

**VEHICLE INSURANCE FRAUD DETECTION USING
MACHINE LEARNING WITH VARIOUS IMBALANCED
DATASET HANDLING METHODS**

KOH RONG SOON

**RESEARCH REPORT SUBMITTED IN FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF DATA SCIENCE**

**FACULTY OF COMPUTER SCIENCE &
INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR**

2024

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: **KOH RONG SOON** (I.C/Passport No: **000223050399**)

Registration/Matric No: **22061463**

Name of Degree: **MASTER OF DATA SCIENCE**

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):

**VEHICLE INSURANCE FRAUD DETECTION USING MACHINE LEARNING
WITH VARIOUS IMBALANCED DATASET HANDLING METHODS**

Field of Study:

I do solemnly and sincerely declare that:

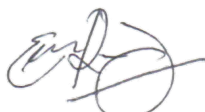
- (1) I am the sole author/writer of this Work;
- (2) This Work is original;
- (3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly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

Candidate's Signature



Date 26/1/2024

Subscribed and solemnly declared before,



Witness's Signature

Date 26/1/2024

Name:

Designation:

DR ERMA RAHAYU MOHD FAIZAL ABDULLAH
HEAD
DEPARTMENT OF ARTIFICIAL INTELLIGENCE
FACULTY OF COMPUTER SCIENCE &
INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
50603 KUALA LUMPUR

[VEHICLE INSURANCE FRAUD DETECTION USING MACHINE LEARNING WITH VARIOUS IMBALANCED DATASET HANDLING METHODS]

ABSTRACT

Predicting vehicle fraud insurance claims is difficult as the datasets used to train the machine learning model are imbalanced. The imbalance vehicle insurance fraud dataset contains more legitimate claims compared to fraud claim data. It is difficult for machine learning models to have good performance to predict the fraud claim as the trained model with an imbalanced dataset will be biased toward the majority class. The different imbalanced datasets can be suitable for different handling techniques. In addition, the literature review of recent research papers also will be carried out to find out the current problems of the existing research and form the research gap. Therefore, this research focuses on applying several imbalanced dataset handling techniques which are SMOTE, Random Undersampling, and hybrid sampling to sample the data before the data is used for modelling. The Ensemble Learning of Boosting and Bagging is applied as well to handle the imbalanced dataset. The performance of the trained models by applying different techniques in handling imbalanced datasets was evaluated by using accuracy, recall, specificity and F1-Score. The best performance of the trained model which is Ensemble Learning of a Combination of Boosting and Decision Tree with a ratio of 0.4 undersampling outperformed all the trained models.

Keywords: Vehicle Fraud Insurance Claim, Imbalanced Dataset, Machine Learning, Python, Sampling

TABLE OF CONTENTS

[VEHICLE INSURANCE FRAUD DETECTION USING MACHINE LEARNING WITH VARIOUS IMBALANCED DATASET HANDLING METHODS] Abstract ...	iii
Table of Contents	iv
List of Figures	vi
List of Tables.....	viii
List of Symbols and Abbreviations.....	ix
List of Appendices	x
 CHAPTER 1: INTRODUCTION.....	1
1.1 Research Background	1
1.2 Problem Statement.....	2
1.3 Research Questions.....	4
1.4 Research Objectives.....	5
1.5 Research Significance.....	5
 CHAPTER 2: LITERATURE REVIEW.....	6
2.1 Introduction.....	6
2.2 Existing Research	6
2.3 Imbalanced Dataset Solutions.....	13
2.3.1 Sampling.....	13
2.3.2 Ensemble Learning.....	17
2.4 Evaluation Metric	24
2.5 Summary	25
 CHAPTER 3: RESEARCH METHODOLOGY	27

3.1	Research Design	27
3.2	Data Understanding	27
3.3	Data Preparation	31
3.4	Modelling.....	34
3.5	Evaluation.....	35
3.6	Workflow Diagram.....	35
CHAPTER 4: RESULT AND DISCUSSION.....		37
4.1	Without Applying Data Sampling Techniques.....	37
4.2	Oversampling (SMOTE)	38
4.3	Undersampling.....	41
4.4	Hybrid Sampling.....	43
4.5	Comparison between Oversampling, Undersampling and Hybrid Sampling.....	51
4.6	Ensemble Learning	52
4.7	Comparison.....	60
CHAPTER 5: CONCLUSION.....		62
References		64
Appendix		68

LIST OF FIGURES

Figure 2.1: Confusion matrix's format (Kulkarni et al., 2020).....	24
Figure 3.1: Bar chart for the count of target variable of “FraudFound_P”	30
Figure 3.2: Bar chart of distribution for each class to the target variable of “FraudFound_P”	31
Figure 3.3: Workflow Diagram.....	35
Figure 4.1: Results of Decision Tree by applying SMOTE	39
Figure 4.2: Results of Naïve Bayes with applying SMOTE.....	39
Figure 4.3: Result of Logistic Regression with applying SMOTE.....	40
Figure 4.4: Result of Decision Tree with applying Undersampling	41
Figure 4.5: Result of Naïve Bayes with applying Undersampling	42
Figure 4.6: Result of Logistic Regression with applying Undersampling.....	42
Figure 4.7: Accuracy result of Decision Tree with applying Hybrid Sampling	44
Figure 4.8: Recall result of Decision Tree by applying Hybrid Sampling	44
Figure 4.9: Specificity result of Decision Tree with applying Hybrid Sampling.....	45
Figure 4.10: F1-Score result of Decision Tree by applying Hybrid Sampling	45
Figure 4.11: Accuracy result of Naive Bayes with applying Hybrid Sampling.....	46
Figure 4.12: Recall result of Naive Bayes with applying Hybrid Sampling	46
Figure 4.13: Precision result of Naive Bayes with applying Hybrid Sampling	47
Figure 4.14: F1-Score result of Naive Bayes with applying Hybrid Sampling.....	47
Figure 4.15: Accuracy result of Logistic Regression applying Hybrid Sampling...	48
Figure 4.16: Recall result of Logistic Regression with applying Hybrid Sampling	48
Figure 4.17: Specificity result of Logistic Regression with applying Hybrid Sampling	49

Figure 4.18: F1-Score result of Logistic Regression with applying Hybrid Sampling	49
Figure 4.19: Result of Ensemble Learning (Boosting + Decision Tree) with applying SMOTE	53
Figure 4.20: Result of Ensemble Learning (Boosting + Decision Tree) with applying Undersampling	53
Figure 4.21: Result of Ensemble Learning (Bagging + Decision Tree) with applying SMOTE	54
Figure 4.22: Result of Ensemble Learning (Bagging + Decision Tree) with applying Undersampling	55
Figure 4.23: Accuracy result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling	55
Figure 4.24: Recall result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling	56
Figure 4.25: Specificity result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling	56
Figure 4.26: F1-Score result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling	56
Figure 4.27: Accuracy result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling	57
Figure 4.28: Recall result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling	57
Figure 4.29: Specificity result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling	58
Figure 4.30: F1-Score result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling	58

LIST OF TABLES

Table 2.1: Summary of existing research in detecting vehicle fraud insurance.....	10
Table 2.2: Summary of Literature Review of Hybrid Sampling Method to Handle Imbalanced Dataset.....	15
Table 2.3: Summary of Literature Review of Ensemble Learning Method to Handle Imbalanced Dataset.....	20
Table 2.4: Formula and description of evaluation metric	24
Table 3.1: Data description for the dataset.....	28
Table 3.2: Performance of 5 base models with using three data splitting ratios of 40:60, 70:30 and 80:20	33
Table 4.1: Classification result of five trained models without using any data sampling method compared with (Aslam et al., 2022).....	37
Table 4.2: Summary result for models with and without apply data sampling methods	51
Table 4.3: Result of trained Ensemble Learning model	52
Table 4.4: Summary result Ensemble Learning (Boosting & Bagging + Decision Tree) with and without applying handling method	59
Table 4.5: Summary results of all best trained model with application of data sampling method.	60

LIST OF SYMBOLS AND ABBREVIATIONS

For examples:

DT	:	Decision Tree
LR	:	Logistic Regression
MLP	:	Multi-Layer Perceptron
KNN	:	K-Nearest Neighbors
NB	:	Naïve Bayes

LIST OF APPENDICES

Appendix A: Decision Tree Result	68
.....	
Appendix B: Navies Bayes Result	70
.....	
Appendix C: Logistic Regression Result	72
.....	
Appendix D: Ensemble Learning Result	75
.....	
Appendix E: Python Project Code	79
.....	

CHAPTER 1: INTRODUCTION

1.1 Research Background

Nowadays, the number of vehicles in most countries is increasing quickly. It has become very common for each family to have their cars and use them in daily routines. According to (Malaysia Number of Registered Vehicles, n.d.), Malaysia reached a total of 14,430,256 registered vehicles in 2015 and it increased year by year until 2021 which recorded a total of 17,782,482 registered vehicles. Due to the number of vehicles increasing year by year, the number of accidents that happen daily also increases and causes more vehicle injuries. This will lead to the vehicle owners needing to bear the cost of repairing the vehicle and spend more. To avoid the financial burden for vehicle owners, the majority of vehicle owners prefer to purchase vehicle insurance as it can provide financial protection to them. By subscribing to vehicle insurance, the car insurance companies are responsible for covering the cost of repairing vehicles when an accident occurs.

However, some people use car insurance for a bad purpose, which is fraud. They carried out fraud to gain more claims from insurance companies and benefit them. According to (Tan, 2021), the author mentioned that insurance claims fraud in Malaysia cost RM 1 billion per year. There are normally 2 types of fraud which are soft fraud and hard fraud. Hard fraud is the fraud that happens with someone purposely planning for a loss of the vehicle and claim from the insurance while soft fraud is someone purposely magnifying the claim to claim more from the insurance company. According to (5 types of car insurance fraud, 2022), most vehicle insurance fraud cases originate from soft fraud cases instead of hard fraud. Moreover, if fraud happens frequently, it also causes a lot of negative impacts on the insurance company and policyholders. The impact on the insurance company will cause the insurance companies to lose money as they pay the claims for the policyholders who perform fraud. Since the insurance companies lose

money in insurance policies, they will increase the price of insurance plans and cause the policyholders required to pay more to get protected by the insurance plans. Therefore, fraud is an immoral action and it should be prevented from everyone.

Nowadays, most insurance companies detect vehicle insurance fraud through traditional ways which are manpower inspections. Inspecting fraud through traditional ways is ineffective, costly, and time-consuming. In addition, if the cases of insurance claims increase, it causes the manpower inspection of fraud to be inaccurate due to human mistakes more easily happening because there is a lot of data that needs to be reviewed, processed and analyzed to identify whether it is a fraud case. According to (Nur Prasasti et al., 2020), in recent years, several researchers have proposed various types of machine learning methods for automobile insurance companies to apply in automobile fraud detection. By applying machine learning in fraud detection, it can bring some benefits which are reduced cost of operation by decreasing the manual labor to detect fraud and providing automobile fraud detection continuously throughout the day.

1.2 Problem Statement

The various types of machine learning models to predict fraud detection have some limitations which is the performance of the prediction model of fraud affected by imbalanced datasets. The automobile insurance fraud dataset in real life normally is an imbalanced dataset that contains the majority of legitimate claims and the minority of fraud claims. The author has mentioned that one of the major issues is that training the machine learning model using the imbalance datasets will affect the model to predict the outcome biased on the majority class data (Nur Prasasti et al., 2020). The model will have low accuracy to identify the minority class data compared to the majority class data. According to (Caruana & Grech, 2021), the author used Naive Bayes (NB) to prove the

accuracy of the model affected by the imbalanced datasets. The author used the 70:30 legit-to-fraud dataset to train the model and get 0.759 accuracy, 0.147 precision and 0.588 sensitivity for the NB model. While the author used a 94:6 legit-to-fraud dataset to train the model, the NB model with accuracy of 0.923, 0.038 precision and 0.010 sensitivity. From the study, it can be proven that the imbalanced dataset affected the machine learning model performance and the author suggested resampling the imbalance dataset before using it to train the model.

There are some research papers done by the author in developing a predictive model for automobile fraud detection that did not handle the imbalanced dataset. According to (Aslam et al., 2022), the author has proposed to develop a predictive model for automobile insurance detection using Linear Regression, SVM and Naive Bayes. According to (Yankol-Schalck, 2022), the author came out with predictive machine learning models using Neural Network and Gradient Boost Model for predicting automobile fraud. The author also did not mention the application of the imbalanced dataset handling technique although the dataset used to train the model is highly imbalanced.

Some researchers proposed to carry out resampling the imbalance dataset before using it to train the model but only applied only single solution. According to (Nur Prasasti et al., 2020), the authors proposed the data sampling method that was used to resample the imbalance dataset which is the Synthetic Minority Oversampling Technique (SMOTE). According to (Kini et al., 2022), the author also used SMOTE in resampling the imbalance dataset before using it to train the machine learning model. According to (Sundarkumar et al., 2015), the author applied the Undersampling method to treat the imbalanced dataset before proceeding to the modelling. These three studied research papers showed the common problem of only a single imbalanced dataset handling method applied.

The imbalanced dataset that is used to train the model is required to handle effectively because it affects the performance model. Through the literature review, it was found that some works done by the authors do not handle the imbalanced dataset and in some work, only one technique was used in handling the imbalanced dataset. However, there are still many techniques that can be used to resample the imbalance datasets which are the Adaptive Synthetic Sampling Technique (ADASYN), Selective Pre-processing of imbalance data (SPIDER), and Balanced Bagging Classifier (Malhotra & Kamal, 2019). In addition, it also found that existing past studies did not apply the different ratios to oversample and undersample the imbalanced dataset. The commonly used ratios included the 50:50 ratio and 70:30 ratio from past studies. Different characteristics of the imbalance dataset can be suitable for the different methods and ratios to handle the imbalance dataset to prevent overfitting. From the current existing problem of handling the imbalanced data in the vehicle fraud insurance domain, the research gap formed is most research applies none or single imbalanced dataset handling techniques and only applies fixed ratio to sample the imbalanced data for vehicle fraud claim detection.

1.3 Research Questions

1. What methods can be used to improve the performance of vehicle fraud detection by handling the imbalanced dataset?
2. What is the optimized resampling ratio of the dataset for each trained model?
3. What is the performance of machine learning models with applying different methods of handling the imbalanced dataset?

1.4 Research Objectives

1. To identify the methods for handling imbalanced datasets to improve the performance of the vehicle fraud detection model.
2. To identify the best resampling ratio of the dataset to get the optimized performance of each trained model.
3. To evaluate the performance of methods used in handling imbalanced datasets for each machine learning model.

1.5 Research Significance

The research scope is to solve the problem by developing a machine learning model using the most suitable technique in handling imbalanced datasets that are used to train the model. This research will study different techniques that can be used to handle and resample the imbalanced data and compare the most suitable techniques that can apply to the model to get higher accuracy of prediction. The most optimized ratio of sampling method applied also would be identified for each model throughout this project.

The research contribution is to help the vehicle insurance company to detect fraud claims more accurately by using the machine learning model with the most optimized data sampling method and decrease the loss to the company. In addition, it also benefits the policyholders because they can decrease the premium that they pay to vehicle insurance companies as the paid premium includes the cost to cover the fraud amount.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

In this chapter, it will focus on carrying out the literature review on several topics including the existing research in the vehicle fraud insurance detection domain, imbalanced dataset solution and evaluation metrics. The findings from each reviewed research paper were discussed in detail in the following section.

2.2 Existing Research

According to the study from (Aslam et al., 2022), this study proposed insurance fraud detection for the vehicle insurance industry by using predictive models. In this study, the author has developed 3 machine learning models to predict vehicle insurance fraud which are Logistic Regression, Support Vector Machine and Naïve Bayes. In addition, in the research, the author has implemented the Feature Selection algorithm which is Boruta to process the dataset and it does not mention any methods to handle the imbalanced dataset used to train the model. Every model that is trained is evaluated by six metrics which are accuracy, precision, recall, f1-score, sensitivity and specificity. Among the three models that the author trained, the model that trained with the Linear Regression outperformed the others algorithm which has the Accuracy of 88.02%, Precision of 22.29%, Recall of 42.25%, F1-Score of 26.27%, Sensitivity of 42.25% and Specificity of 90.93%.

The research study from (Caruana & Grech, 2021) was aimed to detect fraudulent claims in the automobile insurance field by applying statistical techniques to develop predictive models. There are two algorithms which are Artificial Neural Networks and Naïve Bayes used to train the model and both algorithms are evaluated and compared. In addition, the researcher also mentioned the imbalanced dataset which consists of only a small number of fraudulent claims in the dataset used in training the model. The imbalanced dataset handling method which is random under sampling has been discussed

and used in the research. The ratio of 7:3 which is 70% of legit data and 30% of fraud data is chosen by the author to sample the highly imbalanced dataset. The original dataset and the resampled dataset are used by trained with the Naïve Bayes and Artificial Neural Networks. The trained models are evaluated by five metrics which are Accuracy, AUC, Precision, Sensitivity and F1-Score. From the results, the performance of Naïve Bayes when using a dataset with undergoes under sampling is better performance compared to the dataset that does not undergo under sampling. The Sensitivity value is 0.010 when using the original dataset and the value of Precision is 0.546 when using the resampled dataset. The higher sensitivity means that the model able to predict for a positive prediction which means the model is better at predicting the positive class of fraud.

According to (Harjai et al., 2019), the study has proposed and developed a machine learning model by using the algorithm of random forest together with SMOTE. In the study, the dataset used to train the model is highly imbalanced which consists of 14497 non-fraudulent data and 923 fraud data. Due to the imbalanced data, SMOTE has been applied in the study to handle the imbalanced dataset problem. After SMOTE was applied, the minority class oversampled to 1846 while the majority class remained the same. After the model was trained, it was used to compare with the various models which are Support Vector Machine, Decision Tree and Multi-layer Perceptron (MLP). The proposed model outperformed compared to other models which have 94.33 accuracy, 99.9 recall and 45.1 specificity.

According to (Yankol-Schalck, 2022), research has been carried out to develop the machine learning model for the detection of vehicle insurance fraud. In the research, the dataset used is collected from the French fraud insurance dataset from 2012 to 2016. Two models which are Neural Network and Gradient Boosting have been trained with highly imbalanced data which contain 183804 non-fraud and 1411 fraud. The model is further

to compare the performance. The result shows that Gradient Boosting has a better performance compared to Neural Networks. Gradient Boosting has the result of 0.563 Precision, 0.751 Recall, 0.643 F1-Score, 0.456 FDR, 0.965 AUC, 0.72 AUC-PR, 0.015 log-loss and 0.003 Brier score.

Another study (Dilkhaz Y, 2021) also proposed to detect vehicle fraud detection by using various machine learning methods which are J48, Random Forest, Naïve Bayes, Bayes Net and Random Tree. In the preprocessing stage, there are some steps carried out which are data transformation, handling missing values and data sampling. The result showed that J48 has the highest accuracy while the Random Forest is the best performing model compared to other trained models in terms of accuracy, precision, recall, F1-Score and MCC.

Another study by (Nur Prasasti et al., 2020), it has proposed using supervised classifiers which are Multi-Layer Perceptron, Random Forest and Decision Tree to detect for automobile insurance fraud. The dataset that was used to train the model contained 1849 rows of legit data and 32 fraud data which is highly imbalanced. The SMOTE has been applied in the data preprocessing stage to sample the imbalanced dataset. For the results, Random Forest is the best performing model which has an accuracy of 0.985, sensitivity of 0.1 and specificity of 0.985 and 0.997 of AUC value.

According to (Moon et al., 2019), the authors have proposed using four machine learning models which are Logistic Regression, Random Forest, LASSO and SVM to develop an automobile fraud insurance detection system. The dataset used to train the models is highly imbalanced and stratified random sampling has been used to sample out the 500 fraud data and 500 non-fraud data. The result of the model evaluation shows that LASSO outperformed among other trained models which has the 0.977 accuracy, 0.7 sensitivity, 0.986 specificity, 0.607 PPV, 0.979 NPV and 0.853 AUC.

In the study of (Sundarkumar et al., 2015), the authors have proposed the One Class Support vector machine with undersampling to handle the imbalanced dataset. There are two datasets which are the credit card customer churn dataset and the automobile fraud insurance dataset have been used to train the models. To evaluate the proposed method, the authors have trained other 5 models which are Support Vector Machine, Logistic Regression, Decision Tree, Group Method of Data Handling and Probabilistic Neural Network to compare the performance. The evaluation metrics used to evaluate the performance the accuracy, sensitivity, specificity, and AUC. For the proposed method by the authors, it has successfully improved the performance to classify the correct value.

In the study of (Panigrahi & Palkar, 2018), the authors have carried out a comparative analysis of the different types of classification models to detect vehicle fraud insurance claims. In the study, there are three proposed feature selection methods which are L1-Based Feature Selection, Tree-Based Feature Selection and Univariate Feature Selection to preprocess the dataset before the dataset is used to train the model. Four machine learning algorithms were applied which were the K-Nearest Neighbour, Decision Tree, Random Forest and Naïve Bayes. The evaluation metrics used to evaluate the train model included the accuracy, precision and recall. Random Forest with applied L1-based feature selection outperforms among the trained model with the 96.73 accuracy, 99.10 precision and 94.32 recall.

The study of (Rukhsar et al., 2022) was proposed to predict insurance fraud by using various machine learning algorithms. A comparative study was conducted on eight machine learning algorithms proposed by authors which are multi-layer perceptron, support vector machine, linear regression, AdaBoost, Naïve Bayes, random forest, decision tree and k-nearest neighbour. The metrics used to evaluate the model were recall, precision and f1-score. In the pre-processing stage, the data transformation with using

one hot encoder was carried out to handle the categorical data. For the result part, the authors came out with the decision tree outperforming compared to other algorithms which achieved 79% of accuracy, 62% precision, 61% of recall and 61% of F1-Score.

Table 2.1: Summary of existing research in detecting vehicle fraud insurance

Authors	Imbalanced dataset ratio	Data sampling method	Machine Learning algorithms	Evaluation Metric
(Aslam et al., 2022)	Total rows of data = 15420 Ratio = 14497 legit: 923 frauds	No stated	- SVM - LR - NB	- Accuracy - Precision - Recall - F1-Score - Sensitivity - Specificity
(Caruana & Grech, 2021)	Total rows of data = 15420 Ratio = 14497 legit: 923 frauds	Undersampling with a 70:30 ratio	- NB - ANN	- Accuracy - AUC - Precision - Sensitivity F1 Score
(Harjai et al., 2019)	Total rows of data = 15420 Ratio = 14497 legit: 923 frauds	SMOTE	- RF - SVM - MLP - DT	- TP Rate - FP Rate - Precision - Recall - F-Measure Roc Area

Table 2.1: continued

Authors	Imbalanced dataset ratio	Data sampling method	Machine Learning algorithms	Evaluation Metric
(Yankol-Schalck, 2022)	Total rows of data = 183804 Ratio = 182393 legit: 1411 frauds	No stated	- NN - GB	- Precision - Recall - F1-Score - FDR - AUC - AUC-PR - Log loss Brier score
(Dilkhaz Y, 2021)	No stated	Data sampling	- J48 - RF - NB - Bayes Net Random Tree	- Accuracy - Precision - Recall - F1-score MCC
(Nur Prasasti et al., 2020)	Total rows of data = 1881 Ratio = 1849 legit: 32 frauds	SMOTE	- MLP - RF - DT	- Accuracy - Sensitivity - Specificity AUC

Table 2.1: continued

Authors	Imbalanced dataset ratio	Data sampling method	Machine Learning algorithms	Evaluation Metric
(Moon et al., 2019)	Total rows of data = 15420 Ratio = 14497 legit: 923 frauds	Stratified Random Sampling to 500:500 data	- LR - RF - LASSO - SVM	- Accuracy - Sensitivity - Specificity - PPV - NPV AUC
(Sundarkumar et al., 2015)	Total rows of data = 15420 Ratio = 14497 legit: 923 frauds	Undersampling	- SVM - LR - DT - Group Method of Data Handling Neural Network	- Accuracy - Sensitivity - Specificity AUC
(Panigrahi & Palkar, 2018)	Total rows of data = 28,994	No stated	- KNN - DT - RF NB	- Accuracy - Precision Recall

Table 2.1: continued

Authors	Imbalanced dataset ratio	Data sampling method	Machine Learning algorithms	Evaluation Metric
(Rukhsar et al., 2022)	No stated	No stated	<ul style="list-style-type: none"> - MLP - SVM - Linear Regression - AdaBoost - NB - RF - DT - KNN 	Best performance model = Decision Tree

2.3 Imbalanced Dataset Solutions

2.3.1 Sampling

The study carried out by (Choirunnisa & Lianto, 2018) mentioned that an imbalance of data is commonly found in various kinds of data sources. Therefore, the authors proposed to solve the imbalanced dataset problem by using the hybrid sampling method of undersampling and oversampling. In the research done by the authors, five datasets collected from the Keel Data Repository have a high imbalance ratio of 8.79, 3.25, 8.1, 6.38 and 8.6 respectively. In addition, The machine learning models used in the research are the random forest and decision tree. The accuracy, recall, precision and f1-score were used to evaluate the performance of the trained model with applied the proposed hybrid sampling method. As a result, the authors claimed that the performance of using a hybrid

sampling method was increased from 0.1% to 0.4% in the value of accuracy and ROC compared to only applying a single sampling method.

The study from (Ependi1 et al., 2023) proposed a new hybrid sampling method by using SMOTE to oversample the minority class and using the undersampling to undersample the majority class. In the research, a total of twelve imbalanced dataset was gathered from KEEL and the collected datasets had an imbalanced ratio from 1 to 9. Other than using the proposed hybrid sampling method, the authors also implemented the SMOTE and TOMEKLINK to compare it. There were two models which are NB and DT used in the research and the performance was evaluated by using accuracy and F1-Score. As a result, the proposed hybrid sampling method by the authors achieved the highest predictive performance compared to SMOTE and Tomek Link with higher accuracy and higher F-measure.

From the study (Jonathan et al., 2020), a study was carried out to evaluate the performance of SMOTE, Tomek and SMOTE-Tomek in handling the imbalanced data text that collected from a beauty platform of Female Daily. Two models which are SVM and LR were trained with three data sampling methods to investigate the best data sampling method applied to predict the correct outcome. The evaluation metrics of precision, recall and G-Mean were used to perform the evaluation of the trained model. As a result, the trained SVM model with applied SMOTE-Tomek hybrid sampling methods had the best performance compared to SMOTE and Tomek.

Another study from (Mînaştireanu & Meşniţă, 2020) stated that the performance of traditional machine learning models was affected by the highly imbalanced dataset. The three methods which are the resampling method of undersampling and oversampling, tree algorithms and cost-sensitive training were proposed by the authors to solve the present problem of imbalanced datasets in financial credit card fraud detection. For the

evaluation, the evaluation metrics used by the author to evaluate the trained model in the research were recall, precision, F1-score and AUC. As the result of comparing between three proposed methods to handle an imbalanced dataset, the oversampling by using SMOTE techniques comes out with the best performance.

Another study by (Swana et al., 2022), the research carried out to compare the performance between the sampling method of SMOTE, Tomek Link and the combination of SMOTE and Tomek in handling the imbalanced dataset problem. The dataset used was an imbalanced dataset in the research. Moreover, there were three models which are Naïve Bayes Classification, SVM and KNN applied to evaluate the performance of each sampling method applied. The evaluation metrics used to determine the performance of the trained model were accuracy, precision, recall and F1-score. For the result, three sampling methods had improved the performance of all trained classification models.

Table 2.2: Summary of Literature Review of Hybrid Sampling Method to Handle Imbalanced Dataset

Authors	Problem	Solution	Result / Finding
(Choirunnisa & Lianto, 2018)	Five datasets collected were highly imbalanced with an imbalance ratio lower than 9	Hybrid sampling with the combination of undersampling and oversampling	The proposed hybrid sampling method increases the accuracy and AUC value from 0.1% to 0.4%

Table 2.2: continued

Authors	Problem	Solution	Result / Finding
(Ependil et al., 2023)	Twelve imbalanced datasets gathered from KEEL had an imbalance ratio lower than 9	- Hybrid sampling with SMOTE and undersampling - SMOTE Tomek Link	Hybrid sampling had better performance compared to SMOTE and Tomek Link
(Jonathan et al., 2020)	The imbalanced dataset collected from the Female Daily beauty platform	- SMOTE-Tomek - SMOTE Tomek Link	SMOTE-Tomek hybrid sampling method outperformed compared to SMOTE and Tomek Link
(Mînaştireanu & Meşniţă, 2020)	The financial credit card fraud dataset faced the issue of an imbalanced dataset	- Undersampling with SMOTE - Oversampling - Tree algorithms Cost-sensitive Learning	Oversampling with SMOTE performed better compared to other solution

Table 2.2: continued

Authors	Problem	Solution	Result / Finding
(Swana et al., 2022)	Imbalanced dataset issues found in the machine data	- SMOTE - Tomek Link - Combination and SMOTE and Tomek	All applied sampling methods show improvement in the performance of the trained model

2.3.2 Ensemble Learning

The study by (Liu et al., 2020) proposed to solve the issues of the imbalanced dataset for the medical diagnosis domain. There are three phases proposed by the authors which are data pre-processing, model training with base classifier and model training with ensemble classifier to solve the imbalanced medical diagnosis dataset issues. SMOTE had been applied by authors to resample the dataset before the dataset was used for modelling. The ensemble support vector machine proposed by the authors was used in the research. There were a total of 12 previous study paper results used by authors to compare the proposed ensemble support vector machine model. As a result, by comparing with the 12 previous work results, the proposed ensemble support vector machine model achieved the better results and showed the effectiveness of the proposed ensemble support vector machine model used in dealing with the imbalanced dataset of the medical diagnosis domain.

One of the studies (Malhotra & Jain, 2020) was carried out to predict the software defect by using an ensemble learning method to handle the imbalanced dataset. In the

field of software defection, the authors mentioned that the software defect datasets in the real world were imbalanced which consisting of the majority data being defect-free and minority data of defect. There were seven methods proposed by the authors to carry out research. Next, each method is evaluated with the G-Mean, Balance, AUC and sensitivity. The result showed that RUSB outperformed compared to all trained models in terms of G-Mean, Balance, AUC and sensitivity. Precision, recall, f-measure, G-mean and AUC were used as the evaluation metric to evaluate the proposed model.

The study from (Sanabila & Jatmiko, 2018) proposed to apply ensemble learning to handle the imbalanced data in the financial data. There are a total of five models which consisting of 3 ensemble learning models and 2 base machine learning models. The 3 ensemble learning models were Boosting, Bagging and Random Forest while the 2 base machine learning models were naive Bayes and logistic regression. In the data preprocessing phase, the feature selection, data cleaning and SMOTEENN were used before the dataset was used for modeling. The evaluation metrics used to evaluate the performance of each model are precision, recall, specificity and f1-score. As a result, the ensemble learning of bagging had a better performance compared to other trained models.

According to the research done by (Grzyb & Woźniak, 2022), the authors stated that by using an ensemble learning model able to solve the imbalanced dataset issue. In the research, three ensemble models of Support Vector Machine with application multi-objective optimization selection, multi-objective optimization selection with bootstrapping and multi-objective optimization selection with bootstrapping and pruning were proposed by the authors to handle the imbalanced dataset. There were 78 imbalanced datasets were used in the research. The evaluation metrics used to evaluate the performance of the proposed method were BAC, g-mean, recall, and precision. Among the three proposed methods by the authors, Support Vector Machine with application

multi-objective optimization selection had the best performance by comparing another two proposed ensemble models.

According to the previous study done by (Salunkhe & Mali, 2016), an imbalanced training dataset caused the performance of the classification model to degrade. The authors also mentioned that the common method which is resampling that functions to resample the imbalanced data might cause removed necessary data during the resampling process and cause overfitting. Therefore, the authors proposed an approach to apply the data sampling method which is oversampling and undersampling to the imbalanced dataset before the dataset is used as the training dataset to train the ensemble learning model. The ensemble learning models used in the project were Bagging and StackingC. There are eight imbalanced datasets collected to examine the performance proposed method by the author. For the model evaluation part, the author plots out the confusion matrix in order to use the data from the confusion matrix to calculate the AUC, specificity and sensitivity for the trained model. As a result, the proposed approach outperformed other trained models in terms of AUC value.

Table 2.3: Summary of Literature Review of Ensemble Learning Method to Handle Imbalanced Dataset

Authors	Domain	Problem	Solution	Result / Finding
(Liu et al., 2020)	Healthcare: Medical Diagnosis	The dataset used in the medical diagnosis domain is highly imbalanced.	Application of proposed ensemble support vector machine and SMOTE applied to resample the imbalanced dataset	By comparing with the result of 12 previous works, the proposed ensemble support vector machine showed effectiveness in solving the imbalanced dataset issues.

Table 2.3: continued

Authors	Domain	Problem	Solution	Result / Finding
(Malhotra & Jain, 2020)	Technology: Software Defect	The imbalance data which consists of majority data of defect-free and minority data of defect	Application of ensemble learning model to solve the issues and application of the combination of ensemble model with sampling	Random Undersampler Boosting outperformed compared to other trained model.

Table 2.3: continued

Authors	Domain	Problem	Solution	Result / Finding
(Sanabila & Jatmiko, 2018)	Financial	The exists of issue of imbalanced large-scale financial data	Application of three ensemble models of Boosting, Bagging and Random Forest and two base machine learning models of Naïve Bayes and Logistic Regression Application of SMOTEENN to resample the dataset	Ensemble learning of bagging had a better performance compared to other trained model.

Table 2.3: continued

Authors	Domain	Problem	Solution	Result / Finding
(Grzyb & Woźniak, 2022)	General	Highly imbalanced dataset found in 72 collected datasets	Application of SEMOOS, SEMOOSB, and SEMOOSBP ensemble learnings to solve the issues	SEMOOS has the best performance among the proposed method
(Salunkhe & Mali, 2016)	General	Eight collected datasets show in high imbalanced ratio	Used ensemble learning model of Bagging and Stacking with applying data sampling method	The proposed approach by the author overperforms other trained models in terms of AUC.

2.4 Evaluation Metric

In this project, the classification models which are Decision Tree, Naïve Bayes, KNN, MLP and LR were trained to classify the fraud and non-fraud target values. From the previous similar domain research study, it came out with the four evaluation metrics that are commonly used to evaluate the machine learning models trained to detect vehicle fraud insurance claims. The four metrics which are accuracy, recall, specificity and F1-Score were considered to access the trained models in this project.

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Figure 2.1: Confusion matrix's format (Kulkarni et al., 2020)

According to (Kulkarni et al., 2020), the confusion matrix is the table used to evaluate the classification model. In this project, all the trained models are the binary classification model as all models were classifying the outcome between Fraud and Non-fraud.

Table 2.4: Formula and description of evaluation metric

Type of evaluation metric	Calculation formula	Description
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$	The measure of the correctness of the classification model predicted

Table 2.4: continued

Type of evaluation metric	Calculation formula	Description
Recall (Also known as Sensitivity or True Positive Rate)	$TP / (TP + FN)$	The measure of the positive instance that was predicted correctly out of all actual positive predicted instances.
Specificity (Also known as False Negative Rate or selectivity)	$TN / (TN + FP)$	The measure of the negative instances that were predicted correctly out of all negative instances that were predicted.
F1-Score	$2 * (Precision * Recall) / (Precision + Recall)$	The measure of the harmonic means between precision and recall

2.5 Summary

From the literature review that was carried out on the similar domain, it was observed that most of the automobile fraud insurance claim dataset is an imbalanced dataset. However, there are still major of the models done by the researchers do not apply and only apply one imbalanced dataset handling method. With the research paper with applying the single sampling method, there was only apply fixed ratio on it and did not determine other ratios of sampling to sample the data. In addition, from the research paper that did not apply the imbalanced dataset handling method, the researchers have

mentioned that for future work can add in the imbalanced dataset handling methods to improve the performance of the model. Therefore, the literature review has provided a piece of important information that imbalanced datasets must be handled with imbalanced dataset handling techniques to improve the performance of the model to predict fraud.

In addition, from the literature review, the common machine learning models by the researchers in the vehicle insurance fraud detection domain that are used to train the models for prediction fraud such as Decision Tree, Naïve Bayes, KNN, MLP and Logistic Regression were found, and the machine learning algorithms selected to be used as the algorithms to train the machine learning model in this project.

Moreover, the solution to handle the imbalanced dataset is identified through the literature review. The solutions to handle the imbalanced dataset including the Oversampling, Undersampling, Hybrid Sampling and Ensemble Learning were found and will be applied in this project to handle the imbalanced vehicle fraud insurance dataset.

Last but not least, the literature review was also carried out to study the common evaluation metrics to evaluate the imbalanced dataset of the trained model. From the study, recall is important as it accesses the model to predict fraud cases. The accuracy, recall, specificity and F1-Score were studied through a literature review and will be applied in this project to assess the performance of the trained model.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Research Design

In this research, a better performance model to detect vehicle insurance fraud was targeted to be developed. The study from (Aslam et al., 2022) will be used as the benchmark to compare and evaluate the performance of each trained machine learning model. From the study (Aslam et al., 2022), the authors came out with the best performance model of Linear Regression which has the accuracy of 0.8802, precision of 0.2229, recall of 0.4225, F1-Score of 0.2627, and specificity of 0.9093. In the study, the authors performed the pre-processing stage by only applying the feature selection method and did not handle the imbalanced dataset.

To ensure that the research was carried out successfully, the project was carried out by dividing into four main processes which are data understanding, data preparation, modeling and evaluation. Python has been used throughout the four main processes to come out with vehicle insurance fraud detection models.

3.2 Data Understanding

For the data understanding process, the dataset used for this project was collected from Kaggle. The dataset collected from Kaggle was same as the dataset used by (Aslam et al., 2022) to carry out the research. After the dataset was collected, the Exploratory Data Analysis process was carried out to explore and understand more about the dataset.

The collected dataset was observed, and it consists of 15420 rows and 33 columns of data and the dataset is labeled data.

Table 3.1: Data description for the dataset

Column Name	Description	Data Type
Month	Represent the month of accident occurred	Object
WeekOfMonth	Represent the week of accident occurred	Integer
DayOfWeek	Represent the day of accident occurred	Object
Make	Represent the car manufacturer	Object
AccidentArea	Represent the area where of accident occurred	Object
DayOfWeekClaimed	Represent the day when the car insurance claimed	Object
MonthClaimed	Represent the month where the car insurance claimed	Object
WeekOfMonthClaimed	Represent the week when the car insurance claimed	Integer
Sex	Represent male or female	Object
MaritalStatus	Represent the person who is divorce, widow, married or single	Object
Age	Represent the person's age	Integer
Fault	Represent the accident caused by policyholder or third party	Object
PolicyType	Represent the policy applied	Object
VehicleCategory	Represent the category of cars in sport, sedan and utility	Object
VehiclePrice	Represent the price range of car	Object
FraudFound_P	Is the target variable indicating a fraud case	Integer
PolicyNumber	Represent the indexing for each row	Integer

Table 3.1: continued

RepNumber	Represent the assigned number	Integer
Deductible	Represent the amount deduct for the claim	Integer
DriverRating	Represent the customer experience	Integer
Days_Policy_Accident	Represent the days which	Object
Days_Policy_Claim	Represent the days which can be claimed for the policy	Object
PastNumberClaims	Represent the previous claim number	Object
AgeOfVehicle	Represent the vehicles manufacturing year	Object
AgeOfPolicyHolder	Represent the age of policy holder	Object
PoliceReportFiled	Represent the whether the accident report to police	Object
WitnessPresent	Represent the present of witness at accident area	Object
AgentType	Represent the internal and external agent	Object
NumberOfSuppliments	Represent the extra spend out from the estimation	Object
AddressChange_Claim	Represent whether the policyholder change address	Object
NumberOfCars	Represent number of cars for policyholder	Object
Year	Represent the year of claimed	Integer
BasePolicy	Represent the base policy used for insurance company	Object

Moreover, the dataset is a highly imbalanced dataset which has 14496 non-fraud data and 923 fraud data. The imbalanced ratio of the dataset has achieved 15.7 which is calculated by the formula of the count of the majority class divide by the count of the minority class.

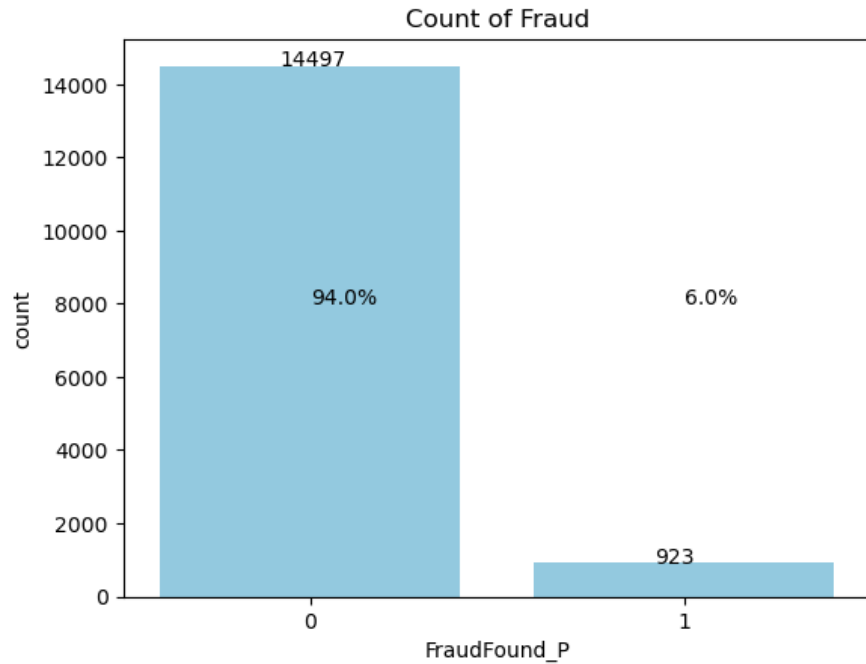


Figure 3.1: Bar chart for the count of target variable of “FraudFound_P”

Moreover, the collected dataset is further studied to understand that the distribution of each class to the target variable of “FraudFound_P”.

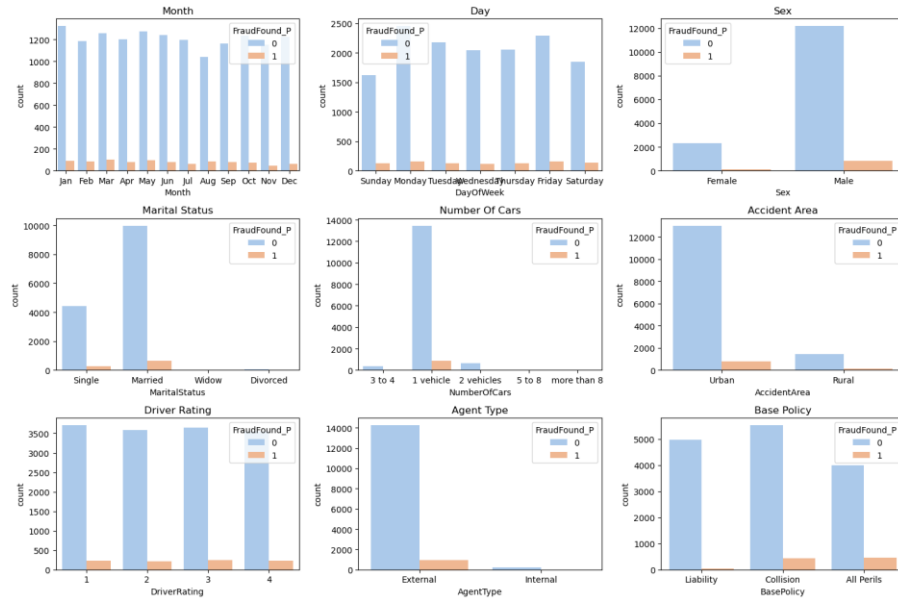


Figure 3.2: Bar chart of distribution for each class to the target variable of “FraudFound_P”

In the EDA phase, it also identified that the column of “Age” contains of value of 0. In addition, it was found that the column of “DayOfWeekClaimed” and “MonthClaimed” contained the value of 0. Both of two issues found on the dataset was solved in the data preparation stage.

3.3 Data Preparation

In the data preparation phase, there are several major steps performed which are data cleaning, feature selection, data transformation, data splitting and data sampling. First of all, the data cleaning will be carried out. The duplicate data and null data were checked on the dataset. The first issue about the “Age” is replaced by the value of 16.5 and the second issue about the dataset was solved by removing the row. Next, the feature selection part was carried out by using the chi-squared test to assess the column and determine the association between two categorical columns.

After the data cleaning and feature selection process was done, the data transformation process was carried out. In the data transformation process, the “OrdinalEncoder” and “OneHotEncoder” were used to handle the categorical data and transform the categorical data into numeric values. “OrdinalEncoder” was used to transform the ordinal column into a numerical value while the “OneHotEncoder” was used to transform the binary column into a numerical value. Both “OrdinalEncoder” and “OneHotEncoder” are the libraries that are available in Python.

Before proceeding with the modelling phase, the data splitting process was taken. By referencing to the previous similar domain works carried out for automobile insurance fraud detection, the majority of the past works train the model with the data splitting ratio of 80:20. However, there are three sets of data splitting that have been working which are 60:40, 70:30 and 80:20 and determined that which data splitting ratio will be the most suitable and have the best performance. Each data splitting ratio has been trained with five models which are Decision Tree, KNN, Naïve Bayes MLP and LR to find out the best-performed data splitting ratio.

Table 3.2: Performance of 5 base models with using three data splitting ratios of 40:60, 70:30 and 80:20

Data Splitting Ratio	Algorithm Used	Accuracy	Recall (TPR)	Specificity (TNR)	F1-Score
40:60	Decision Tree	0.8906	0.2412	0.9319	0.2087
	KNN	0.9275	0.0569	0.9829	0.0859
	Naive Bayes	0.3234	0.9350	0.2845	0.1419
	MLP	0.9399	0	0.9997	0
	LR	0.9402	0	1	0
70:30	Decision Tree	0.8967	0.2671	0.9377	0.2364
	KNN	0.9303	0.0867	0.9841	0.1297
	Naive Bayes	0.5696	0.8953	0.5489	0.1994
	MLP	0.9401	0	1	0
	LR	0.9401	0	1	0
80:20	Decision Tree	0.8889	0.3189	0.9251	0.2560
	KNN	0.9313	0.0811	0.9855	0.1240
	Naive Bayes	0.5778	0.8865	0.5581	0.2012
	MLP	0.94	0	1	0
	LR	0.94	0	1	0

From the table above, it showed that the performance of 5 models with data splitting of 80:20 has better overall performance compared to the 40:60 and 70:30 data splitting ratio. The splitting ratio of 90:10 has not exanimated as the model should have mode training dataset to evaluate of the performance. Therefore, the data splitting ratio of 80:20

was chosen and it applied to the used prepared dataset to split it into 80% of the data used as the training data and the rest of 20% of the data used as testing data. After data splitting, the training dataset was preprocess by applying the sampling methods to solve the imbalanced dataset. There are three sampling techniques applied to solve the imbalanced training data. The sampling methods are:

- Undersampling with RandomUndersampler for the majority class from 0.1 to 0.8
- Oversampling with SMOTE for minority class from 0.1 to 1
- Hybrid sampling = Combination of Undersampling with RandomUndersampler from 0.1 to 0.8 and Oversampling with SMOTE for minority class from 0.1 to 1

3.4 Modelling

In the modelling phase, there are five machine learning methods were applied which are Decision Tree, KNN, Naïve Bayes, MLP and Logistic Regression. The five selected machine learning algorithms are the commonly used machine learning algorithms in the vehicle fraud insurance domain. Firstly, five models were trained with the training dataset without handling the imbalanced dataset problem and the performance was evaluated with the testing dataset respectively. After that, the Decision Tree, Naïve Bayes and Logistic Regression were further applied with the three sampling methods and the performance of the model with applying data sampling methods was evaluated. For the next step, the Decision Tree algorithm was used as the base learner for ensemble learning of boosting and bagging. The boosting algorithm used was the XGBoost and the Bagging algorithm used was the Bagging. The performance of each ensemble learning model was evaluated with the testing dataset and the result was recorded.

3.5 Evaluation

In the evaluation phase, all the trained models are evaluated. The confusion matrix will be visualized for each model. After the confusion matrix is built, the performance metrics which are Accuracy, Recall, Specificity and F1-Score were calculated by using the confusion matrix. After all models have been evaluated, the best performance of the model by applying the imbalanced dataset handling method will be identified. Moreover, the best performance model was compared with the Logistic Regression model proposed by (Aslam et al., 2022).

3.6 Workflow Diagram

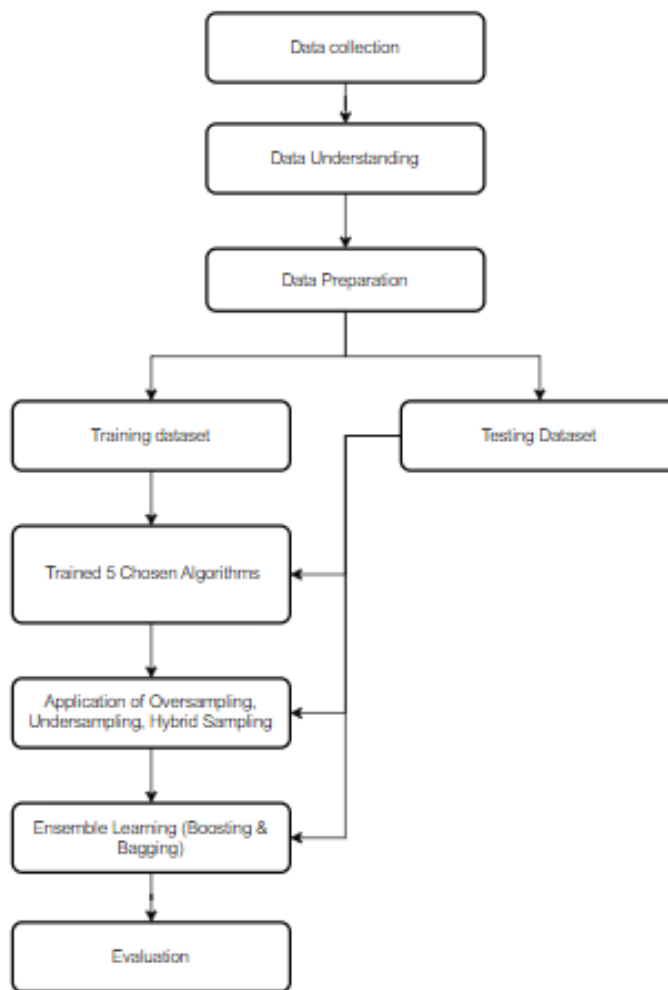


Figure 3.3: Workflow Diagram

Figure 3.3 shows the workflow diagram that will be carried out in this project. The first process carried out is the data collection followed by data understanding. In data understanding, a deep explore is conducted to the dataset to understand more about the dataset. Then the data preparation process is carried out by splitting the dataset into 80% of the training dataset and 20% of the testing set. The training model is firstly trained with the 5 base models and evaluate using the testing dataset. After three models which are NB, LR and DT applied with the data sampling method and trained again. The ensemble learning model of boosting and bagging by using the Decision Tree as base model are also trained. Lastly, all trained model are evaluated and the best performance model was identified.

CHAPTER 4: RESULT AND DISCUSSION

4.1 Without Applying Data Sampling Techniques

First and foremost, the training dataset that without applying the data sampling technique was used to train the selected five machine learning algorithms. The metrics that were used to evaluate the performance of trained models are accuracy, recall, specificity and F1-Score. The table below shows the results of trained models with the evaluation metrics.

Table 4.1: Classification result of five trained models without using any data sampling method compared with (Aslam et al., 2022)

Algorithm Used	Accuracy	Recall (TPR)	Specificity (TNR)	F1- Score	Method
Linear Regression (Aslam et al., 2022)	88.02%	42.25%	90.93%	29.27%	Feature Selection
Decision Tree	88.89%	31.89%	92.515	25.6%	-
KNN	93.13%	8.11%	98.55%	12.4%	-
Naive Bayes	57.78	88.65%	55.81%	20.12%	-
MLP	94%	0%	100%	0%	-
LR	94%	0%	100%	0%	-

In Table 4.1., the highlighted rows data is the best-performing model which is the Linear Regression model proposed by (Aslam et al., 2022). The best-performing model by (Aslam et al., 2022) was used to compare with the trained models in this project. The compared results from Table 4.1 show that all trained models without applying data

sampling techniques were uncompetitive with the result provided by (Aslam et al., 2022). Therefore, further techniques which are the three data sampling methods will be applied to achieve a performance that comparative to the results from (Aslam et al., 2022).

Among the five trained models, Naïve Bayes and Decision Tree models have the higher recall of 88.65% and 31.89% respectively which indicates that the two models have a better ability to classify correctly for the actual positive class which is class 1 represented for Fraud. Therefore, these two models were selected to apply different sampling methods and tried to get better results. On the other hand, MLP and LR were showed that the recall value of 0 as both models trained with an imbalanced dataset caused the models to fail to classify correctly for the minority class. Therefore, the LR was also selected to further experiment by applying different sampling methods to get the better results. There were three models which are Decision Tree, Naïve Bayes and LR selected for further experimentation with applied different data sampling methods.

4.2 Oversampling (SMOTE)

The oversampling method which is SMOTE was applied in the training dataset to oversample the minority class (1: Fraud). The oversampled training dataset was used to train the models and the trained models were evaluated with the testing dataset. The results of applied oversampling in different ratios were recorded and shown in the figures below:

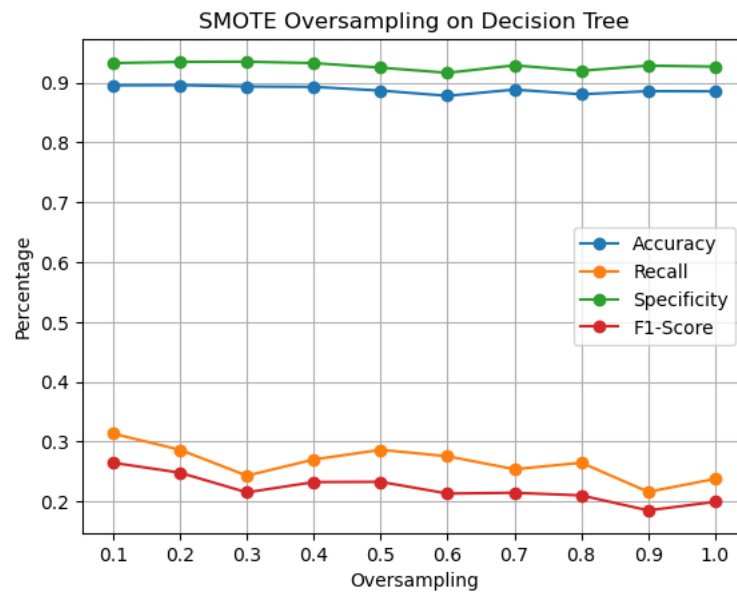


Figure 4.1: Results of Decision Tree by applying SMOTE

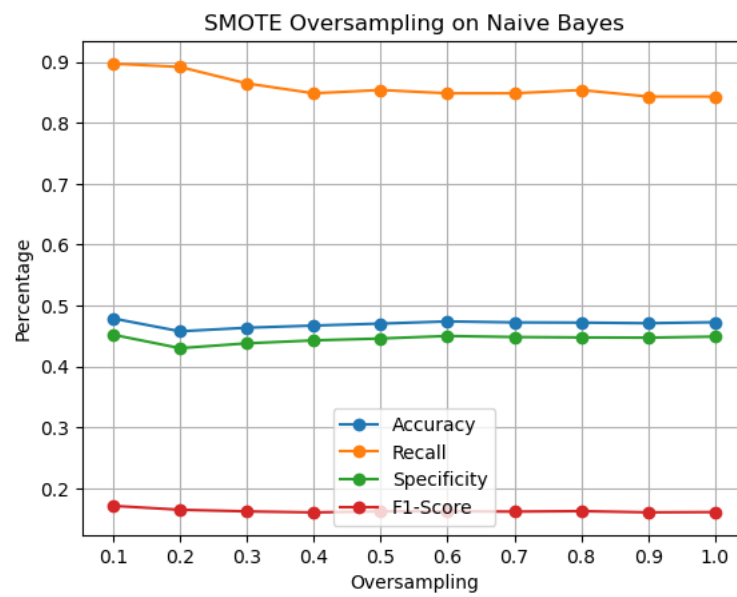


Figure 4.2: Results of Naïve Bayes with applying SMOTE

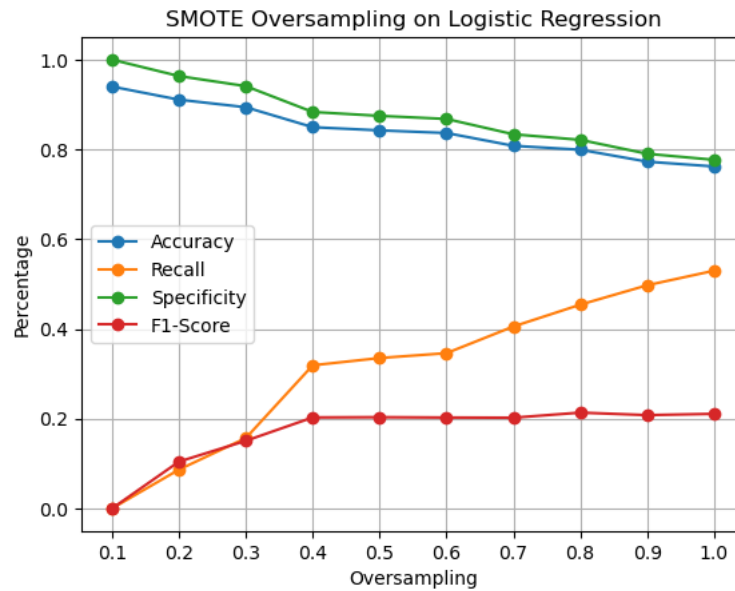


Figure 4.3: Result of Logistic Regression with applying SMOTE

From the Figures above, it shows that applied SMOTE oversampling improved the recall of three models compared to these three models that were trained before without using any data sampling method. With the better recall after applying SMOTE oversampling, it indicated that the models have better performance in classifying the minority class which is the class of 1 that represents “Fraud”. For Decision Tree and Naïve Bayes, the maximum recall achieved respectively which are 31.35% for Decision Tree and 89.73% for Naïve Bayes with oversampling of 0.1 and the recall continuously shows up and down trend with increasing of oversampling ratio. However, the Logistic Regression model showed that the continuously increased the recall with applied oversampling meanwhile the accuracy and specificity continuously decreased. The F1-Score of the Linear Regression model showed an uptrend when the oversampling ratio increased while the F1-Score of the Decision Tree and Naïve Bayes decreased when the oversampling ratio increased.

4.3 Undersampling

After the Oversampling applied to the three models, the Undersampling method which is random under sampler was applied in the training dataset to undersample and reduce the number of majority class of 0 that represent non-fraud. The undersampled training dataset has proceeded to be used in training the models. After the models were trained, each trained model was evaluated with the testing dataset. The results of applied oversampling were recorded and shown in the figure below.

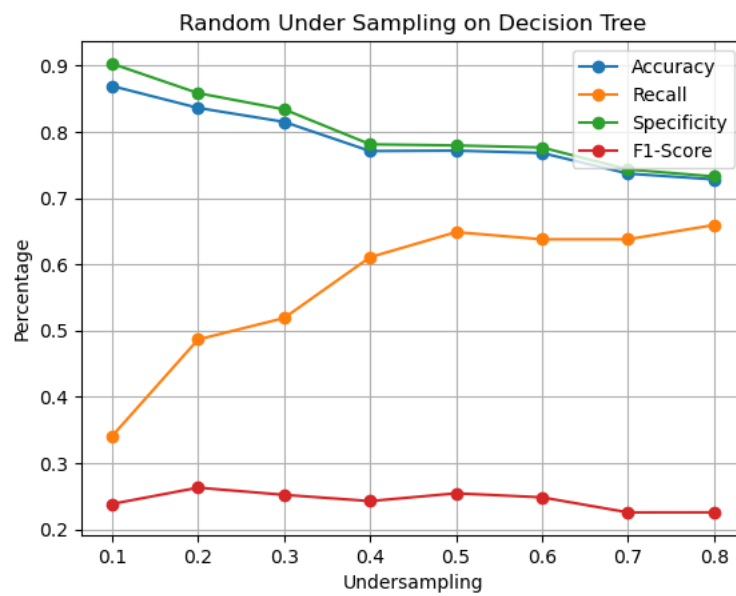


Figure 4.4: Result of Decision Tree with applying Undersampling

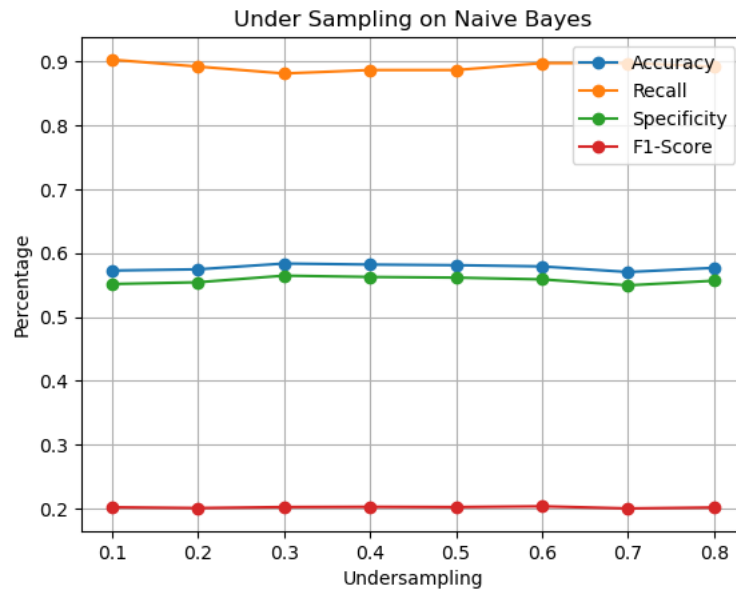


Figure 4.5: Result of Naïve Bayes with applying Undersampling

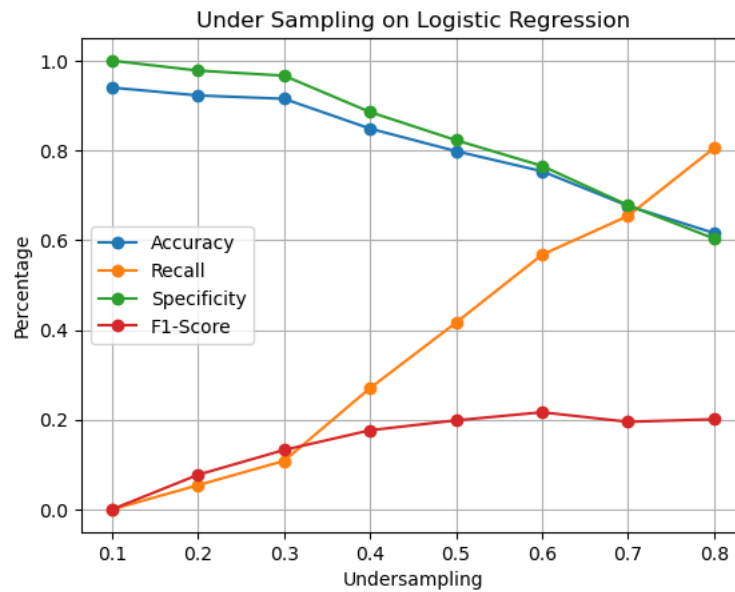


Figure 4.6: Result of Logistic Regression with applying Undersampling

From 4.4, it showed that the recall is increased when the undersampling ratio increased. The highest recall value for Decision Tree achieved the maximum recall which is 68.95% when undersampling with a 0.8 ratio.

From Figure 4.5, it showed that Naïve Bayes with applied undersampling had no big changed in terms of accuracy, recall, specificity and F1-Score. The maximum recall achieved for the Naïve Bayes is 90.27% when undersampling with the ratio of 0.1.

From Figure 4.6, it obviously shows that the recall of Logistic Regression increases when the undersampling ratio increases. The best results of Logistic Regression achieved with the highest recall is undersampling with a 0.8 ratio which has 61.58% accuracy, 80.54% recall, 60.37 specificity and 20.09% F1-Score.

From the Figure above, it shows that applied undersampling had also improved the recall of three models compared to these three models that were trained before without using any data sampling method. The Naïve Bayes model only had slight improvement in the recall while the Decision Tree and Logistic Regression had the most significant improvement for the recall when applying the undersampling. For Decision Tree and Logistic Regression models, the accuracy and the specificity were decreased while the recall increased.

4.4 Hybrid Sampling

The hybrid sampling method which is the combination of Oversampling with SMOTE and Undersampling with RandomUndersampler was also used to resample the imbalanced training dataset. SMOTE functions to oversample the minority class from 0.1 to 1 while RandomUndersampler functions to undersample the majority class from 0.1 to 0.8. The obtained results of the application of hybrid sampling are illustrated below.

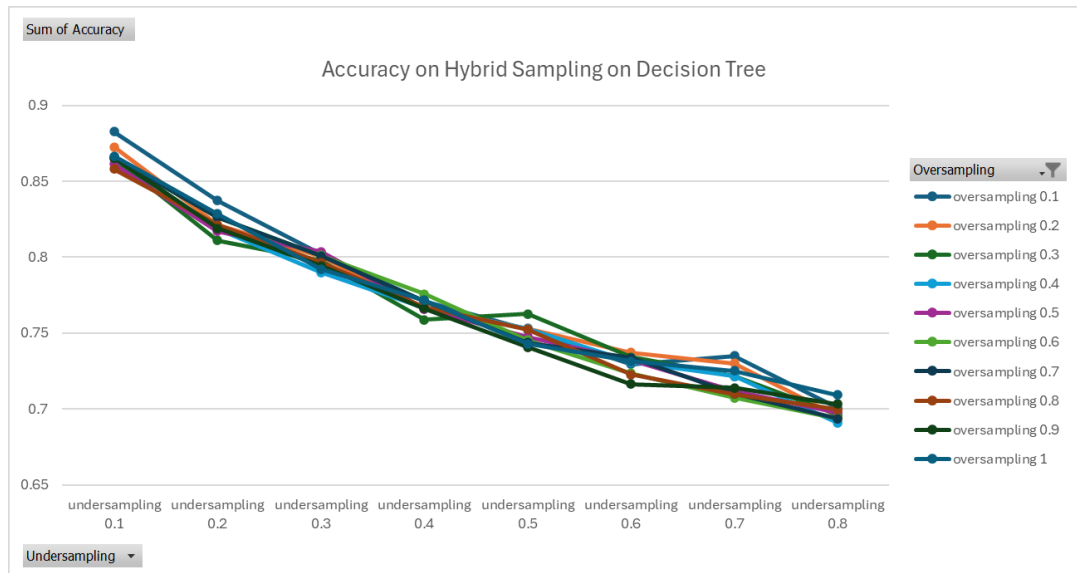


Figure 4.7: Accuracy result of Decision Tree with applying Hybrid Sampling

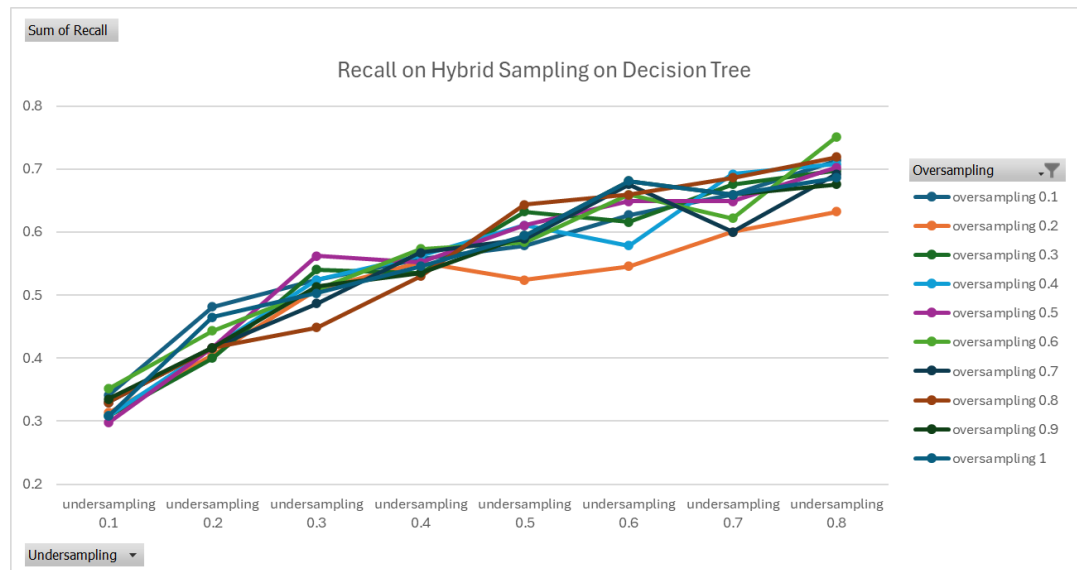


Figure 4.8: Recall result of Decision Tree by applying Hybrid Sampling

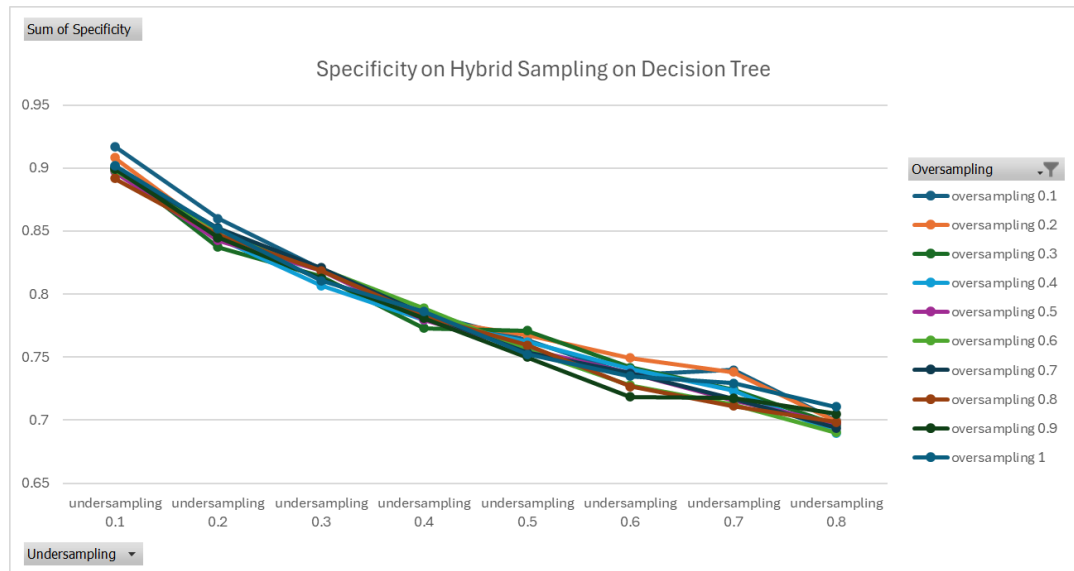


Figure 4.9: Specificity result of Decision Tree with applying Hybrid Sampling

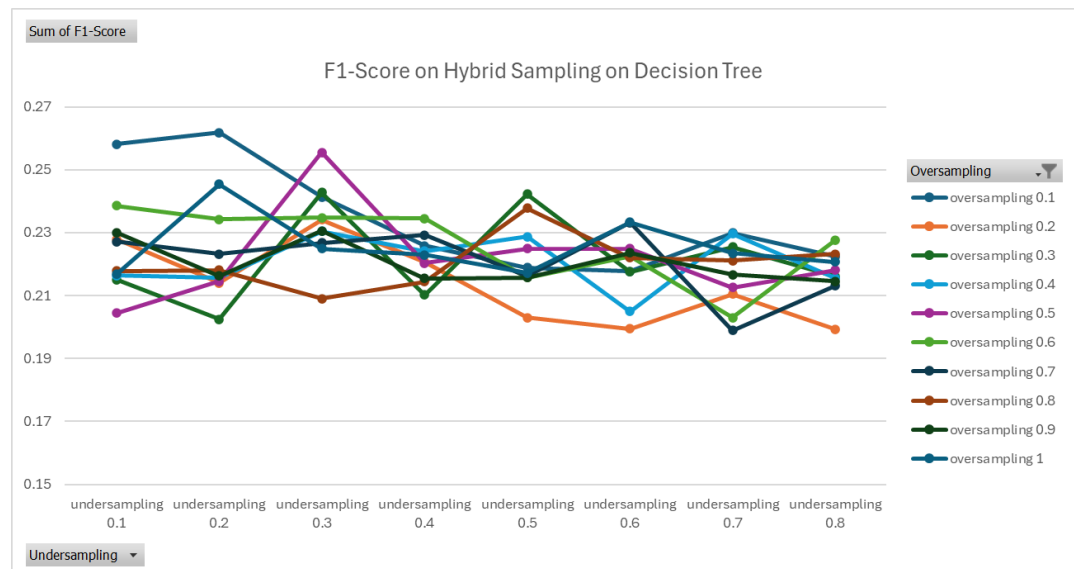


Figure 4.10: F1-Score result of Decision Tree by applying Hybrid Sampling

Figure 4.7 to 4.10 show the graph to explain the performance of applied hybrid sampling to the Decision Tree model. The accuracy and specificity with applied hybrid sampling showed that the values were getting lower when the ratio of undersampling and oversampling increased. Meanwhile, the recall increased when the ratio undersampling

and oversampling increased. The best combination of hybrid sampling is Oversampling 0.6 and Undersampling 0.8 which had 69.39% accuracy, 75.14% recall, 69.02 specificity and 22.75% recall.

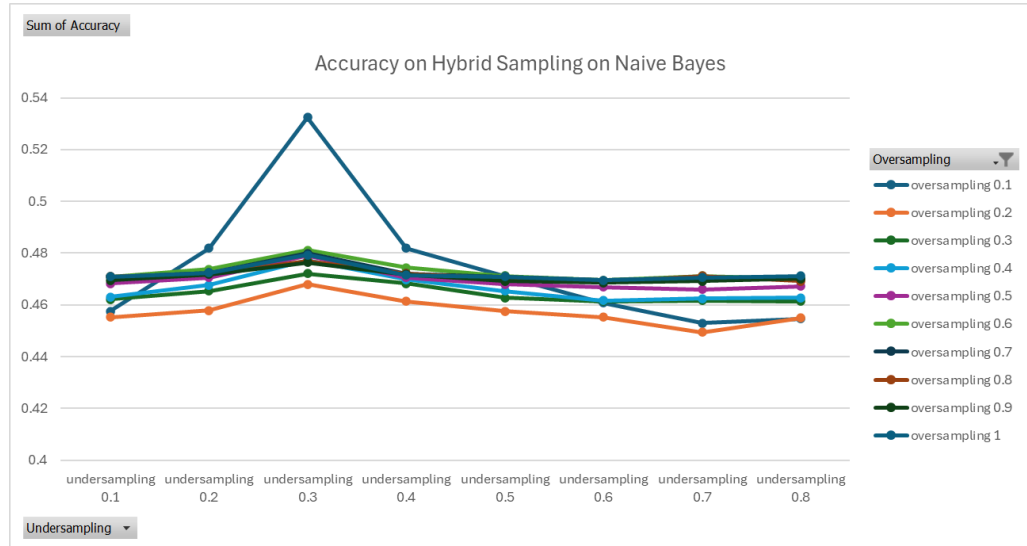


Figure 4.11: Accuracy result of Naive Bayes with applying Hybrid Sampling

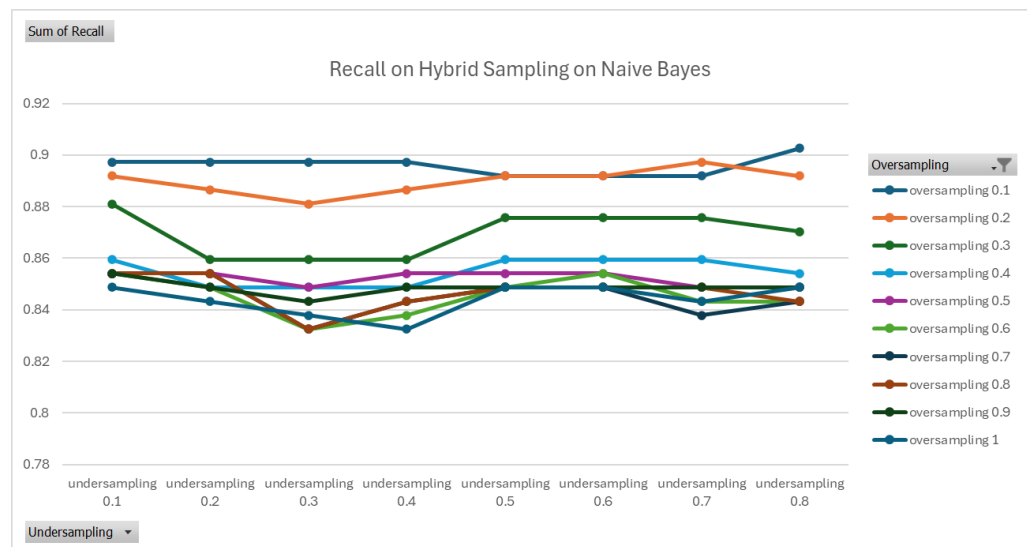


Figure 4.12: Recall result of Naive Bayes with applying Hybrid Sampling

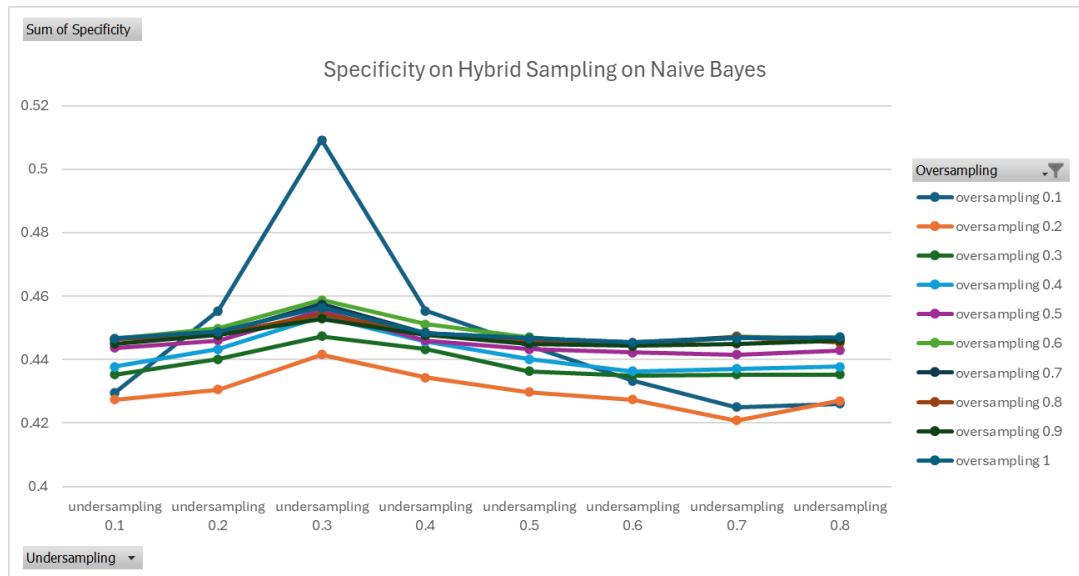


Figure 4.13: Precision result of Naive Bayes with applying Hybrid Sampling

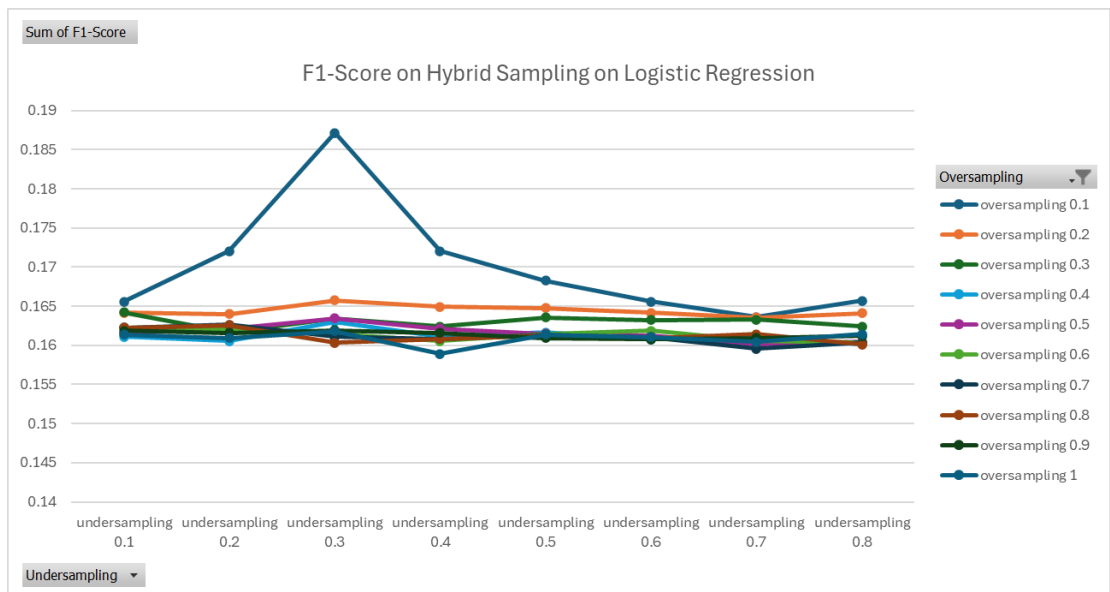


Figure 4.14: F1-Score result of Naive Bayes with applying Hybrid Sampling

The performance of applied hybrid sampling to the Naïve Bayes model in terms of accuracy, recall, specificity and F1-Score was plotted with the graph from Figure 4.11 until Figure 4.14. From Figure 4.12, the oversampling of 0.1 and 0.2 ratio has a higher recall compared to the oversampling 0.3 ratios onwards. The best combination of hybrid

sampling is Oversampling 0.1 and Undersampling 0.8 which had 45.46% accuracy, 90.27% recall, 42.60% specificity and 16.57% recall.

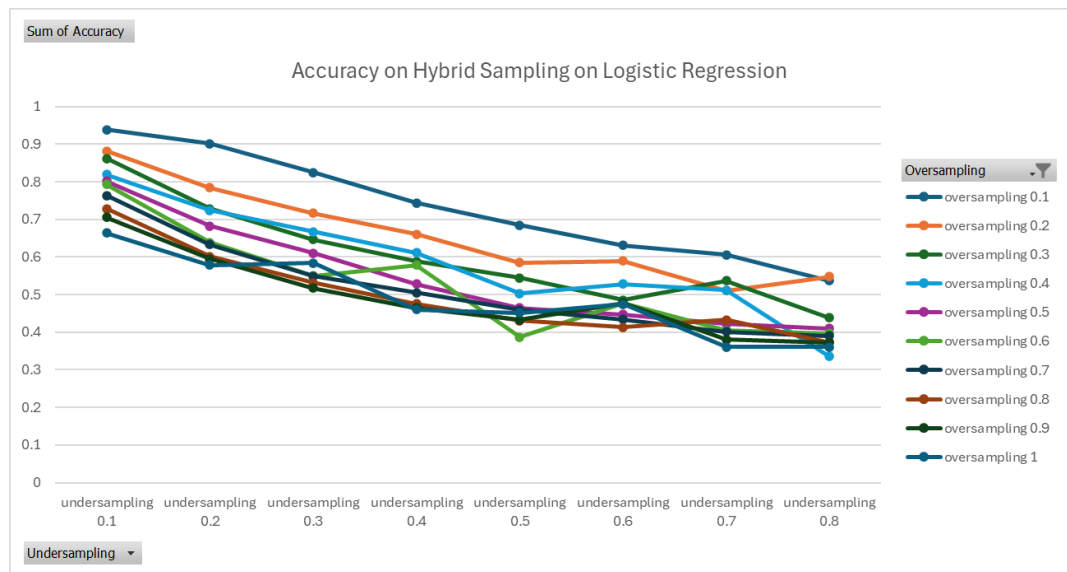


Figure 4.15: Accuracy result of Logistic Regression applying Hybrid Sampling

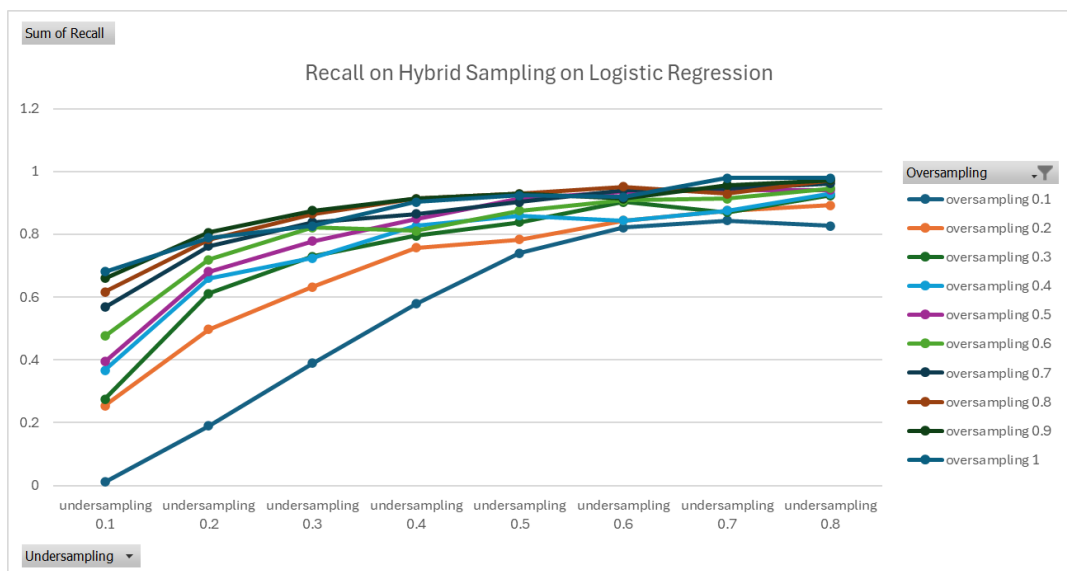


Figure 4.16: Recall result of Logistic Regression with applying Hybrid Sampling

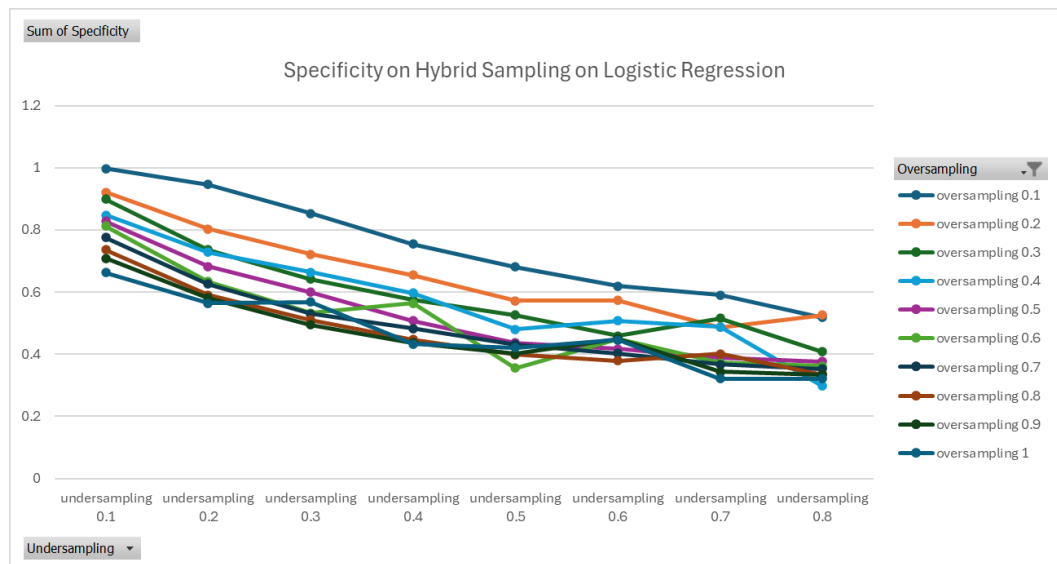


Figure 4.17: Specificity result of Logistic Regression with applying Hybrid Sampling

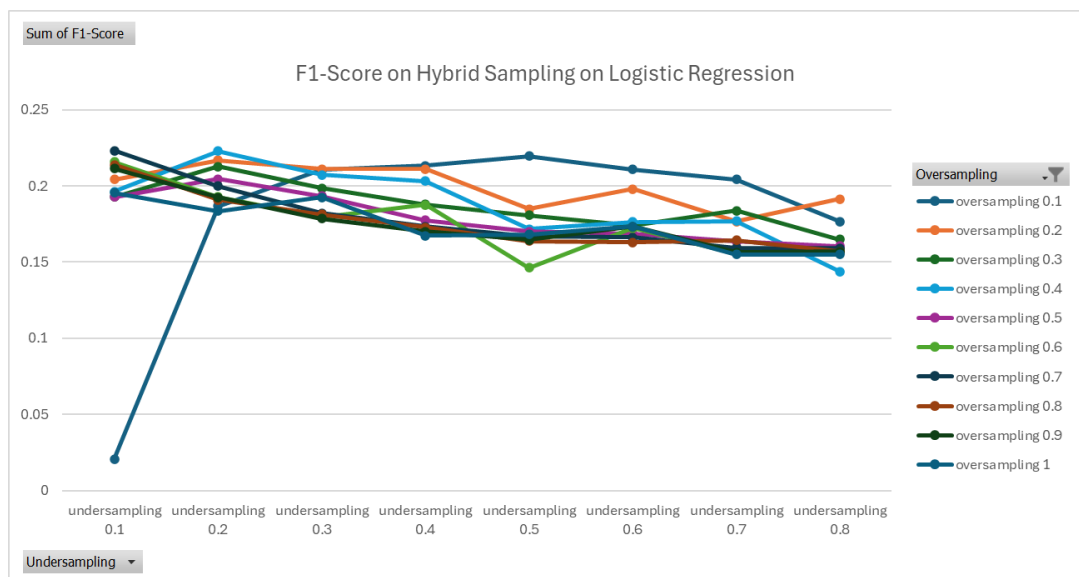


Figure 4.18: F1-Score result of Logistic Regression with applying Hybrid Sampling

Figures 4.15 to 4.18 show the graph to explain the performance of applied hybrid sampling to the Logistic Regression model. When the ratio of undersampling and

oversampling increased, the accuracy and specificity were getting lower but the recall increased when the ratio of undersampling and oversampling increased. The best performance of the Logistic Regression model is the combination of hybrid sampling with 0.6 ratio Oversampling and 0.8 ratio Undersampling which had 69.39% accuracy, 75.14% recall, 69.02 specificity and 22.75% recall.

4.5 Comparison between Oversampling, Undersampling and Hybrid Sampling

Table 4.2: Summary result for models with and without apply data sampling methods

Algorithm Used	Method	Accuracy	Recall (TPR)	Specificity (TNR)	F1-Score
Decision Tree	-	88.89%	31.89%	92.51%	25.60%
	Oversampling (0.1)	89.56%	31.35%	93.27%	26.48%
	Undersampling (0.8)	72.83%	65.95%	73.27%	22.55%
	Hybrid sampling (Over 0.6 + Under 0.8)	69.39%	75.14%	69.02%	22.75%
Naïve Bayes	-	57.78%	88.65%	55.81%	20.12%
	Oversampling (0.1)	47.89%	89.73%	45.22%	17.12%
	Undersampling (0.1)	57.26%	90.27%	55.16%	20.22%
	Hybrid sampling (Over 0.1 + Under 0.8)	45.46%	90.27%	42.60%	16.57%
Logistic Regression	-	94%	0%	100%	0%
	Oversampling (1.0)	76.20%	52.97%	77.68%	21.08%
	Undersampling (0.8)	61.58%	80.54%	60.37%	20.09%
	Hybrid sampling (Over 1.0 + Under 0.7)	36.06%	97.84%	32.11%	15.51%

From Table 4.2, it showed that the Oversampling, Undersampling and Hybrid Sampling had improved the performance of models in terms of recall compared to the model trained without using sampling methods. The improvement in recall of getting

higher has affected accuracy and sensitivity. The higher recall value indicated that the model has a better ability to classify the minority class which is “Fraud”. Although recall is the important metric to study for each model in this project, this does not represent that accuracy and specificity are not important at all. The chosen optimized model was evaluated on the overall performance of the model. Table 4.2 with highlighted rows showed the best performed method with the most suitable data sampling method for each model respectively. Among the twelve results of the model shown in Table 4.2, the model of the Decision Tree using hybrid sampling is the best-performed model and the Decision Tree algorithms were selected as the base model to apply ensemble learning to improve for the performance.

4.6 Ensemble Learning

For the ensemble learning of boosting and bagging, the XGBoost and Bagging were selected. The base learner used for the ensemble learning was the Decision Tree. The ensemble learning model was trained the result was obtained as below:

Table 4.3: Result of trained Ensemble Learning model

Algorithm Used	Accuracy	Recall (TPR)	Specificity (TNR)	F1-Score
Decision Tree	88.89%	31.89%	92.51%	25.60%
Boosting + Decision Tree	93.84%	8.65%	99.28%	14.41%
Bagging + Decision Tree	93.90%	15.14%	98.93%	22.95%

From the table, by applying the ensemble learning of boosting and bagging, there was no improvement compared to the Decision Tree base model and the recall is getting

worse. Since the recall value is important in this project to classify for the minority class of “Fraud”. Therefore, the three sampling methods which are oversampling, undersampling and hybrid sampling that used above were applied again to ensemble model as further investigation on using ensemble model combined with the three sampling methods.

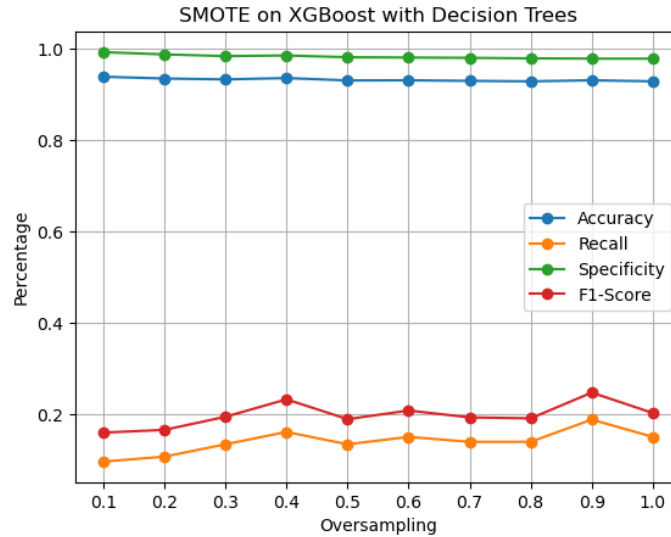


Figure 4.19: Result of Ensemble Learning (Boosting + Decision Tree) with applying SMOTE

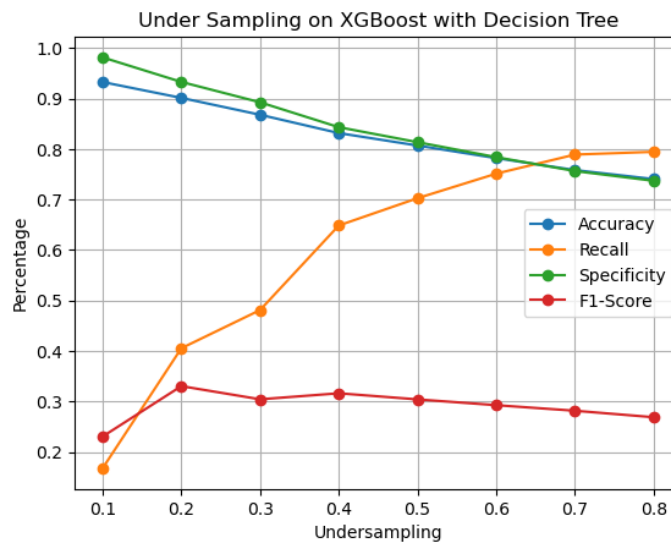


Figure 4.20: Result of Ensemble Learning (Boosting + Decision Tree) with applying Undersampling

From Figure 4.19, the result showed that SMOTE applied on the Boosting model has slightly improved in the recall and F1-Score while the accuracy and precision had not been affected significantly. It reached the optimized performance when oversampling with a ratio of 0.9. For Figure 4.20, the results showed that the undersampling applied to the Boosting model had significantly increase for the recall value and achieved the optimal performance when oversampling of 0.4 ratio.

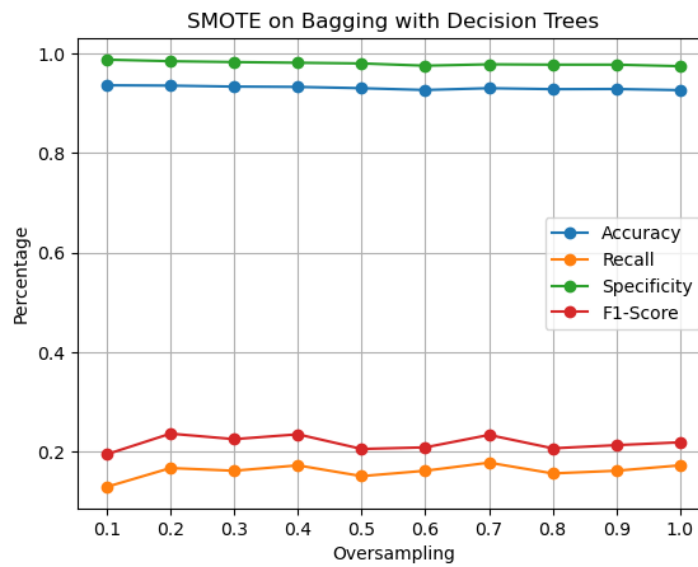


Figure 4.21: Result of Ensemble Learning (Bagging + Decision Tree) with applying SMOTE

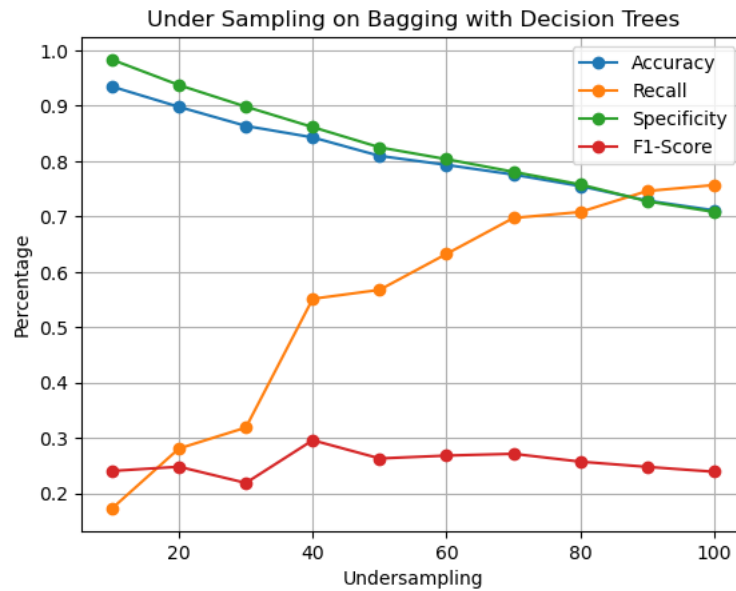


Figure 4.22: Result of Ensemble Learning (Bagging + Decision Tree) with applying Undersampling

From Figure 4.21 and Figure 4.22, the results show that both trained Bagging models increase in performance when applying oversampling and undersampling. The bagging model with underrrsampling showed a bigger improved of recall compared to the Bagging model with applying Oversampling.

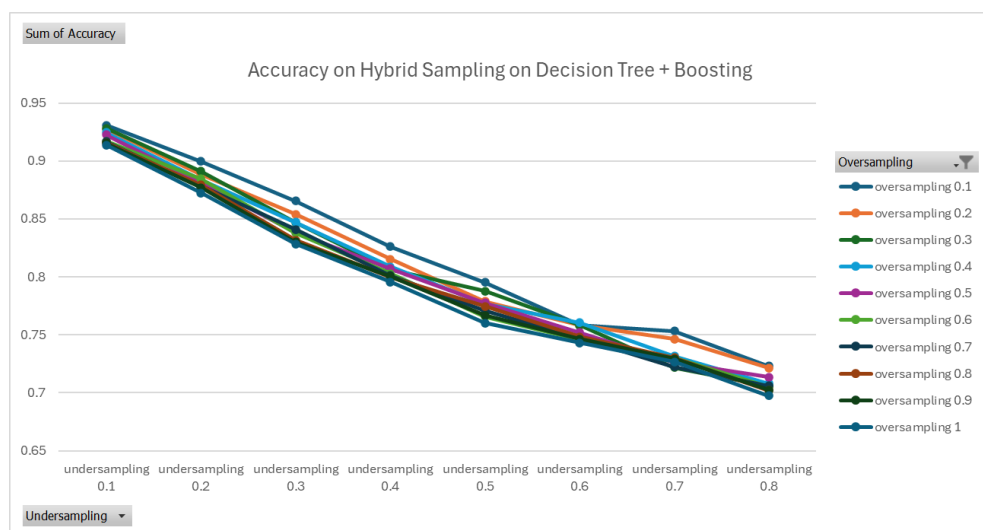


Figure 4.23: Accuracy result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling

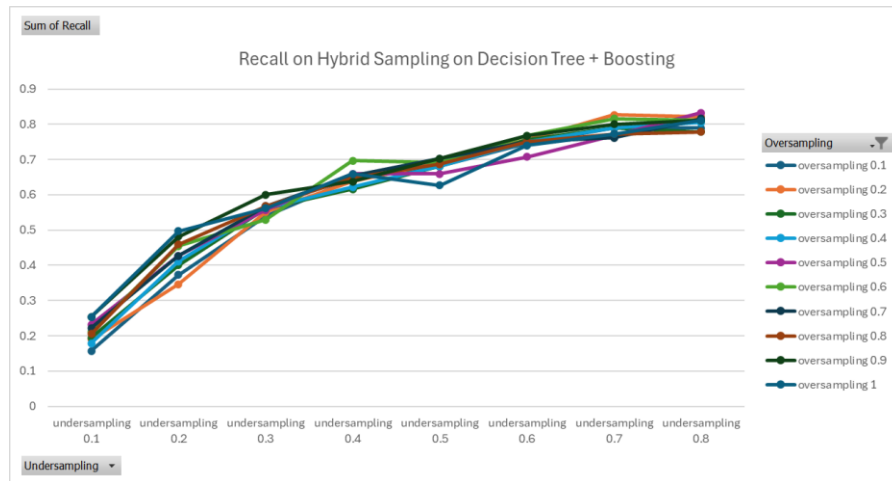


Figure 4.24: Recall result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling

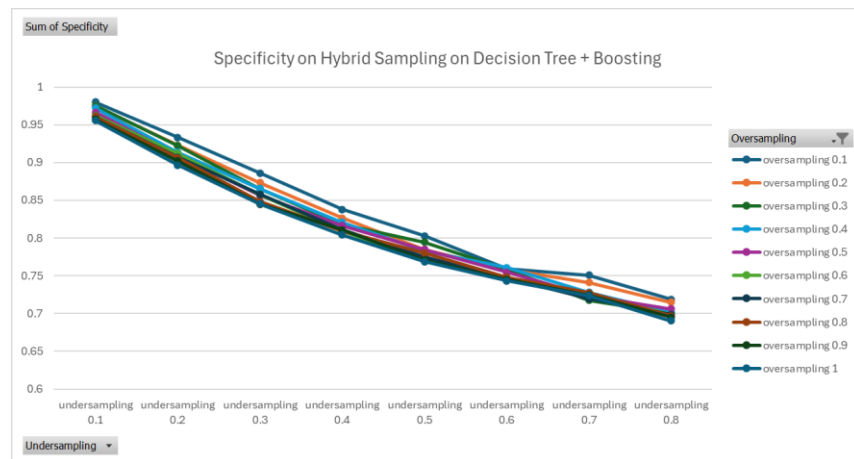


Figure 4.25: Specificity result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling

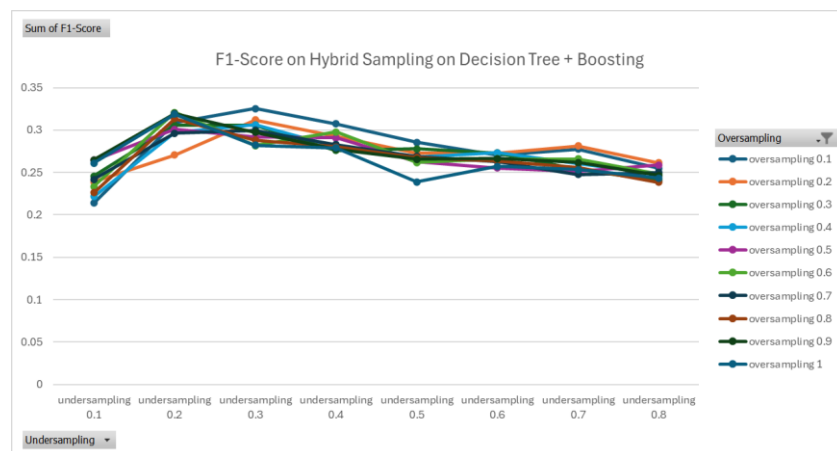


Figure 4.26: F1-Score result of Ensemble Learning (Boosting + Decision Tree) with applying Hybrid Sampling

Figure 4.23 to Figure 4.26 showed the result of applied Hybrid Sampling on the Boosting model. For accuracy and specificity, the results showed that both values continuously decreased when the ratio of hybrid sampling increased. Meanwhile, for the recall, it showed that the value continuously increased when the hybrid sampling ratio increased. For the F1-Score, it is increased before the ratio of 0.2 and decreased after 0.2 ratio.

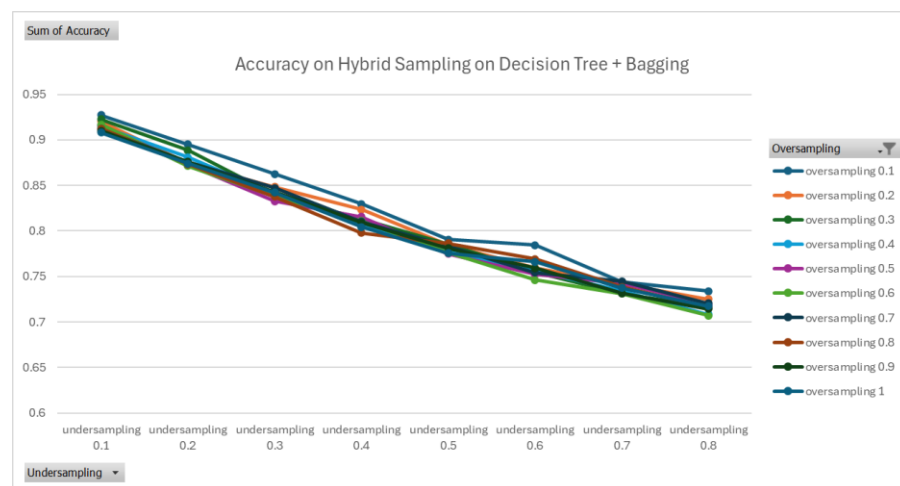


Figure 4.27: Accuracy result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling

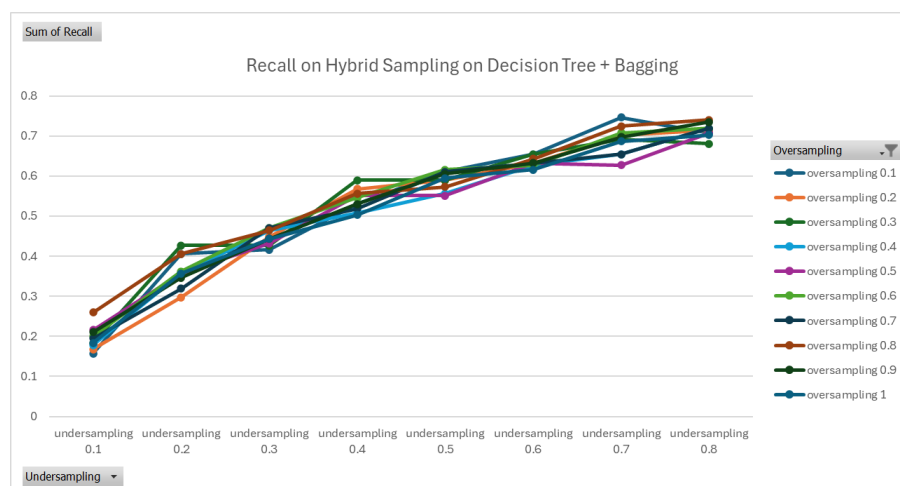


Figure 4.28: Recall result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling

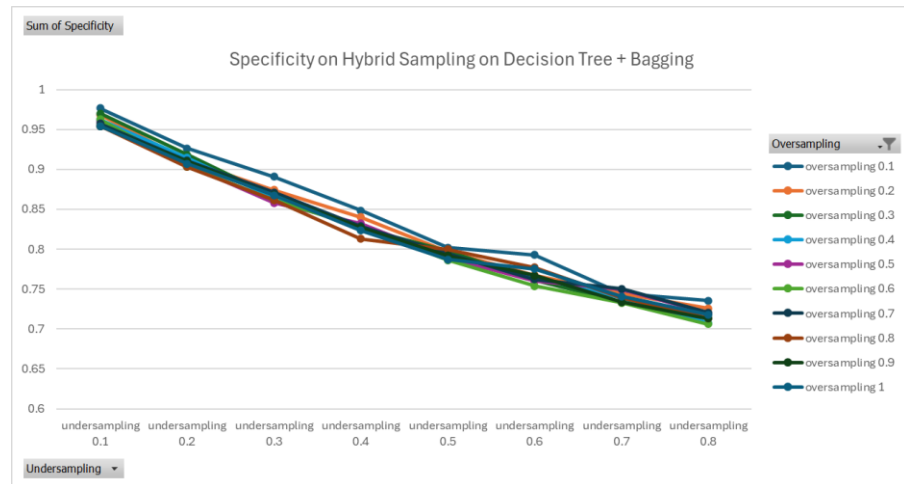


Figure 4.29: Specificity result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling

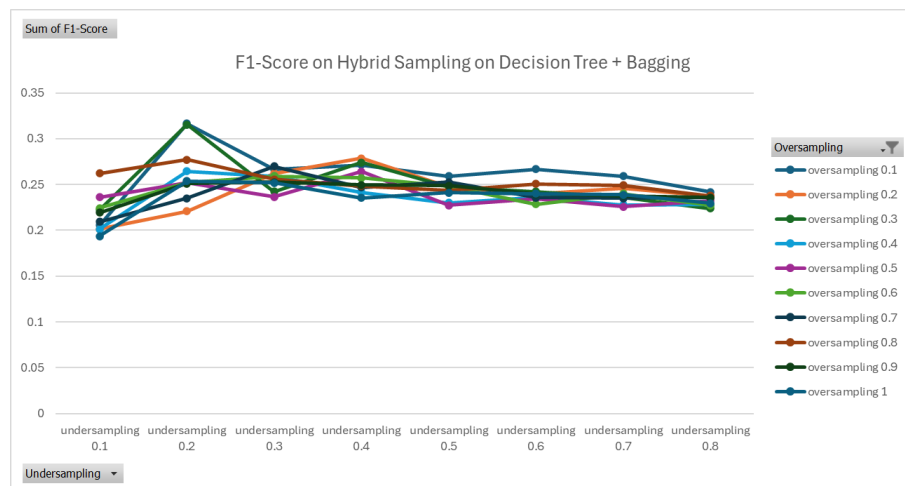


Figure 4.30: F1-Score result of Ensemble Learning (Bagging + Decision Tree) with applying Hybrid Sampling

Figure 4.27 to Figure 4.30 shows the result of applying Hybrid Sampling to the Bagging Model. Same with the Hybrid Sampling method applied to the Boosting model, the accuracy and specificity decreased when the ratio increased while the recall increased when the ratio increased.

Table 4.4: Summary result Ensemble Learning (Boosting & Bagging + Decision Tree) with and without applying handling method

Algorithm Used	Method	Accuracy	Recall (TPR)	Specificity (TNR)	F1- Score
Boosting + Decision Tree	-	93.84%	8.65%	99.28%	14.41%
	Oversampling (0.9)	93.13%	18.92%	97.86%	24.82%
	Undersampling (0.4)	83.17%	64.86%	84.34%	31.62%
	Hybrid sampling (Over 0.1 + Under 0.3)	86.54%	54.05%	88.62%	32.52%
Bagging + Decision Tree	-	93.90%	15.14%	98.93%	22.95%
	Oversampling (0.7)	93%	17.84%	97.79%	23.40%
	Undersampling (0.8)	75.49%	70.81%	75.78%	25.74%
	Hybrid sampling (0.1 Over, Under 0.7)	74.42%	74.59%	74.40%	25.92%

Figure 4.19 to Figure 4.30 illustrated the performance of accuracy, recall, specificity and F1-Score for ensemble learning of boosting and bagging by applying handling method. From the figure about recall, it showed that the recall of each model has significantly increased by using three of the methods. By referring to Table 4.3, the best performance of the ensemble model of boosting and decision tree was using the

Undersampling with a 0.4 ratio. For the ensemble model of bagging and decision tree, the best result was applied with the hybrid sampling with the combination of 0.1 Oversampling and 0.7 Undersampling. Both details value of accuracy, recall, specificity and F1-Score were shown in Table 4.4.

4.7 Comparison

Table 4.5: Summary results of all best trained model with application of data sampling method.

Algorithm Used	Method	Accuracy	Recall (TPR)	Specificity (TNR)	F1-Score
Linear Regression (Aslam et al., 2022)	Feature Selection	88.02%	42.25%	90.93%	29.27%
Decision Tree	Hybrid sampling (Over 0.6 + Under 0.8)	69.39%	75.14%	69.02%	22.75%
Naive Bayes	Undersampling (0.1)	57.26%	90.27%	55.16%	20.22%
LR	Oversampling (1.0)	76.20%	52.97%	77.68%	21.08%
Boosting + Decision Tree	Undersampling (0.4)	83.17%	64.86%	84.34%	31.62%
Bagging + Decision Tree	Hybrid sampling (0.1 Over, Under 0.7)	74.42%	74.59%	74.40%	25.92%

Table 4.5 recorded all the best performed models with the application of sampling methods for each trained model. Based on the table, the best performed model is the Ensemble Learning model of Boosting and Bagging with applied Undersampling (Ratio of 0.4) which has the 83.17% accuracy, 64.86% recall, 84.34% specificity and 31.62% F1-Score. By comparing the Linear Regression model proposed by (Aslam et al., 2022) with applying feature selection, the identified best model was achieved a comparable result with it. Although the accuracy and specificity of the best model were lower than the Linear Regression model by (Aslam et al., 2022), but the recall and F1-Score increased which indicated that the best performance model has a higher ability to classify the minority class which is “Fraud”. Another finding from applying different sampling methods to the model was found that when the recall increased, the accuracy and specificity decreased. In this domain of vehicle fraud insurance detection of class of “1” represent for fraud, the higher recall value was preferring to make sure the model can be able to classify the fraud case correctly.

CHAPTER 5: CONCLUSION

The dataset used in this project is highly imbalanced with an imbalanced ratio of 15.7 and consists of 14497 non-fraud and 923 fraud data. In this project, the imbalanced dataset undergoes several pre-processing methods which are oversampling, undersampling and hybrid sampling to make it more balanced before the dataset is used to train the model. All the work carried out in this project is using Python. The modelling algorithms used in this project were the DT, NB, LR, KNN, MLP and Ensemble Learning. All the results of the trained model are visualized and tabulated for better understanding.

Three objectives that were set in the initial stage of the project were achieved. In this project, the first objective was achieved by identifying the methods that can be used to handle the imbalanced dataset were identified through the Literature Review. The imbalanced dataset handling methods applied in this project included oversampling of SMOTE, undersampling of Random Undersampling and hybrid sampling of the combination of SMOTE and random undersampling. In addition, Ensemble Learning also showed better performance in treating the imbalanced dataset in the Literature Review and it was also implemented in this project as one method to handle imbalanced dataset.

The second objective was achieved in this project as all the optimized ratios of sampling data to get the best performance of each model were identified in Chapter 4 of Result. The third objective was also achieved in this project as each best performed model for each algorithm was evaluated and the benchmark model of Linear Regression model from (Aslam et al., 2022) was used to compare with the model. The best performed model in this project was the Ensemble Model of a combination of boosting and decision tree with applying undersampling 0.4 of ratio. The best performed ensemble boosting model achieved 83.17% accuracy, 64.86% recall, 84.34% specificity and 31.62% F1-Score. The

result of best performed model in this project was comparable to the best performed model from (Aslam et al., 2022)

In the future, other ensemble learning models such as Stacking and Voting can be applied to come out with a better performance ensemble learning model. In addition, the hyperparameter tuning method can be applied to each model to tune the performance of the model for better results since the hyperparameter tuning method did not apply in this project. By applying the hyperparameter tuning methods, it can figure out the best parameter of the model and achieve the best result. From (Aslam et al., 2022), Boruta algorithm which used as the feature selection method to preprocess the dataset. In future, more feature selection model can be apply to achieve a better performance model.

REFERENCES

- Aslam, F., Hunjra, A. I., Ftiti, Z., Louhichi, W., & Shams, T. (2022). Insurance fraud detection: Evidence from Artificial Intelligence and machine learning. *Research in International Business and Finance*, 62, 101744. <https://doi.org/10.1016/j.ribaf.2022.101744>
- Caruana, M. A., & Grech, L. (2021). Automobile insurance fraud detection. *Communications in Statistics: Case Studies, Data Analysis and Applications*, 7(4), 520–535. <https://doi.org/10.1080/23737484.2021.1986169>
- Choirunnisa, S., & Lianto, J. (2018). Hybrid method of undersampling and oversampling for handling imbalanced data. *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*. <https://doi.org/10.1109/isriti.2018.8864335>
- Dilkhaz Y, M. (2021). Detection of Vehicle Insurance Claim Fraud A Fraud Detection Use-Case for the Vehicle Insurance Industry. *International Journal of Progressive Sciences and Technologies (IJPSAT)*. <https://doi.org/10.52155/ijpsat.v30.1.3919>
- Ependi1, U., Fatchur Rochim, A., & Wibowo, A. (2023). A hybrid sampling approach for improving the classification of imbalanced data using ROS and NCL methods. *International Journal of Intelligent Engineering and Systems*, 16(3), 345–361. <https://doi.org/10.22266/ijies2023.0630.28>
- Grzyb, J., & Woźniak, M. (2022). SVM ensemble training for imbalanced data classification using multi-objective optimization techniques. *Applied Intelligence*, 53(12), 15424–15441. <https://doi.org/10.1007/s10489-022-04291-9>

- Harjai, S., Khatri, S. K., & Singh, G. (2019). Detecting fraudulent insurance claims using random forests and synthetic minority oversampling technique. *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*. <https://doi.org/10.1109/iscon47742.2019.9036162>
- Jonathan, B., Putra, P. H., & Ruldeviyani, Y. (2020). Observation imbalanced data text to predict users selling products on female daily with smote, Tomek, and smote-tomek. *2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*. <https://doi.org/10.1109/iaict50021.2020.9172033>
- Kulkarni, A., Chong, D., & Batarseh, F. A. (2020). Foundations of data imbalance and solutions for a data democracy. *Data Democracy*, 83–106. <https://doi.org/10.1016/b978-0-12-818366-3.00005-8>
- Liu, N., Li, X., Qi, E., Xu, M., Li, L., & Gao, B. (2020). A novel ensemble learning paradigm for medical diagnosis with Imbalanced Data. *IEEE Access*, 8, 171263–171280. <https://doi.org/10.1109/access.2020.3014362>
- Malaysia Number of Registered Vehicles*. (n.d.). Retrieved April 10, 2023, from <https://www.ceicdata.com/en/indicator/malaysia/number-of-registered-vehicles>
- Malhotra, R., & Jain, J. (2020). Handling imbalanced data using ensemble learning in software defect prediction. *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. <https://doi.org/10.1109/confluence47617.2020.9058124>
- Mînaştireanu, E.-A., & Meşniţă, G. (2020). Methods of handling unbalanced datasets in credit card fraud detection. *BRAIN. BROAD RESEARCH IN ARTIFICIAL*

INTELLIGENCE AND NEUROSCIENCE, 11(1), 131–143.
<https://doi.org/10.18662/brain/11.1/19>

Moon, H., Pu, Y., & Ceglia, C. (2019). A predictive modeling for detecting fraudulent automobile insurance claims. *Theoretical Economics Letters*, 09(06), 1886–1900.
<https://doi.org/10.4236/tel.2019.96120>

Nur Prasasti, I. M., Dhini, A., & Laoh, E. (2020). Automobile Insurance Fraud Detection Using supervised classifiers. *2020 International Workshop on Big Data and Information Security (IWBIS)*. <https://doi.org/10.1109/iwbis50925.2020.9255426>

Panigrahi, S., & Palkar, B. (2018). Comparative analysis on classification algorithms of auto-insurance fraud detection based on feature selection algorithms. *International Journal of Computer Sciences and Engineering*, 6(9), 72–77.
<https://doi.org/10.26438/ijcse/v6i9.7277>

Rukhsar, L., Haider Bangyal, W., Nisar, K., & Nisar, S. (2022). Prediction of insurance fraud detection using machine learning algorithms. *Mehran University Research Journal of Engineering and Technology*, 41(1), 33–40.
<https://doi.org/10.22581/muet1982.2201.04>

Salunkhe, U. R., & Mali, S. N. (2016). Classifier ensemble design for Imbalanced Data Classification: A hybrid approach. *Procedia Computer Science*, 85, 725–732.
<https://doi.org/10.1016/j.procs.2016.05.259>

Sanabila, H. R., & Jatmiko, W. (2018). Ensemble learning on large scale financial imbalanced data. *2018 International Workshop on Big Data and Information Security (IWBIS)*. <https://doi.org/10.1109/iwbis.2018.8471702>

- Sundarkumar, G. G., Ravi, V., & Siddeshwar, V. (2015). One-class support vector machine based undersampling: Application to churn prediction and insurance fraud detection. *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. <https://doi.org/10.1109/iccic.2015.7435726>
- Swana, E. F., Doorsamy, W., & Bokoro, P. (2022). Tomek link and smote approaches for Machine Fault Classification with an imbalanced dataset. *Sensors*, 22(9), 3246. <https://doi.org/10.3390/s22093246>
- Tan, D. (2021, July 5). *Motor insurance claims fraud cost RM1 billion a year in Malaysia, fighting corruption is a lonely road - allianz*. Paul Tan's Automotive News. Retrieved 10 April 2023, from <https://paultan.org/2021/07/05/motor-insurance-claims-fraud-cost-rm1-billion-a-year-in-malaysia-fighting-corruption-is-a-lonely-road-allianz/>
- Yankol-Schalck, M. (2022). The value of cross-data set analysis for Automobile Insurance Fraud Detection. *Research in International Business and Finance*, 63, 101769. <https://doi.org/10.1016/j.ribaf.2022.101769>
- 5 types of car insurance fraud*. (2022, August). Retrieved April 10, 2023, from <https://www.allstate.com/resources/car-insurance/types-of-car-insurance-fraud>

APPENDIX

Appendix A: Decision Tree Result

Oversampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.8956	0.3135	0.9327	0.2648
0.2	0.8959	0.2865	0.9348	0.2482
0.3	0.8936	0.2432	0.9352	0.2153
0.4	0.8930	0.2703	0.9327	0.2326
0.5	0.8868	0.2865	0.9251	0.2330
0.6	0.8781	0.2757	0.9165	0.2134
0.7	0.8885	0.2541	0.9289	0.2146
0.8	0.8807	0.2649	0.9200	0.2103
0.9	0.8859	0.2162	0.9286	0.1852
1	0.8855	0.2378	0.9269	0.1995

Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.8693	0.3405	0.9031	0.2382
0.2	0.8363	0.4865	0.8586	0.2628
0.3	0.8152	0.5189	0.8341	0.2520
0.4	0.7711	0.6108	0.7813	0.2425
0.5	0.7717	0.6486	0.7796	0.2542
0.6	0.7682	0.6378	0.7765	0.2482
0.7	0.7370	0.6378	0.7434	0.2254
0.8	0.7283	0.6595	0.7327	0.2255

Oversampling	Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.1	0.8826	0.3405	0.9172	0.2582
0.1	0.2	0.8372	0.4811	0.8600	0.2618
0.1	0.3	0.8022	0.5243	0.8199	0.2413
0.1	0.4	0.7711	0.5568	0.7848	0.2259
0.1	0.5	0.7523	0.5784	0.7634	0.2188
0.1	0.6	0.7296	0.6270	0.7361	0.2176
0.1	0.7	0.7351	0.6595	0.7399	0.2300
0.1	0.8	0.7007	0.7135	0.6999	0.2224
0.2	0.1	0.8726	0.3135	0.9082	0.2279
0.2	0.2	0.8213	0.4054	0.8479	0.2140
0.2	0.3	0.8006	0.5081	0.8192	0.2341
0.2	0.4	0.7665	0.5514	0.7803	0.2208
0.2	0.5	0.7529	0.5243	0.7675	0.2029
0.2	0.6	0.7370	0.5459	0.7492	0.1994
0.2	0.7	0.7299	0.6000	0.7382	0.2104
0.2	0.8	0.6952	0.6324	0.6992	0.1993
0.3	0.1	0.8651	0.3081	0.9007	0.2151

0.3	0.2	0.8110	0.4000	0.8372	0.2025
0.3	0.3	0.7977	0.5405	0.8141	0.2427
0.3	0.4	0.7588	0.5351	0.7730	0.2102
0.3	0.5	0.7626	0.6324	0.7710	0.2422
0.3	0.6	0.7341	0.6162	0.7416	0.2176
0.3	0.7	0.7215	0.6757	0.7244	0.2254
0.3	0.8	0.6965	0.6973	0.6964	0.2161
0.4	0.1	0.8661	0.3081	0.9017	0.2163
0.4	0.2	0.8184	0.4162	0.8441	0.2157
0.4	0.3	0.7899	0.5243	0.8068	0.2304
0.4	0.4	0.7665	0.5622	0.7796	0.2241
0.4	0.5	0.7529	0.6108	0.7620	0.2287
0.4	0.6	0.7309	0.5784	0.7406	0.2050
0.4	0.7	0.7215	0.6919	0.7234	0.2296
0.4	0.8	0.6910	0.7081	0.6899	0.2156
0.5	0.1	0.8612	0.2973	0.8972	0.2045
0.5	0.2	0.8171	0.4162	0.8427	0.2145
0.5	0.3	0.8035	0.5622	0.8189	0.2555
0.5	0.4	0.7659	0.5514	0.7796	0.2203
0.5	0.5	0.7474	0.6108	0.7561	0.2249
0.5	0.6	0.7318	0.6486	0.7372	0.2249
0.5	0.7	0.7117	0.6486	0.7158	0.2126
0.5	0.8	0.6978	0.7027	0.6975	0.2181
0.6	0.1	0.8654	0.3514	0.8982	0.2385
0.6	0.2	0.8262	0.4432	0.8506	0.2343
0.6	0.3	0.8012	0.5081	0.8199	0.2347
0.6	0.4	0.7756	0.5730	0.7885	0.2345
0.6	0.5	0.7455	0.5838	0.7558	0.2158
0.6	0.6	0.7234	0.6595	0.7275	0.2224
0.6	0.7	0.7072	0.6216	0.7127	0.2030
0.6	0.8	0.6939	0.7514	0.6902	0.2275
0.7	0.1	0.8654	0.3297	0.8996	0.2272
0.7	0.2	0.8262	0.4162	0.8524	0.2232
0.7	0.3	0.8009	0.4865	0.8210	0.2267
0.7	0.4	0.7711	0.5676	0.7841	0.2293
0.7	0.5	0.7438	0.5892	0.7537	0.2163
0.7	0.6	0.7335	0.6757	0.7372	0.2332
0.7	0.7	0.7101	0.6000	0.7171	0.1989
0.7	0.8	0.6936	0.6919	0.6937	0.2132
0.8	0.1	0.8580	0.3297	0.8917	0.2179
0.8	0.2	0.8210	0.4162	0.8468	0.2181
0.8	0.3	0.7964	0.4486	0.8186	0.2091
0.8	0.4	0.7672	0.5297	0.7823	0.2144
0.8	0.5	0.7526	0.6432	0.7596	0.2378
0.8	0.6	0.7228	0.6595	0.7268	0.2220
0.8	0.7	0.7098	0.6865	0.7113	0.2211
0.8	0.8	0.6997	0.7189	0.6985	0.2232
0.9	0.1	0.8654	0.3351	0.8993	0.2301
0.9	0.2	0.8191	0.4162	0.8448	0.2163
0.9	0.3	0.7944	0.5135	0.8123	0.2306

0.9	0.4	0.7662	0.5351	0.7810	0.2155
0.9	0.5	0.7406	0.5946	0.7499	0.2157
0.9	0.6	0.7163	0.6811	0.7185	0.2236
0.9	0.7	0.7140	0.6595	0.7175	0.2167
0.9	0.8	0.7033	0.6757	0.7051	0.2146
1.0	0.1	0.8664	0.3081	0.9020	0.2167
1.0	0.2	0.8285	0.4649	0.8517	0.2454
1.0	0.3	0.7922	0.5027	0.8106	0.2249
1.0	0.4	0.7717	0.5459	0.7861	0.2230
1.0	0.5	0.7429	0.5946	0.7523	0.2172
1.0	0.6	0.7315	0.6811	0.7347	0.2333
1.0	0.7	0.7250	0.6595	0.7292	0.2234
1.0	0.8	0.7091	0.6865	0.7106	0.2207

Appendix B: Naïve Bayes Result

Oversampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.4789	0.8973	0.4522	0.1712
0.2	0.4578	0.8919	0.4301	0.1648
0.3	0.4637	0.8649	0.4381	0.1621
0.4	0.4673	0.8486	0.4429	0.1604
0.5	0.4705	0.8541	0.446	0.1621
0.6	0.4741	0.8486	0.4502	0.1622
0.7	0.4724	0.8486	0.4484	0.1618
0.8	0.4721	0.8541	0.4477	0.1626
0.9	0.4711	0.8432	0.4474	0.1606
1.0	0.4728	0.8432	0.4491	0.1610

Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.5726	0.9027	0.5516	0.2022
0.2	0.5746	0.8919	0.5543	0.2010
0.3	0.5837	0.8811	0.5647	0.2025
0.4	0.5820	0.8865	0.5626	0.2028
0.5	0.5811	0.8865	0.5616	0.2025
0.6	0.5791	0.8973	0.5588	0.2037
0.7	0.5704	0.8973	0.5495	0.2004
0.8	0.5768	0.8919	0.5567	0.2018

Oversampling	Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.1	0.4575	0.8973	0.4295	0.1656
0.1	0.2	0.4818	0.8973	0.4553	0.1720
0.1	0.3	0.5324	0.8973	0.5091	0.1871

0.1	0.4	0.4818	0.8973	0.4553	0.1720
0.1	0.5	0.4711	0.8919	0.4443	0.1683
0.1	0.6	0.4608	0.8919	0.4333	0.1656
0.1	0.7	0.4530	0.8919	0.4250	0.1636
0.1	0.8	0.4546	0.9027	0.4260	0.1657
0.2	0.1	0.4553	0.8919	0.4274	0.1642
0.2	0.2	0.4578	0.8865	0.4305	0.1640
0.2	0.3	0.4679	0.8811	0.4415	0.1657
0.2	0.4	0.4614	0.8865	0.4343	0.1649
0.2	0.5	0.4575	0.8919	0.4298	0.1648
0.2	0.6	0.4553	0.8919	0.4274	0.1642
0.2	0.7	0.4494	0.8973	0.4208	0.1635
0.2	0.8	0.4549	0.8919	0.4270	0.1641
0.3	0.1	0.4621	0.8811	0.4353	0.1642
0.3	0.2	0.4653	0.8595	0.4402	0.1617
0.3	0.3	0.4721	0.8595	0.4474	0.1634
0.3	0.4	0.4682	0.8595	0.4433	0.1624
0.3	0.5	0.4627	0.8757	0.4364	0.1636
0.3	0.6	0.4614	0.8757	0.4350	0.1632
0.3	0.7	0.4617	0.8757	0.4353	0.1633
0.3	0.8	0.4614	0.8703	0.4353	0.1624
0.4	0.1	0.4630	0.8595	0.4377	0.1611
0.4	0.2	0.4676	0.8486	0.4433	0.1605
0.4	0.3	0.4773	0.8486	0.4536	0.1630
0.4	0.4	0.4698	0.8486	0.4457	0.1611
0.4	0.5	0.4653	0.8595	0.4402	0.1617
0.4	0.6	0.4617	0.8595	0.4364	0.1608
0.4	0.7	0.4624	0.8595	0.4370	0.1609
0.4	0.8	0.4627	0.8541	0.4377	0.1602
0.5	0.1	0.4682	0.8541	0.4436	0.1616
0.5	0.2	0.4705	0.8541	0.4460	0.1621
0.5	0.3	0.4789	0.8486	0.4553	0.1635
0.5	0.4	0.4705	0.8541	0.4460	0.1621
0.5	0.5	0.4679	0.8541	0.4433	0.1615
0.5	0.6	0.4669	0.8541	0.4422	0.1612
0.5	0.7	0.4660	0.8486	0.4415	0.1601
0.5	0.8	0.4673	0.8486	0.4429	0.1604
0.6	0.1	0.4708	0.8541	0.4464	0.1622
0.6	0.2	0.4737	0.8486	0.4498	0.1621
0.6	0.3	0.4812	0.8324	0.4588	0.1614
0.6	0.4	0.4744	0.8378	0.4512	0.1605
0.6	0.5	0.4711	0.8486	0.4471	0.1614
0.6	0.6	0.4695	0.8541	0.4450	0.1619
0.6	0.7	0.4711	0.8432	0.4474	0.1606
0.6	0.8	0.4702	0.8432	0.4464	0.1603
0.7	0.1	0.4708	0.8541	0.4464	0.1622
0.7	0.2	0.4724	0.8541	0.4481	0.1626
0.7	0.3	0.4799	0.8324	0.4574	0.1611
0.7	0.4	0.4721	0.8432	0.4484	0.1608
0.7	0.5	0.4708	0.8486	0.4467	0.1614

0.7	0.6	0.4695	0.8486	0.4453	0.1610
0.7	0.7	0.4705	0.8378	0.4471	0.1595
0.7	0.8	0.4702	0.8432	0.4464	0.1603
0.8	0.1	0.4708	0.8541	0.4464	0.1622
0.8	0.2	0.4721	0.8541	0.4477	0.1626
0.8	0.3	0.4770	0.8324	0.4543	0.1603
0.8	0.4	0.4721	0.8432	0.4484	0.1608
0.8	0.5	0.4702	0.8486	0.4460	0.1612
0.8	0.6	0.4689	0.8486	0.4446	0.1609
0.8	0.7	0.4711	0.8486	0.4471	0.1614
0.8	0.8	0.4692	0.8432	0.4453	0.1601
0.9	0.1	0.4695	0.8541	0.4450	0.1619
0.9	0.2	0.4718	0.8486	0.4477	0.1616
0.9	0.3	0.4763	0.8432	0.4529	0.1619
0.9	0.4	0.4718	0.8486	0.4477	0.1616
0.9	0.5	0.4692	0.8486	0.4450	0.1609
0.9	0.6	0.4685	0.8486	0.4443	0.1608
0.9	0.7	0.4692	0.8486	0.4450	0.1609
0.9	0.8	0.4702	0.8486	0.4460	0.1612
1.0	0.1	0.4708	0.8486	0.4467	0.1614
1.0	0.2	0.4724	0.8432	0.4488	0.1609
1.0	0.3	0.4792	0.8378	0.4564	0.1618
1.0	0.4	0.4715	0.8324	0.4484	0.1589
1.0	0.5	0.4708	0.8486	0.4467	0.1614
1.0	0.6	0.4695	0.8486	0.4453	0.1610
1.0	0.7	0.4705	0.8432	0.4467	0.1604
1.0	0.8	0.4711	0.8486	0.4471	0.1614

Appendix C: Logistic Regression Result

Oversampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9400	0.0000	1.0000	0.0000
0.2	0.9108	0.0865	0.9634	0.1042
0.3	0.8940	0.1568	0.9410	0.1506
0.4	0.8495	0.3189	0.8834	0.2027
0.5	0.8424	0.3351	0.8748	0.2033
0.6	0.8366	0.3459	0.8679	0.2025
0.7	0.8080	0.4054	0.8337	0.2022
0.8	0.7993	0.4541	0.8213	0.2135
0.9	0.7727	0.4973	0.7903	0.2079
1	0.7620	0.5297	0.7768	0.2108

Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9400	0.0000	1.0000	0.0000
0.2	0.9225	0.0541	0.9779	0.0772

0.3	0.9150	0.1081	0.9665	0.1325
0.4	0.8486	0.2703	0.8855	0.1764
0.5	0.7983	0.4162	0.8227	0.1985
0.6	0.7536	0.5676	0.7654	0.2165
0.7	0.6764	0.6541	0.6778	0.1952
0.8	0.6158	0.8054	0.6037	0.2009

Oversampling	Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.1	0.9384	0.0108	0.9976	0.0206
0.1	0.2	0.9011	0.1892	0.9465	0.1867
0.1	0.3	0.8249	0.3892	0.8527	0.2105
0.1	0.4	0.7438	0.5784	0.7544	0.2131
0.1	0.5	0.6842	0.7405	0.6806	0.2196
0.1	0.6	0.6310	0.8216	0.6188	0.2108
0.1	0.7	0.6060	0.8432	0.5909	0.2043
0.1	0.8	0.5370	0.8270	0.5185	0.1765
0.2	0.1	0.8813	0.2541	0.9214	0.2043
0.2	0.2	0.7844	0.4973	0.8027	0.2167
0.2	0.3	0.7163	0.6324	0.7216	0.2110
0.2	0.4	0.6608	0.7568	0.6547	0.2112
0.2	0.5	0.5850	0.7838	0.5723	0.1847
0.2	0.6	0.5898	0.8432	0.5736	0.1978
0.2	0.7	0.5104	0.8757	0.4871	0.1767
0.2	0.8	0.5477	0.8919	0.5257	0.1913
0.3	0.1	0.8615	0.2757	0.8989	0.1928
0.3	0.2	0.7286	0.6108	0.7361	0.2126
0.3	0.3	0.6466	0.7297	0.6413	0.1985
0.3	0.4	0.5879	0.7946	0.5747	0.1879
0.3	0.5	0.5441	0.8378	0.5254	0.1807
0.3	0.6	0.4851	0.9027	0.4584	0.1738
0.3	0.7	0.5363	0.8703	0.5150	0.1838
0.3	0.8	0.4384	0.9243	0.4074	0.1649
0.4	0.1	0.8194	0.3676	0.8482	0.1962
0.4	0.2	0.7241	0.6595	0.7282	0.2228
0.4	0.3	0.6673	0.7243	0.6637	0.2071
0.4	0.4	0.6109	0.8270	0.5971	0.2032
0.4	0.5	0.5026	0.8595	0.4798	0.1717
0.4	0.6	0.5276	0.8432	0.5074	0.1764
0.4	0.7	0.5113	0.8757	0.4881	0.1770
0.4	0.8	0.3359	0.9297	0.2980	0.1438
0.5	0.1	0.8019	0.3946	0.8279	0.1929
0.5	0.2	0.6822	0.6811	0.6823	0.2045
0.5	0.3	0.6102	0.7784	0.5995	0.1933
0.5	0.4	0.5276	0.8486	0.5071	0.1773
0.5	0.5	0.4647	0.9135	0.4360	0.1699
0.5	0.6	0.4475	0.9351	0.4164	0.1688
0.5	0.7	0.4222	0.9405	0.3891	0.1634
0.5	0.8	0.4095	0.9405	0.3756	0.1604

0.6	0.1	0.7925	0.4757	0.8127	0.2157
0.6	0.2	0.6385	0.7189	0.6333	0.1926
0.6	0.3	0.5496	0.8216	0.5323	0.1796
0.6	0.4	0.5788	0.8108	0.5640	0.1876
0.6	0.5	0.3865	0.8757	0.3553	0.1462
0.6	0.6	0.4763	0.9081	0.4488	0.1722
0.6	0.7	0.4043	0.9135	0.3719	0.1554
0.6	0.8	0.3972	0.9459	0.3622	0.1584
0.7	0.1	0.7630	0.5676	0.7754	0.2232
0.7	0.2	0.6336	0.7622	0.6254	0.1997
0.7	0.3	0.5483	0.8378	0.5298	0.1820
0.7	0.4	0.5052	0.8649	0.4822	0.1733
0.7	0.5	0.4591	0.9027	0.4308	0.1668
0.7	0.6	0.4342	0.9405	0.4019	0.1663
0.7	0.7	0.4014	0.9459	0.3667	0.1594
0.7	0.8	0.3898	0.9622	0.3532	0.1591
0.8	0.1	0.7283	0.6162	0.7354	0.2139
0.8	0.2	0.6015	0.7838	0.5899	0.1909
0.8	0.3	0.5315	0.8649	0.5102	0.1813
0.8	0.4	0.4750	0.9135	0.4471	0.1727
0.8	0.5	0.4306	0.9297	0.3988	0.1638
0.8	0.6	0.4131	0.9514	0.3788	0.1628
0.8	0.7	0.4326	0.9297	0.4008	0.1643
0.8	0.8	0.3726	0.9730	0.3343	0.1569
0.9	0.1	0.7049	0.6595	0.7078	0.2114
0.9	0.2	0.5944	0.8054	0.5809	0.1924
0.9	0.3	0.5169	0.8757	0.4940	0.1786
0.9	0.4	0.4647	0.9135	0.4360	0.1699
0.9	0.5	0.4332	0.9297	0.4015	0.1644
0.9	0.6	0.4773	0.9135	0.4495	0.1733
0.9	0.7	0.3807	0.9568	0.3439	0.1564
0.9	0.8	0.3726	0.9730	0.3343	0.1569
1.0	0.1	0.6634	0.6811	0.6623	0.1953
1.0	0.2	0.5781	0.7892	0.5647	0.1833
1.0	0.3	0.5840	0.8270	0.5685	0.1926
1.0	0.4	0.4604	0.9027	0.4322	0.1672
1.0	0.5	0.4514	0.9243	0.4212	0.1681
1.0	0.6	0.4741	0.9189	0.4457	0.1733
1.0	0.7	0.3606	0.9784	0.3211	0.1551
1.0	0.8	0.3602	0.9784	0.3208	0.1550

Appendix D: Ensemble Learning Result

Boosting + Decision Tree

Oversampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9390	0.0973	0.9928	0.1607
0.2	0.9351	0.1081	0.9879	0.1667
0.3	0.9332	0.1351	0.9841	0.1953
0.4	0.9361	0.1622	0.9855	0.2335
0.5	0.9309	0.1351	0.9817	0.1901
0.6	0.9313	0.1514	0.9810	0.2090
0.7	0.9300	0.1405	0.9803	0.1940
0.8	0.9290	0.1405	0.9793	0.1919
0.9	0.9313	0.1892	0.9786	0.2482
1	0.9290	0.1514	0.9786	0.2036

Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9329	0.1676	0.9817	0.2305
0.2	0.9014	0.4054	0.9331	0.3304
0.3	0.8680	0.4811	0.8927	0.3043
0.4	0.8317	0.6486	0.8434	0.3162
0.5	0.8071	0.7027	0.8137	0.3041
0.6	0.7821	0.7514	0.7841	0.2926
0.7	0.7584	0.7892	0.7565	0.2816
0.8	0.7406	0.7946	0.7372	0.2687

Oversampling	Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.1	0.9309	0.1568	0.9803	0.2140
0.1	0.2	0.8998	0.3730	0.9334	0.3087
0.1	0.3	0.8654	0.5405	0.8862	0.3252
0.1	0.4	0.8262	0.6432	0.8379	0.3075
0.1	0.5	0.7954	0.6811	0.8027	0.2854
0.1	0.6	0.7584	0.7459	0.7592	0.2703
0.1	0.7	0.7532	0.7892	0.7509	0.2773
0.1	0.8	0.7231	0.7892	0.7189	0.2548
0.2	0.1	0.9277	0.1892	0.9748	0.2389
0.2	0.2	0.8881	0.3459	0.9227	0.2706
0.2	0.3	0.8541	0.5514	0.8734	0.3119
0.2	0.4	0.8152	0.6378	0.8265	0.2928
0.2	0.5	0.7789	0.6919	0.7844	0.2729
0.2	0.6	0.7581	0.7568	0.7582	0.2729
0.2	0.7	0.7464	0.8270	0.7413	0.2813

0.2	0.8	0.7215	0.8216	0.7151	0.2614
0.3	0.1	0.9283	0.1946	0.9752	0.2457
0.3	0.2	0.8911	0.4000	0.9224	0.3058
0.3	0.3	0.8470	0.5622	0.8651	0.3059
0.3	0.4	0.8061	0.6162	0.8182	0.2760
0.3	0.5	0.7876	0.6811	0.7944	0.2778
0.3	0.6	0.7581	0.7568	0.7582	0.2729
0.3	0.7	0.7218	0.7892	0.7175	0.2539
0.3	0.8	0.7056	0.7784	0.7009	0.2408
0.4	0.1	0.9248	0.1784	0.9724	0.2215
0.4	0.2	0.8836	0.4108	0.9138	0.2975
0.4	0.3	0.8473	0.5622	0.8655	0.3063
0.4	0.4	0.8087	0.6216	0.8206	0.2805
0.4	0.5	0.7766	0.6811	0.7827	0.2678
0.4	0.6	0.7604	0.7514	0.7610	0.2734
0.4	0.7	0.7315	0.7892	0.7278	0.2607
0.4	0.8	0.7078	0.8054	0.7016	0.2485
0.5	0.1	0.9225	0.2324	0.9665	0.2646
0.5	0.2	0.8810	0.4270	0.9100	0.3010
0.5	0.3	0.8382	0.5568	0.8562	0.2922
0.5	0.4	0.8071	0.6595	0.8165	0.2908
0.5	0.5	0.7776	0.6595	0.7851	0.2624
0.5	0.6	0.7519	0.7081	0.7547	0.2551
0.5	0.7	0.7260	0.7676	0.7234	0.2516
0.5	0.8	0.7137	0.8324	0.7061	0.2586
0.6	0.1	0.9170	0.2108	0.9621	0.2335
0.6	0.2	0.8846	0.4541	0.9120	0.3206
0.6	0.3	0.8379	0.5297	0.8575	0.2816
0.6	0.4	0.8029	0.6973	0.8096	0.2979
0.6	0.5	0.7659	0.6919	0.7706	0.2618
0.6	0.6	0.7461	0.7676	0.7447	0.2662
0.6	0.7	0.7299	0.8162	0.7244	0.2661
0.6	0.8	0.7053	0.8108	0.6985	0.2481
0.7	0.1	0.9167	0.2216	0.9610	0.2419
0.7	0.2	0.8784	0.4270	0.9072	0.2964
0.7	0.3	0.8408	0.5676	0.8582	0.2996
0.7	0.4	0.8012	0.6541	0.8106	0.2830
0.7	0.5	0.7711	0.7027	0.7754	0.2692
0.7	0.6	0.7471	0.7514	0.7468	0.2628
0.7	0.7	0.7224	0.7622	0.7199	0.2478
0.7	0.8	0.7059	0.8162	0.6989	0.2498
0.8	0.1	0.9157	0.2054	0.9610	0.2262
0.8	0.2	0.8794	0.4595	0.9062	0.3137
0.8	0.3	0.8317	0.5676	0.8486	0.2881
0.8	0.4	0.7999	0.6486	0.8096	0.2800
0.8	0.5	0.7746	0.6865	0.7803	0.2677
0.8	0.6	0.7484	0.7514	0.7482	0.2638
0.8	0.7	0.7302	0.7730	0.7275	0.2558
0.8	0.8	0.7020	0.7784	0.6971	0.2386
0.9	0.1	0.9154	0.2541	0.9576	0.2648

0.9	0.2	0.8771	0.4811	0.9024	0.3196
0.9	0.3	0.8301	0.6000	0.8448	0.2976
0.9	0.4	0.8006	0.6378	0.8110	0.2773
0.9	0.5	0.7669	0.7027	0.7710	0.2656
0.9	0.6	0.7464	0.7676	0.7451	0.2664
0.9	0.7	0.7292	0.8000	0.7247	0.2617
0.9	0.8	0.7023	0.8108	0.6954	0.2463
1	0.1	0.9137	0.2541	0.9558	0.2611
1	0.2	0.8726	0.4973	0.8965	0.3189
1	0.3	0.8285	0.5622	0.8455	0.2822
1	0.4	0.7957	0.6595	0.8044	0.2792
1	0.5	0.7601	0.6270	0.7685	0.2387
1	0.6	0.7432	0.7405	0.7434	0.2570
1	0.7	0.7267	0.7730	0.7237	0.2533
1	0.8	0.6975	0.8108	0.6902	0.2433

Bagging + Decision Tree

Oversampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9358	0.1297	0.9872	0.1951
0.2	0.9351	0.1676	0.9841	0.2366
0.3	0.9332	0.1622	0.9824	0.2256
0.4	0.9326	0.1730	0.9810	0.2353
0.5	0.9300	0.1514	0.9796	0.2059
0.6	0.9264	0.1622	0.9752	0.2091
0.7	0.9300	0.1784	0.9779	0.2340
0.8	0.9280	0.1568	0.9772	0.2071
0.9	0.9283	0.1622	0.9772	0.2135
1	0.9261	0.1730	0.9741	0.2192

Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.9345	0.1730	0.9831	0.2406
0.2	0.8979	0.2811	0.9372	0.2482
0.3	0.8635	0.3189	0.8982	0.2189
0.4	0.8427	0.5514	0.8613	0.2961
0.5	0.8093	0.5676	0.8248	0.2632
0.6	0.7931	0.6324	0.8034	0.2683
0.7	0.7756	0.6973	0.7806	0.2716
0.8	0.7549	0.7081	0.7578	0.2574

Oversampling	Undersampling	Accuracy	Recall	Specificity	F1-Score
0.1	0.1	0.9274	0.1568	0.9765	0.2057
0.1	0.2	0.8949	0.4054	0.9262	0.3165
0.1	0.3	0.8625	0.4162	0.8910	0.2664
0.1	0.4	0.8294	0.5297	0.8486	0.2715
0.1	0.5	0.7905	0.6108	0.8020	0.2592
0.1	0.6	0.7844	0.6541	0.7927	0.2668
0.1	0.7	0.7442	0.7459	0.7440	0.2592
0.1	0.8	0.7338	0.7081	0.7354	0.2419
0.2	0.1	0.9202	0.1676	0.9683	0.2013
0.2	0.2	0.8742	0.2973	0.9110	0.2209
0.2	0.3	0.8486	0.4486	0.8741	0.2622
0.2	0.4	0.8236	0.5676	0.8399	0.2785
0.2	0.5	0.7847	0.5892	0.7972	0.2472
0.2	0.6	0.7594	0.6324	0.7675	0.2398
0.2	0.7	0.7419	0.7027	0.7444	0.2462
0.2	0.8	0.7250	0.7135	0.7258	0.2374
0.3	0.1	0.9228	0.1838	0.9700	0.2222
0.3	0.2	0.8888	0.4270	0.9182	0.3154
0.3	0.3	0.8398	0.4270	0.8662	0.2423
0.3	0.4	0.8129	0.5892	0.8272	0.2742
0.3	0.5	0.7850	0.5892	0.7975	0.2474
0.3	0.6	0.7545	0.6541	0.7610	0.2422
0.3	0.7	0.7312	0.6919	0.7337	0.2359
0.3	0.8	0.7166	0.6811	0.7189	0.2238
0.4	0.1	0.9150	0.1784	0.9621	0.2012
0.4	0.2	0.8810	0.3568	0.9145	0.2645
0.4	0.3	0.8405	0.4649	0.8644	0.2590
0.4	0.4	0.8087	0.5081	0.8279	0.2416
0.4	0.5	0.7766	0.5568	0.7906	0.2302
0.4	0.6	0.7562	0.6270	0.7644	0.2358
0.4	0.7	0.7341	0.6541	0.7392	0.2279
0.4	0.8	0.7082	0.7189	0.7075	0.2281
0.5	0.1	0.9160	0.2162	0.9607	0.2360
0.5	0.2	0.8729	0.3568	0.9058	0.2519
0.5	0.3	0.8327	0.4324	0.8582	0.2367
0.5	0.4	0.8155	0.5514	0.8324	0.2639
0.5	0.5	0.7750	0.5514	0.7892	0.2272
0.5	0.6	0.7523	0.6324	0.7599	0.2345
0.5	0.7	0.7422	0.6270	0.7496	0.2259
0.5	0.8	0.7192	0.7081	0.7199	0.2323
0.6	0.1	0.9170	0.2000	0.9627	0.2242
0.6	0.2	0.8713	0.3622	0.9038	0.2524
0.6	0.3	0.8385	0.4703	0.8620	0.2589
0.6	0.4	0.8110	0.5459	0.8279	0.2573
0.6	0.5	0.7759	0.6162	0.7861	0.2481
0.6	0.6	0.7461	0.6270	0.7537	0.2286
0.6	0.7	0.7312	0.7081	0.7327	0.2401
0.6	0.8	0.7069	0.7189	0.7061	0.2274

0.7	0.1	0.9118	0.1946	0.9576	0.2093
0.7	0.2	0.8755	0.3189	0.9110	0.2351
0.7	0.3	0.8473	0.4703	0.8713	0.2698
0.7	0.4	0.8100	0.5189	0.8286	0.2468
0.7	0.5	0.7831	0.6108	0.7941	0.2525
0.7	0.6	0.7542	0.6324	0.7620	0.2359
0.7	0.7	0.7445	0.6541	0.7503	0.2350
0.7	0.8	0.7205	0.7189	0.7206	0.2358
0.8	0.1	0.9125	0.2595	0.9541	0.2623
0.8	0.2	0.8732	0.4054	0.9031	0.2773
0.8	0.3	0.8375	0.4649	0.8613	0.2556
0.8	0.4	0.7977	0.5568	0.8130	0.2482
0.8	0.5	0.7863	0.5730	0.7999	0.2434
0.8	0.6	0.7691	0.6432	0.7772	0.2505
0.8	0.7	0.7383	0.7243	0.7392	0.2493
0.8	0.8	0.7153	0.7405	0.7137	0.2378
0.9	0.1	0.9099	0.2108	0.9545	0.2191
0.9	0.2	0.8761	0.3459	0.9100	0.2510
0.9	0.3	0.8427	0.4432	0.8682	0.2527
0.9	0.4	0.8090	0.5297	0.8268	0.2497
0.9	0.5	0.7808	0.6054	0.7920	0.2489
0.9	0.6	0.7591	0.6324	0.7672	0.2395
0.9	0.7	0.7315	0.6973	0.7337	0.2376
0.9	0.8	0.7143	0.7351	0.7130	0.2359
1	0.1	0.9082	0.1838	0.9545	0.1937
1	0.2	0.8742	0.3568	0.9072	0.2538
1	0.3	0.8421	0.4432	0.8675	0.2519
1	0.4	0.8042	0.5027	0.8234	0.2354
1	0.5	0.7756	0.5946	0.7872	0.2412
1	0.6	0.7659	0.6162	0.7754	0.2400
1	0.7	0.7374	0.6865	0.7406	0.2387
1	0.8	0.7173	0.7027	0.7182	0.2297

Appendix E: Project python code

<https://github.com/RSSST/Research>