CIE 417 project

Hazem Muhammad Tarek 201800283

Ibrahim Hamada Ibrahim 201800739

Sohaila Islam Zaki 201800998

Credit card transactions fraud detection

Problem definition

In this project, we are investigating one of the most common problems encountered in machine learning which is the binary classification problem. In this problem, the goal is to categorize the data points into two buckets: 0’s or 1’s, true or false, etc.

Examples:

- E-mail spam detection.
- Fraud detection.
- Cat-dog categorization.

Data set

The data set used is a simulated credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. It covers credit cards of 1000 customers doing transactions with a pool of 800 merchants. The data set contains 23 features which are:

- #: the transaction number.
- cc_num: credit card number.
- Merchant: merchant’s name.
- Category: transaction’s category.
- Amt: amount of money transferred.

- First name: client’s first name.
- Last name: client’s last name.
- Gender: client’s gender.
- Street: client’s street.
- City: client’s city.
- State: client’s state.
- Zip Code: client’s zip code.
- Long: customer’s longitude.
- Lat: customer’s latitude.
- City_pop: city’s population.
- Job: customer’s job.
- Dob: customer’s date of birth.
- Trans_num: transaction ID.
- Unix_time: transaction’s occurrence time.
- Merch_lat: merchant’s latitude.
- Merch_long: merchant’s longitude.
- Is_fraud: determines if the transaction is legitimate (0) or fraud (1).

Approach and methodology

I. Data pre-processing

- No nulls were found.
- No duplicates were found.

- Feature engineering: the datatype of dob and data_trans_time were changed to date time.
- Ordinal encoding was used to encode the features category, trans_day, gender, and job.
- The redundant columns trans_date_trans_time, cc_num, merchant, category, first, last, gender, street, city, state, job, dob, trans_num, trans_day, trans_month, and trans_year_month were dropped.
- The data was completely imbalanced towards legitimate transactions, so we used oversampling to overcome such a problem.

II. Model selection and evaluation

- Logistic regression.
- Random forest.
- Decision tree.
- AdaBoost
- Bagging

These models were selected as they have proven to be efficient when used in binary classification problems.

Implementation

Logistic regression

logistic regression was imported from sk-learn linear model.

hyperparameter tuning:

Randomized search CV was used to choose between the following hyperparameters:

solvers = ['newton-cg', 'lbfgs', 'liblinear']

penalty = ['l2'],

c_values = [100, 10, 1.0, 0.1, 0.01]

Best Parameters {'solver': 'newton-cg', 'penalty': 'l2', 'C': 0.1}

Best run: 88.36

Decision tree

Decision Tree Classifier was imported from sk-learn linear model.

Hyper-parameter Tuning:

Grid search CV was used to choose between the following hyperparameters:

solvers = [{'criterion': ['entropy', 'gini'}]

Max_depth = { [2,5,7,10,12]}

Best Parameters {'criterion': 'entropy', 'max_depth': 10}

F1 Score 86.01.

Random forest

Random Forest Classifier was imported from sk-learn linear model.

Hyper-parameter Tuning:

Grid search CV was used to choose between the following hyper parameters:

solvers = param_grid = {'bootstrap': [True], 'max_depth': [10,15], 'max_features': [2, 3], 'min_samples_leaf': [3, 4, 5,6], 'min_samples_split': [3,4,5,6], 'n_estimators': [1150, 1200, 1250, 1300,1350]}

AdaBoost

AdaBoost Classifier was imported from sk-learn linear model.

The model is trained on both unbalanced and over-sampled data.

Bagging

Bagging Classifier was imported from sk-learn linear model.

The model is trained on both unbalanced and over-sampled data.

Model performance

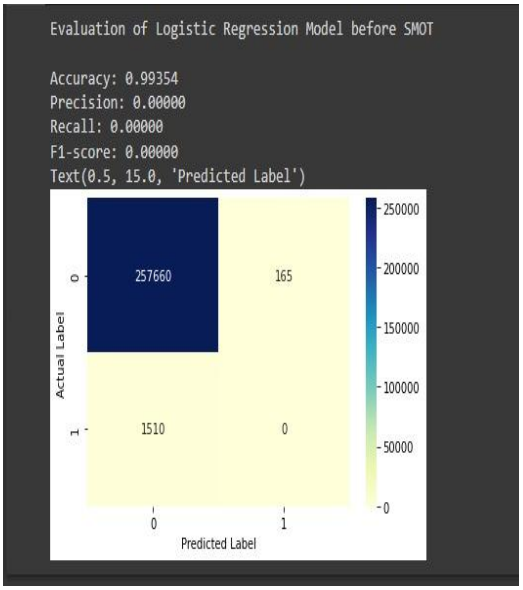
Two functions, **metrics**, and **results** were implemented.

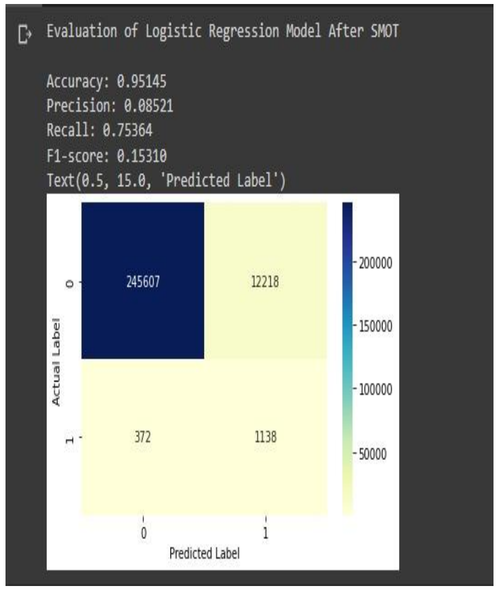
metrics: takes actual and prediction as an input and returns accuracy, recall, precision, and F1 as output.

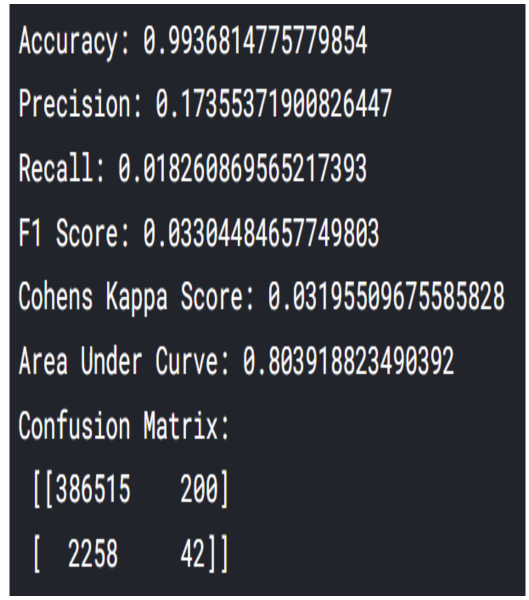
Result: takes model, X-test, and y-test and compares between the actual and the predicted.

Model comparisons

- **Logistic regression** results compared to kaggle model:



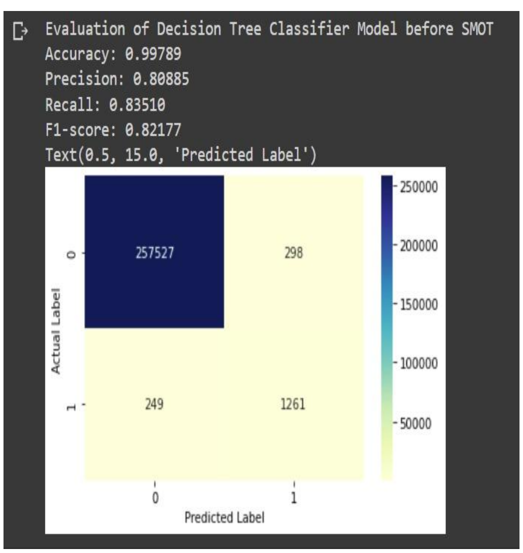


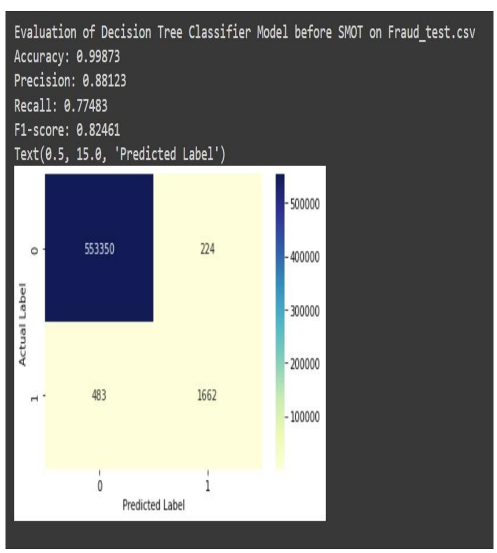


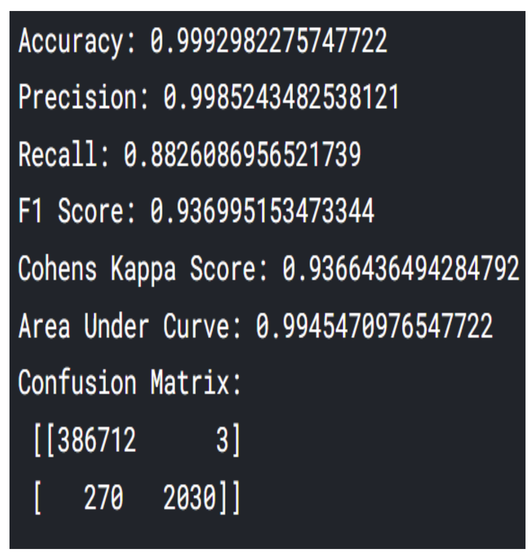
Our Model

Kaggle's Model

- **Decision trees** compared to kaggle model:



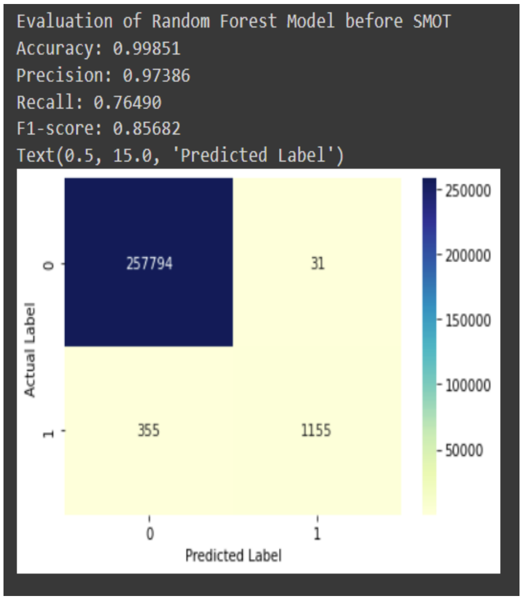


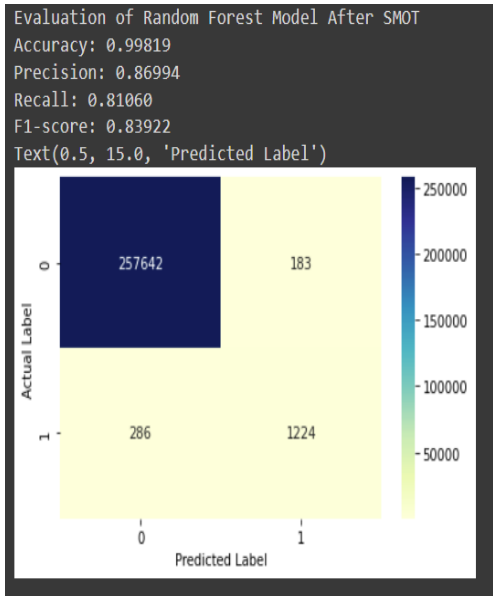


Our Model

Kaggle's Model

- Random forest compared to kaggle model:



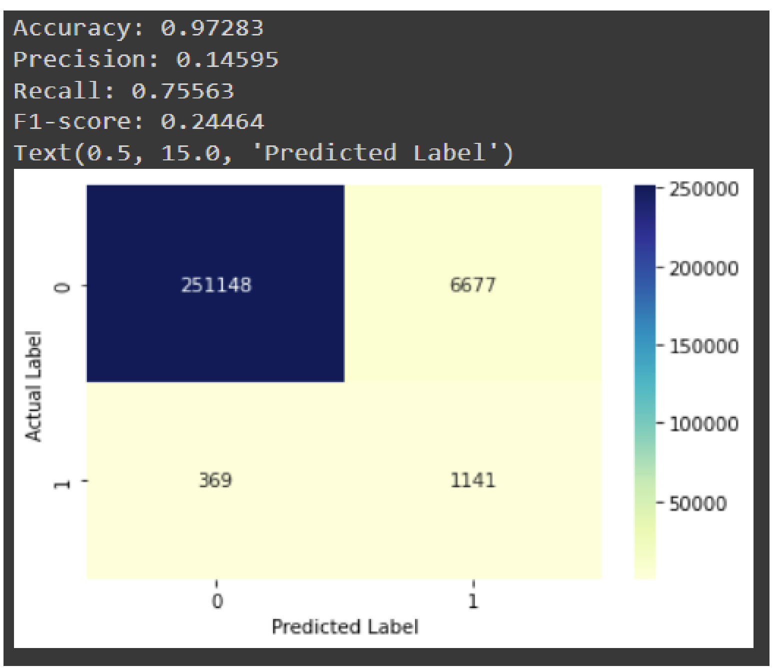



```
Accuracy: 0.9995270105265863
Precision: 0.9976481655691439
Recall: 0.9221739130434783
F1 Score: 0.9584274740171713
Cohens Kappa Score: 0.9581899959286818
Area Under Curve: 0.9971499109837658
Confusion Matrix:
 [[386710      5]
 [   179   2121]]
```

Our Model

Kaggle Model

- AdaBoost compared to kaggle model:



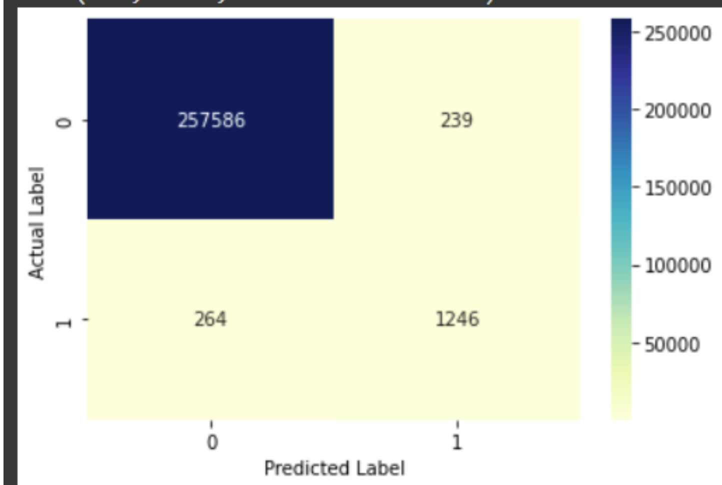

```
[[28189  1509]
 [   56   246]]

              precision    recall  f1-score   support

           0       1.00      0.95      0.97     29698
           1       0.14      0.81      0.24       302

    accuracy                           0.95     30000
   macro avg       0.57      0.88      0.61     30000
weighted avg       0.99      0.95      0.97     30000
```

Our Model

Kaggle Model

- Bagging compared to kaggle model:

```
Evaluation of bagging Model before SMOT
Accuracy: 0.99806
Precision: 0.83906
Recall: 0.82517
F1-score: 0.83205
Text(0.5, 15.0, 'Predicted Label')
```




```
[[29304   394]
 [   44   258]]

              precision    recall  f1-score   support

           0       1.00      0.99      0.99     29698
           1       0.40      0.85      0.54       302

    accuracy                           0.99     30000
   macro avg       0.70      0.92      0.77     30000
weighted avg       0.99      0.99      0.99     30000
```

Conclusion

I. Model evaluation

- Logistic regression:

The outcome did not change much after balancing the data using oversampling which means that logistic regression is not the best choice for this data set.

- Decision tree and random forest:

After balancing the dataset using the oversampling technique, the values of recall and F1 were improved which implies that the model is more likely to detect a fraud transaction.

- AdaBoost and Bagging Models:

After balancing the dataset using the oversampling technique, the values of recall and F1 were improved which implies that the model is more likely to detect a fraud transaction.

Bagging model achieves a higher performance than AdaBoost Model.

II. Learning outcomes

- Searching for datasets using the problem definition.
- Manipulating datasets and identifying the redundant information within a dataset.
- Choosing a suitable model based on the given problem.
- Ordinal encoding is a really efficient way of encoding.
- Comparing models to each other and deciding which model is better based on their performances.
- Oversampling to balance the dataset using SMOTE.
- Ensemble Models achieve the best performance which agrees with what obtained in our models since **Bagging Model** reaches the highest metrics.