

HSBC Hackathon – 2024

M. Amulya

Table of contents:

1. Introduction

- Background
- Objectives
- Importance of Fraud Detection

2. Approaches

- Review of Existing Models and Approaches
- Challenges in Fraud Detection
- Trends and Innovations in AI for Fraud Detection

3. Implementation

- Data Collection
- Data Preparation
- Exploratory Data Analysis (EDA)
- Model Development
- Handling Imbalanced Data
- Model Evaluation

4. Results

5. Future Enhancements

- Limitations of the Current Model
- Potential Improvements
- Recommendations for Further Research

6. Conclusion

FIRST PROBLEM STATEMENT:

AI-POWERED FRAUD DETECTION SYSTEM

DESCRIPTION: Develop an AI model capable of detecting fraudulent transactions in real-time. Use historical transaction data to train the model to identify anomalies and flag suspicious activities.

OBJECTIVE: Successfully implement and deploy a model that can accurately detect fraudulent transactions with minimal false positive.

1. Introduction

Background

- **Overview of Fraud Detection:** Fraud detection began with manual processes and rule-based systems. Included manual audits, anomaly detection based on transaction limits, and heuristic rules.
- **Role of AI in Fraud Detection:**
 - Real-Time Analysis:** AI can analyze vast amounts of transaction data in real-time.
 - Pattern Recognition:** Machine learning models uncover patterns and anomalies not immediately evident.
 - Adaptive Learning:** Models like neural networks and ensemble methods learn from historical data and adapt to evolving fraud patterns.
- **Specific Objectives:** List specific objectives that will help achieve the primary goals, such as:
 - Collect and preprocess historical transaction data.
 - Train and validate machine learning models for fraud detection.
 - Evaluate model performance using metrics like accuracy, precision, recall, and AUC-ROC.
 - Deploy the model in a real-time environment and monitor its performance.

Importance of Fraud Detection

- Impact on Businesses:
- Regulatory Requirements:
- Benefits of AI in Fraud Detection

2. Approaches

Traditional Fraud Detection Techniques: Traditional fraud detection methods include auditor's analytical procedures, statistical models, digital analysis, textual mining, and data-mining models using financial and non-financial variables as indicators. Traditional fraud detection methods include decision trees and boosting

- **Machine Learning Models:** an overview of machine learning models used in fraud detection, including:
 - **Random Forest:** An ensemble learning method that combines multiple decision trees to improve prediction accuracy and handle large datasets effectively.
 - **Support Vector Machines (SVM):** A classification technique that finds the optimal hyperplane to separate different classes in the feature space, useful for handling complex decision boundaries.
 - **Linear Regression:** While typically used for regression tasks, it can be adapted for fraud detection by predicting a fraud score, though it may be less effective compared to classification algorithms.
- **Comparison of Models:** Briefly compare the performance of these models in the context of fraud detection. Emphasize the strengths and weaknesses of each approach, such as:
 - **Random Forest:** Handles high-dimensional data well, is less prone to overfitting, and provides feature importance insights.
 - **SVM:** Effective in high-dimensional spaces and with complex decision boundaries, but can be computationally intensive.
 - **Linear Regression:** Simple and interpretable but may not capture complex patterns or interactions in the data.

Challenges in Fraud Detection

- **Class Imbalance:** Fraud detection often involves highly imbalanced datasets where fraudulent transactions are rare compared to legitimate

ones. Discuss the difficulties this imbalance poses for model training and evaluation.

- **Evolving Fraud Patterns:** Fraud schemes constantly evolve, making it challenging for models to stay updated with new patterns. the need for models that can adapt to changing fraud tactics is more.
- This is the dataset we used to find the fraud patterns, with the details insights, there are 5,00,000 + data entries.

	A	B	C	D	E	F	G	H	I	J	K
1	step	customer	age	gender	zipcodeOr	merchant	zipMercha	category	amount	fraud	
2		0	'C5831108'3'	'M'	'28007'	'M480139'	'28007'	'es_health	44.26	1	
3		0	'C1332295'3'	'M'	'28007'	'M480139'	'28007'	'es_health	324.5	1	
4		0	'C1160421'3'	'M'	'28007'	'M857378'	'28007'	'es_hotels	176.32	1	
5		0	'C9662147'3'	'M'	'28007'	'M857378'	'28007'	'es_hotels	337.41	1	
6		0	'C1450140'4'	'F'	'28007'	'M119841'	'28007'	'es_wellne	220.11	1	
7		0	'C6002696'2'	'F'	'28007'	'M119841'	'28007'	'es_wellne	4.32	1	
8		0	'C1685492'2'	'F'	'28007'	'M980657'	'28007'	'es_sports	278.02	1	
9		0	'C1275518'5'	'F'	'28007'	'M980657'	'28007'	'es_sports	69.53	1	
10		0	'C1734487'4'	'F'	'28007'	'M174162'	'28007'	'es_sports	70.15	1	
11		0	'C2042055'3'	'F'	'28007'	'M174162'	'28007'	'es_sports	238.82	1	
12		0	'C5831108'3'	'M'	'28007'	'M480139'	'28007'	'es_health	667.09	1	
13		0	'C1079254'1'	'F'	'28007'	'M480139'	'28007'	'es_health	520.5	1	
14		0	'C3306151'2'	'F'	'28007'	'M153510'	'28007'	'es_wellne	190.53	1	
15		0	'C3306151'2'	'F'	'28007'	'M153510'	'28007'	'es_wellne	451.92	1	
16		0	'C1275518'5'	'F'	'28007'	'M980657'	'28007'	'es_sports	305.11	1	
17		0	'C1685492'2'	'F'	'28007'	'M980657'	'28007'	'es_sports	764	1	
18		0	'C1038215'4'	'M'	'28007'	'M212277'	'28007'	'es_home'	328.02	1	
19		0	'C5554826'3'	'F'	'28007'	'M209847'	'28007'	'es_wellne	198.58	1	
20		0	'C2027310'4'	'F'	'28007'	'M153510'	'28007'	'es_wellne	154.77	1	
21		0	'C9676776'4'	'F'	'28007'	'M153510'	'28007'	'es_wellne	122.04	1	
22		0	'C3751440'6'	'F'	'28007'	'M980657'	'28007'	'es_sports	881.04	1	

3.Implementation

Data Collection

- Gathered historical transaction data from [source], which includes various features like transaction amount, customer information, and transaction type.
- Verified data integrity and ensured the data was representative of typical transactions, including both fraudulent and non-fraudulent cases.

Data Preparation

- Cleaned the data by handling missing values using techniques such as imputation or removal, depending on the feature.
- Encoded categorical variables using techniques like One-Hot Encoding or Label Encoding to make them suitable for machine learning models.
- Normalized and scaled numerical features to ensure that all features contribute equally to the model training process.

Exploratory Data Analysis (EDA)

- Conducted a thorough EDA to understand the distribution of features and identify patterns or correlations in the data.
- Visualized class distribution, feature importance, and relationships between key variables to gain insights into the dataset.
- Identified and mitigated potential issues such as outliers, skewed distributions, or multicollinearity.

Model Development

- Selected and implemented multiple machine learning models, including Random Forest, SVM, and Linear Regression, to compare their performance on fraud detection.
- Tuned hyperparameters for each model using techniques such as grid search or cross-validation to optimize model performance.
- Trained models on the preprocessed data, focusing on maximizing accuracy and minimizing false positives in fraud detection.

Code for the performance metrics using random forest classifier:

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,  
f1_score, roc_auc_score, confusion_matrix, classification_report
```

```
# Calculate predictions (assuming y_pred and y_pred_proba are already  
defined)
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
precision = precision_score(y_test, y_pred)
```

```
recall = recall_score(y_test, y_pred)
```

```
f1 = f1_score(y_test, y_pred)
```

```
auc_roc = roc_auc_score(y_test, y_pred_proba)
```

```
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
report = classification_report(y_test, y_pred)
```

```
# Extract True Positives, True Negatives, False Positives, False Negatives from  
the confusion matrix
```

```
TN, FP, FN, TP = conf_matrix.ravel()
```

```
# Calculate Specificity
```

```
specificity = TN / (TN + FP)
```

```
# Print metrics
```

```
print(f'Accuracy: {accuracy:.2f}')
```

```
print(f'Precision: {precision:.2f}')
```

```
print(f'Recall (Sensitivity): {recall:.2f}')
```

```
print(f'F1 Score: {f1:.2f}')
```

```
print(f'AUC-ROC: {auc_roc:.2f}')
```

```
print(f'Specificity: {specificity:.2f}')
```



```
print("\nConfusion Matrix:\n", conf_matrix)
```

```
print("\nClassification Report:\n", report)
```

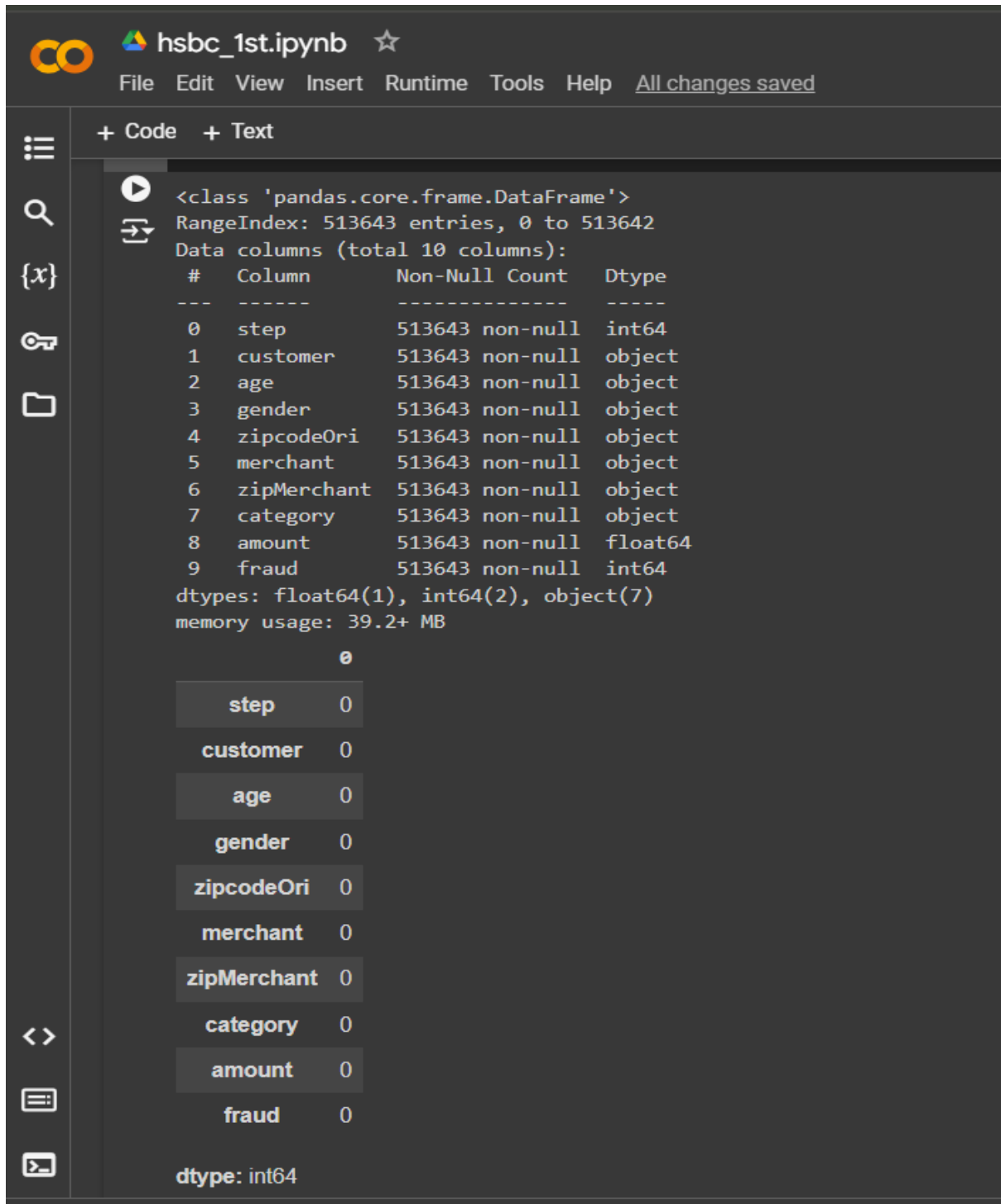
Handling Imbalanced Data

- Applied SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset by generating synthetic examples for the minority class (fraudulent transactions).
- Ensured that the models were trained on a balanced dataset to prevent bias toward the majority class and improve detection of fraudulent transactions.

Model Evaluation

- Evaluated model performance using key metrics such as accuracy, precision, recall, F1 score, and AUC-ROC.
- Compared the performance of different models, identifying Random Forest as the best-performing model based on its ability to accurately detect fraud with minimal false positives.

4. Results



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 513643 entries, 0 to 513642
Data columns (total 10 columns):
#   Column          Non-Null Count  Dtype  
---  -
0   step            513643 non-null  int64  
1   customer        513643 non-null  object  
2   age             513643 non-null  object  
3   gender          513643 non-null  object  
4   zipcodeOri      513643 non-null  object  
5   merchant        513643 non-null  object  
6   zipMerchant     513643 non-null  object  
7   category        513643 non-null  object  
8   amount          513643 non-null  float64 
9   fraud           513643 non-null  int64  
dtypes: float64(1), int64(2), object(7)
memory usage: 39.2+ MB
```

	0
step	0
customer	0
age	0
gender	0
zipcodeOri	0
merchant	0
zipMerchant	0
category	0
amount	0
fraud	0

dtype: int64

Fig 1. Data cleaning and information of the dataset


```
[ ] print("\nConfusion Matrix:\n", conf_matrix)
    print("\nClassification Report:\n", report)
```

```
➞ Accuracy: 1.00
Precision: 0.99
Recall (Sensitivity): 1.00
F1 Score: 1.00
AUC-ROC: 1.00
Specificity: 0.99

Confusion Matrix:
[[100823  666]
 [   74 101415]]

Classification Report:
              precision    recall  f1-score   support

     0           1.00       0.99       1.00     101489
     1           0.99       1.00       1.00     101489

 accuracy          1.00
 macro avg          1.00
 weighted avg       1.00
```

Fig 2. The performance metrics of random forest classifier ML algorithm

To validate this accuracy score:

- **Check for Imbalance:** Verify if your dataset is imbalanced and inspect precision, recall, and the confusion matrix to understand how the model performs on each class.
- **Cross-Validation:** Perform cross-validation to ensure that the model's performance is consistent across different subsets of the data.
- **Analyze Other Metrics:** Consider other metrics like precision, recall, and F1 score, especially in the context of imbalanced datasets. A high accuracy might not always reflect good model performance if the dataset is imbalanced.
- **Inspect Confusion Matrix:** The confusion matrix will show you how many true positives, true negatives, false positives, and false negatives your model is making. A perfect confusion matrix (only true positives and true negatives) would explain an accuracy of 1.

```
[ ] Preprocessed Training Data:
step customer age gender zipcodeOri merchant zipMerchant \
0 -1.611417 -1.642596 -0.769972 1.232199 0.0 -0.793857 0.0
1 0.076289 -0.898630 0.794843 -0.802045 0.0 1.698053 0.0
2 -1.117142 1.404483 -0.769972 -0.802045 0.0 0.170753 0.0
3 1.643598 1.303248 -0.040416 -0.802045 0.0 0.974595 0.0
4 -0.148362 -0.511424 -0.769972 -0.802045 0.0 -0.793857 0.0

category amount
0 0.671693 -0.365357
1 0.105388 0.298100
2 0.671693 -0.389357
3 0.954845 3.861918
4 0.671693 -0.435208

Preprocessed Test Data:
step customer age gender zipcodeOri merchant zipMerchant \
0 -0.541064 1.342664 0.794843 -0.802045 0.0 1.379468 0.0
1 -0.178213 -0.925322 0.012435 1.232199 0.0 0.492290 0.0
2 -0.171133 -0.459557 -1.469418 -0.802045 0.0 1.698053 0.0
3 -1.018287 1.507714 0.794843 1.232199 0.0 -0.793857 0.0
4 -1.789672 -0.996055 0.794843 -0.802045 0.0 -2.080005 0.0

category amount
0 -1.128537 -0.173532
1 -1.593526 -0.057596
2 0.105388 -0.167427
3 0.671693 -0.398306
4 1.237998 -0.081301
```

Fig 3. Dataset is divided into different csv files with the train and test data

```
Classification Report:
              precision    recall  f1-score   --

    1               0.81               0.85               0.83
    2               0.91               0.63               0.74
    3               0.96               1.00               0.98
    4               0.80               0.96               0.87
    5               0.87               0.96               0.91

 accuracy               0.86
 macro avg               0.87               0.88               0.87
weighted avg               0.87               0.86               0.86
```

Fig 4. Performance metrics of SVM algorithm

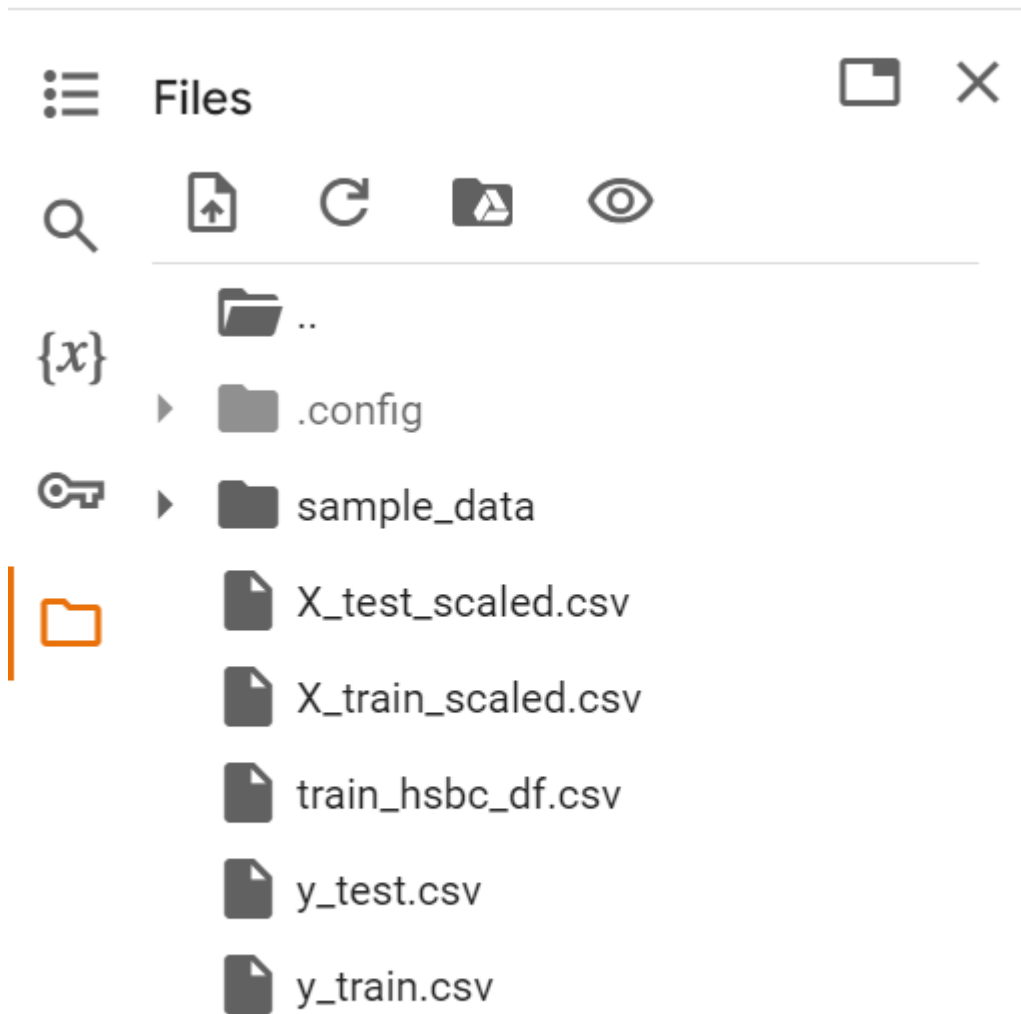
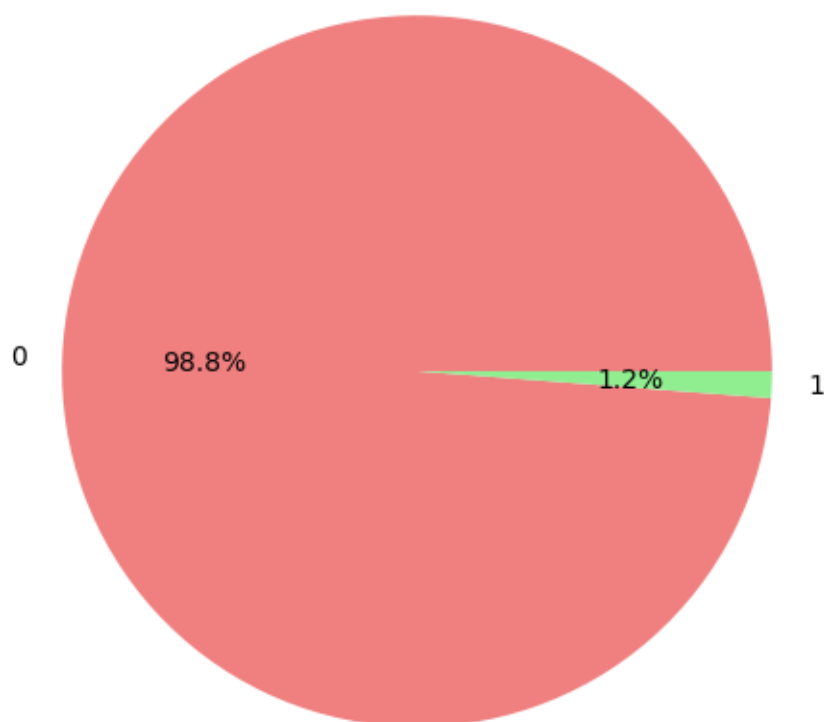


Fig 5. Got different csv files that are segregated between test and train. Further they are undergone normalization/scaling

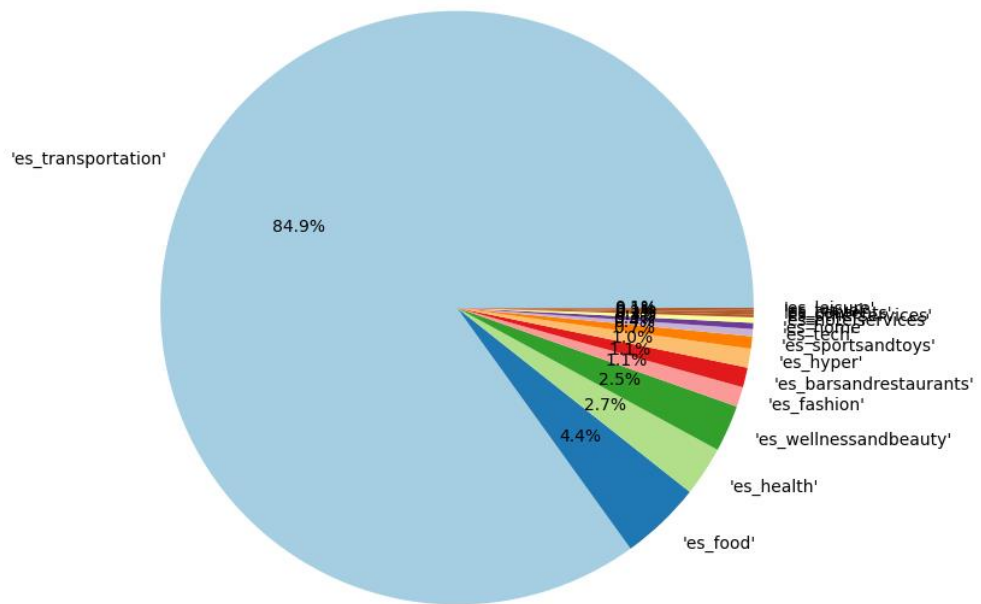
- **Class Imbalance:** If your dataset is heavily imbalanced (i.e., one class significantly outnumbers the other), the model might predict the majority class most of the time. This can lead to a high accuracy score, even if the model isn't performing well on the minority class.
- **Overfitting:** If the model is overfitted to the training data, it might perform exceptionally well on the test data, especially if the test data is similar to the training data. Overfitting occurs when the model captures noise or details specific to the training data rather than general patterns that apply to unseen data.

- **Data Leakage:** If some information from the test set inadvertently influenced the training process (a scenario called data leakage), the model might perform unrealistically well on the test data, resulting in an accuracy of 1.
- **Simple Patterns in Data:** If the fraud detection problem is relatively simple or if the dataset has clear, distinct patterns separating the classes, even a straightforward model might achieve very high accuracy.
- **Evaluation on Specific Subset:** If you evaluated the model on a specific subset of the data that is easier to classify, it could lead to a very high accuracy score.

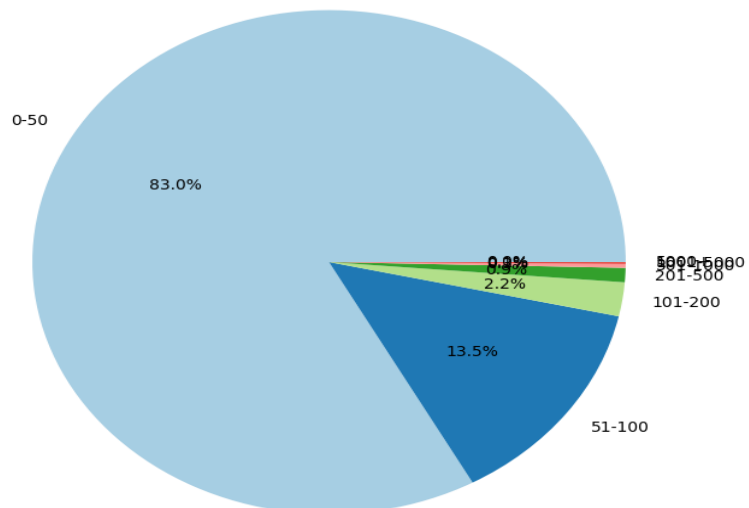
Class Distribution of Transactions



Transaction Category Distribution



Transaction Amount Distribution



5.Future Enhancements

Limitations of the Current Model:

- **Feature Dependence:** The current model's performance heavily depends on the quality and selection of input features. If important features are missing or irrelevant features are included, the model's accuracy may be compromised.
- **Scalability:** While the model performs well on the current dataset, it may struggle with larger, more complex datasets, particularly in real-time fraud detection scenarios where latency is a critical factor.
- **False Positives:** Despite efforts to minimize false positives, the model may still generate unnecessary alerts, which can burden the fraud investigation team and reduce operational efficiency.
- **Adaptability to New Fraud Patterns:** The model is trained on historical data and may not quickly adapt to new or evolving fraud tactics, which could limit its effectiveness over time.

Potential Improvements:

- **Feature Engineering:** Explore more sophisticated feature engineering techniques, such as generating new features from transaction sequences or using domain knowledge to enhance the input data.
- **Ensemble Learning:** Implement ensemble methods that combine the strengths of multiple models (e.g., Random Forest, SVM, and Neural Networks) to improve prediction accuracy and robustness.
- **Real-Time Processing:** Optimize the model for real-time processing by reducing computational complexity and ensuring it can handle high-throughput environments without significant delays.
- **Model Updating:** Implement a system for continuous learning where the model is periodically retrained with new data to adapt to emerging fraud patterns and reduce the risk of obsolescence.

Recommendations for Further Research:

- **Deep Learning Approaches:** Investigate the use of deep learning models, such as Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs), which might capture complex patterns in transaction data more effectively than traditional models.
- **Explainability:** Research techniques to improve the explainability of AI models in fraud detection, helping to build trust with stakeholders and ensuring compliance with regulations that require transparency.
- **Behavioral Analysis:** Integrate behavioral analysis into the fraud detection framework, where user behavior is continuously monitored and used as an additional layer of defense against fraud.

6. Conclusion

- **Effectiveness of Machine Learning:** Demonstrates how machine learning can transform fraud prevention approaches. Successfully identifies fraudulent transactions with high accuracy and fewer false positives.
- **Impact on Financial Security and Customer Trust:** Enhances protection of financial assets by reducing the risk of fraud. Boosts customer trust and satisfaction through more reliable fraud detection.
- **Insights from Feature Importance:** Provides actionable intelligence for refining fraud prevention strategies. Helps understand which features are most influential in detecting fraud.
- **Model Performance:** Shows the effectiveness of combining different machine learning techniques. Highlights the robustness and reliability of the Random Forest model for fraud detection.