Fraud Detection Individual Project

BC3409 AI in Accounting and Finance

Ng Chen Ee Kenneth

U1721316F

20 Sep. 20

Table of Contents

Introduction

Fraud detection is a set of activities undertaken to prevent money or property from being obtained through false pretences. Fraud detection is applied to many industries such as banking or insurance. In banking, fraud may include forging checks or using stolen credit cards. Other forms of fraud may involve exaggerating losses or causing an accident with the sole intent for the pay-out.

With an unlimited and rising number of ways someone can commit fraud, detection can be difficult to accomplish. Activities such as reorganization, downsizing, moving to new information systems or encountering a cybersecurity breach could weaken an organization's ability to detect fraud. This means techniques such as real-time monitoring for frauds is recommended. Organizations should look for fraud in financial transactions, location, devices used, initiated sessions and authentication system.

Project Objectives

The objective of this project is to predict if a transaction made is fraudulent and comparing the performance of classical machine learning models. Additionally, suggestions on how to improve the results as well as the business aspect of fraud detection are to be included.

Overview

Below is a summary of the steps taken to analyse the performance of the models. The report will elaborate on each step of the process in detail.

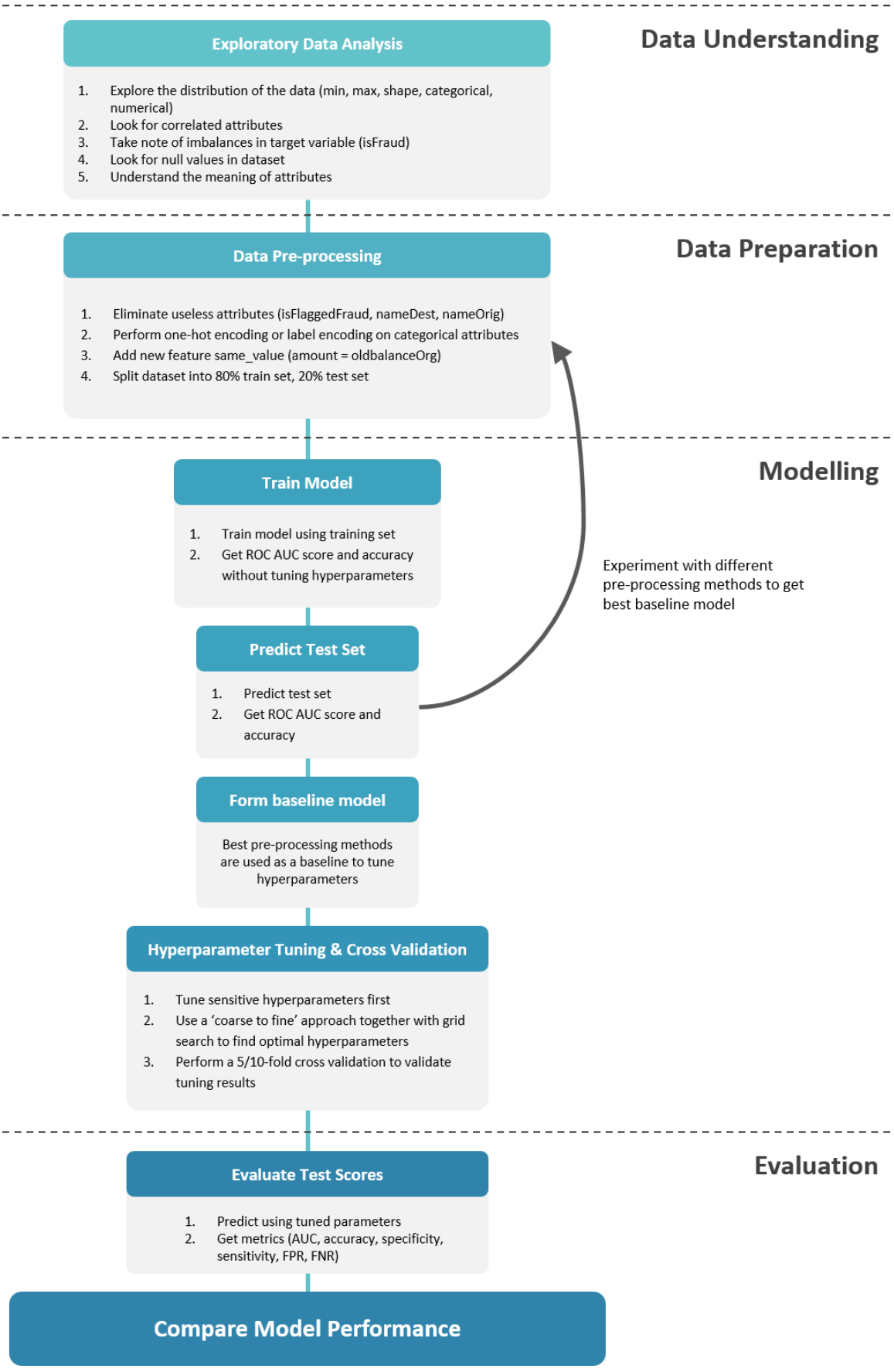

Fig 1. Overview of data mining process

Exploratory Data Analysis

Facets

Exploratory data analysis is an approach to analysing data sets to summarize their main characteristics. In this project, Facets is used to visualize the dataset. Facets is an open source project from Google Research that can visualize data easily by just uploading the csv file onto the site.

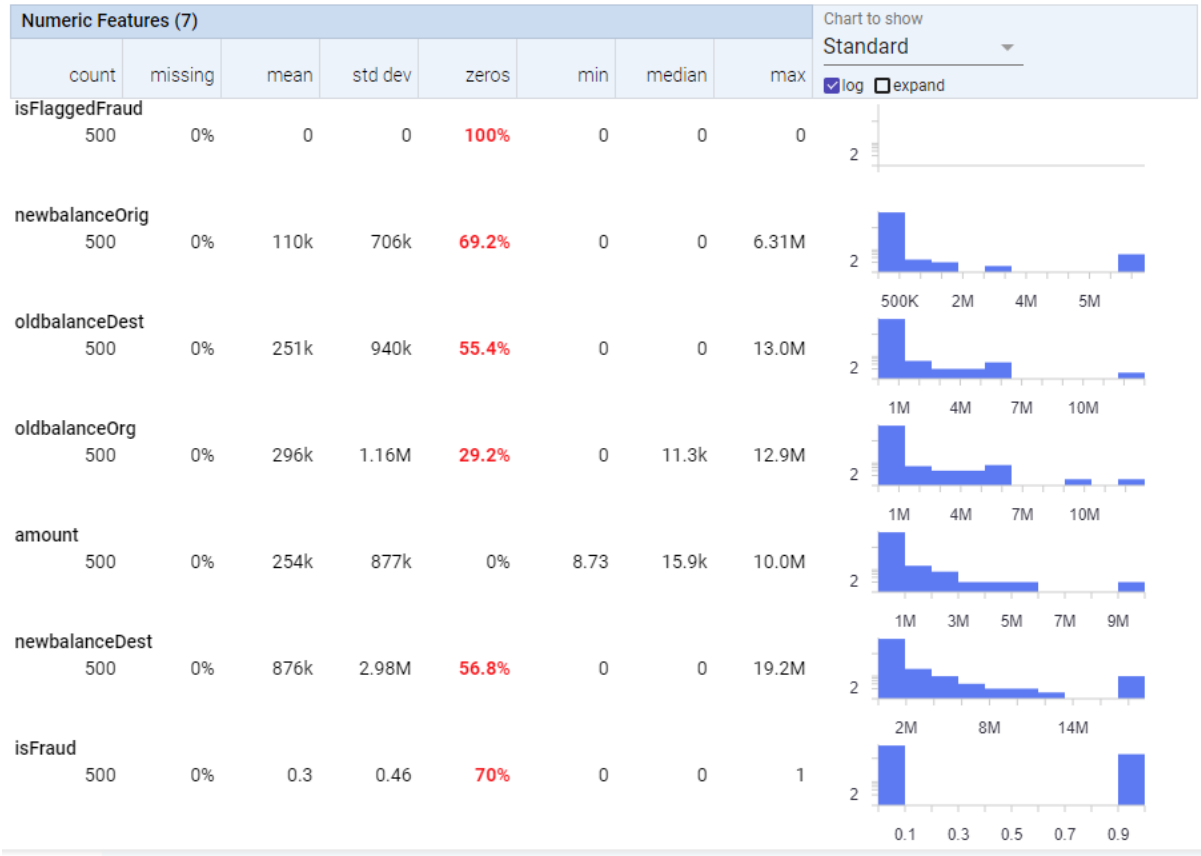

Fig 2. Screenshot of numerical features from Facets

Numerical Features

For numerical features, we can see that isFlaggedFraud is completely made of zeros. We can discard this column as it would not prove to be useful for prediction. There are also no missing values in the dataset. We can also see that there are 30% of the transactions that are fraudulent.

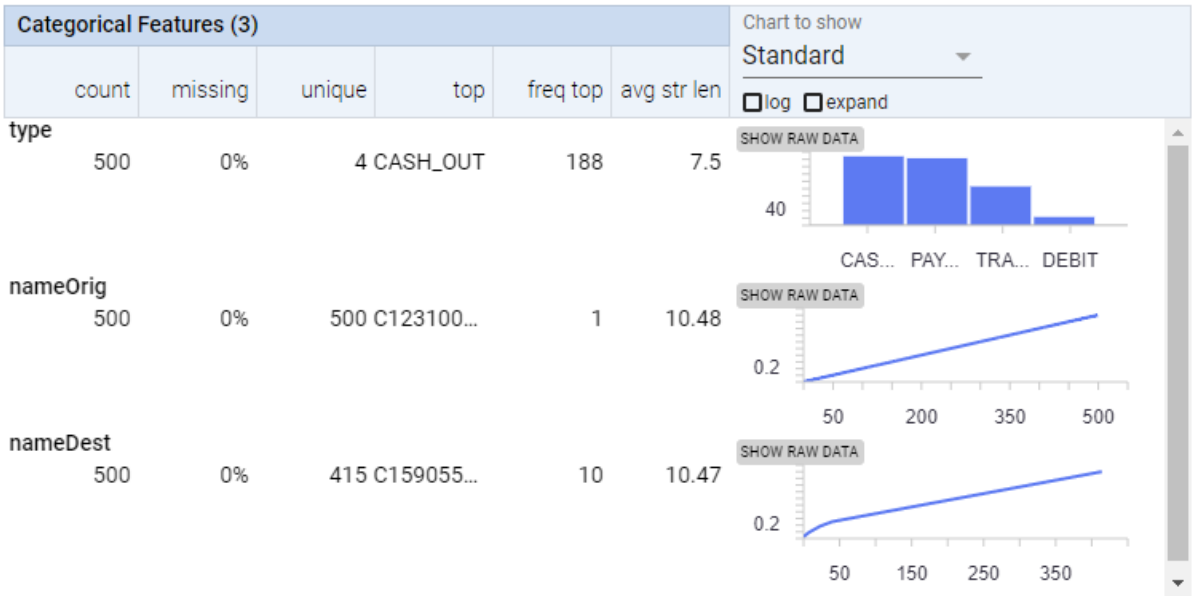

Fig 3. Screenshot of categorical features from Facets

Categorical Features

For categorical features of the dataset, we can observe that nameOrig is entirely unique and serves as an identifier for the transaction. nameDest tells us that some of the transactions may be sent to the same person. As such, it may be useful in predicting fraud.

Fig 4. Type of transaction against transaction count

When plotted against the type of transaction, it becomes clear that fraudulent transactions appear only for CASH OUT and TRANSFER type transactions.

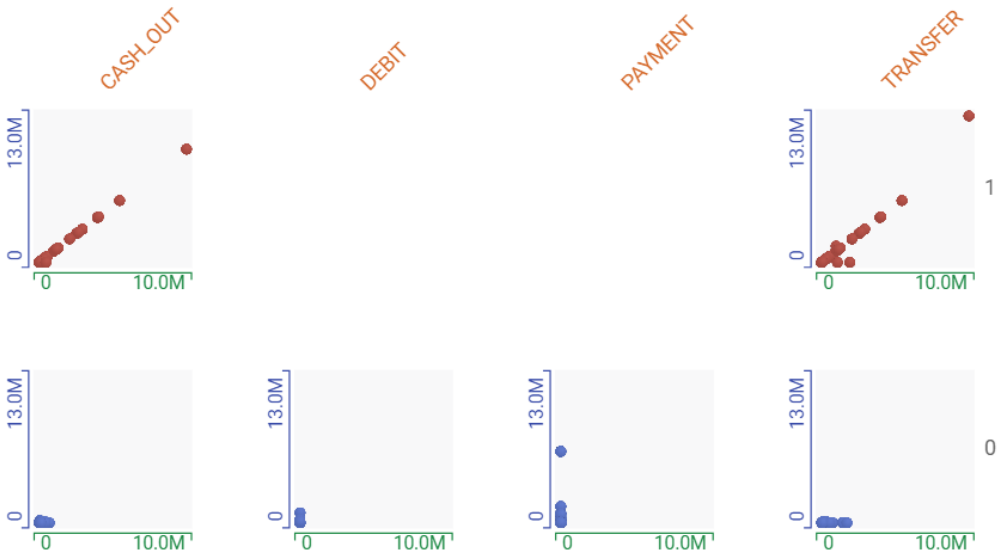

Fig. 5 Amount (X) against oldbalanceOrg (Y) faceted using transaction type (Red – Fraud, Blue – Not Fraud)

There is also an evident sign of a positive correlation between amount and oldbalanceOrg, indicating the possibility that the transaction made is fraudulent

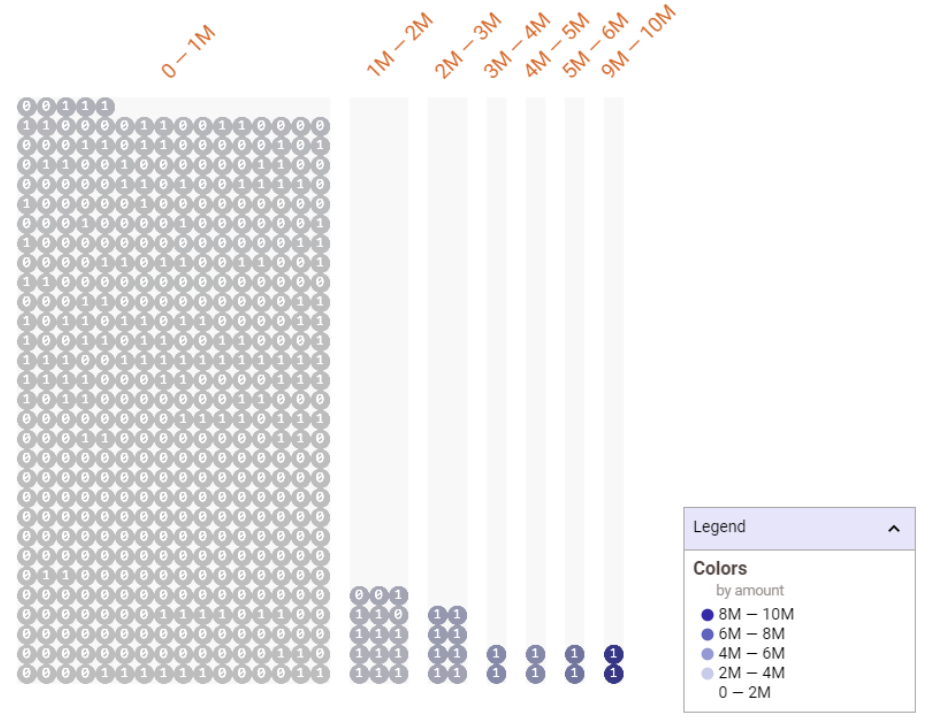

Fig 6. Analysing the transaction amount with fraudulent transactions

One interesting fact that we can observe from the dataset is that nearly all transactional amount above 1 million are fraudulent. This makes it ever more so important to impose stricter checks for large sum transactions.

Feature Information

FEATURE	DESCRIPTION	DATA TYPE
Type	Type of payment (CASH-OUT, DEBIT, PAYMENT, TRANSFER)	String
Amount	Amount of the transaction in local currency	Float
nameOrig	Customer who started the transaction	String
oldbalanceOrg	Initial balance before the transaction	Float
newbalanceOrig	New balance after transaction	Float
nameDest	Customer who is the recipient of the transaction	String
oldbalanceDest	Initial balance of the recipient before transaction	Float
newbalanceDest	New balance of the recipient before transaction	Float
isFraud	Determines if a transaction is fraudulent or not (0 for not a fraud, 1 for Fraud)	Integer
isFlaggedFraud	All zeros	Integer

Data Pre-processing

Removal of irrelevant features

After inspecting the data, the features nameOrig and isFlaggedFraud are dropped as they do not contribute to the prediction. nameOrig is entirely unique while isFlaggedFraud appears to be a redundant column as it is entirely filled with zeros.

Same_value feature

As observed in figure 5, when amount is equals to oldbalanceOrig, the transaction is most definitely a fraud. As such, a same_value feature is created which labels whether the transaction has amount equals to oldbalanceOrig.

Using CSV files

NAME	INDEX IN CSV FILE	DESCRIPTION
Dataset	a1	mini_fraud dataset is used
	a2	fraud dataset is used
same_value	b1	same_value is used
	b2	same_value is not used
One-hot encoding (type)	c1	One-hot encoding is used to transform categorical feature (type) into a numerical feature
Label encoding (type)	c2	Label encoding is used to transform categorical feature (type) into a numerical feature 0 for CASH_OUT and TRANSFER 1 for DEBIT and PAYMENT

nameDest prefix	d1	The first character of nameDest (C/M) is extracted and transformed into a numerical feature using one-hot encoding

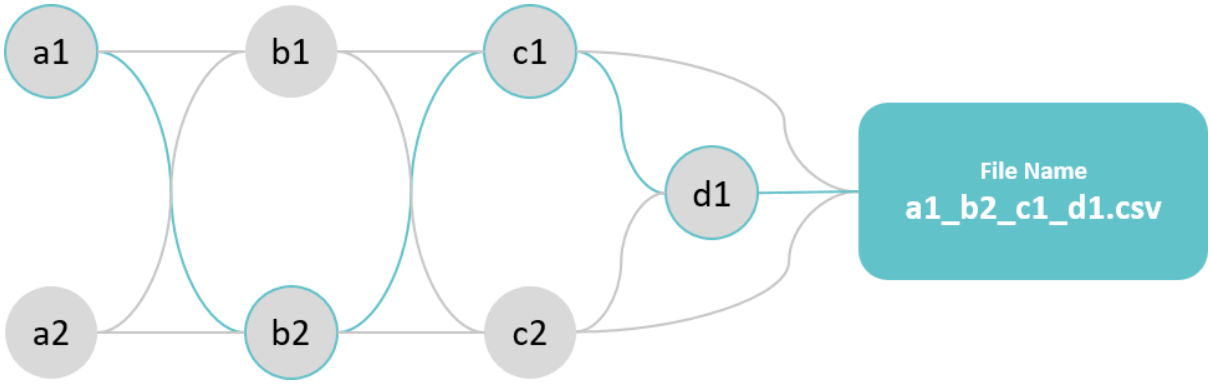

Fig 7. Example of file name saved for modelling

As there exist a different combination of pre-processing methods that will lead to better scores, various combinations of pre-processing methods are processed and saved as a csv file for modelling. The best combination for a model is then derived from the ROC AUC score and recorded in Model_results.xlsx. For example, a1_b2_c1_d1.csv will indicate that the mini_fraud dataset is used, has the same_value feature, has one-hot encoding performed for categorical features type and nameDest.

	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	same_value	C	M	CASH_IN	CASH_OUT
step	1	-0.03	-0.01	-0.01	-0	-0.02	0.05	0.05	-0.02	0.02	-0.01	-0.01
amount	-0.03	1	0	-0	0.22	0.31	0.13	0.12	0.4	-0.4	0.02	0.07
oldbalanceOrg	-0.01	0	1	1	0.09	0.06	0	0	0.19	-0.19	0.51	-0.2
newbalanceOrig	-0.01	-0	1	1	0.1	0.06	-0.01	-0.01	0.19	-0.19	0.53	-0.21
oldbalanceDest	-0	0.22	0.09	0.1	1	0.98	-0.01	-0.01	0.3	-0.3	0.11	0.13
newbalanceDest	-0.02	0.31	0.06	0.06	0.98	1	-0	-0	0.33	-0.33	0.06	0.16
isFraud	0.05	0.13	0	-0.01	-0.01	-0	1	0.99	0.02	-0.02	-0.02	0.01
same_value	0.05	0.12	0	-0.01	-0.01	-0	0.99	1	0.02	-0.02	-0.02	0.01
C	-0.02	0.4	0.19	0.19	0.3	0.33	0.02	0.02	1	-1	0.38	0.53
M	0.02	-0.4	-0.19	-0.19	-0.3	-0.33	-0.02	-0.02	-1	1	-0.38	-0.53
CASH_IN	-0.01	0.02	0.51	0.53	0.11	0.06	-0.02	-0.02	0.38	-0.38	1	-0.39
CASH_OUT	-0.01	0.07	-0.2	-0.21	0.13	0.16	0.01	0.01	0.53	-0.53	-0.39	1
DEBIT	-0.01	-0.05	-0.02	-0.02	0.01	0.01	-0	-0	0.06	-0.06	-0.04	-0.06
PAYMENT	0.02	-0.4	-0.19	-0.19	-0.3	-0.33	-0.02	-0.02	-1	1	-0.38	-0.53
TRANSFER	0	0.54	-0.08	-0.09	0.13	0.2	0.05	0.05	0.21	-0.21	-0.16	-0.22

Fig 8. Correlation Matrix using Fraud Dataset

From the pre-processing results, it was discovered that using method d1 leads to a decrease in score. In addition, checking the correlation between C, M features on the fraud dataset (million data points) shows a weak correlation of 0.02 and -0.02. Hence, it was not selected for the baseline model.

Pre-processing results: Model_results.xlsx

Pre-processed Data files: Prediction -> a1_b1_c1.csv, a1_b1_c1_d1.csv, a1_b1_c2.csv, a1_b1_c2_d1.csv, a1_b2_c1.csv, a1_b2_c2.csv, a1_b2_c2_d1.csv

Notebooks: Preprocessing-> Preprocess mini_fraud data.ipynb, Preprocess fraud data.ipynb

Modelling

The modelling stage is the primary place where data mining techniques are applied to the data. Below are the models used.

1. Logistic Regression
2. Decision Tree
3. Random Forest
4. Support Vector Machines
5. Gradient Boosting
6. Neural Network

For all 6 models, it is first trained on 80% of the dataset while leaving 20% of the dataset to be used as the test set.

Selecting Pre-processing methods

Fig 9. Selecting the best pre-processing methods

The pre-processed data is first trained on a model with default hyperparameters before evaluating the results using the ROC AUC score and the accuracy score. The AUC metric (Area Under the ROC Curve) is used as it is a simple metric that is able to summarize the performance of the model. It varies from 0 to 1. A value of 0.5 corresponds to randomness, meaning that the classifier cannot distinguish between positives and negatives while a value of 1 indicates that the classifier is able to correctly classify all positives and negatives.

Notebook for models: Prediction -> Logistic_Regression.ipynb, Decision_Tree.ipynb, Random_Forest.ipynb, SVM_with_d1.ipynb, Gradient_Boosting.ipynb, NN.ipynb

Hyperparameter Optimization

Once the best pre-processing methods are selected for a model, it will form the baseline model for comparison against the tuned models.

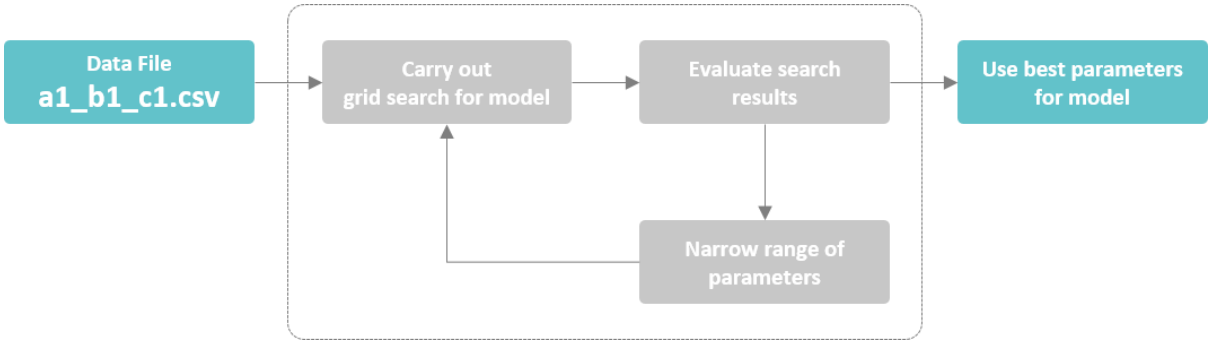

Fig 10. Diagram showing how hyperparameters are selected

Optimization Strategy

The optimization strategy involves performing both grid search and coarse-to-fine search to narrow in on the best parameters.

GridSearchCV is a function in scikit-learn that determines the best hyperparameters for a model by passing in a range of parameter values. Choosing which hyperparameters to optimize is difficult as some are more sensitive than others and are dependent on the choice of the model.

Hence, a wide range of values (with log intervals) is first set for the parameters. The range of values that do not get a good score are then eliminated until the best hyperparameters are found. This strategy helps to narrow in on only the high performing hyper-parameters. A 5/10-fold cross validation is also performed to avoid overfitting.

Model Results: Model_results.xlsx

Individual hyperparameter search: Optimization -> model_cv_result_n.xlsx

Evaluation of Results

Model	TP	TN	FP	FN	FPR	FNR	Specificity / TNR	Sensitivity / TPR	Test Accuracy	Test AUC Score	Rank
Gradient Boosting	283	112	5	0	0.0427	0	0.9573	1	0.9875	0.9786	1
Decision Tree	283	108	9	0	0.0769	0	0.9231	1	0.9775	0.9615	2
Random Forest	283	108	9	0	0.0769	0	0.9231	1	0.9775	0.9615	2
SVM	283	108	9	0	0.0769	0	0.9231	1	0.9775	0.9615	2
Logistic Regression	279	107	10	4	0.0855	0.0141	0.9145	0.9859	0.9650	0.9502	3
Neural Network	271	97	20	12	0.1709	0.0424	0.8291	0.9576	0.9200	0.9028	4

Fig 10. Table showing performance of models

Ranking of models

The best performing model appears to be the Gradient Boosting model as it has the highest AUC score and accuracy as compared to all the other models. The next 3 models (Decision Tree, Random Forest, Support Vector Machines) all have similar results after tuning of hyperparameters. This is unexpected as random forests is an ensemble model that uses many decision trees to predict a final outcome. This is usually far more accurate than using a single decision tree model. Furthermore, the top 4 models all have a false negative rate of 0. This means that the models are able to capture fraudulent transactions seamlessly. On the other hand, the false positive rate reveals the rate of which the classifier classified the transaction as fraudulent when it is not.

The neural network model seems to fall short, performing the worst out of all the models compared. This is likely due to the parameters not being tuned well and the lack of data points in the dataset.

Improving model results

Apart from optimizing the hyperparameters and using different pre-processing methods, what we can do to improve the results is to:

1. Increase the quantity of data
2. Improve quality of data
3. Perform error analysis
4. Rebalance the dataset

Increase the quantity of data

By increasing the number of data points, the model is able to be trained on more examples and be able to generalize better. It also decreases the chances of overfitting a model.

Improve the quality of data

Improving the quality of data can mean adding more input features into the dataset that may be relevant to predicting fraudulent transactions. It can also mean treating the outlier values separately so that the model is able to generalize better.

Perform error analysis

By searching for patterns in the errors made by the model, we can use this knowledge to add more data points that are similar to the errors made, so that the model is able to learn and predict the correct outcome.

Rebalance the dataset

As the chances of fraud is rare among the general population, the class distribution becomes unbalanced. This often leads to more errors made in testing as the model learns more from the majority class but is unable to distinguish what makes the other class "different". What we can do is perform oversampling (SMOTE - Synthetic Minority Over-sampling Technique) or ROSE – Random Oversampling) on the dataset, increasing the number of instances for the minority class.

Additional Analysis

	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	same_value	CASH_OUT	DEBIT	PAYMENT	TRANSFER
amount	1.000000	0.770000	0.050000	0.060000	0.230000	0.290000	0.230000	0.070000	-0.060000	-0.220000	0.200000
oldbalanceOrg	0.770000	1.000000	0.660000	-0.030000	0.070000	0.190000	0.150000	-0.030000	-0.040000	-0.010000	0.080000
newbalanceOrig	0.050000	0.660000	1.000000	-0.040000	-0.040000	-0.080000	-0.100000	-0.120000	-0.010000	0.170000	-0.060000
oldbalanceDest	0.060000	-0.030000	-0.040000	1.000000	0.740000	-0.020000	-0.020000	0.230000	-0.000000	-0.200000	-0.030000
newbalanceDest	0.230000	0.070000	-0.040000	0.740000	1.000000	-0.040000	-0.050000	0.180000	-0.040000	-0.220000	0.070000
isFraud	0.290000	0.190000	-0.080000	-0.020000	-0.040000	1.000000	0.950000	0.190000	-0.140000	-0.500000	0.440000
same_value	0.230000	0.150000	-0.100000	-0.020000	-0.050000	0.950000	1.000000	0.160000	-0.140000	-0.470000	0.430000
CASH_OUT	0.070000	-0.030000	-0.120000	0.230000	0.180000	0.190000	0.160000	1.000000	-0.170000	-0.590000	-0.400000
DEBIT	-0.060000	-0.040000	-0.010000	-0.000000	-0.040000	-0.140000	-0.140000	-0.170000	1.000000	-0.170000	-0.110000
PAYMENT	-0.220000	-0.010000	0.170000	-0.200000	-0.220000	-0.500000	-0.470000	-0.590000	-0.170000	1.000000	-0.390000
TRANSFER	0.200000	0.080000	-0.060000	-0.030000	0.070000	0.440000	0.430000	-0.400000	-0.110000	-0.390000	1.000000

By plotting a correlation matrix, we can observe that the same_value feature is highly correlated with isFraud. This indicates that it is an important feature that can be used to predict isFraud.

Notebook: Preprocessing -> Preprocessing_mini_fraud.ipynb

Below is a list of the most important features of the dataset

1. Same_value

2. PAYMENT
3. newbalanceOrig
4. CASH_OUT
5. Amount
6. oldbalanceOrg
7. newbalanceDest
8. TRANSFER
9. oldbalanceDest
10. DEBIT

Notebook: Prediction -> Random_forest.ipynb

Conclusion

In this project, what was done was to first explore the dataset. By understanding the nature of the dataset, what can be worked on, we can use the information to guide us in how we can pre-process the data and choosing an appropriate model. The next step was to perform data pre-processing and cleaning. A combination of pre-processing methods was selected based on the AUC score and accuracy of the model. One of the challenging problems was trying to optimize the model's hyperparameters. Due to the lack of data points, there were multiple choices of what could be the best hyperparameters for the model. To narrow it down would require a good understanding of how the parameters contribute to the model. Lastly, the evaluation of the models is carried out by collating the confusion matrices of the models and comparing their scores across different metrics.

Links

Facets Visualization

https://pair-code.github.io/facets/

Grid Search CV

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html