# Fraud Detection in Financial Transactions

|  |  |  |
| --- | --- | --- |
| **SR NO.** | **TABLE CONTENT** | **PAGE NO.** |
|  | Introduction   * 1. Background   2. Motivation   3. Goals   4. Expected Outcomes | 03  03  03  04  04 |
|  | Methodology   * 1. Load and Prepossessing Data   2. Classification Models   3. Flow Chart | 05  05  08  14 |
|  | Description of the Dataset  3.1 Data Sources | 15  16 |
|  | Result and Analysis | 17 |
|  | Conclusion | 21 |
|  | References | 22 |

**List of Figures**

Figure 1: [Code Snippet] Loading the model

Figure 2: Count of fraud transactions based on transaction category

Figure 3: Distribution of transaction amount by transaction type

Figure 4: [Code Snippet] New features creation

Figure 5: Decision Tree Overview

Figure 6: Confusion Matrix of Decision Tree

Figure 7: Confusion Matrix of Support Vector Classifier

Figure 8: ROC of Logistic Regression

Figure 9: K-Nearest Neighbor Cross Validation Scores

Figure 10: XGBoost Algorithm

Figure 11: Flowchart

Figure 12: Description of the Dataset

Figure 13: Raw Data

Figure 14: K-Nearest Neighbors Analysis and Accuracy

Figure 15: Logistic Regression Analysis and Accuracy

Figure 16: Decision Tree Analysis and Accuracy before Optimization

Figure 17: Decision Tree Analysis and Accuracy after Optimization

Figure 18: XGBoost Analysis and Accuracy

Figure 19: Support Vector Classifier Analysis and Accuracy with RBF Kernel

Figure 20: Support Vector Classifier ROC and Confusion Matrix

1. **Introduction:**
   1. **Background**

Financial fraud is a significant global issue that incurs billions of dollars in losses annually. The surge of electronic transactions has made it challenging for conventional fraud detection techniques to keep up. Consequently, financial institutions are implementing advanced technologies such as machine learning algorithms and data analytics to detect and prevent fraudulent activities. These technologies can scrutinize massive amounts of data and spot unusual patterns and anomalies that may indicate fraudulent behavior. Machine learning algorithms learn from previous data to recognize potential fraudulent transactions in real-time. Similarly, data analytics can pinpoint suspicious behaviors like sudden transaction volume spikes or unusual transaction locations. In summary, by investing in advanced technologies, financial institutions can tackle fraud and safeguard customers' assets and sensitive financial information.

* 1. **Motivation**

The project's motivation is to create a robust machine learning model that can analyze financial transaction data to detect instances of fraud accurately and efficiently. One of the primary objectives of developing this model is to detect fraudulent activities in real-time. By doing so, financial losses can be minimized, and customers can be alerted immediately about any suspicious activities on their accounts.

The project aims to design a model that can handle large volumes of data and identify unusual patterns and anomalies that may indicate fraudulent behavior. With machine learning algorithms, the model can learn from previous data to recognize and predict potentially fraudulent transactions. This allows financial institutions to act promptly and prevent fraudulent activities before they cause significant losses. Additionally, by analyzing transaction data, the model can detect unusual transaction patterns, such as sudden increases in transaction volume or a spike in transactions from a specific location, which are indicative of fraudulent activities.

The successful development of an accurate fraud detection model can significantly reduce financial losses for companies, which would otherwise incur losses from fraudulent activities. Companies can also save on overhead costs associated with dealing with fraudulent activities. Moreover, detecting fraudulent activities in real-time helps maintain customer trust in digital financial services. Customers feel more secure using these services, knowing that their financial information is protected from fraudulent activities.

The project's outcomes can have a significant impact on the financial services industry, especially in developing countries where fraudulent activities are rampant. By implementing accurate fraud detection models, financial institutions can build trust in digital financial services, attract more customers, and promote financial inclusion. The model can also help identify and prevent fraudulent activities related to money laundering and terrorism financing, which have severe implications for national security.

* 1. **Goals**

The primary objective of this project is to develop an accurate machine learning model capable of detecting fraudulent transactions. To achieve this, the project has identified several goals:

1. Conduct exploratory data analysis and preprocess the dataset to identify missing data, outliers, and inconsistencies that may affect model performance.
2. Train and evaluate various machine learning algorithms to identify the most effective models for detecting fraudulent transactions.
3. Conduct a comparative analysis of different machine learning models' performance and select the best model for further analysis.
4. Identify the most significant features contributing to fraud detection and integrate them into the model to improve its accuracy.
5. Provide recommendations and insights to financial institutions on how to improve their fraud detection techniques using the results of the project.
   1. **Expected Outcomes**
6. Analysis of the distribution of the features and the target variable to understand the characteristics of the data
7. Selection of appropriate machine learning algorithms for the classification task, such as logistic regression, decision trees, random forests, and support vector machines
8. Evaluation of the performance of the models using various metrics such as accuracy, precision, recall, F1 score, and ROC-AUC curve
9. Interpretation of the feature importance to understand the factors contributing to fraudulent transactions
10. Discussion of the limitations and challenges encountered during the analysis, such as class imbalance, missing data, and model overfitting
11. Recommendations for future research and improvements to the model, such as incorporating additional features or applying advanced techniques like deep learning
12. **Methodology**

The methodology of this project involves various phases, including understanding the dataset, exploratory data analysis, and modeling.

The first phase involves data acquisition, cleaning, preprocessing, and handling missing values, outlier removal, feature scaling, and engineering.

In the exploratory data analysis phase, a comprehensive report will be produced, detailing the analysis conducted and critical findings obtained. This will include relevant hypotheses concerning the differentiation between fraudulent and non-fraudulent transactions, with visualizations and charts to demonstrate the differences.

The modeling phase will provide a detailed report on the outcomes of various techniques utilized, including iterations attempted, data transformations performed, and modeling approach employed. The models chosen for this project are logistic regression, decision trees, and support vector machines.

Logistic regression is a simple and efficient algorithm that works well with binary classification problems. Decision trees can handle both categorical and numerical data, making them a good choice for this problem. Support vector machines are a powerful algorithm that can handle both linear and nonlinear data, making them a good choice for this problem.

Finally, the project will combine the models to achieve the best performance in detecting fraudulent transactions and provide recommendations and insights for financial institutions to improve their fraud detection techniques.

* 1. **Loading and Pre-processing of data**

**Loading the Data**

The first step in any machine learning project is to load the data. In this case, we have a CSV file containing information about financial transactions. The fraud.csv dataset contains information about various financial transactions, such as transaction amount, type of transaction, transaction initiator and so on. The dataset also includes a binary label indicating whether the transaction is fraudulent or not.

To load the dataset, we will use the Pandas library. Pandas is a powerful data manipulation library in Python that allows us to load, manipulate, and analyze data easily. We will use the read\_csv function from the Pandas library to load the CSV file into a Pandas DataFrame. Once we have loaded the dataset, we can start preprocessing the data.



*Figure 1: [Code Snippet] Loading the data*

**Preprocessing the Data**

Preprocessing the data involves cleaning, transforming, and normalizing the data to make it suitable for analysis. In the case of the fraud.csv dataset, we will perform the following preprocessing steps:

1. Handling Missing Values:

The first step is to check for missing data. Missing data can occur due to various reasons, such as data entry errors, system failures, or incomplete data. Missing data can affect the accuracy of our model, so we need to handle them carefully. In this case, we don’t have any missing values to deal with.

1. Handling Outliers:

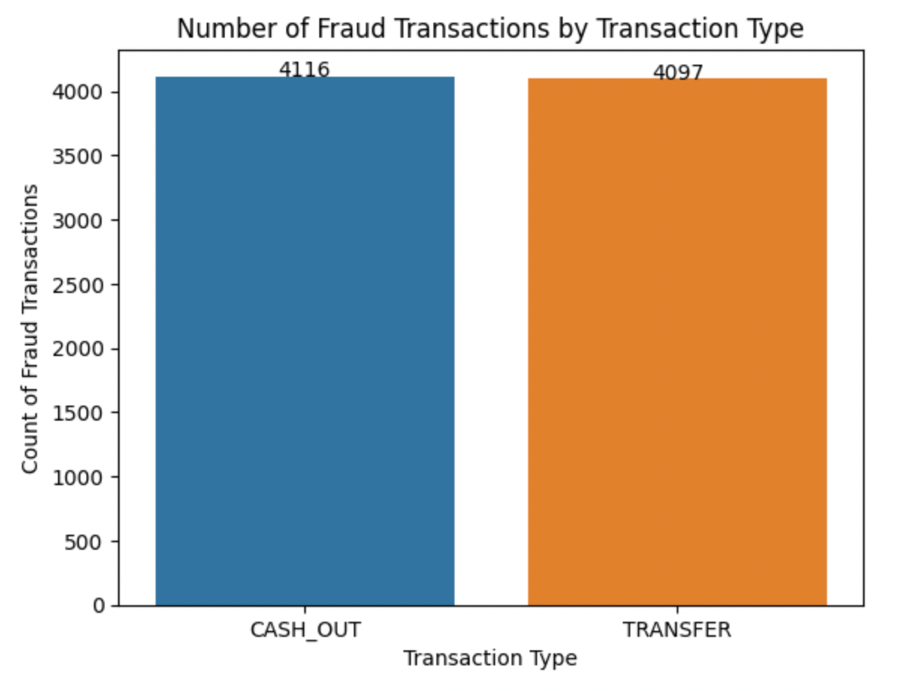
Outliers are extreme values that are significantly different from the rest of the data. Outliers can be caused by data entry errors or may be genuine data points that are significantly different from the rest of the data. In either case, outliers can affect the accuracy of our model, so we need to handle them carefully. In this case, we will use the z-score method to identify and remove outliers.

1. Encoding Categorical Variables:

The fraud.csv dataset contains categorical variables, such as merchant ID and account number. Machine learning algorithms work with numerical data, so we need to convert these categorical variables into numerical data. We will use the one-hot encoding method to encode categorical variables.

1. Splitting the Data:

Finally, we will split the data into training and testing sets. The training set will be used to train our machine learning model, while the testing set will be used to evaluate the performance of our model. In this case, we will use the train\_test\_split function from the Scikit-learn library to split the data.



*Figure 2: Count of fraud transactions based on transaction category*

Chart, box and whisker chart

Description automatically generated

*Figure 3: Distribution of Transaction Amount by Transaction Type*

**Feature Engineering**

We frequently come across various kinds of variables in the same dataset. The fact that the variables' ranges can be very different is a serious problem. The variables with a wide range may receive greater weight if the original scale is used. In the stage of data pre-processing, we must use the x technique of features rescaling to independent variables or features of the data to address this issue. Feature engineering is the process of taking raw data and extracting features that are useful for modeling. This is necessary for feature selection and the discovery of independent and dependent variables. The act of altering the amplitude of these features is called SCALING or Feature Scaling. As a result, we must scale them appropriately to account for the analysis and model preparations. The goal of normalization is to change the feature values so that they fall within the specified intervals (min and max) and so for that, we use MinMaxScaler class of sklearn. preprocessing module the **MinMaxScaler** **estimator** will **fit on the training data set**and the **same estimator**will be used to **transform the dataset.**

**Shape

Description automatically generated with medium confidence**

*Figure 4: [Code Snippet] New features creation*

* 1. **Classification Models**

1. **Decision Tree Classifier**

Decision Tree is a versatile supervised learning algorithm that can handle both classification and regression problems. By learning simple decision rules from historical data, it constructs a model that can predict the target variable's class or value for new data. The decision-making process involves comparing attribute values at each node and following the corresponding branch to the next node. Data is split into a training set and a test set in an 8:2 ratio, and no feature scaling techniques are necessary for decision tree classification. To increase the node's purity and homogeneity, various techniques are used to determine whether to create sub-nodes. Decision trees divide nodes based on all available variables and select the split that produces the most homogeneous sub-nodes. The algorithm used considers the target variable's type.

There are several types of decision tree algorithms:

**ID3:** A top-down greedy search algorithm that creates decision trees by exploring all possible branches without backtracking.

**C4.5:** An improvement on ID3 that uses the gain ratio, a variant of information gain that reduces bias, making it a popular choice. By considering the intrinsic information of a split, C4.5 corrects information gain.

**C5.0:** A popular algorithm due to its strong opinions on pruning and ability to automatically handle many decisions by selecting generally acceptable defaults. Post-pruning is part of the plan for C5.0.

**CART:** Also known as Classification and Regression Trees, is like C4.5 but accepts numerical target variables for regression and doesn't calculate rule sets. Instead, CART constructs binary trees using the feature and threshold that generates the most information gain at each node.

Our approach involves utilizing the Gini Index, which serves as the foundation for decision-making in Scikit-learn's optimized version of CART. The Gini Index is a metric used to assess dataset splits, and it functions as a cost function. It is computed by subtracting the sum of squared probabilities of each class from one. The Gini Index is advantageous in that it prioritizes larger partitions and is straightforward to implement, whereas information gain prioritizes smaller partitions with unique values.

Chart, scatter chart

Description automatically generated

*Figure 5: Decision Tree Overview*

*Chart

Description automatically generated with medium confidence*

*Figure 6: Confusion Matrix of Decision Tree*

1. **Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a powerful classification algorithm that can easily handle multiple continuous and categorical variables. It works by creating a hyperplane in a multidimensional space to separate different classes. The goal of SVM is to find a maximum marginal hyperplane (MMH) that optimally classifies the given dataset. SVM is a supervised learning technique that requires a set of labeled data to train the model. One of the key benefits of SVM is that it can be used for both classification and regression problems.

Support Vector Classification (SVC) is a variant of SVM that is specifically designed for classification tasks. Like SVM, SVC aims to create a hyperplane that separates the different classes in the dataset. However, SVC has a more flexible margin than SVM and can handle non-linearly separable data. SVC achieves this by introducing a kernel function that maps the data into a higher-dimensional space where the classes become separable by a hyperplane. In practice, SVC is often used to classify data in image and text recognition applications. It is also commonly used in bioinformatics and genetics research to classify and predict outcomes of diseases based on genetic data. To evaluate the performance of an SVC model, the dataset is typically split into training and testing subsets. The model is then trained on the training subset and evaluated on the testing subset to measure its accuracy. When working with an SVC model, it is important to consider the choice of kernel function, as it can have a significant impact on the model's performance. The most common kernel functions used in SVC are the linear kernel, polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel. The optimized result obtained for this dataset is RBF kernel.

Chart, treemap chart

Description automatically generated

*Figure 7: Confusion Matrix of Support Vector Classifier*

1. **Logistic Regression**

Logistic regression is a statistical method used to analyze data with one or more independent variables that determine an outcome. It is used to model the probability of a binary outcome, such as a categorical variable with two possible outcomes, as a function of one or more independent variables.

The logistic regression model uses a linear combination of independent variables to predict the probability of the binary outcome. The linear combination is transformed using the logistic function, also known as the sigmoid function, to ensure that the predicted probabilities are between 0 and 1.

The logistic regression equation can be expressed mathematically as:

p = 1 / (1 + e^(-z))

where p is the probability of the binary outcome, e is the base of the natural logarithm, and z is the linear combination of the independent variables. The linear combination can be represented as:

z = β0 + β1x1 + β2x2 + … + βkxk

where β0, β1, β2, ..., βk are the coefficients or parameters of the model,

and x1, x2, ..., xk are the independent variables.

The coefficients of the logistic regression model can be estimated using maximum likelihood estimation. This method finds the values of the parameters that maximize the likelihood of the observed data. Once the coefficients are estimated, they can be used to predict the probability of the binary outcome for new observation.

We are using grid search cross-validation to find the best hyperparameters for the logistic regression model. By tuning the regularization parameter (C) and the penalty (l1 or l2), we can improve the performance of the model and prevent overfitting.

Finally, we can see that evaluating the performance of the logistic regression model on both the training and test sets using cross-validation and accuracy score. We also plot the confusion matrix to visualize the model's performance in terms of true positives, true negatives, false positives, and false negatives. This helps us to identify which class the model is misclassifying and to assess the overall performance of the model.

Chart, line chart

Description automatically generated

*Figure 8: ROC of Logistic Regression*

1. **K-Nearest Neighbors**

The K-nearest neighbor (KNN) algorithm is a popular supervised learning approach that can be used for both classification and regression tasks. It is a simple, intuitive, and flexible algorithm that is easy to implement and understand. The basic idea behind the KNN algorithm is to predict the class of a new data point based on the classes of the K-nearest data points in the training dataset. The algorithm does not learn a discriminative function but instead relies on the similarity between the input data and the training data to make predictions.

The KNN algorithm is a lazy learner because it does not require any training or model building. It simply stores the training data and uses it to make predictions for new data points. The algorithm is non-parametric, which means that it does not make any assumptions about the distribution of the data or the nature of the data itself. This makes it very flexible and adaptable to different types of data.

The KNN algorithm assumes that similar data points tend to belong to the same class. The similarity between two data points can be measured using a distance metric such as Euclidean distance or cosine similarity. The KNN algorithm works by finding the K-nearest data points to the new data point in the training dataset, where K is a user-defined parameter. The class of the new data point is then assigned based on the majority class of the K-nearest neighbors.

To use the KNN algorithm for classification, we first split the dataset into a training set and a test set using the train-test split method from the Scikit-Learn library. The training set is used to train the KNN classifier, and the test set is used to evaluate its performance. We can choose the value of K based on the performance of the classifier on the test set.

Chart, line chart

Description automatically generated

*Figure 9: K-Nearest Neighbor Cross Validation Scores*

Bottom of Form

1. **XGBoost**

XGBoost is a powerful machine learning algorithm that is used for supervised learning tasks, such as classification and regression. It is based on the gradient boosting framework, which is an ensemble learning technique that combines multiple weak models to form a strong model. In simple terms, XGBoost takes a set of input data and a set of labeled outputs, and it trains a decision tree model to predict the output values based on the input data. It then iteratively adds more decision trees to the model, each one correcting the errors of the previous trees, until the overall model achieves high accuracy on the training set.

One of the key features of XGBoost is that it allows for both L1 and L2 regularization techniques, which helps to prevent overfitting and improve the generalization ability of the model. Additionally, XGBoost can handle missing data and has built-in cross-validation to help find the optimal number of iterations for the specific problem.

To evaluate the accuracy of an XGBoost model, the most commonly used metric is the classification accuracy (for classification problems) or the mean squared error (for regression problems). These metrics measure how well the model predicts the correct output values on a set of test data that the model has not seen before.

Overall, XGBoost is a highly effective algorithm for achieving high accuracy in supervised learning tasks, and it is widely used in industry and academia for a variety of applications.

**Chart

Description automatically generated**

*Figure 10: XGBoost Algorithm*

* 1. **Flowchart**

**Timeline

Description automatically generated**

*Figure 11: Flowchart*

1. **Description of the Dataset**

The dataset is available on Kaggle in the form of csv file. It consists of 6362620 rows and 11 columns. It is combined of both numerical and categorical data. Each row represents a single transaction, and the columns provide details about the transaction such as the transaction type, the amount of money involved, the customer who initiated the transaction, etc. The dataset also contains a column indicating whether the transaction was fraudulent or not, which serves as the target variable. About 13% of the transactions in the dataset are labeled as fraudulent.

Table

Description automatically generated

*Figure 12: Description of the Dataset*

Details of the Columns

1. step   
   Represents a unit of time in the dataset. It is a continuous number that starts from 1 and goes up to 743.
2. type  
   Represents the type of transaction, which can be one of the following: CASH\_IN, CASH\_OUT, DEBIT, PAYMENT, or TRANSFER.
3. amount  
   Represents the transaction amount in the local currency.
4. nameOrig   
   Represents the customer who started the transaction.
5. nameDest  
   Represents the customer who received the transaction.
6. oldbalanceOrg  
   Represents the initial balance of the customer who started the transaction.
7. newbalanceOrig  
   Represents the new balance of the customer who started the transaction.
8. oldbalanceDest  
   Represents the new balance of the customer who received the transaction.
9. newbalanceDest  
   Represents the new balance of the customer who received the transaction.
10. isFraud  
    Represents whether the transaction is fraudulent (1) or not (0).
11. isFlaggedFraud  
    Represents whether the transaction is flagged as fraudulent by the bank’s anti-fraud system (1) or not (0).

A snapshot of the raw dataset:

Graphical user interface, application, table

Description automatically generated

*Figure 13: Raw Data*

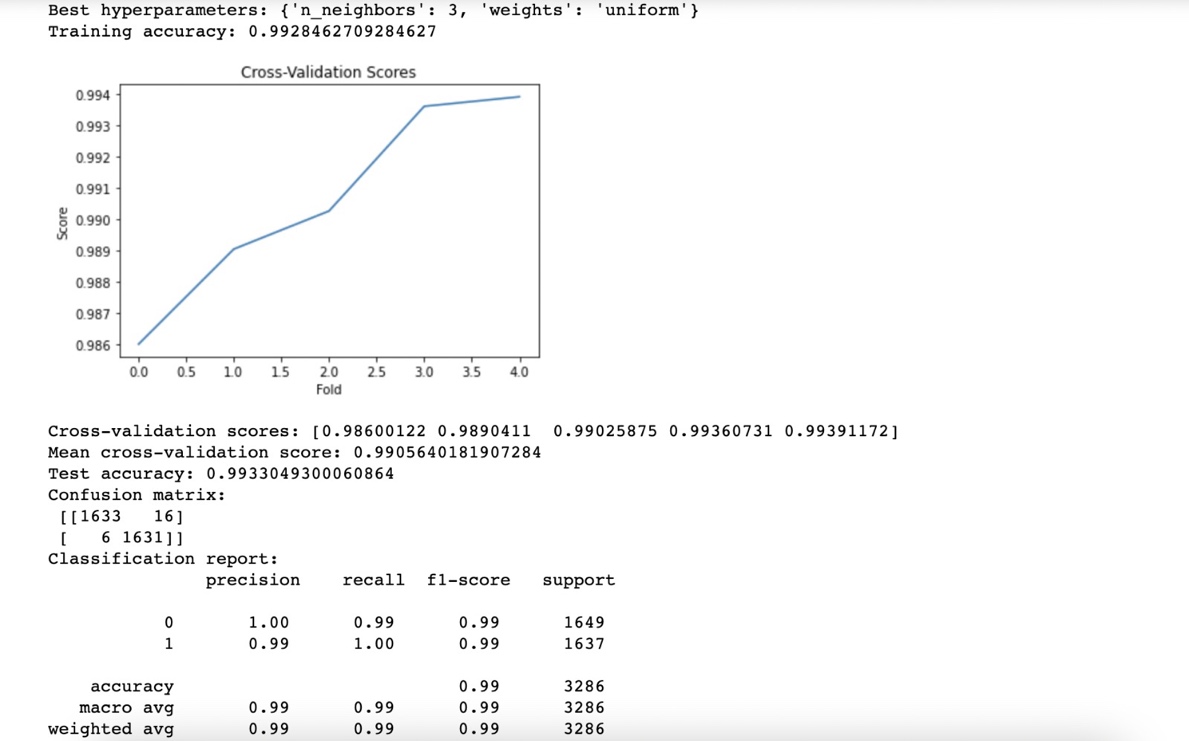
* 1. **Data Source**

Kaggle: <https://www.kaggle.com/datasets/chitwanmanchanda/fraudulent-transactions-data>

1. **Result and Analysis**

Our final analysis summarizes the train and test accuracies of various machine learning models as follows:

**K-Nearest Neighbor**



*Figure 14: K-Nearest Neighbors Analysis and Accuracy*

**Logistic Regression**

**A picture containing table

Description automatically generated**

*Figure 15: Logistic Regression Analysis and Accuracy*

**Decision Tree**

**A screenshot of a computer

Description automatically generated with medium confidence**

*Figure 16: Decision Tree Analysis and Accuracy before Optimization*

*Table

Description automatically generated with low confidence*

*Figure 17: Decision Tree Analysis and Accuracy after Optimization*

**XGBoost**

**Table

Description automatically generated**

*Figure 18: XGBoost Analysis and Accuracy*

**Support Vector Classifier (SVC)**

*Chart, treemap chart

Description automatically generated*

*Figure 19: SVC Analysis and Accuracy with RBF Kernel – Part 1*

**Chart

Description automatically generated**

*Figure 20: Support Vector Classifier ROC and Confusion Matrix*

1. **Conclusion**

We have successfully built a framework for detecting fraudulent transactions in financial data, which considers important factors such as the creation of new variables to better differentiate between fraud and non-fraud transactions, dealing with imbalanced data, and selecting the appropriate machine learning algorithm. Our framework includes experimenting with five commonly used algorithms: Logistic Regression, K-Nearest Neighbor, XGBoost, Decision Tree and Support Vector Machine. Results show that the XGBoost algorithm and K- Nearest Neighbor outperformed other algorithms, highlighting the effectiveness of tree-based models for transaction data with distinct classes. Our exploratory analysis of the data also revealed some derived features that performed better in distinguishing between classes compared to the original data. This highlights the importance of thorough data exploration before building machine learning models.

* 1. **Future Scope**
* Utilize deep learning models such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs) to improve the accuracy of fraud detection. These models have been shown to be effective in other domains such as image classification and natural language processing and may be able to capture more complex relationships in the transaction data.
* Incorporate additional features into the model such as transaction location, time of day, and merchant information. This could improve the model's ability to detect fraudulent transactions, as fraudsters may exhibit specific patterns or behaviors that can be detected through these features.
* Investigate the use of explainable artificial intelligence (XAI) techniques to better understand how the models are making predictions. This could help identify areas for improvement in the model or highlight specific features that are particularly important for detecting fraud.
* Expand the scope of the project beyond transaction data to include other sources of information such as customer behavior and social media activity. This could provide a more comprehensive view of potential fraudsters and enable the model to detect fraud more accurately.
* Incorporate real-time monitoring and alerting capabilities into the system to enable faster detection and response to potential fraud. This could be particularly valuable in situations where fraudsters are able to rapidly change their behavior and tactics.
* Investigate the use of transfer learning to apply the fraud detection model to other domains such as insurance fraud or healthcare fraud. This could potentially reduce theneed for large amounts of labeled data and enable the model to be adapted to new use cases more quickly.

1. **References**

* [**https://www.kaggle.com/datasets/chitwanmanchanda/fraudulent-transactions-data**](https://www.kaggle.com/datasets/chitwanmanchanda/fraudulent-transactions-data)
* [**https://pub.towardsai.net/the-confusion-matrix-for-classification-eb3bcf3064c7**](https://pub.towardsai.net/the-confusion-matrix-for-classification-eb3bcf3064c7)
* [**https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a**](https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a)
* [**https://www.kaggle.com/code/vinyksd/fd-accuracy-99-89**](https://www.kaggle.com/code/vinyksd/fd-accuracy-99-89)
* [**https://www.kaggle.com/code/aayushmanjain/easy-analysis-using-xgboost**](https://www.kaggle.com/code/aayushmanjain/easy-analysis-using-xgboost)
* [**https://link.springer.com/book/10.1007/978-3-031-02363-7**](https://link.springer.com/book/10.1007/978-3-031-02363-7)
* [**https://chat.openai.com/**](https://chat.openai.com/)