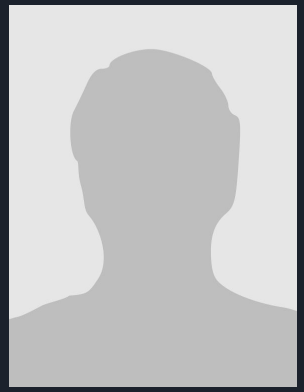
Group Project Presentation

Credit Card Fraud Detection Using AdaBoost Algorithm

CSCI 4340 Machine Learning

Lecturer Name: Dr. AMELIA RITAHANI BT. ISMAIL

Team Member:



Mohamed Moubarak Mohamed Misbahou Mkouboi
(1820705)

Karim Fazlul

(1822049)

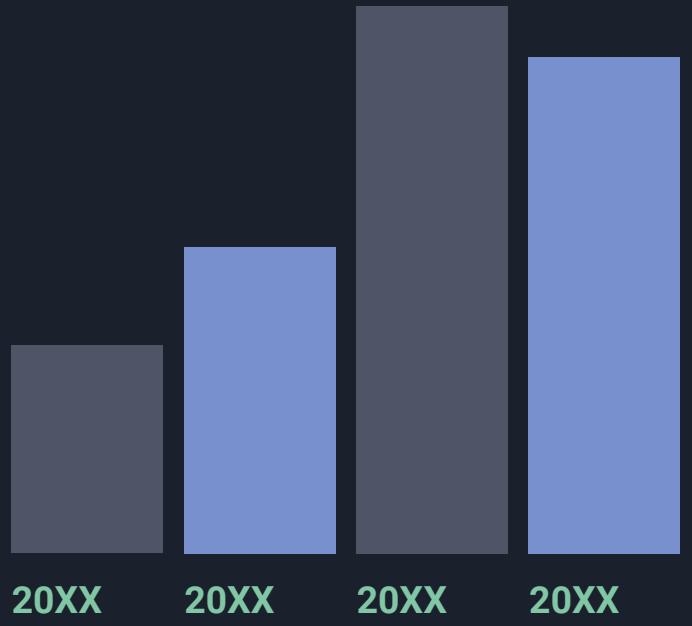
Mohammad Afif Muhajir

(1816203)

Introduction

Introduction

- The use of breakthrough in technology such as Artificial Intelligence and Machine Learning provides many benefits.
- Trying to tackle pattern detection problem in the cases of credit card fraud.
- Attempt of using Machine Learning algorithm “AdaBoost” as prediction model for solving the problem being mentioned.



Objectives

What we are trying to achieve by doing this project

Project Objectives

To develop a working model to solve Credit Card Fraud Detection problem

To research and do proper implementation of AdaBoost Algorithm

To evaluate the accuracy of AdaBoost with other Algorithm in solving the problem

Model Objectives

To determine connection between each variable for identifying credit card fraud.

To discover the biggest factor in determining a credit card fraud in credit card transaction dataset

To understand if the desired algorithm can accurately credit card fraud data with a provided dataset

EXPERIMENTAL SETUP

Literature Review

Introduction to Credit Card Fraud

A credit card is a bank-issued card that enables its owner to pay for products and services at businesses that accept card payments. Each credit card has a defined amount of cash (credit) that you may access through the card and use in advance to purchase goods and services, paying back the funds at the end of the month or according to the repayment schedule with or without interest charges. Credit card fraud can also occur when a credit card number is used without the physical card being present. This is a more dangerous type of credit card fraud since it works in tandem with identity theft.

Credit Card Default and Its Consequences

A credit card default happens when you fail to pay the minimum amount owing for multiple months. Contrary to popular belief, it takes more than a handful of missed payments to fall into default. A default notice is usually delivered after six months of missing payments. If your card issuer is unhappy with your response, your account will be cancelled and a no-payment report will be issued to credit bureaus.

However, the ramifications of credit card fraud are

The account sent to collections

A credit card company can shut down your account or send the debt to a collection agency, which may sue you, or they can sue you. This may be highly inconvenient since you may receive a flood of calls from the collection agency and may face a lien on your wages.

Legal action

Some creditors are more aggressive than others. If they need an urgent resolution, they may file a local court judgment against you, which might result in a payroll lien, which requires your company to transmit a percentage of your salary straight to your creditors.

Decrease your credit score

The most significant thing on your credit record is a late payment. Failure to make regular payments for six months might result in a loss of hundreds of points on your credit score, which can take years to restore.

Increase in interest rates

If you are 60 days late on a payment, your interest rate will most likely skyrocket. Because you're skipping payments and carrying a balance, your interest payments will only increase, making it much more difficult to settle the loans.

Decrease in credit limit

If you fail on a credit card, your credit limit on other cards may be reduced. Lower credit limits bring additional issues, such as an increase in your credit use rate.

Machine Learning Used

We will examine the k-nearest Neighbour (KNN) method with boosting and the Adaboosting technique. The boosting algorithm is used to improve the performance of KNN, which has a high classification power. In other words, we'll see which method performs best: a simple KNN algorithm or a hybrid with boosting. We will compare the accuracy and performance of the Adaboost algorithm to those of alternative methods such as logistic, SVM, Decision Tree.

Machine Learning Used

Logistics Regression

Logistic regression is a supervised classification technique at its core. For a given collection of characteristics (or inputs), X, the target variable (or output), y, can only take discrete values in a classification issue. Logistic regression, contrary to common opinion, is a regression model.

k-Nearest Neighbor

KNN is an acronym for "K-Nearest Neighbour." It is a machine learning algorithm that is supervised. The algorithm can tackle classification and regression problem statements. The sign 'K' represents the number of nearest neighbours to a new unknown variable that must be predicted or categorized.

SVM (Support Vector Machine)

The SVM algorithm goal is to identify a hyperplane in an N-dimensional space that classifies the input points. It may be used for both classification and regression, but it has issues that are best suited for categorization.

Machine Learning Used

Decision Tree

Decision Trees are a sort of Supervised Machine Learning. Data is continually separated based on a certain parameter. The decisions or consequences are represented by the leaves. And the data is separated at the decision nodes, which are the products of the previous training data.

AdaBoost Algorithm

AdaBoost, also known as Adaptive Boosting, is a Machine Learning approach. AdaBoost employs an iterative strategy to improve poor classifiers by learning from their mistakes. The most frequent AdaBoost method is decision trees with one level, which are often referred to as Decision Stumps.

Modeling

General Machine Learning Algorithms Procedure

Shown below is the steps and figure of implementation of the general process of using classification machine learning algorithms:

- Load up libraries and dataset
- Do dataset split into training and testing sets
- Proceed to do anomaly detection and clearance
- Apply the appropriate machine algorithm function
- Proceed to do training and testing of the model
- Finally, evaluate the accuracy result.

Coding for implemented algorithms:

Logistics Regression

```
log_pred = log_reg.predict(X_test)
print('---' * 45)
print("Logistic")
recall_scores.append(recall_score(y_test, log_pred))
print('Recall Score: {:.2f}'.format(recall_score(y_test, log_pred)))
precision_scores.append(precision_score(y_test, log_pred))
print('Precision Score: {:.2f}'.format(precision_score(y_test, log_pred)))
f1_scores.append(f1_score(y_test, log_pred))
print('F1 Score: {:.2f}'.format(f1_score(y_test, log_pred)))
accuracy_scores.append(accuracy_score(y_test, log_pred))
print('Accuracy Score: {:.2f}'.format(accuracy_score(y_test, log_pred)))
print('---' * 45)
```

k-Nearest Neighbor

```
svc_pred = svc.predict(X_test)
print('---' * 45)
print("SVM")
recall_scores.append(recall_score(y_test, svc_pred))
print('Recall Score: {:.2f}'.format(recall_score(y_test, svc_pred)))
precision_scores.append(precision_score(y_test, svc_pred))
print('Precision Score: {:.2f}'.format(precision_score(y_test, svc_pred)))
f1_scores.append(f1_score(y_test, svc_pred))
print('F1 Score: {:.2f}'.format(f1_score(y_test, svc_pred)))
accuracy_scores.append(accuracy_score(y_test, svc_pred))
print('Accuracy Score: {:.2f}'.format(accuracy_score(y_test, svc_pred)))
print('---' * 45)
```

SVM (Support Vector Machine)

```
svc_pred = svc.predict(X_test)
print('---' * 45)
print("SVM")
recall_scores.append(recall_score(y_test, svc_pred))
print('Recall Score: {:.2f}'.format(recall_score(y_test, svc_pred)))
precision_scores.append(precision_score(y_test, svc_pred))
print('Precision Score: {:.2f}'.format(precision_score(y_test, svc_pred)))
f1_scores.append(f1_score(y_test, svc_pred))
print('F1 Score: {:.2f}'.format(f1_score(y_test, svc_pred)))
accuracy_scores.append(accuracy_score(y_test, svc_pred))
print('Accuracy Score: {:.2f}'.format(accuracy_score(y_test, svc_pred)))
print('---' * 45)
```

Decision Tree

```
tree_pred = tree_clf.predict(X_test)
print('---' * 45)
print("DT")
recall_scores.append(recall_score(y_test, tree_pred))
print('Recall Score: {:.2f}'.format(recall_score(y_test, tree_pred)))
precision_scores.append(precision_score(y_test, tree_pred))
print('Precision Score: {:.2f}'.format(precision_score(y_test, tree_pred)))
f1_scores.append(f1_score(y_test, tree_pred))
print('F1 Score: {:.2f}'.format(f1_score(y_test, tree_pred)))
accuracy_scores.append(accuracy_score(y_test, tree_pred))
print('Accuracy Score: {:.2f}'.format(accuracy_score(y_test, tree_pred)))
print('---' * 45)
```

AdaBoost Algorithm

```
# make predictions using adaboost for classification
from sklearn.datasets import make_classification
from sklearn.ensemble import AdaBoostClassifier
# define the model
model = AdaBoostClassifier()
# fit the model on the whole dataset
model.fit(X_train, y_train)
# make a single prediction
boosted_pred = model.predict(X_test)

print('---' * 45)
print("AdaBoosting")
recall_scores.append(recall_score(y_test, boosted_pred))
print('Recall Score: {:.2f}'.format(recall_score(y_test, boosted_pred)))
precision_scores.append(precision_score(y_test, boosted_pred))
print('Precision Score: {:.2f}'.format(precision_score(y_test, boosted_pred)))
f1_scores.append(f1_score(y_test, boosted_pred))
print('F1 Score: {:.2f}'.format(f1_score(y_test, boosted_pred)))
accuracy_scores.append(accuracy_score(y_test, boosted_pred))
print('Accuracy Score: {:.2f}'.format(accuracy_score(y_test, boosted_pred)))
print('---' * 45)
```

Data Analysis

Dataset Description

Dataset Description

Our dataset consists of 284,807 credit cards transactions instances made by European cardholders and 30 attributes

No.	Attribute Name	Description
1	Time	Integer, time unit value
2	Amount	Float, continuous value
3	V1	Float, continuous value
4	V2	Float, continuous value
5	V3	Float, continuous value
6	V4	Float, continuous value
7	V5	Float, continuous value
8	V6	Float, continuous value
9	V7	Float, continuous value
10	V8	Float, continuous value
11	V9	Float, continuous value
12	V10	Float, continuous value
13	V11	Float, continuous value
14	V12	Float, continuous value
15	V13	Float, continuous value
16	V14	Float, continuous value
17	V15	Float, continuous value
18	V16	Float, continuous value
19	V17	Float, continuous value
20	V18	Float, continuous value
21	V19	Float, continuous value
22	V20	Float, continuous value
23	V21	Float, continuous value
24	V22	Float, continuous value
25	V23	Float, continuous value
26	V24	Float, continuous value
27	V25	Float, continuous value
28	V26	Float, continuous value
29	V27	Float, continuous value
30	V28	Float, continuous value

Data Preprocessing

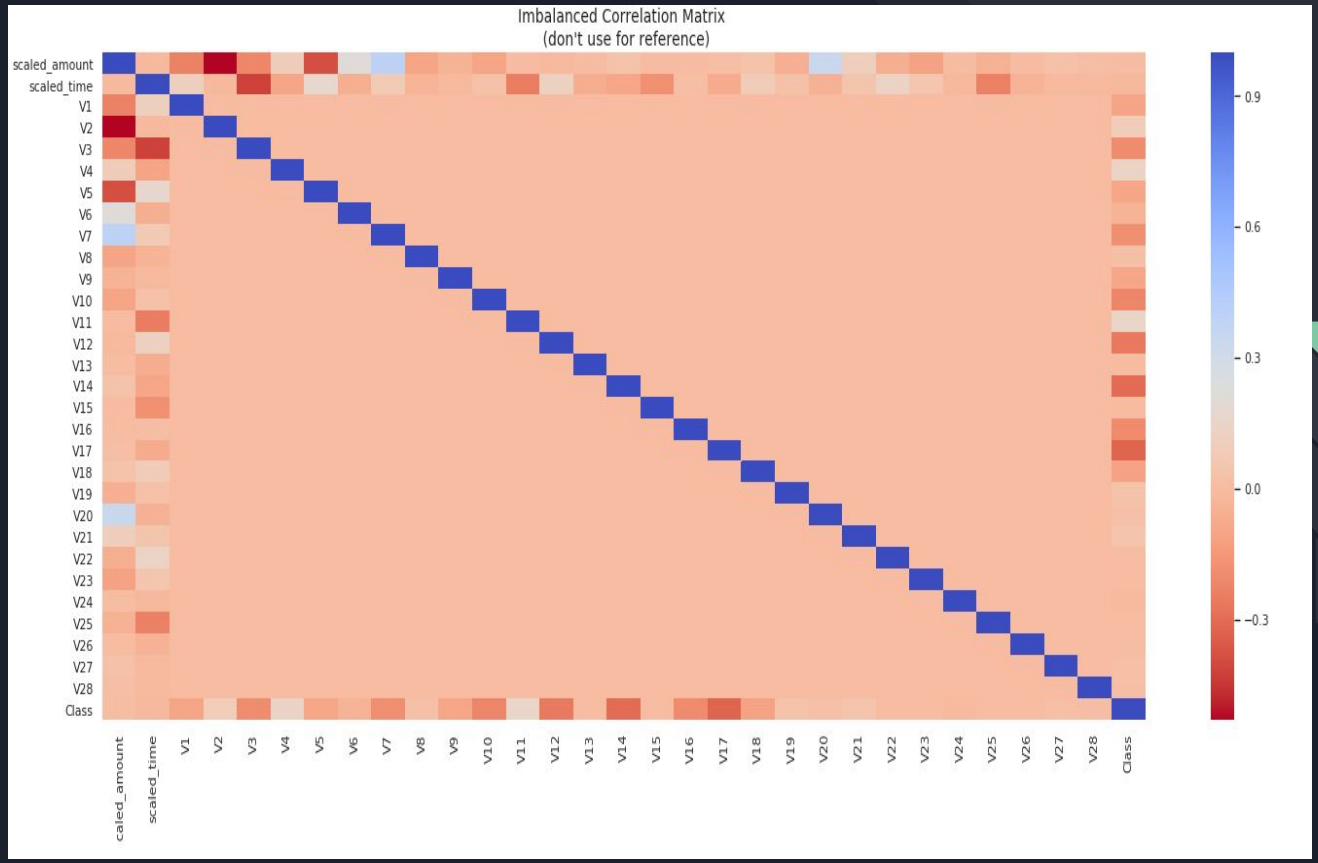
Anomaly Detection

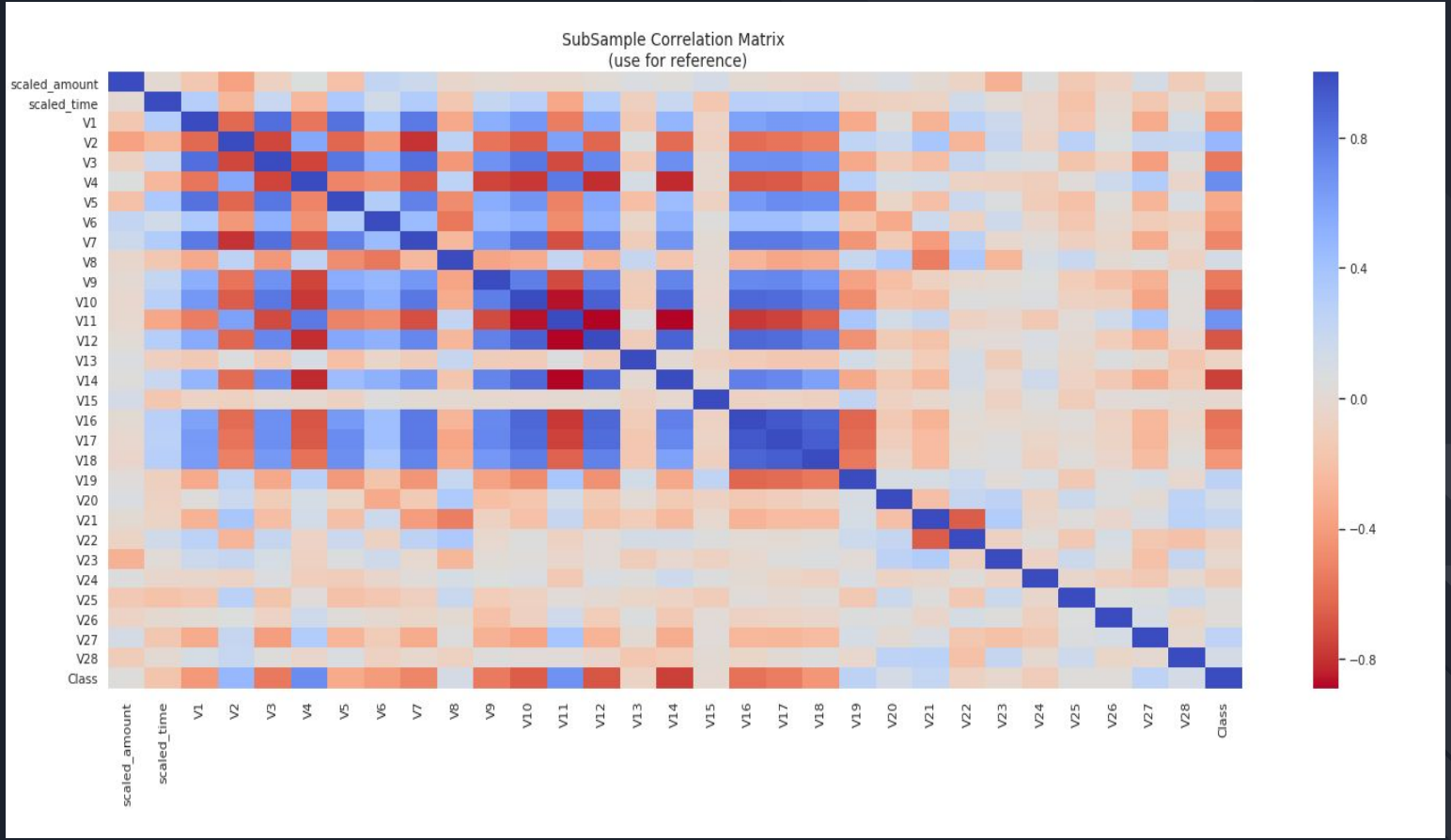
In our process of decreasing the chances of inaccuracy in the developed machine learning model, we processed the data through anomaly detection for outliers by following these steps:

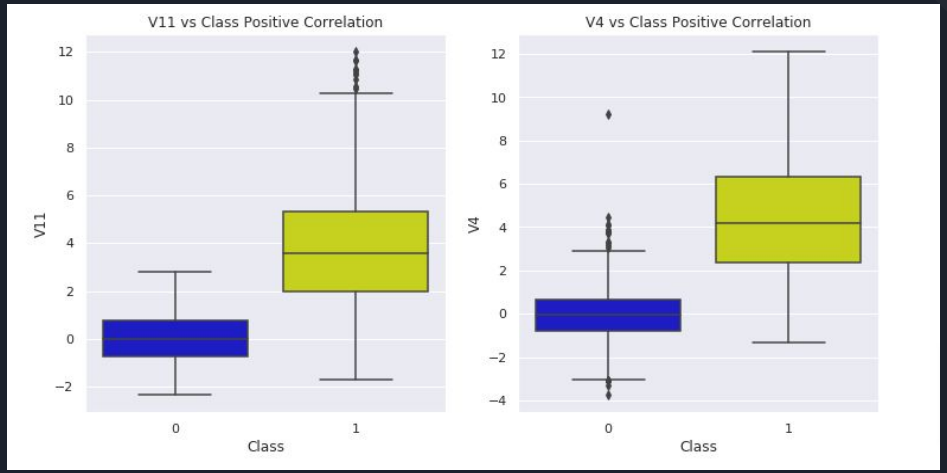
- Visualize Distribution
- Determining threshold
- Conditional Dropping
- Boxplot Representation

Data Correlation

In this part of the Data Analysing, by doing data correlation we have acquired necessary data representation for understanding the prime attributes that can help identifying which credit card transaction data is actually a fraud


Imbalanced Correlation Matrix
(don't use for reference)
scaled_amount
scaled_time
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V20
V21
V22
V23
V24
V25
V26
V27
V28
Class
0.9
0.6
0.3
0.0
−0.3


SubSample Correlation Matrix
(use for reference)
scaled_amount
scaled_time
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V20
V21
V22
V23
V24
V25
V26
V27
V28
Class
0.8
0.4
0.0
−0.4
−0.8



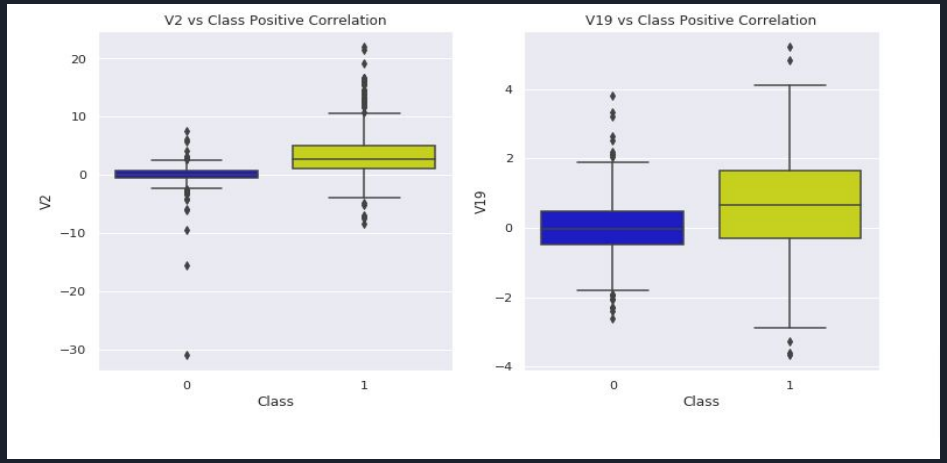
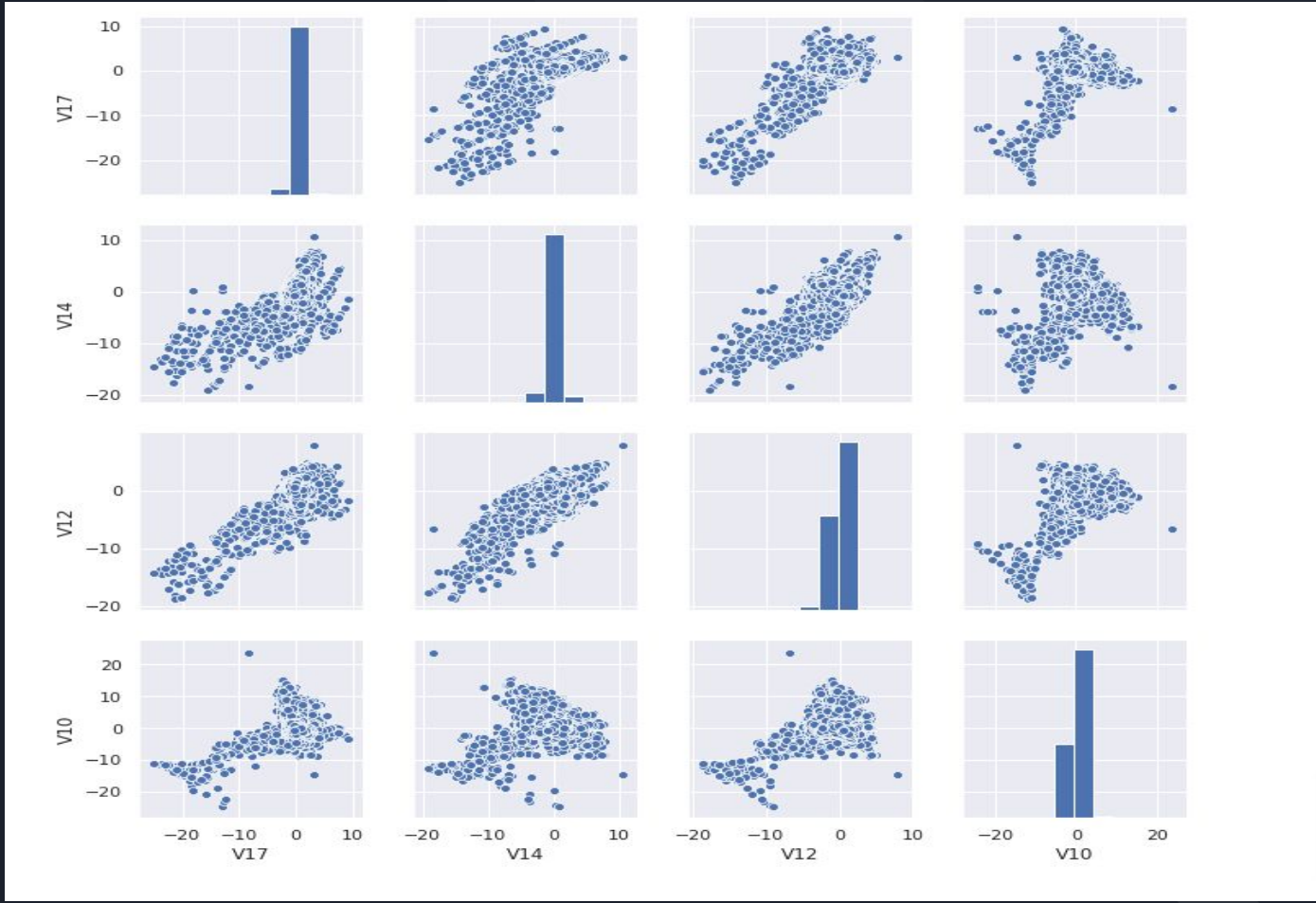


Diagram representation of attributes that contributes highly to identifying fraudulent credit card transaction cases

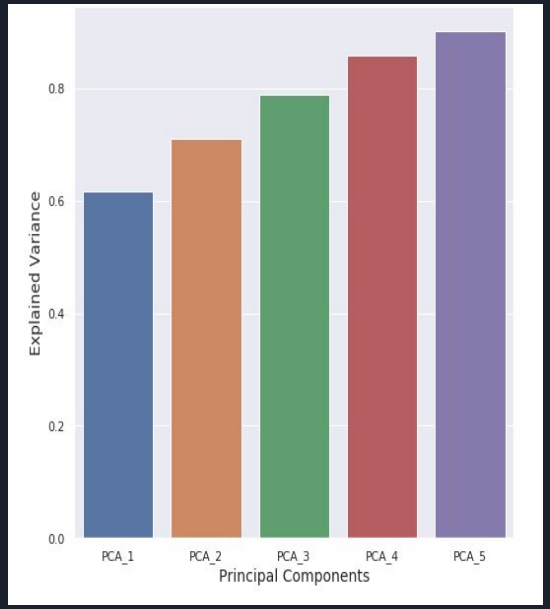
Resulting Boxplot of Strongly Correlated Attributes

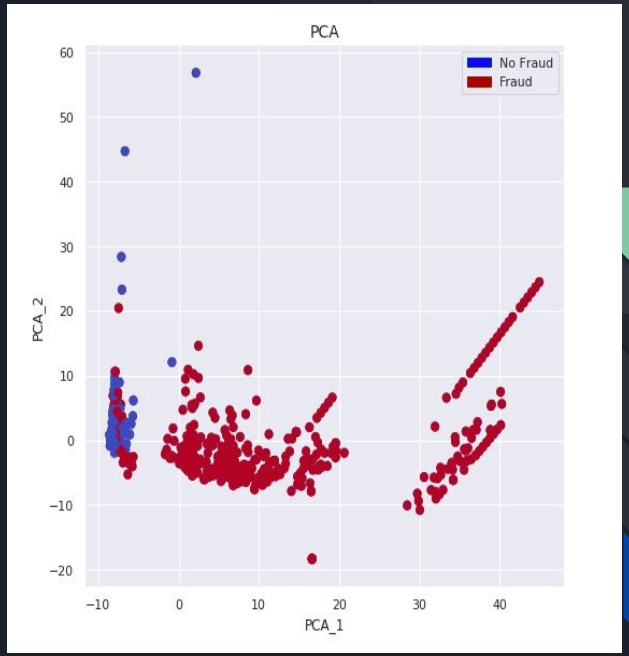


Principal Component Analysis

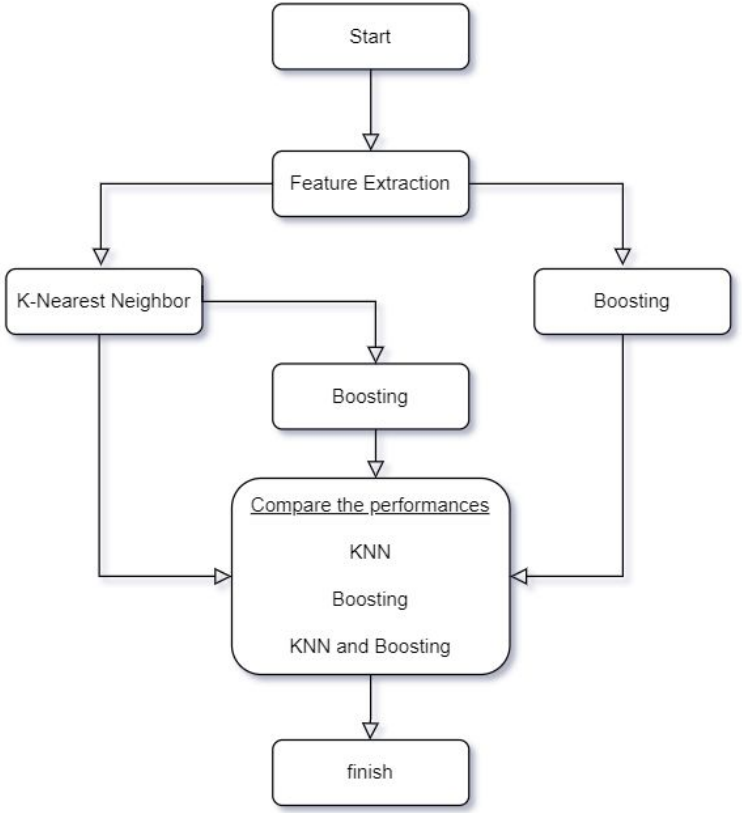
In order for the dataset to provide a greater accuracy to the developed machine learning model it has gone through the process of Principal Component Analysis (PCA) to resize the dataset size into a more compact data sample size

Result of PCA procedure

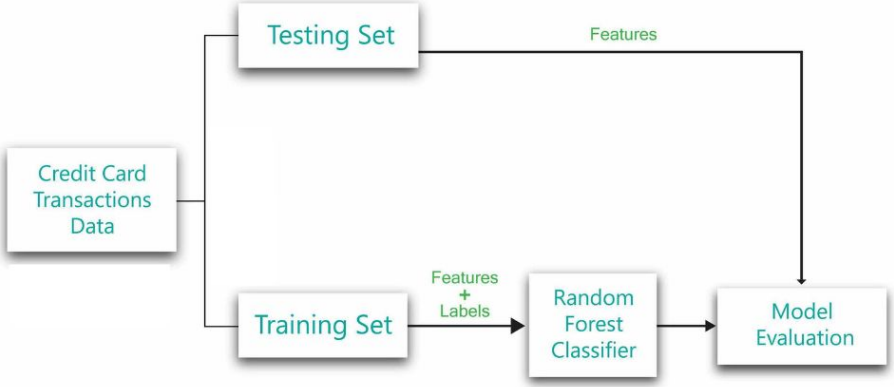




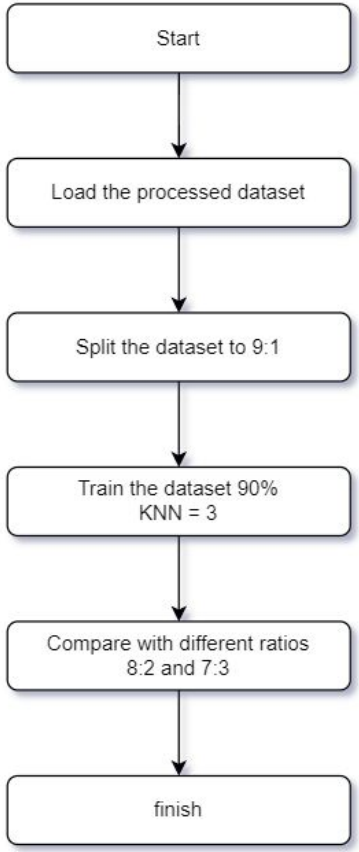
Model Development



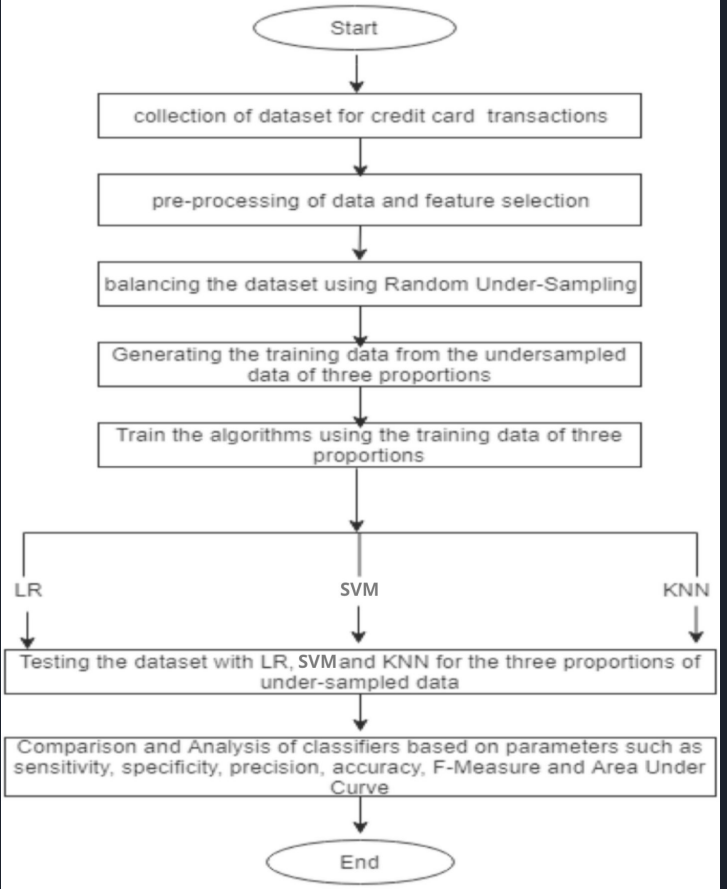
Flowchart for model development



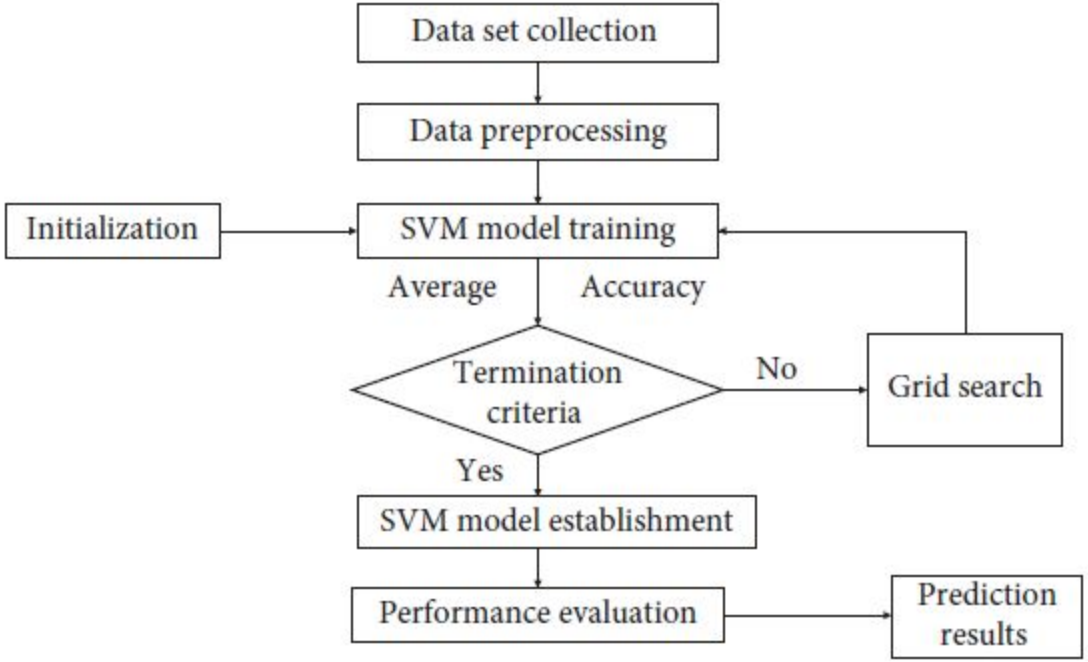
Flowchart for Decision Tree



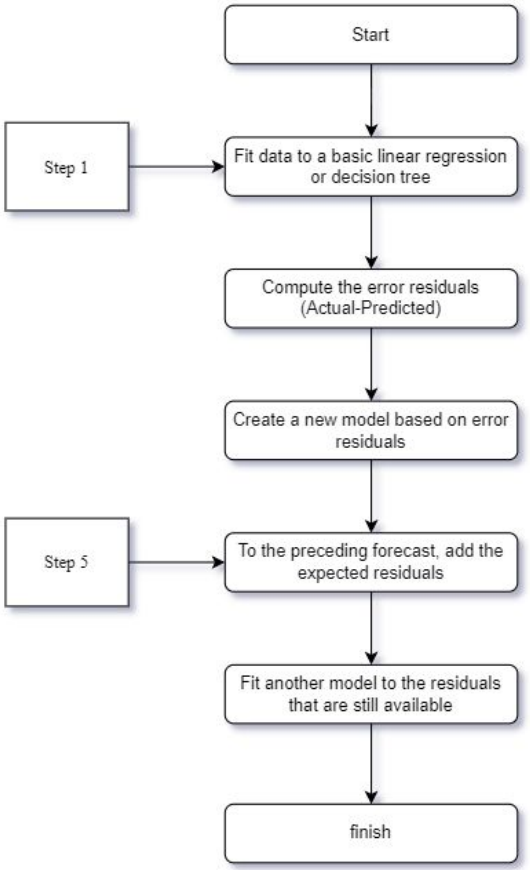
Flowchart for K-Nearest Neighbours algorithm



Flowchart for Logistics Regression



Flowchart of SVM.



Flowchart in AdaBoost algorithm

Result and Discussion

Results of accuracy from the myriad of Classification Machine Learning models

No.	Algorithm/Model Name	Accuracy Score	F1 Score	Recall Score	Precision Score
1	Logistics Regression	0.94	0.94	0.92	0.97
2	k-Nearest Neighbors	0.93	0.94	0.89	0.98
3	SVM	0.93	0.94	0.93	0.94
4	Decision Tree	0.91	0.95	0.92	0.93

We will use these results of accuracy and f1-score to be compared with AdaBoost Algorithm performances to determine which method is better suited for this problem of credit card fraud detection and dataset.

Result of accuracy from AdaBoost Algorithm model

No.	Algorithm/Model Name	Accuracy Score	F1 Score	Recall Score	Precision Score
1	Adaptive Boosting	0.93	0.93	0.92	0.94

THANK YOU !!!