CREDIT CARD FRAUD DETECTION

RUNSHENG ZHOU

Dr. Radhakrishnan Sridhar

SUMMER 2022

DSA-5900

4 Credits

Introduction

According to Survey of Consumer Payment Choice, in 2020, the Federal Reserve Bank of Atlanta claimed 3.5% of people who have used credit card have had credit card fraud-related issues in the past 1 year (Foster, 2022). Credit card fraud issue brought various social problems. The credit agents, card owners, banks, insurance companies and any other related parties need to spend amount of time and money to resolve the issue. Therefore, it is vital to detect the fraud transactions to ensure the transactions are made by card users rather than someone else. In order to accurately detect the fraud transactions, Machine learning algorithms and statistics techniques are involved to help us analyze the data.

Objectives

The primary purpose of this project is to build machine learning models and use them to classify whether a transaction is fraud or not.

Secondary purposes are:

- To explore and understand how the data is distributed.
- To get familiar with Over sampling and Under sampling. Using one of the method to generate a 50/50 sub-dataset.
- To determine the best classifiers which has the highest accuracy.
- To become familiar with machine learning techniques such as Logistic Regression, SVC, t-SNE, PCA, etc.
- In learning how to use Pandas, Sklearn, from Python to preprocess data and build models.

- In learning how to use Qlik Sense, Seaborn, plotly to generate proper visualizations.
- In learning how to interpret each of the plot in the project and model parameters.

Data

Exploratory Data Analysis:

The dataset is from Kaggle, named “Credit Card Fraud Detection”. It contains the transactions that are made by European credit card holders in September 2013.

This dataset only presents 2 days transactions. The dataset is highly imbalanced because there are only 492 fraud transactions and 284,315 non-fraud transactions, alternatively, only 0.17% of the transactions are fraud (*Figure 1*).

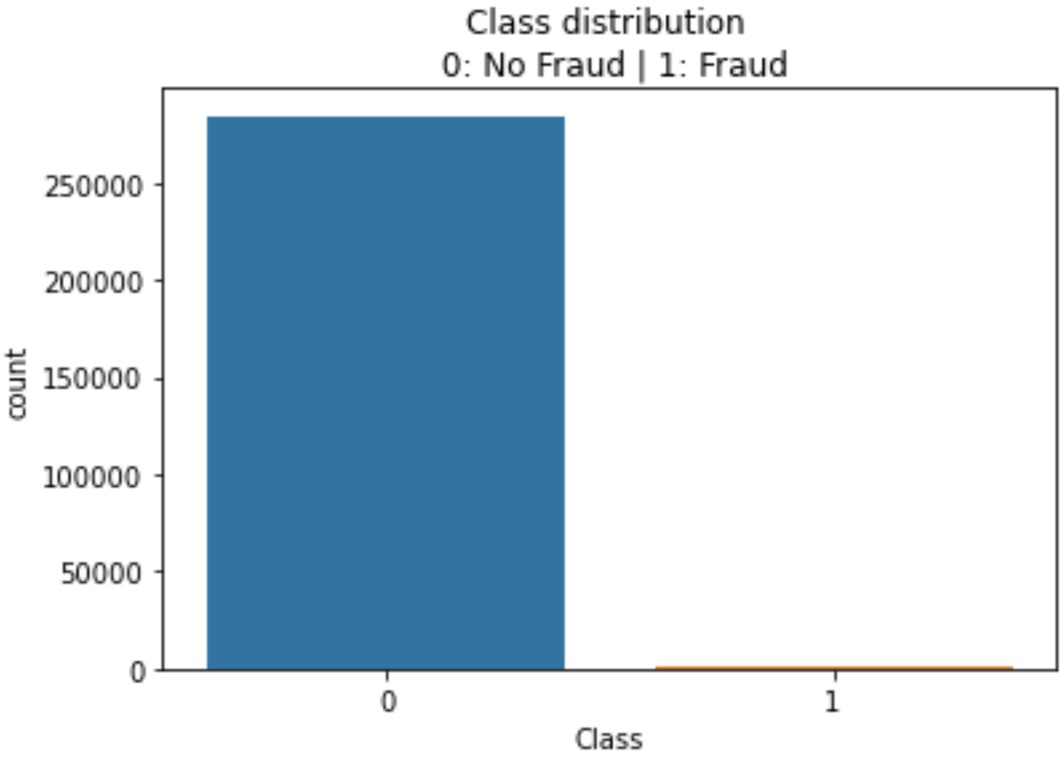
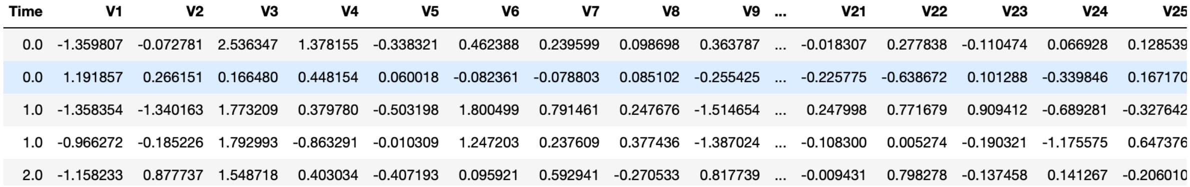


Figure 1: The Skewness in Two Different Classes

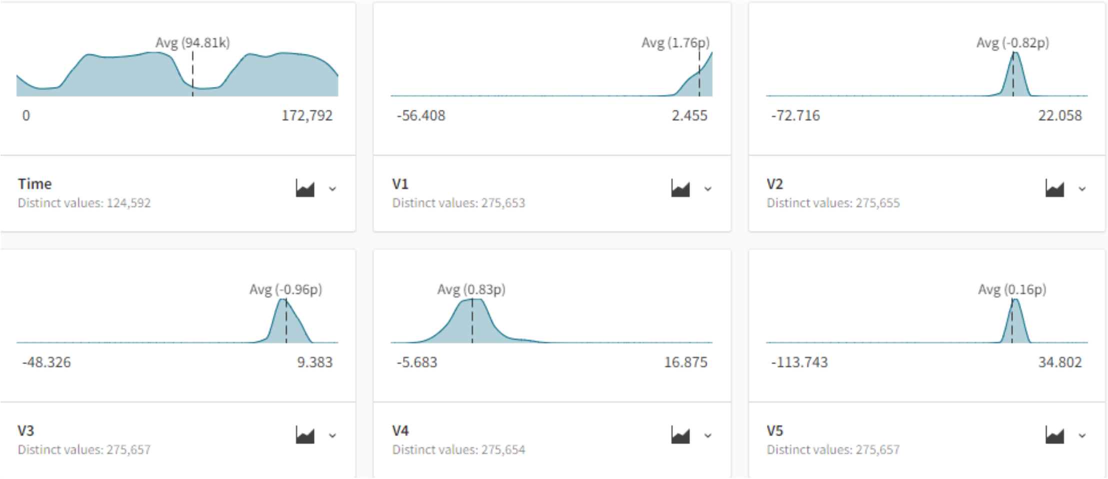
Because of some of the confidential issues, the original data features are not able to be provided. So, there is only a few background information about the data. Feature V1, V2, V3, …, V27, V28 are derived by using Principal Component Analysis (PCA). There are only two features, 'Time' and 'Amount', these are not transformed from PCA. The feature 'Time' recorded "number of seconds elapsed between the transaction and the first transaction in the dataset." The feature 'Amount' records the transaction amount. The feature 'Class' is a binary response variable, 1 = fraud, 0 = non-fraud (Kaggle).

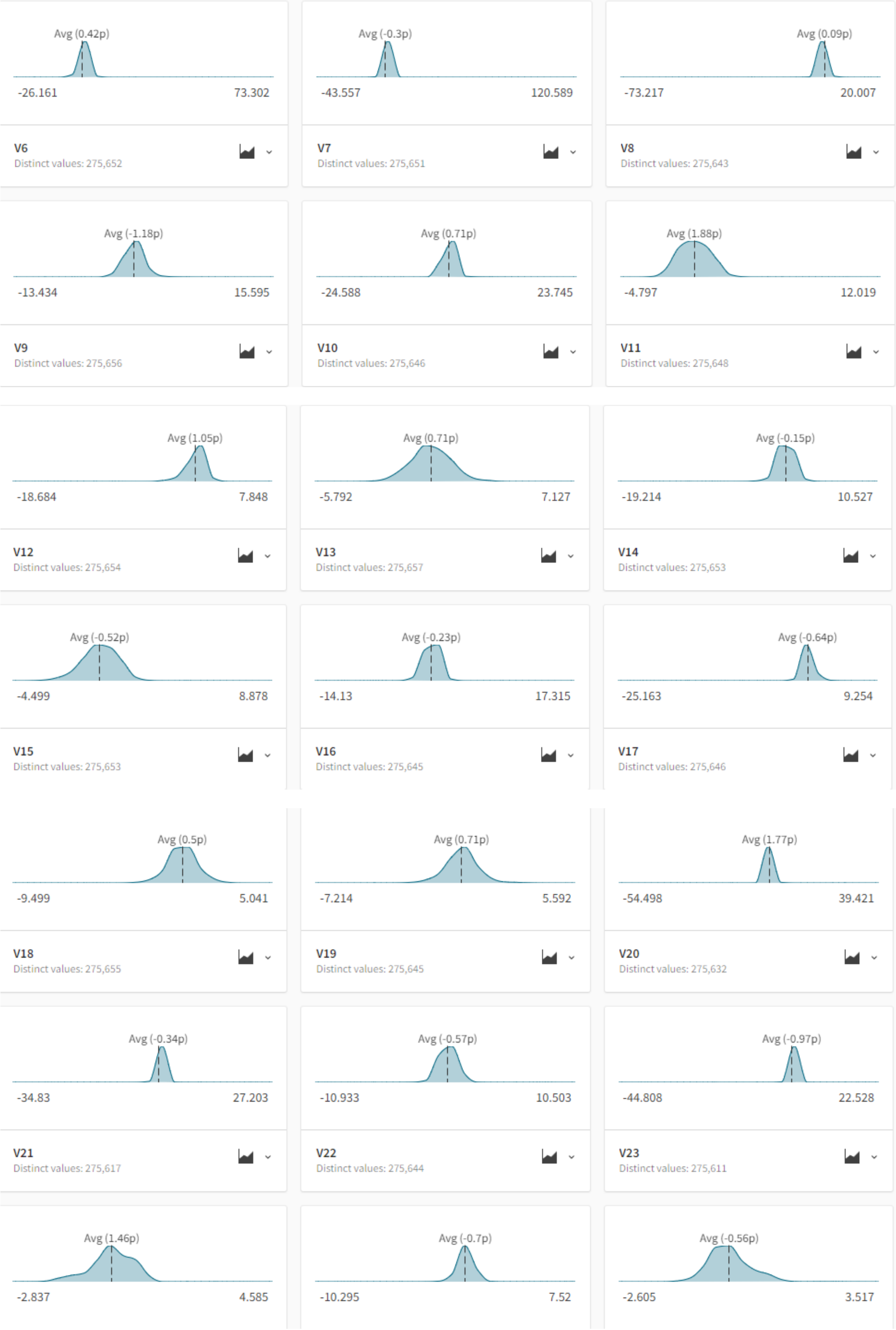


Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25
0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539
0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170
1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642
1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376
2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010

Figure 2 Sample of Data

Unlike most standard dataset. This one has no missing values, and there are only numerical variables. By using Qlik Sense, the distributions and distinct values of each variable are shown below.





V6
Distinct values: 275,652

V7
Distinct values: 275,651

V8
Distinct values: 275,643

V9
Distinct values: 275,656

V10
Distinct values: 275,646

V11
Distinct values: 275,648

V12
Distinct values: 275,654

V13
Distinct values: 275,657

V14
Distinct values: 275,653

V15
Distinct values: 275,653

V16
Distinct values: 275,645

V17
Distinct values: 275,646

V18
Distinct values: 275,655

V19
Distinct values: 275,645

V20
Distinct values: 275,632

V21
Distinct values: 275,617

V22
Distinct values: 275,644

V23
Distinct values: 275,611

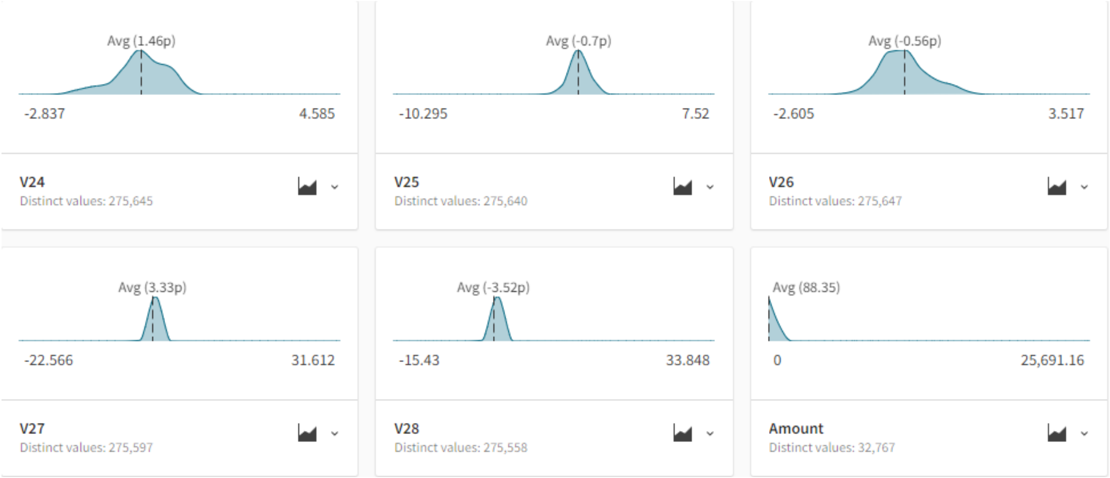


Figure 3 Variables Distribution

Although we do not know exactly the name of each variable, the relationships can be clearly visualized. Among all these features, they can be categorized into two types of features:

i. Negative Correlations with Class, the lower the feature value, the higher chance it will be a fraud transaction.

ii. Positive Correlations with Class, the higher the feature value, the more likely it will be a fraud transaction.

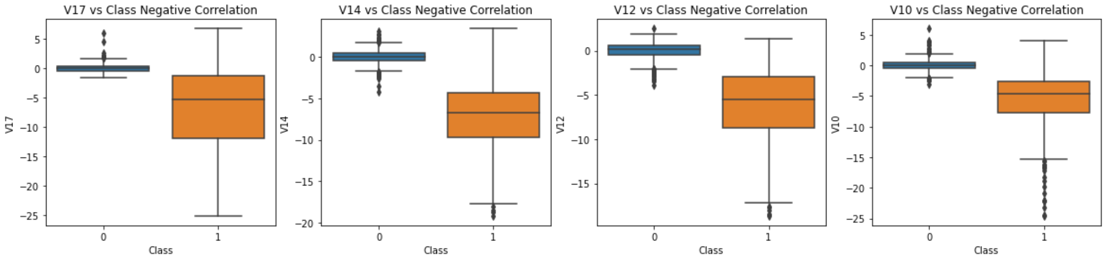


Figure 4 Boxplot of Variables Have Negative Correlation with Class

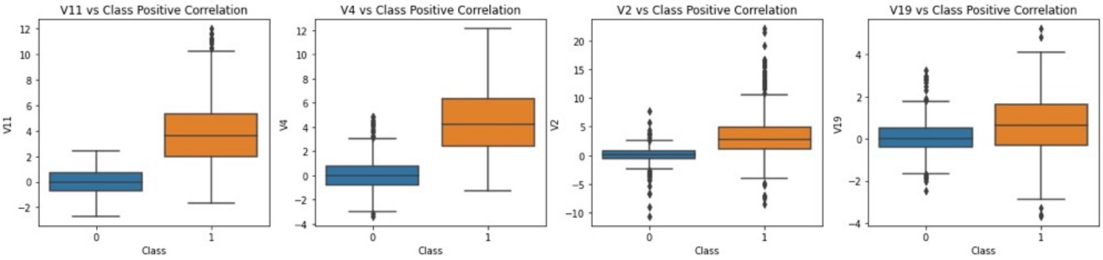


Figure 5 Boxplot of Variables Have Positive Correlation with Class

Outliers

One of the obstacles in this project is to choose what outliers need to be excluded. Because the data is highly imbalanced, we do not want to lose any information about fraud transactions. We will be using Interquartile Range to change the lower bound and upper bound, in other words, we want to expand our bounds in order to focus on the extreme outliers rather than just the outliers. The default way of calculating the lower bound is $Q1 - 1.5 \times IQR$, and the upper bound is $Q3 + 1.5 \times IQR$. But in order to only detect the major outliers, we use $Q1 - \mathbf{3.0} \times IQR$ for the lower bound, and $Q3 + \mathbf{3.0} \times IQR$ for the Upper bound. After the process, we eliminated 0.27% rows.

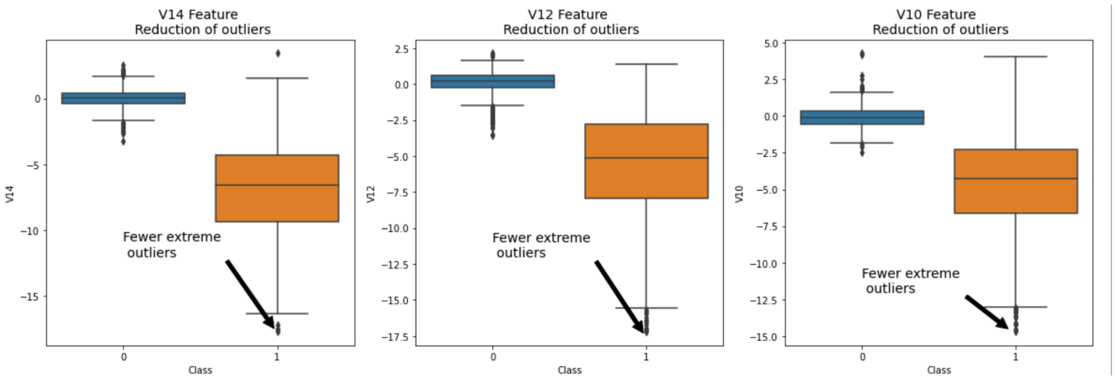


Figure 6 Taking off the extreme outliers

Methodology

Random Under Sampling

We hardly see the true correlations between the class and features if we just use the original imbalanced dataset. The result is shown below.

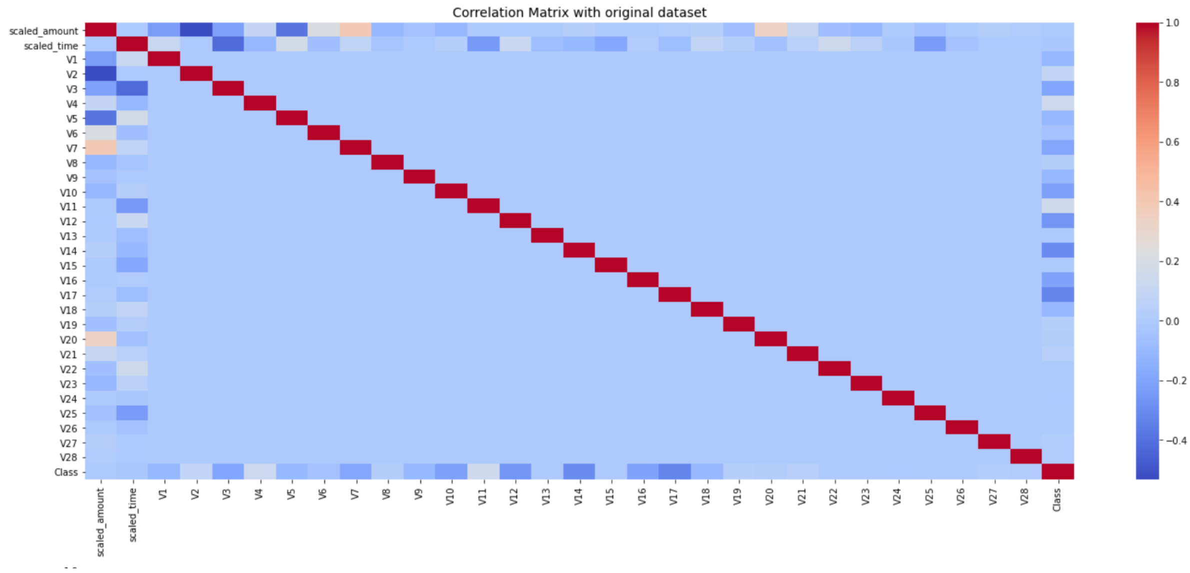


Figure 7 Correlation Matrix with Imbalanced Data

In order to solve this issue, we created a sub sample by using random sampling. Since we want our model to be as accurate as possible to detect fraud transactions. Hence, we will be using Random Under Sampling. So, we will take all the fraud transactions and same amount of random non-fraud transactions and shuffle them as the sub sample. Then the sub sample has 492 rows of the fraud and 492 rows of the non-fraud. We will be using this run cross validation and train the models in the later stages.

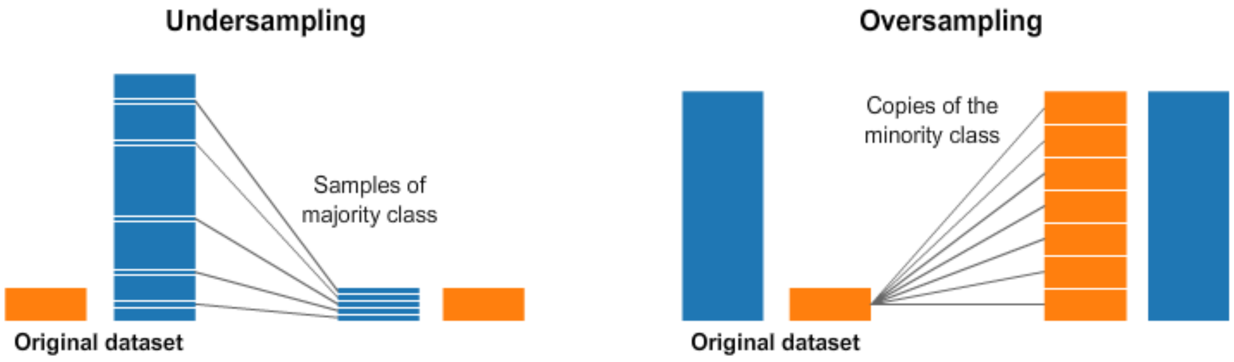


Figure 8 Random Sampling
(Source: https://medium.com/analytics-vidhya/undersampling-and-oversampling-an-old-and-a-new-approach-4f984a0e8392)

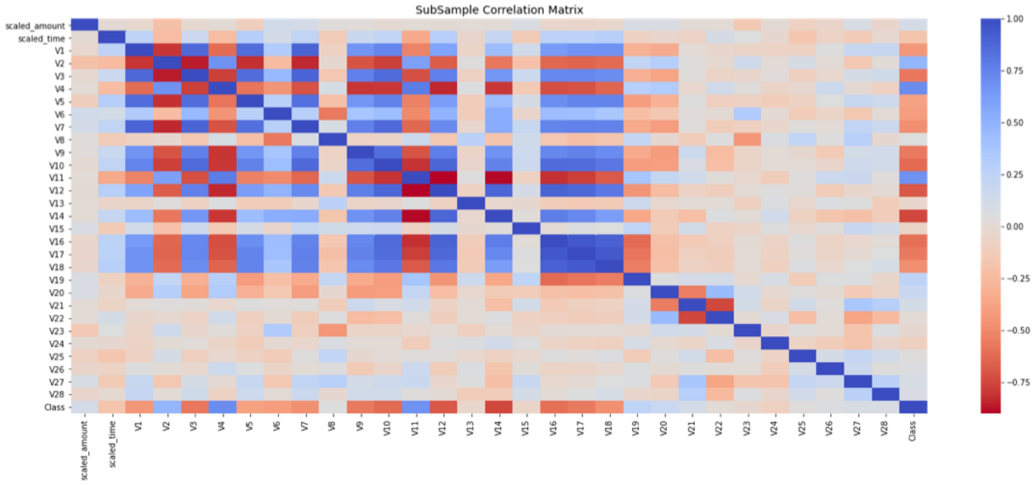


Figure 9 Correlation Matrix with Balanced Data

Dimension Reduction

Although the sub sample is relatively small, the T-distributed Stochastic Neighbor Embedding(T-SNE) algorithm still can detect clusters accurately. So does PCA, also a dimension reduction method to help us indicate whether the data is well clustered or not. If the data is well clustered, it is a good indication that further predictive models will perform well in separating fraud cases from non-fraud cases.

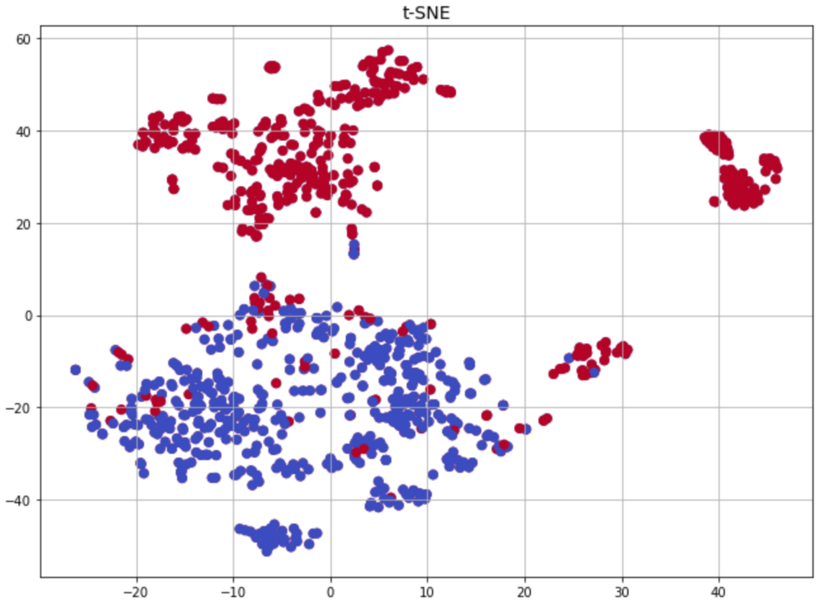


Figure 10 T-SNE (Sub Sample)

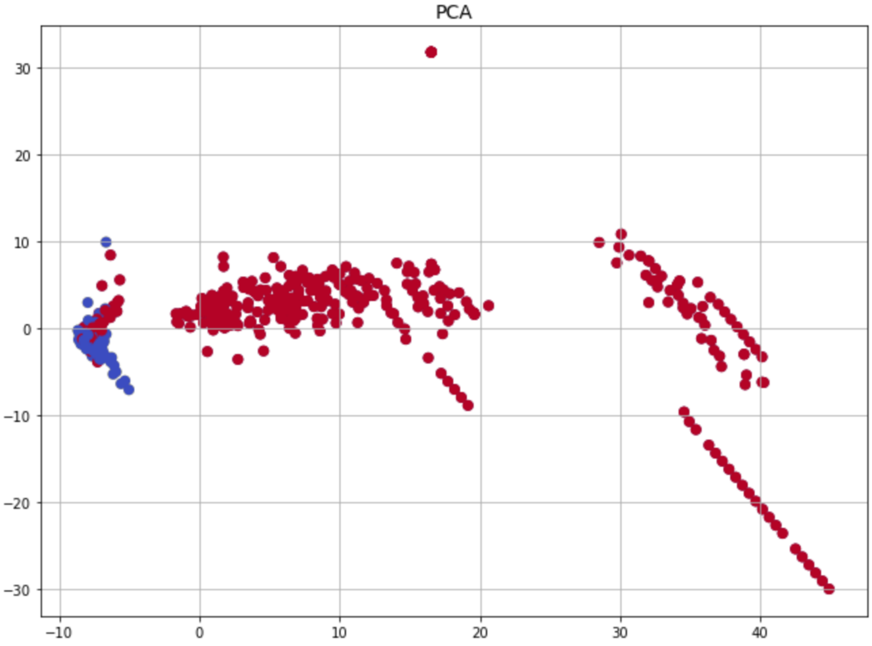


Figure 11 PCA (Sub Sample)

Classifiers

For this project, we will be using four types of machine learning algorithms, Logistic Regression, K-Nearest Neighbor Classifier, Support Vector Classifier and Decision Tree Classifier. We will be using our sub sample, we created in the early stage, to do cross-validation. Then we will be using GridSearchCV in Python package to determine the parameters that gives the best predictive score for each classifier in order to train our models.

The learning curves for each classifier are shown below. The wider the gap between the training score and the cross-validation score, the more likely our model is overfitting, which has high variance. And if the score is low in both training and cross-validation sets, this is an indication that our model has a high bias (Underfitting).

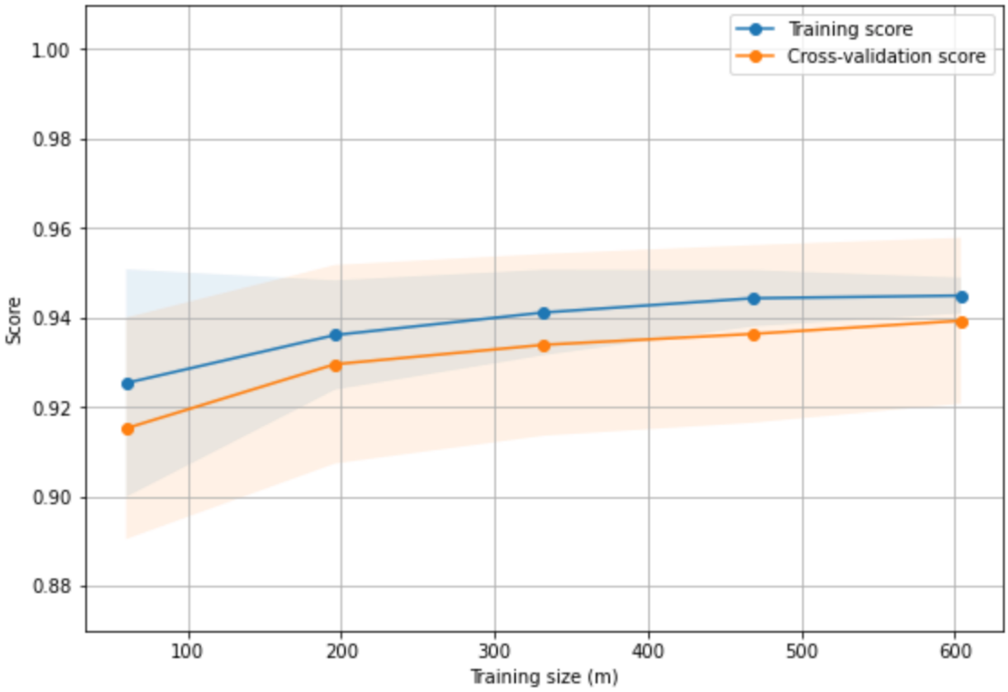


Figure 12 Logistic Regression Learning Curve

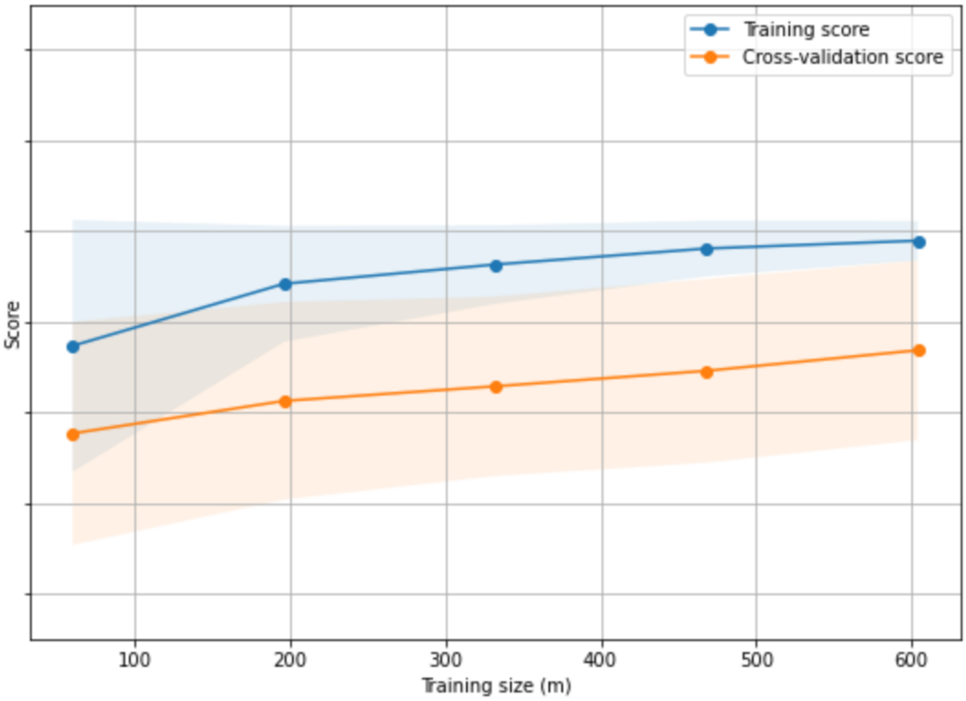


Figure 13 K-Nearest Neighbor Classifier Learning Curve

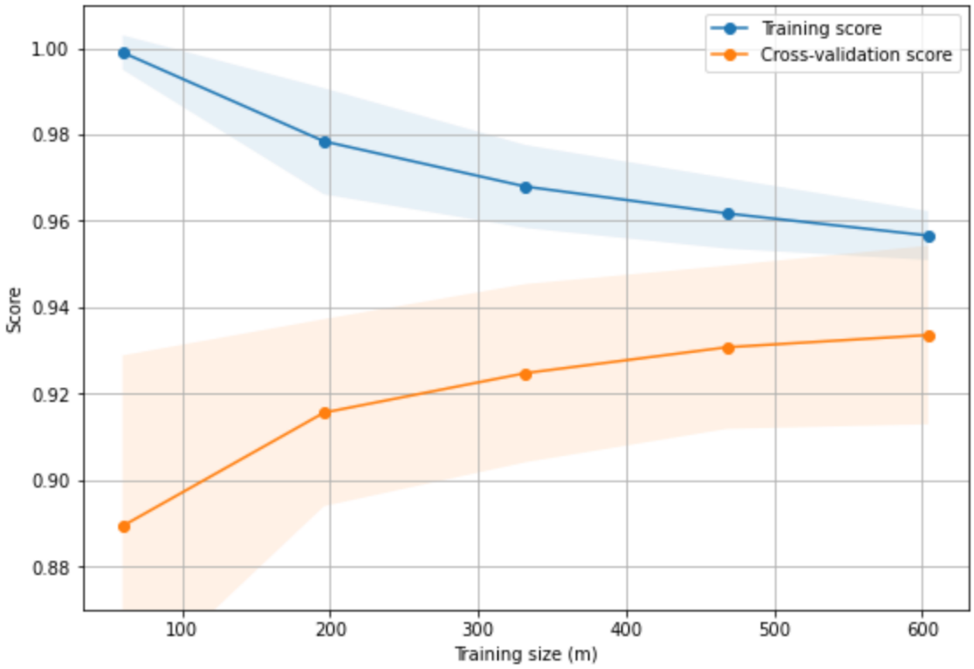


Figure 14 Support Vector Machine Classifier Learning Curve

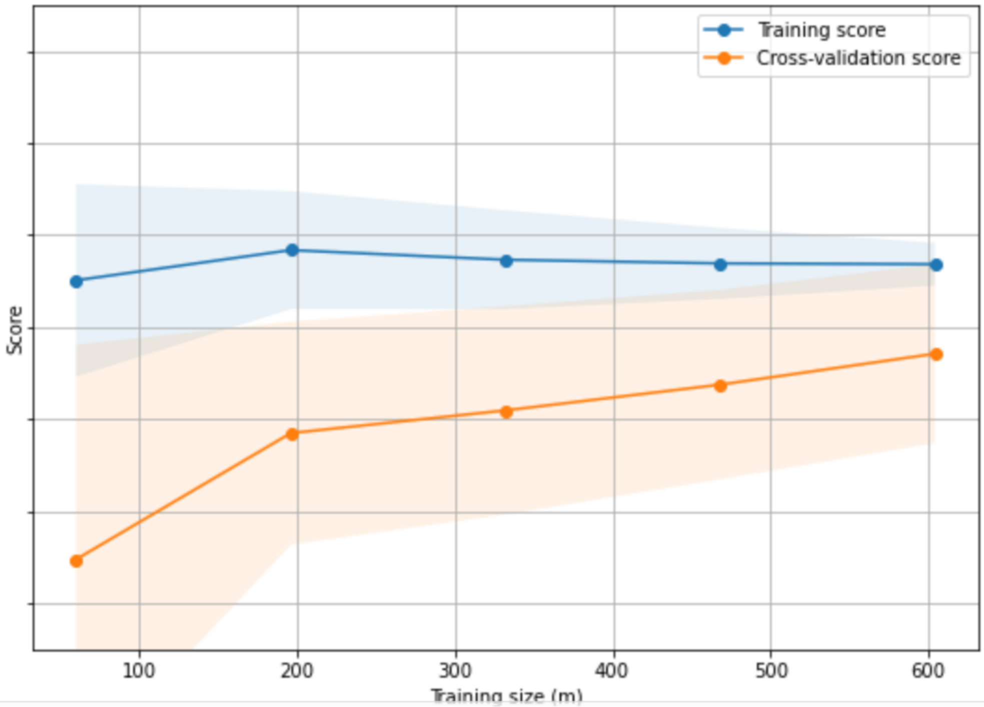


Figure 15 Decision Tree Classifier Learning Curve

The following figure shows how the model predicts in our sub sample test data. Although the test data is small, it still can give us a hint on which classifier is good at in predicting what certain values. For example, the logistic regression model has the most correct Ture Positive values, whereas the SVC has the most correct True Negative values. As previous mentioned, we want our model to be as sensitive as possible to the fraud transactions. Therefore, the negative recall rate is essential in our task. In this stage, we can tell that the k-Nearest Neighbor and SVC have higher recall rate than the other two classifiers.

$$Negative\ Recall = \frac{\text{True Negatives}}{\text{True Negatives } + \text{ False Positives}}$$

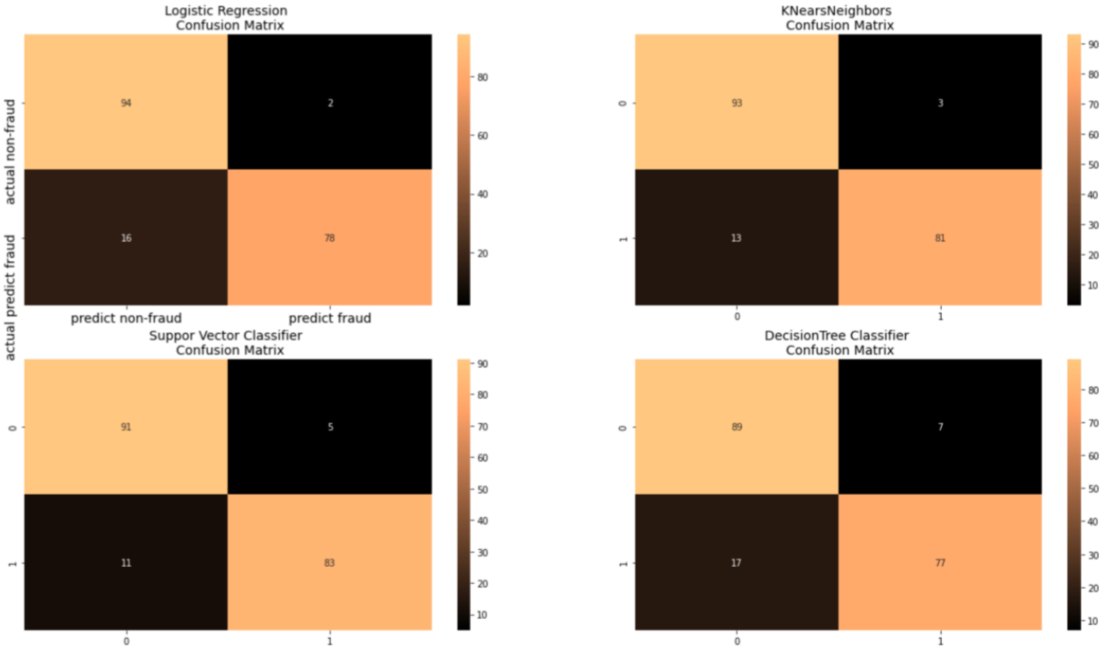


Figure 16 Confusion Matrix for (Sub Sample Test Data)

The Logistic Regression model and SVC give us more descent ROC Curves than the other two. Both the precision score and recall rate are high. Note: This is still the sub sample data. We will be applying our model with original test data in the result section.

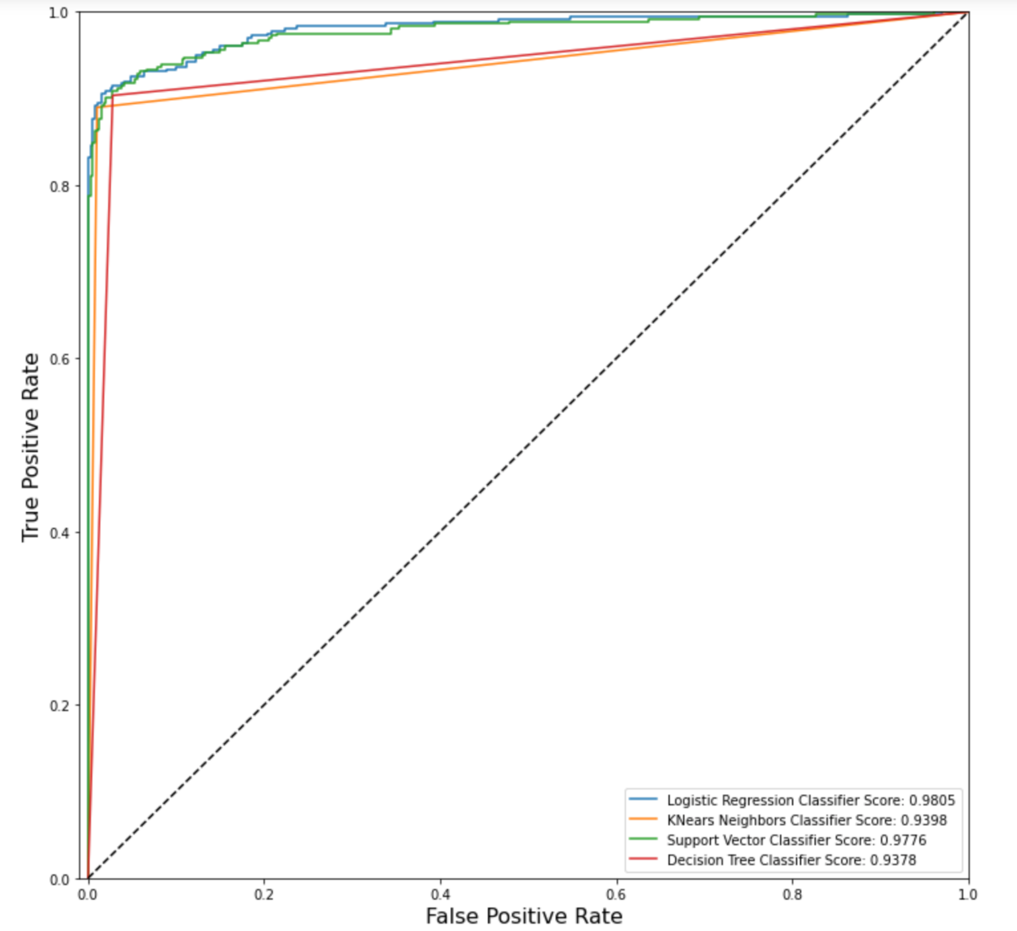


Figure 17 ROC Cure for The Classifiers.

Result

We applied the final models into our original test data. From the Summary statistic tables, we can tell that both Logistic Regression and k-Nearest Neighbor Classifier get high accuracy which is 98%. Since we are more focusing on the negative recall rate. k-Nearest Neighbor and SVC do better jobs. Overall, k-Nearest Neighbor Classifier is a more successful model among these classifiers.

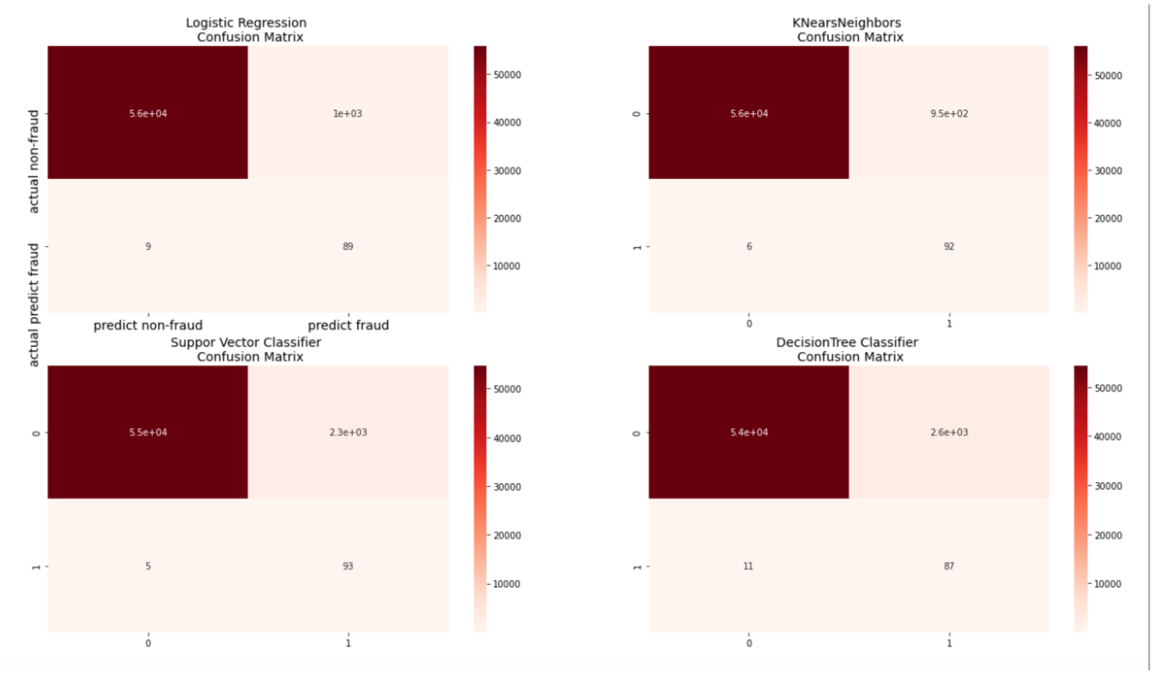


Figure 18 Confusion Matrix for the Classifiers (Original Test Data)

Table 1 Logistic Regression Summary Statistic

	Precision	Recall	F1-score	Support
non-fraud	1.00	0.98	0.99	56864
fraud	0.08	0.91	0.15	98
accuracy			**0.98**	56962

Table 2 k-Nearest Neighbor Summary Statistic

	Precision	Recall	F1-score	Support
non-fraud	1.00	0.98	0.99	56864
fraud	0.09	0.94	0.16	98
accuracy			**0.98**	56962

Table 3 Support Vector Classifier Summary Statistic

	Precision	Recall	F1-score	Support
non-fraud	1.00	0.96	0.98	56864
fraud	0.04	0.95	0.08	98
accuracy			**0.96**	56962

Table 4 Decision Tree Classifier Summary Statistic

	Precision	Recall	F1-score	Support
non-fraud	1.00	0.95	0.98	56864
fraud	0.03	0.89	0.06	98
accuracy			**0.95**	56962

Deliverables

Although this is a self-project with no other party, the algorithm may still help the credit agent to identify the majority fraud transactions. The project background information is on Kaggle, and the code is hosted on my personal GitHub.

Self-Assessment

Most of the techniques I applied in this project are from DSA program. For example, all the machine learning algorithm in this project I learned from Dr. Nicolson's course. But some of the techniques I was not very familiar with. Hence, I get a chance to master them a bit. I independently learned Random Under-sampling and apply it to this project. This practicum is 4 hours. And the topic was chosen among the DSA topic pool.

References

Al-Serw, N. A.-R. (2021, February 21). *Undersampling and oversampling: An old and a new approach*. Medium. Retrieved July 17, 2022, from https://medium.com/analytics-vidhya/undersampling-and-oversampling-an-old-and-a-new-approach-4f984a0e8392

Kevin Foster, Claire Greene, Joanna Stavins. (2022). *The 2020 survey of Consumer Payment Choice: Summary Results*. The 2020 Survey of Consumer Payment Choice. Retrieved July 17, 2022, from https://www.atlantafed.org/-/media/documents/banking/consumer-payments/survey-of-consumer-payment-choice/2020/2020-survey-of-consumer-payment-choice.pdf

Sklearn.svm.SVC. scikit. (n.d.). Retrieved July 17, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

ULB, M. L. G.-. (2018, March 23). *Credit Card Fraud Detection*. Kaggle. Retrieved July 17, 2022, from https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud