Credit Card Fraud Detection

Chetan Choudhary, Mudit Dixit, Poojitha Mutteneni, Priyanka Gopi
Department of Computer Science, University of Central Florida
chetan@knights.ucf.edu, mudit.dixit@knights.ucf.edu, Poojitha@knights.ucf.edu, PriyankaGopi@Knights.ucf.edu

Abstract - The goal of creating a cashless, digital society has been a constant in modern civilization. Digital wallets and the use of credit card payment mechanisms have made it possible for online and offline payments without cash. However, Credit Card Payment Methods are vulnerable to fraud. Modern fraud does not require the attackers to be present physically at the scene of the crime. They have a variety of techniques to conceal their identity and can carry out these attacks in the privacy of their houses. Our Proposed System aims to design and develop a fraud detection method by analyzing the past transaction details of the customer, based on the European Dataset. Supervised Machine Learning Algorithms were applied to predict and classify the data based on which accuracy and F1 scores are calculated.
Keywords: DetectionFraud, System, Machine Learning Algorithms, Accuracy, F1.

I. Introduction

This study aims to distinguish between fraudulent and legitimate credit card transactions and to apply machine learning algorithms to effectively and reliably anticipate fraud. Based on the nature of the fraudulent acts, there are several sorts of credit card fraud, such as card theft, obtaining cards using fake information, people using credit cards while unable to pay bills, and bank staff stealing card details to use it remotely.
We have been keeping up with the news on information security and financial crime over the course of the past academic year because these topics are crucial to both online and offline financial transaction systems. Even if a small fraction of most medium-sized credit card transactions are fraudulent, when a consumer is unlucky enough to use a credit card, a financial loss to the firm and a crisis of customer confidence can result. According to several assessments, credit card fraudsters can simply achieve their goals. Large sums of money can exchange hands quickly without any indication of risk and with the owner's consent.

As a result, we have strong enough motivation to desire to train a pass-through machine learning classification algorithm in order to improve credit card fraud detection.

A. Background

Credit card fraud is a type of financial fraud that involves the unauthorized use of a credit card to make purchases or withdraw cash. It is a serious problem for both credit card companies and consumers, as it can result in significant financial losses and damage to a person's credit score. To combat this problem, credit card companies use a variety of techniques to detect and prevent fraud. These techniques can include analyzing patterns of card usage to identify suspicious activity, using fraud detection software to flag potentially fraudulent transactions, and verifying the identity of cardholders through measures such as requiring a PIN or security code for certain transactions. In some cases, credit card companies may also work with law enforcement agencies to investigate and prosecute individuals who engage in credit card fraud.

B. Problem

Companies incur annual costs of billions of dollars due to credit card fraud. Although consumers are never responsible for unauthorized payments, it can be annoying and time-consuming to report thefts and replace credit cards. Around 127 million adult Americans, or close to half of the population, have experienced fraudulent charges on their credit or debit cards. More than one in three people who use credit or debit cards have been the victim of card fraud multiple times.

C. Importance

The importance of credit card fraud detection lies in the fact that it helps to protect both credit card companies and consumers from significant financial losses and damage to their credit scores. Effective fraud detection can help to maintain consumer trust in the credit card industry and promote the continued use of credit cards as a safe and secure form of payment. ML algorithms can be used to analyze large amounts of data from past credit card transactions to identify patterns and trends that may indicate fraudulent activity. This can be a highly effective way to detect fraud, as ML algorithms can learn to recognize patterns and make predictions that may not be obvious to human analysts. Additionally, ML algorithms can continue to learn and improve over time, making them more effective at detecting fraud as they process more data. This can help credit card companies to stay ahead of the ever-evolving methods that fraudsters use to carry out their crimes.

D. Existing Literature

Many real-time event-driven applications are inherently uncertain. Credit card fraud detection is a typical uncertain field, requiring real-time identification and tagging of suspected fraud instances before the transaction is approved or rejected. There have been various techniques to predict fraud detection. Outlier detection identifies transactions that are unusually large or different in kind from past transactions made by the user. However, these types of transactions may actually be carried out by the consumer, making the forecast inaccurate.

On the other hand, unsupervised outlier detection does forecast the necessary data, it only needs to comprehend the customer transactional behavior. a peer group. Another technique that has been employed is analysis, which compares things that have similar characteristics.

A breakpoint occurs when a data structure, such as an anomaly. Simply put, breakpoint analysis involves examining these in order to comprehend their existence and occurrence.

Even though fraud often employs supervised learning techniques detection, there is a chance that they will occasionally fail.

E. System Overview

Our system typically involves several key components and steps. The first step is to collect and process a large amount of data on past credit card transactions. This data may include information such as the transaction amount, the location of the transaction, the time of the transaction, and other relevant details. Then this data is preprocessed and made ready for applying various ML methods like Naive Bayes, XGBoost, Random Forest, etc. Data preprocessing may include steps like cleaning the dataset, removing noisy data or outliers, etc. Also, since the number of fraudulent transactions in the dataset chosen were very less compared to genuine transactions, scaling the data by introducing more fraudulent transactions and performing oversampling was necessary for getting better prediction results.

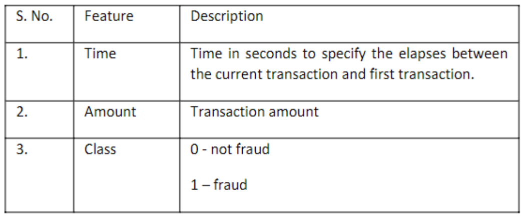
This data is then used to train ML algorithms and methods. The algorithm is trained using a supervised learning approach, in which it is provided with examples of both fraudulent and legitimate transactions, and it learns to identify patterns and characteristics that are common to each type of transaction. Once the ML algorithm has been trained, test data can be used to analyze other credit card transactions and make predictions about whether they are fraudulent or not.

To assess and improve the accuracy we compared the accuracies of used ML methods and applied an oversampling technique called 'SMOTE' on it to get better results. This is typically done by comparing the characteristics of the new transaction to the patterns and trends that the algorithm has learned from the training data. In practical usage, if the algorithm predicts that a transaction is fraudulent, it can flag the transaction for further review by a human analyst, who can then take appropriate action, such as blocking the transaction or contacting the cardholder to verify their identity.

F. Data Collection

We have 492 frauds out of 284,807 transactions in our dataset of transactions that took place over the course of two days. The dataset is very skewed, with frauds making up 0.173% of all transactions in the positive class.

It only has numeric input variables that have undergone PCA transformation. Unfortunately, we are unable to offer the original characteristics and additional context for the data due to confidentiality concerns. Functions V1, V2, ... The primary components obtained by PCA are V28; Time and Amount are the only features that were not altered.



S. No.	Feature	Description
1.	Time	Time in seconds to specify the elapses between the current transaction and first transaction.
2.	Amount	Transaction amount
3.	Class	0 - not fraud 1 – fraud

Fig 1 – Dataset variables

The confusion matrix depicted below explains that the attribute class is independent of the transaction's size and timing.

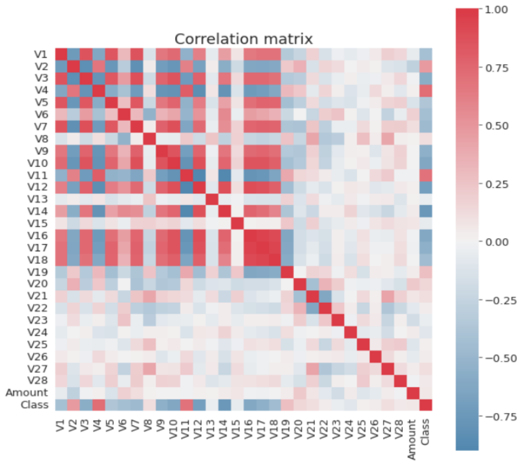


Fig 2 – Correlation Matrix for variables

G. Components of the ML System

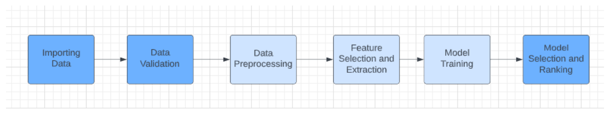


Fig 3 – Proposed project methodology

i. Data Validation

Verify the accuracy of the data. It determines if the data falls within the range of permitted values specified for the field by applying a set of criteria. Before training a new model version, make sure the source data is accurate and of high quality. Make certain that the inputs adhere to the established standards, such as the type, uniqueness, format, or consistency of the data.

ii. Data Preprocessing

- Removing null values
- Standardization
- Handling Categorical data
- Working on Data Imbalance
- Dropping Duplicates

iii. Feature Selection and Extraction

To create new features from the current ones in order to decrease the number of features in the dataset.

iv. Model Training

Giving an ML algorithm training data to use as a basis for learning is the process of training an ML model. Divided into training and testing data to perform prediction.

v. Model Selection and Ranking

Various ML models like Decision Tree, Regression Analysis, Random Forest Algorithms are used to predict the fraud. The highest accuracy and F1 scored is ranked.

II. Important Definitions

A. Data

The data has various columns from V1 to V28 that represent different attributes. They are mentioned this way to conceal sensitive information.

Only 0.17% of the transactions are fraudulent transactions in the above dataset.

The seconds that passed between each sale and the first sale in the dataset are contained in the Point' Time field. The sale quantum is the "point"; this point can be used in place of money. The answer variable, called Point Class, has two possible values for fraud and authenticity: 1 and 0, respectively.

B.Prediction Target

The goal is to predict if the sample is classified under fraudulent data or genuine from the transactional database. Numerous supervised learning algorithms have been applied to determine the target variable.

C.Variables

Time,Amount of transactions during that hour - Total number of transactions occurred within the timespan, Density of fraud Transaction time for genuine transactions

D. Problem Statement

Credit card fraud costs the world billions of dollars every year. Credit card use has considerably facilitated transactions for both users and retailers, but it has also given rise to numerous fraud incidents. Fraud done with a payment card, such as a credit or debit card, is referred to as "credit card fraud" in general. The goal could be to purchase products or services or to transfer money to another account under the offender's control. Machine Learning Algorithms can be used to classify credit card transaction data to predict whether a transaction is genuine or fraud.

III. Overview of proposed system

In the proposed system, We will use various machine learning algorithms, such as Decision Trees, Random Forest, and others, in our suggested system, calculate their accuracy scores, and then select the algorithm with the highest accuracy score. To determine the optimal algorithm, we will also compute the F1 score and precision recall for each algorithm and take those results into account in addition to the accuracy score. Later, We analyzed the accuracy of the various ML algorithms utilized and improved the accuracy .We also need to take into account how severely unbalanced the data collection we're about to examine is.

IV. Technical details of proposed system

A. Exploratory Data Analysis

We explored the dataset and obtained some amazing results.

1. Fraudulent transactions are 0.17% of the complete dataset. As expected, most of the samples are legitimate transactions. Only 0.17% of the transactions are flagged as fraudulent. It's clear now dataset is highly unbalanced with skewed features.

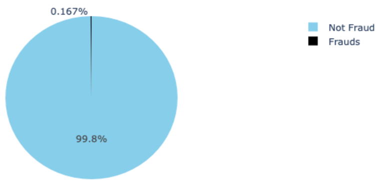


Fig 4 – Pie chart depicting the data distribution of Fraud and Genuine transactions

2. Below is the histogram plot for all variables. This helps us make sure nothing unusual stands out. The primary components are beyond our control, but it appears that the Amount and Time qualities need a closer look..

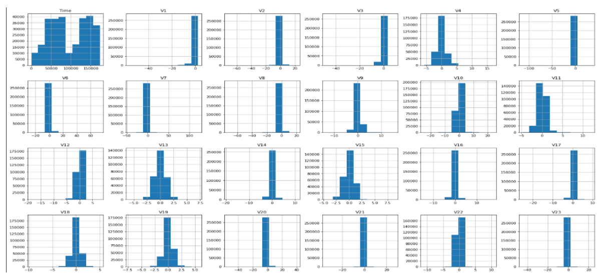


Fig 4 – Variables histogram

3. The fraudulent transactions (values of Class equal 1) have a skewed (asymmetric) distribution.

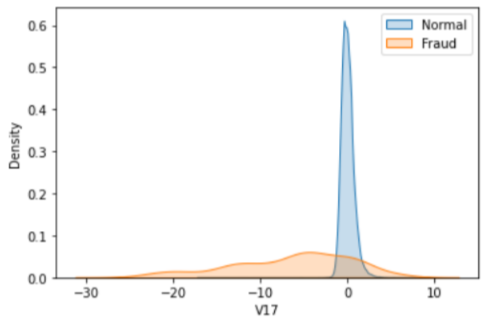


Fig 5 – Depicts the skewness in the dataset response variable

4. Maximum fraud transactions are being carried out during the early hours of the day and in the late night.

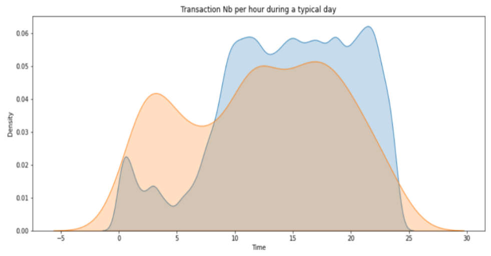


Fig 6 – Number of transactions made per hour in a day

5. Depicts correlation between every pair of variables. All the PCA variables are correlated and only non-pca variables, such as, 'Amount' and 'Time' have least correlation with the response.

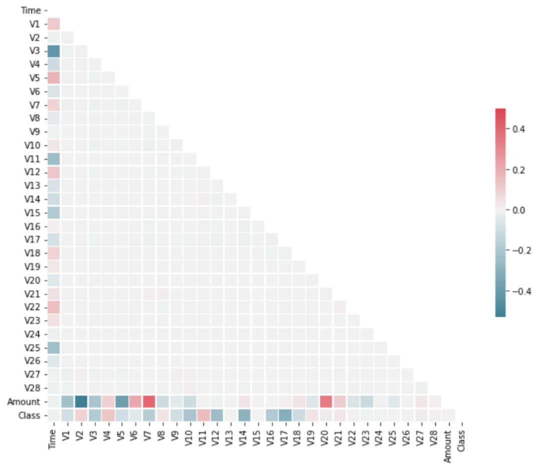


Fig 7– Correlation matrix for attributes

6. Below graphs depict how the number of transactions varies in time. Ups and downs show a day and night trend, respectively. Although most of them are under $50, it is also probably where most fraudulent transactions take place. According to the second figure below, most purchases are made during the day, and after individuals leave work or school and travel home, less purchasing is done until the following day.

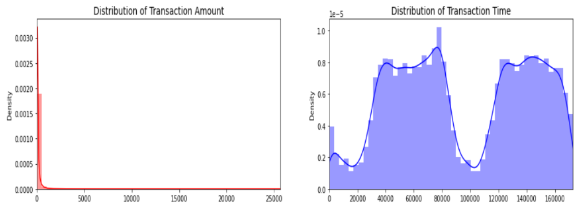


Fig 8 – Distribution of transaction amount and time

7. The number of genuine transactions rises during late night and early morning hours. The volume of legal transactions declines at night and picks up again once the working day begins. On the other hand, there appears to be an unexpected rise in fraudulent transactions at around 2am, which makes sense as most people sleep after this time. Additionally, it appears that the false transaction data is distributed more uniformly.

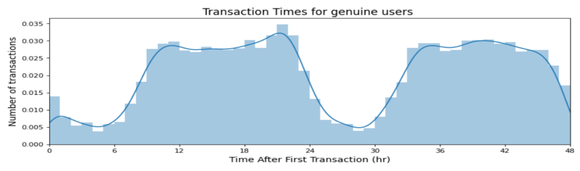


Fig 9 – Genuine transaction times

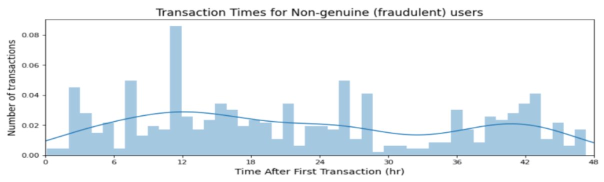


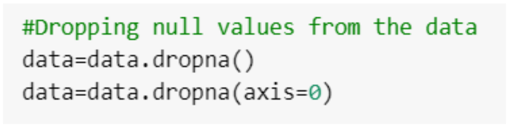
Fig 10 – Fraudulent transaction times

B. Data Preprocessing

The Preprocessing steps include -

1. Data Cleaning

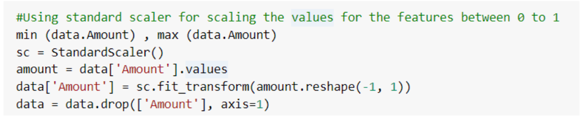
The goal of the data preprocessing stage is to reduce any possible model flaws as much as feasible. In general, the quality of a model depends on the data that is used to build it, so we perform data preprocessing to provide the model with the most accurate dataset possible. Even while we can't completely clean the dataset, we can at least take some simple actions to make sure that our dataset has the best chance of producing a solid model. Let's first look at the null values for this dataset. Our dataset's null values are empty, pointless entries that we don't require. Our model will be wrong if we don't eliminate null values since we'll build connections for pointless data rather than concentrating all our energy on a single goal.



```
#Dropping null values from the data
data=data.dropna()
data=data.dropna(axis=0)
```

Fig 11 – Check and drop Null values

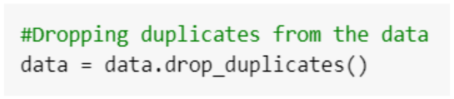
We observed in our data 28 features are transformed versions of PCA, but the Amount is the original one. Checking the minimum and maximum is in the amount, the difference in Amount column is huge that can deviate our result. In this case, it is good practice to scale this predictor. A standard scaler can fix this.



```
#Using standard scaler for scaling the values for the features between 0 to 1
min (data.Amount) , max (data.Amount)
sc = StandardScaler()
amount = data['Amount'].values
data['Amount'] = sc.fit_transform(amount.reshape(-1, 1))
data = data.drop(['Amount'], axis=1)
```

Fig 12 – Code for standard scaling of 'Amount' variable

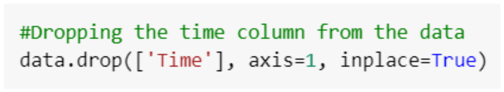
Secondly, Duplicate data was removed as they take up unnecessary storage space and slows down calculations to a minimum. At worst, these can skew analysis results.



```
#Dropping duplicates from the data
data = data.drop_duplicates()
```

Fig 13 – Dropping of duplicate values from the dataset

To effectively predict the response with good accuracy, we need to eliminate some unnecessary columns. Here, we have a variable, ‘Time’ which can be an external deciding factor but in our modelling process, we can drop it.



```
#Dropping the time column from the data
data.drop(['Time'], axis=1, inplace=True)
```

Fig 14 – Dropping of ‘Time’ column

2. Data Transformation

i.Normalization

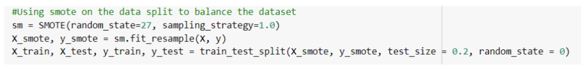
The column "Amount" was subjected to a process known as max-min normalization, which rescales features having a distribution value between 0 and 1. As a result, for this column, the minimum value of that feature becomes 0 and the maximum value becomes 1. It is typically applied to equalize the magnitude of all values.

$$\mathbf{Xnorm = x - min(x) / max(x) - min(x)}$$

(1) - Xnorm equation

ii.Sampling

An issue with our imbalanced classification is that the Fraud class contains too few samples for our model to correctly learn the decision boundary. To generate more data from the Fraud classes and balance the dataset, methods including oversampling and Synthetic Minority Oversampling Technique, or SMOTE, were used. The SMOTE technique is successful because acceptable new synthetic samples from the minority class are produced, which are quite near in feature space to the minority class's existing examples.



```
#Using smote on the data split to balance the dataset
sm = SMOTE(random_state=27, sampling_strategy=1.0)
X_smote, y_smote = sm.fit_resample(X, y)
X_train, X_test, y_train, y_test = train_test_split(X_smote, y_smote, test_size = 0.2, random_state = 0)
```

Fig 15 – SMOTE implementation

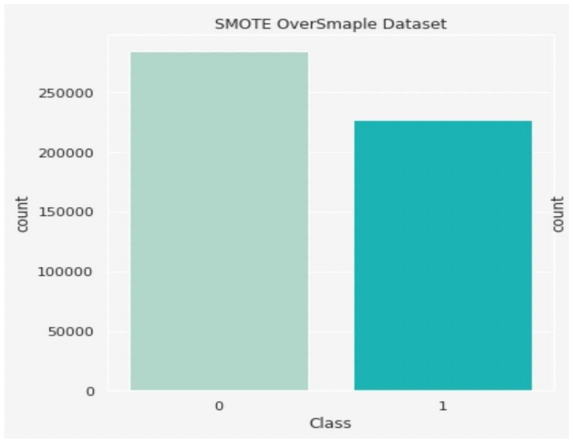


Fig 16 – Reduction in data imbalance post SMOTE

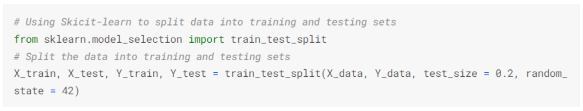
3. Feature Selection and Extraction

One advantage of gradient boosted trees is the relevance of the features. Retrieving significant scores for each attribute is a simple task. In general, significance assigns a score based on the effectiveness or value of each feature in building the boosted decision trees within the model. The more attribute's relative relevance increases the more decision trees use it to make important decisions.

Each property in the dataset has its significance estimated explicitly, enabling attributes to be ranked and contrasted with one another. Predicted target values against several criteria will be used to assess the feature relevance of the submitted data.

4. Predictive Modeling

Over 4 machine learning models were trained and tested to note their performances on the test data. We have the predictors and target, and in our case, there are no categorical features. The data was split into the train and test data as below.

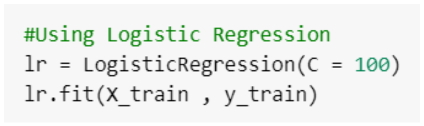


```
# Using Skicit-learn to split data into training and testing sets
from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X_data, Y_data, test_size = 0.2, random_
state = 42)
```

Fig 17 – Code to split and obtain train and test data

(i) Logistic Regression

Logistic regression models are used to predict dichotomous outcomes (e.g., success/failure), or in our case normal/fraudulent credit card transactions.

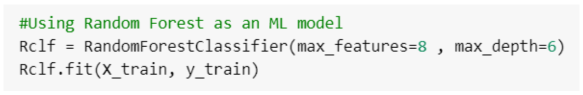


```
#Using Logistic Regression
lr = LogisticRegression(C = 100)
lr.fit(X_train , y_train)
```

Fig 18 – Logistic regression model implementation

(ii) Random Forest Classifier

Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithms of the same type i.e., multiple decision trees, resulting in a forest of trees, hence the name "Random Forest". The random forest algorithm can be used for both regression and classification tasks.



```
#Using Random Forest as an ML model
Rclf = RandomForestClassifier(max_features=8 , max_depth=6)
Rclf.fit(X_train, y_train)
```

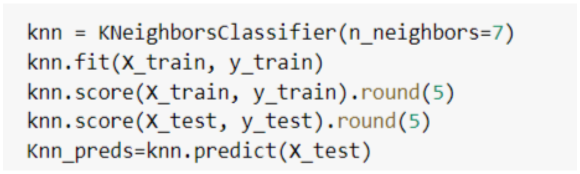
Fig 19 – Random Forest model implementation

(iii) Naïve Bayes Classifier

Naive Bayes is a Supervised Machine Learning algorithm based on the Bayes Theorem that is used to solve classification problems by following a probabilistic approach.

(iii) k-Neighbours Classifier

K Nearest Neighbor (KNN) is a Supervised Machine Learning algorithm that classifies a new data point into the target class, depending on the features of its neighboring data points.

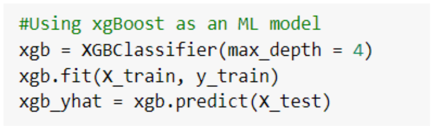


```
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train, y_train)
knn.score(X_train, y_train).round(5)
knn.score(X_test, y_test).round(5)
Knn_preds=knn.predict(X_test)
```

Fig 20 – kNN model implementation

(iv) XGBoost Classifier

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.



```
#Using xgBoost as an ML model
xgb = XGBClassifier(max_depth = 4)
xgb.fit(X_train, y_train)
xgb_yhat = xgb.predict(X_test)
```

Fig 21 – XGBoost model implementation

D. Model Evaluation Metrics

(i) Accuracy

Accuracy is the percentage of data points that are accurately anticipated. It is among the most basic types of evaluation measures. The accuracy score is calculated as the number of correct points divided by the total number of points.

(ii) Recall

Recall is a binary classifier metric that assesses the percentage of positive labels that are correctly predicted out of all positive labels. Formally, it is the ratio of the number of actual positive answers to the total of false negative responses that were incorrectly categorized as positive (false negative).

(iii) Precision

Precision is a binary classifier statistic that assesses the consistency of all positive labels. A binary classifier only produces two values (i.e., positive and negative). In order to solve an issue with multiple values (in our case, three), we must first convert it into a binary classification problem.

$$\text{Precision} = \frac{tp}{tp + fp}$$
$$\text{Recall} = \frac{tp}{tp + fn}$$

(2) – Precision-Recall calculation

(iv) F1-Score

The harmonic mean of a classifier's precision and recall is used to create the F1-score, which integrates both metrics into a single number. In order to compare the effectiveness of two classifiers, this method is usually employed.

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

(3) – F1 score equation

(v) ROC Curve

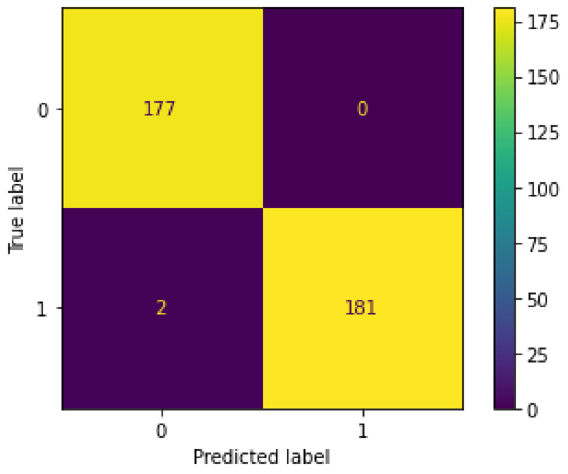
The receiver operating characteristic curve (ROC curve) is a graph that displays how well a classification model performs across all categorization levels. This curve plots two parameters, which are True Positive rate (TPR) and False Positive rate (FPR). TPR is the same as Recall and FPR is defined as below :

$$FPR = \frac{FP}{FP + TN}$$

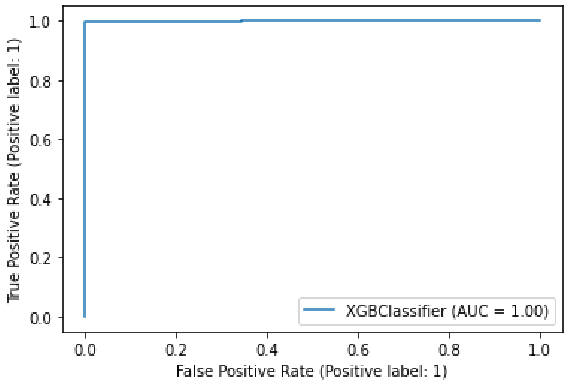
(4) – FPR equation

V. Experiment Results

We tested a number of different baseline classification methods, including logistic regression, Naive Bayes, K-Nearest Neighbors, Random Forests, and XG-Boost Classifier. The Random Forest Classifier and XG-Boost Classifier provided the most promising results, but the dataset had a data imbalance, with only 0.173% of total transactions being fraudulent and the remaining 99.8% being genuine transactions. Because of the data mismatch, greater weight would be attributed to real transactions, resulting in more false positives As a result, we used different data balancing techniques to handle the data imbalance and used oversampling methods like Random Oversampling and SMOTE, which gave us better results than the initial baseline methods experiment, but the results of both data balancing techniques, namely Random OverSampling and SMOTE, were quite similar. SMOTE, as well as for random forest and XGBoost, were chosen as our final model, using SMOTE as our data balancing strategy. The confusion matrix for the XGBoost model after employing SMOTE for data balancing is shown below.



And the following is the AUC-ROC graph for the XGBoost model with an AUC score of 1.00



The results for the evaluation metrics represented as follows:

Accuracy Score: 0.994444
Precision Score: 1.0
Recall Score: 0.989071
F1-score: 0.994505

As demonstrated by the evaluation metrics, ROC graph, and confusion matrix. The XGBoost model is an excellent model that outperforms the other algorithm approaches. As a result, we may conclude that the XGBoost excels at finding anomalies in a heavily skewed dataset.

VI. Related work

New methods for credit card fraud are developed researching various fraud detection techniques.
"Implementation of credit card fraud detection" A work on decision trees, random forests, SVM, and logistic regression. They used a dataset that was heavily skewed and worked with it. A performance review is done based on precision, sensitivity, accuracy, and specificity.
"A novel approach for credit card fraud detection" uses the combination of Hidden Markov Model, Behavior Based Technique, and Genetic Algorithm is developed to stop these fraudulent transactions. Every transaction is

examined using the aforementioned method, and a fraud detection system examines each transaction to look for signs of fraud. The objective is to identify fraud as accurately and minimally as possible.

"Application of Credit Card Fraud Identification Using Bagging and Boosting Methods" Ensemble learning techniques are recognized as a frequent and popular approach, not only for their remarkably accurate prediction on real-world issues but also for their relatively simple implementation.

VII. Conclusions

In this paper, we developed a classification model to explicitly categorize fraud transactions from genuine transactions. We visualized various aspects of the data, understood the correlation between all the features, performed feature extraction, adjusted data unbalancing and performed predictive modeling using 5 algorithms. From the experiments the result that has been concluded is that Random Forest Classifier has an accuracy of 95% while kNN shows accuracy of 91% , and support vector machine shows an accuracy of 89% but the best results are obtained by XGBoost using SMOTE for dealing with the data imbalance and received a precise accuracy of 99.6%. The results obtained thus conclude that XGBoost Classifier shows the most precise and high accuracy of 99.96% in the problem of credit card fraud detection with a dataset provided by ULB machine learning. Since the entire dataset consists of only two days' transaction records, it's only a fraction of data that can be made available if this project were to be used on a commercial scale. Being based on machine learning algorithms, the program will only increase its efficiency over time as more data is put into it.

The link to code is given below:
https://colab.research.google.com/drive/1-PYV_szTEvcVBLH9JCko1D3D2J6nyDrn#scrollTo=epc5VpGJra0X&uniqifier=1

VIII. Acknowledgment

We thank Dr. Yanjie Fu (University of Central Florida) for giving us the opportunity to build this real-time project as part of the course Machine Learning (CAP5610). Thanks to the Machine Learning Group at ULB once again for providing the dataset. It was vitally important to receive support and direction throughout the coursework and research assignment.

IX. References

[1 https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
[2]]https://www.security.org/digital-safety/credit-card-fraud-report/
[3]https://www.inscribe.ai/fraud-detection/credit-fraud-detection
[4]https://www.ijitee.org/wp-content/uploads/papers/v10i6/C84000110321.pdf
[5]https://fraud-detection-handbook.github.io/fraud-detection-handbook/Chapter_2_Background/MachineLearningForFraudDetection.html
[6]https://www.projectpro.io/article/credit-card-fraud-detection-project-with-source-code-in-python/568
[7]https://www.questjournals.org/jrhss/papers/vol8-issue2/B08020411.pdf