

Fraud Detecti in Financial Transactions



**Project Team:**

* Mohamed Abdelhamid Mohamed
* Abdelrhman Ashraf Abdelkarim
* Yassin Mohsen Yassin

Mahmoud Hesham Shaban

October 24, 2024

# Program: Digital Egypt Pioneers Initiative - DEPI Project Supervisor: Eng./Mahmoud Safian

**Acknowledgement**

We would like to express our deepest gratitude to our supervisor Eng/Mahmoud Safian for the continuous support and guidance throughout the development of this project. We are also thankful to the DEPI for providing the knowledge and resources necessary to complete this work. Furthermore, We would like to acknowledge our colleagues and family for their encouragement and patience during the completion of this project. Lastly, our thanks go out to our supervisor for providing access to financial transaction datasets and resources for our project on fraud detection.

**Abstract**

The goal of this project is to design a fraud detection model capable of identifying fraudulent financial transactions using machine learning techniques. The project covers data preprocessing, statistical analysis, model development, and deployment using various advanced methods including NLP and MLOps. The project leverages Python, Scikit-learn, and MLflow for model evaluation and tracking The final model demonstrates high accuracy, precision and recall on a labeled dataset, making it an effective tool for detecting anomalous transaction behaviors.This project aims to develop an intelligent system for detecting bank fraud using artificial intelligence techniques while addressing the challenge of imbalanced data. Our model analyzes both structured transactional data and unstructured text data. To handle the imbalance, we applied techniques such as SMOTE and random sampling, while text data were augmented to improve the model’s robustness. Initial accuracy was high but biased, necessitating further optimization. After applying the balancing techniques, the recall and F1-score values significantly improved, ensuring a more reliable fraud detection system. Additionally, we utilized MLflow for tracking all models, metrics, and reports, ensuring a streamlined experiment management process.

**Introduction & Background**

**Introduction**:

As digital banking transactions become more prevalent, fraud attempts have also increased, posing significant risks to financial institutions. Detecting fraudulent activities in a timely manner is essential to mitigate financial losses and maintain trust in banking systems. Machine learning techniques provide a promising approach for building automated fraud detection systems.

**Background:**

One of the primary challenges in fraud detection is the imbalance in datasets, where fraudulent transactions constitute a small percentage of the data. Standard models trained on such data often result in biased predictions. Additionally, missing values (nulls) are common in financial datasets, requiring careful data preprocessing. Incorporating NLP to analyze transaction descriptions offers added value by identifying hidden patterns. This project aims to address these challenges through a systematic approach combining machine learning, NLP, and MLOps.

**Problem Statement**

The dataset used in this project contains an imbalanced distribution between fraudulent and non-fraudulent transactions, which impacts model performance by skewing predictions toward the majority class. Additionally, missing values in the data can reduce the quality of the results. A further challenge lies in managing multiple experiments effectively while maintaining transparency. The goal of this project is to develop a fraud detection system that accurately identifies fraudulent transactions, even with imbalanced data, and effectively manages models through MLflow

**Project Aims and Objectives**

**Aim**:

To develop an AI-powered fraud detection system capable of handling imbalanced data and tracking models effectively using MLOps tools.

**Objectives:**

1. Data Preprocessing: Clean the dataset and handle missing values.

2. Statistical Analysis: Analyze the distribution of features related to fraud detection.

3. Model Development: Build machine learning models (e.g., Logistic Regression, Random Forest) for fraud detection.

4. NLP Integration: Use NLP techniques to analyze transaction notes for improved insights.

5. MLOps Management: Utilize MLflow to log, manage, and compare multiple models.

6. Performance Evaluation: Optimize models based on key metrics, including Recall and F1-score

**Proposed Approach (Methodology):**

**Week 1: Data Collection and Preprocessing**

Tasks:

Obtain financial transaction data, including labeled fraudulent and non-fraudulent transactions.

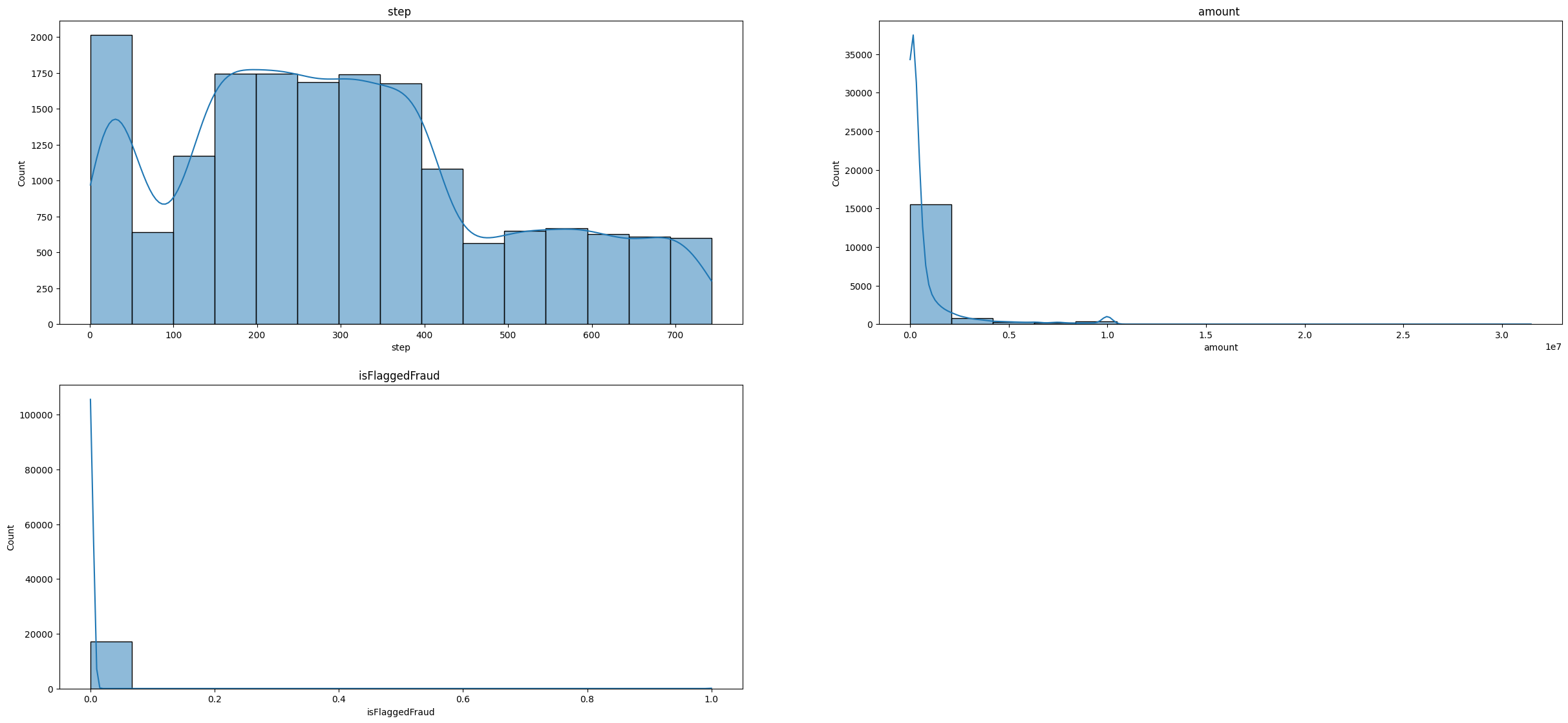
Clean and preprocess the data by addressing missing values and normalizing features.

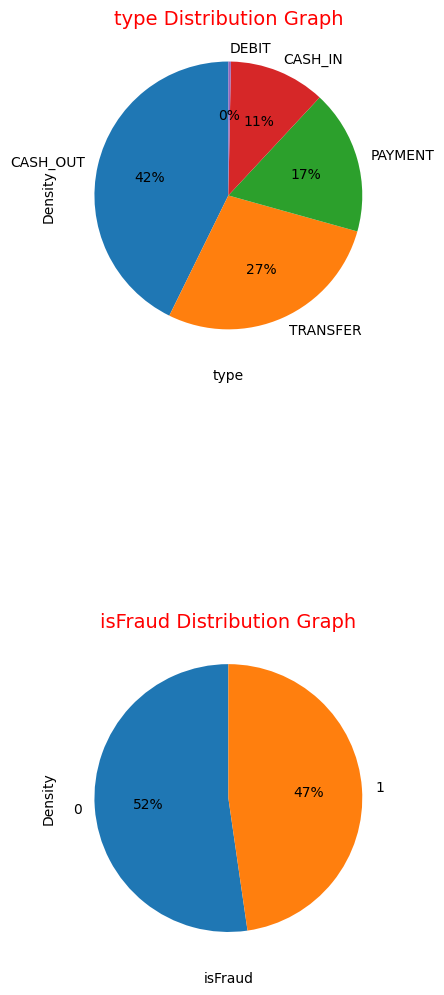
Tools: Python (Pandas, NumPy)

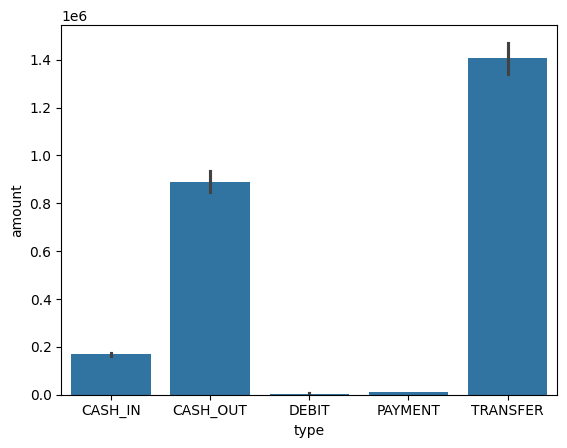
Deliverables:

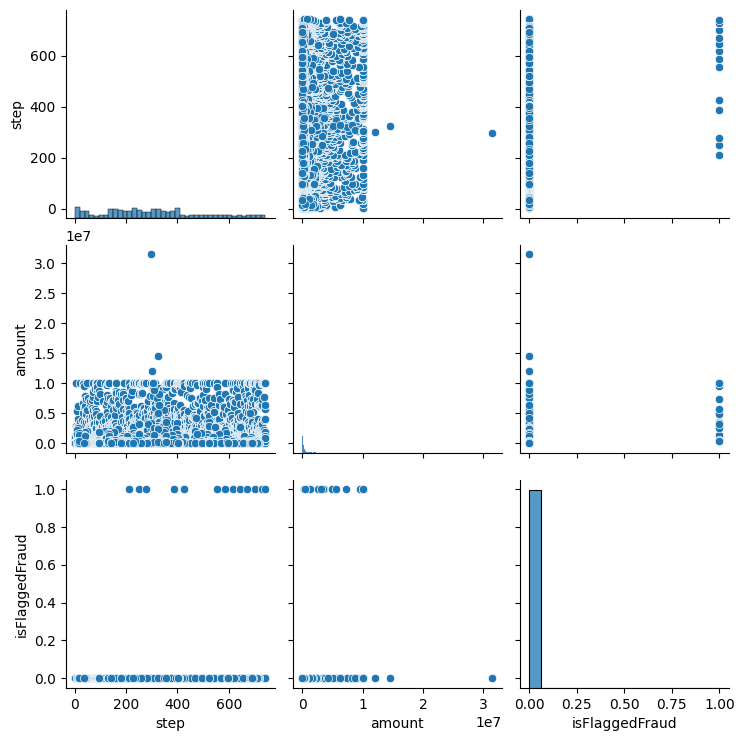
* Printing & Understanding the Data
* Check for Data types
* data reduction
* Visualize our data
* Normalization
* Encoding
* split the data
* resolve the unbalanced data

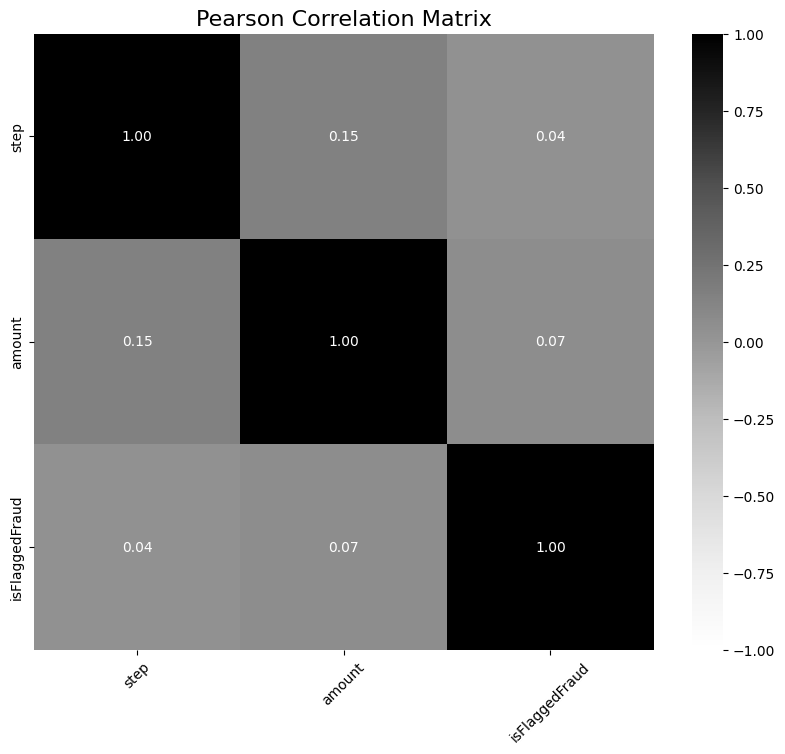
Visualize



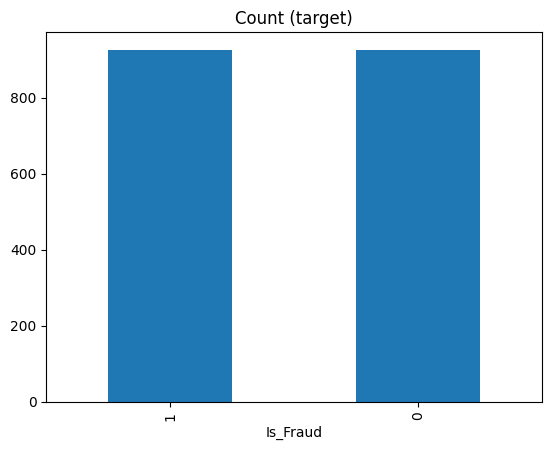


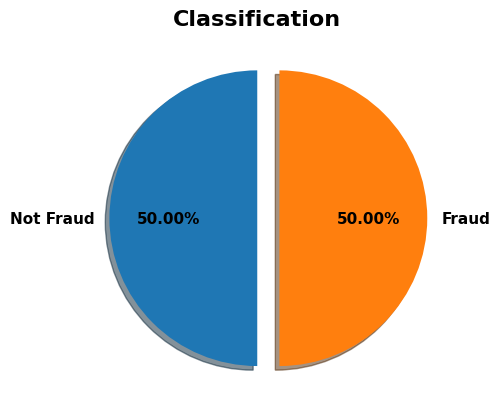


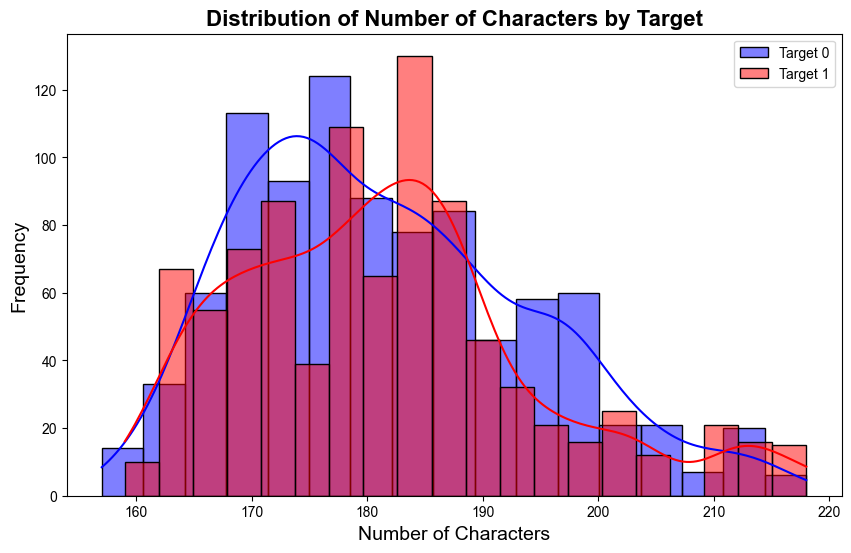


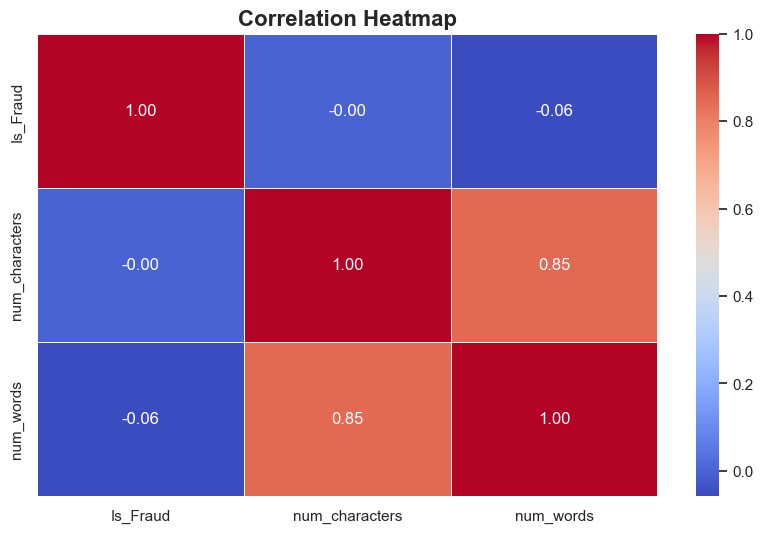


NLP Data:









**Week 2: Statistical Analysis and Machine Learning**

Tasks:

Perform statistical analysis to understand the distribution of fraud-related features.

Develop classification models such as Logistic Regression and Random Forest for fraud detection.

Tools: Python (Scikit-learn, Statsmodels)

Deliverables:

Statistical analysis report.

Fraud detection models and performance metrics.

**Week 3: Advanced Techniques**

Tasks:

Apply NLP techniques to analyze transaction descriptions or notes.

Integrate models.

Tools: Python (NLTK, SpaCy)

Deliverables:

Enhanced fraud detection model with NLP integration.

**Week 4: MLOps and Final Presentation**

Tasks:

Use MLflow to track and manage fraud detection models.

Prepare the final report and presentation documenting all aspects of the project.

Tools: MLflow

Deliverables:

Deployed fraud detection model.

**Challenges and Solutions with Evidence:**

**1. Challenge: Imbalanced Data**

Issue:

Initially, the dataset was heavily imbalanced, skewing predictions toward the majority class. As a result, the model’s accuracy was high, but the performance metrics such as Recall and F1-Score were low due to High Bias.

Solution:

To address the imbalance, we applied SMOTE (Synthetic Minority Over-sampling Technique) and Random Sampling to balance the class distribution.

**2. Challenge: Handling Null Values**

Issue:

The dataset contained several Null (missing) values, which could negatively impact the model's performance.

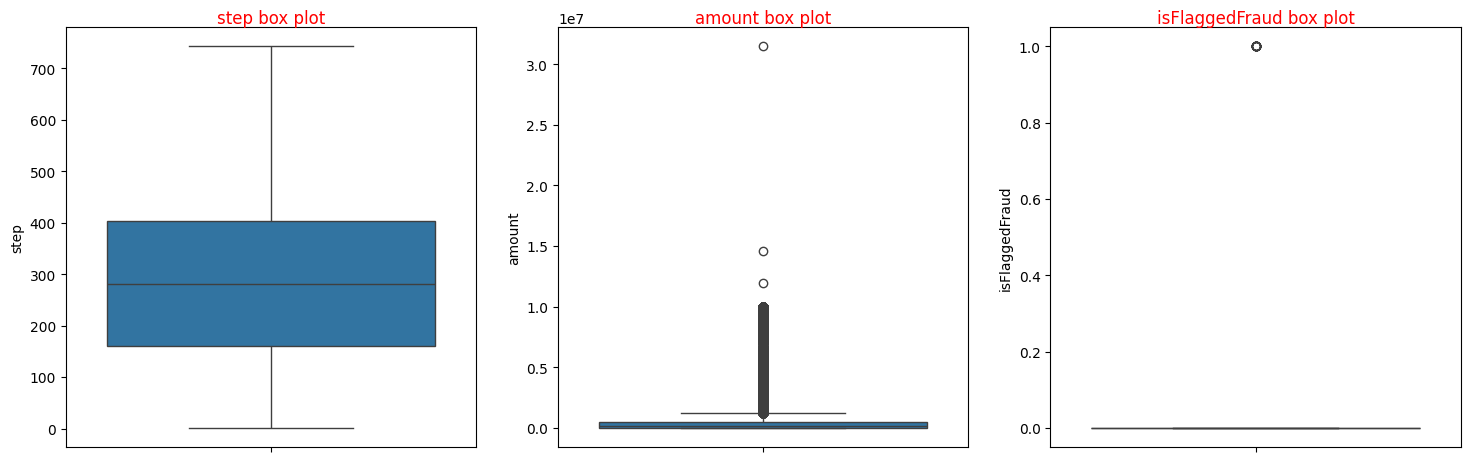
Solution:

We used data cleaning techniques, such as removing rows with excessive missing values or filling them with the average values of the column to maintain data quality.

**4. Challenge: Handling outliers**

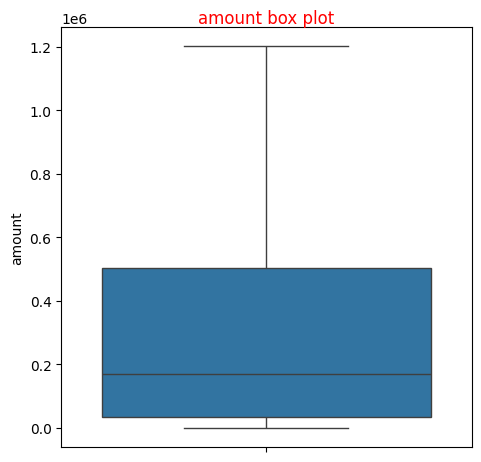
Issue:

The dataset contained outliers values, which could negatively impact the model's performance.



Solution:

We used data outliers handel techniques



**3. Challenge: Initial Model Performance - High Accuracy but High Bias**

Issue:

Initially, the model achieved high Accuracy, but this was misleading as the Recall and F1-Score were low, indicating the presence of High Bias.

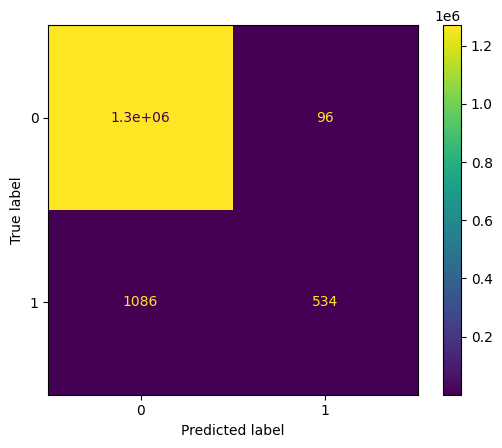
Solution:

After balancing the dataset using SMOTE, the Recall and F1-Score improved significantly, providing a more accurate and fair assessment of the model’s performance.

**Evidence:**

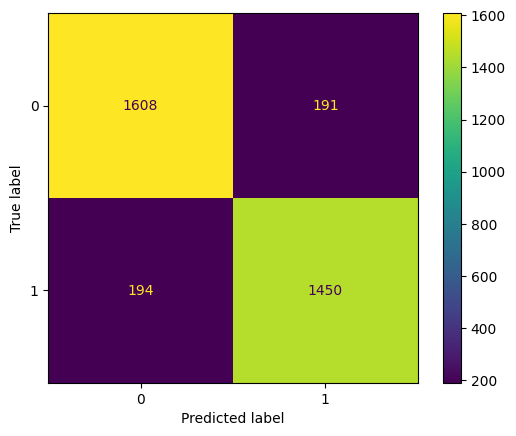
Before Optimization:

Insert screenshot showing the confusion matrix and metrics with high accuracy but poor Recall and F1-Score (evidence of bias).

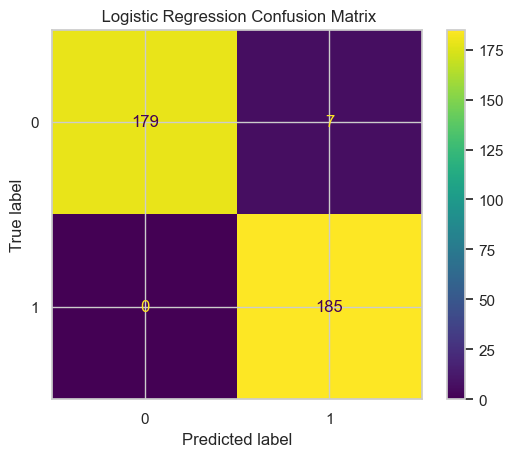


After Optimization:

Insert screenshot showing the confusion matrix and metrics after applying balancing techniques, with improved Recall and F1-Score.



**NLP data:**

****

**4. Challenge: Tracking Models and Metrics with MLflow**

Issue:

Managing multiple experiments and comparing model performance across various algorithms was challenging without a structured tracking tool.

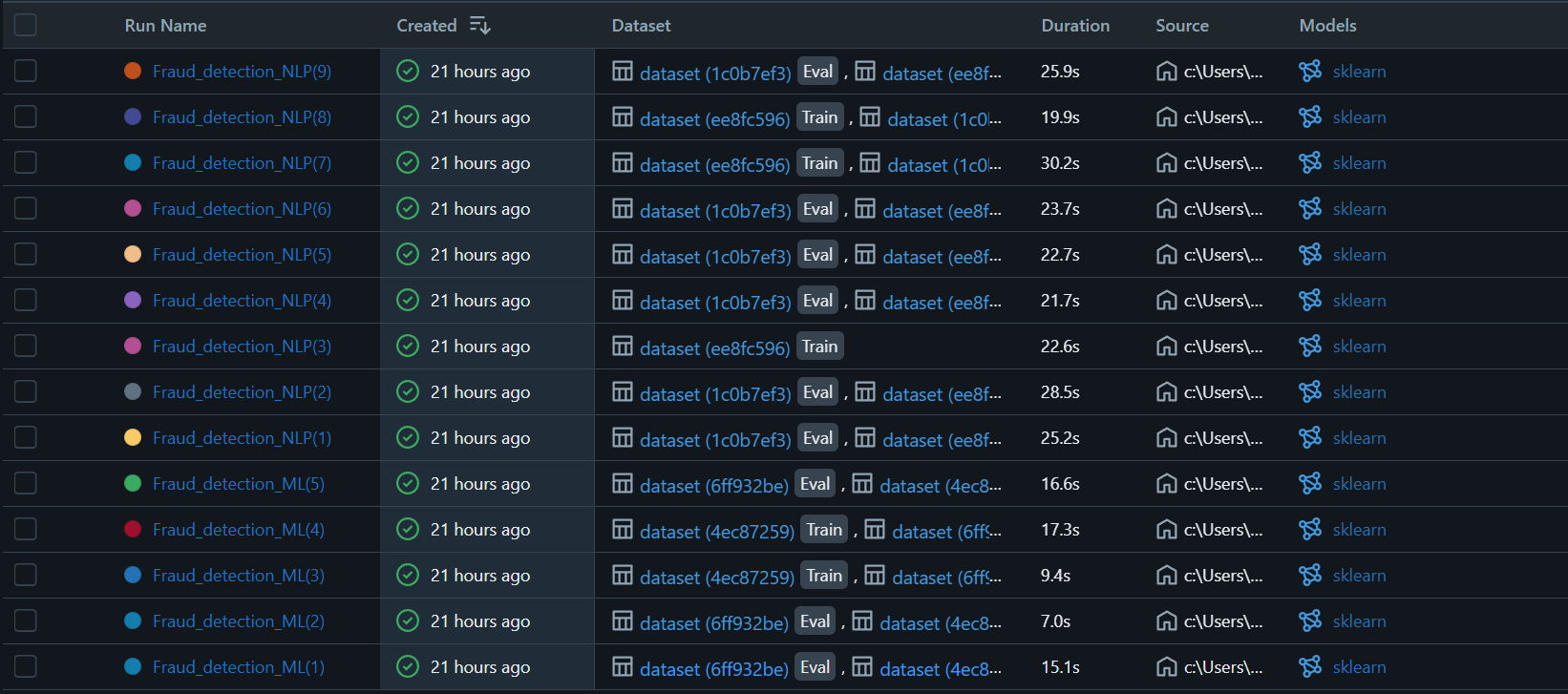
Solution:

We used MLflow to log all experiments, metrics, and models systematically. Each algorithm and its respective results were registered for easy comparison and reproducibility.

**Evidence:**

MLflow Dashboard:

screenshot of the MLflow dashboard showing registered models, logged metrics, and performance comparisons.



**Results**

The use of SMOTE improved the Recall and F1-score, balancing the model’s performance.

NLP integration provided deeper insights from transaction notes, improving the fraud detection capabilities.

MLflow streamlined experiment tracking, enhancing transparency and reproducibility.

Insert graphs and screenshots showing the key performance metrics.

**Conclusion**

This project successfully developed a fraud detection system that effectively handles imbalanced data and achieves reliable results. The initial model displayed high accuracy, but balancing techniques such as SMOTE significantly improved recall and F1-score, reducing bias. MLflow proved to be a valuable tool in managing experiments, tracking models, and maintaining transparency. Future improvements could include using deep learning techniques like RNNs to enhance the system’s ability to analyze sequential data and integrating real-time data streams to improve responsiveness.

**References**

1. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.

2. Chawla, N. V., et al. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research.

3. Zaharia, M., et al. (2018). MLflow: An Open-Source Platform for Managing the Machine Learning Lifecycle. MLflow Documentation.