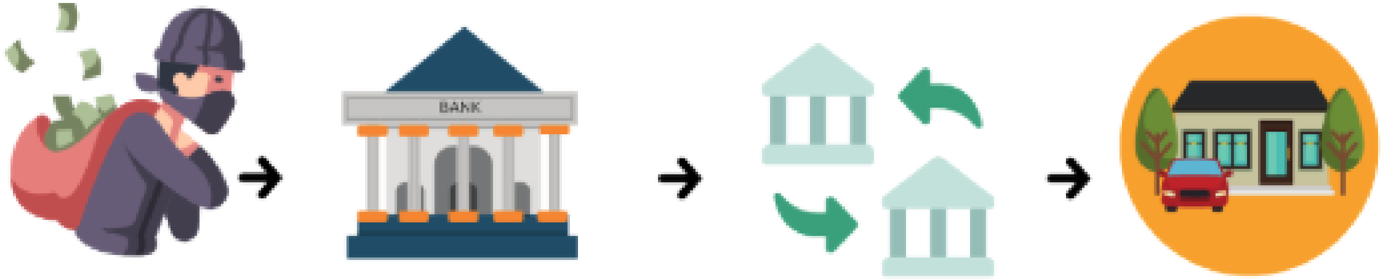
Anti-Money Laundering (AML) Fraud Detection



Robins Yadav
Feb 18, 2025

Table of Contents

Project Overview & Goals

Developed a Machine Learning system to detect potentially money laundering transactions, enhancing AML systems.

Addressed the challenge of high false positive and false negative rates in traditional AML systems.

Improved Precision & Recall compared to baseline methods.

Deployed the best model on AWS using Docker with ECR and EC2, and integrated into a CI/CD pipeline via GitHub Actions

Problem Statement

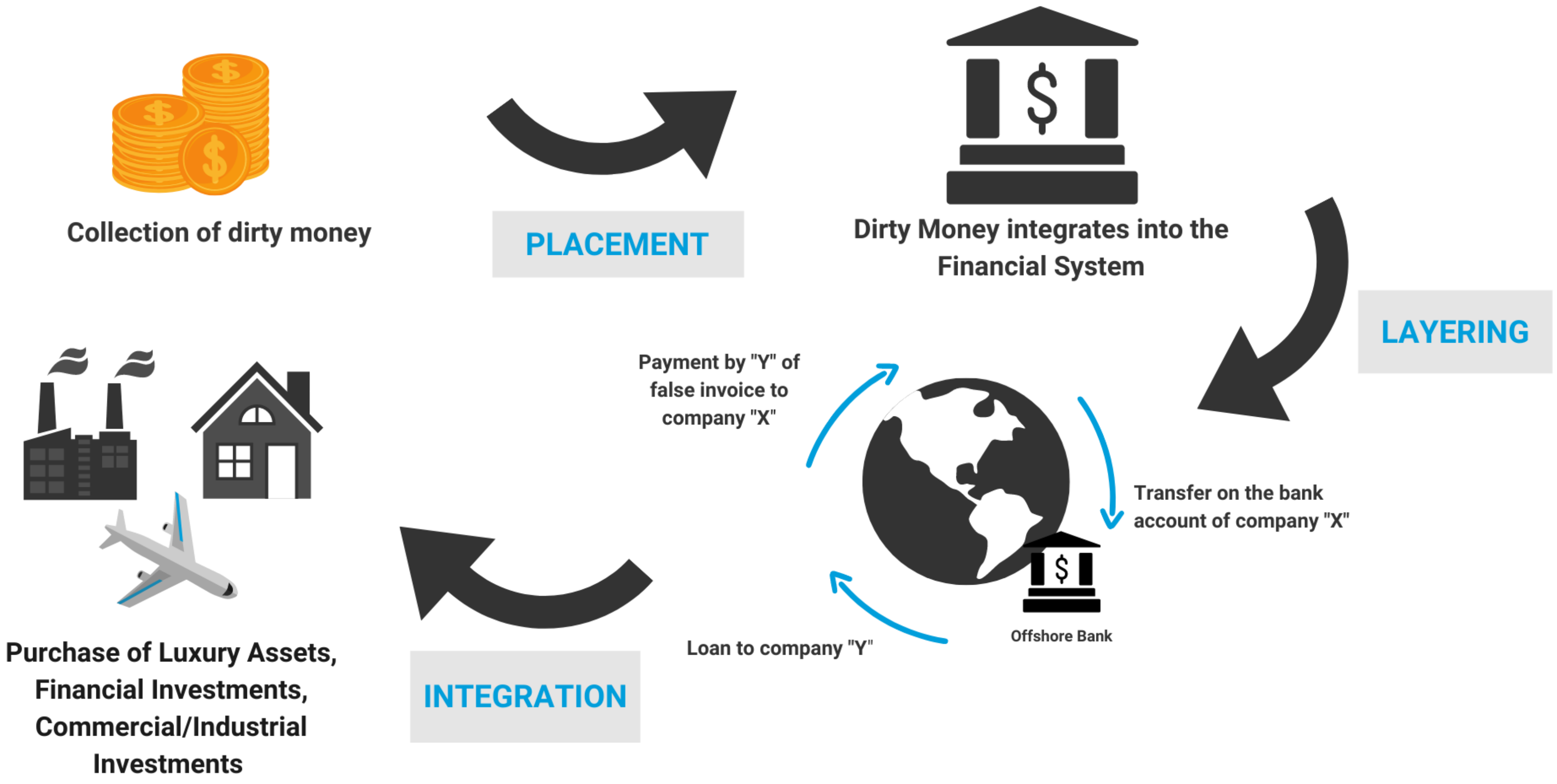
Problem: Money laundering is a massive financial problem (multi-billion dollar).

Challenge: Traditional AML systems suffer from High **False Positive** Rate and High **False Negative** Rate

Goal: Develop AI/ML system that significantly reduces both false positives and false negatives, improving efficiency and effectiveness.

Problem Statement

- Big Picture



Source: United Nations – Office on Drugs and Crime

Exploratory Data Analysis (EDA)

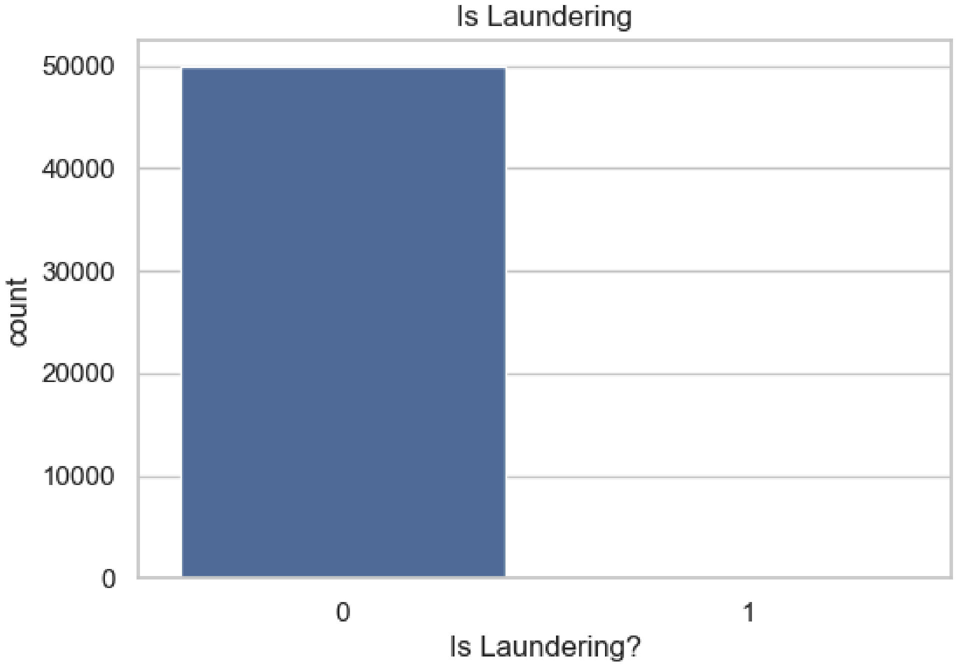
- Understanding the Data Structure and Content
  - Data source: IBM (Kaggle) – Research Paper [arXiv] on Jan 25, 2024 describing generation of data
  - Data loading and inspection
  - Data Shape and Size
  - Data Types
  - Check for duplicates
  - Missing Values
  - Data Statistics

	Timestamp	From Bank	Account	To Bank	Account.1	Amount Received	Receiving Currency	Amount Paid	Payment Currency	Payment Format	Is Laundering
3507139	2022/09/07 12:15	29	80CF063F0	235843	80CFE1EB0	386006.86	Brazil Real	386006.86	Brazil Real	Cheque	0
2054082	2022/09/03 21:15	70	100428660	22732	80BFEBFF0	8638.95	US Dollar	8638.95	US Dollar	Cheque	0
4745576	2022/09/09 19:22	338871	8144F97F0	15964	8144FEB20	80.84	Euro	80.84	Euro	Credit Card	0

Exploratory Data Analysis (EDA)

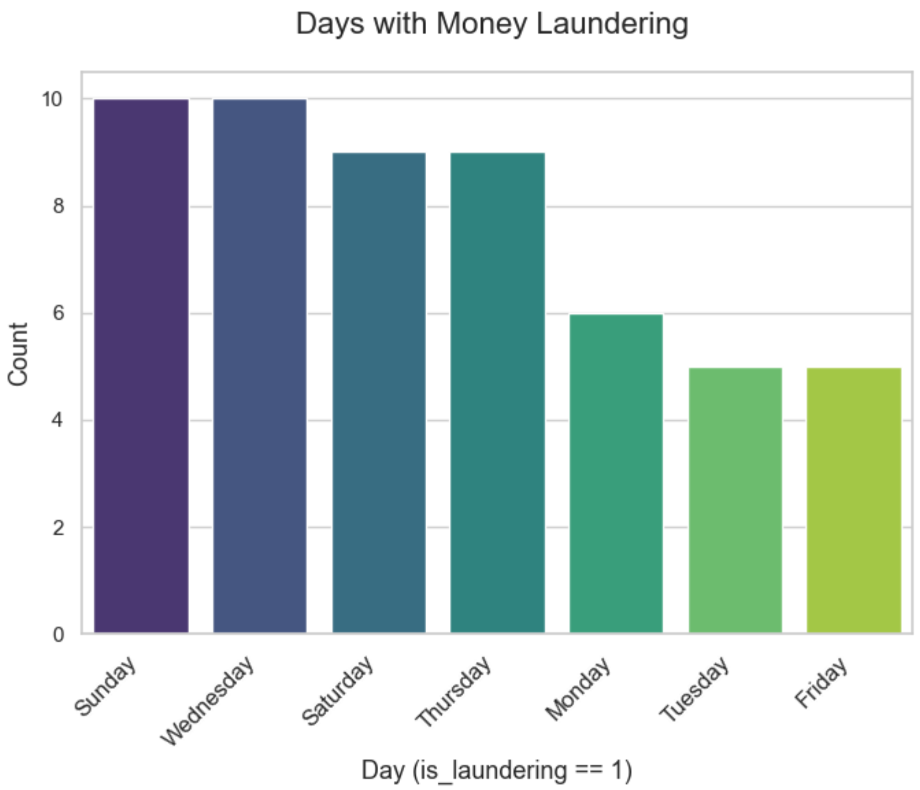
- Target Variable Analysis

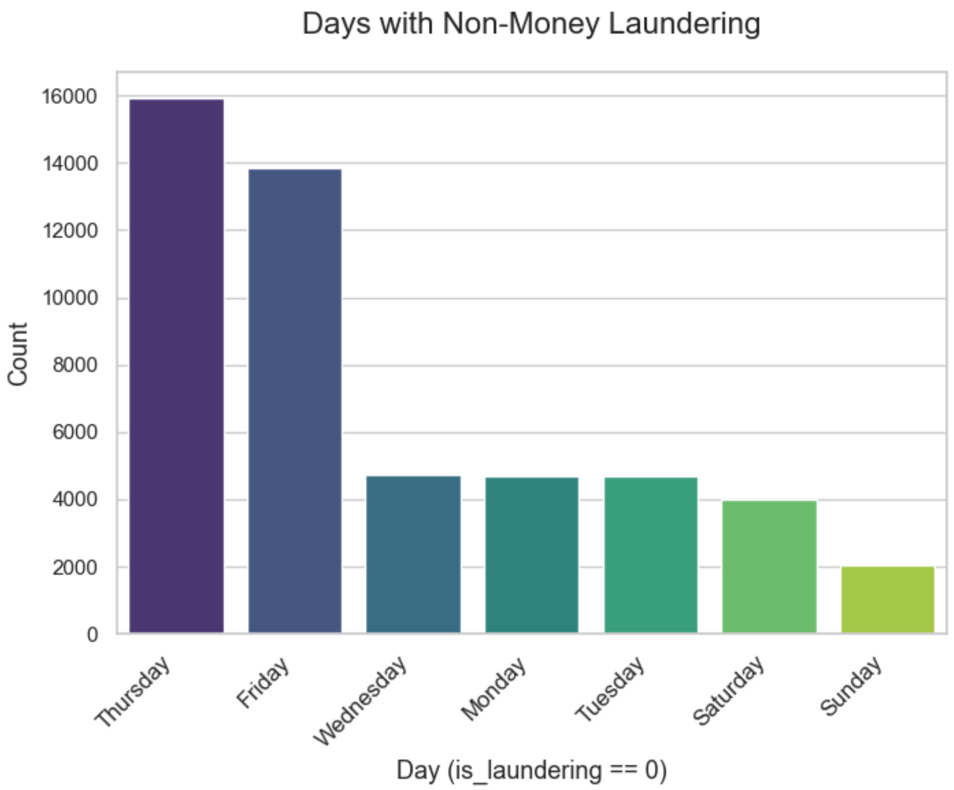
```
is_laundering
0    49946
1       54
```



Exploratory Data Analysis (EDA)

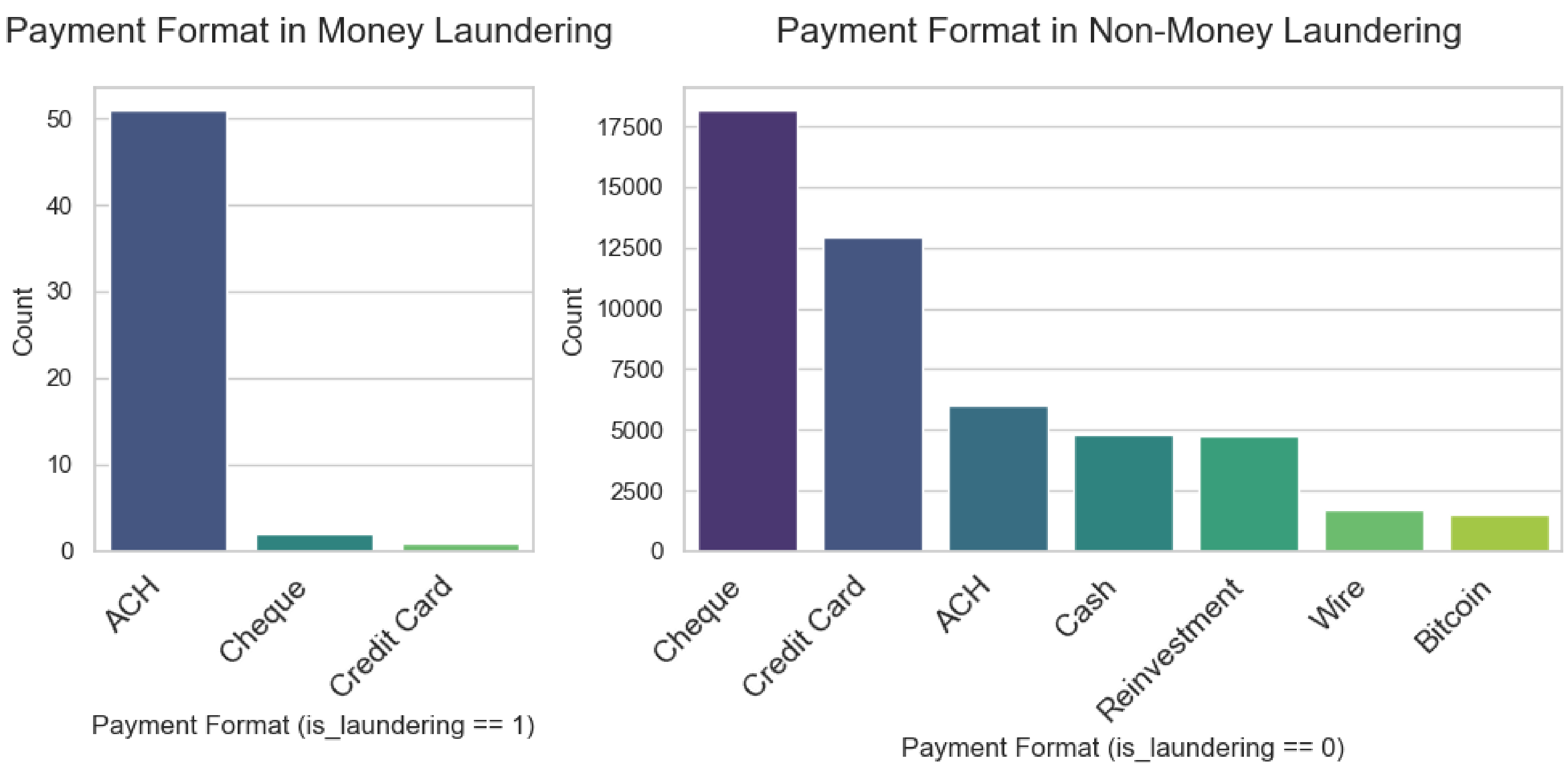
- Days in week where Money Laundering occurred





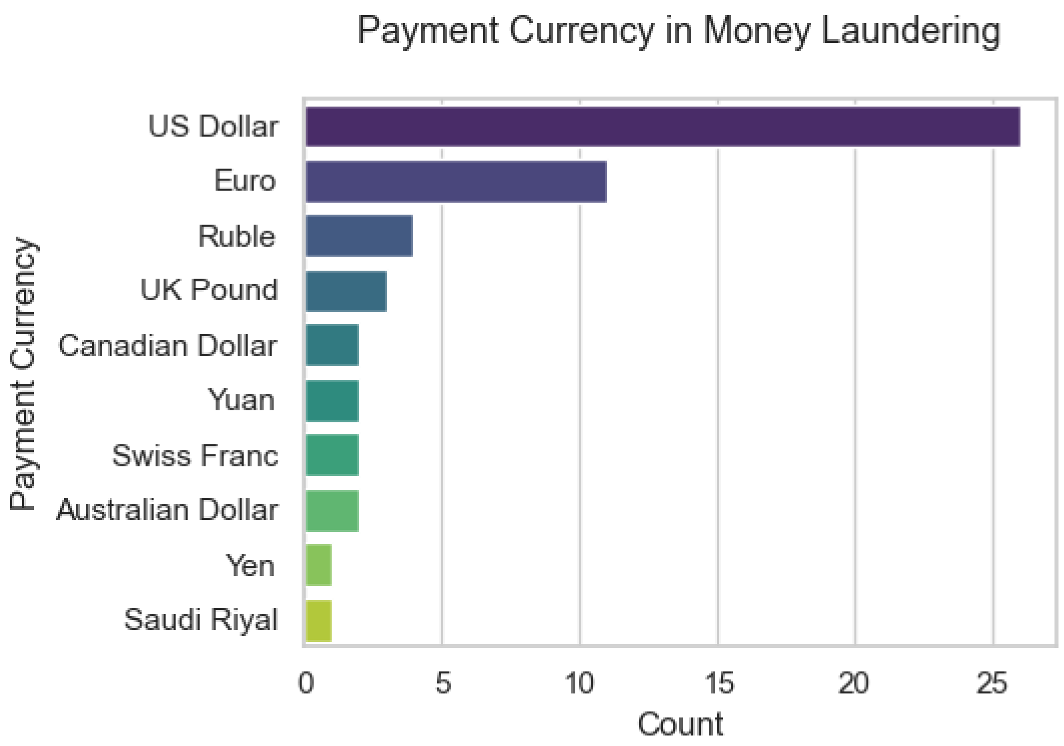
Exploratory Data Analysis (EDA)

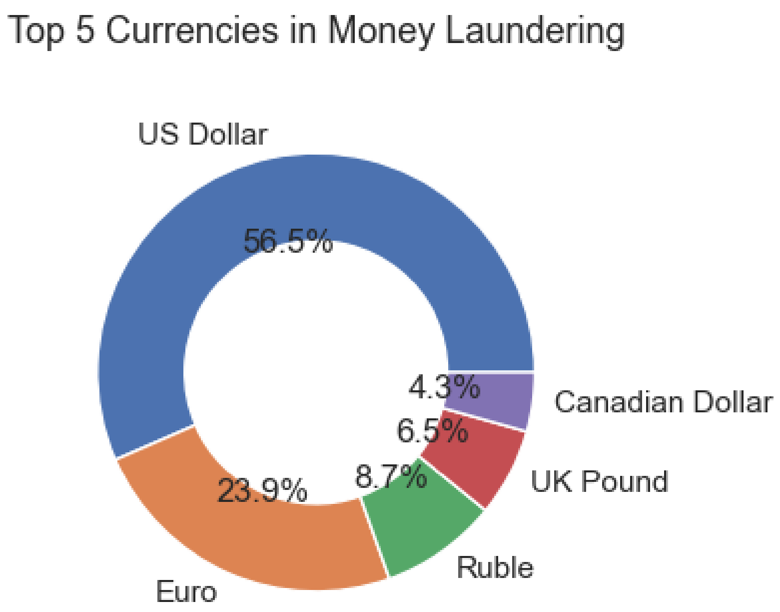
- Types of payment format in Money Laundering



Exploratory Data Analysis (EDA)

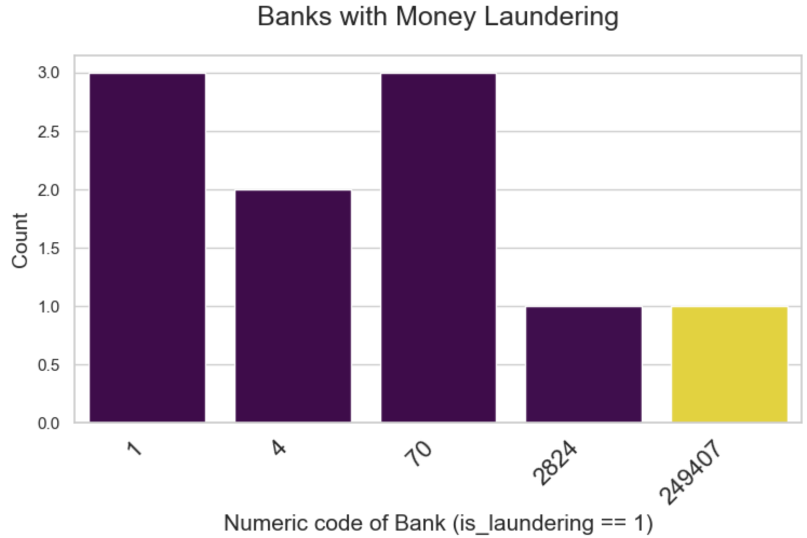
- Types of payment currency in Money Laundering

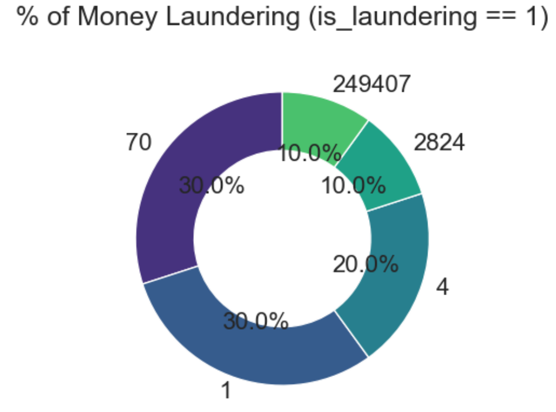


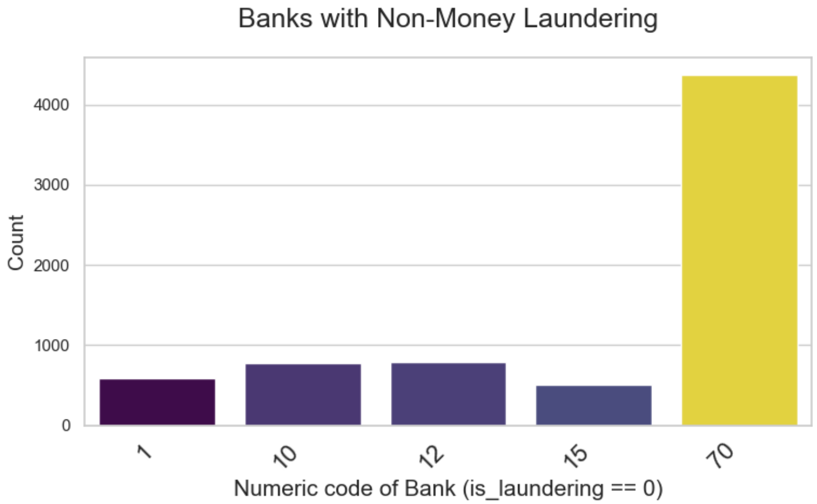


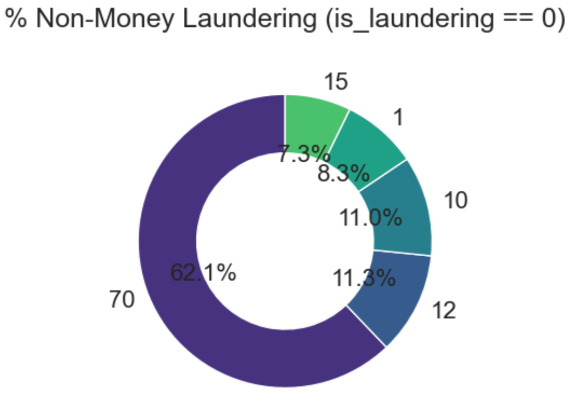
Exploratory Data Analysis (EDA)

- Banks involves in Money Laundering



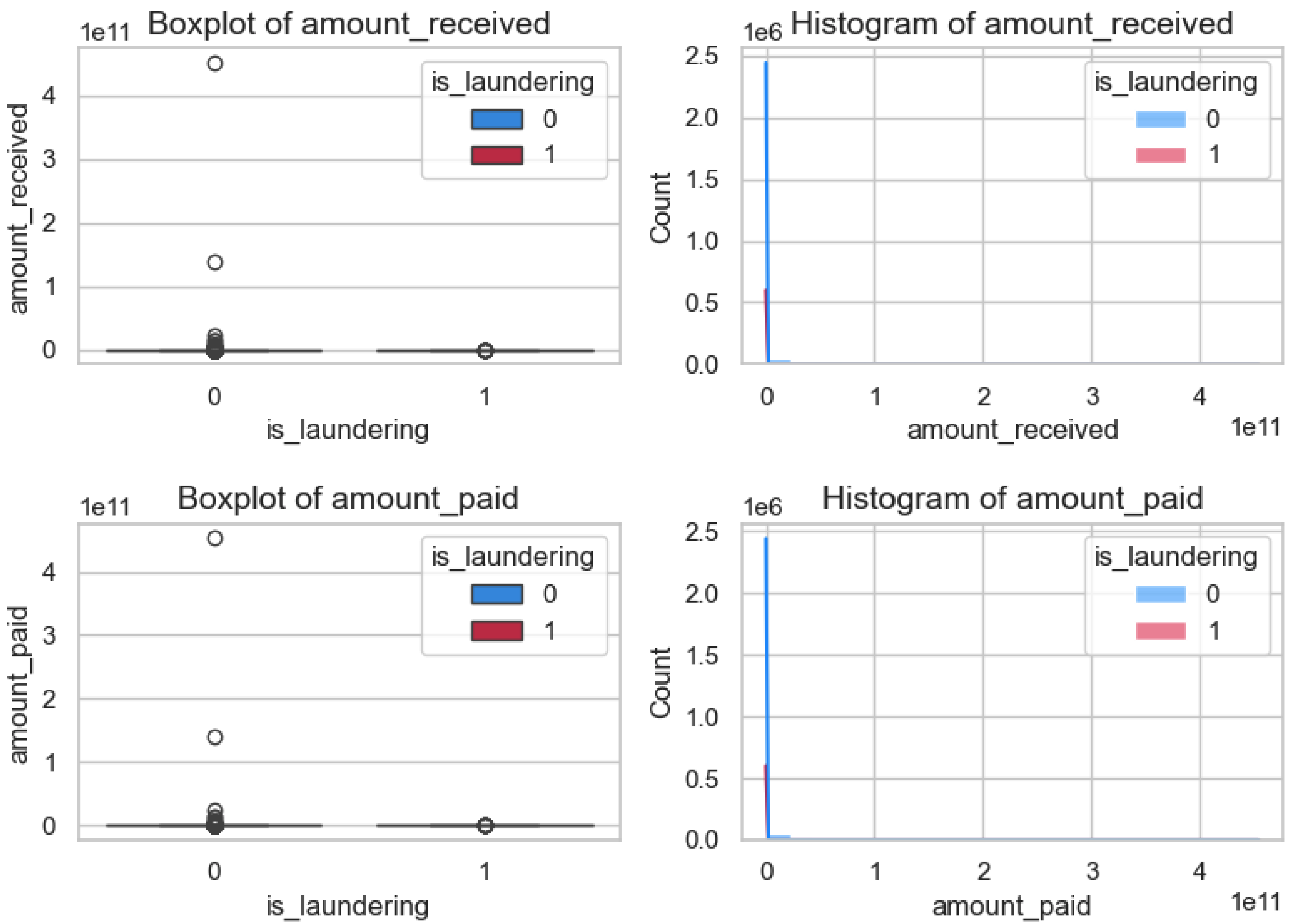






Exploratory Data Analysis (EDA)

- Amount Received and Amount Paid



Feature Engineering and Data Preprocessing

- Feature Selection: Numerical Features
  - Check Multicollinearity
    - VIF is used to assess the multicollinearity among the independent (predictor) variables

feature	VIF
const	1.000189
amount_received	2854.096145
amount_paid	2854.096145

VIF = 1: No multicollinearity.

1 < VIF < 10: Moderate multicollinearity.

VIF > 10: High multicollinearity.

Conclusion: They are highly correlated therefore one of them need to be dropped

  - Feature Selection method: *Recursive Feature Elimination, Feature Importance - ExtraTreesClassifier()*

Feature Engineering and Data Preprocessing

- Feature Selection: Categorical Features
  - Check Multicollinearity
    - Chi-square statistic is one way to show a relationship between two categorical variables.

Column	Hypothesis Result
from_bank	Fail to Reject Null Hypothesis - There is no relationship
account	Reject Null Hypothesis - There is a relationship
to_bank	Fail to Reject Null Hypothesis - There is no relationship
account_1	Reject Null Hypothesis - There is a relationship
receiving_currency	Fail to Reject Null Hypothesis - There is no relationship
payment_currency	Fail to Reject Null Hypothesis - There is no relationship
payment_format	Reject Null Hypothesis - There is a relationship
date	Reject Null Hypothesis - There is a relationship
day	Reject Null Hypothesis - There is a relationship
time	Reject Null Hypothesis - There is a relationship

Conclusion: Features like ***account***, ***account_1***, ***date***, ***day***, and ***time*** are important for model training and predictions

Feature Engineering and Data Preprocessing

```python
"""
    - Preprocessing datasets for modeling
    - Imputing, Scaling and encoding
"""
def num_cat_transformer(numerical_features, categorical_features):
    # Preprocessing for numerical features:
    num_transformer = make_pipeline(
        SimpleImputer(strategy='median'),  # Impute missing values with median
        RobustScaler()  # Scale numerical features
    )

    # Preprocessing for categorical features:
    # Frequency Encoding for high cardinality features
    freq_encoder = CountEncoder(normalize=True)  # Normalize frequency encoding
    # One-Hot Encoding for low cardinality features
    one_hot_encoder = OneHotEncoder(handle_unknown='ignore')

    # Apply different encodings to different categorical features
    cat_transformer = make_column_transformer(
        (freq_encoder, ['account', 'account_1']),  # Frequency Encoding for account and account_1
        (one_hot_encoder, ['payment_format', 'day']),  # One-Hot Encoding for others
        remainder="drop"  # Drop columns not explicitly transformed
    )

    column_transformer = make_column_transformer(
        (num_transformer, numerical_features),  # Apply numerical transformer to numerical features
        (cat_transformer, categorical_features),  # Apply categorical transformer to categorical features
        remainder="drop"  # Drop columns not explicitly transformed
    )

    return column_transformer
```

Model Development and Evaluation

- **Model Selection and Training**
    - **Model selected**: Random Forest, AdaBoost, XGBoost
    - **Training**: Cross-Validation, Hyperparameters Tunning (GridSearch)
    - **Class Imbalance**: SMOTE

Model Development and Evaluation

- **Performance Metrics**

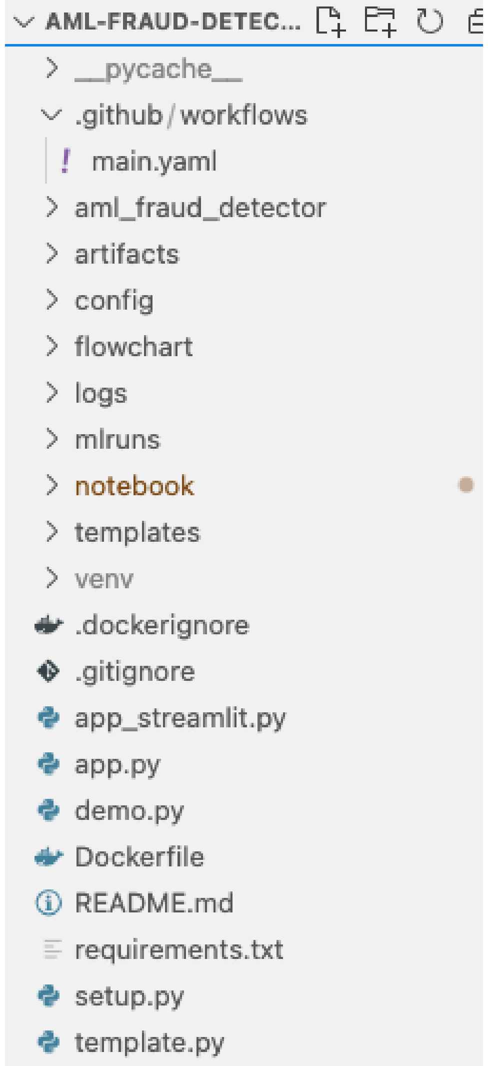
Model performance on Train data:

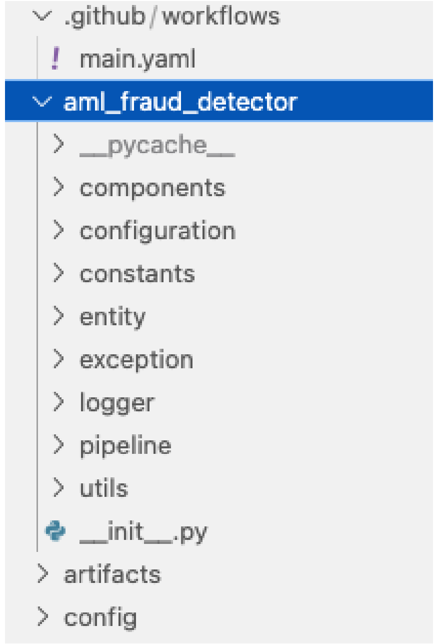
	Model	Precision	Recall	F1 Score	Confusion Matrix
0	Random_Forest	0.999900	0.999900	0.999900	[[39952, 1], [7, 39958]]
1	AdaBoost	0.968477	0.967404	0.967422	[[37731, 377], [2228, 39582]]
2	XGBoost	0.990638	0.990503	0.990503	[[39251, 51], [708, 39908]]

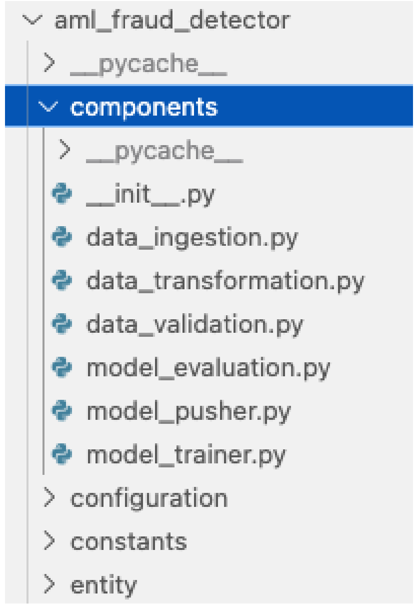
Model performance on Test data:

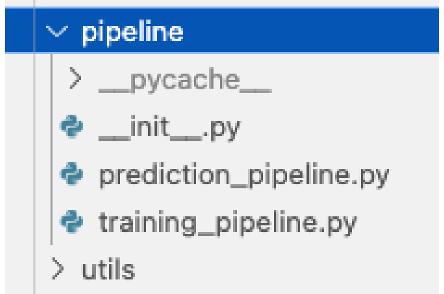
	Model	Precision	Recall	F1 Score	Confusion Matrix
0	Random_Forest	0.961823	0.9788	0.969589	[[9787, 12], [200, 1]]
1	AdaBoost	0.921879	0.9371	0.907923	[[9362, 4], [625, 9]]
2	XGBoost	0.957494	0.9765	0.966164	[[9764, 12], [223, 1]]

Deployment with GitHub Actions and AWS









Deployment with GitHub Actions and AWS

GitHub Actions

Configure GitHub Actions workflows in the `.github/workflows` directory, the `main.yaml`

AWS IAM

IAM User Creation :
`AmazonEC2ContainerRegistryFullAccess,`
`AmazonEC2FullAccess`

Amazon ECR

Create ECR repository for Docker image
ECR Repo URI:
####.dkr.ecr.us-east-1.amazonaws.com/
aml_fraud_detector-container

Amazon EC2

Create and Launch EC2 instance.
Steps: Update & then install Docker

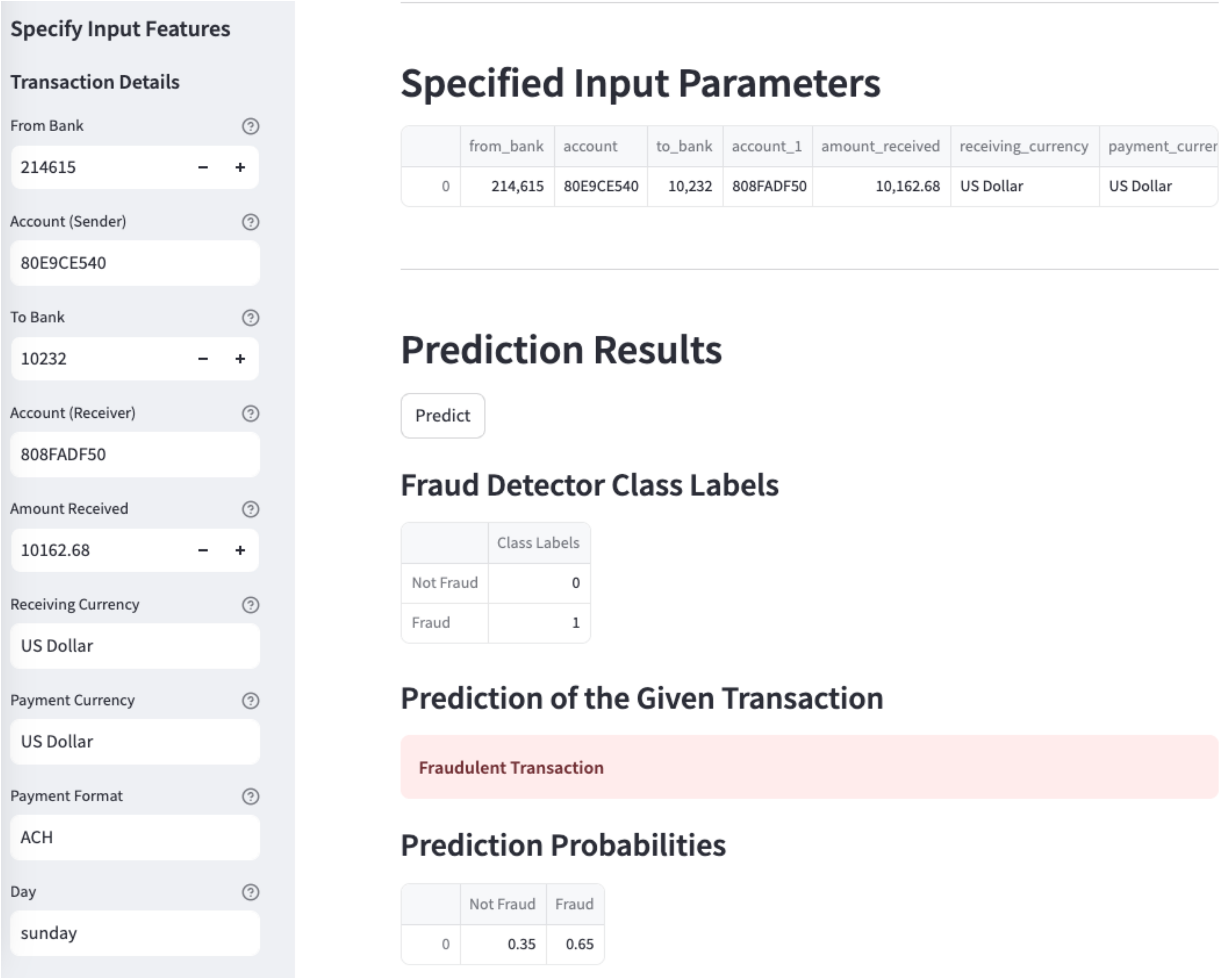
EC2 Instance Self-hosted runner

Configure EC2 as Self-Hosted Runner
Steps: GitHub > Settings > Actions > Runners >
New self-hosted runner > choose os (Linux) >
then run command one by one in EC2 instance
> finally Enter runner name : self-hosted

GitHub Secrets

GitHub Secrets Setup:
Steps: Settings > Secrets and variables > actions >
New repository secret >
AWS_ACCESS_KEY_ID = #####
AWS_SECRET_ACCESS_KEY = ####
AWS_REGION = us-east-1
AWS_ECR_LOGIN_URI = ####.dkr.ecr.us-east-1.amazonaws.com
ECR_REPOSITORY_NAME = aml_fraud_detector-container

User Interface (Streamlit)



Business Impact

Reduced Financial Losses : Stops fraud and avoids fines by improving the detection of fraudulent transactions .

Enhanced Customer Trust : Fewer mistakes, happier customers.

Improved Operational Efficiency : Automating fraud detection using machine learning models , making it faster.

Follows the AML Rules: Keeps the company safe and trusted.

Thank You!

Questions?