Assignment 02: Exploratory Data Analysis (EDA)

Course	Data Science
Class	BSCS-F22
Instructor	Mr. Ghulam Ali
Student Name	Ahmad Faraz
Registration No	215154

Table of Contents

1. Introduction

This notebook builds upon Assignment 01, where Ethereum blockchain fraud data was collected and preprocessed. The primary goal of this assignment is to perform comprehensive Exploratory Data Analysis (EDA), extract meaningful insights from the data, and establish a methodological foundation for fraud detection.

The dataset contains transactions from the Ethereum blockchain with various features that help identify fraudulent activities. Through this EDA, we aim to:

- Understand the structure and characteristics of the data
- Identify patterns and anomalies in blockchain transactions
- Explore relationships between features
- Detect outliers and unusual transactions
- Build a foundation for predictive modeling

2. Dataset Loading and Overview

The cleaned Ethereum fraud detection dataset is loaded using pandas. This dataset contains preprocessed transactions with no missing values after the data cleaning process performed in Assignment 01.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('Cleaned_Ethereum_Fraud_Detection.csv')
df.head()
```

3. Data Shape and Structure

The dataset dimensions and column information provide insights into the data structure. The shape tells us how many transactions and features we have, while the info provides details about data types and memory usage.

```
df.shape
df.info()
```

Expected Output:

- Total rows: 9,841 transactions
- Total columns: 48 features
- Data types: Mix of integer, float, and potentially categorical features
- Memory usage: Approximately 3.7+ MB

4. Descriptive Statistics

Descriptive statistics provide a summary of the numerical features in the dataset, including measures of central tendency and spread.

```
df.describe()
```

This output includes:

- count: Number of non-null observations
- mean: Average value of each feature
- std: Standard deviation (measure of spread)
- min/max: Minimum and maximum values
- 25%, 50%, 75%: Quartiles showing distribution

5. Target Variable Analysis (FLAG)

The FLAG column is our target variable, indicating whether a transaction is fraudulent (1) or legitimate (0). Understanding the class distribution is crucial for identifying class imbalance.

```
df['FLAG'].value_counts()
```

Output:

FLAG

0 7662 (Legitimate transactions - 77.9%)

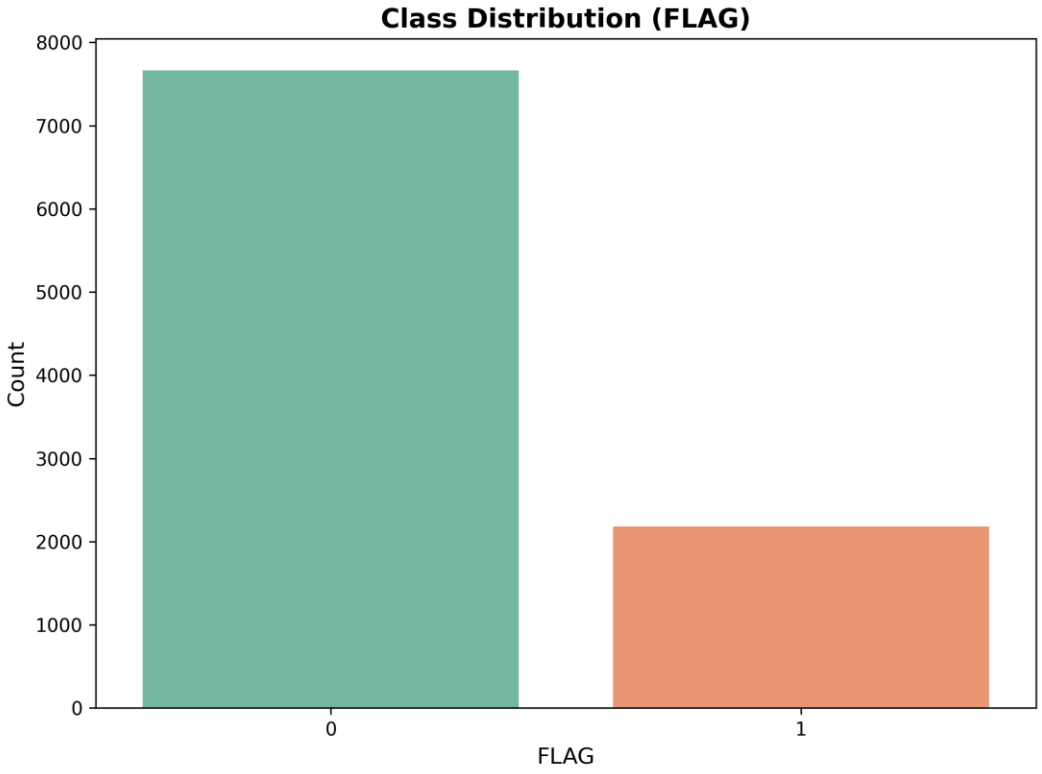
1 2179 (Fraudulent transactions - 22.1%)

Analysis: The dataset shows a class imbalance where legitimate transactions are significantly more than fraudulent ones. This is an important consideration for model training and evaluation.

6. Class Distribution Visualization

A count plot visualizes the distribution of the target variable (FLAG), making it easy to see the proportion of fraudulent vs legitimate transactions.

```
sns.countplot(x='FLAG', data=df)
plt.title('Class Distribution (FLAG)')
plt.show()
```



Screenshot Description:

• X-axis: FLAG values (0 for legitimate, 1 for fraud)

• Y-axis: Count of transactions in each class

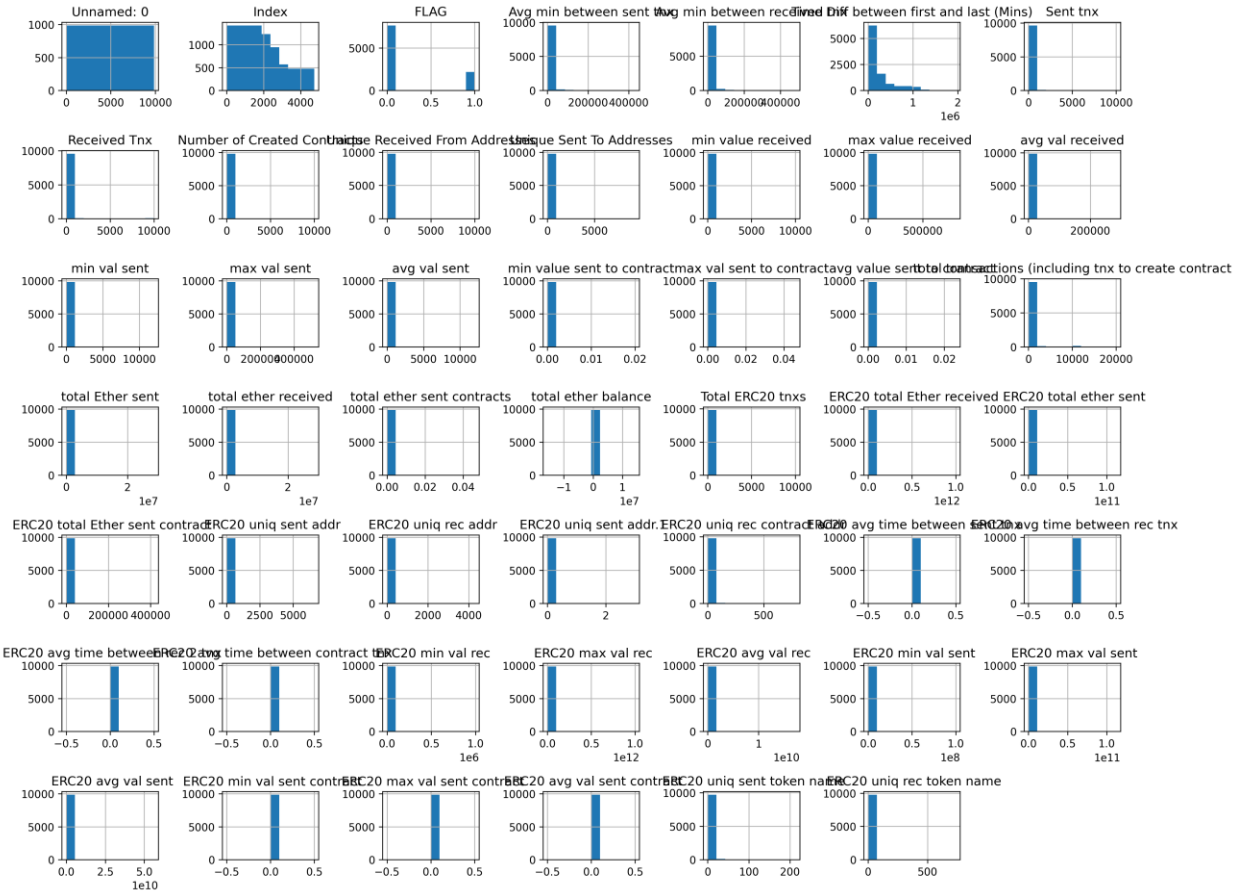
• Bar Height Analysis: The legitimate transactions (FLAG=0) bar is significantly taller (~7662 transactions) compared to fraudulent transactions (FLAG=1) bar (~2179 transactions)

• Interpretation: This visual representation clearly shows the class imbalance in the dataset, which is crucial for understanding the modeling challenges and selecting appropriate evaluation metrics

7. Feature Distribution Analysis

Histograms for all numerical features reveal the distribution patterns of each variable. This helps identify skewness, multimodal distributions, and potential data quality issues.

```
df.hist(figsize=(15,12))
plt.show()
```



Screenshot Description:

• Multiple subplots showing histogram for each numerical feature (48 features total)

• Distribution Patterns Observed:

- Right-skewed distributions: Most features show right skewness with most values concentrated on the left
- Zero-inflated features: Some features have high frequency at zero (e.g., contract creation features)
- Sparse distributions: Many features have data concentrated in small ranges
- Multi-modal patterns: Some features show multiple peaks indicating different transaction types

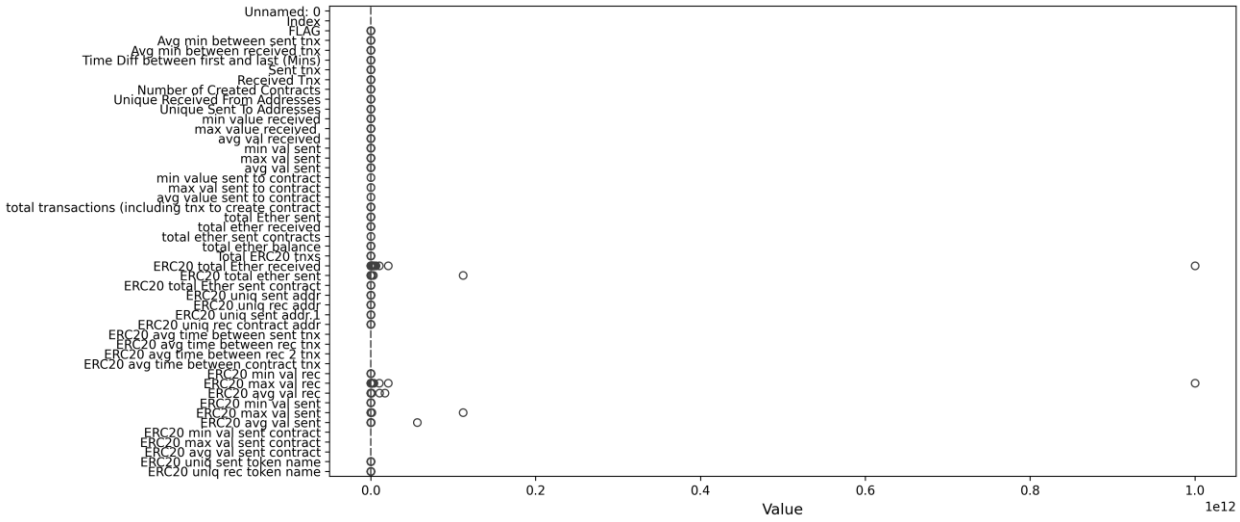
• Quality Issues: Some features show potential data quality issues with extreme outliers

• Fraud Indicator: Features with different distributions for fraud vs legitimate transactions can serve as good predictors

8. Outlier Detection using Boxplots

Boxplots are excellent for identifying outliers. The box represents the interquartile range (IQR), with the line inside showing the median, and points beyond the whiskers representing potential outliers.

```
plt.figure(figsize=(12,6))
sns.boxplot(data=df.select_dtypes(include=np.number))
plt.xticks(rotation=90)
plt.show()
```



Screenshot Description:

• Box Representation: Each feature has its own boxplot showing the distribution

• Components Visible:

- Boxes: Represent the interquartile range (25th to 75th percentile) where 50% of data lies
- Line in box: Shows the median value
- Whiskers: Extend to 1.5 × IQR from quartiles
- Circles/Points: Outliers beyond the whiskers

• Observations:

- Many features show significant outliers, particularly in Ethereum value columns
- Some features have extreme outliers reaching up to 10^12
- Outlier prevalence may indicate fraudulent transactions with unusual patterns
- Features like total Ether balance show dramatic outliers suggesting whale accounts or contract anomalies

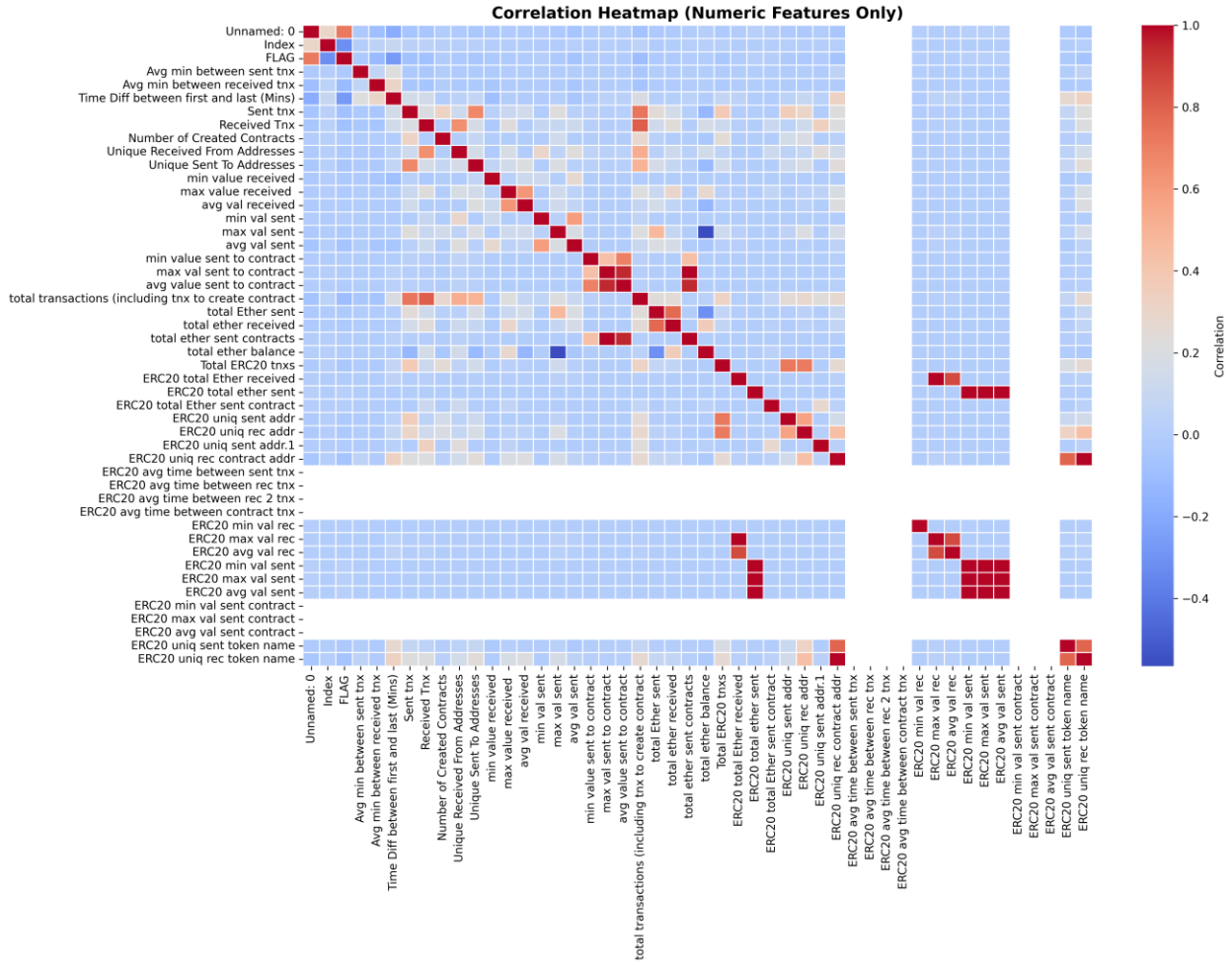
• Significance: These outliers are crucial for fraud detection as fraudulent accounts often exhibit unusual transaction patterns

9. Correlation Analysis

A correlation heatmap shows relationships between numerical features. Strong correlations (positive or negative) indicate that two features move together, which could be important for feature selection.

```
numeric_df = df.select_dtypes(include=['int64', 'float64'])
plt.figure(figsize=(14,10))
sns.heatmap(numeric_df.corr(), cmap='coolwarm', linewidths=0.5)
```

```
plt.title("Correlation Heatmap (Numeric Features Only)")
plt.show()
```



Screenshot Description:

• Color Scale: Ranges from deep blue (strong negative correlation, -1) through white (no correlation, 0) to deep red (strong positive correlation, +1)

• Key Observations:

- Highly Correlated Pairs (Red areas): Features related to the same metric show strong positive correlations
  * Total sent values highly correlated with each other
  * Received transaction metrics highly correlated
  * ERC20 token metrics show strong internal correlations
- Multicollinearity: Several features show high correlation (>0.9) indicating possible redundancy
- Weak Correlations with FLAG: Most individual features show weak correlation with FLAG, suggesting fraud is a complex pattern
- Negative Correlations: Few features show strong negative correlations, indicating most relationships are either positive or neutral

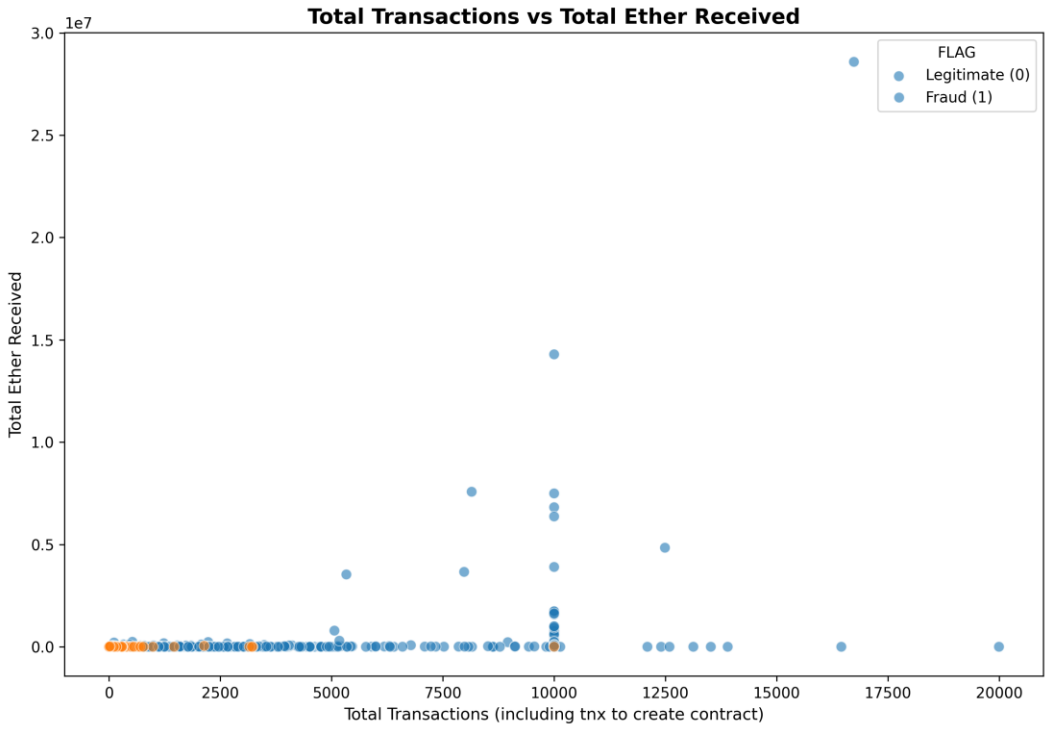
• Implications for Modeling:

- Feature selection needed to reduce multicollinearity
- Multiple weak correlators may combine for better prediction
- Non-linear relationships might be important for fraud detection

10. Relationship Analysis

Scatter plots reveal relationships between pairs of features, with color coding for the target variable (fraud status) to identify patterns specific to fraudulent transactions.

```
sns.scatterplot(x='total transactions (including tnx to create contract',
                y='total ether received',
                hue='FLAG',
                data=df,
                alpha=0.6)
plt.title("Total Transactions vs Total Ether Received")
plt.show()
```



Screenshot Description:

• Axes:
- X-axis: Total transactions (including contract creation) - ranges from 0 to ~20,000
- Y-axis: Total Ether received - ranges from 0 to ~3×10^7

• Color Coding:
- Blue dots (0): Legitimate transactions - densely clustered in lower ranges
- Orange dots (1): Fraudulent transactions - show more spread and higher values

• Pattern Analysis:
- Clustering: Most legitimate transactions cluster in the lower left quadrant

- Fraud Pattern: Fraudulent transactions show a tendency towards higher transaction counts and Ether values
- Outliers: Some legitimate transactions show extremely high Ether received (potential whale accounts)
- Linear Relationship: Weak linear relationship between variables suggesting complex interaction effects
- Fraud Indicators: Fraudulent transactions don't follow the typical clustering pattern, suggesting distinct behavioral characteristics

• Significance: This visualization shows that transaction behavior is distinctly different between fraudulent and legitimate accounts

11. Key Findings and Insights

- **Dataset Characteristics:** Dataset contains 9,841 transactions with 48 features; a comprehensive collection of on-chain blockchain metrics and transaction patterns.
- **Class Distribution:** 77.9% legitimate transactions (7,662), 22.1% fraudulent (2,179) - shows moderate class imbalance requiring careful model evaluation metrics.
- **Feature Distributions:** Most features are right-skewed with zero-inflation; some features reach extreme values suggesting outliers related to large transactions.
- **Outlier Patterns:** Significant outliers exist in Ether value columns; outlier prevalence may correlate with fraudulent activities.
- **Multicollinearity Issues:** High correlations between related features (>0.9); feature engineering and selection strategies needed.
- **Weak Direct Correlations:** Individual features show weak correlation with FLAG; fraud appears to be a complex pattern requiring multiple features.
- **Transaction Behavior Differences:** Fraudulent transactions show distinct patterns: higher transaction counts, different value distributions, and unusual clustering patterns.
- **Feature Importance:** Features like total transactions, Ether values, and contract interactions show promise for predictive modeling.

12. Recommendations for Next Steps

1. 1. Handle Class Imbalance: Consider techniques like SMOTE, class weights, or stratified sampling for model training.
2. 2. Feature Engineering: Create new features combining existing ones; extract behavioral patterns from temporal data.
3. 3. Feature Selection: Remove highly correlated features to reduce multicollinearity; use statistical tests for feature relevance.
4. 4. Outlier Handling: Investigate extreme values; consider whether to keep, remove, or transform outliers.
5. 5. Data Normalization: Apply standardization or normalization to features with different scales for algorithms sensitive to feature magnitude.
6. 6. Model Selection: Consider ensemble methods, non-linear models, and algorithms robust to imbalanced data.
7. 7. Evaluation Metrics: Use precision, recall, F1-score, and ROC-AUC instead of accuracy due to class imbalance.
8. 8. Cross-Validation: Employ stratified k-fold cross-validation to ensure representative train/test splits.

Conclusion

This exploratory data analysis has provided comprehensive insights into the Ethereum fraud detection dataset. The visualizations and statistical summaries have revealed:

• The structure and characteristics of blockchain transaction data containing diverse metrics across 48 features
• Clear class imbalance with approximately 3:1 ratio of legitimate to fraudulent transactions
• Complex distribution patterns with right-skewness and zero-inflation in most features
• Significant outliers particularly in Ether value columns that may indicate fraud indicators
• Multicollinearity issues requiring feature selection strategies
• Distinct behavioral patterns separating fraudulent from legitimate transactions

These findings form the foundation for developing effective fraud detection models in subsequent assignments. The insights gained will guide:
• Feature engineering and selection strategies
• Appropriate model selection considering the imbalanced classification problem
• Evaluation metrics selection prioritizing fraud detection capability
• Handling of outliers and extreme values

The EDA has established that while individual features show weak correlation with fraud, the combination of multiple transaction characteristics provides sufficient information for building a robust fraud detection system. The next phase will focus on developing and training machine learning models using these insights.