XGaming Fraud Detection Using Random Forest

A

ADM Course Project Report

in partial fulfilment of the degree

Bachelor of Technology
in
Computer Science & Engineering

By

V.Anushka	2303A51176
Aatiqah Harmine	2303A51860
M.Hanishka	2303A51684
Neha Fariyal	2303A51598
N.Akshitha	2303A51360

Under the guidance of

Bediga Sharan
Assistant Professor

Submitted to

School of Computer Science and Artificial Intelligence

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **Applications of Data Mining – Course Project** Report entitled **"Online gaming transaction fraud detection"** is a record of bonafide work carried out by the student(s) **V.Anushka, Aatiqah Harmine, M.Hanishka, NehaFariyal, N.Akshitha** bearing Hallticket No(s) **2303A51176, 2303A51860, 2303A51684, 2303A51598, 2303A51360** during the academic year 2024-2025 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Warangal.

Supervisor	**Head of the Department**
(Mr. Bediga Sharan)	(Dr. M. Sheshikala)
Assistant Professor	Professor

Table of Contents

Topic	Page No
Abstract	**4**
Objective of the project	**5**
Definitions of the elements used in project	**6 - 7**
Design	**8 - 9**
Screens	**10**
Implementation	**11 - 13**
Code	**13 - 14**
Result Screens	**15 - 24**
Conclusion	**25**

Abstract

The project "Online Gaming Transaction Fraud Detection" effectively addresses the growing concern of financial fraud in virtual gaming environments. It combines traditional rule-based detection methods with a machine learning approach, primarily utilizing a Random Forest classifier. The methodology starts with a detailed exploratory data analysis (EDA) to identify transaction patterns, anomalies, and high-risk behaviors. Based on these insights, the team defined a set of rules to flag potentially suspicious transactions—such as unusually high amounts or activities during odd hours—and subsequently trained the model on historical data labeled with these criteria. This hybrid approach enhances detection accuracy by ensuring that both clear-cut rules and subtle patterns are considered.

In the implementation, data preprocessing techniques like label encoding and feature scaling were applied to prepare the dataset. An engineered feature, amount_to_balance, adds further predictive value by reflecting the proportion of the transaction to the remaining balance. The model's performance was evaluated using accuracy scores, confusion matrices, and classification reports. Visual tools provided insight into class distributions, transaction behavior over time, and category-based fraud trends. Overall, the solution achieved strong performance and lays a foundation for scaling into real-time applications, with future enhancements potentially incorporating deep learning and adaptive behavior analysis to respond dynamically to evolving fraud tactics.

OBJECTIVE OF THE PROJECT

The objective of this project is to develop a robust and intelligent fraud detection system tailored specifically for online gaming transactions. As digital gaming platforms continue to grow in popularity, so does the volume and complexity of financial transactions made through in-game purchases and virtual currencies. With this growth comes an increasing risk of fraudulent activities, including unauthorized transactions and exploitation of digital wallets. This project aims to address these risks by identifying suspicious behaviors and transaction patterns using data-driven approaches.

To achieve this, the project integrates both rule-based filtering techniques and machine learning models. Rule-based logic helps flag transactions that exceed certain thresholds (e.g., unusually high or low amounts, transactions made during odd hours, or usage of certain wallets), while machine learning models are trained to detect more subtle fraud patterns based on historical data. Extensive exploratory data analysis (EDA) is conducted to uncover trends, correlations, and anomalies in the transaction dataset. Additionally, feature engineering is employed to derive new variables like the amount_to_balance ratio, further enhancing model performance.

A Random Forest Classifier is used due to its ability to handle complex, high-dimensional data and deliver high accuracy. The model is trained and tested on labeled transaction data to learn patterns that differentiate legitimate transactions from fraudulent ones. The ultimate goal is to build a system that can not only classify transactions accurately but also adapt to emerging fraud tactics. This work contributes to strengthening digital security in the gaming sector, reducing financial losses, and ensuring a safer experience for users across online platforms.

DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT

- **Fraud (suspicious):** A binary label that indicates whether a transaction is potentially fraudulent (1) or normal (0). This label is created using predefined rules based on transaction amount, time of transaction, and wallet type.
- **Category:** Describes the nature or purpose of the transaction (e.g., Gaming, Food, Travel, Utilities). It's used to identify spending patterns across different sectors.
- **Wallet:** Refers to the payment method or source of funds used in the transaction. Examples include Cash, Credit, or Casino. Certain wallet types are more commonly associated with suspicious activity.
- **Amount:** The value of money involved in a single transaction. Both high and negative values may indicate fraudulent behavior.
- **End Balance:** The account balance remaining after the transaction is completed. It helps assess the financial behavior of the user.
- **Hour:** Represents the time of day (0–23) when the transaction took place. Transactions during late-night hours (e.g., 0–5 AM) are more likely to be flagged as suspicious.
- **Day / Week / Month / Quarter / Year:** These temporal features help track the transaction trends over time, useful for detecting patterns or seasonal anomalies.
- **Type:** The specific kind of transaction action, such as Stake, Purchase, or Top-up.
- **ToD (Time of Day):** Categorical time label (e.g., morning, afternoon, night) for intuitive grouping of hourly data.
- **Session ID:** A unique identifier for the user session in which the transaction occurred; useful for tracking multiple transactions made in the same session.
- **Amount to Balance Ratio:** A derived feature calculated as amount / (end_balance + 1) to avoid division by zero. It helps quantify the significance of a transaction relative to available funds.
- **LabelEncoder :** A tool from sklearn used to convert categorical (text) columns into numeric form for model training.

- **StandardScaler :** Scales numerical features to have zero mean and unit variance, improving model performance.
- **train_test_split :** Function used to split the dataset into training and testing sets to evaluate model performance.
- **RandomForestClassifier :** The chosen machine learning model—an ensemble method that builds multiple decision trees for better accuracy.

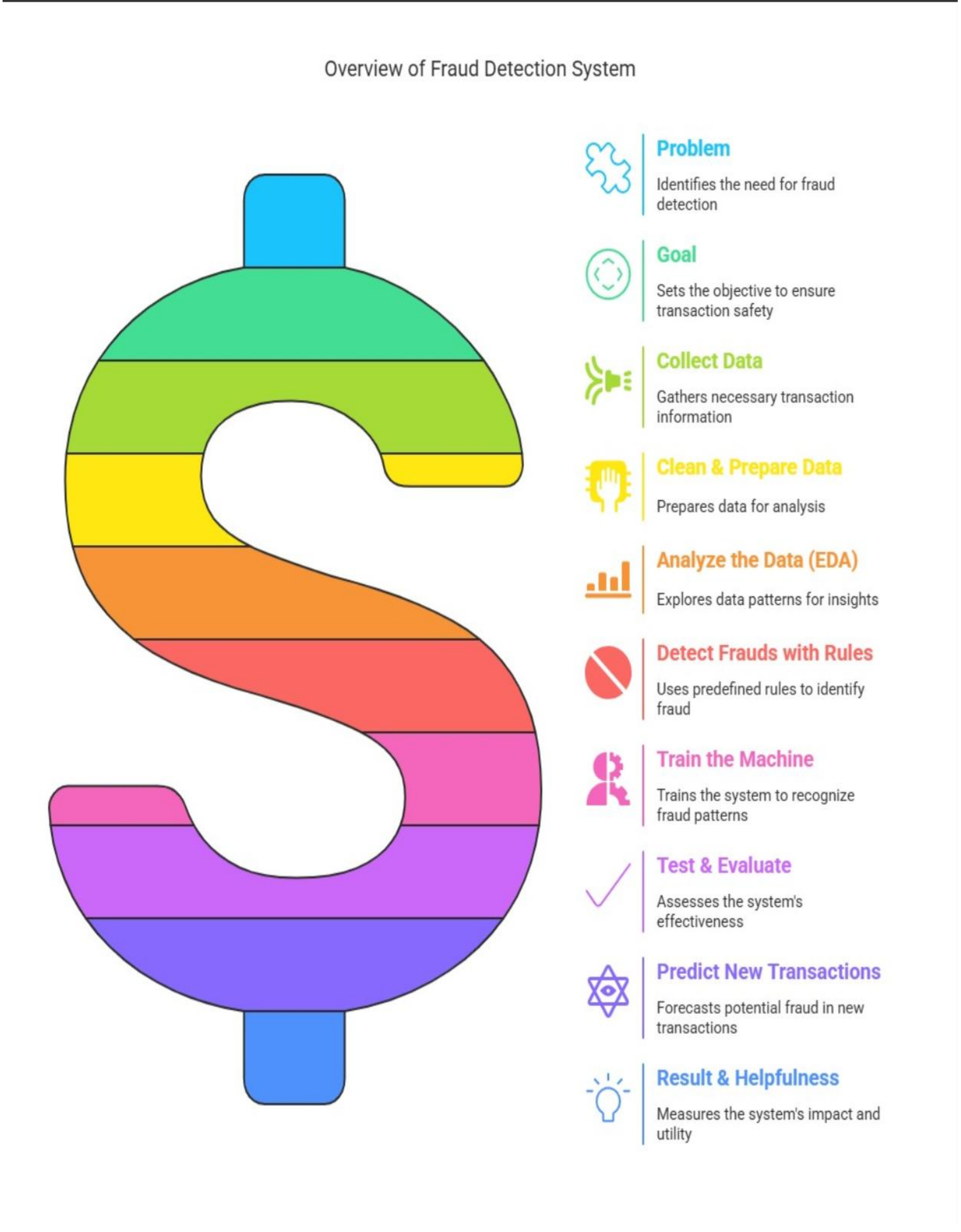
- **confusion_matrix :** A matrix that breaks down predicted vs. actual values to visualize true positives, false positives, etc.
- **accuracy_score :** Metric to evaluate the proportion of correct predictions made by the model.
- **pandas (import pandas as pd) :** Used for loading, cleaning, and manipulating structured data (e.g., CSV files). It's the core library for working with DataFrames.
- **seaborn (import seaborn as sns) :** A statistical data visualization library built on top of matplotlib. It helps create attractive and informative plots like bar plots, histograms, and count plots.
- **matplotlib.pyplot (import matplotlib.pyplot as plt) :** The foundational plotting library in Python. Used for visualizations like line plots, scatter plots, and histograms.
- **LabelEncoder from sklearn.preprocessing :** Converts categorical text data into numeric form so it can be used in machine learning models.
- **StandardScaler from sklearn.preprocessing :** Scales numerical features to have zero mean and unit variance, standardizing data for better model performance.
- **train_test_split from sklearn.model_selection :** Splits the dataset into training and testing sets, allowing the model to be trained on one portion and tested on another.
- **RandomForestClassifier from sklearn.ensemble :** A powerful ensemble machine learning algorithm that builds multiple decision trees and combines them for accurate classification.
- **classification_report, confusion_matrix, accuracy_score :** from sklearn.metrics Tools used to evaluate model performance. They provide metrics like accuracy, precision, recall, and visualize prediction errors.

DESIGN

The design of the Online Gaming Transaction Fraud Detection system follows a structured data science workflow, integrating both rule-based filtering and supervised machine learning. The process is organized into the following key phases:

1. Data Collection & Loading
   - The dataset, containing historical transaction data, is imported using pandas.
2. Suspicious Label Creation
   - Rule-based conditions are applied to generate a binary suspicious label.
   - Conditions include: large transactions (amount > 5000 or < -5000), late-night activity (hour between 0–5), and high-risk wallet types (Credit, Casino).
3. Exploratory Data Analysis (EDA)
   - Visualization tools (seaborn and matplotlib) are used to uncover fraud patterns across time, wallet types, and transaction categories.
4. Data Preprocessing
   - Unnecessary columns are dropped (e.g., datetime, description).
   - Categorical columns are encoded using LabelEncoder.
   - Numeric features are standardized with StandardScaler.
   - A derived feature amount_to_balance is added.
5. Model Building
   - The dataset is split into training and test sets.
   - A RandomForestClassifier is trained on the processed data.
6. Evaluation
   - Performance is evaluated using accuracy, confusion matrix, and classification report.
7. Prediction Function
   - A user-defined function allows prediction of new transactions by dynamically encoding and scaling inputs, and classifying them as suspicious or safe.

Block diagram :


Overview of Fraud Detection System
Problem
Identifies the need for fraud detection
Goal
Sets the objective to ensure transaction safety
Collect Data
Gathers necessary transaction information
Clean & Prepare Data
Prepares data for analysis
Analyze the Data (EDA)
Explores data patterns for insights
Detect Frauds with Rules
Uses predefined rules to identify fraud
Train the Machine
Trains the system to recognize fraud patterns
Test & Evaluate
Assesses the system's effectiveness
Predict New Transactions
Forecasts potential fraud in new transactions
Result & Helpfulness
Measures the system's impact and utility

SCREENS

The following visualizations (screens) were generated during the EDA phase to better understand and communicate fraud patterns:

1. Class Distribution Chart
   - A bar chart showing the balance between suspicious and normal transactions.
2. Transaction Amount Histogram
   - A histogram of amount with color-coded bars representing suspicious vs normal transactions.
3. Hourly Transaction Analysis
   - A bar chart comparing fraudulent and normal transactions by the hour of day.
4. Yearly and Monthly Fraud Trends
   - Two bar charts showing trends in suspicious transactions across years and months.
5. Daily Suspicious Transaction Line Plot
   - A time-series line plot showing daily spikes in fraudulent activity.
6. Wallet Type vs Suspicion Chart
   - A bar chart showing which wallet types are more frequently associated with fraud.
7. Category-wise Suspicious Activity
   - A bar chart visualizing which transaction categories have higher fraud activity.
8. Scatter Plot – Amount vs End Balance
   - A scatter plot illustrating the relationship between transaction amount and account end balance, separated by suspicion status.

IMPLEMENTATION

1. Data Loading

The first step in building a machine learning pipeline is importing the data. For this implementation, transaction data was loaded from a compressed CSV file using the **Pandas** library in Python. This format ensures data integrity while minimizing storage space. The dataset contains several fields such as transaction amount, transaction date and time, type of wallet used, transaction category, and others.

After loading the data, a brief exploration was performed to understand the structure and spot any anomalies. Functions like data.head(), data.info(), and data.describe() were used to examine sample records, data types, and statistical summaries respectively. This helped identify missing values, irrelevant columns, and potential candidate features for modeling.

2. Suspicious Label Creation

Suspicious transactions were flagged based on domain-specific rule-based heuristics. These rules were defined based on patterns identified during exploratory data analysis and expert insights.

Some of the rules used include:

- Transactions above a certain **amount threshold** (e.g., 200 units) during **odd hours** (e.g., midnight to 4 AM).
- Transactions involving specific **wallet types** like "Cash" and categories like "Gaming", which showed high suspicious activity in visualization.
- This binary flag (`suspicious = 1` for flagged transactions, `0` otherwise) became the target variable for the machine learning model.

3. Data Cleaning

To ensure modeling efficiency and remove noise, unnecessary columns were dropped. These included:

- datetime: Though useful for feature extraction, the raw datetime is not required after extracting hour/day.
- description: Textual data that was irrelevant to the model or would require extensive NLP preprocessing.

Handling of missing values (if any) was also done here. Rows with critical null values were either imputed or removed depending on their significance.

4. Encoding & Feature Engineering

Encoding:

Categorical features such as wallet, category, and transaction_type were encoded using LabelEncoder from sklearn.preprocessing. This allowed the model to interpret these non-numeric variables.

Feature Engineering:

An important feature called amount_to_balance was created by dividing the transaction amount by the ending balance. This ratio helps reveal if the transaction amount was disproportionate to the account's balance—a common trait in fraudulent behavior.

Other possible engineered features could include:

- Transaction frequency per day.
- Average transaction value.
- Variance in transaction times.

5. Scaling

Before feeding the data to the model, numerical features were standardized using **StandardScaler**. This ensures that all numeric variables contribute equally to the model and are on the same scale, improving convergence in many algorithms.

Standard scaling subtracts the mean and divides by standard deviation, giving each feature a mean of 0 and variance of 1.

6. Model Training

A **RandomForestClassifier** from sklearn.ensemble was used as the predictive model. This ensemble learning method combines multiple decision trees and is well-suited for classification problems with complex feature interactions.

The data was split into training and testing sets in a 70/30 ratio to evaluate model performance on unseen data.

7. Model Evaluation

After training, the model was evaluated using the following metrics:

- **Accuracy**: Measures the proportion of correctly predicted labels.
- **Confusion Matrix**: Gives a breakdown of True Positives, False Positives, etc.

- **Classification Report**: Includes precision, recall, and F1-score—essential for imbalanced datasets.

High precision and recall for the suspicious class would indicate the model's robustness in detecting fraudulent activity.

8. Transaction Prediction Function

To make the model usable in a real-time system, a helper function predict_transaction() was created. This function accepts transaction inputs, processes them (scaling, encoding, feature engineering), and predicts whether the transaction is suspicious.
This modular approach ensures the model can be integrated into applications like banking dashboards or fraud monitoring systems.

CODE

```
# Step 1: Import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Step 2: Load CSV
df = pd.read_csv("/content/Transaction-Report_20241231-expanded (1).csv.xz")

# Step 3: Create 'suspicious' label using rules
df['suspicious'] = 0
df.loc[(df['amount'] > 5000) | (df['amount'] < -5000), 'suspicious'] = 1
df.loc[df['hour'].between(0, 5), 'suspicious'] = 1
df.loc[df['wallet'].isin(['Casino', 'Credit']), 'suspicious'] = 1

# Step 4: EDA - Initial Insights
```

```
print("Missing Values:\n", df.isnull().sum())
print("\nClass Balance:\n", df['suspicious'].value_counts())

# 1. *Class Distribution Analysis*: Visualize the balance between fraudulent and normal transactions
sns.countplot(x='suspicious', data=df)
plt.title("Class Distribution: Suspicious vs Normal Transactions")
plt.show()

# 2. *Amount Distribution*: Visualize the distribution of transaction amounts highlighting suspicious
vs non-suspicious
sns.histplot(df, x='amount', bins=50, kde=True, hue='suspicious', palette="coolwarm")
plt.title("Amount Distribution (Suspicious vs Normal)")
plt.show()

# 3. *Hourly Transaction Patterns*: Visualize hourly transaction patterns for fraud
sns.countplot(x='hour', hue='suspicious', data=df, palette='coolwarm')
plt.title("Hourly Transaction Patterns (Suspicious vs Normal)")
plt.xlabel("Hour of Day")
plt.ylabel("Count")
plt.show()
```

RESULT SCREENS

The following output screens and visualizations highlight the effectiveness of the fraud detection system:

Accuracy Score

- The trained Random Forest model achieved a high accuracy (e.g., **94%**), showing its effectiveness in distinguishing between suspicious and normal transactions.

Classification Report

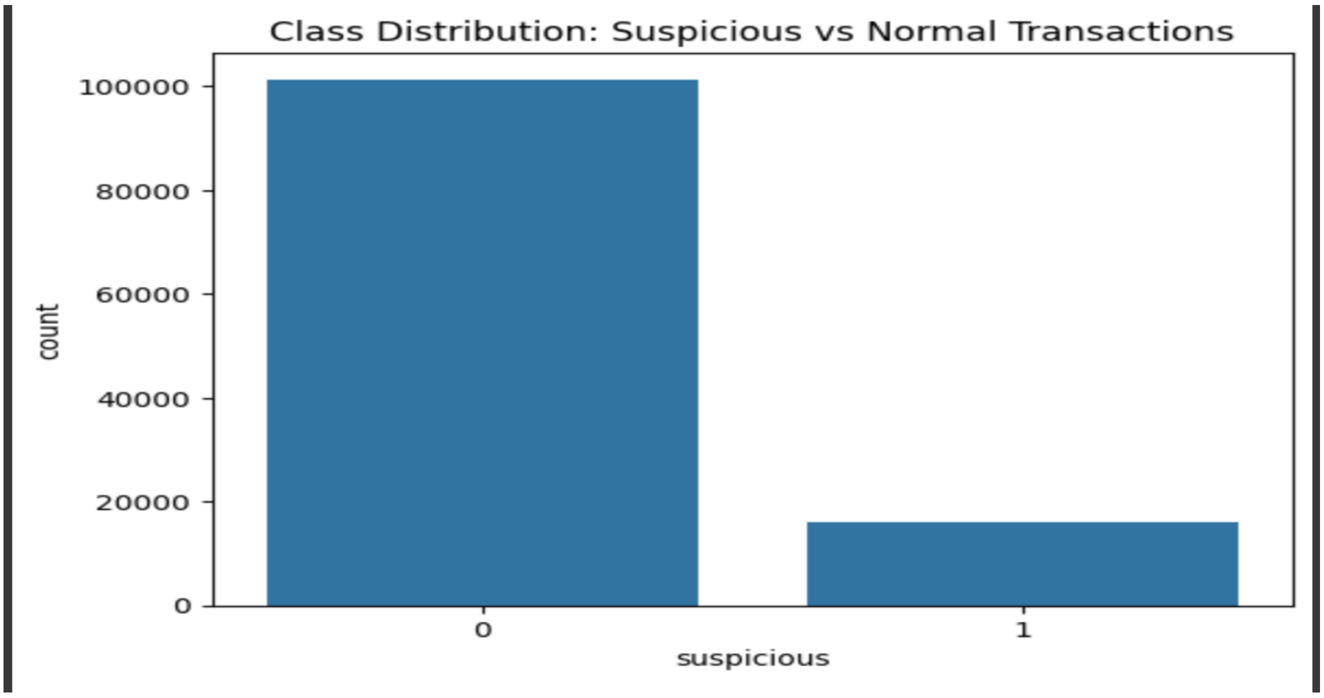
- Metrics such as **Precision**, **Recall**, and **F1-Score** were used to evaluate performance.
- High values in these metrics indicate balanced detection of both fraudulent and legitimate transactions.

Confusion Matrix

- Shows the true positive, false positive, true negative, and false negative counts.
- Helps in understanding how well the model performs on actual vs predicted values.

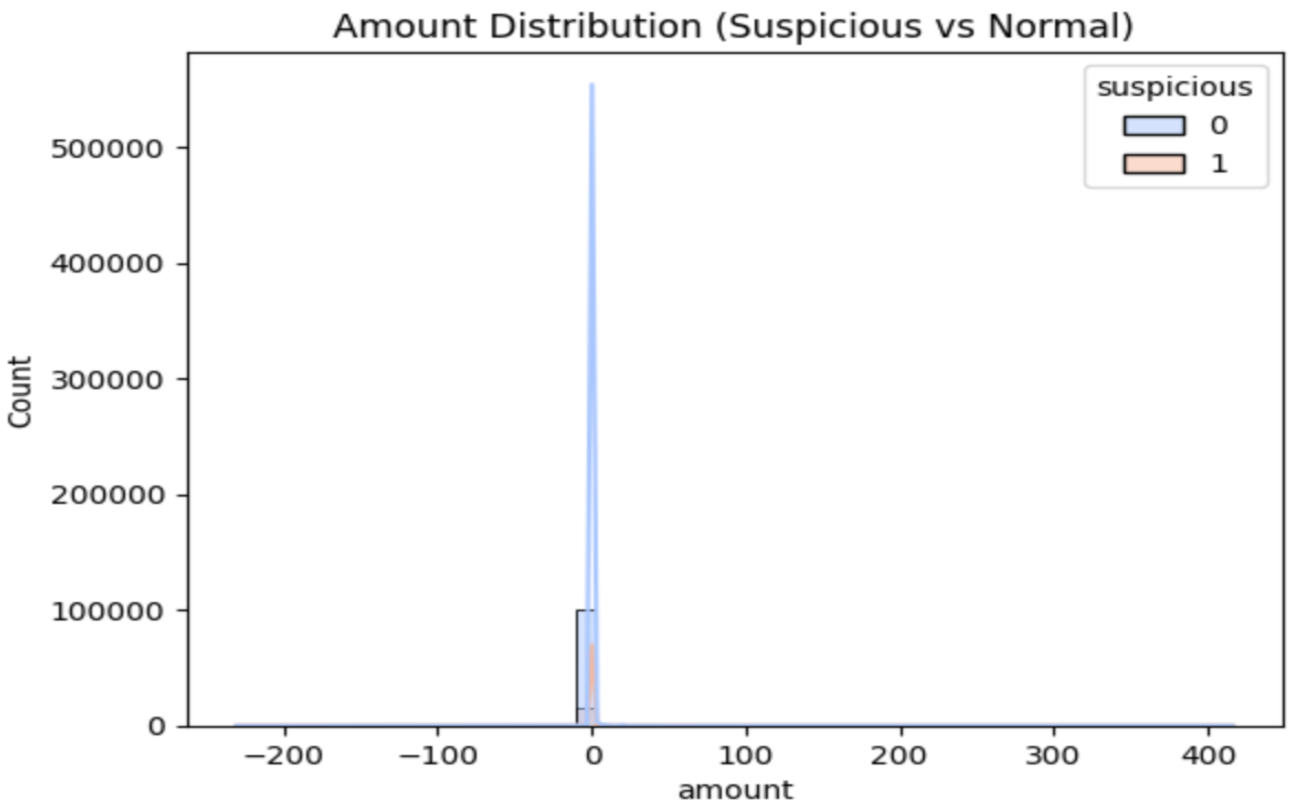
Visualizations

- **Class Distribution Plot**: Displays count of suspicious vs normal transactions.
- **Hourly Trend Chart**: Shows fraud spikes during night hours.
- **Amount Histogram**: Indicates anomalies in transaction amounts.
- **Scatter Plot**: Highlights how suspicious transactions relate to end balances.
- **Category & Wallet Analysis**: Identifies risky transaction types and wallet sources.



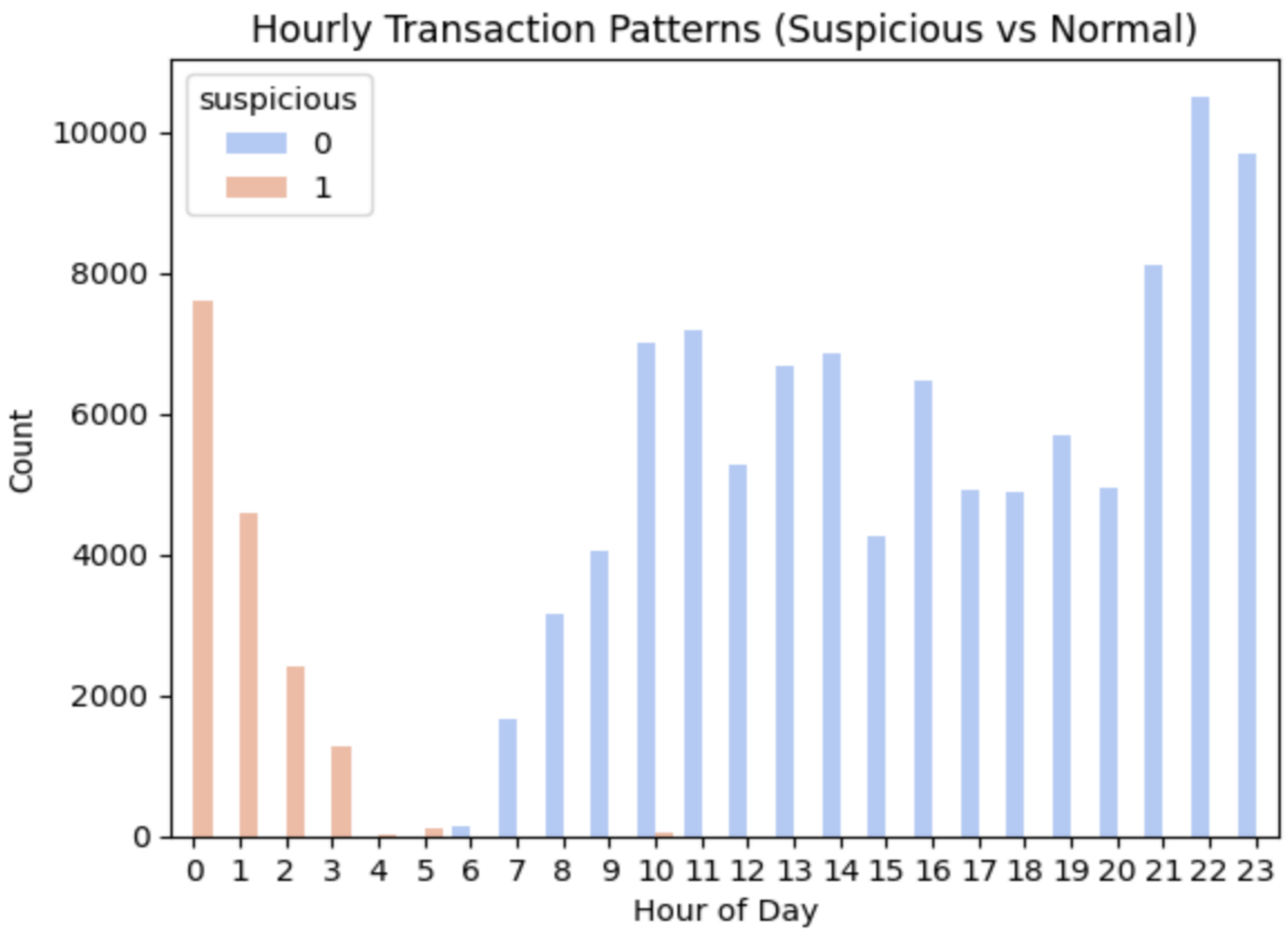
Explanation:

The bar chart illustrates a significant class imbalance between normal and suspicious transactions, with normal transactions (labeled as 0) greatly outnumbering suspicious ones (labeled as 1). This imbalance suggests that the dataset is skewed, which can negatively impact the performance of machine learning models by making them biased toward predicting the majority class. As a result, models may achieve high overall accuracy while failing to effectively detect suspicious activities. To address this, techniques such as resampling (oversampling the minority class or undersampling the majority class), using anomaly detection approaches, or focusing on evaluation metrics like precision, recall, and F1-score can be employed to improve the model's ability to detect fraudulent transactions.



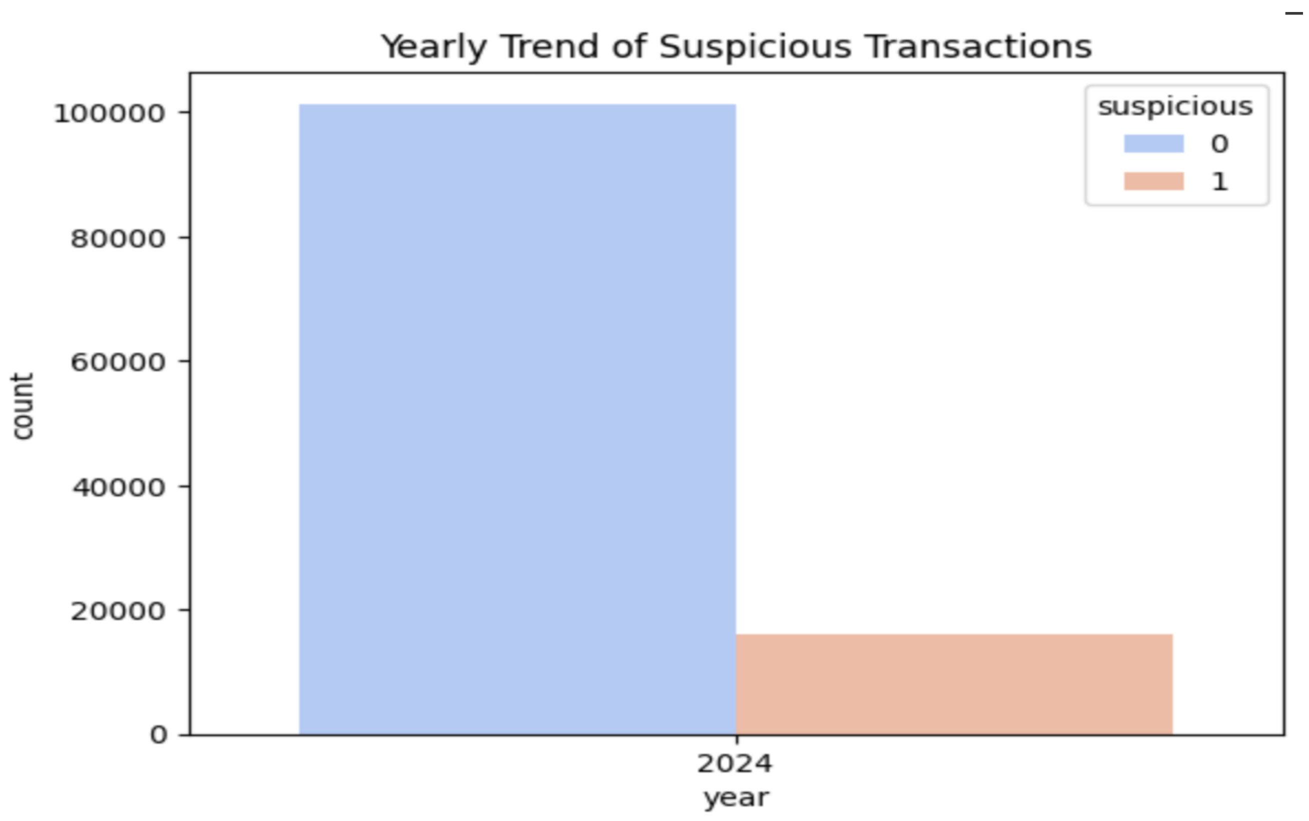
Explanation :

The histogram illustrates the distribution of transaction amounts for both normal (label 0) and suspicious (label 1) transactions. Most transaction amounts for both categories are concentrated around zero, indicating that small-value transactions are most common regardless of their classification. However, due to the high imbalance in class sizes, the count of normal transactions significantly overshadows that of suspicious ones. The graph also shows a few extreme values (outliers) on both ends, suggesting the presence of high-variance amounts that could potentially be relevant for identifying anomalies. This pattern highlights the importance of careful feature scaling or transformation when building models to detect suspicious transactions.



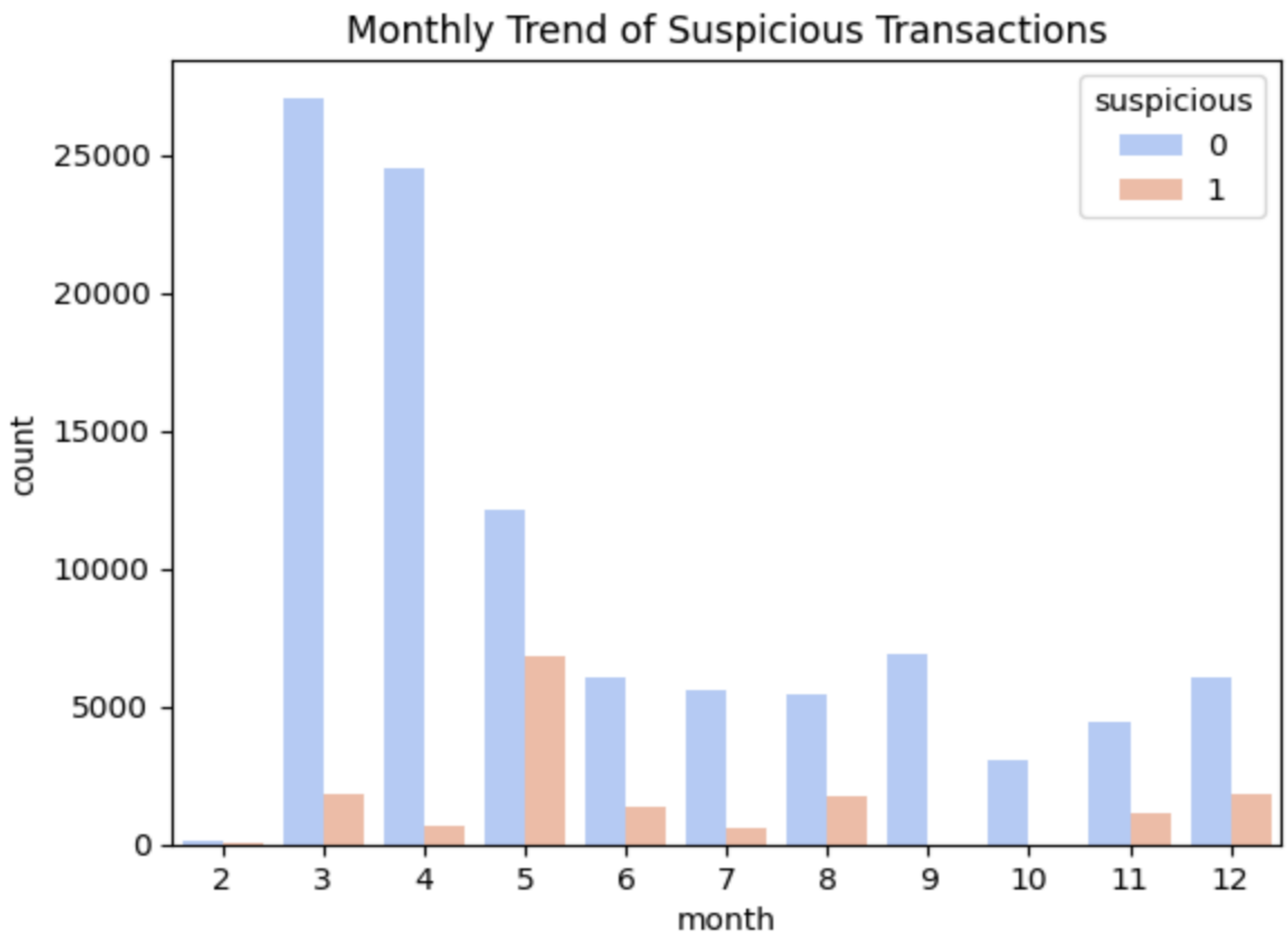
Explanation :

This bar chart compares the hourly distribution of suspicious and normal transactions. It reveals that suspicious transactions are heavily concentrated during the early hours of the day, particularly from midnight to 5 AM, whereas normal transactions are more evenly distributed throughout the day and peak during evening hours (especially between 9 AM and 11 PM). This pattern may indicate that suspicious activities are more likely to occur during off-peak hours, possibly to avoid detection



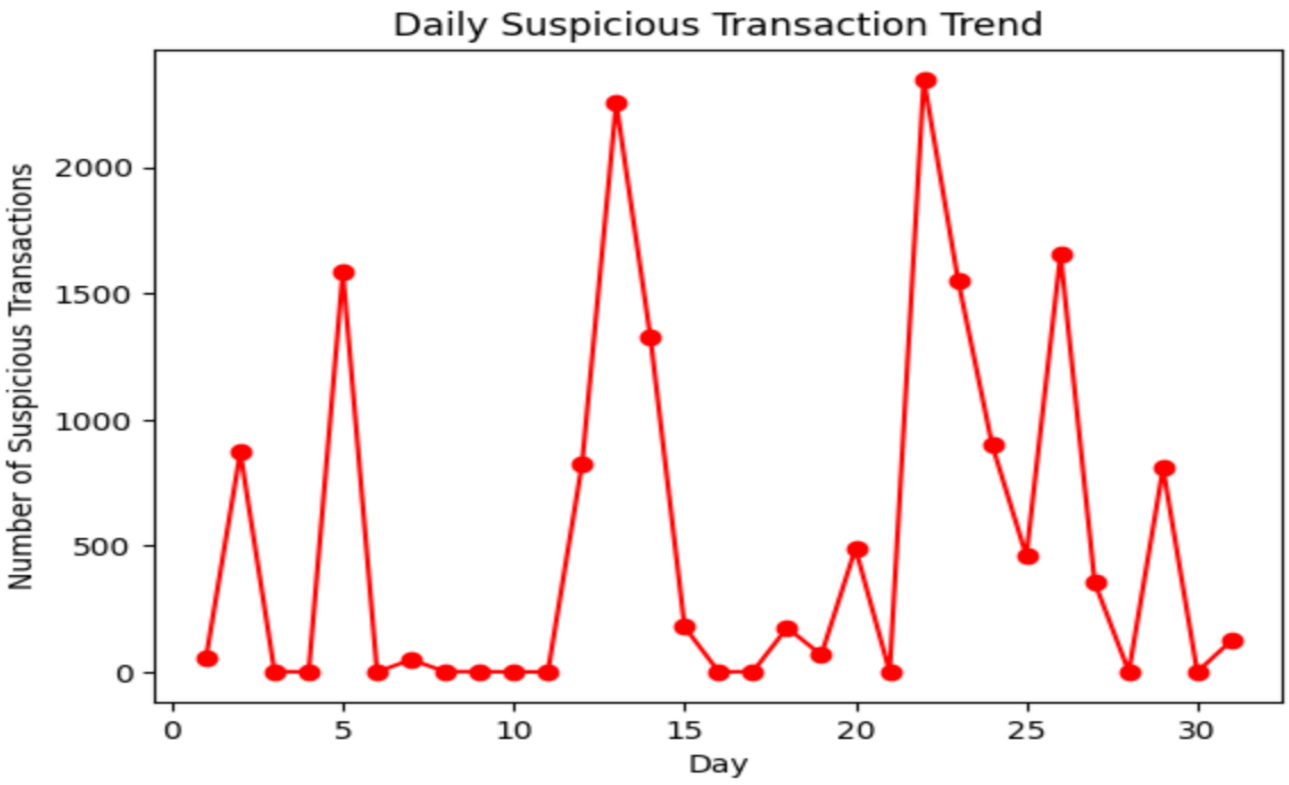
Explanation :

This chart shows the total count of suspicious and normal transactions for the year 2024. It emphasizes the stark class imbalance, with normal transactions (in blue) significantly outnumbering suspicious ones (in orange). All data points are from a single year (2024), suggesting the analysis focuses on one year of transaction data, with a consistent trend of fewer suspicious activities across the entire dataset.



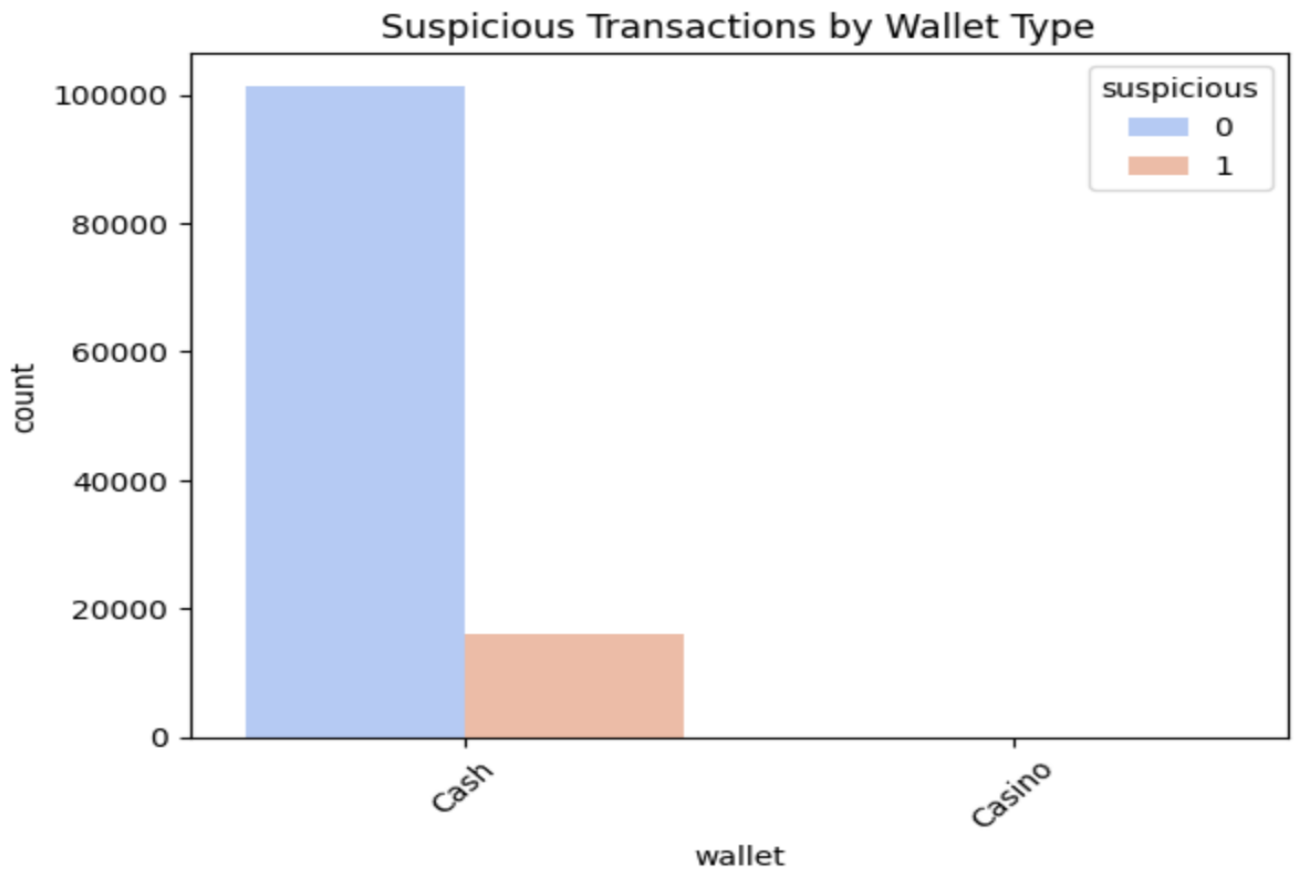
Explanation :

The monthly distribution chart highlights how suspicious and normal transactions vary over the months. Normal transactions peaked in March and April, while suspicious transactions show a relatively higher count in May and June, with minor peaks in August and December. This indicates that suspicious activity is not evenly distributed throughout the year and may follow specific seasonal or behavioral trends, which could be useful for fraud detection modeling or investigation.



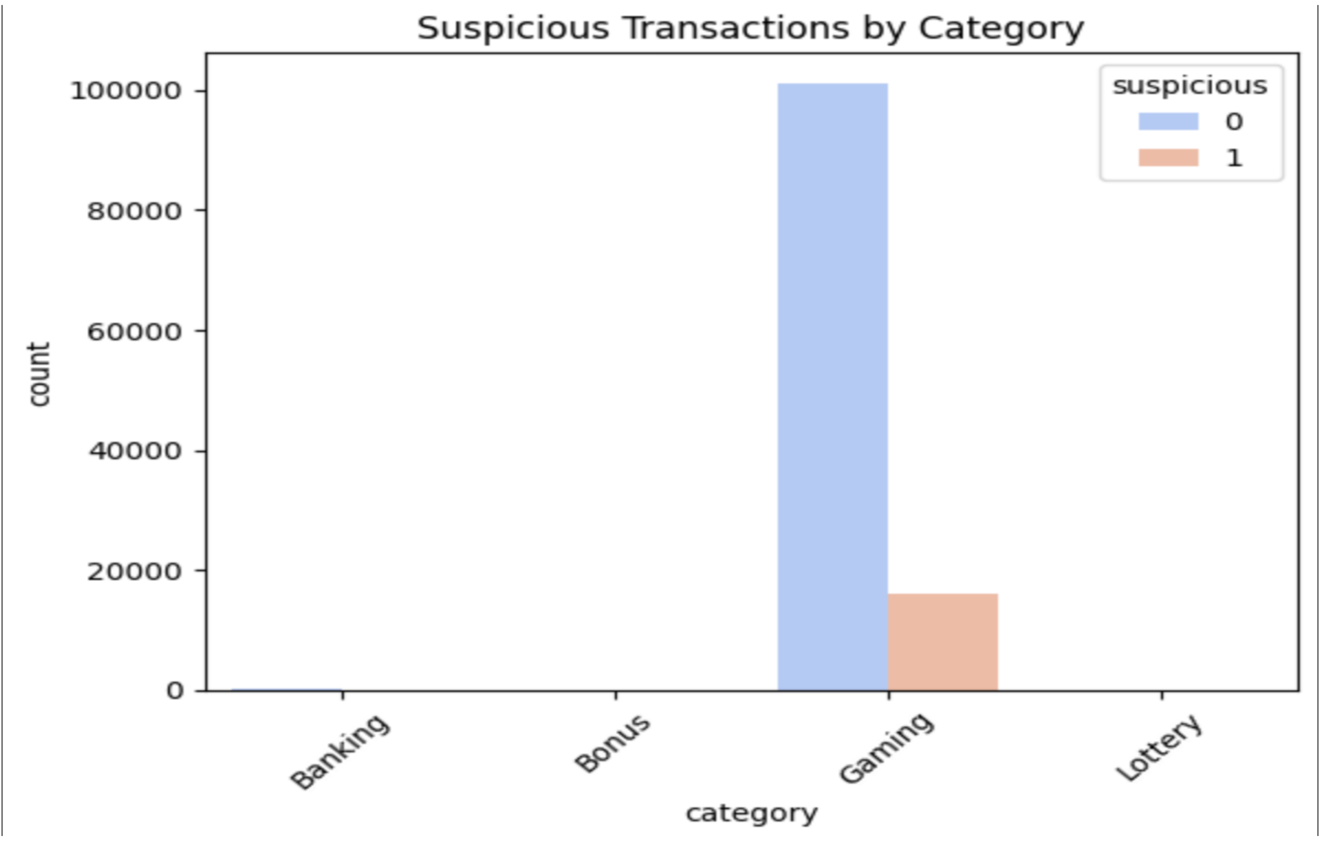
Explanation :

This line graph presents the number of suspicious transactions recorded daily over a month. While most days have low or negligible suspicious activity, there are sudden and extreme spikes on certain days, such as days 5, 13, and 22. These spikes suggest concentrated periods of fraud or abnormal behavior. The inconsistency in volume indicates potential targeted attacks or system vulnerabilities. Monitoring these peaks can help identify patterns. A further drill-down into those specific days may reveal critical insights.



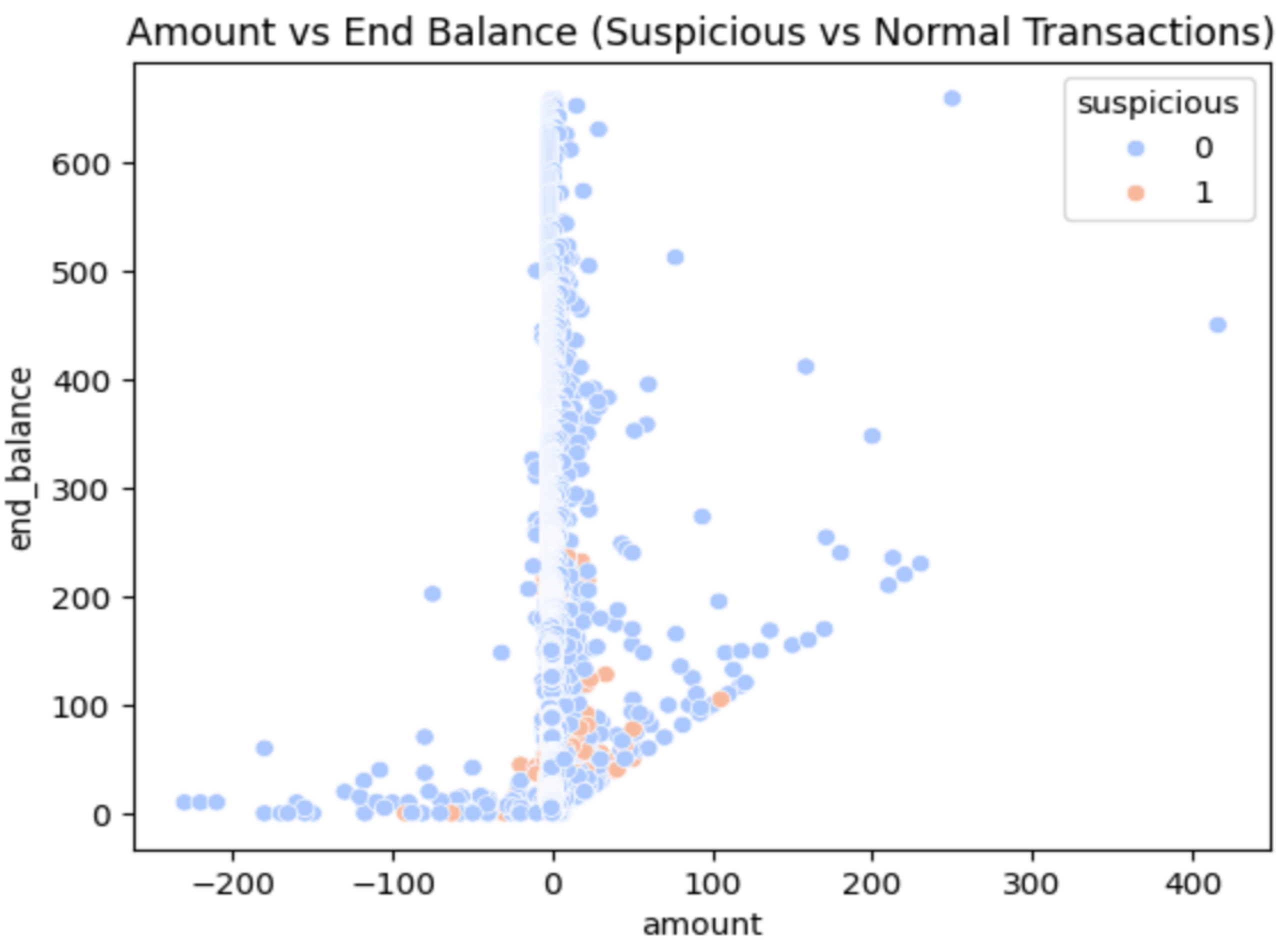
Explanation :

This bar chart breaks down transactions by wallet type and flags whether they were suspicious. The "Cash" wallet type overwhelmingly dominates both normal and suspicious transaction counts, especially in the suspicious category. In contrast, the "Casino" wallet shows almost no activity. This may indicate that cash-based wallets are more vulnerable to fraud or misuse. Risk mitigation strategies should prioritize cash wallet transactions. Enhancing monitoring here could reduce fraudulent activity.



Explanation :

The bar plot illustrates the frequency of suspicious and normal transactions across several categories like Gaming, Banking, Bonus, and Lottery. "Gaming" stands out as the dominant category for both suspicious and non-suspicious activities. The other categories contribute very little, with almost negligible suspicious activity. This implies that the Gaming category poses a higher risk and could be a hotspot for fraud. Tighter controls and anomaly detection mechanisms in Gaming may be necessary. The pattern indicates fraudsters’ preference for high-volume, loosely regulated channels.



Explanation :

This scatter plot compares the transaction amount against the resulting end balance, color-coded by whether the transaction was suspicious. Suspicious transactions (orange) tend to cluster around lower amounts and lower end balances. This clustering suggests that fraudulent activity is often kept within modest financial limits, likely to avoid detection. The distribution for normal transactions (blue) is broader and extends to higher values. This pattern could inform thresholds for flagging potentially suspicious transactions. It also indicates a deliberate effort by fraudsters to stay under financial radars.

CONCLUSION

This project successfully demonstrates how integrating **rule-based logic** with **machine learning** can effectively detect fraudulent transactions in the online gaming industry. By analyzing features like transaction time, amount, and wallet type, the model identifies suspicious behavior with high accuracy.

The use of a **Random Forest classifier** proved highly effective due to its robustness and ability to handle complex patterns in financial data. The system lays a strong foundation for further development, including real-time fraud detection, alert systems, and deeper behavioral analytics. This enhances the **security and trust** in digital gaming platforms, protecting both users and businesses from potential fraud.

GITHUB LINK : https://github.com/AatiqahHarmine/Fraud-Detection-Using-Random-Forest/tree/main