

COMM054 – Final Report

To What Extent Can a Credit-Card Transaction Be Identified as Fraudulent?

Abubakar Jauro Sani (6769836)
Adarsh Manoj (6532510)
Piyush Patel (6760850)
Ramandeep Singh (6730378)

Table of Contents

Introduction	4
Problem Definition	4
Existing Works.....	4
Software Implementations	4
Table of Previous Published Works.....	5
Dataset.....	5
Methods	7
Data Exploration	7
Data Preparation	9
Data Transformation	9
Class Balancing	10
Dimensionality Reduction	11
Data Splitting	12
Machine Learning Models.....	12
Logistic Regression	12
Decision Tree	13
Random Forest.....	14
Hyperparameter Tuning.....	14
Evaluation Methods	14
AUC-ROC.....	15
Cohens(k) Value	15
Confusion Matrix.....	16
Changes to Project Plan	16
Results	17
Fully Processed Dataset	17
LR Implementation	17
Logistic Regression Results for Under-sampled Data	17
Evaluation.....	17
Logistic Regression results for Oversampled data.....	18
Evaluation.....	18
Logistic Regression Results for Hybrid Sampled Data	18
Evaluation.....	18
DT Implementation	19
Decision Tree Results for Under-sampled Data	19
Evaluation.....	19
Decision Tree results for Oversampled Data	19
Evaluation.....	19
Decision Tree Results for Hybrid Sampled Data	20
Evaluation.....	20

Conclusions	21
Data Driven Conclusions.....	21
Business Driven Conclusions.....	22
Reference List.....	23
Appendix.....	23
Table of Figures	25
Table of Tables.....	26
Gantt Table	26
Gantt Chart	28
Hybrid Sampling Training Results Logistic Regression	28
Hybrid Sampling Training Results Decision Tree	29
Oversampling Results	30
Undersampling Results	32
Hybrid Sampling Final Results	35

Introduction

The number of credit card fraud transactions has three folded from 2011 to 2020 based on the report by Merchant Savvy, accumulating a loss of greater than \$32.39 billion (Whatman, 2021). Credit card fraud is any fraud committed by stealing /using someone's money to buy goods or services under someone else's credit or debit cards or by unauthorized means. In 2018 alone, the UK suffered a loss of approximately £678 million through acts of credit card fraud (Whatman, 2021). This leads to a series of concerns amongst the Credit Card issuing companies to protect their customers from these frauds hence they need an efficient way to handle these – leading to the credit card fraud detection systems. These systems help these companies in answering key questions, such as:

- What percentage of credit card transactions are frauds?
- How efficient and accurate the strategy is?
- How much amount of money can be saved by implementing an efficient strategy?

Problem Definition

Credit Card Fraud Detection is a method of detecting fraud by implementing complex algorithms and machine learning knowledge to identify patterns in data that can protect customer's money from falling into the wrong hands. Traditional techniques for fraud heavily rely on manually analysing patterns regarding time, geolocation, and the principles of fraud – this is a laborious task, which machine learning-enabled credit card fraud detection can aide in.

The aim is to determine an efficient way to detect whether a particular transaction is a fraud or not by choosing various strategies to maximize the efficiency and accuracy of the models. The model will examine the transactions using Business Analytics techniques including but not limited to the use of machine learning models to determine the patterns amongst the data.

Existing Works

Software Implementations

Before progressing, a brief background into the history of credit card fraud detection. The first measures of fraud detection were physical/electronic implementations through the late 90s, they include real time verification of account status, CVV for ecommerce transactions, PIN for daily transactions, and many more. As technology developed so did the measures for fraud detection which focussed on digital innovations that are ongoing till date. The key developments in the modern-day field of fraud detection lie in AI and data powered tools. Visa provides their VAA software to help manage risks, and reduce fraudulent transactions, their innovations have led them to boost their ability to detect fraudulent transaction up to 130% for debit transactions and 175% for credit card transaction (October 04 and 2013, 2013).

The countermeasures to prevent fraud can be divided into two categories: prevention, which involves preventing the fraud at the source, and detection, which is the action taken after the incident has occurred. Objective of any system for detecting credit card fraud is to spot suspicious activity and report it while allowing regular transactions to go automatically. Financial institutions have been entrusting rule-based systems such as Address Verification System (AVS) and Card Verification System (CVM), which use rule sets created by specialists for years.

Table of Previous Published Works

ARTICLE TITLE	PUBLISHER	DATA SIZE (ENTRIES)	MODELS	FINDINGS
CREDIT CARD FRAUD DETECTION WITH MACHINE LEARNING (HAN, 2022)	Cardiff University	284,807	LR, K-NN, SVM, DT and Catboost	Best performing model was the LR model with 97.92% accuracy, this was achieved with the implementation of oversampling.
CREDIT CARD FRAUD DETECTION - MACHINE LEARNING METHODS (VARMEDJA ET AL., 2019)	IEEE 2019		LR, RF, Naive Bayes and Multilayer Perceptron	Results indicate that each algorithm can accurately detect credit card fraud. Also utilised SMOTE as a means of oversampling.
FRAUD DETECTION IN CREDIT CARDS USING LOGISTIC REGRESSION (ALENZI AND O, 2020)	International Journal of Advanced Computer Science and Applications	284,807	LR, SVM	Didn't utilise oversampling and implemented voting classifiers to boost model performance.

Table 1 : Previous Published Works

Dataset

The dataset used for this investigation is available on Kaggle and is the Machine Learning ULB Dataset. The dataset contains the transactions of European cardholders in September 2013 (www.kaggle.com, 2017). The dataset has transactions of 2 days, containing 284,807 transactions, of which 492 are labelled as fraud (www.kaggle.com, 2017). *Because the positive class accounts for only 0.172% of total transactions, the dataset is significantly unbalanced.*

The table below indicates the metadata for the dataset (www.kaggle.com, 2017):

Columns	Description	Type	No. of Columns
Time	Seconds Elapsed between each successive transaction. (Not transformed by PCA)	Integer	1
V1 to V28	PCA Components	Double	28
Amount	Amount spent by cardholder (in euros) (Not transformed by PCA)	Double	1

Class	Contains only two values: 0 -Non-Fraud Transaction 1-Fraud-Transaction (Not transformed by PCA)	Integer	1
--------------	--	---------	---

Table 2 : Metadata from Dataset

The columns in the dataset are, V1 to V28, Time, Amount, and Class (Non-PCA values) due to data privacy issues. V1 to V28 are the numerical variables that are the result of PCA (Principal Component Analysis). The key variables will be the Time and Amount, due to investigation into industry standard practices (Chuprina, 2021).

Methods

The business analytics tasks will follow the Cross Industry Standard Process for Data Mining (CRISP-DM) approach, CRISP-DM is a set of guide rails that helps plan, organize, and implement a machine learning project (knowledgeburrow.com, n.d.). The key phases include Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation, and Deployment (Rodrigues, 2020). Using this knowledge, an investigation into the potential links, correlations, and patterns in the data.

Below is a breakdown of the BA tasks in series, to implement the desired credit card fraud detection system:

1. Set up the environment by importing data and libraries.
2. Explore the dataset to see its variables and their relationships.
3. Clean data of any missing values, outliers, and duplicates.
4. Implement various models for the purpose of analysis.
5. Address class imbalance to avoid overfitting.
6. Split data using the train test method and cross-validate.
7. Reduce dimensionality using the best-suited method.
8. Implement the logistic regression machine learning model.
9. Hyper tune the parameters of the model to achieve ideal performance
10. Evaluate model performance.

Data Exploration

It is preferable to visualize the data first before getting into implementation. It can be depicted that the data is in terms of the correlation between variables, where correlation refers to the variables' mutual relationship. In general, feature variables that have a higher correlation with response variables have a greater impact during the training phase (Bruce and Bruce, 2017).

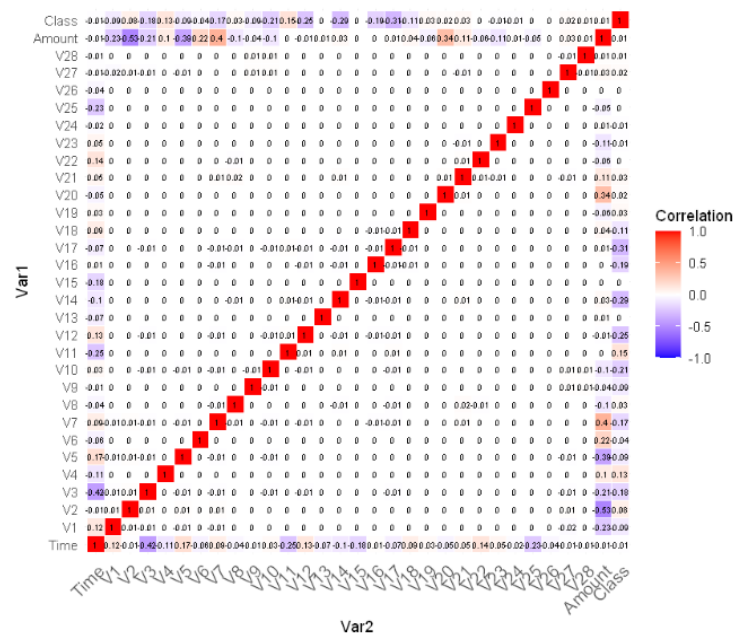


Figure 1 : Correlation matrix showing the correlations in the data

Figure 1 depicts a correlation matrix that explains the pairwise correlation between each variable. The correlation matrix demonstrates that there is no association between the V1 through V28 main components. However, as it can be seen, the response variable 'Class' has some positive and negative correlations with the principal components, but not with 'Time' and 'Amount'.

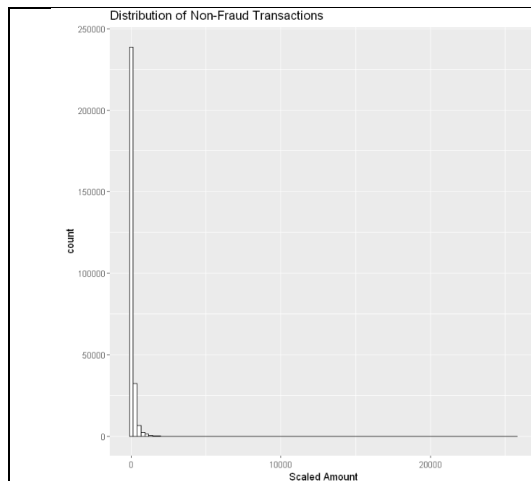


Figure 2 : Distribution of Non-Fraud Transactions

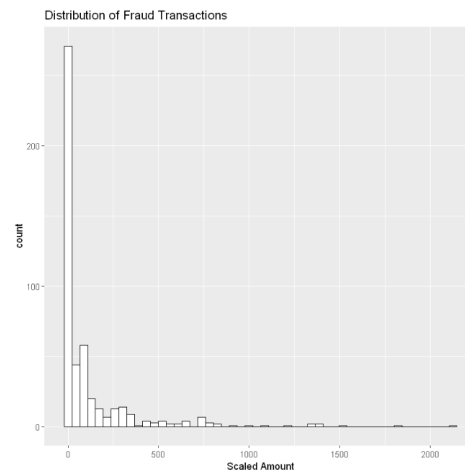


Figure 3: Distribution of Fraud Transactions

First, the dataset will be split by class type, fraudulent and non-fraudulent transactions. The histograms are then plotted side by side to look for any odd behaviour. As a result, the non-fraud transactions were strongly skewed to the right, making it difficult to compare the plots. Hence, logarithmic transformation has been used to tackle this problem, making it easier to identify and hence evaluate any parallels and differences. In Fig 2 and Fig 3, It is evident that both distributions follow a similar pattern, with most transactions falling toward the bottom of the graph. Even with maximum values averaging USD 20,000, it remains consistent with the mean value of USD88.

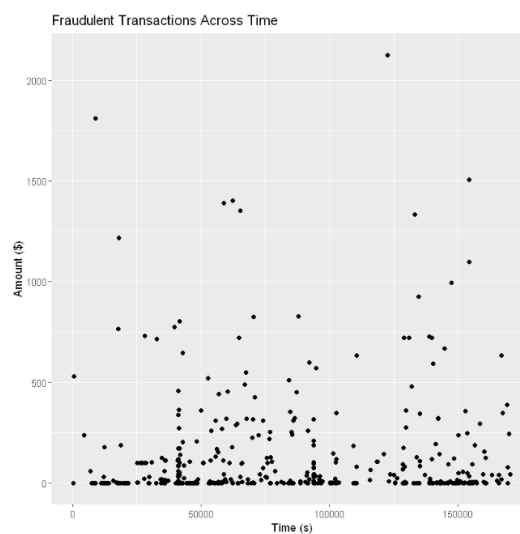


Figure 4 : Fraudulent Transactions over time

Next, transaction timing will be examined to see if anything out of the ordinary occurs. Fraud dataset will be used and will create a scatter plot based on it. There does not appear to be a clustering pattern on a time interval in Fig 4. As a result, it is visible that fraud occurred at random over time.

Data Preparation

Data preparation is the process of handling, manipulating, and working with a dataset to best prepare the data to be analysed, this process is typically done via a data pipeline either on-premises or as a cloud-based solution. For this project, the entirety is completed as an on-premises solution. Data preparation can be broken down into 11 different sub-categories – this report will focus on Data Transformations, Class Balancing, Dimensionality Reduction, and Data Splitting (Pramoditha, 2021).

Data Transformation

The process of altering the format, structure, or values of data is known as data transformation; it's a vital step in any data analysis, not to mention it's the third step in CRISP-DM architecture (Stirrup and Ramos, 2017). In this project the aspects of data transformations implemented extend the works of normalization and data manipulation.

Normalization

Normalization eliminates raw data and numerous dataset difficulties by scaling the data to follow a normal distribution, which is proven to aide ML algorithms, in particular the ones' that have been selected in this paper (Gogia, 2019). By normalizing the dataset in the pre-processing phase to log-based 10, the dataset has skewed down to highlight on the relative changes from one data point to another. To implement a normalization method over the dataset, the scale package provided by R was used (www.rdocumentation.org, n.d.).

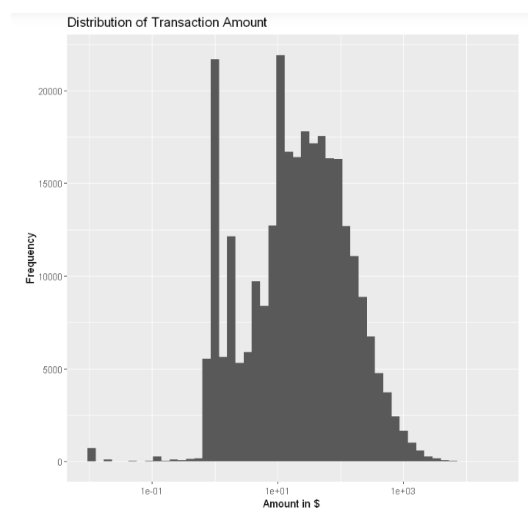


Figure 5 : Distribution of Transaction Amount

Amount appeared to be the most informative for the feature variable analysis. Therefore, Log scale was used to better comprehend the variable's distribution.

Data Manipulation

Data manipulation is the process of handling and manipulating the contents of the variables in the dataset to ensure that the dataset contains useful and necessary information. This process extends

aggregating data values and ensuring outliers don't exist (Kwak and Kim, 2017). For this project redundant values had to be taken out, and important categorical information needed aggregation in the form of factoring. This process eliminated 1081 rows from the finalized dataset, this includes duplicate and/or null values.

Factoring in R has been implemented with the aim of converting the numerical information in the dataset to a categorical format that the model can comprehend (Github.io, 2018). This, converts the existing integer variable into a factor form, which is a data type of R.

Class Balancing

Class Balancing entails adding a bias to choose more samples from one class than another to make up for an imbalance that would either already exist in the data or be likely to arise if a random sample were used (Analytics Vidhya, 2020). Seeing as the problem is for binary classification (fraud or not fraud), and the dataset being handled has a class imbalance ratio of 1:X, the two methods of class balancing that can be introduced are oversampling, and under sampling.

Most predictive models perform poorly in the context of uneven class distribution. As a result, some data pre-processing must be conducted before delivering data as input to the model. In the case of a class imbalance problem, such data preparation is carried out utilizing a data-level approach known as resampling.

Under sampling

Under sampling is defined in the ML environment as a class balancing tool that lowers the population of the larger class sizes to a more equal and balanced class distribution. The majority class is reduced in the under-sampling procedure to balance the dataset. The advantage of under sampling is when a large data set is used, it considerably increases runtime and reduces storage issues (Analytics Vidhya, 2020). The table below indicates the under-sampled dataset class distribution.

CLASS	CNT	FREQ
1	473	0.5097
0	455	0.4903

Table 3: Under Sampling Results

Oversampling

Oversampling, like under sampling is a class balancing tool, it acts in the inverse of an under-sampling method. This strategy is effective in the case of this project due they're only being two classes with a significant imbalance. It replicates the minority class observations to balance the ratio between the majority and minority samples (Analytics Vidhya, 2020). The table below indicates the oversampling class distribution.

CLASS	CNT	FREQ
1	283545	0.50026
0	283253	0.49974

Table 4: Oversampling Results

Hybrid sampling

Hybrid sampling, as the name suggests, finds a common point between over and under sampling, by oversampling the smaller class – 1, and under-sampling the larger class – 0 (Analytics Vidhya, 2020). The table below indicates the hybrid sampling class distribution.

CLASS	CNT	FREQ
1	142128	0.50093
0	141598	0.49907

Table 5:Hybrid Sampling Results

Dimensionality Reduction

Dimensionality reduction refers to methods for lowering the input variables in the training data. In doing so, a more simplistic but effective machine learning model structure can be achieved while avoiding overfitting. Three different Linear Algebra methods will be evaluated: PCA, SVD, and choose the one that captures the most variability in the datasets after reducing it to principal components.

Principal Component Analysis (PCA)

Principle Component Analysis (PCA) takes data with m-columns and projects it to a subspace with n-features ($n < m$) while retaining the important information from the original data; in other words, PCA seeks to locate the principal components as the name implies. Fig. Below shows that the explained variance for the PCA improves relatively less beyond 5 Principal Components. [PCA]

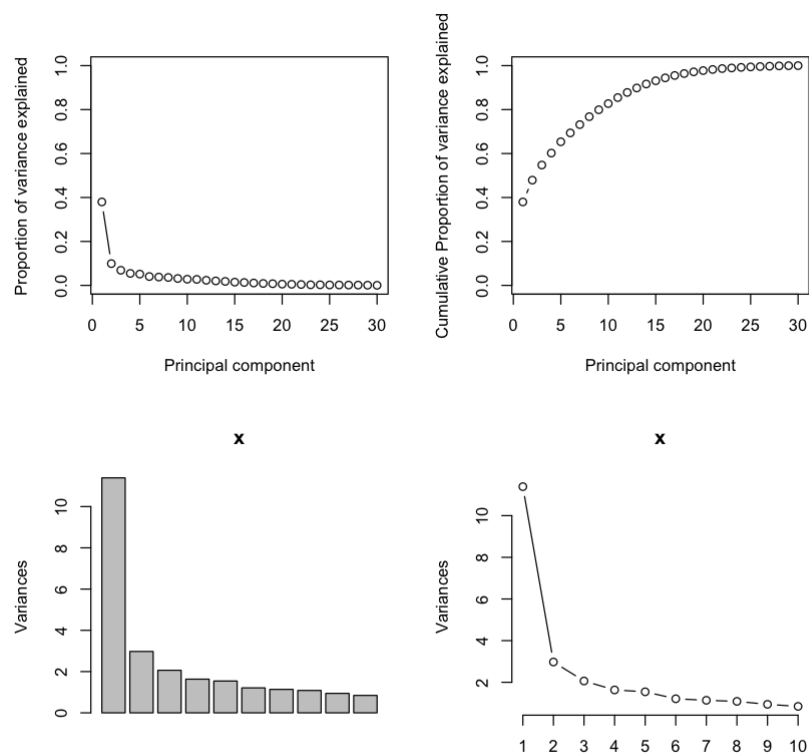


Figure 6: PCA Charts

Singular Value Decomposition (SVD)

It is the process of decomposing a matrix into its constituent elements by factorizing it into three distinct matrices: $M=UVt$.

M: The original matrix

U: Left singular matrix (columns are left singular vectors) comprising MM^T 's eigenvectors

Σ : A diagonal matrix with singular (eigen)values

V: Right singular matrix (columns are right singular vectors) comprising M^TM 's eigenvectors. [SVD]

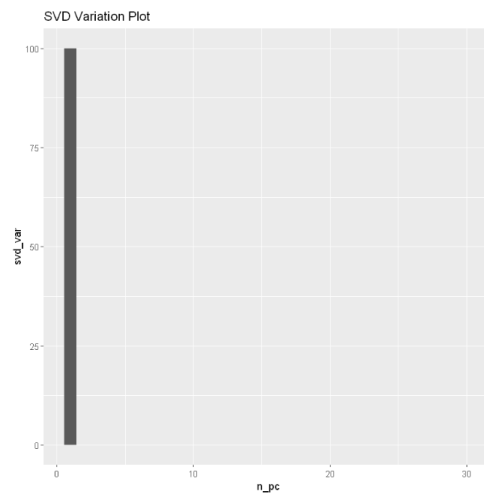


Figure 7: SVD Charts

There is virtually little improvement in the explained variance for PCA beyond 5 principal components, whereas the first column explained most of the variation using SVD. A smaller data frame from PCA will be employed, resulting in a more stable result.

Data Splitting

One of the final components in processing a dataset to be handled by an ML algorithm is to split the data into training, testing, and validation. The reasoning is to ensure that there is both enough data for the model to learn, and enough data to ensure if the model had identified patterns (Brownlee, 2020). Typically, datasets can be split into – training, testing and validation, however, for this project only training and testing sets were instantiated. The reason a validation set was deemed redundant was due to the availability of an explicit testing data. The split is 70:30, training to testing.

Machine Learning Models

Since the task at hand is of a classification nature this narrows down the potential models that can be implemented to those suited for classification. The task at hand also commends the use of supervised learning, due to the dataset being labelled, however explorations into the use of clustering as a means of classification using unsupervised learning is also considered. The machine learning models that will be used to carry out this task include those in supervised learning Logistic Regression, Decision Tree, and Random Forest.

Logistic Regression

Logistic regression is one of the most popular machine learning algorithms that is used for classification (Ray, 2019). Although the term 'regression' appears in the name, it is not a regression algorithm. Logistic regression has its name as it was built on another very popular machine learning algorithm, linear regression, which is used for regression problems. In logistic regression, the prediction is expressed in terms of probability of outcome belonging to each class.

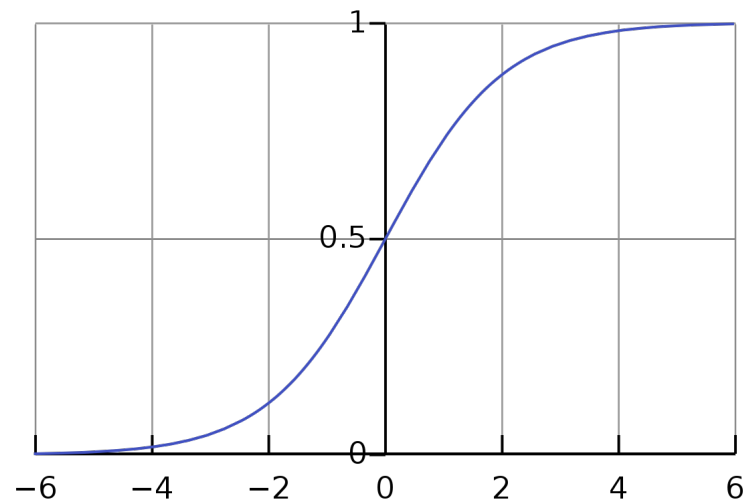


Figure 8 : Sigmoid function graph [Gan18]

The reason for using logistic regression to develop the classifier is due to its effectiveness in detecting fraud based on its ability to segregate data from different binary classifications. (Ray, 2019)

Decision Tree

Decision tree is a supervised learning algorithm, which can be used for both regression and classification (Khare and Subramania, 2010). But it is mostly used in classification problems. It is composed of several internal nodes where each node represents a test in an attribute (e.g., whether tomorrow's weather is sunny, overcast, or rainy). Each branch in the tree represents the outcome of the test and leaf nodes represent the outcome (class label). It involves breaking down a training set into several subsamples.

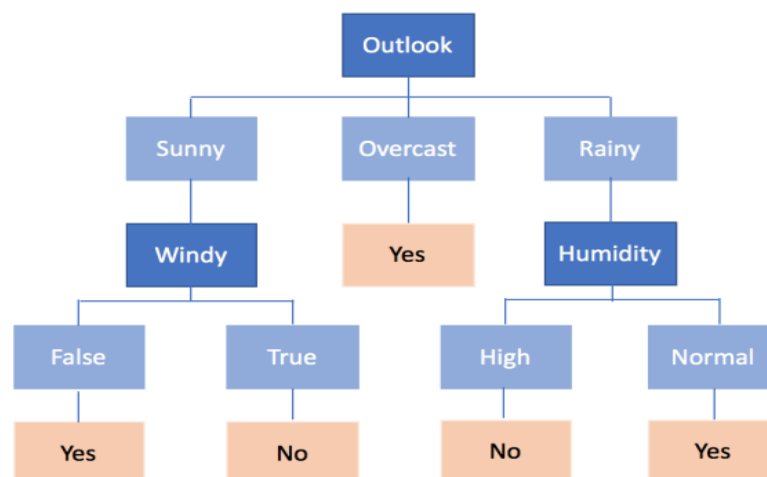


Figure 9 : An example of decision tree [Lib17]

In the field of Machine Learning, the decision tree model is a powerful tool. A single decision tree can encompass an entire financial service provider's credit card fraud dataset which is the reason behind considering the model for the problem statement (Ileberi, Sun and Wang, 2022).

Random Forest

Random forest algorithm by L. Breiman, 2001, has been successful as a general-purpose classification and regression method (Breiman, 2001). This approach uses several randomized decisions trees & aggregates their predictions by averaging, has shown excellent performance in the setting, and has a greater number of observations than the number of variables. It is applied to large-scale problems and is easily adapted to various ad-hoc learning tasks and returns measures of variable importance. The random forest model is implemented in the Random Forest Classifier.

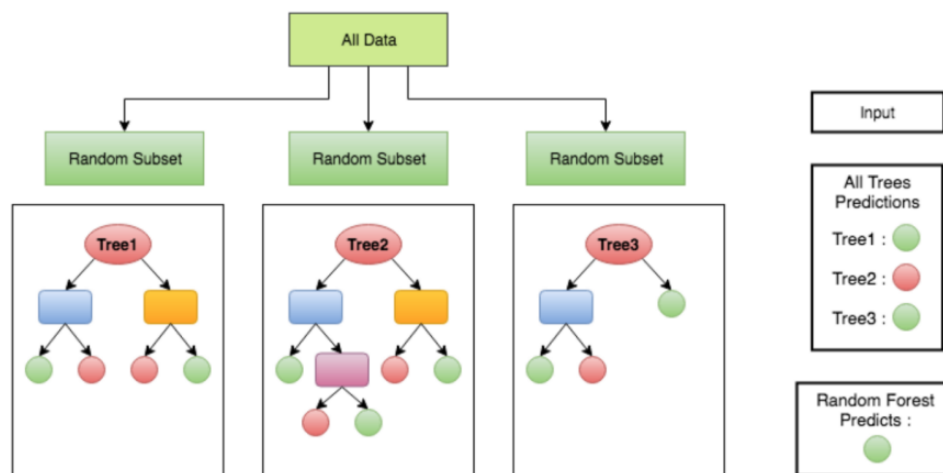


Figure 10 : An example of random forest [Lib17]

One of the most often used machine learning algorithms is Random forests. They are so effective because they provide strong prediction accuracy in general, little overfitting, and simple interpretability (Breiman, 2001). Therefore, it will be used in the analysis.

Hyperparameter Tuning

Hyperparameter tuning is the process of finding the perfect model architecture by adjusting the parameters that determine the model architecture, or hyperparameters. The hyperparameters for any given model vary, and as such the hyperparameter utilised for this project is K-Fold Cross Validation, efforts were also spent investigating Early Stopping, and Gradient Descent (Jordan, 2017).

K-Fold Cross Validation

K-Fold Cross Validation was used while implementing the models since it gives enough data for training while also leaving enough data for validation. The data is separated into k subsets in this case. The holdout approach has been repeated k times, with one of the k subsets serving as the test set/validation set and the other k-1 subsets forming a training set each time (Jordan, 2017). This validation method ensures the reliability of the models.

Evaluation Methods

Through various implementations of neural networks, and establishing a CRISP-DM approach, ultimately a Credit Card Fraud detection system will be implemented. To evaluate, and understand the performances of the various models, key decisions must be made to identify a suitable metric for

evaluating performance. The options available include, but aren't limited to – MAE, RMSE, F1 Scores. The key metric chosen is the Area under the Receiver-Operating Characteristics (AUC-ROC), which is a widely used metric for classification and regression tasks (Srivastava, 2019).

AUC-ROC

AUC-ROC is an evaluation matrix used in classification problems to know how well a model can categorize between 1 or 0 or "Yes" or "No". AUC represents the degree or measure of spread ability while ROC is the probability (Srivastava, 2019). AUC measures the area under the curve plotted from (0,0) to (1,1) against False Positive Rate and True Positive rate where :

$$\text{True Positive Rate} = \text{True Positive} / (\text{True Positive} + \text{False Negative})$$

$$\text{False Positive Rate} = \text{False Positive} / (\text{False Positive} + \text{True Negative})$$

The table below highlights how to comprehend an AUC graph, as well as the standardised method for evaluating AUC values in the table below.

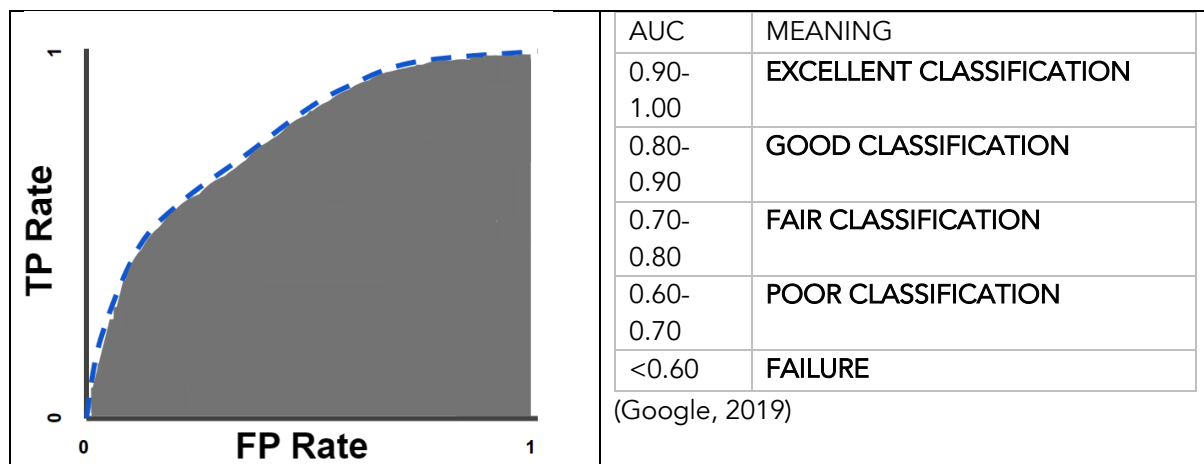


Figure 11 : AUC-ROC CHART

Table 6 : AUC Classification

Cohens(k) Value

Cohen's value or k value is used to measure the inter-rater reliability for categorical items. K values take the possibility of agreement occurring by chance. The k value is said to be proportionate to the inter-rater reliability of model, ergo the higher the k value the better the model performed.





COHEN'S K	DESCRIPTION
0.81-1.0	ALMOST PERFECT
0.61-0.80	SUBSTANTIAL
0.41-0.60	MODERATE
0.21-0.40	FAIR
0.0-0.20	SLIGHT
<0.00	POOR

Table 7 : Cohen's K Description

(Wikipedia Contributors, 2019)

Confusion Matrix

A confusion Matrix is the comparison of the predicted values against the actual value. it's a 2-D matrix where one says predicted other says actual values. The rows indicate the Predicted Class, and the columns indicate the Actual Class. From the confusion matrix, other metrics for evaluation can be derived, these include precision, f1 scores, recall, and others. The table below highlights how to comprehend a confusion matrix.

Two-class Binary model			
Predicted Class			
		A	B
Actual Class	A		
	B		

- True Positive (TP): Correctly Classified as target class.
- True Negative (TN): Correctly Classified as non-class target.
- False Positive (FP): Wrongly classified as target class
- False Negative (FN): Wrongly classified as non-class target

Table 8 : Confusion Matrix

(www.digitalocean.com, n.d.)

Changes to Project Plan

To structure and organize the project, the project management approach implemented was a Gantt chart, which helps to monitor and organize the structure of the project as it's being developed. The Gantt chart and table can be found in the appendix in greater detail, and this section will cover the changes made to the plan during this project.

- Due to time constraints and difficulties with overfitting the Random Forest model was dropped from consideration.
- Efforts into investigating the many potential parameters in hyperparameter tuning was cut short.

Results

Fully Processed Dataset

The dataset has been found to have no null values. Some duplicates were found and cleared. Outliers have also been taken care of. Class imbalance has been addressed using sampling and dimensionality reduced using principal component analysis PCA. The class variable has been factored ready for model implementation. Below is a peek at the fully processed dataset.

	PCA1	PCA2	PCA3	PCA4	PCA5
[1]	0.3797612388	0.0991361753	0.0687554909	0.0544240281	0.0514316083
[6]	0.0403052275	0.0378618068	0.0361171023	0.0313986756	0.0281089530
[11]	0.0273719740	0.0232984132	0.0203393451	0.0181000561	0.0147774863
[16]	0.0131634896	0.0108708691	0.0088399093	0.0077065926	0.0057233428
[21]	0.0050840958	0.0040682542	0.0027419093	0.0027070933	0.0020184762
[26]	0.0017903686	0.0015175157	0.0013806036	0.0008735662	0.0003263323

Table 9 : Fully Processed Dataset

LR Implementation

Logistic Regression	Accuracy	Precision	K-Value	F1 Score
Under Sampling	0.91	0.99	0.78	0.88
Over Sampling	0.89	0.98	0.83	0.908
Hybrid Sampling	0.91	0.989	0.83	0.908

Table 10: Logistic Regression Results

Logistic Regression Results for Under-sampled Data

Evaluation

The logistic regression model performed well, when under sampling was used, which is shown in above table. Even though the K-value of 0.78 is not a very good score, precision can't be neglected. Also, Figure 12 below shows AUC curve in which AUC is 0.949, which is good. Table 11 shows the confusion matrix of the logistic regression model when the under-sampling method is used.

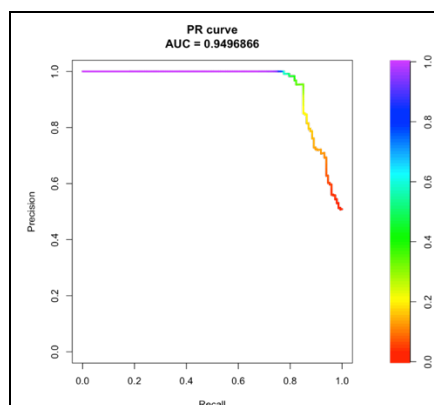


Figure 12: PR curve-LR under sampling

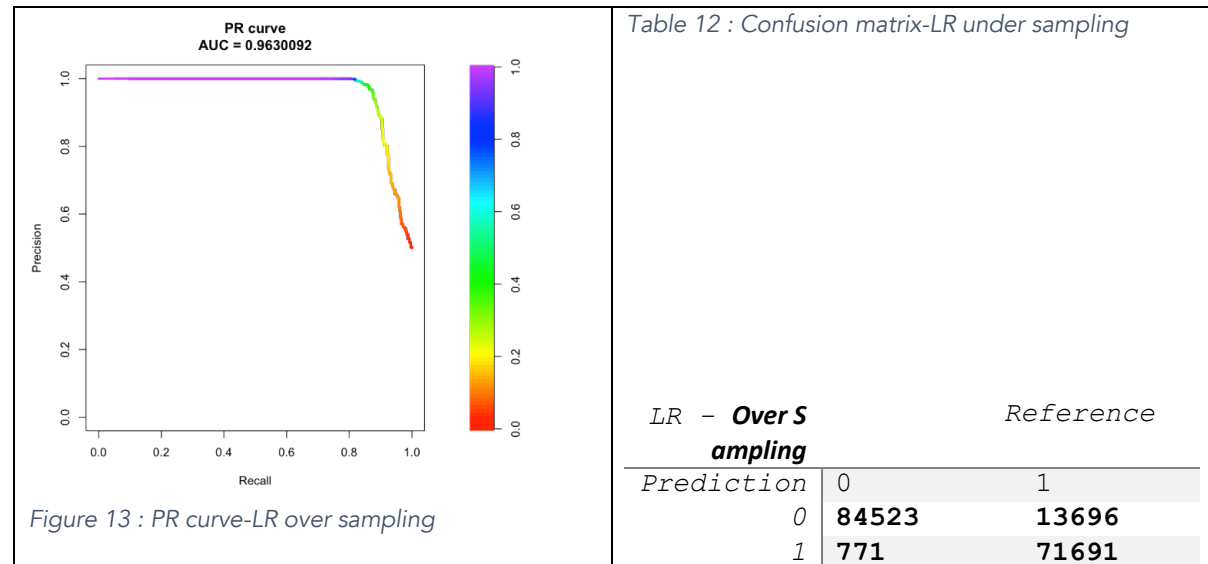
Table 11: Confusion matrix-LR under sampling

LR - UNDER SAMPLING		REFERENCE	
PREDICTION		0	1
0		141	30
1		1	117

Logistic Regression results for Oversampled data

Evaluation

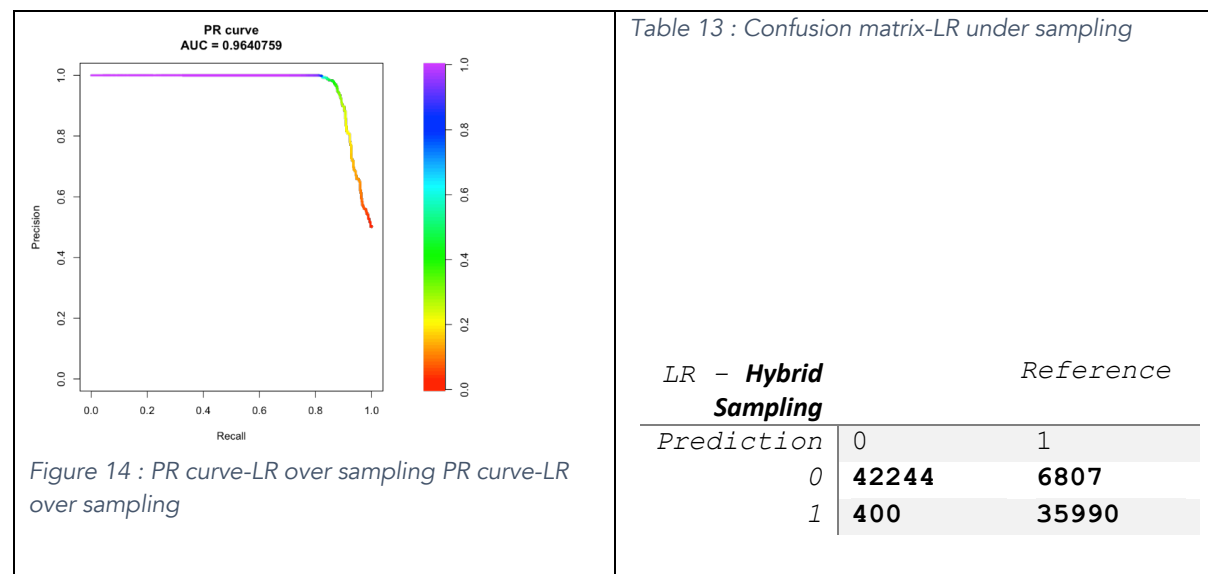
The table above shows the evaluation metrics of the logistic regression when the over sampling method is used. Similarly, figure 13 and table 12 shows the AU curve and confusion matrix respectively. The logistic regression with random oversampling performed very well in terms of precision and performed better in terms of k-value, thus giving a good f1 score of 0.908.



Logistic Regression Results for Hybrid Sampled Data

Evaluation

The table above shows the evaluation metrics of the logistic regression when a combination of both over sampling and under sampling is used. figure 14, and table 13 show the AU curve and confusion matrix respectively. Therefore, using the combination of over sampling and under sampling also improved the model performance, as the f1 score was 0.908 and the k-value reaches 0.83.



DT Implementation

<i>Decision Tree</i>	<i>Accuracy</i>	<i>Precision</i>	<i>K-Value</i>	<i>F1 Score</i>
<i>Under Sampling</i>	0.88	0.98	0.77	0.879
<i>Over Sampling</i>	0.91	0.969	0.83	0.86
<i>Hybrid Sampling</i>	0.92	0.97	0.84	0.91

Table 14 :Decision Tree Implementation

Decision Tree Results for Under-sampled Data

Evaluation

The table above shows the evaluation metrics of the Decision Tree when under sampling is used. figure 15 and table 15 show the AU curve and confusion matrix respectively. Under sampling approach, when used with the Decision Tree, it didn't perform well in terms of k-value, but the result was good in terms of precision. It happened to be a similar case when compared with Under sampling for Logical Regression but with low accuracy.

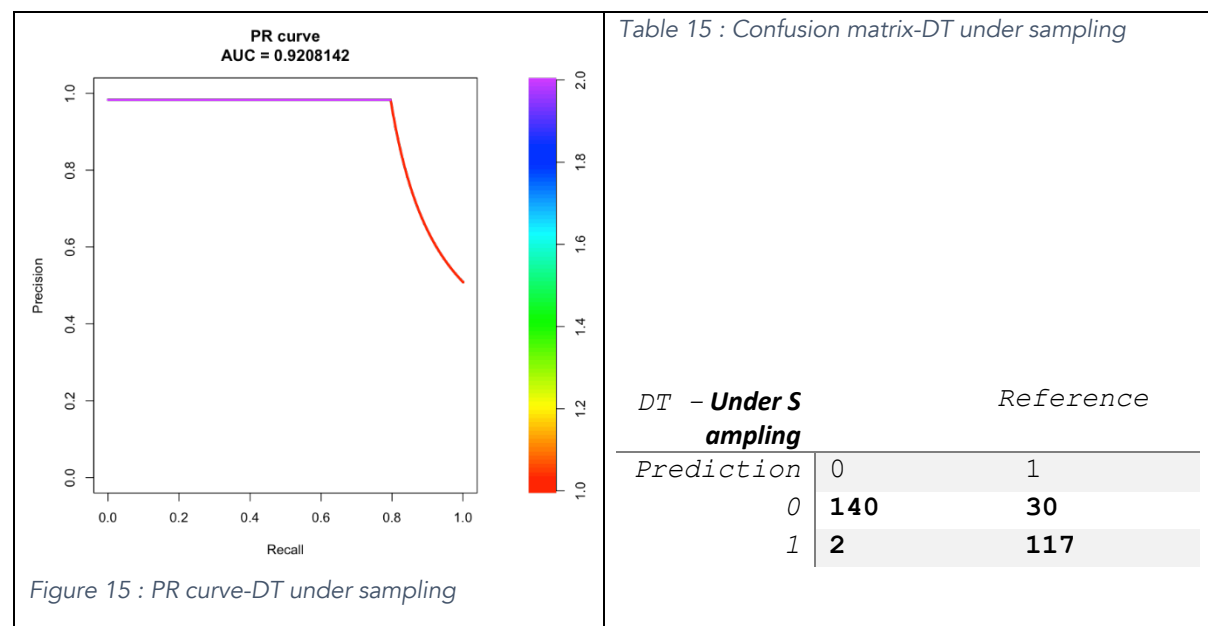
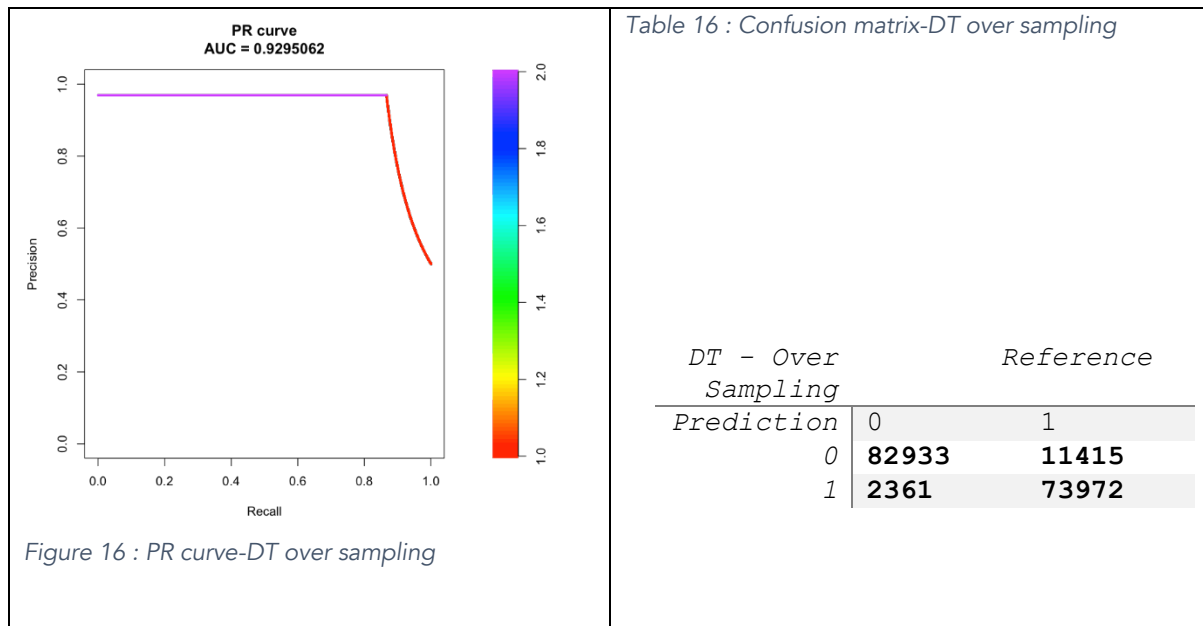


Figure 15 : PR curve-DT under sampling

Decision Tree results for Oversampled Data

Evaluation

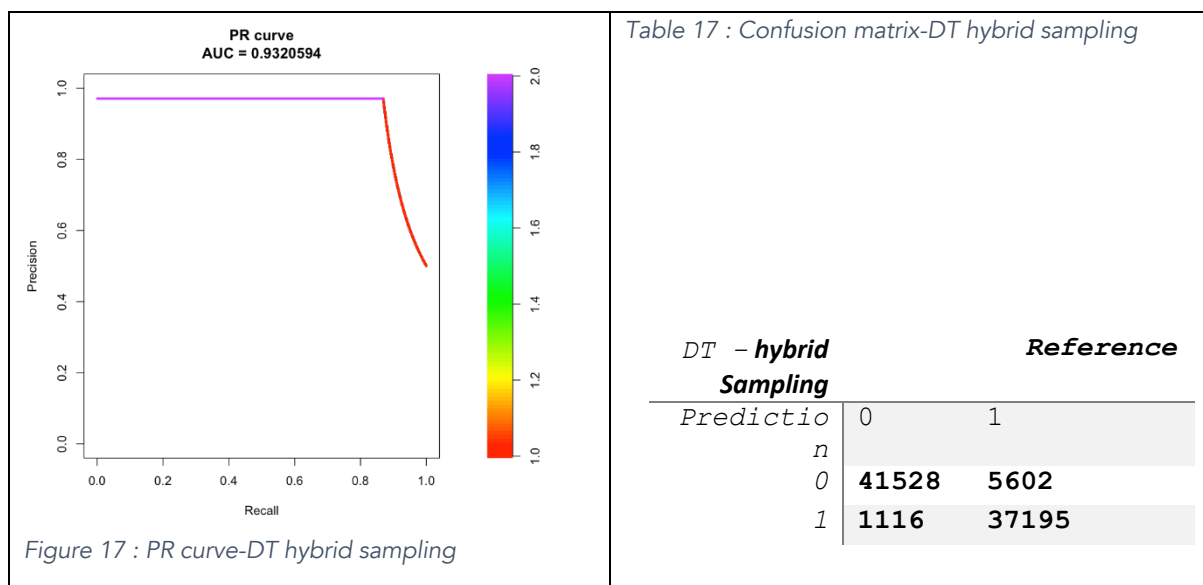
The table above shows the evaluation metrics of the Decision Tree when random over sampling is used. figure 16, and table 16 show the AU curve and confusion matrix respectively. Similar to under sampling, the precision score was high and the F1 score was decent, k-value has also been improved but accuracy a bit higher than that.



Decision Tree Results for Hybrid Sampled Data

Evaluation

The table above shows the evaluation metrics of the Decision Tree when a combination of Under and over sampling is used. figure 17 and table 17 show the AU curve and confusion matrix respectively. The decision tree, when used with a combination of both, performed very well considering the precision score of 0.97 and k-value of 0.84, which can also be seen from the higher AUC score. The decision tree performed very similar when used with any resampling technique as compared to logistic regression.



Conclusions

Data Driven Conclusions

Model	Resampling Method	Precision	K Value	F1 Score	AUC
Logical Regression	Under Sampling	0.99	0.78	0.88	0.94
	Over Sampling	0.98	0.83	0.908	0.96
	Hybrid Sampling	0.989	0.83	0.908	0.96
Decision Tree	Under Sampling	0.98	0.77	0.879	0.92
	Over Sampling	0.969	0.83	0.91	0.92
	Hybrid Sampling	0.97	0.84	0.91	0.93

The above table 18 clearly shows the logical regression performed better than the decision tree model considering their overall f1 score, precision, k-value & AUC score. Both models, when used under sampling, gave good precision but were not very well in terms of k-value. Both logical regression and decision tree performed better in terms of f1 score and k-value when over sampling was used. When the hybrid combination of over sampling and under sampling was applied with logical regression, it gave balanced precision & k-value scores of 0.98 & 0.83 and AUC score of 0.96. Whereas when the same resampling technique was used, the decision tree failed to improve the AUC score. Also, there was very little difference in the precision, k-value & f1 scores between the models when applied with hybrid sampling.

When providing input data of a highly unbalanced class distribution to the predictive model, the model tends to be biased toward most samples. As a result, it tends to misinterpret a fraudulent transaction as a genuine transaction (Gandler, 2020). To tackle this problem, a data level approach was implemented which includes various resampling techniques namely, under sampling, oversampling, and a hybrid resampling approach of over sampling and under sampling. Logical regression and Decision tree models were used for the comparison. After analysing both models using resampling techniques. The comparison results revealed that the Logistic Regression in combination with a hybrid resampling approach performed better than the decision tree model.

Having completed multiple variations of credit card fraud 'detectors', there is aspects of the project that can be improved on in the future, these extend the choice of the models' already implemented, other models that can be considered, as well as other parameters that may be considered. This also provides an opportunity to reflect on the challenges faced during this project, and the potential to rectify them in the future.

Random forest was used while implementing the models but because of the overfitting issues and time constraints, it hasn't been implemented (Breiman, 2001). Therefore, future work will include a few more models such as Random Forest and XGboost as these are tree-based models and proved to show better results for classification as well as regression problems. Other resampling techniques

such as SMOTE over sampling method that synthesizes new plausible examples in the minority class and class links removal for identifying pair of nearest neighbours in the dataset that has different classes, which will result in better sampled data, and hyperparameters such as learning rate, epochs, maximum depth, etc. for models will also be implemented to improve the overall performance of models.

The biggest worry throughout the model development and testing phase was the risk of overfitting. Overfitting is defined as "the production of an analysis that corresponds too closely or exactly to a particular set of data and may therefore fail to fit to additional data or predict future observations reliably" (dbpedia.org, n.d.). The risk of overfitting occurred if the model architecture was too advanced for the problem at hand, as well as some models – Random Forest, resulting in models with near-zero loss, or impossibly high accuracies on training data. To help mitigate the role of overfitting more efforts can be spent investigating as such, with a particular focus on either using a new bigger dataset, or by implementing alternate feature selection processes (IBM Cloud Education, 2021). For this project to really push the limits, it would've been interesting to utilise an alternate dataset, as the existing dataset has been investigated prior.

Business Driven Conclusions

The purpose of this investigation was to understand the extent to which credit card fraud transactions can be detected. This thought was kept in mind through all stages of the CRISP-DM cycle, the pursuit to solve the issue of credit card fraud detection ended up being fruitful as indicated by the results shown above. With this knowledge in hand, businesses, in particular Banks, should be able to implement variations of the proposed code to identify and classify fraudulent transactions to a high degree, furthermore, it can be leveraged to extend more protection and insight in the field of risk detection (Frösdrom, 2022).

Improvements that can be made from a business point of view include – extending the model to perform forecasting tasks, the handling of real-time data, transferring the solution to a cloud-based platform, and many more. To perform forecasting tasks, would mean the implementation of a temporal aspect as well as investigating the model architecture that best suits a forecasting problem, like ARIMA or LSTMs (Siarni-Namini, Tavakoli and Siarni Namin, 2018).

An alternative work that can extend the existing platform would be to take the solution to the cloud. This would entail creating a data pipeline that handles the streaming of new data, pre-processing the incoming data, updating the model with the new data, and predicting using the model in real-time.

The proposed architecture could be completed with the help of Google Cloud Platforms (GCP) which leverages existing cloud technologies from google. The pipeline would have to implement Pub/Sub for data streaming, Cloud Storage for data storage, Dataflow for managing the streaming data as well as processing the functions and the utilisation of the model, and Vertex AI for hosting and training the model.

The field of credit card fraud detection has come along way, this investigation highlighted the insights into what solutions can be developed using Machine Learning and data science to create positive imprint on the world, and as a group can be proud of the solution created.

Reference List

Alenzi, H.Z. and O, N. (2020). Fraud Detection in Credit Cards using Logistic Regression. *International Journal of Advanced Computer Science and Applications*, [online] 11(12). doi:10.14569/ijacsa.2020.0111265.

Analytics Vidhya. (2020). *Imbalanced Classification | Handling Imbalanced Data using Python*. [online] Available at: <https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/>.

Breiman, L. (2001). *Random Forests*. [online] Available at: <https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf>.

Brownlee, J. (2016). *Logistic Regression for Machine Learning*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/logistic-regression-for-machine-learning/>.

Brownlee, J. (2020). *Train-Test Split for Evaluating Machine Learning Algorithms*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/>.

Bruce, P. and Bruce, A. (2017). *Practical Statistics for Data Scientists*. 'O'Reilly Media, Inc.'

Chuprina, R. (2021). *Credit Card Fraud Detection: Top ML Solutions in 2021 - SPD Group Blog*. [online] Full-cycle Software Development Solutions. Available at: <https://spd.group/machine-learning/credit-card-fraud-detection> [Accessed 7 Nov. 2022].

dbpedia.org. (n.d.). *About: Overfitting*. [online] Available at: <https://dbpedia.org/page/Overfitting> [Accessed 28 Nov. 2022].

Frösdröm, H. (2022). *Deutsche Bank partners with Visa to prevent fraud in online retail*. [online] www.db.com. Available at: https://www.db.com/news/detail/20220922-deutsche-bank-partners-with-visa-to-prevent-fraud-in-online-retail?language_id=1 [Accessed 28 Nov. 2022].

Gandler, G.Z. (2020). *Training models on imbalanced data*. [online] Medium. Available at: <https://towardsdatascience.com/training-models-on-imbalanced-data-561fa3f842b5>.

Github.io. (2018). *Programming with R: Understanding Factors*. [online] Available at: <https://swcarpentry.github.io/r-novice-inflammation/12-supp-factors/index.html>.

Gogia, N. (2019). *Why Scaling is Important in Machine Learning?* [online] Analytics Vidhya. Available at: <https://medium.com/analytics-vidhya/why-scaling-is-important-in-machine-learning-aee5781d161a>.

Google (2019). *Classification: ROC Curve and AUC | Machine Learning Crash Course*. [online] Google Developers. Available at: <https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>.

Han, S. (2022). *Credit Card Fraud Detection With Machine Learning*. [online] Cf.ac.uk. Available at: https://pats.cs.cf.ac.uk/@archive_file?p=1859&n=final&f=1-report.pdf&SIG=04e2353a2ff38fdc8bb9e309ba2cc0d4da37276d5cfd6748c922decf449e44ee.

IBM Cloud Education (n.d.). *What is a Decision Tree | IBM*. [online] www.ibm.com. Available at: <https://www.ibm.com/uk-en/topics/decision-trees>.

IBM Cloud Education (2020). *What is Random Forest?* [online] www.ibm.com. Available at: <https://www.ibm.com/cloud/learn/random-forest>.

IBM Cloud Education (2021). *What is Overfitting?* [online] www.ibm.com. Available at: <https://www.ibm.com/cloud/learn/overfitting>.

Ileberi, E., Sun, Y. and Wang, Z. (2022). A machine learning based credit card fraud detection using the GA algorithm for feature selection. *Journal of Big Data*, 9(1). doi:10.1186/s40537-022-00573-8.

Jordan, J. (2017). *Hyperparameter tuning for machine learning models*. [online] Jeremy Jordan. Available at: <https://www.jeremyjordan.me/hyperparameter-tuning/>.

Khare, V.R. and Subramania, H.S. (2010). A Modular Decision-Tree Architecture for Better Problem Understanding. *Lecture Notes in Computer Science*, pp.647–656. doi:10.1007/978-3-642-17298-4_73.

knowledgeburrow.com. (n.d.). *What is CRISP-DM model? – KnowledgeBurrow.com*. [online] Available at: <https://knowledgeburrow.com/what-is-crisp-dm-model> [Accessed 7 Nov. 2022].

Kwak, S.K. and Kim, J.H. (2017). Statistical data preparation: management of missing values and outliers. *Korean Journal of Anesthesiology*, 70(4), p.407. doi:10.4097/kjae.2017.70.4.407.

October 04 and 2013 (2013). *Visa Announces Improved Payment Card Fraud Detection*. [online] Dark Reading. Available at: <https://www.darkreading.com/risk/visa-announces-improved-payment-card-fraud-detection> [Accessed 28 Nov. 2022].

Pramoditha, R. (2021). *11 Dimensionality reduction techniques you should know in 2021*. [online] Medium. Available at: <https://towardsdatascience.com/11-dimensionality-reduction-techniques-you-should-know-in-2021-dcb9500d388b>.

Ray, S. (2019). *A Quick Review of Machine Learning Algorithms*. [online] IEEE Xplore. doi:10.1109/COMITCon.2019.8862451.

Rodrigues, I. (2020). *CRISP-DM methodology leader in data mining and big data*. [online] Medium. Available at: <https://towardsdatascience.com/crisp-dm-methodology-leader-in-data-mining-and-big-data-467efd3d3781>.

Siarni-Namini, S., Tavakoli, N. and Siarni Namin, A. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. doi:10.1109/icmla.2018.00227.

Srivastava, T. (2019). *Evaluation Metrics Machine Learning*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>.

Stirrup, J. and Ramos, R.O. (2017). *Advanced Analytics with R and Tableau*. [online] Google Books. Packt Publishing Ltd. Available at: https://www.google.co.uk/books/edition/Advanced_Analytics_with_R_and_Tableau/JJZGDwAAQBAJ?hl=en&gbpv=1&dq=crisp-dm&printsec=frontcover [Accessed 28 Nov. 2022].

Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M. and Anderla, A. (2019). Credit Card Fraud Detection - Machine Learning methods. *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*. [online] doi:10.1109/infoteh.2019.8717766.

Whatman, P. (2021). *Credit card statistics 2021: 65+ facts for Europe, UK, and US*. [online] blog.spendesk.com. Available at: <https://blog.spendesk.com/en/credit-card-statistics>.

Wikipedia Contributors (2019). *Cohen's kappa*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Cohen%27s_kappa.

www.digitalocean.com. (n.d.). *Confusion Matrix in R | A Complete Guide | DigitalOcean*. [online] Available at: <https://www.digitalocean.com/community/tutorials/confusion-matrix-in-r>.

www.inscribe.ai. (n.d.). *Credit Card Fraud Detection: Everything You Need to Know*. [online] Available at: <https://www.inscribe.ai/fraud-detection/credit-fraud-detection> [Accessed 7 Nov. 2022].

www.kaggle.com. (2017). *Credit Card Fraud Detection*. [online] Available at: <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>.

www.rdocumentation.org. (n.d.). *scale function | R Documentation*. [online] Available at: <https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/scale>.

Appendix

Table of Figures

Figure 1 : Correlation matrix showing the correlations in the data.....	7
Figure 2 : Distribution of Non-Fraud Transactions.....	8
Figure 3: Distribution of Fraud Transactions.....	8

Figure 4 : Fraudulent Transactions over time	8
Figure 5 : Distribution of Transaction Amount.....	9
Figure 6: PCA Charts	11
Figure 7: SVD Charts.....	12
Figure 8 : Sigmoid function graph [Gan18].....	13
Figure 9 : An example of decision tree [Lib17].....	13
Figure 10 : An example of random forest [Lib17]	14
Figure 11 : AUC-ROC CHART	15
Figure 12: PR curve-LR under sampling	17
Figure 13 : PR curve-LR over sampling	18
Figure 14 : PR curve-LR over sampling PR curve-LR over sampling.....	18
Figure 15 : PR curve-DT under sampling	19
Figure 16 : PR curve-DT over sampling	20
Figure 17 : PR curve-DT hybrid sampling	20

Table of Tables

Table 1 : Previous Published Works.....	5
Table 2 : Metadata from Dataset.....	6
Table 3: Under Sampling Results.....	10
Table 4: Oversampling Results	10
Table 5:Hybrid Sampling Results.....	11
Figure 11 : AUC-ROC CHART	15
Table 7 : Cohen's K Description	15
Table 8 : Confusion Matrix.....	16
Table 9 : Fully Processed Dataset.....	17
Table 10: Logistic Regression Results	17
Table 11: Confusion matrix-LR under sampling.....	17
Table 12 : Confusion matrix-LR under sampling.....	18
Table 13 : Confusion matrix-LR under sampling.....	18
Table 14 :Decision Tree Implementation.....	19
Table 15 : Confusion matrix-DT under sampling	19
Table 16 : Confusion matrix-DT over sampling	20
Table 17 : Confusion matrix-DT hybrid sampling	20

Gantt Table

On Track: item under consideration has been fully planned and allocated for.

Low Risk: task at hand hasn't been fully planned for but seems feasible.

Med Risk: task at hand hasn't been planned for and seems challenging.

High Risk: task at hand hasn't been planned for and seems considerably more challenging.

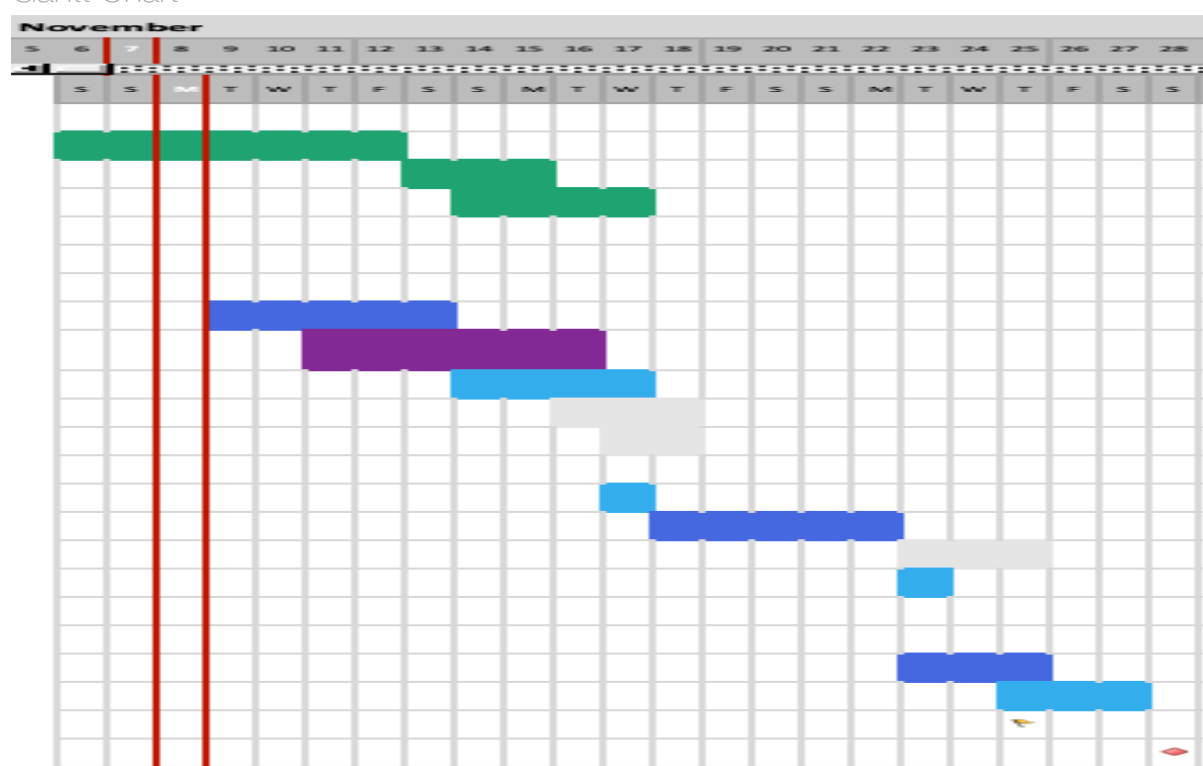
Report: Task at hand, is concerning only the report.

Milestone description	Category	Assigned to	Progress	Start	Days
-----------------------	----------	-------------	----------	-------	------

Planning					
Discuss and choose working datasets	On Track	All	100%	05/11/2022	7
Identify steps to complete project plan	On Track	Abubakar, Adarsh	85%	12/11/2022	3
Discuss and prepare report structure	On Track	All	50%	13/11/2022	4
Data Exploration and Pre-processing					
Explore dataset and identify patterns	Med Risk	All	60%	08/11/2022	5
Using identified knowledge, generate base model assumptions	High Risk	All	10%	10/11/2022	6
Pre-process and split the dataset	Low Risk	Abubakar, Adarsh, Piyush	0%	13/11/2022	4
Describe in the report exploration, pre-processing steps, and findings	Report	Adarsh, Raman	0%	15/11/2022	3
Identify suitable metrics for analysis based on initial model designs	Report	Adarsh, Raman	0%	16/11/2022	2
Modelling					
Start instantiating the model	Low Risk	All		16/11/2022	1
Achieve optimum model performance	Med Risk	Abubakar, Piyush		17/11/2022	5
Describe in report the model summary and architecture.	Report	Abubakar, Adarsh		22/11/2022	3
Ensure the code architecture and commentary are up to standard.	Low Risk	Abubakar, Piyush		22/11/2022	1
Evaluation					

Gather data for evaluating model performance	Med Risk	Piyush, Raman	22/11/2022	3
Plot, explore and analyse results	Low Risk	Piyush, Raman	24/11/2022	3
Describe and analyse findings in report.	Report	Abubakar, Adarsh	24/11/2022	1
Finalise report	Report	All	27/11/2022	1

Gantt Chart



Hybrid Sampling Training Results Logistic Regression

Call:
NULL

Deviance Residuals:

Min	1Q	Median	3Q	Max
-7.2669	-0.5119	0.0000	0.0165	3.1335

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	4.57489	0.04916	93.05	<2e-16	***
PC1	-2.86895	0.02388	-120.16	<2e-16	***
PC2	-0.33349	0.01531	-21.79	<2e-16	***
PC3	-0.60341	0.01503	-40.16	<2e-16	***
PC4	0.59263	0.01077	55.05	<2e-16	***

```
PC5          -0.42741      0.01152  -37.10   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 276379 on 199365 degrees of freedom
Residual deviance: 94670 on 199360 degrees of freedom
AIC: 94682
```

Number of Fisher Scoring iterations: 9

Hybrid Sampling Training Results Decision Tree

Call:

```
rpart(formula = Class ~ ., data = train, method = "class", control = rp
art.control(xval = 3))
n= 199366
```

	CP	nsplit	rel error	xerror	xstd
1	0.79877393	0	1.0000000	1.0000000	0.002243623
2	0.04106326	1	0.2012261	0.2012261	0.001348765
3	0.01000000	2	0.1601628	0.1601729	0.001216968

Variable importance

PC1	PC2	PC5	PC4	PC3
45	23	16	11	5

Node number 1: 199366 observations, complexity param=0.7987739

predicted class=1 expected loss=0.4991072 P(node) =1

class counts: 99505 99861

probabilities: 0.499 0.501

left son=2 (119432 obs) right son=3 (79934 obs)

Primary splits:

PC1 < 0.7457081 to the right, improve=66318.380, (0 missing)

PC2 < -0.4131688 to the right, improve=27337.560, (0 missing)

PC5 < -0.6075385 to the right, improve=15303.360, (0 missing)

PC4 < 1.040715 to the left, improve=11630.500, (0 missing)

PC3 < -0.4794189 to the right, improve= 3808.894, (0 missing)

Surrogate splits:

PC2 < -0.4489896 to the right, agree=0.801, adj=0.504, (0 split)

PC5 < -0.4509986 to the right, agree=0.745, adj=0.365, (0 split)

PC4 < 1.538478 to the left, agree=0.664, adj=0.161, (0 split)

PC3 < 0.8230545 to the left, agree=0.642, adj=0.106, (0 split)

Node number 2: 119432 observations, complexity param=0.04106326

predicted class=0 expected loss=0.16725 P(node) =0.599059

class counts: 99457 19975

probabilities: 0.833 0.167

left son=4 (110160 obs) right son=5 (9272 obs)

Primary splits:

PC4 < 1.125369 to the left, improve=6150.2550, (0 missing)

PC3 < -0.6611109 to the right, improve=2514.7690, (0 missing)

PC2 < -0.3275353 to the right, improve=1649.8230, (0 missing)

PC1 < 1.940952 to the right, improve=1417.7070, (0 missing)

PC5 < -2.005203 to the right, improve= 743.1476, (0 missing)

Surrogate splits:

PC3 < -0.6998333 to the right, agree=0.925, adj=0.029, (0 split)

```

PC1 < 1.566298    to the right, agree=0.924, adj=0.022, (0 split)
PC2 < 5.495216    to the left,  agree=0.922, adj=0.001, (0 split)

Node number 3: 79934 observations
  predicted class=1  expected loss=0.0006004954  P(node) =0.400941
  class counts:      48 79886
  probabilities: 0.001 0.999

Node number 4: 110160 observations
  predicted class=0  expected loss=0.1206972  P(node) =0.5525516
  class counts: 96864 13296
  probabilities: 0.879 0.121

Node number 5: 9272 observations
  predicted class=1  expected loss=0.2796592  P(node) =0.04650743
  class counts: 2593 6679
  probabilities: 0.280 0.720

n= 199366

node), split, n, loss, yval, (yprob)
* denotes terminal node

1) root 199366 99505 1 (0.4991071697 0.5008928303)
  2) PC1>=0.7457081 119432 19975 0 (0.8327500167 0.1672499833)
    4) PC4< 1.125369 110160 13296 0 (0.8793028322 0.1206971678) *
    5) PC4>=1.125369 9272 2593 1 (0.2796591890 0.7203408110) *
  3) PC1< 0.7457081 79934 48 1 (0.0006004954 0.9993995046) *
```

Oversampling Results

ROC-AUC - Logistic Regression
[1] 1

Precision, Recall and F1-Score - Logistic Regression
[1] "Precsion : 0.989359940382545"
[1] "Recall : 0.839600876011571"
[1] "F1-Score : 0.908349118461314"

Confusion Matrix - Logistic Regression
Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	84523	13696
1	771	71691

Accuracy : 0.9152
95% CI : (0.9139, 0.9166)
No Information Rate : 0.5003
P-Value [Acc > NIR] : < 2.2e-16

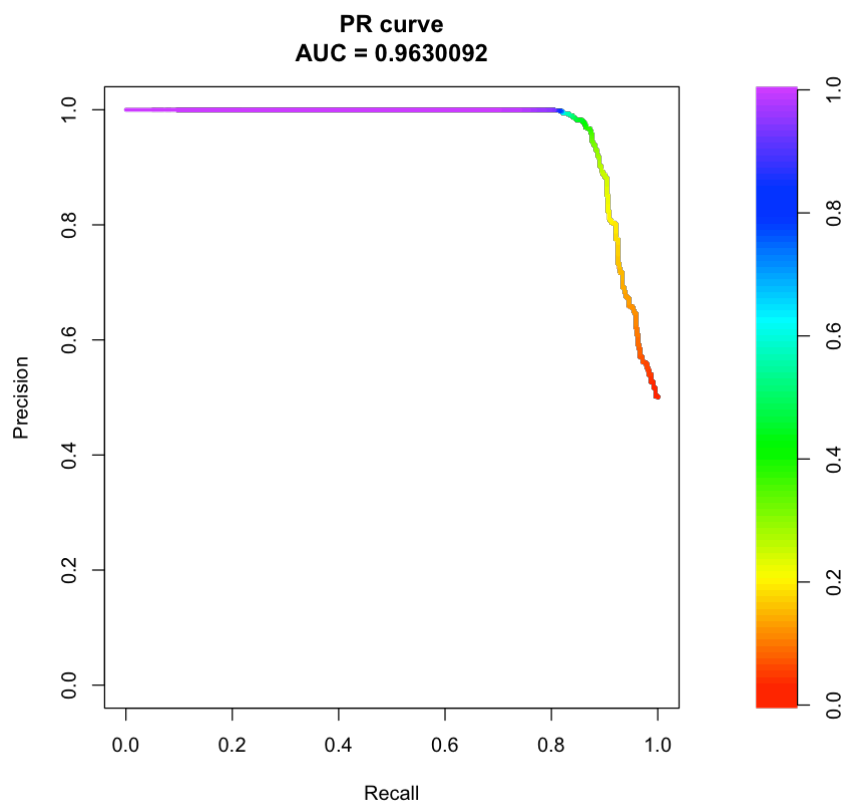
Kappa : 0.8305

Mcnemar's Test P-Value : $< 2.2e-16$

Sensitivity : 0.8396
Specificity : 0.9910
Pos Pred Value : 0.9894
Neg Pred Value : 0.8606
Prevalence : 0.5003
Detection Rate : 0.4200
Detection Prevalence : 0.4245
Balanced Accuracy : 0.9153

'Positive' Class : 1

ROC-AUC - Decisition Tree



[1] 1

Precision, Recall and F1-Score - Decisition Tree

[1] "Precsion : 0.969069733928969"
[1] "Recall : 0.866314544368581"
[1] "F1-Score : 0.864815730892901"

Confusion Matrix - Decisition Tree
Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	82933	11415
1	2361	73972

31

Monday, 28 November 2022

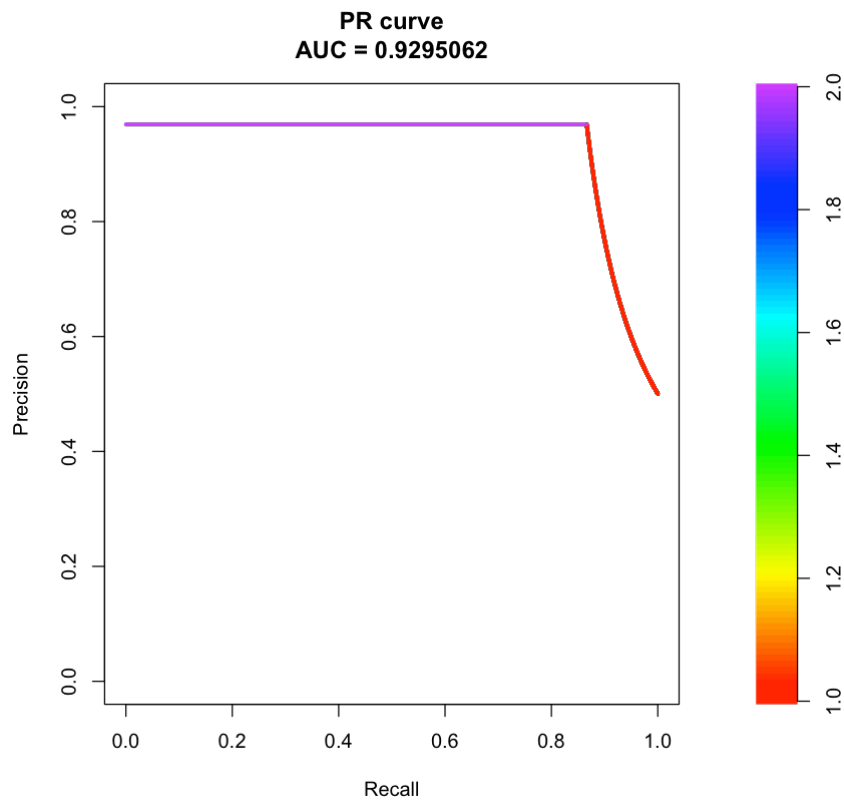
Accuracy : 0.9193
95% CI : (0.918, 0.9206)
No Information Rate : 0.5003
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.8386

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.8663
Specificity : 0.9723
Pos Pred Value : 0.9691
Neg Pred Value : 0.8790
Prevalence : 0.5003
Detection Rate : 0.4334
Detection Prevalence : 0.4472
Balanced Accuracy : 0.9193

'Positive' Class : 1



Undersampling Results

ROC-AUC - Logistic Regression
[1] 1

Precision, Recall and F1-Score - Logistic Regression


```
[1] "Precision : 0.991525423728814"
[1] "Recall : 0.795918367346939"
[1] "F1-Score : 0.883018867924528"
```

Confusion Matrix - Logistic Regression
Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	141	30
1	1	117

Accuracy : 0.8927
95% CI : (0.8512, 0.9259)
No Information Rate : 0.5087
P-Value [Acc > NIR] : < 2.2e-16

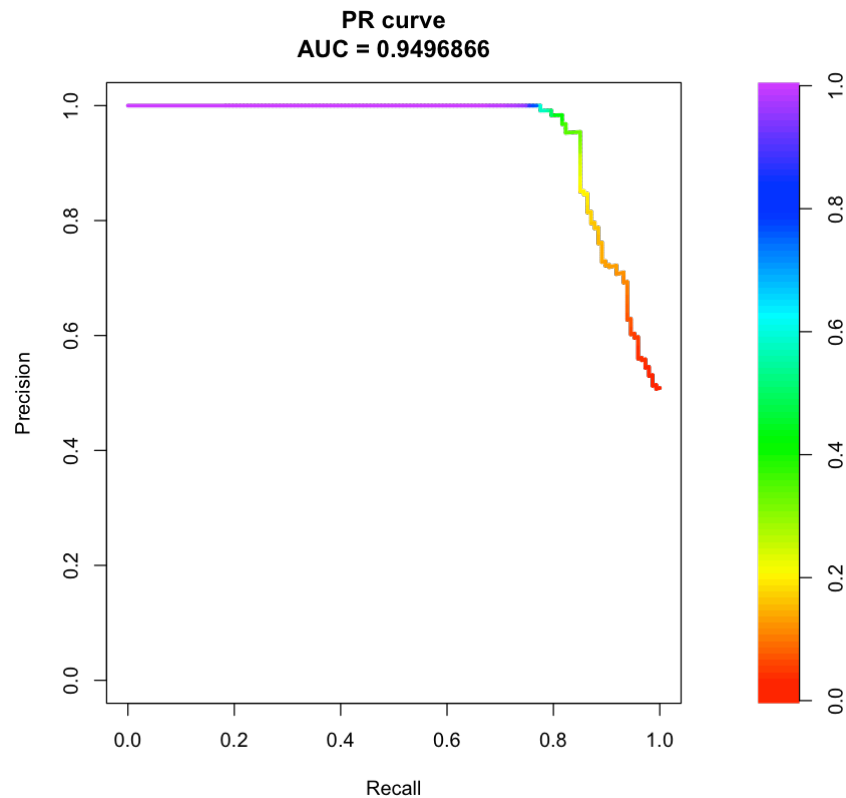
Kappa : 0.7861

McNemar's Test P-Value : 4.932e-07

Sensitivity : 0.7959
Specificity : 0.9930
Pos Pred Value : 0.9915
Neg Pred Value : 0.8246
Prevalence : 0.5087
Detection Rate : 0.4048
Detection Prevalence : 0.4083
Balanced Accuracy : 0.8944

'Positive' Class : 1

ROC-AUC - Decision Tree



```
[1] 1
```

Precision, Recall and F1-Score - Decision Tree

```
[1] "Precision : 0.983193277310924"
[1] "Recall : 0.795918367346939"
[1] "F1-Score : 0.879699248120301"
```

Confusion Matrix - Decision Tree
Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	140	30
1	2	117

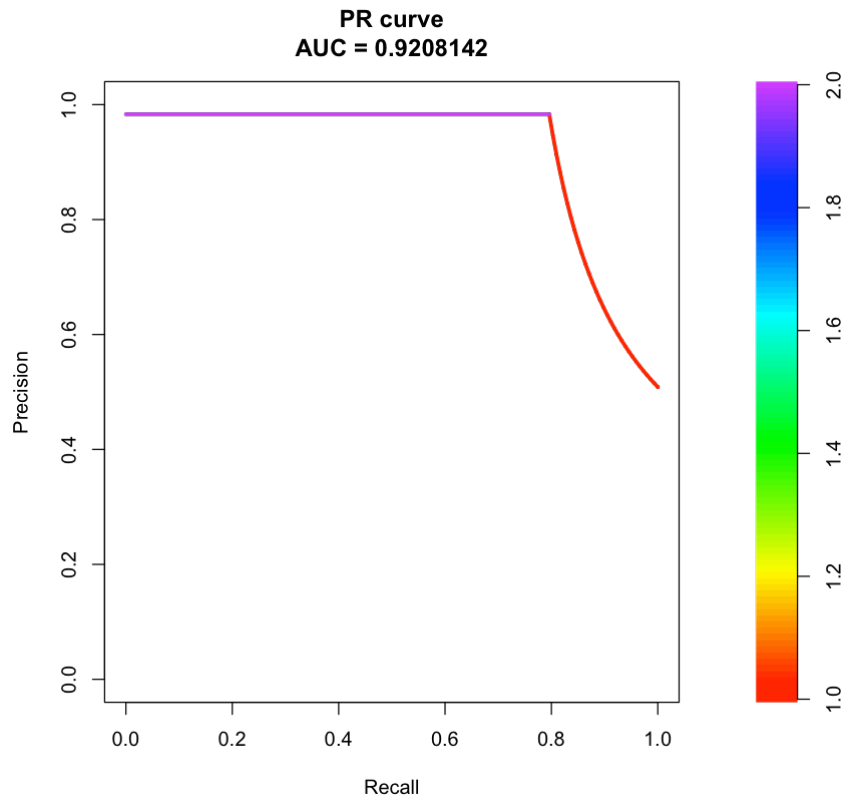
```
Accuracy : 0.8893
 95% CI : (0.8473, 0.923)
No Information Rate : 0.5087
P-Value [Acc > NIR] : < 2.2e-16
```

```
Kappa : 0.7792
```

```
Mcnemar's Test P-Value : 1.815e-06
```

```
Sensitivity : 0.7959
Specificity : 0.9859
Pos Pred Value : 0.9832
Neg Pred Value : 0.8235
Prevalence : 0.5087
Detection Rate : 0.4048
```

Detection Prevalence : 0.4118
 Balanced Accuracy : 0.8909
 'Positive' Class : 1



Hybrid Sampling Final Results

ROC-AUC - Logistic Regression
 [1] 1

Precision, Recall and F1-Score - Logistic Regression
 [1] "Precision : 0.989007969222314"
 [1] "Recall : 0.840946795336122"
 [1] "F1-Score : 0.908987586346244"

Confusion Matrix - Logistic Regression
 Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	42244	6807
1	400	35990

Accuracy : 0.9156
 95% CI : (0.9138, 0.9175)
 No Information Rate : 0.5009
 P-Value [Acc > NIR] : < 2.2e-16

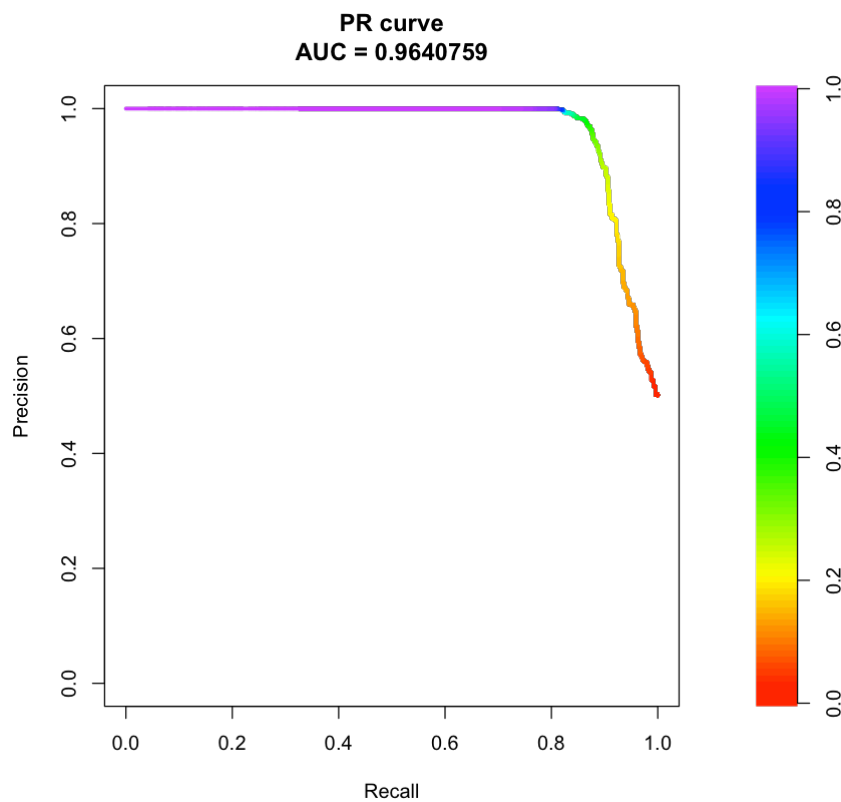
Kappa : 0.8313

McNemar's Test P-Value : $< 2.2e-16$

Sensitivity : 0.8409
Specificity : 0.9906
Pos Pred Value : 0.9890
Neg Pred Value : 0.8612
Prevalence : 0.5009
Detection Rate : 0.4212
Detection Prevalence : 0.4259
Balanced Accuracy : 0.9158

'Positive' Class : 1

ROC-AUC - Decision Tree



[1] 1

Precision, Recall and F1-Score - Decision Tree

[1] "Precision : 0.970869985121767"
[1] "Recall : 0.869102974507559"
[1] "F1-Score : 0.917172165507718"

Confusion Matrix - Decision Tree
Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	41528	5602

1 1116 37195

Accuracy : 0.9214
95% CI : (0.9195, 0.9232)
No Information Rate : 0.5009
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.8428

McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.8691
Specificity : 0.9738
Pos Pred Value : 0.9709
Neg Pred Value : 0.8811
Prevalence : 0.5009
Detection Rate : 0.4353
Detection Prevalence : 0.4484
Balanced Accuracy : 0.9215

'Positive' Class : 1

