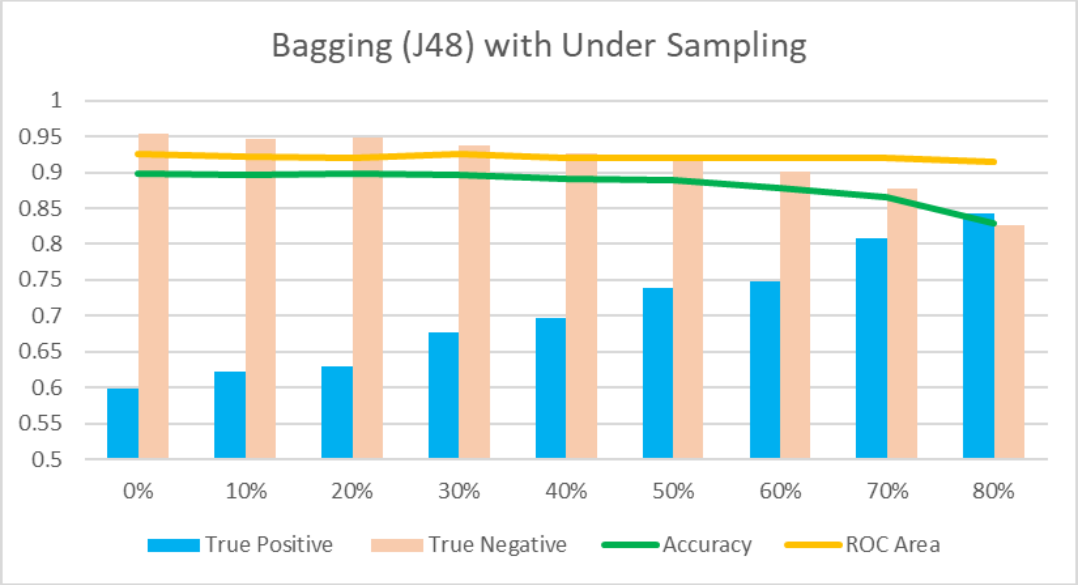


Figure 4.17: The classification results obtained using Bagging (J48) after applying under-sampling.

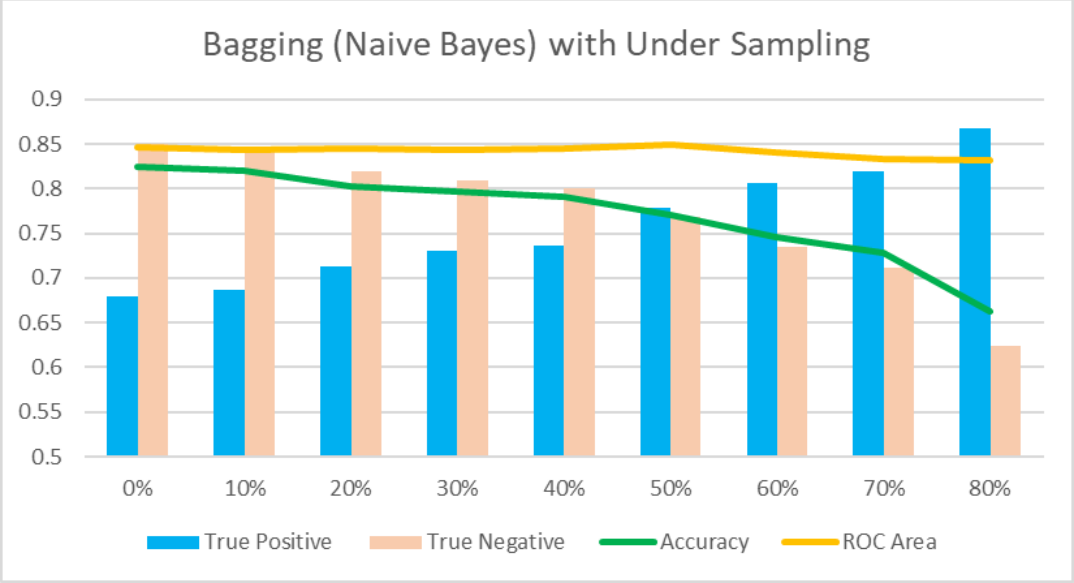


Figure 4.18: The classification results obtained using Bagging (Naïve Bayes) after applying under-sampling.

For the ensemble method with under-sampling, the best was bagging with J48 by 70% under-sampling. The TPR was 80.9%, TNR was 87.7% and accuracy was 86.6%.

4.5.3 Ensemble Learning with Hybrid Sampling

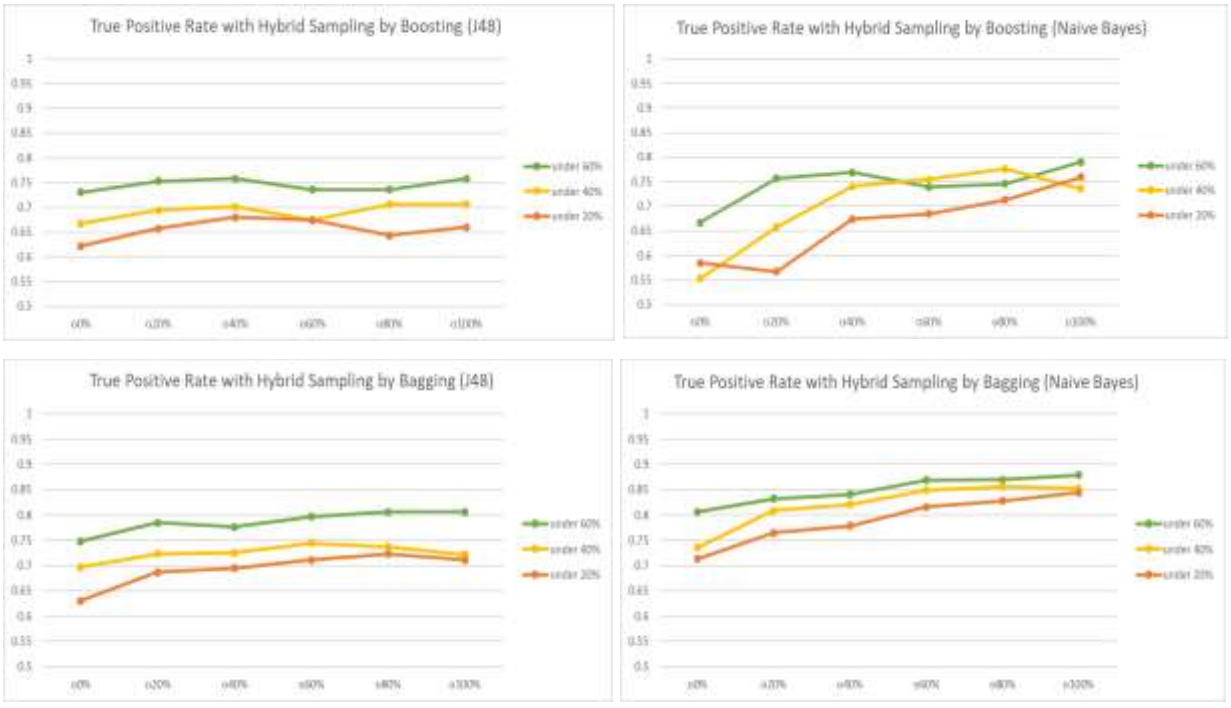


Figure 4.19: The TPR obtained using the ensemble method after applying hybrid sampling.

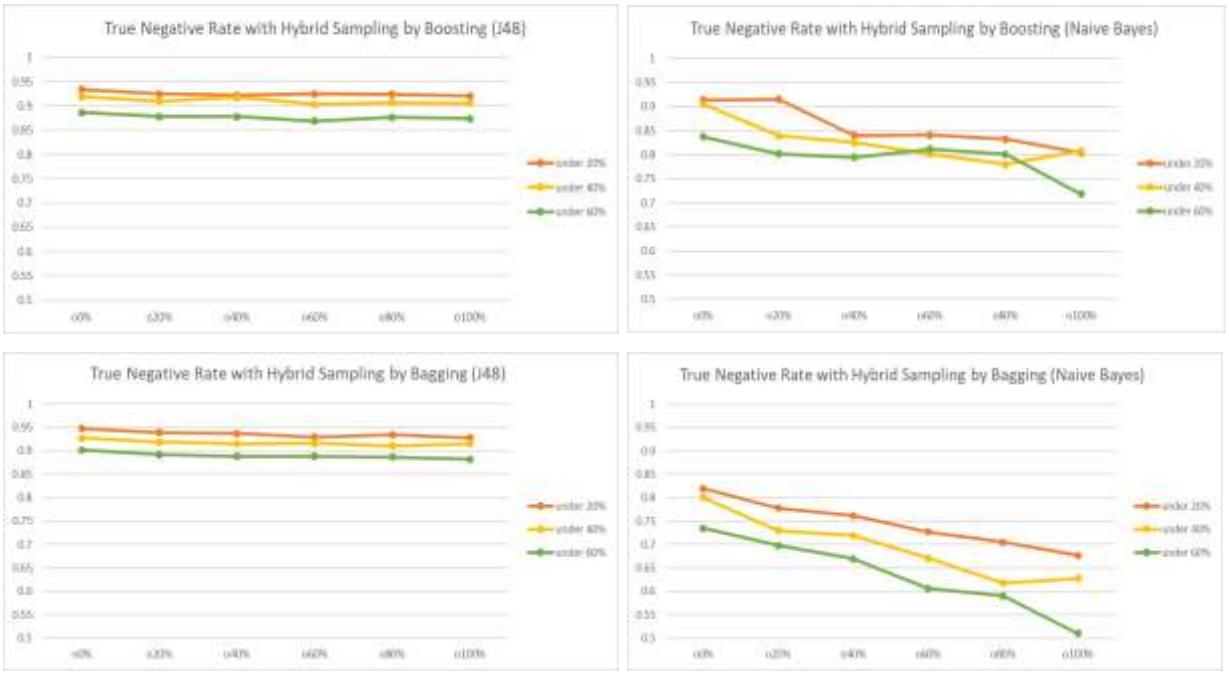


Figure 4.20: The TNR obtained using the ensemble method after applying hybrid sampling.

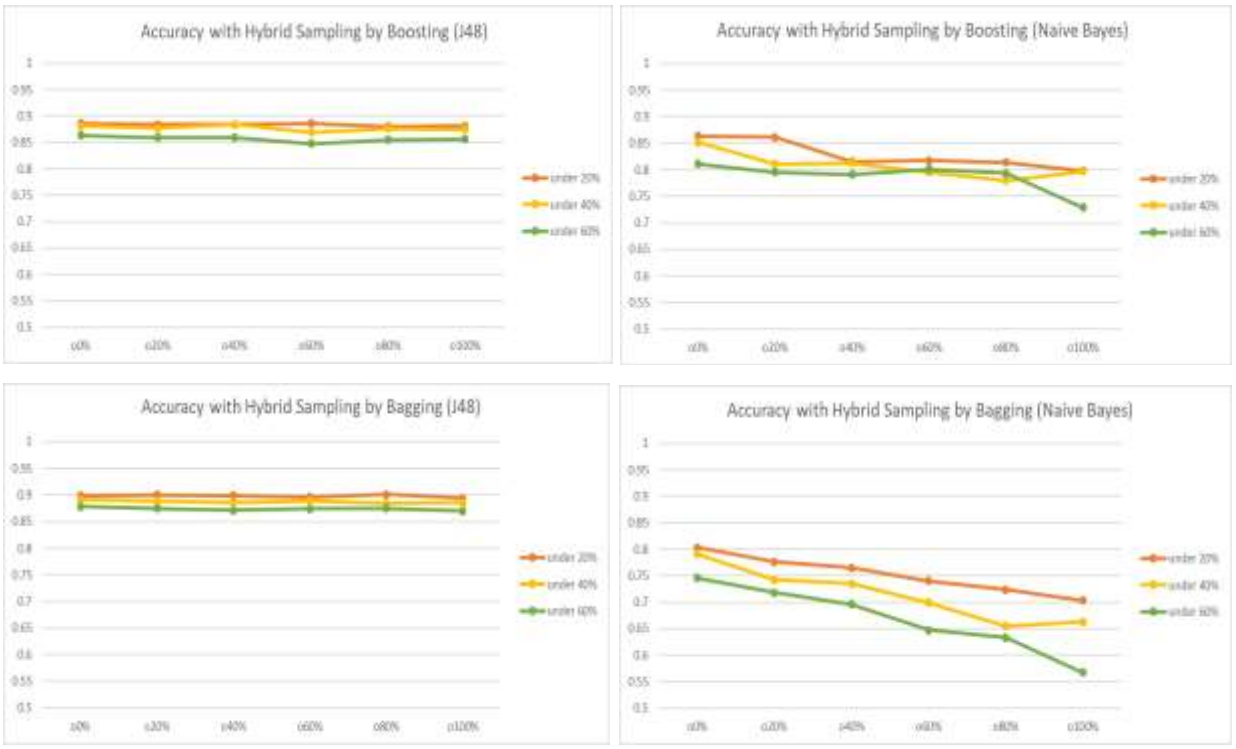


Figure 4.21: The Accuracy obtained using the ensemble method after applying hybrid sampling.

For the ensemble method with hybrid sampling, the best was bagging with J48 by 60% under-sampling and 80% over-sampling. The TPR was 80.6%, TNR was 88.7% and accuracy was 87.5%.

4.6 Comparing Results

Table 4.5: Summary Result Between Different Sampling Method and Sakar *et al.* (2018)

Pre-processing method	Algorithm approach	TPR/TNR (Buy / No Buy)	Accuracy
Sakar *et al*. (2018)	MLP	84.0% / 92.0%	87.9%
SMOTE (120%)	J48	73.2% / 92.5%	89.5%
Under Sampling (80%)	J48	84.2% / 87.0%	86.6%
Hybrid Sampling (SMOTE 60% + Under Sampling 80%)	J48	84.2% / 87.5%	87.0%
SMOTE (0%)	Bagging (J48)	68.7% / 93.9%	90.0%
Under Sampling (70%)	Bagging (J48)	80.9% / 87.7%	86.6%
Hybrid Sampling (SMOTE 80% + Under Sampling 60%)	Bagging (J48)	80.6% / 88.7%	87.5%

Based on the table above, the best result obtained from the sampling method was hybrid with 60% over-sampling (SMOTE) plus 80% under-sampling; the learning algorithm applied is J48. By comparing Sakar's best result, the TPR was 84.2% from the hybrid method and 84.0% from Sakar et al. (2018). TNR was 87.5% from the hybrid method and 92.0% from Sakar *et al.* (2018). Accuracy was 87.0% from the hybrid method and 97.9% from Sakar *et al.* (2018). The result showed that the hybrid sampling with J48 can achieve comparable performance as Sakar *et al.* (2018). As shown in Table 4.6, there was no improvement achieved using the ensemble learning method.

CHAPTER 5

5. CONCLUSION

In this project, the data set provided by Sakar et al. (2018) was used. The data set consists of 15.5% of the positive class (Buy) and 84.5% of the negative class (No Buy). This project is to modify the unbalanced class distribution data to obtain better predicting results. Weka - a tool for data mining, analysis and visualisation was used. The results obtained with different unbalanced data set yielded by the sampling methods and different machine learning algorithms were visualised and compared with the tool.

The data set was split into a train set (70%) and test set (30%) and models were built using the unsampled data set, sampled data sets and ensemble learning. The best result was obtained by the hybrid sampling method with 60% over-sampling + 80% under-sampling and applied with single learning algorithms J48. The TPR, TNR and accuracy were 84.2%, 87.5% and 87.0%, respectively. The result is comparable to Sakar *et al.* (2018). However, the learning algorithm J48 is faster than Multilayer Perceptron used in Sakar *et al.* (2018).

The best model built in this project only achieved comparable performance with Sakar *et al.* (2018). The ensemble learning AdaBoost and Bagging, with base learner Naïve Bayes and J48 showed no improvement. Therefore, to obtain a classification result better than Sakar et al. (2018), the other ensemble learning methods, such as voting and stacking, will be considered for future experiments.

REFERENCES

ACI Worldwide, 2020, *Global eCommerce Retail Sales Up 209 Percent in April, ACI Worldwide Research Reveals* [Online]. Available at: https://www.aciworldwide.com/news-and-events/press-releases/2020/may/global-ecommerce-retail-sales-up-209-percent-in-april-aci-worldwide-research-reveals [Accessed: 20 August 2020].

Ahmed, M., Siddiqi, I., Afzal, H. and Khan, B., 2018. MCS: Multiple classifier system to predict the churners in the telecom industry. *2017 Intelligent Systems Conference, IntelliSys 2017*, 2018-Janua(September), pp.678–683.

Albert, B.T.C. and Hartford, W., 2004. R Esearch a Rticle Gist : a M Odel for D Esign and M Anagement of C Ontent and I Nteractivity of C Ustomer -C Entric W Eb S Ites 1. *MIS Quarterly*, 28(2), pp.161–182.

Arafat, M.Y., Hoque, S., Xu, S. and Farid, D.M., 2019. An under-sampling method with support vectors in multi-class imbalanced data classification. *2019 13th International Conference on Software, Knowledge, Information Management and Applications, SKIMA 2019*, (August).

C, S. and Ravikumar, P., 2019. Community Mining for Predicting the Purchasing. , pp.287–292.

Chaffey, D., 2020, *E-commerce conversion rates 2020 compilation - How do yours compare?* [Online]. Available at: https://www.smartinsights.com/ecommerce/ecommerce-analytics/ecommerce-conversion-rates/ [Accessed: 11 October 2020].

Choirunnisa, S. and Lianto, J., 2017. for Handling Imbalanced Data. *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pp.276–280.

Cronin-Gilmore, J., 2012. Exploring Marketing Strategies in Small Businesses. *Journal of Marketing Development and Competitiveness*, 6(1), pp.96–107.

Dang, V.T., Wang, J. and Vu, T.T., 2020. An integrated model of the younger generation's online shopping behavior based on empirical evidence gathered from an emerging economy. *PLoS ONE*, 15(5), pp.1–20.

Dou, X., 2020. Online Purchase Behavior Prediction and Analysis Using Ensemble Learning. *2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics, ICCCBDA 2020*, pp.532–536.

Di Fatta, D., Patton, D. and Viglia, G., 2018. The determinants of conversion rates in SME e-commerce websites. *Journal of Retailing and Consumer Services*, 41(December 2017), pp.161–168. Available at: https://doi.org/10.1016/j.jretconser.2017.12.008.

Ganganwar, V., 2012. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4), pp.42–47. Available at: http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf.

Grandón, E.E., Nasco, S.A. and Mykytyn, P.P., 2011. Comparing theories to explain e-commerce adoption. *Journal of Business Research*, 64(3), pp.292–298. Available at: http://dx.doi.org/10.1016/j.jbusres.2009.11.015.

Hu, X., Yang, Y., Chen, L. and Zhu, S., 2020. Research on a Prediction Model of Online Shopping Behavior Based on Deep Forest Algorithm. *2020 3rd International Conference on Artificial Intelligence and Big Data, ICAIBD 2020*, pp.137–141.

Jacobusse, G. and Veenman, C.J., 2016. On Selection Bias with Imbalanced Classes. , (October 2019).

Jiang, M.H. and Hu, J.H., 2014. Combining multiple classifiers based on Dempster-Shafer theory for personal credit scoring. *International Conference on Management Science and Engineering - Annual Conference Proceedings*, 1(1), pp.167–172.

Kaur, K., 2013. Evaluation Measures for Data Mining Tasks. Available at: http://iasri.res.in/ebook/win_school_aa/notes/Evaluation_Measures.pdf.

Kennedy, A. and Coughlan, J., 2006. Online shopping portals: An option for traditional retailers? *International Journal of Retail & Distribution Management*, 34(7), pp.516–528.

Krawczyk, B., 2016. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), pp.221–232.

Lee, H.K. and Kim, S.B., 2018. An overlap-sensitive margin classifier for imbalanced and overlapping data. *Expert Systems with Applications*, 98, pp.72–83. Available at: https://doi.org/10.1016/j.eswa.2018.01.008.

Li, X. and Zhou, Q., 2019. Research on improving SMOTE algorithms for unbalanced data set classification. *Proceedings - 2019 International Conference on Electronic Engineering and Informatics, EEI 2019*, pp.476–480.

Malhotra, R. and Jain, J., 2020. Handling imbalanced data using ensemble learning in software defect prediction. *Proceedings of the Confluence 2020 - 10th International Conference on Cloud Computing, Data Science and Engineering*, pp.300–304.

McLean, L.B. and Weaver, A.C., 2018. Classification of imbalanced data in E-commerce. *2017 Intelligent Systems Conference, IntelliSys 2017*, 2018-Janua(September), pp.744–750.

Mînăstireanu, E.-A. and Meșniță, G., 2020. Methods of Handling Unbalanced Datasets in Credit Card Fraud Detection. *Brain. Broad Research in Artificial Intelligence and Neuroscience*, 11(1), pp.131–143.

Moe, W.W., 2003. Buying , Searching , or Browsing : Differentiating Between Online Shoppers Using In-Store Navigational Clickstream. , 13(2000), pp.29–39.

Mohammed, M., Yadwad, S. and Kassie, A., 2018. Data Mining Application in Prediction of potential Customers of POS Machine Users in Fund Transaction. 2018 IEEE, pp. 115–120.

Nayyar, T., 2019. Analyzing Customer Buying Behavior Creative Component Project Report By : Master of Science in Information Systems Major Professor : Ivy College of Business Iowa State University.

Rajamma, R.K., Paswan, A.K. and Hossain, M.M., 2009. Why do shoppers abandon shopping cart? Perceived waiting time, risk, and transaction inconvenience. *Journal of Product and Brand Management*, 18(3), pp.188–197.

Rusmee, K. and Chumuang, N., 2019. Predicting System for the Behavior of Consumer Buying Personal Car Decision by Using SMO. *Proceedings - 2019 14th International Joint Symposium on Artificial Intelligence and Natural Language Processing, iSAI-NLP 2019*.

Sakar, C.O., Polat, S.O., Katircioglu, M. and Kastro, Y., 2018. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. *Neural Computing and Applications*, 31(10), pp.6893–6908. Available at: https://doi.org/10.1007/s00521-018-3523-0.

Taser, P.Y., 2021. Application of Bagging and Boosting Approaches Using Decision Tree-Based Algorithms in Diabetes Risk Prediction. *Proceedings*, 74(1), p.6.

Xiao, J. et al., 2020. Cost-sensitive semi-supervised selective ensemble model for customer credit scoring. *Knowledge-Based Systems*, 189, p.105118. Available at: https://doi.org/10.1016/j.knosys.2019.105118.

Xu, X. et al., 2020. Data-driven decision and analytics of collection and delivery point location problems for online retailers. *Omega (United Kingdom)*, (xxxx), p.102280. Available at: https://doi.org/10.1016/j.omega.2020.102280.

Zhang, Y., Lu, R., Huang, J.I. and Gao, D., 2019. EVOLUTIONARY-BASED ENSEMBLE UNDER-SAMPLING FOR IMBALANCED DATA. , pp.212–216.

APPENDICES

Result for applying Single Classifiers Without Any Data Pre-Processing

10 fold cross validate with 70% train set and verify result with 30% test set.

J48 Parameter	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
C0.25 M2	0.57	0.587	0.951	0.952	0.892	0.896	0.767	0.773
C0.05 M2	0.587	0.615	0.952	0.951	0.895	0.899	0.82	0.835
C0.25 M7	0.589	0.553	0.955	0.962	0.898	0.899	0.846	0.868
C0.10 M9	0.595	0.573	0.955	0.958	0.899	0.899	0.851	0.869

KNN Parameter	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
K1	0.312	0.363	0.898	0.895	0.807	0.812	0.608	0.632
K6	0.123	0.135	0.986	0.986	0.852	0.854	0.722	0.717

SMO Parameter	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Polykernel	0.335	0.356	0.981	0.982	0.881	0.885	0.658	0.669
normalized	0.22	0.233	0.992	0.99	0.872	0.873	0.606	0.611
Puk	0.162	0.193	0.992	0.993	0.863	0.87	0.577	0.593

LibSVM Parameter	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
polynomial	0.409	0.995	0.599	0.003	0.57	0.156	0.504	0.499
linear	0.473	0.636	0.694	0.231	0.66	0.294	0.583	0.433

NaïveBayes Parameter	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
	0.68	0.676	0.844	0.851	0.819	0.824	0.841	0.842

Result with Data Pre-Processing

Over Sampling

o10% (total 8,764 , positive 1,470 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.598	0.599	0.95	0.951	0.891	0.896	0.837	0.841
Naïve Bayes	0.704	0.694	0.832	0.837	0.811	0.815	0.842	0.84

o20% (total 8,898 , positive 1,604 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.652	0.669	0.939	0.937	0.887	0.896	0.834	0.861
Naïve Bayes	0.729	0.715	0.816	0.82	0.8	0.804	0.842	0.84

o30% (total 9,032 , positive 1,738 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.677	0.697	0.942	0.933	0.891	0.897	0.854	0.85
Naïve Bayes	0.747	0.723	0.81	0.81	0.797	0.796	0.845	0.84

o40% (total 9,165 , positive 1,871 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.709	0.641	0.936	0.943	0.89	0.897	0.846	0.829
Naïve Bayes	0.763	0.739	0.793	0.794	0.787	0.785	0.842	0.84

o50% (total 9,299 , positive 2,005 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.719	0.665	0.937	0.938	0.89	0.896	0.854	0.85
Naïve Bayes	0.77	0.743	0.789	0.788	0.785	0.781	0.845	0.84

o60% (total 9,433 , positive 2,139 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.727	0.653	0.934	0.936	0.887	0.892	0.859	0.833
Naïve Bayes	0.783	0.753	0.774	0.777	0.776	0.773	0.843	0.84

o70% (total 9,566 , positive 2,272 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.746	0.688	0.934	0.934	0.889	0.896	0.858	0.85
Naïve Bayes	0.798	0.783	0.754	0.756	0.764	0.76	0.843	0.838

o80% (total 9,700 , positive 2,406 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.753	0.68	0.935	0.934	0.89	0.895	0.866	0.811
Naïve Bayes	0.806	0.781	0.753	0.757	0.766	0.76	0.845	0.837

o90% (total 9,834 , positive 2,540 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.774	0.679	0.934	0.937	0.893	0.897	0.865	0.83
Naïve Bayes	0.82	0.795	0.74	0.746	0.76	0.753	0.848	0.837

o100% (total 9,968 , positive 2,674 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.776	0.692	0.933	0.927	0.891	0.891	0.874	0.816
Naïve Bayes	0.837	0.816	0.717	0.719	0.749	0.734	0.847	0.835

o110% (total 10,101 , positive 2,807 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.785	0.708	0.932	0.929	0.891	0.895	0.873	0.838
Naïve Bayes	0.836	0.795	0.726	0.727	0.756	0.738	0.849	0.835

o120% (total 10,235 , positive 2,941 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.798	0.732	0.924	0.925	0.887	0.895	0.875	0.827
Naïve Bayes	0.846	0.814	0.706	0.706	0.746	0.722	0.847	0.835

o130% (total 10,369 , positive 3075 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.805	0.711	0.926	0.926	0.89	0.892	0.88	0.821
Naïve Bayes	0.846	0.811	0.706	0.709	0.748	0.725	0.851	0.834

o140% (total 10,502 , positive 3208 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.81	0.685	0.933	0.937	0.895	0.898	0.88	0.837
Naïve Bayes	0.858	0.821	0.692	0.693	0.742	0.713	0.85	0.835

o150% (total 10,636 , positive 3,342 ; negative 7,294)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.811	0.681	0.929	0.934	0.892	0.895	0.888	0.835
Naïve Bayes	0.877	0.837	0.665	0.674	0.731	0.699	0.855	0.833

Under Sampling

u10% (total 7,900, positive 1,337 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.61	0.602	0.94	0.95	0.884	0.896	0.835	0.855
Naïve Bayes	0.684	0.692	0.836	0.843	0.81	0.819	0.84	0.841

u20% (total 7,171, positive 1,337 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.624	0.664	0.943	0.945	0.884	0.902	0.82	0.855
Naïve Bayes	0.704	0.711	0.828	0.832	0.805	0.814	0.84	0.841

u30% (total 6,441, positive 1,337 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.619	0.709	0.936	0.928	0.87	0.894	0.816	0.844
Naïve Bayes	0.723	0.722	0.811	0.813	0.793	0.799	0.838	0.839

u40% (total 5,713, positive 1,337 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.699	0.701	0.924	0.937	0.871	0.9	0.829	0.847
Naïve Bayes	0.755	0.748	0.792	0.786	0.784	0.78	0.839	0.84

u50% (total 4,984, positive 1,337 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.716	0.737	0.923	0.925	0.867	0.896	0.835	0.855
Naïve Bayes	0.8	0.793	0.758	0.75	0.769	0.757	0.842	0.844

u60% (total 4,254, positive 1,337 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.729	0.764	0.917	0.916	0.858	0.892	0.836	0.849
Naïve Bayes	0.79	0.797	0.748	0.741	0.761	0.75	0.833	0.837

u70% (total 3,525, positive 1,337 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.8	0.818	0.875	0.89	0.846	0.879	0.827	0.854
Naïve Bayes	0.814	0.82	0.718	0.711	0.754	0.728	0.83	0.833

u80% (total 2,795, positive 1,337 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.846	0.842	0.855	0.87	0.85	0.866	0.837	0.85
Naïve Bayes	0.857	0.865	0.634	0.627	0.741	0.664	0.827	0.829

Hybrid Sampling (Over Sampling + Under Sampling)

o10% u10% (total 8,033 , positive 1,470 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.633	0.643	0.94	0.945	0.884	0.898	0.83	0.848
Naïve Bayes	0.715	0.713	0.819	0.82	0.8	0.803	0.838	0.84

o20% u10% (total 8,167 , positive 1,604 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.686	0.625	0.931	0.948	0.883	0.898	0.841	0.863
Naïve Bayes	0.728	0.725	0.812	0.814	0.795	0.8	0.839	0.84

o30% u10% (total 8,301 , positive 1,738 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.693	0.681	0.935	0.936	0.884	0.896	0.848	0.851
Naïve Bayes	0.751	0.744	0.794	0.794	0.785	0.786	0.84	0.839

o40% u10% (total 8,434 , positive 1,871 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.71	0.708	0.931	0.935	0.882	0.9	0.844	0.848
Naïve Bayes	0.769	0.76	0.777	0.778	0.775	0.775	0.84	0.839

o50% u10% (total 8,568 , positive 2,005 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.732	0.695	0.932	0.932	0.885	0.896	0.845	0.843
Naïve Bayes	0.799	0.776	0.757	0.763	0.767	0.765	0.843	0.838

o60% u10% (total 8,702 , positive 2,139 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.744	0.658	0.93	0.943	0.884	0.899	0.856	0.825
Naïve Bayes	0.792	0.769	0.759	0.764	0.767	0.765	0.841	0.838

o70% u10% (total 8,835 , positive 2,272 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.751	0.695	0.927	0.935	0.882	0.898	0.848	0.824
Naïve Bayes	0.817	0.804	0.737	0.738	0.758	0.748	0.844	0.837

o80% u10% (total 8,969 , positive 2,406 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.793	0.704	0.915	0.927	0.882	0.893	0.865	0.838
Naïve Bayes	0.824	0.795	0.739	0.745	0.762	0.753	0.849	0.837

o90% u10% (total 9,103 , positive 2,540 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.785	0.718	0.922	0.928	0.884	0.896	0.862	0.85
Naïve Bayes	0.834	0.814	0.71	0.718	0.745	0.733	0.844	0.836

o100% u10% (total 9,237 , positive 2,674 ; negative 6,563)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.798	0.723	0.923	0.926	0.887	0.895	0.868	0.856
Naïve Bayes	0.845	0.823	0.705	0.71	0.745	0.727	0.847	0.835

o10% u20% (total 7,304 , positive 1,470 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.654	0.65	0.938	0.946	0.88	0.901	0.828	0.855
Naïve Bayes	0.734	0.73	0.806	0.808	0.792	0.796	0.839	0.84

o20% u20% (total 7,438 , positive 1,604 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.683	0.695	0.933	0.94	0.879	0.902	0.839	0.853
Naïve Bayes	0.747	0.746	0.796	0.795	0.785	0.788	0.839	0.84

o30%
u20% (total 7,572 , positive 1,738 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.717	0.685	0.931	0.94	0.882	0.901	0.847	0.847
Naïve Bayes	0.773	0.769	0.774	0.771	0.774	0.771	0.84	0.839

o40%
u20% (total 7,705 , positive 1,871 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.729	0.73	0.93	0.931	0.881	0.9	0.855	0.855
Naïve Bayes	0.794	0.783	0.758	0.756	0.767	0.76	0.841	0.838

o50%
u20% (total 7,839 , positive 2,005 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.745	0.674	0.93	0.943	0.883	0.901	0.862	0.863
Naïve Bayes	0.795	0.776	0.763	0.764	0.771	0.766	0.845	0.838

o60%
u20% (total 7,973 , positive 2,139 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.777	0.755	0.92	0.919	0.882	0.894	0.854	0.851
Naïve Bayes	0.814	0.802	0.736	0.737	0.757	0.747	0.843	0.838

o70%
u20% (total 8,106 , positive 2,272 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.793	0.758	0.916	0.919	0.882	0.894	0.861	0.847
Naïve Bayes	0.824	0.809	0.73	0.734	0.756	0.746	0.845	0.837

o80%
u20% (total 8,240 , positive 2,406 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.799	0.769	0.916	0.917	0.882	0.894	0.867	0.842
Naïve Bayes	0.847	0.821	0.711	0.718	0.751	0.734	0.849	0.838

o90%
u20% (total 8,374 , positive 2,540 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.794	0.739	0.921	0.926	0.883	0.897	0.869	0.859
Naïve Bayes	0.845	0.834	0.695	0.698	0.74	0.719	0.846	0.837

o100%
u20% (total 8,508 , positive 2,674 ; negative 5,834)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.83	0.783	0.911	0.91	0.885	0.89	0.869	0.847
Naïve Bayes	0.855	0.841	0.682	0.687	0.736	0.71	0.846	0.835

o10%
u30% (total 6,574 , positive 1,470 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.678	0.694	0.928	0.933	0.872	0.896	0.824	0.846
Naïve Bayes	0.746	0.741	0.792	0.793	0.782	0.785	0.836	0.837

o20%
u30% (total 6,708 , positive 1,604 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.717	0.706	0.927	0.935	0.877	0.899	0.849	0.852
Naïve Bayes	0.778	0.776	0.768	0.759	0.77	0.762	0.835	0.836

o30%
u30% (total 6,842 , positive 1,738 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.743	0.711	0.924	0.93	0.878	0.896	0.854	0.851
Naïve Bayes	0.793	0.793	0.752	0.746	0.763	0.753	0.837	0.837

o40%
u30% (total 6,975 , positive 1,871 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.749	0.688	0.924	0.934	0.877	0.896	0.854	0.841
Naïve Bayes	0.797	0.79	0.754	0.75	0.765	0.756	0.836	0.837

o50%
u30% (total 7,109 , positive 2,005 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.769	0.72	0.919	0.93	0.876	0.898	0.852	0.85
Naïve Bayes	0.812	0.802	0.737	0.737	0.758	0.747	0.84	0.836

o60%
u30% (total 7,243 , positive 2,139 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.788	0.783	0.914	0.911	0.877	0.891	0.861	0.85
Naïve Bayes	0.813	0.806	0.732	0.729	0.756	0.741	0.839	0.835

o70%
u30% (total 7,376 , positive 2,272 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.791	0.699	0.918	0.936	0.879	0.9	0.867	0.859
Naïve Bayes	0.833	0.821	0.711	0.712	0.749	0.729	0.841	0.835

o80%
u30% (total 7,510 , positive 2,406 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.814	0.772	0.904	0.918	0.875	0.895	0.863	0.849
Naïve Bayes	0.847	0.841	0.679	0.676	0.733	0.702	0.842	0.834

o90%
u30% (total 7,644 , positive 2,540 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.808	0.706	0.912	0.932	0.878	0.897	0.866	0.854
Naïve Bayes	0.863	0.844	0.662	0.67	0.729	0.697	0.844	0.834

o100%
u30% (total 7,778 , positive 2,674 ; negative 5,104)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.823	0.781	0.907	0.913	0.878	0.892	0.87	0.844
Naïve Bayes	0.864	0.849	0.653	0.659	0.726	0.689	0.843	0.834

o10%
u40% (total 5,846 , positive 1,470 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.713	0.746	0.923	0.924	0.87	0.896	0.84	0.836
Naïve Bayes	0.779	0.781	0.764	0.757	0.768	0.761	0.836	0.839

o20%
u40% (total 5,980 , positive 1,604 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.733	0.73	0.92	0.922	0.87	0.893	0.843	0.856
Naïve Bayes	0.813	0.813	0.738	0.727	0.758	0.74	0.839	0.839

o30%
u40% (total 6,114 , positive 1,738 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.751	0.713	0.923	0.93	0.874	0.897	0.856	0.819
Naïve Bayes	0.823	0.821	0.717	0.717	0.747	0.733	0.838	0.838

o40%
u40% (total 6,247 , positive 1,871 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.757	0.709	0.922	0.928	0.873	0.895	0.853	0.842
Naïve Bayes	0.842	0.828	0.709	0.709	0.749	0.727	0.842	0.837

o50%
u40% (total 6,381 , positive 2,005 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.783	0.727	0.916	0.928	0.874	0.897	0.861	0.82
Naïve Bayes	0.848	0.839	0.676	0.679	0.73	0.703	0.842	0.837

o60%
u40% (total 6,515 , positive 2,139 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.792	0.718	0.917	0.93	0.876	0.897	0.86	0.848
Naïve Bayes	0.849	0.841	0.683	0.685	0.738	0.709	0.841	0.838

o70%
u40% (total 6,648 , positive 2,272 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.801	0.748	0.915	0.918	0.876	0.892	0.864	0.853
Naïve Bayes	0.861	0.849	0.658	0.66	0.727	0.69	0.841	0.836

o80%
u40% (total 6,782 , positive 2,406 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.806	0.709	0.917	0.928	0.878	0.895	0.869	0.835
Naïve Bayes	0.869	0.849	0.639	0.639	0.721	0.672	0.844	0.836

o90%
u40% (total 6,916 , positive 2,540 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.827	0.8	0.902	0.9	0.874	0.885	0.866	0.852
Naïve Bayes	0.881	0.855	0.62	0.624	0.716	0.66	0.845	0.835

o100%
u40% (total 7,050 , positive 2,674 ; negative 4,376)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.837	0.744	0.899	0.924	0.876	0.896	0.873	0.84
Naïve Bayes	0.881	0.86	0.604	0.607	0.709	0.646	0.844	0.834

o10%
u50% (total 5,117 , positive 1,470 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.735	0.753	0.921	0.92	0.867	0.895	0.847	0.861
Naïve Bayes	0.831	0.828	0.717	0.711	0.75	0.729	0.842	0.843

o20%
u50% (total 5,251 , positive 1,604 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.751	0.737	0.921	0.923	0.869	0.894	0.852	0.854
Naïve Bayes	0.856	0.841	0.675	0.677	0.73	0.703	0.843	0.843

o30%
u50% (total 5,385 , positive 1,738 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.765	0.783	0.919	0.912	0.869	0.892	0.848	0.861
Naïve Bayes	0.869	0.86	0.639	0.635	0.713	0.67	0.843	0.843

o40%
u50% (total 5,518 , positive 1,871 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.784	0.751	0.912	0.914	0.868	0.889	0.859	0.843
Naïve Bayes	0.877	0.867	0.618	0.622	0.706	0.66	0.844	0.843

o50%
u50% (total 5,652 , positive 2,005 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.796	0.786	0.91	0.905	0.87	0.887	0.858	0.855
Naïve Bayes	0.886	0.867	0.604	0.606	0.704	0.646	0.842	0.843

o60%
u50% (total 5,786 , positive 2,139 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.798	0.757	0.908	0.918	0.867	0.893	0.858	0.856
Naïve Bayes	0.904	0.87	0.582	0.581	0.701	0.626	0.849	0.842

o70%
u50% (total 5,919 , positive 2,272 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.814	0.75	0.909	0.92	0.873	0.894	0.861	0.849
Naïve Bayes	0.895	0.87	0.58	0.576	0.701	0.622	0.846	0.839

o80%
u50% (total 6,053 , positive 2,406 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.838	0.8	0.897	0.893	0.874	0.878	0.878	0.861
Naïve Bayes	0.891	0.87	0.59	0.586	0.709	0.63	0.847	0.841

o90%
u50% (total 6,187 , positive 2,540 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.83	0.79	0.897	0.905	0.87	0.887	0.868	0.857
Naïve Bayes	0.904	0.869	0.567	0.564	0.705	0.611	0.85	0.839

o100%
u50% (total 6,321 , positive 2,674 ; negative 3,647)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.827	0.755	0.902	0.918	0.87	0.893	0.863	0.85
Naïve Bayes	0.9	0.877	0.539	0.535	0.692	0.588	0.847	0.838

o10%
u60% (total 4,387 , positive 1,470 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.755	0.764	0.912	0.916	0.859	0.892	0.831	0.839
Naïve Bayes	0.818	0.814	0.715	0.715	0.749	0.73	0.83	0.837

o20%
u60% (total 4,521 , positive 1,604 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.774	0.748	0.909	0.924	0.862	0.897	0.848	0.844
Naïve Bayes	0.835	0.834	0.693	0.696	0.743	0.717	0.832	0.836

o30%
u60% (total 4,655 , positive 1,738 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.784	0.755	0.905	0.914	0.86	0.889	0.845	0.85
Naïve Bayes	0.854	0.844	0.663	0.668	0.734	0.695	0.832	0.835

o40%
u60% (total 4,788 , positive 1,871 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.807	0.769	0.893	0.916	0.859	0.893	0.853	0.863
Naïve Bayes	0.849	0.844	0.658	0.664	0.732	0.692	0.831	0.833

o50%
u60% (total 4,922 , positive 2,005 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.819	0.771	0.89	0.913	0.861	0.891	0.851	0.861
Naïve Bayes	0.864	0.858	0.637	0.64	0.73	0.674	0.832	0.833

o60%
u60% (total 5,056 , positive 2,139 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.827	0.8	0.889	0.899	0.863	0.884	0.858	0.851
Naïve Bayes	0.882	0.869	0.605	0.606	0.723	0.647	0.834	0.833

o70%
u60% (total 5,189 , positive 2,272 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.847	0.814	0.878	0.894	0.864	0.881	0.858	0.853
Naïve Bayes	0.879	0.87	0.588	0.59	0.716	0.633	0.833	0.833

o80%
u60% (total 5,323 , positive 2,406 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.854	0.802	0.88	0.9	0.868	0.885	0.864	0.853
Naïve Bayes	0.892	0.874	0.569	0.571	0.715	0.617	0.836	0.833

o90%
u60% (total 5,457 , positive 2,540 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.856	0.8	0.874	0.899	0.865	0.884	0.873	0.85
Naïve Bayes	0.886	0.87	0.578	0.576	0.721	0.621	0.835	0.832

o100%
u60% (total 5,591 , positive 2,674 ; negative 2,917)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.862	0.818	0.875	0.89	0.869	0.879	0.864	0.854
Naïve Bayes	0.901	0.876	0.545	0.542	0.715	0.593	0.84	0.83

o10%
u70% (total 3,658 , positive 1,470 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.825	0.841	0.867	0.87	0.85	0.865	0.845	0.857
Naïve Bayes	0.838	0.844	0.685	0.679	0.747	0.705	0.83	0.831

o20%
u70% (total 3,792 , positive 1,604 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.832	0.837	0.867	0.864	0.852	0.86	0.843	0.859
Naïve Bayes	0.857	0.858	0.652	0.648	0.738	0.681	0.829	0.832

o30%
u70% (total 3,926 , positive 1,738 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.854	0.837	0.86	0.864	0.858	0.86	0.86	0.859
Naïve Bayes	0.873	0.865	0.627	0.619	0.736	0.657	0.831	0.831

o40%
u70% (total 4,059 , positive 1,871 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.856	0.842	0.863	0.871	0.86	0.867	0.858	0.86
Naïve Bayes	0.881	0.874	0.599	0.602	0.729	0.644	0.83	0.831

o50%
u70% (total 4,193 , positive 2,005 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.866	0.835	0.863	0.875	0.865	0.869	0.861	0.857
Naïve Bayes	0.876	0.867	0.614	0.617	0.739	0.655	0.83	0.831

o60%
u70% (total 4,327 , positive 2,139 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.866	0.835	0.861	0.874	0.863	0.868	0.864	0.854
Naïve Bayes	0.891	0.874	0.587	0.585	0.737	0.629	0.834	0.83

o70%
u70% (total 4,460 , positive 2,272 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.875	0.837	0.859	0.866	0.867	0.861	0.868	0.853
Naïve Bayes	0.898	0.877	0.565	0.56	0.735	0.609	0.833	0.828

o80%
u70% (total 4,594 , positive 2,406 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.894	0.842	0.841	0.871	0.869	0.866	0.886	0.895
Naïve Bayes	0.893	0.877	0.553	0.548	0.731	0.599	0.832	0.828

o90%
u70% (total 4,728 , positive 2,540 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.893	0.862	0.839	0.853	0.868	0.854	0.887	0.887
Naïve Bayes	0.91	0.881	0.519	0.516	0.729	0.572	0.835	0.827

o100%
u70% (total 4,862 , positive 2,674 ; negative 2,188)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.906	0.853	0.831	0.859	0.872	0.858	0.897	0.894
Naïve Bayes	0.9	0.877	0.529	0.522	0.733	0.577	0.834	0.824

o10%
u80% (total 2,928 , positive 1,470 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.854	0.842	0.85	0.869	0.852	0.865	0.85	0.859
Naïve Bayes	0.882	0.874	0.603	0.592	0.743	0.636	0.829	0.827

o20%
u80% (total 3,062 , positive 1,604 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.05 M2	0.858	0.844	0.848	0.869	0.853	0.865	0.85	0.861
Naïve Bayes	0.887	0.877	0.593	0.582	0.747	0.628	0.831	0.826

o30%
u80% (total 3,196 , positive 1,738 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.864	0.842	0.852	0.872	0.858	0.867	0.85	0.862
Naïve Bayes	0.89	0.877	0.586	0.582	0.752	0.627	0.829	0.826

o40%
u80% (total 3,329 , positive 1,871 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.881	0.842	0.838	0.87	0.862	0.865	0.871	0.86
Naïve Bayes	0.899	0.89	0.547	0.536	0.745	0.591	0.827	0.826

o50%
u80% (total 3,463 , positive 2,005 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.887	0.842	0.838	0.87	0.866	0.865	0.866	0.86
Naïve Bayes	0.904	0.893	0.527	0.512	0.745	0.571	0.826	0.825

o60%
u80% (total 3,597 , positive 2,139 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.891	0.842	0.831	0.875	0.867	0.87	0.876	0.865
Naïve Bayes	0.907	0.89	0.527	0.522	0.753	0.579	0.829	0.825

o70%
u80% (total 3,730 , positive 2,272 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.919	0.879	0.799	0.816	0.872	0.826	0.885	0.892
Naïve Bayes	0.917	0.897	0.489	0.475	0.75	0.54	0.83	0.825

o80%
u80% (total 3,864 , positive 2,406 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.929	0.879	0.798	0.796	0.879	0.809	0.887	0.875
Naïve Bayes	0.916	0.897	0.756	0.465	0.491	0.532	0.831	0.825

o90%
u80% (total 3,998 , positive 2,540 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.932	0.891	0.799	0.809	0.884	0.822	0.893	0.885
Naïve Bayes	0.916	0.898	0.478	0.46	0.756	0.528	0.832	0.825

o100%
u80% (total 4,132 , positive 2,674 ; negative 1,458)

Algorithm	True Positive		True Negative		Accuracy		ROC Area	
	70% train	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**	**70% train**	**30% test**
J48 C0.05 M2	0.937	0.884	0.775	0.79	0.88	0.805	0.882	0.872
Naïve Bayes	0.923	0.898	0.479	0.457	0.766	0.525	0.834	0.823

Result with Ensemble Learning: AdaBoost

based learner	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.25 M2	0.57	0.585	0.941	0.943	0.883	0.888	0.892	0.902
J48 C0.05 M2	0.563	0.571	0.943	0.948	0.884	0.89	0.895	0.897
J48 C0.25 M7	0.564	0.557	0.937	0.94	0.879	0.881	0.888	0.89
J48 C0.10 M9	0.578	0.546	0.945	0.949	0.888	0.887	0.894	0.902
J48 C0.35 M2	0.577	0.602	0.941	0.941	0.885	0.889	0.897	0.902
J48 C0.35 M1	0.576	0.609	0.939	0.942	0.883	0.891	0.894	0.905
Naïve Bayes	0.521	0.506	0.932	0.942	0.868	0.875	0.832	0.829
Decision S.	0.583	0.59	0.949	0.955	0.893	0.899	0.909	0.916
REP tree	0.568	0.587	0.943	0.946	0.885	0.891	0.899	0.899

Result with Ensemble Learning: Bagging

based learner	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
J48 C0.25 M2	0.592	0.599	0.954	0.954	0.898	0.899	0.92	0.925
J48 C0.05 M2	0.587	0.595	0.957	0.957	0.9	0.901	0.855	0.874
J48 C0.25 M7	0.603	0.583	0.956	0.959	0.901	0.901	0.916	0.924
J48 C0.10 M9	0.596	0.601	0.956	0.958	0.9	0.903	0.857	0.876
J48 C0.35 M1	0.58	0.618	0.953	0.951	0.895	0.9	0.918	0.921
Naïve Bayes	0.684	0.68	0.843	0.85	0.818	0.824	0.843	0.847
REP tree	0.591	0.608	0.955	0.961	0.899	0.906	0.924	0.927
Decision S.	0.789	0.797	0.89	0.895	0.875	0.88	0.833	0.855

Result with Ensemble Learning Plus Under Sampling

Undersampling 10%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.58	0.608	0.937	0.939	0.876	0.888	0.895	0.899
Boost NB	0.542	0.55	0.922	0.931	0.857	0.872	0.832	0.833
Bag J48	0.604	0.623	0.948	0.946	0.889	0.896	0.918	0.922
Bag NB	0.685	0.687	0.835	0.844	0.81	0.82	0.843	0.844

Undersampling 20%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.601	0.622	0.931	0.934	0.87	0.886	0.895	0.903
Boost NB	0.568	0.585	0.914	0.914	0.85	0.863	0.838	0.835
Bag J48	0.614	0.63	0.942	0.948	0.881	0.899	0.92	0.921
Bag NB	0.697	0.713	0.825	0.82	0.801	0.803	0.841	0.845

Undersampling 30%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.633	0.636	0.922	0.929	0.862	0.884	0.895	0.899
Boost NB	0.585	0.618	0.895	0.893	0.83	0.851	0.836	0.834
Bag J48	0.655	0.676	0.931	0.937	0.874	0.897	0.918	0.925
Bag NB	0.726	0.73	0.807	0.809	0.79	0.797	0.841	0.844

Undersampling 40%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.644	0.667	0.91	0.919	0.847	0.881	0.893	0.901
Boost NB	0.601	0.553	0.886	0.905	0.82	0.851	0.841	0.831
Bag J48	0.675	0.697	0.919	0.927	0.862	0.891	0.916	0.921
Bag NB	0.761	0.736	0.79	0.801	0.783	0.791	0.841	0.845

Undersampling 50%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.68	0.671	0.903	0.899	0.843	0.863	0.901	0.906
Boost NB	0.696	0.743	0.834	0.814	0.797	0.803	0.849	0.844
Bag J48	0.722	0.739	0.91	0.916	0.859	0.889	0.92	0.921
Bag NB	0.792	0.778	0.758	0.769	0.767	0.77	0.844	0.85

Undersampling 60%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.705	0.73	0.872	0.887	0.82	0.863	0.893	0.908
Boost NB	0.678	0.667	0.834	0.837	0.785	0.811	0.838	0.836
Bag J48	0.749	0.748	0.894	0.902	0.848	0.878	0.918	0.92
Bag NB	0.787	0.806	0.751	0.735	0.762	0.746	0.838	0.84

Undersampling 70%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.758	0.76	0.859	0.869	0.82	0.852	0.898	0.906
Boost NB	0.758	0.769	0.777	0.766	0.77	0.767	0.843	0.843
Bag J48	0.797	0.809	0.872	0.877	0.843	0.866	0.919	0.921
Bag NB	0.815	0.82	0.712	0.711	0.751	0.728	0.832	0.834

Undersampling 80%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.805	0.82	0.82	0.825	0.813	0.824	0.899	0.903
Boost NB	0.826	0.825	0.703	0.717	0.762	0.733	0.839	0.847
Bag J48	0.843	0.842	0.822	0.827	0.832	0.829	0.913	0.915
Bag NB	0.85	0.867	0.644	0.624	0.743	0.662	0.83	0.832

Result with Ensemble Learning Plus Over Sampling

Oversampling 20%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.629	0.592	0.938	0.941	0.882	0.887	0.91	0.903
Boost NB	0.552	0.52	0.92	0.915	0.853	0.854	0.841	0.82
Bag J48	0.658	0.648	0.945	0.946	0.893	0.9	0.928	0.924
Bag NB	0.724	0.725	0.812	0.812	0.796	0.798	0.844	0.841

Oversampling 40%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.681	0.623	0.937	0.932	0.885	0.885	0.917	0.901
Boost NB	0.623	0.611	0.896	0.908	0.84	0.862	0.849	0.834
Bag J48	0.711	0.636	0.942	0.946	0.895	0.898	0.937	0.924
Bag NB	0.768	0.73	0.791	0.804	0.786	0.793	0.845	0.843

Oversampling 60%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.736	0.606	0.93	0.94	0.886	0.888	0.931	0.901
Boost NB	0.68	0.727	0.855	0.818	0.816	0.804	0.851	0.832
Bag J48	0.752	0.66	0.937	0.943	0.895	0.899	0.942	0.923
Bag NB	0.787	0.751	0.772	0.776	0.776	0.772	0.845	0.842

Oversampling 80%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.756	0.625	0.929	0.934	0.886	0.887	0.937	0.905
Boost NB	0.682	0.644	0.849	0.859	0.807	0.826	0.847	0.83
Bag J48	0.775	0.687	0.938	0.939	0.897	0.9	0.947	0.921
Bag NB	0.804	0.76	0.752	0.77	0.765	0.769	0.848	0.837

Oversampling 100%

Ensemble	True Positive		True Negative		Accuracy		ROC Area	
	70% train	30% test	70% train	30% test	70% train	30% test	70% train	30% test
Boost J48	0.789	0.618	0.93	0.938	0.892	0.888	0.943	0.9
Boost NB	0.748	0.709	0.826	0.829	0.805	0.81	0.852	0.838
Bag J48	0.805	0.669	0.936	0.935	0.901	0.894	0.951	0.923
Bag NB	0.838	0.828	0.717	0.701	0.749	0.721	0.851	0.84