MID-TERM PROJECT

CHAITANYA SAI GANDI

NUID: 002430495

COLLEGE OF ENGINEERING

NORTHEASTERN UNIVERSITY

TORONTO, ON

gandi.ch@northeastern.edu

Abstract

This report examines the use of four machine learning algorithms—Logistic Regression, Artificial Neural Network (ANN), Random Forest, and XGBoost—on the Credit Card Fraud Detection dataset from Kaggle. The data in this dataset has been anonymized to protect the cardholders' identities. The main goal is to accurately identify fraudulent transactions in a large collection of credit card data. The dataset was prepared by fixing missing values and normalizing features to make it suitable for training the models. Each algorithm was applied to classify transactions, and their effectiveness was measured using accuracy and other evaluation metrics. The results highlight the effectiveness of each model in detecting fraudulent Transactions

Datasets

	Data Set Characteristics	Attribute Characteristics	Associated Tasks	Number of Instances	Number of Attributes
Dataset	Credit Card Fraud Detection	Real	Classification	284807	31

Why I find Datasets Interesting

The **Credit Card Fraud Detection** dataset is interesting because it focuses on identifying fraudulent transactions, which is a major issue in the financial industry. By analysing this dataset, patterns of suspicious activity can be discovered, leading to the development of models that detect fraud early. This helps protect consumers and financial institutions from significant losses, making the dataset highly valuable for improving security and trust in financial systems.

Data Characteristics

For the Credit Card Fraud Detection Dataset, there are **284,807 records** and **31 features**. All columns are numerical, with no categorical data present. The dataset includes a **'Class'** column, which indicates whether a transaction is fraudulent (1) or not (0). Additionally, the dataset contains no missing or null values, and there are no outliers, as the features consist of principal components resulting from a PCA transformation, except for the 'Time,' 'Amount,' and 'Class' columns. Preprocessing steps are minimal, mainly involving scaling the 'Time' and 'Amount' columns.

Data Preparation and Preprocessing for Credit Card Fraud Detection Dataset

The Credit Card Fraud Detection dataset, consisting of 284,807 records and 31 features, was loaded for analysis. The features are a mix of numerical data, where the main numerical features include anonymized variables (V1 to V28), along with "Time," "Amount," and the target variable "Class," which indicates whether a transaction is fraudulent or not.

Initially, the structure of the dataset was checked to verify the number of records and features, as well as their data types. All columns were confirmed to be numerical, with "Class" acting as the binary target variable (0 for non-fraudulent transactions and 1 for fraudulent transactions).

A statistical summary of the numerical features was generated to gain insights into the dataset. The "Time" variable indicates the time elapsed from the first transaction, and "Amount" represents the transaction amount. The summary revealed key statistics, such as the mean, standard deviation,

minimum, and maximum values. For instance, the average transaction amount is approximately 88.35, with a standard deviation of 250.12, highlighting considerable variability. The target variable "Class" is highly imbalanced, with only around 0.17% of the transactions marked as fraudulent.

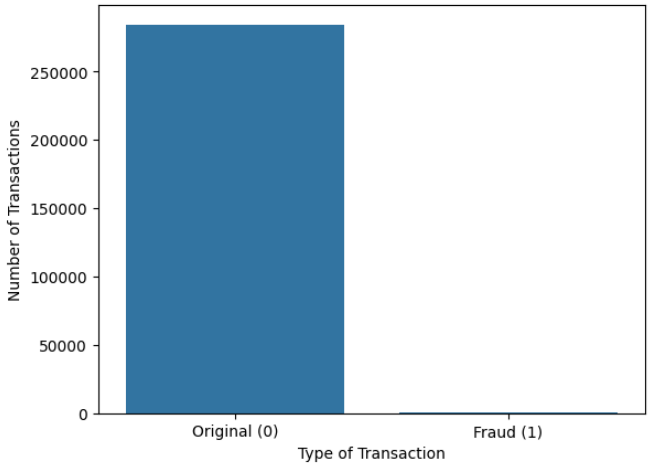
Further data quality checks were conducted, confirming that the dataset contains no missing values, as observed through the results of the data.info() function.

Since all features are numerical, label encoding was not necessary. However, the imbalance in the dataset may require special handling, such as resampling techniques, before moving forward with model training to ensure effective classification of fraudulent transactions.

Data Visualization of Credit Card Fraud Data

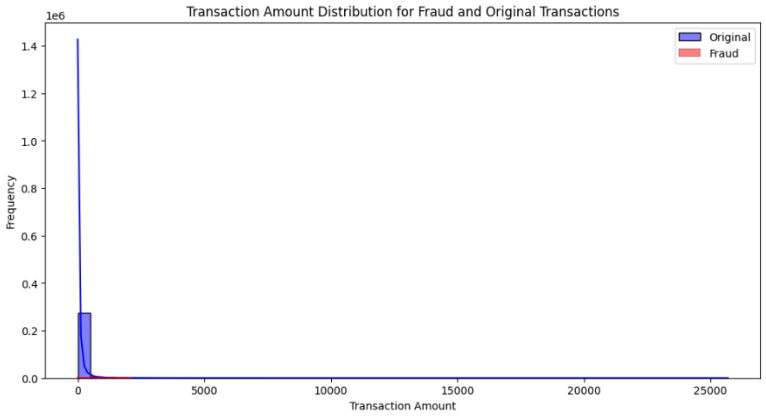
Distribution of Normal vs Fraudulent Banking Transactions

The graph shows a highly imbalanced distribution where normal transactions (approximately 275,000) vastly outnumber fraudulent transactions, indicating that fraud cases are relatively rare events in this banking transaction dataset.



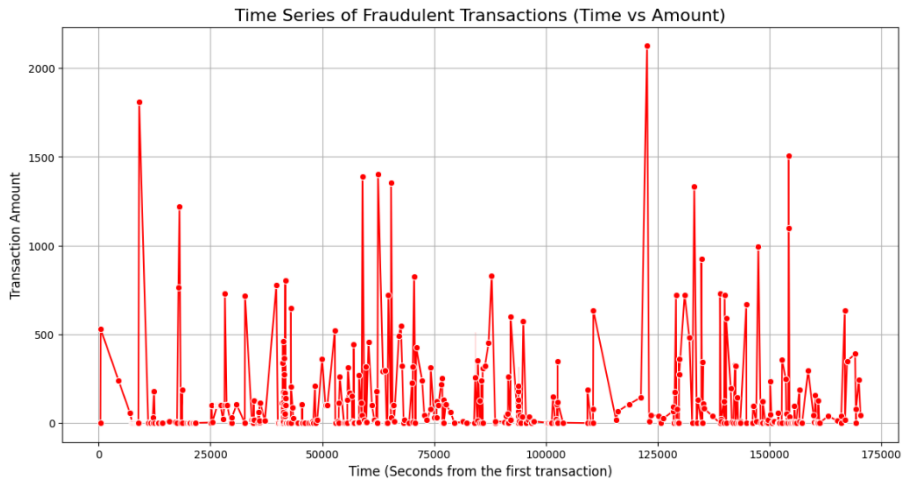
Distribution Analysis of Transaction Amounts: Comparing Fraud vs Legitimate Cases

The visualization reveals that most legitimate transactions (shown in blue) are concentrated at lower amount values with a very high frequency peak near zero, while fraudulent transactions (in red) appear to have a different distribution pattern, though they're harder to see due to their much lower frequency compared to legitimate transactions.



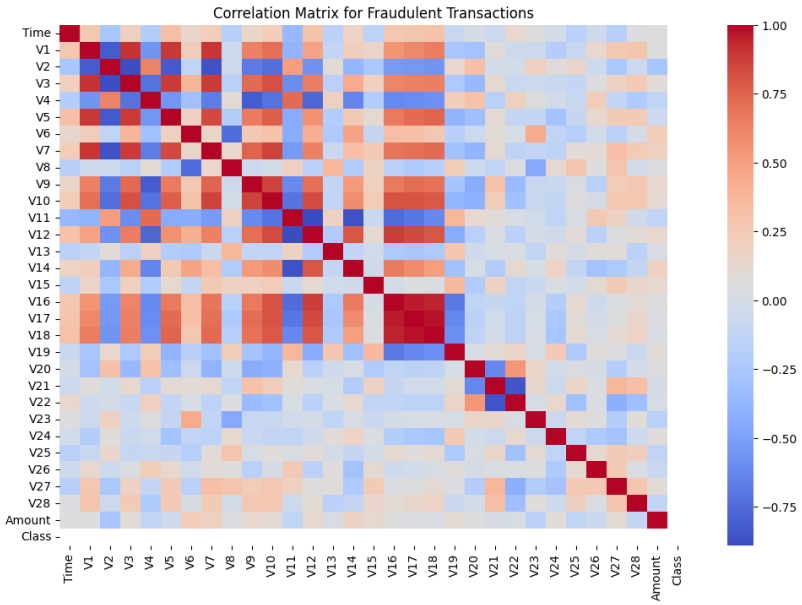
Temporal Analysis of Fraudulent Transaction Amounts Over Time

The time series plot reveals irregular spikes in fraudulent transaction amounts throughout the observed period, with particularly notable peaks reaching around 2000 units, suggesting that fraudsters occasionally attempt large-value transactions while mostly keeping to smaller amounts to avoid detection



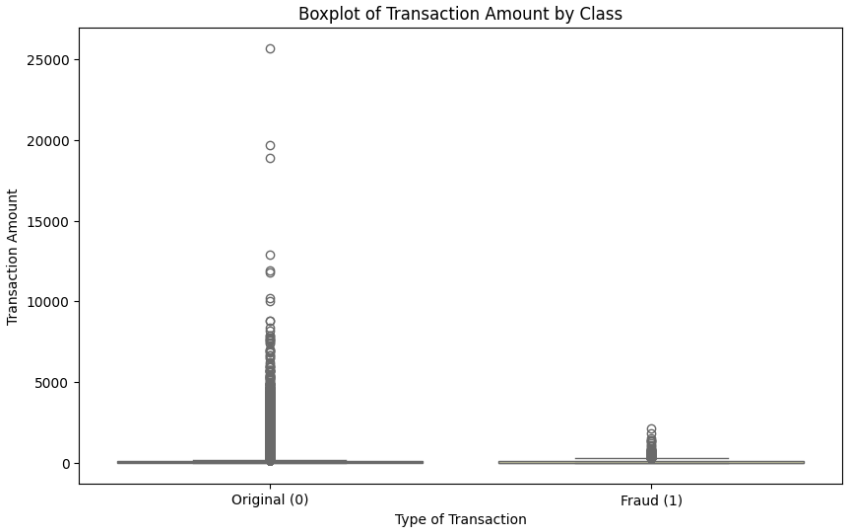
Feature Correlation Analysis in Fraudulent Transaction Dataset

The correlation heatmap reveals strong correlations (shown in dark red and blue colors) between certain V1-V28 features especially in the upper left quadrant, while the Amount and Class variables show relatively weaker correlations with other features, suggesting complex patterns in how fraudulent transactions are characterized across multiple variables.



Distribution of Transaction Amounts: Legitimate vs Fraudulent Cases - A Box Plot Analysis

The boxplot demonstrates that while both legitimate and fraudulent transactions have similar median amounts (shown by the middle line in each box), legitimate transactions show more extreme outliers reaching up to 25,000 units, compared to fraudulent transactions which have fewer and lower-value outliers.



Data Model Training and Evaluation

Feature Selection, Normalization, and Data Splitting for Model Training

In this phase, relevant features were selected from the dataset for analysis. The variable X was created by removing the target variable "Class" from the dataset, while y was defined as the "Class" column, which represents whether a transaction is fraudulent (1) or not (0).

Next, data normalization was applied using StandardScaler to scale the "Amount" and "Time" columns, ensuring that these features have a mean of 0 and a standard deviation of 1. Normalization helps balance the model and improves its performance by preventing certain features from dominating due to larger scales.

Finally, the dataset was split into training and testing sets, with 80% of the data allocated for training the model and 20% for testing. The train_test_split function was used, with stratification to ensure the class distribution (fraud vs. non-fraud) remains consistent across the training and testing sets. This step is important for properly evaluating the model's effectiveness in detecting fraudulent transactions.

Logistic Regression Model Initialization and Training

In this section, a logistic regression model was initialized using the LogisticRegression class. This algorithm is commonly used for binary classification tasks, making it suitable for detecting fraudulent transactions in this dataset.

The model was trained by fitting it to the training data (X_train and y_train). This process involves adjusting the model's internal parameters to learn the relationship between the features and the target

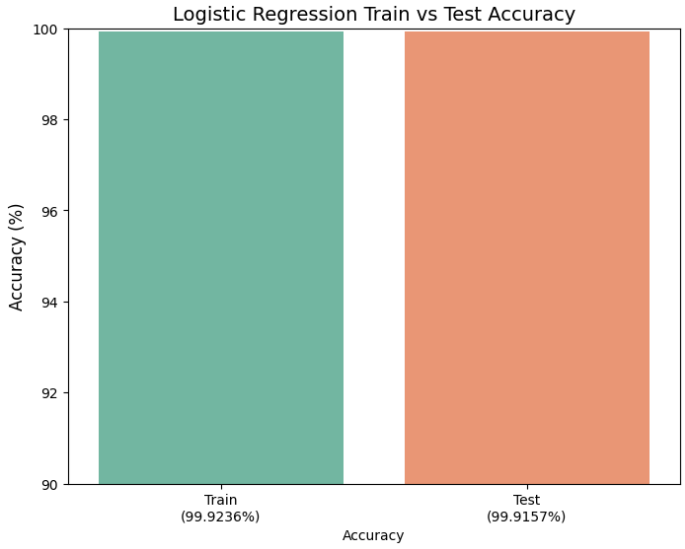
variable. By fitting the model to the training data, it learns how to differentiate between fraudulent and non-fraudulent transactions, enabling it to make predictions on unseen data later in the analysis.

Logistic Regression Model Predictions and Accuracy Evaluation

In this section, predictions were made using the trained logistic regression model on both the training and testing datasets. The predict method was employed to generate predictions for X_test and X_train.

Following this, the accuracy of the model was assessed for both datasets using the accuracy_score function. This function compares the predicted values to the actual values in y_train and y_test, determining how many predictions were correct.

The accuracy results were compiled into a DataFrame, which presents the accuracy percentages for both the training and testing phases. The training accuracy was found to be approximately 99.92%, while the testing accuracy was about 99.92% as well. These high accuracy rates indicate that the logistic regression model performed exceptionally well in distinguishing between fraudulent and non-fraudulent transactions on both datasets.



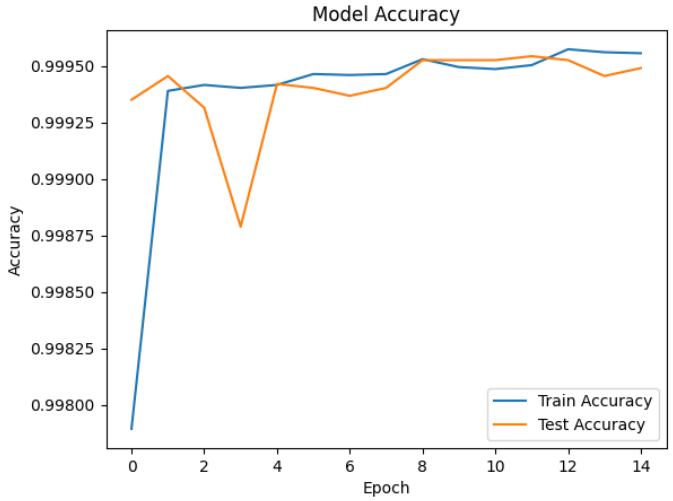
Artificial Neural Network (ANN) Model Initialization and Training

The artificial neural network (ANN) model is set up using Keras in a sequential manner, meaning layers are added one after the other. It starts with an input layer that has 128 units and uses a ReLU activation function, with its input dimension matching the number of features in the training data. Next, two hidden layers are added, one with 64 units and another with 32 units, both also using the ReLU activation function to help the model learn complex patterns in the data. The final output layer has a single unit with a sigmoid activation function, which is suitable for deciding if a transaction is fraudulent or not.

The model is compiled with a binary cross-entropy loss function, appropriate for binary classification tasks, and uses the Adam optimizer with a learning rate of 0.001. The performance is measured using accuracy. The model is then trained on the training dataset for 15 epochs with a batch size of 64, and validation data from the testing set is used to check how well the model is doing on unseen data. This setup aims to create an effective way to classify transactions as either fraudulent or non-fraudulent based on the selected features.

Performance Evaluation of the Artificial Neural Network Model

The performance of the artificial neural network (ANN) model was evaluated using both the test and training datasets. For the test dataset, the model achieved an accuracy of 99.95%, indicating that it correctly classified nearly all transactions in the testing set. Similarly, when evaluated on the training dataset, the model reached an accuracy of 99.96%, showing that it performed exceptionally well on the data it was trained on. These high accuracy rates suggest that the ANN model is effective in distinguishing between fraudulent and non-fraudulent transactions based on the selected features.



Implementation of the Random Forest Classifier

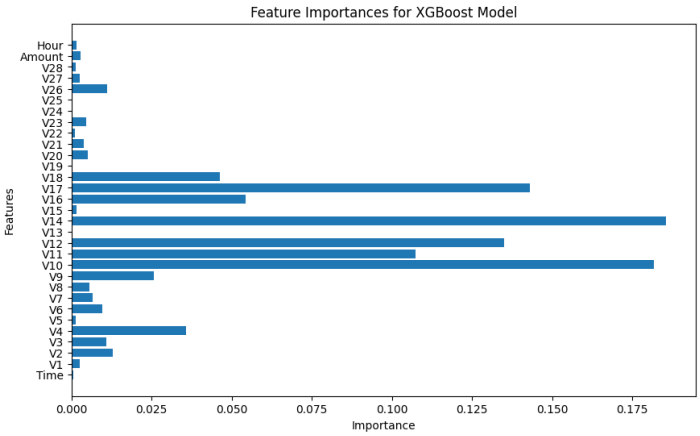
The Random Forest Classifier was initialized with 30 decision trees, a maximum depth of 3 for each tree, and a specified random state to ensure reproducibility. The model was then trained using the training dataset to learn the patterns associated with the target variable. The out-of-bag (OOB) score was enabled to provide an unbiased estimate of the model's accuracy during training without needing a separate validation dataset.

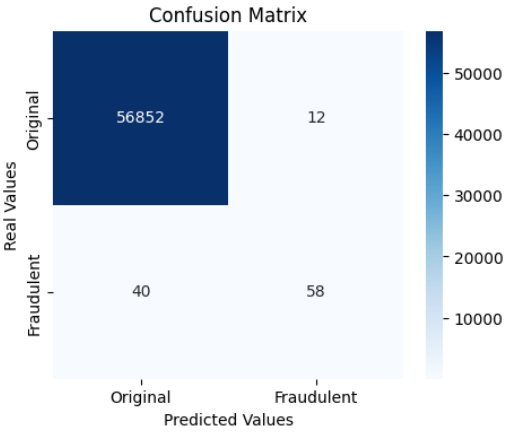
Evaluation of the Random Forest Classifier and Feature Importance Analysis

The out-of-bag score of the Random Forest model was calculated to be approximately 99.92%, indicating a strong performance during training without the need for cross-validation.

Feature importance scores were extracted to identify which features contribute most to the model's predictions. Each feature's name was paired with its importance value, revealing that features like "V14" (importance of 0.1855) and "V10" (importance of 0.1818) were among the most significant predictors. In contrast, features like "Time" and "Hour" showed minimal impact on the model's decisions.

Finally, predictions were made on the test dataset using the trained Random Forest model, and a confusion matrix was generated to assess the classification performance on the test set. This matrix provides insights into the number of true positive, true negative, false positive, and false negative predictions, allowing for a comprehensive evaluation of the model's effectiveness.





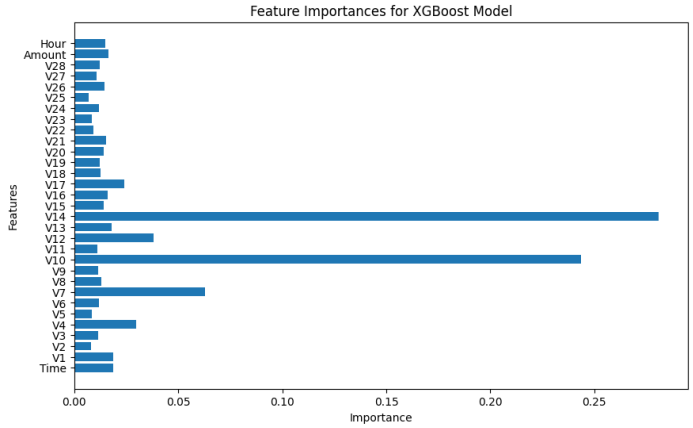
Implementation of the XGBoost Classifier

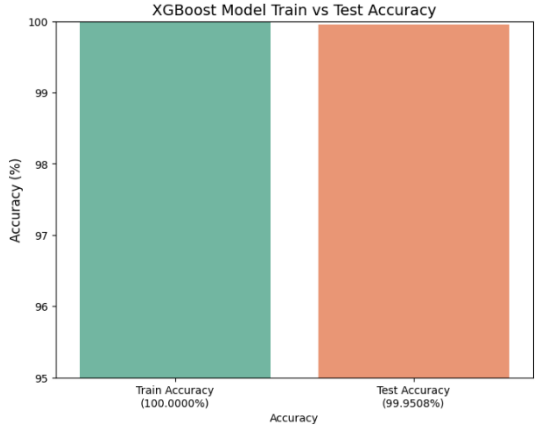
The XGBoost model was initialized using the XGBClassifier with a fixed random state of 42 to ensure reproducibility. Afterward, the model was trained on the training dataset, consisting of the features in X_train and the target variable in y_train. This process allows the model to learn the underlying patterns in the data, which can be leveraged for accurate predictions in subsequent evaluations. XGBoost is known for its efficiency and performance, especially in handling large datasets and dealing with class imbalances, making it a suitable choice for the credit card fraud detection task.

Evaluation of the XGBoost Model

The performance of the XGBoost model was evaluated by predicting the target variable for both the training and testing datasets. The training accuracy achieved was 100%, indicating that the model perfectly classified all training instances. The test accuracy was slightly lower at approximately 99.95%, demonstrating the model's strong ability to generalize to unseen data.

To further understand the model's decision-making process, the importance of each feature was analysed. The feature importances were calculated, revealing that certain features significantly influenced the model's predictions. For instance, "V14" and "V10" had the highest importance scores, indicating their crucial role in detecting fraudulent transactions. Other features, such as "Time" and "Amount," also contributed to the model's predictions but to a lesser extent. This feature importance analysis provides valuable insights into which factors are most relevant for identifying fraud in credit card transactions.





Performance Comparison of Machine Learning Algorithms for Credit Card Fraud Detection

By analyzing the Credit Card Fraud Detection dataset, four algorithms were evaluated: Logistic Regression, Artificial Neural Network (ANN), Random Forest, and XGBoost. Logistic Regression achieved an accuracy of approximately 99.92% on both training and testing datasets, indicating strong performance in distinguishing between fraudulent and non-fraudulent transactions. The ANN slightly outperformed Logistic Regression, achieving 99.95% accuracy on the test set and 99.96% on the training set, demonstrating its effectiveness in capturing complex patterns in the data. The Random Forest Classifier also performed well, with an out-of-bag score of about 99.92% and identified important features like 'V14' and 'V10.' However, XGBoost emerged as the top performer, achieving a perfect training accuracy of 100% and a high-test accuracy of 99.95%, showcasing its efficiency in handling large datasets and class imbalances while delivering accurate predictions. Overall, while all algorithms performed well, XGBoost stood out as the most effective model for this dataset.

Conclusion

In summary, the performance of all four algorithms on the Credit Card Fraud Detection dataset reveals that all models demonstrated high accuracy, successfully identifying fraudulent transactions. However, XGBoost excelled above the others with its perfect training accuracy and strong generalization to the test set, making it the most reliable choice for this task. The ANN also performed impressively, showcasing the ability to learn complex patterns, while Logistic Regression and Random Forest provided solid results as well. This analysis highlights the importance of selecting the right algorithm based on the dataset characteristics, with XGBoost proving to be the most effective for fraud detection in this case.

References

https://openai.com/chatgpt/

https://claude.ai/

https://www.tensorflow.org

https://scikit-learn.org

https://keras.io

https://matplotlib.org

https://seaborn.pydata.org

https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud