CZ4032 – Data Analytics & Mining
Credit Card Fraud Detection

Team Members: Tan Jian Wei (Leader), Kong Alson, Sam Jian Shen, Sun Jinghan, Li Zhaochen, Ang Yongxin

Agenda

1. Problem Statement
2. Data Pre-processing
3. Classification Models
4. Comparison between Different Models
5. Conclusion

Problem Statement

- Increased amount of fraudulent transactions
- Lack of accuracy of credit card fraud detection system

Credit Card Dataset

31 Features:

- V1, V2, V3, ……, V28 : principal components obtained with PCA
- Time, Amount
- Class

Tools Used

- Programing Language: Python
- Software: Google Colab
- Libraries Used: pandas, numpy, sklearn, matplotlib, seaborn, tensorflow, etc …

Classification Models

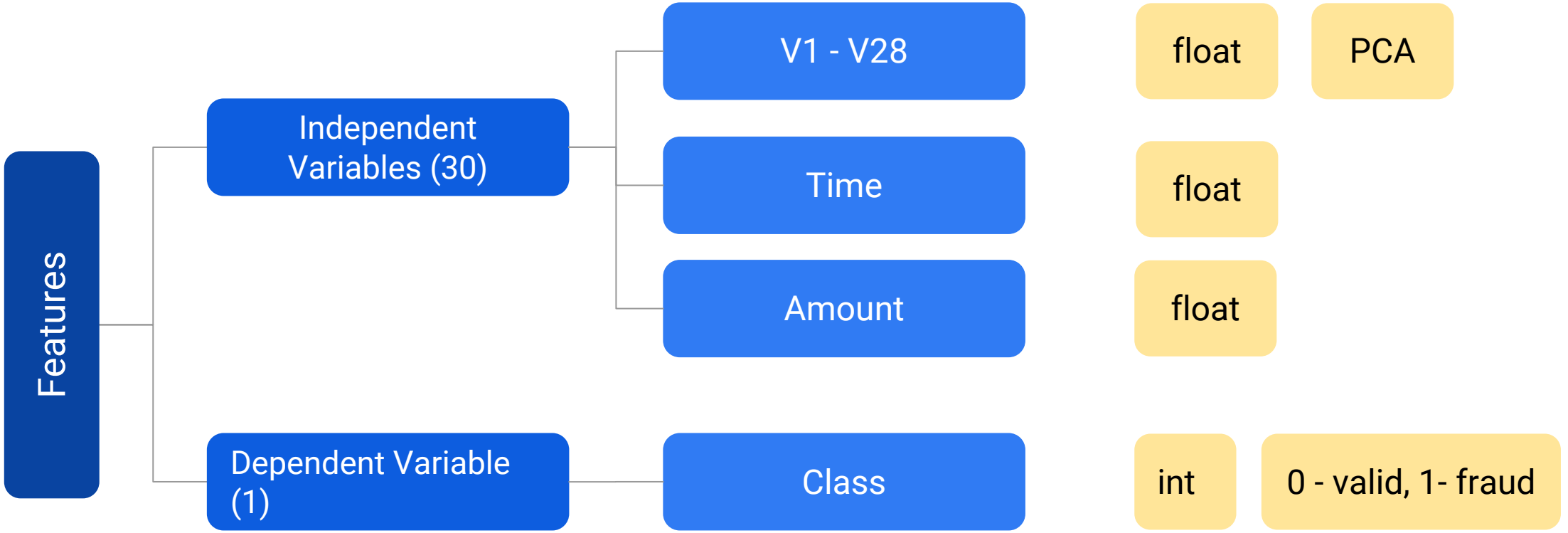
- Logistic Regression (LR)
- Support Vector Machine (SVM)
- Random Forest (RF)
- Artificial Neural Network (ANN)

Data Pre-processing

- Exploratory Analysis
- Removing Outliers
- Feature Selection
- Splitting into training and testing set
- Dealing with imbalanced class - Oversampling

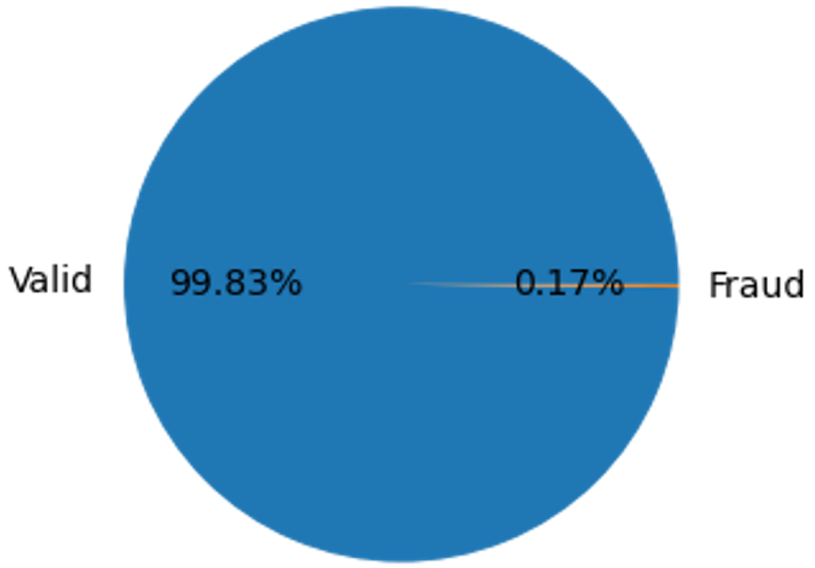
Exploratory Analysis

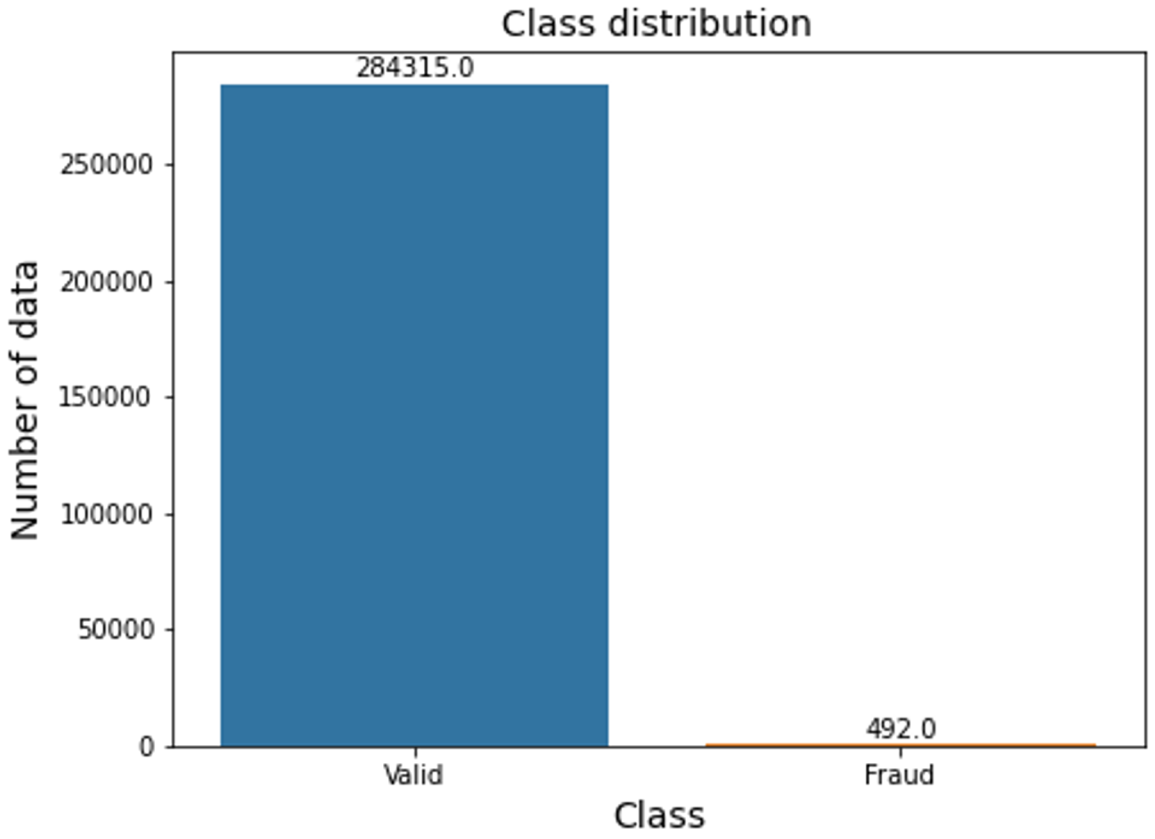
- Dataset statistics : 284807 rows, 31 columns
- Check for missing values : no missing values



Exploratory Analysis

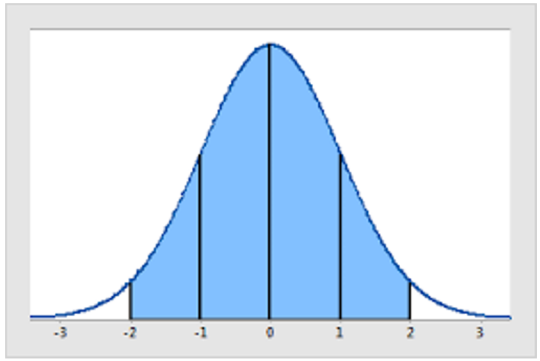
- Class distribution - Imbalanced

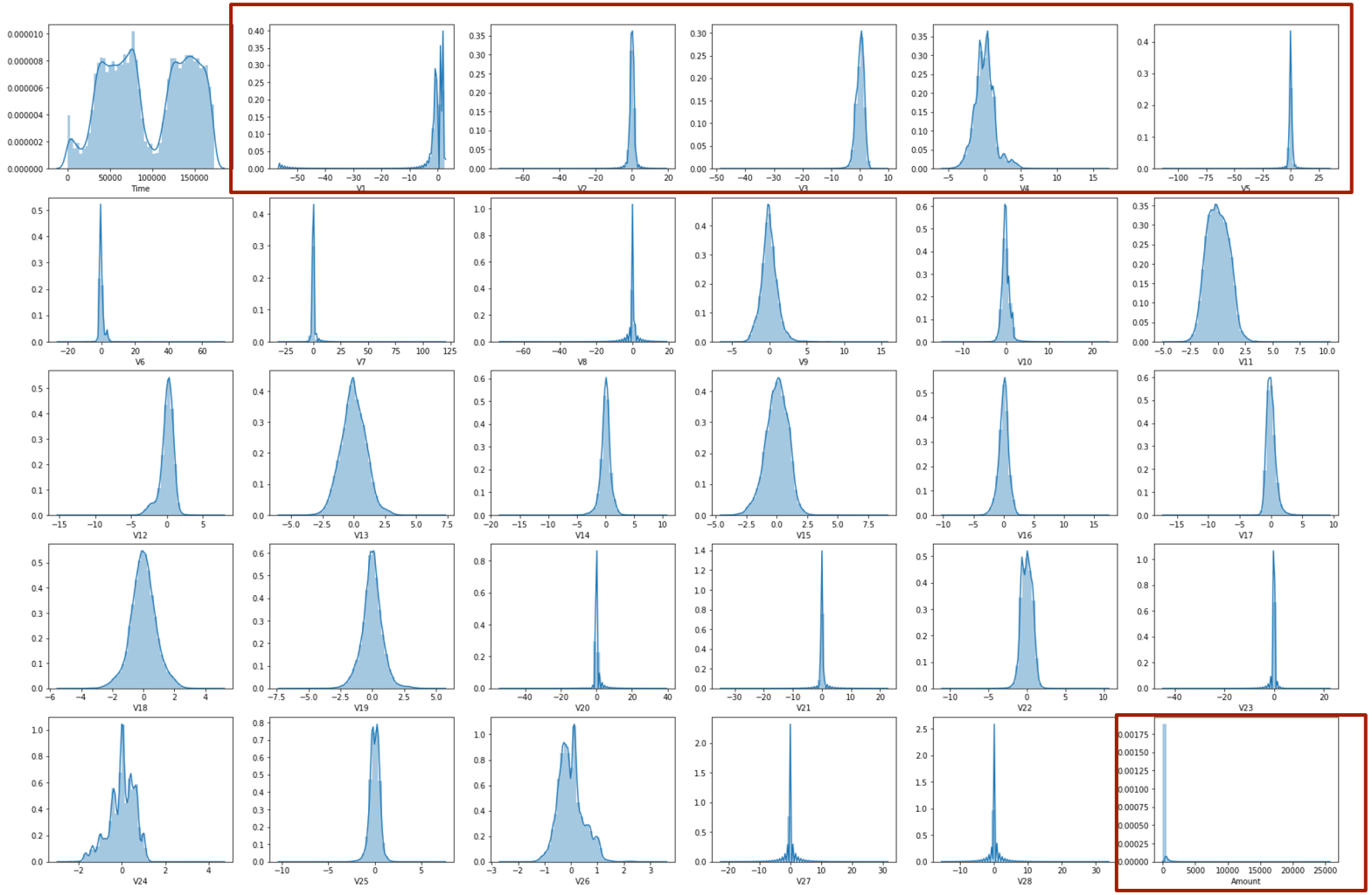




Remove Outliers

- features selected for outlier removal:
  V1, V2, V3, V4, V5, Amount
- remove data outside 3 standard deviation of normal distribution





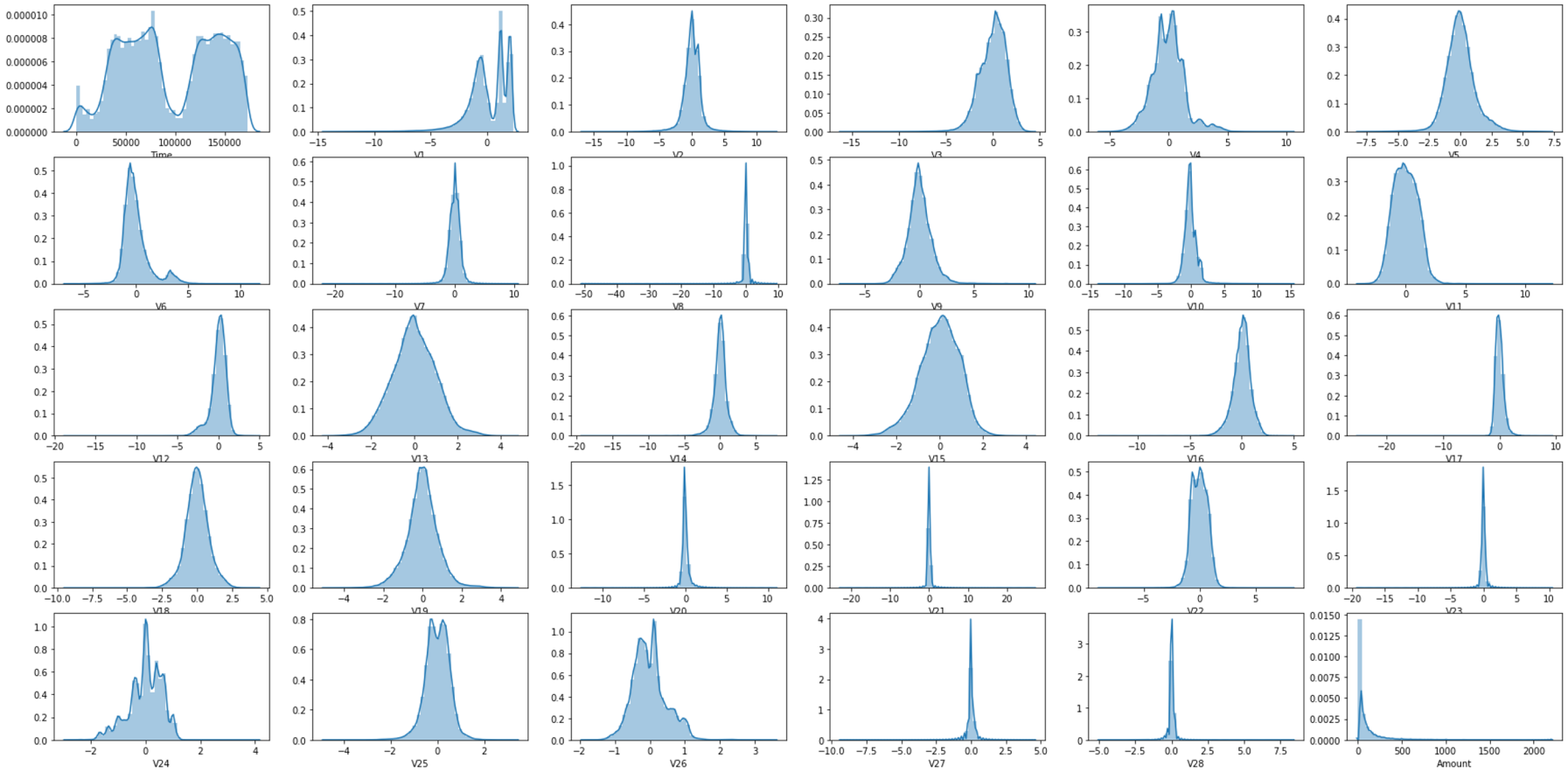
Distribution of each feature

Remove Outliers

- Total removal : 3529
  - valid : 3420
  - fraud: 109

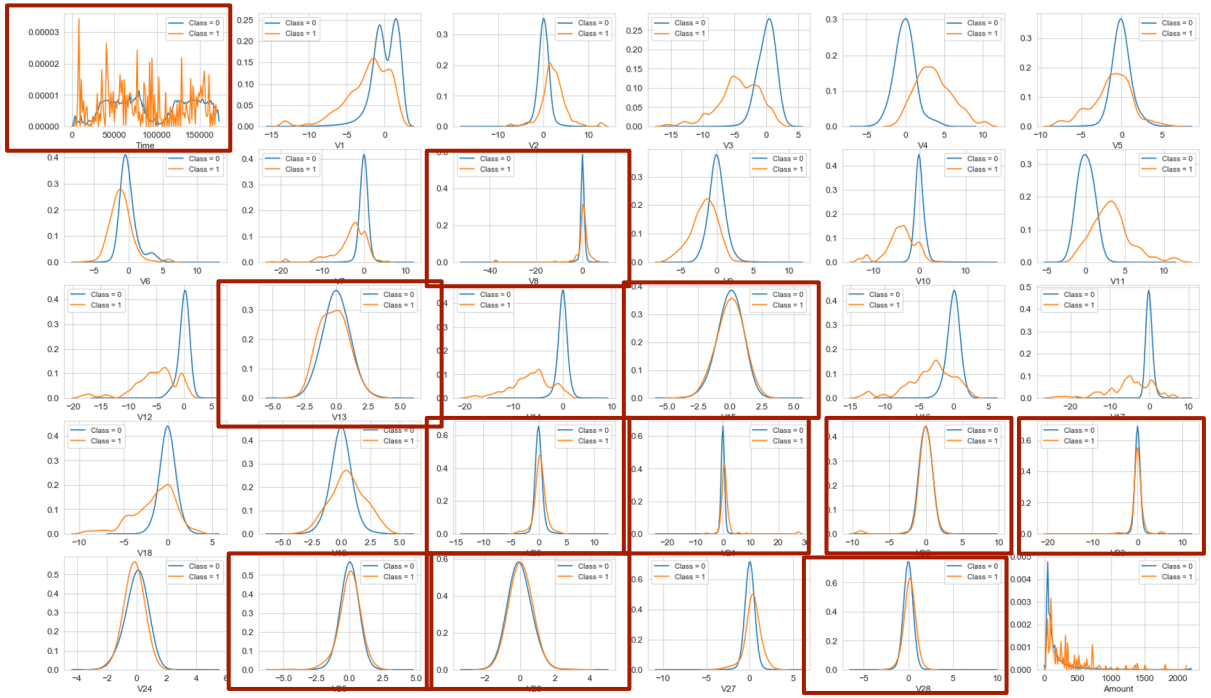
Remove Outliers

Distribution of each feature after removal:

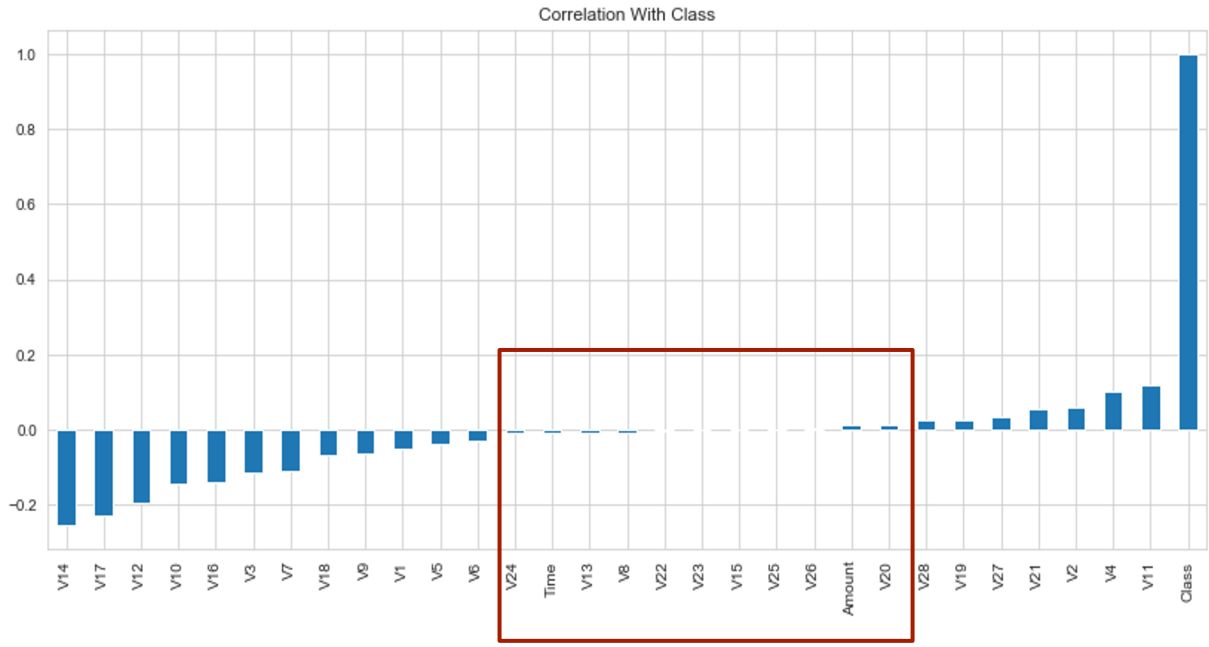


Feature Selection

- Distribution with Class



- Correlation with Class



Feature Selection

- Features to be removed:

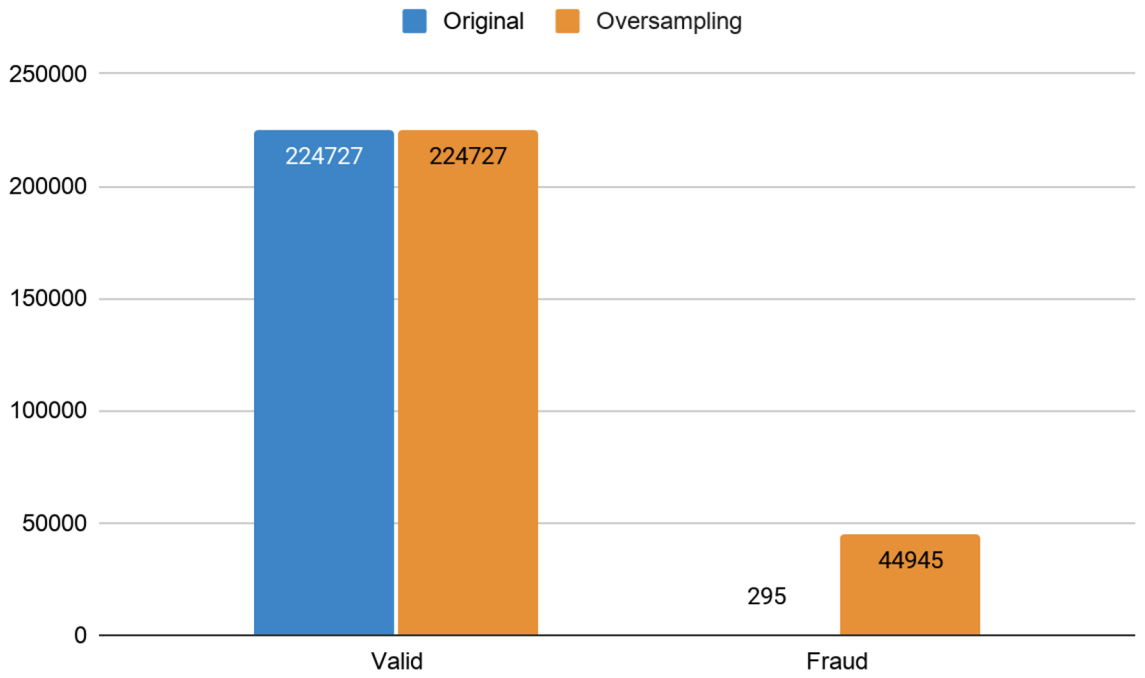
  Time, V8, V13, V15, V20, V21, V22, V23, V25, V26, V28

- Remaining features - **19** independent variables

Dealing with Imbalanced dataset - Oversampling

- Splitting into training and testing set
  - split ratio: 80% : 20%
- Oversampling on training dataset
  - Method: **SMOTE**
  - After oversampling

    Class 0 : Class 1 = 5 : 1



Dataset size after oversampling

Evaluation Metrics

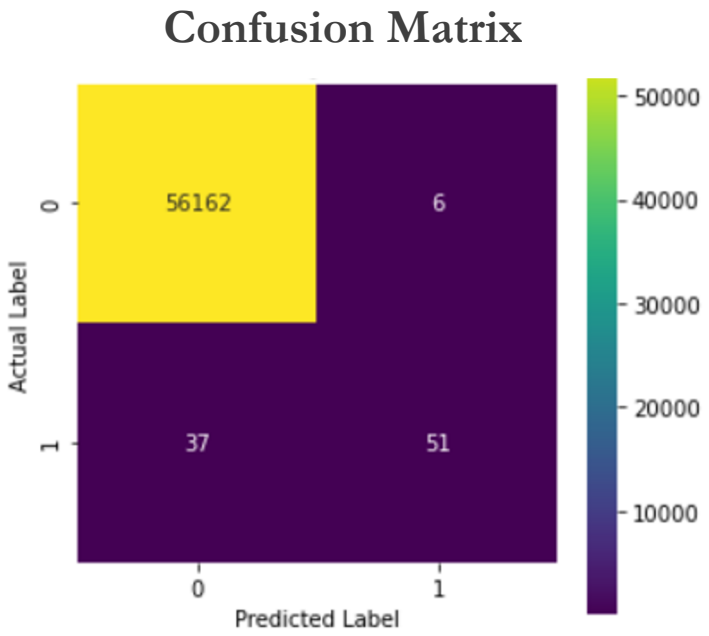
- Confusion matrix
  - Precision Score : $\frac{TP}{TP + FP}$
  - Recall Score : $\frac{TP}{TP + FN}$
  - F1 Score: $\frac{2 * precision * recall}{precision + recall}$

- Precision-Recall Curve → Focus more on **minority** class, i.e.**fraud** transactions
  - AUC score

Classification Models

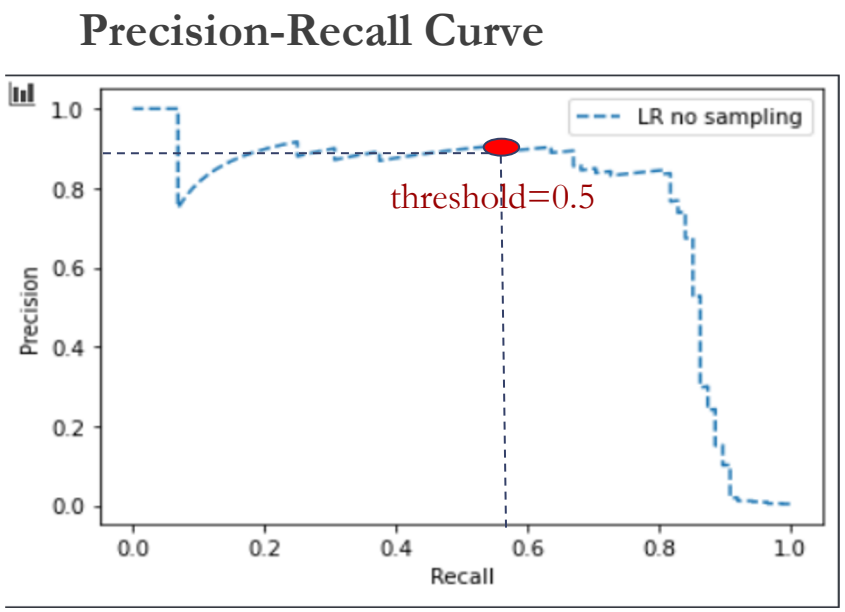
Logistic Regression

Result - before oversampling



Precision: 0.894
Recall: 0.579
f1-score: 0.703

(using default threshold for fraud : 0.5)

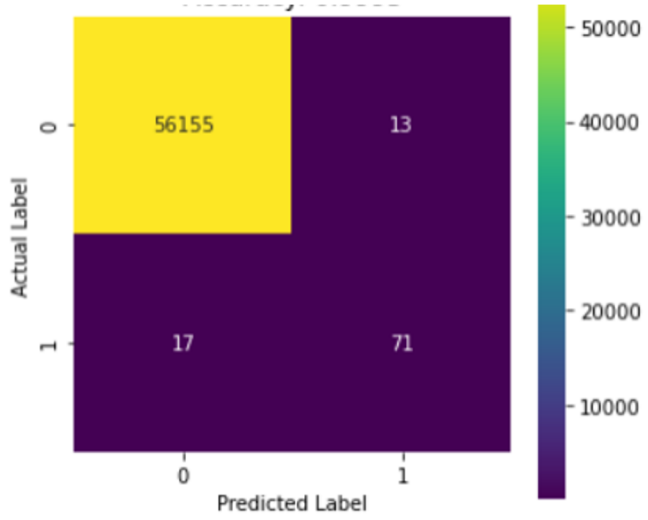


AUC-score: 0.738

Result - before oversampling

Threshold Moving

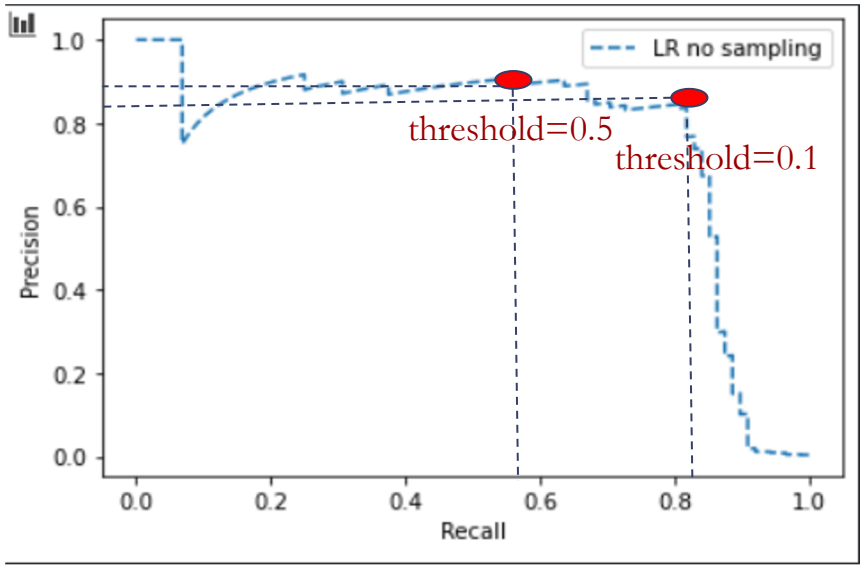
Confusion Matrix




```
Precision: 0.840
Recall: 0.802
f1-score:0.825
```

Change threshold for fraud to 0.1

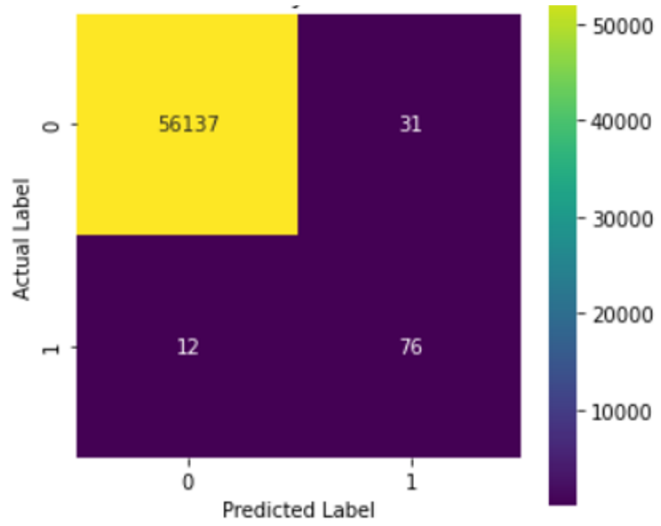
Precision-Recall Curve




```
AUC-score: 0.738
```

Result - after oversampling

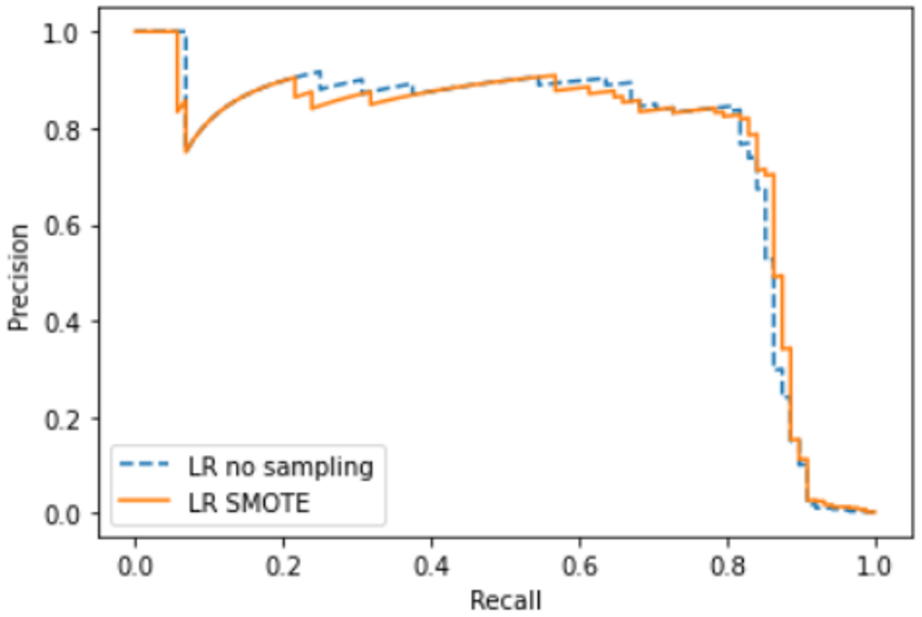
Confusion Matrix



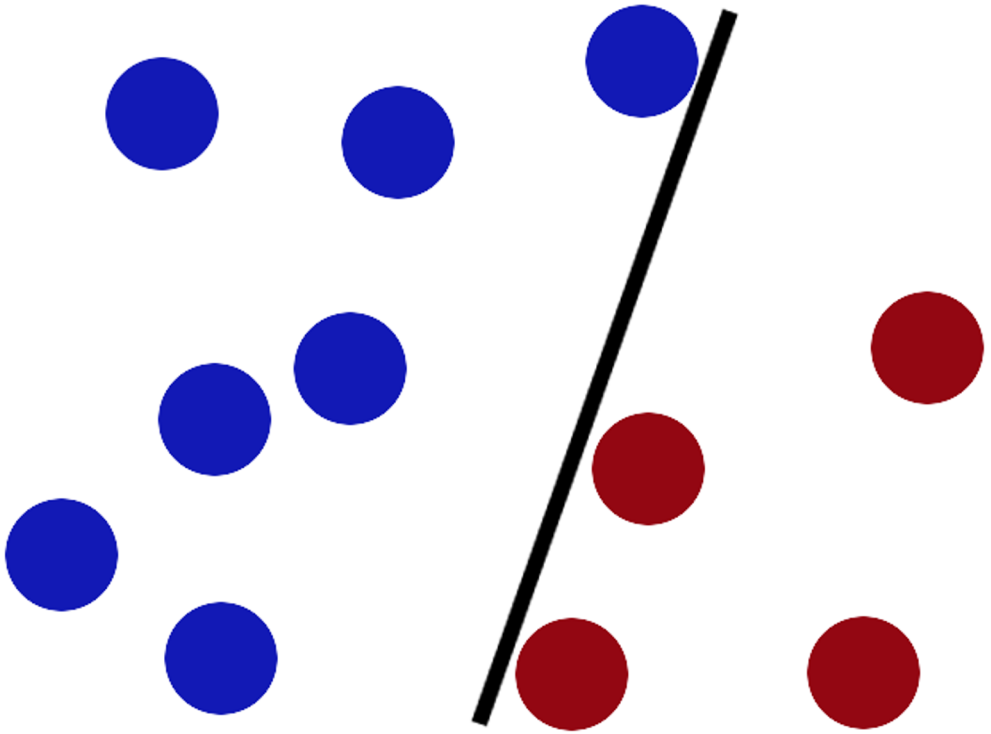

```
Precision: 0.700
Recall: 0.852
f1-score: 0.814
```

(using default threshold for fraud : 0.5)

Precision-Recall Curve




```
AUC-score: 0.763
```



Support Vector Machine(SVM)

Radial Basis Function(RBF) kennel

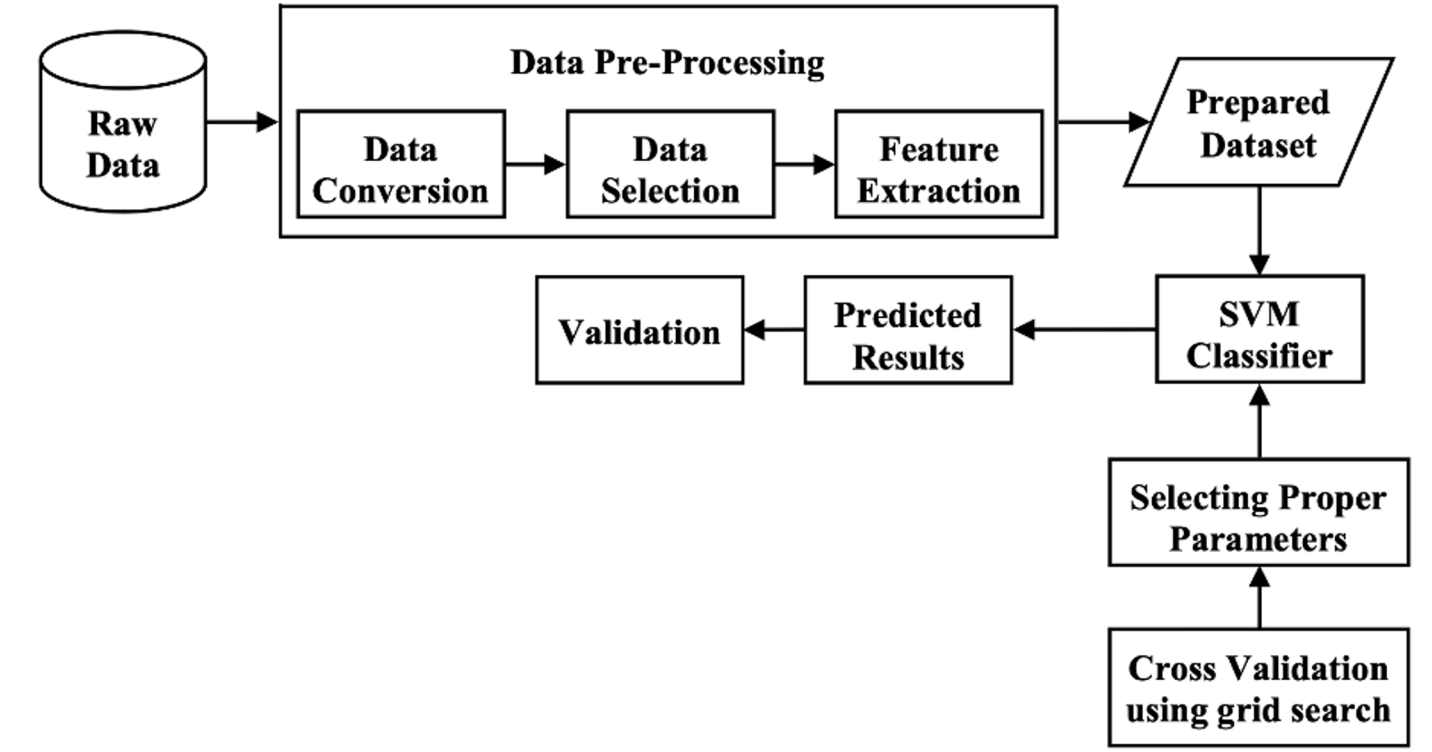
- Gamma(γ)
- C

Fine-tuning

- Grid search method
  - list of parameter (gamma and C)
  - kernel
  - obtain the best g and C.

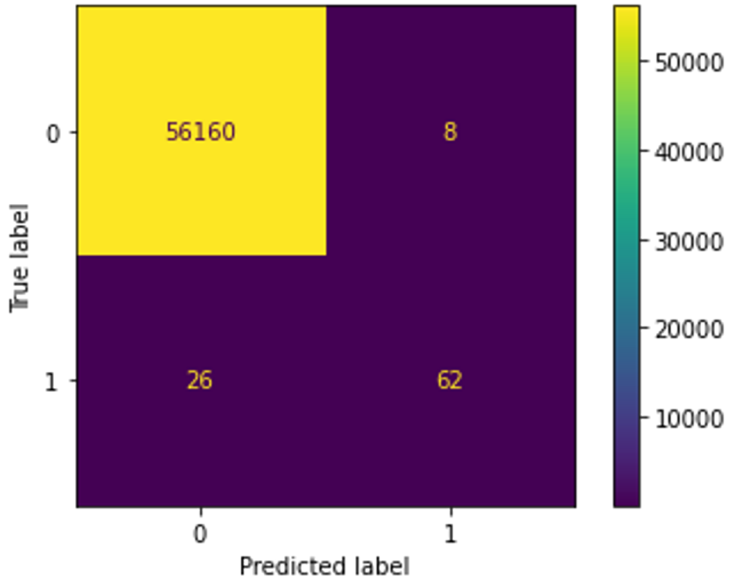
```
from sklearn.svm import SVC
classifier= SVC(kernel='rbf',C = 10,gamma=0.001,probability=True)
classifier.fit(train_x, np.array(train_y).ravel())
```

SVM Model


Raw Data
Data Pre-Processing
Data Conversion
Data Selection
Feature Extraction
Prepared Dataset
SVM Classifier
Predicted Results
Validation
Selecting Proper Parameters
Cross Validation using grid search

Result-without sampling

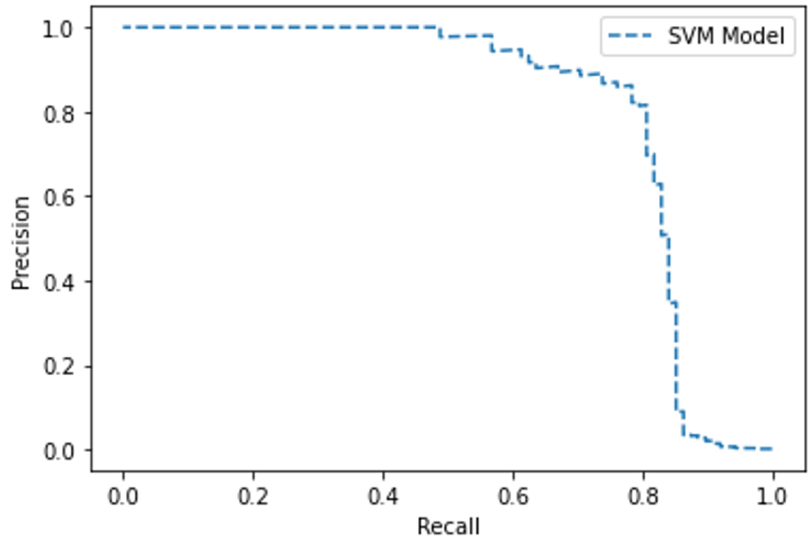
Confusion Matrix



Precision: 0.886
Recall: 0.705
F1 Score: 0.785

(using default threshold for fraud : 0.5)

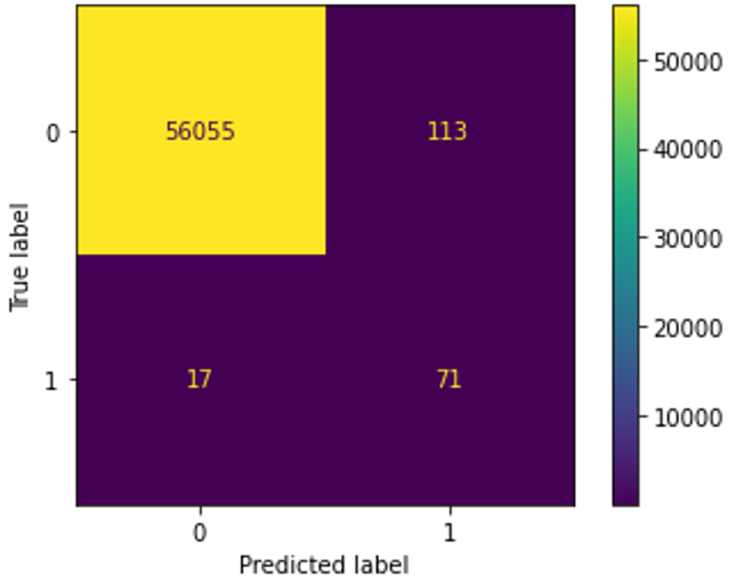
Precision-Recall Curve



AUC : 0.810

Result-with oversampling

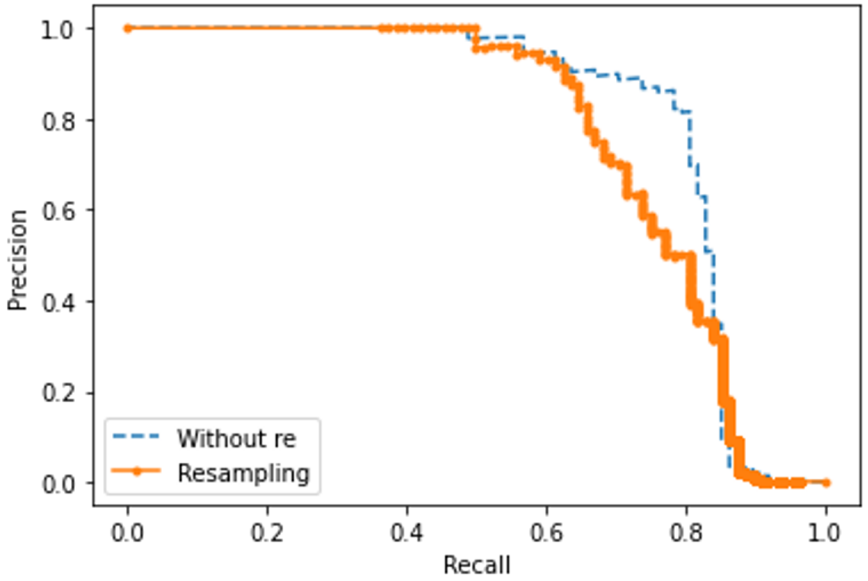
Confusion Matrix



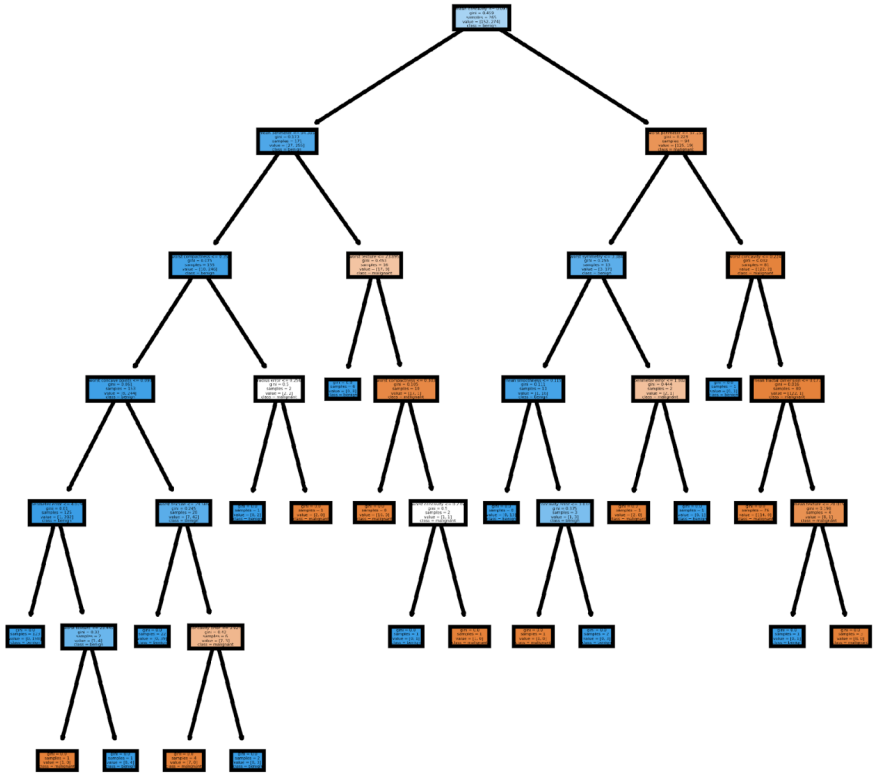

```
Precision: 0.399
Recall: 0.807
F1: 0.526
```

(using default threshold for fraud : 0.5)

Precision-Recall Curve




```
AUC : 0.762
```



Random Forest

Implementation

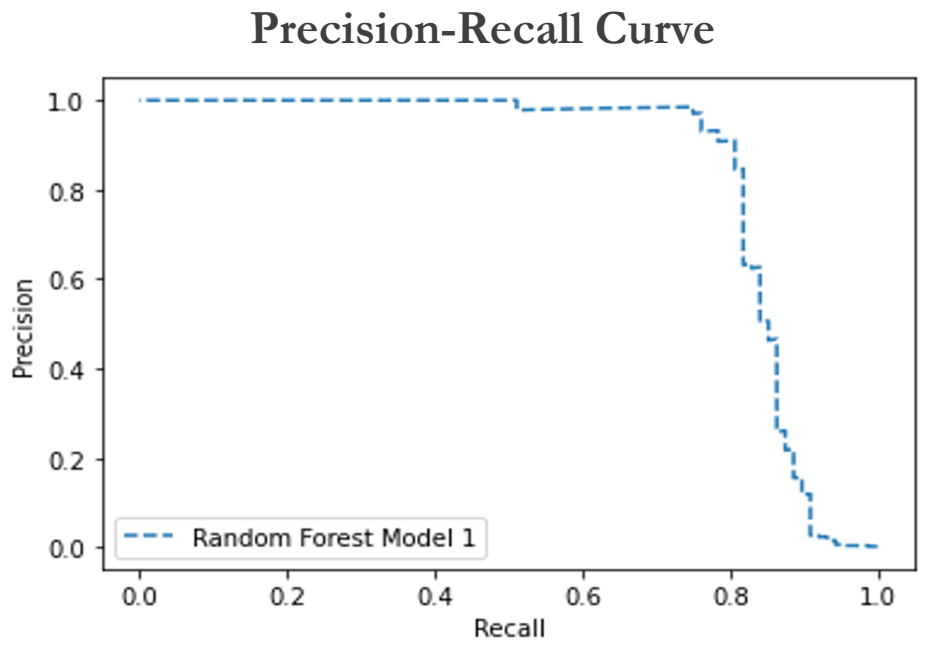
```
from sklearn.ensemble import RandomForestClassifier
```

```
RF1 = RandomForestClassifier(n_estimators=100,
                             max_depth=10,
                             min_samples_split=5,
                             random_state=0)
```

Result - before oversampling

Precision: 0.971
Recall: 0.761
F1: 0.854

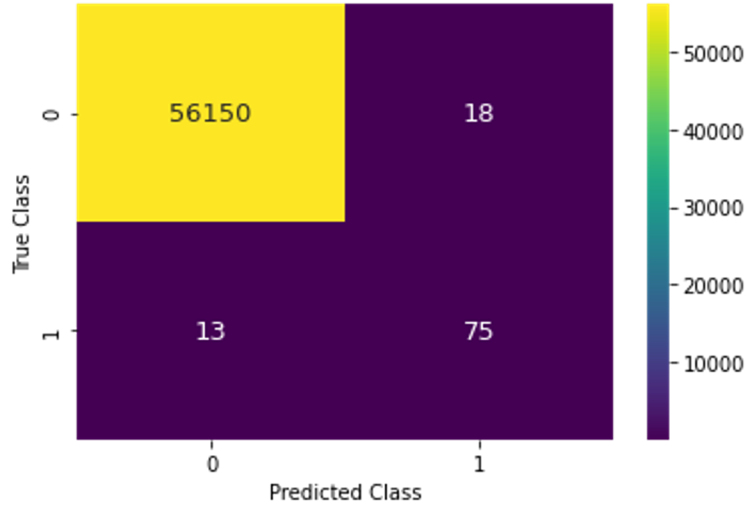
(using default threshold for fraud : 0.5)



AUC score: 0.843

Result - after oversampling

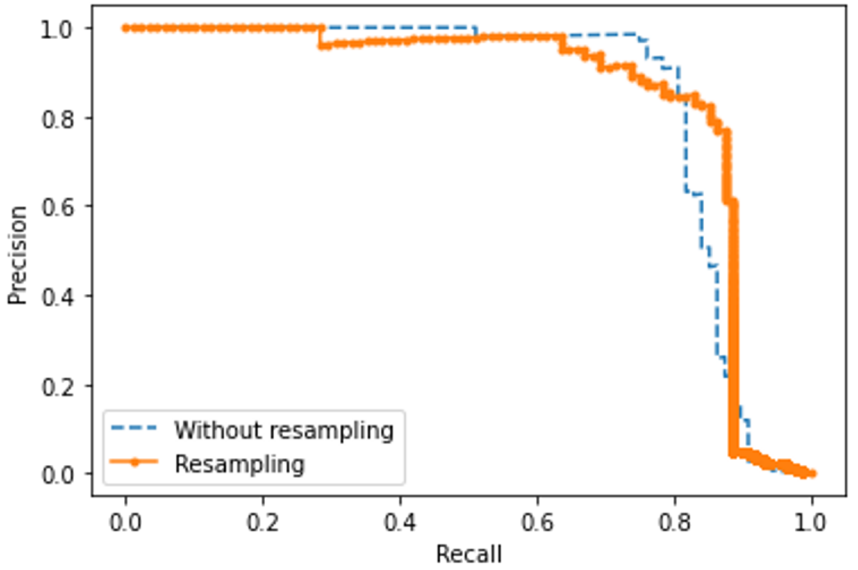
Confusion Matrix




```
Precision: 0.806
Recall: 0.852
F1: 0.829
```

(using default threshold for fraud : 0.5)

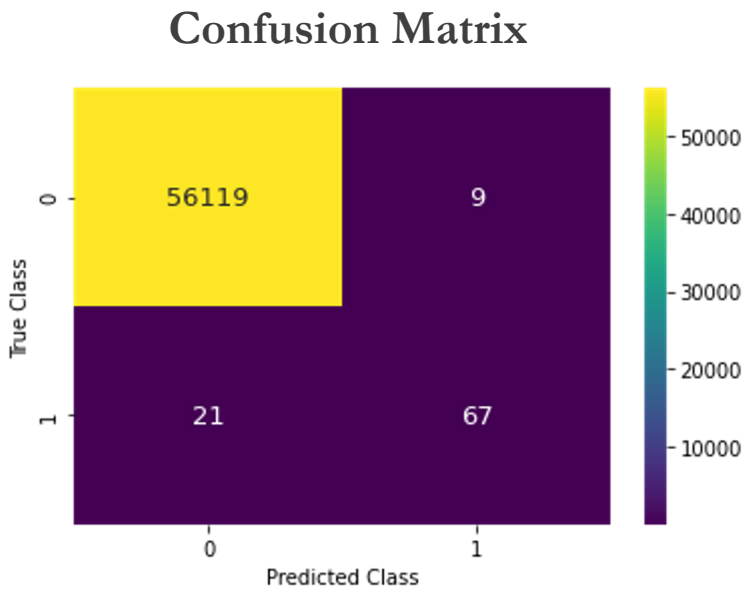
Precision-Recall Curve

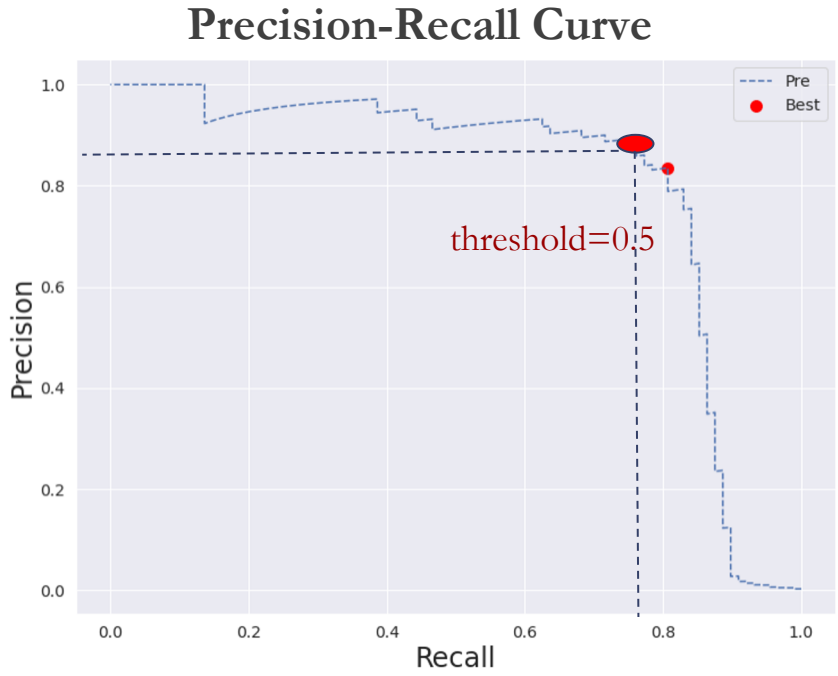



```
AUC score: 0.848
```

Artificial Neural Network

Result - before oversampling





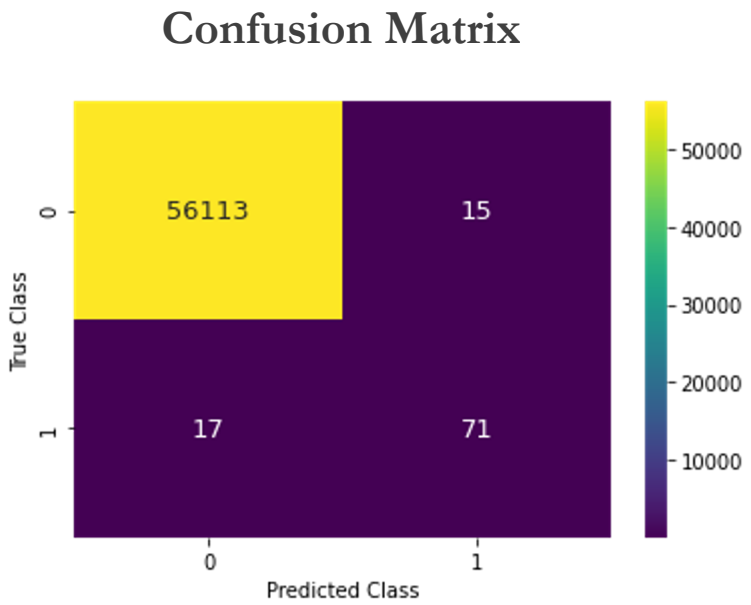

```
Precision:  0.882
Recall:     0.762
F1 Score:   0.817
```

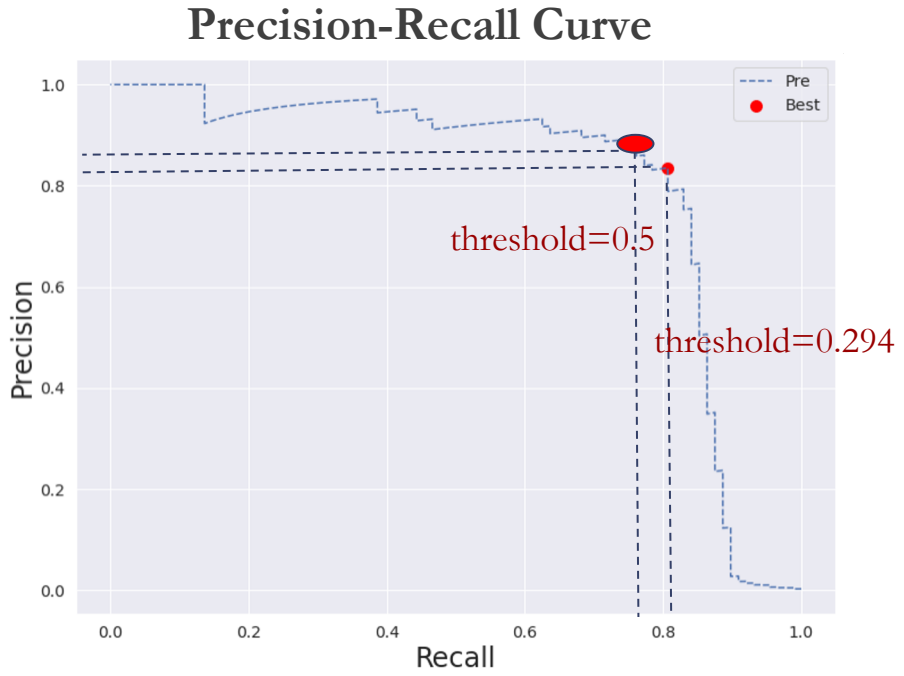
```
AUC:        0.806
```

(using default threshold for fraud : 0.5)

Result - before oversampling

Threshold Moving





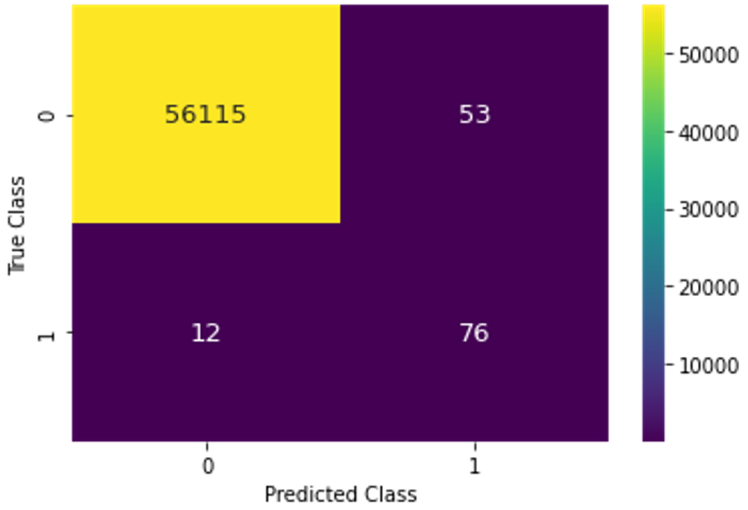

```
Precision:  0.826
Recall:     0.807
F1 Score:   0.821
```

```
AUC:        0.806
```

Change threshold for fraud to 0.294

Result - after oversampling

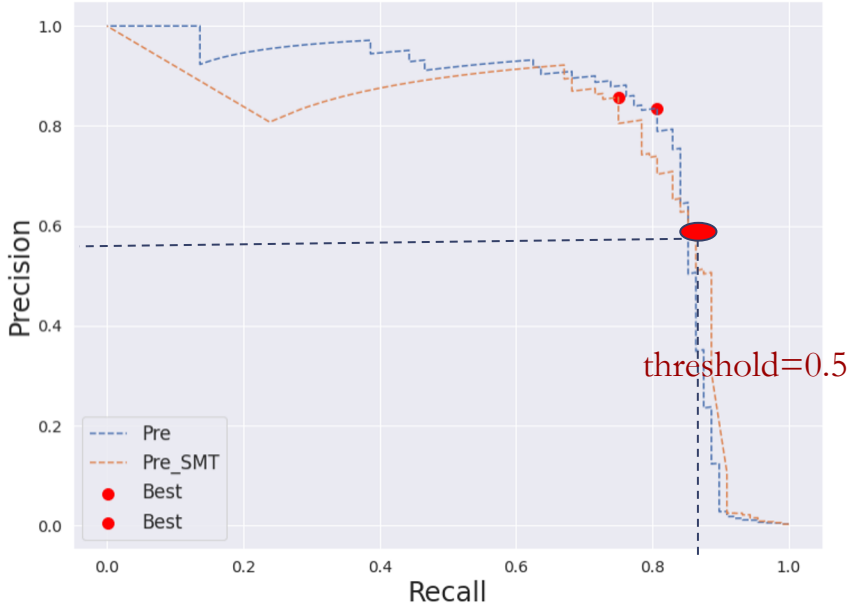
Confusion Matrix




```
Precision:  0.589
Recall:     0.864
F1 Score:   0.700
```

(using default threshold for fraud : 0.5)

Precision-Recall Curve

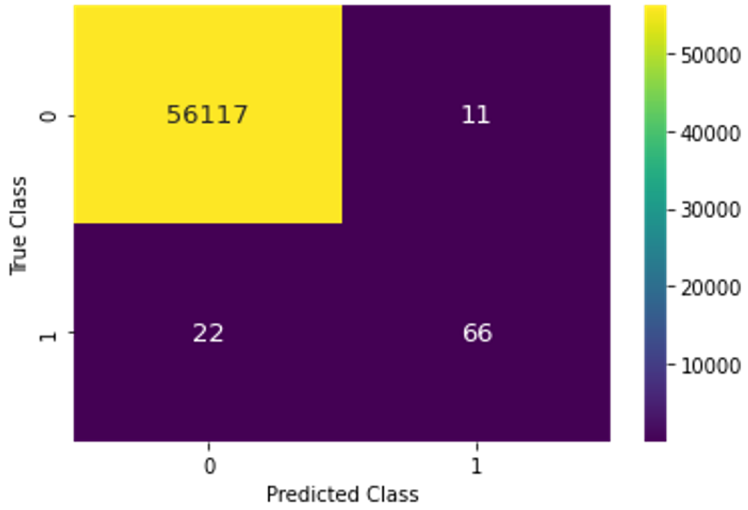



```
AUC:   0.765
```

Result - after oversampling

Threshold Moving

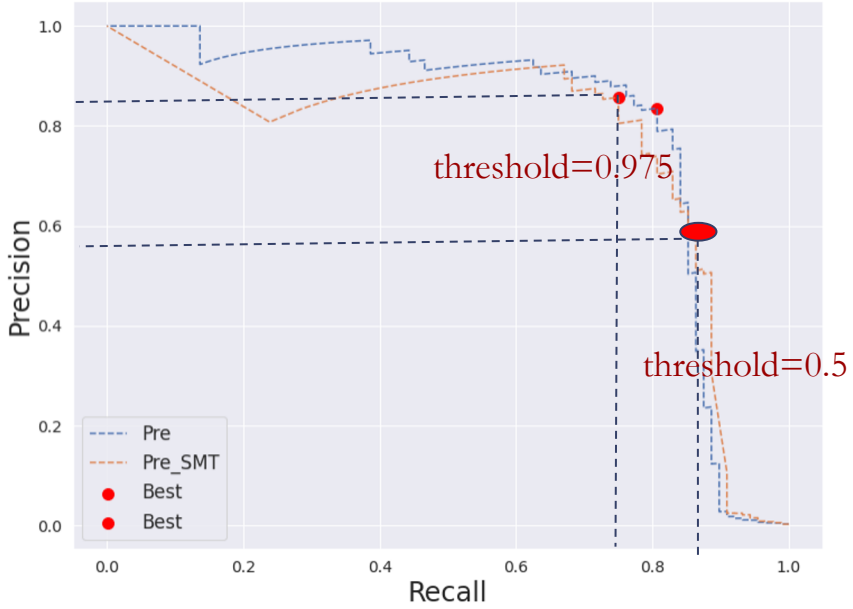
Confusion Matrix




```
Precision:  0.857
Recall:     0.750
F1 Score:   0.800
```

Change threshold for fraud to 0.975

Precision-Recall Curve




```
AUC:   0.765
```

Comparison between Different Models

Comparison between Different Models

Classification Model	**Precision**	**Recall**	**F1 Score**	**AUC Score** Precision Recall Curve
Logistic Regression	0.700	0.852	0.814	0.763
Support Vector Machine (SVM)	0.886	0.705	0.785	0.810
Random Forest	0.806	0.852	0.829	0.848
Artificial Neural Network (ANN)	0.807	0.826	0.821	0.806

Random Forest provides highest precision and recall

Conclusion

Conclusion

- Imbalanced dataset - Oversampling can help improve the model
- Random Forest provides the best prediction result
  - Final prediction on testing set
    - Precision : 80.6%
    - Recall : 85.2%

(Decision threshold for fraud transactions is 0.5)