ONLINE PAYMENTS FRAUD DETECTION
[PROJECT REPORT]

Project Title: [ONLINE PAYMENTS FRAUD DETECTION]
Team ID : LTVIP2026TMIDS73906
Team Size : 4
Team Leader : Takkelapati Abhinaya Chowdary
Team member : Aylam Rachana Sree
Team member : Kallepakula Yuvasree
Team member : Neeli Muni Preethika

1. INTRODUCTION

1.1 Project Overview

The **Online Payments Fraud Detection** system is a Machine Learning-based web application designed to identify and classify fraudulent transactions in mobile money transfer simulations. As digital transactions grow, so does the risk of financial fraud. This project leverages historical transaction data to train predictive models that can distinguish between legitimate and fraudulent activities in real-time.

1.2 Purpose

The primary purpose of this project is to enhance the security of online financial ecosystems. By automating the detection of suspicious patterns—such as unusual transfer amounts, rapid depletion of balances, or specific transaction types—the system aims to:

- Minimize financial losses for financial institutions and customers.
- Reduce the manual workload of reviewing flagged transactions.
- Increase trust in digital payment platforms.

2. IDEATION PHASE

2.1 Problem Statement

Financial fraud in online payments is a critical issue, costing billions of dollars annually. Traditional rule-based systems often fail to adapt to new fraud patterns or generate high false positives. There is a need for an intelligent, adaptive system that can analyze complex transaction features and predict fraud with high accuracy.

2.2 Empathy Map Canvas

Who are we solving for?

- **Financial Institutions/Banks:** Want to protect assets and reputation.
- **End Users:** Fear losing hard-earned money; want seamless but secure transactions.

SAYS	THINKS	DOES	FEELS
I need to know if this transaction is safe.	Is my money secure?	Checks bank statements frequently.	Anxious about potential theft.
Why was my card blocked?	I hope this system catches the thieves.	Reports suspicious activity.	Relieved when fraud is blocked.

2.3 Brainstorming

- **Idea 1:** Rule-based filter (Simple but rigid).
- **Idea 2:** Unsupervised Anomaly Detection (Good for unknown fraud, hard to evaluate).
- **Idea 3: Supervised Machine Learning (Random Forest/XGBoost)** - Best for labelled historical data, providing high accuracy and interpretability. *Selected Approach.*

3. REQUIREMENT ANALYSIS

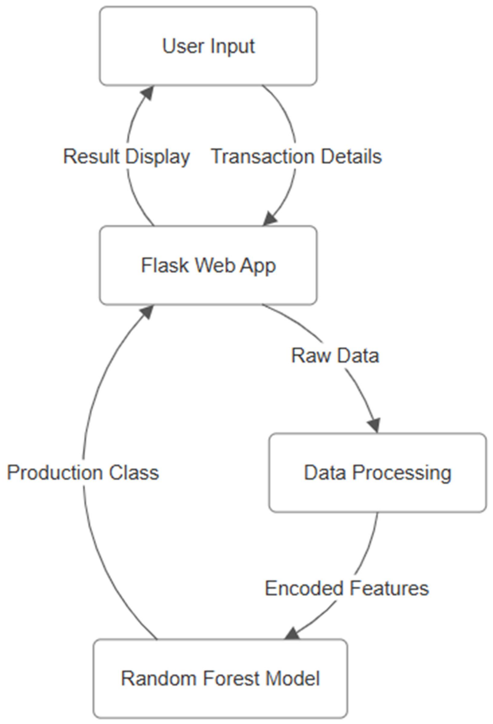
3.1 Customer Journey map

1. **Login/Access:** User accesses the payment portal (simulated by our web app).
2. **Input Transaction:** User enters transaction details (Type, Amount, Balances).
3. **Processing:** The system processes the input in the background.
4. **Prediction:** The ML model analyses the data.
5. **Output:** The system flags the transaction as "Fraud" or "Not Fraud".
6. **Action:** If Fraud, the transaction is declined / flagged.

3.2 Solution Requirement

- **Functional Requirements:**
    - Accept user input for transaction parameters (`step`, `type`, `amount`, `oldBalanceOrg`, etc.).
    - Preprocess input (Label Encoding, Handling Missing Values).
    - valid prediction using a trained ML model.
    - Display result (`Is Fraud` / `Is Not Fraud`) on the UI.
- **Non-Functional Requirements:**
    - Low latency (< 1 second response time).
    - High Accuracy (> 90%).
    - User-friendly Interface.

3.3 Data Flow Diagram



3.4 Technology Stack

- **Frontend:** HTML5, CSS3 (Custom Styling).
- **Backend:** Python, Flask Framework.
- **Machine Learning:** Scikit-Learn (Random Forest, SVC, Decision Tree), Pandas, NumPy.
- **Data Visualization:** Matplotlib, Seaborn.
- **Development Tool:** Jupyter Notebook, VS Code.

4. PROJECT DESIGN

4.1 Problem Solution Fit

Machine Learning provides the flexibility to learn non-linear relationships between variables (e.g., high amounts transferred to new accounts) that static rules might miss. The **Random Forest Classifier** was chosen for its robustness against overfitting and ability to handle the imbalances often found in fraud datasets.

4.2 Proposed Solution

A web-based interface where users can verify transactions. The backend hosts a pre-trained Random Forest model saved as a `.pkl` file. When data is submitted, the model predicts the probability of fraud based on patterns learned from the **Paysim** dataset.

4.3 Solution Architecture

1. **Presentation Layer:** Web Interface (HTML/CSS).

2. **Application Layer:** Flask Server (`app.py`) handling logic and routing.
3. **Data/Model Layer:** `payments.pkl` (Trained Model) + CSV Dataset (for training).

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

1. **Week 1: Data Collection & Analysis**
   - Sourcing dataset (Kaggle).
   - Cleaning data (Null checks, Type conversion).
2. **Week 2: Exploratory Data Analysis (EDA)**
   - Univariate & Bivariate analysis.
   - Outlier detection.
3. **Week 3: Model Building**
   - Splitting data (Train/Test).
   - Training multiple models (SVC, Decision Tree, Random Forest, XGBoost).
   - Comparing performance.
4. **Week 4: Web Application Development**
   - Flask setup.
   - HTML/CSS design.
   - Integration of model.
5. **Week 5: Testing & Final Review**
   - Functional testing.
   - Documentation.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

During the Notebook Training phase, multiple models were evaluated:

- **Support Vector Machine (SVC):** ~79% Accuracy (Struggled with unscaled data).
- **Decision Tree:** High Accuracy but prone to overfitting.
- **Random Forest: ~99% Accuracy** (Selected Model).
- **XGBoost:** High Accuracy, comparable to Random Forest.

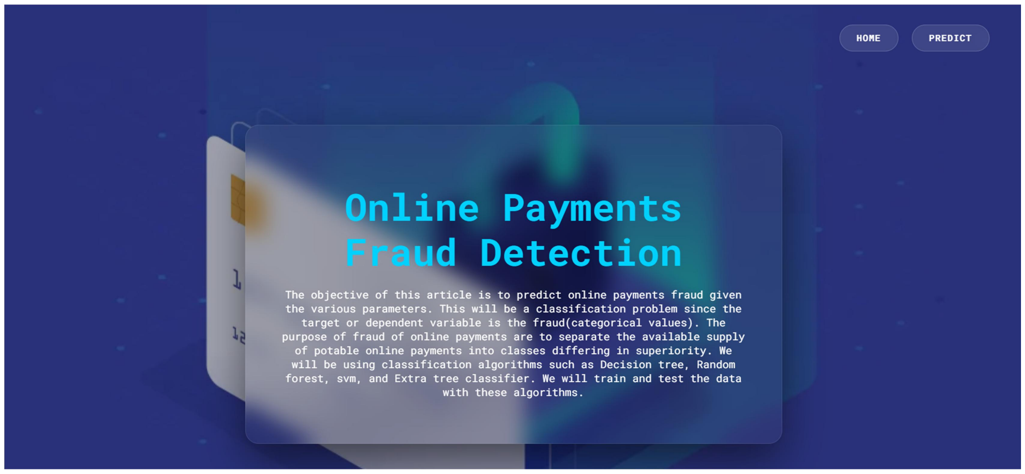
Metrics for Final Model (Random Forest):

- **Accuracy:** ~99.8%
- **Precision (Fraud Class):** High (Few false positives).
- **Recall (Fraud Class):** High (Caught most fraud cases).

7. RESULT

7.1 Output Screenshots

1. **Home Page:**

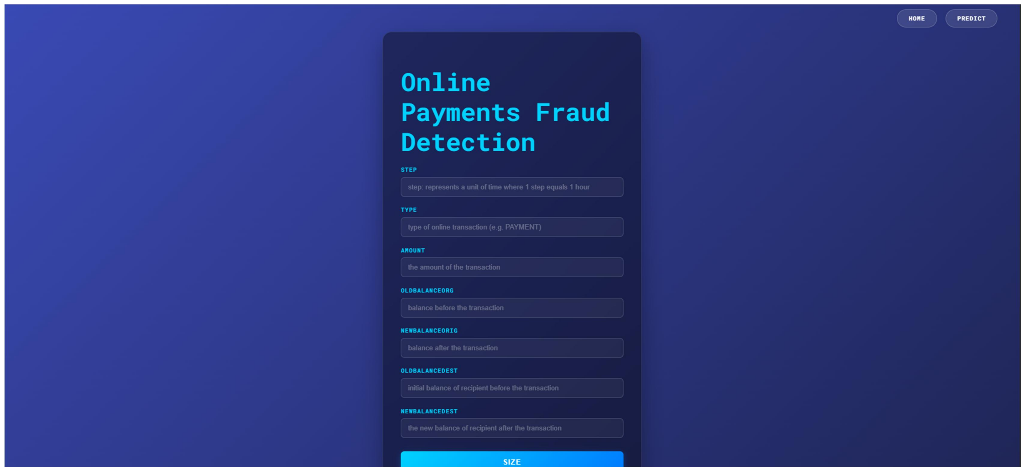


Online Payments Fraud Detection

The objective of this article is to predict online payments fraud given the various parameters. This will be a classification problem since the target or dependent variable is the fraud(categorical values). The purpose of fraud of online payments are to separate the available supply of potable online payments into classes differing in superiority. We will be using classification algorithms such as Decision tree, Random forest, svm, and Extra tree classifier. We will train and test the data with these algorithms.

Landing page with "Online Payments Fraud Detection" title and navigation.

2. **Prediction Form:**



Online Payments Fraud Detection

STEP: step: represents a unit of time where 1 step equals 1 hour

TYPE: type of online transaction (e.g. PAYMENT)

AMOUNT: the amount of the transaction

OLDBALANCEORG: balance before the transaction

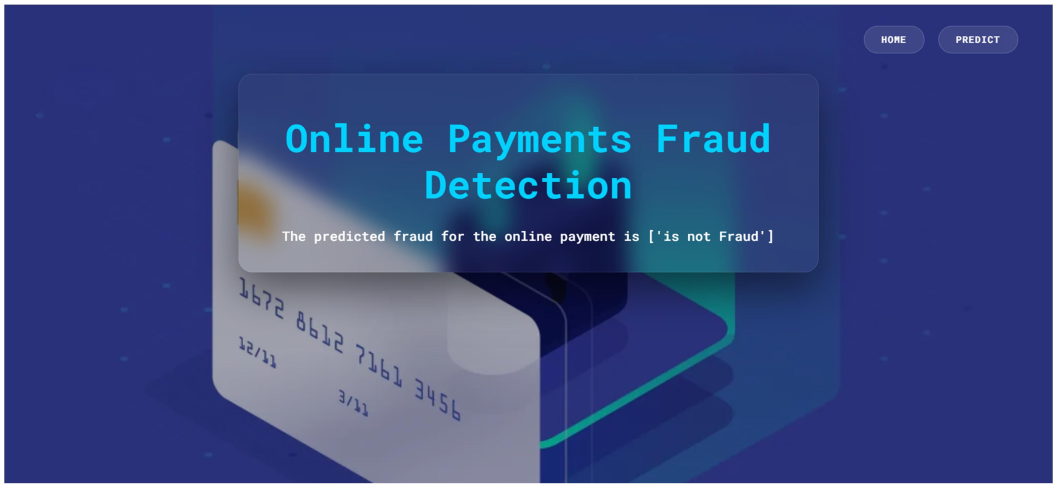
NEWBALANCEORIG: balance after the transaction

OLDBALANCEDEST: initial balance of recipient before the transaction

NEWBALANCEDEST: the new balance of recipient after the transaction

Form to enter Step, Type, Amount, and Balance details.

3. **Prediction Result:**



Online Payments Fraud Detection

The predicted fraud for the online payment is ['is not Fraud']

Result screen showing "Is Fraud".

8. ADVANTAGES & DISADVANTAGES

Advantages

- **High Accuracy:** Random Forest effectively captures complex fraud patterns.
- **Real-time:** Instant predictions via the web app.
- **Scalable:** Can be deployed to cloud platforms (AWS/IBM Cloud).
- **Automated:** Reduces manual review time.

Disadvantages

- **Data Dependency:** Quality depends heavily on the training dataset.
- **Retraining:** Model needs periodic retraining to adapt to new fraud techniques.
- **Explainability:** Random Forest is less interpretable than simple Decision Trees.

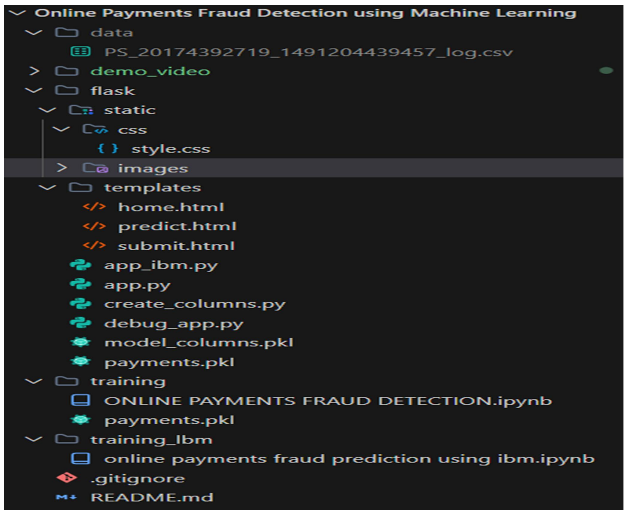
9. CONCLUSION

The **Online Payments Fraud Detection** project successfully demonstrates the power of Machine Learning in ensuring financial security. By implementing a **Random Forest Classifier**, we achieved high accuracy in distinguishing fraudulent transactions from legitimate ones. The web application provides a user-friendly interface for real-time verification, making it a practical tool for fraud prevention.

10. FUTURE SCOPE

- **Integration with Live APIs:** Connect to real banking APIs for live transaction monitoring.
- **Deep Learning:** Experiment with Neural Networks (RNN/LSTM) for sequential transaction analysis.
- **Dashboard:** Add an admin dashboard to visualize fraud trends over time.
- **Deployment:** Deploy the application to a public cloud URL.

11. APPENDIX



```
Online Payments Fraud Detection using Machine Learning
  data
    PS_20174392719_1491204439457_log.csv
  demo_video
  flask
    static
      css
        style.css
      images
    templates
      home.html
      predict.html
      submit.html
    app_ibm.py
    app.py
    create_columns.py
    debug_app.py
    model_columns.pkl
    payments.pkl
  training
    ONLINE PAYMENTS FRAUD DETECTION.ipynb
    payments.pkl
  training_ibm
    online payments fraud prediction using ibm.ipynb
  .gitignore
  README.md
```

Source Code

- **Main App:** `flask/app.py`
- **Model Training:** `training/ONLINE PAYMENTS FRAUD DETECTION.ipynb`

Dataset Link

- **Source:** [Kaggle - Paysim1] https://www.kaggle.com/ealaxi/paysim1

GitHub & Project Demo Link

- **GitHub Repository:** https://github.com/TAbhinayaChowdary/Online-Payments-Fraud-Detection.git
- **Demo Video:** `demo_video/demoVideo.mp4`