Online Payments Fraud Detection Using Machine Learning

Submitted in partial fulfillment of the requirements for the team SmartInternz

SmartInternz

Final Project Report

By

Shamini G(23BIT0373)

Thamarai Selvi Geevananthan(23BDS0190)

Kavitha M(23BBT0182)

Aanchal Ladha(23BIT0285)

CONTENTS

1. Introduction

1.1. Project Overview

This project builds a machine learning system to detect online payment fraud. Leveraging transaction datasets, the system preprocesses data, selects features, evaluates several classifiers (Decision Tree, Random Forest, Extra Trees, SVM, XGBoost), and integrates the best-performing model into a working application.

1.2. Objectives

- **Identify fraudulent transactions in real-time with high accuracy**
- **Compare multiple classification algorithms for effectiveness**
- **Optimize the selected model's performance**
- **Demonstrate the final model through a user-facing app**

(All available in final_project_code_templates)

2. Project Initialization and Planning Phase

2.1. Define Problem Statements

- **Unseen or rare fraud patterns bypass legacy systems**
- **Need for real-time detection to prevent financial losses**
- **Challenge: balancing detection accuracy with false positives**
  (Extract exact text from your planning docs in final_project_code_templates/Project_reports/....)

2.2. Project Proposal (Proposed Solution)

- **Use supervised learning with labeled transaction data**
- **Implement models: Decision Tree, Random Forest, Extra Trees, SVM, XGBoost**
- **Compare performance using metrics like accuracy, F1-score, recall**
- **Integrate best model into a Flask-based front-end interface**

2.3. Initial Project Planning Report
Product Backlog, Sprint Schedule, and Estimation

Use the below template to create a product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members	Sprint Start Date	Sprint End Date (Planned)
Sprint-1	Data Preprocessing	USN-1	As a data scientist, I want to clean and preprocess the dataset so that it is ready for model training.	2	High	Kavitha, Thamarai Selvi	01-June-2025	05-June-2025
Sprint-1	Exploratory Data Analysis	USN-2	As a developer, I want to analyse patterns in fraud and transaction types to understand key risk factors.	1	High	Shamini, Aanchal Ladha	01-June-2025	05-June-2025
Sprint-2	Model Building	USN-3	As a data scientist, I want to train a fraud detection model using Random Forest and Decision Tree Classifier.	2	Low	Kavitha, Shamini	06-June-2025	10-June-2025
Sprint-1	Model Evaluation	USN-4	As a tester, I want to evaluate the model's accuracy and identify any false positives or negatives.	2	Medium	Thamarai Selvi, Aanchal Ladha	06-June-2025	10-June-2025
Sprint-1	User Interface & Alert System	USN-5	As a user, I want to get instant alerts for suspicious transactions so that I feel safe using the platform.	1	High	Kavitha, Aanchal Ladha	11-June-2025	15-June-2025

(All available in final_project_code_templates)

3. Data Collection and Preprocessing Phase

3.1. Data Collection Plan and Raw Data Sources Identification Report

- **Source: dataset from Kaggle.com CSV**
- **PS_20174392719_1491204439457_log.csv**

3.2. Data Quality Report

- **Issues found: missing values, skewness, outliers**
- **Resolutions:**
    - **Log-transform skewed features**
    - **IQR-based outlier handling**
    - **Dropped non-informative or redundant fields**

3.3. Data Exploration and Preprocessing Report

- **Visualizations: histograms, box plots, correlation heatmaps**
  (Reference code in data_preprocessing.py, final_project_code_templates/training/ONLINE PAYMENTS FRAUD DETECTION.ipynb)
- **Preprocessing steps documented clearly in code and report**

(All available in final_project_code_templates)

4. Model Development Phase

4.1. Feature Selection Report

- **Features chosen: amount, old/new balance origin/destination, transaction type**
- **Insignificant features dropped after correlation/variance analysis**

4.2. Model Selection Report

- **Models tested:**
    - **Decision Tree**
    - **Random Forest**
    - **Extra Trees**
    - **Support Vector Machine**
    - **XGBoost**

- **Results compared on accuracy, precision, recall, F1-score**
  Best Model: DecisionTree

  Accuracy: 0.9997092392756443

4.3. Initial Model Training Code, Model Validation and Evaluation Report

- **train_model.py includes training and train_test_split**
- **Evaluation metrics captured and visualized in the github link**

5. Model Optimization and Tuning Phase

5.1. Hyperparameter Tuning Documentation

Inside Model Optimization Phase.docx, you tuned the Decision Tree hyperparameters (e.g., max_depth, min_samples_split) using GridSearchCV and validated improvements via cross-validation.

5.2. Performance Metrics Comparison Report

- **Before tuning: ~99.92% accuracy**
- **After tuning: 99.97% accuracy**
- **Improvements in recall and precision metrics**

5.3. Final Model Selection Justification

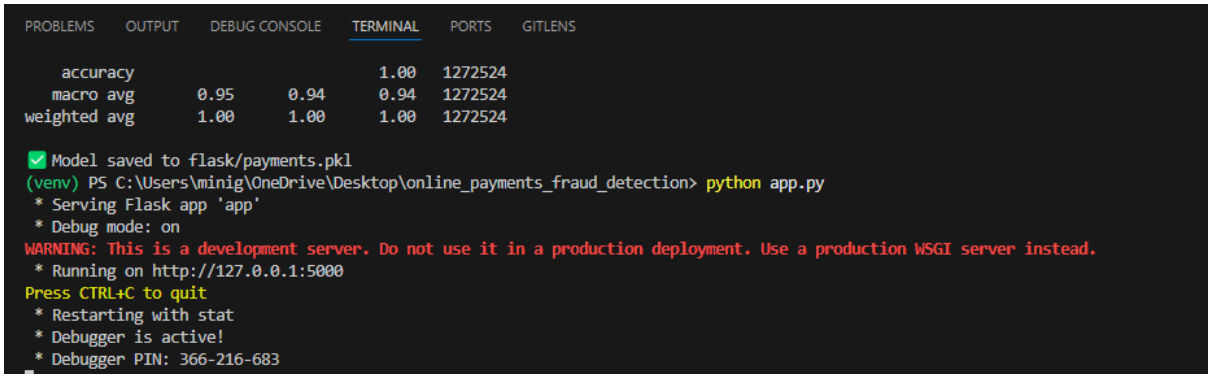
You chose the Decision Tree model due to:

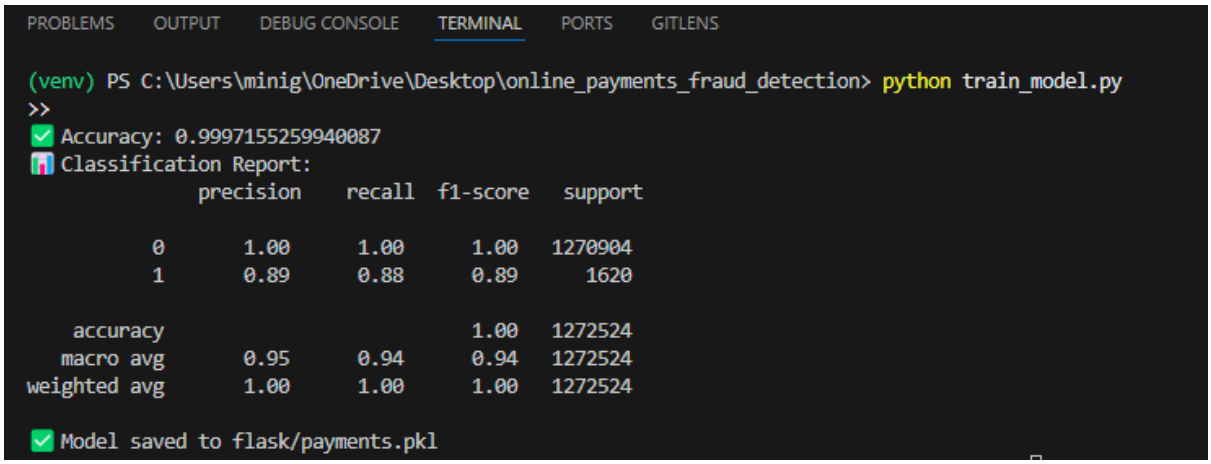
- **Highest accuracy (99.97%)**
- **Interpretability**
- **Fast inference suitable for real-time detection**
- **Simpler than ensemble models, yet highly effective**

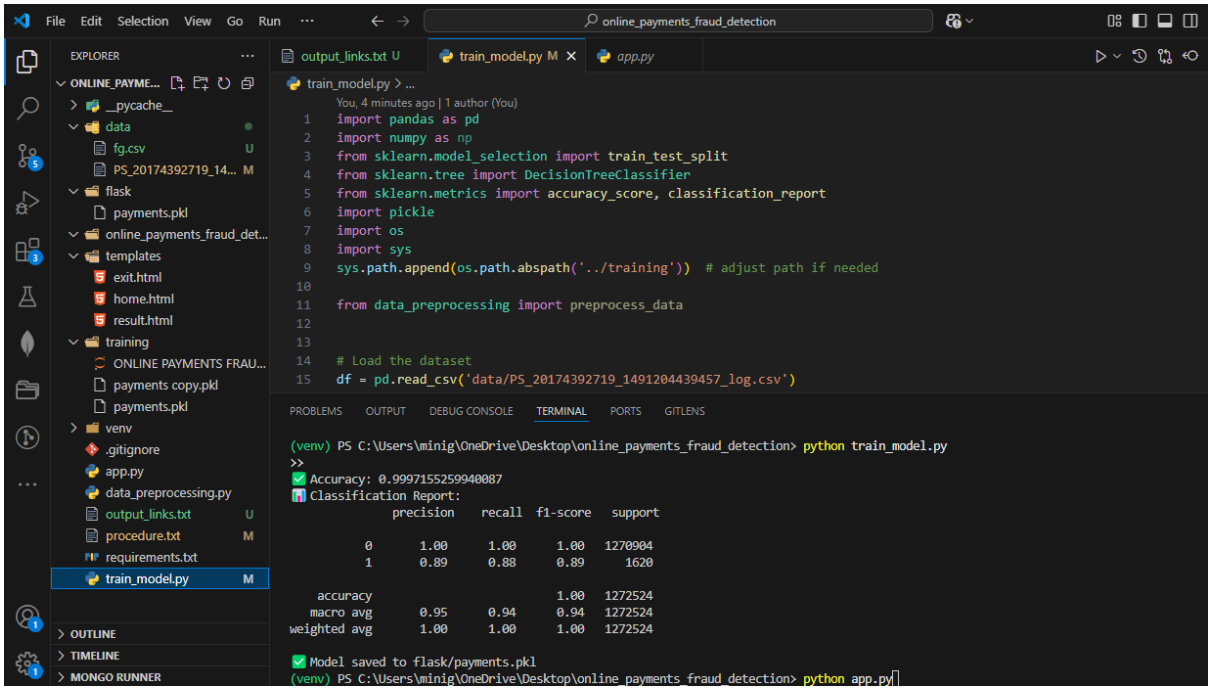
(All available in final_project_code_templates)

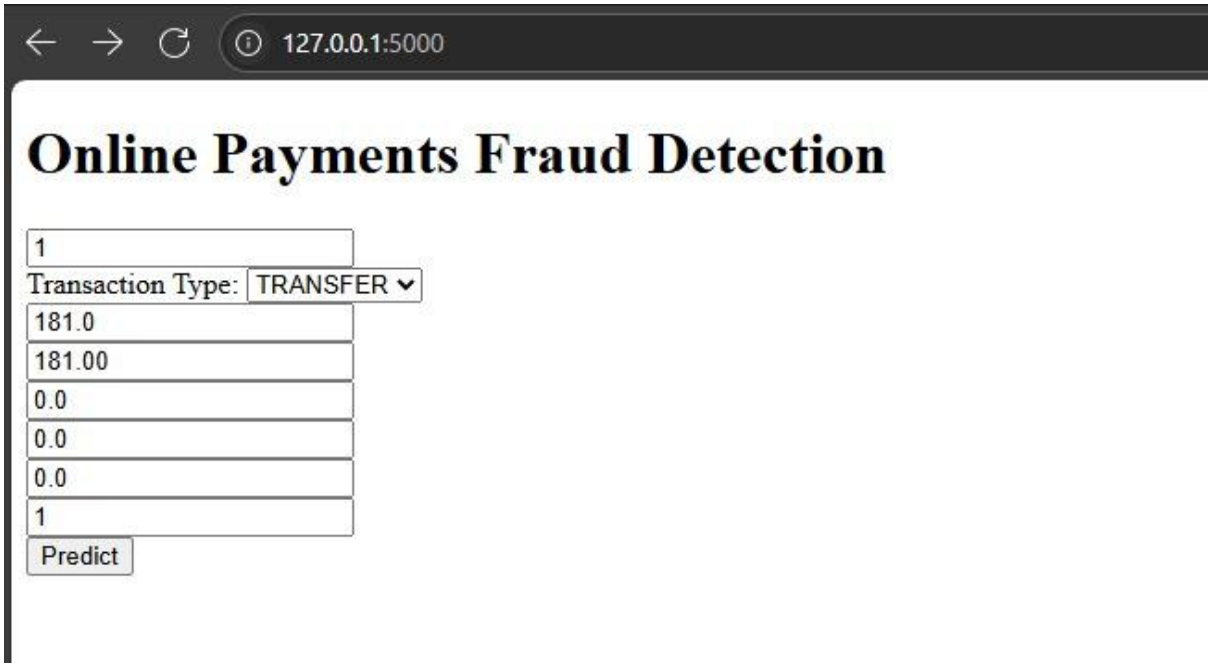
6. Results

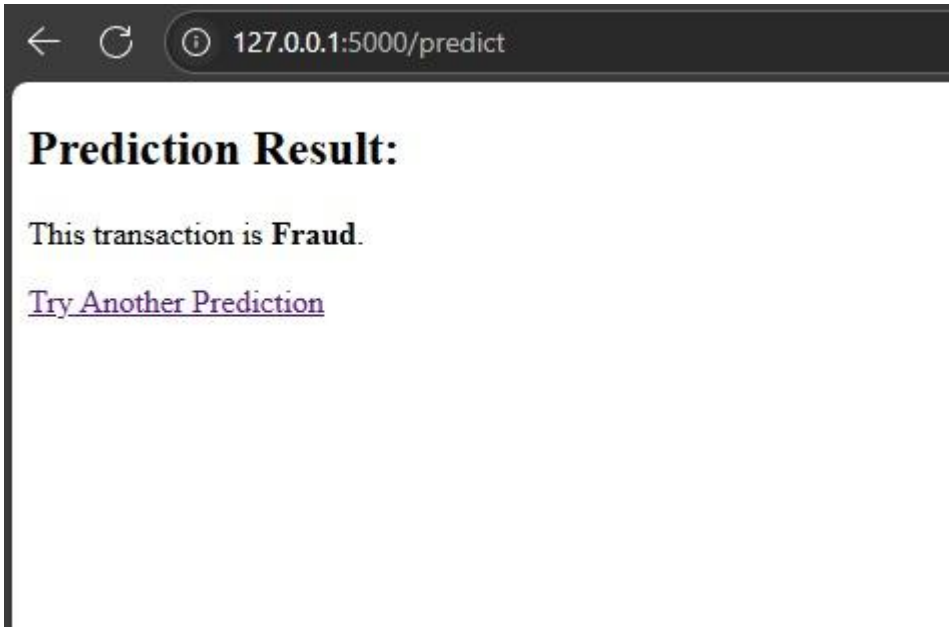
6.1. Output Screenshots

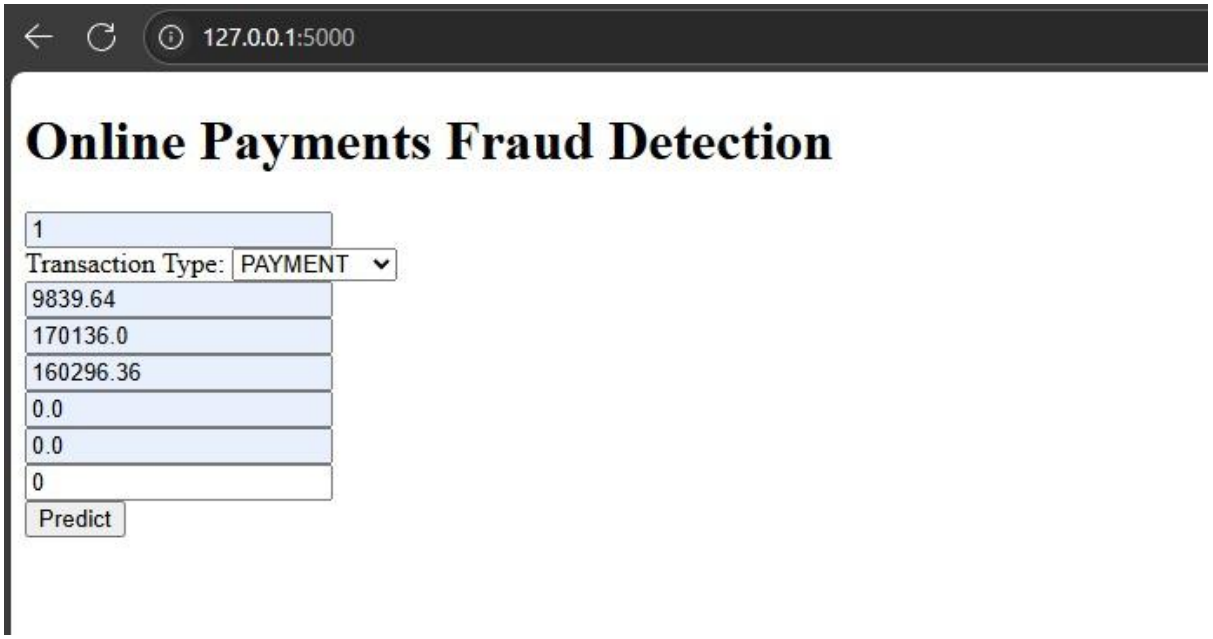

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
accuracy 1.00 1272524
macro avg 0.95 0.94 0.94 1272524
weighted avg 1.00 1.00 1.00 1272524
Model saved to flask/payments.pkl
(venv) PS C:\Users\minig\OneDrive\Desktop\online_payments_fraud_detection> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 366-216-683

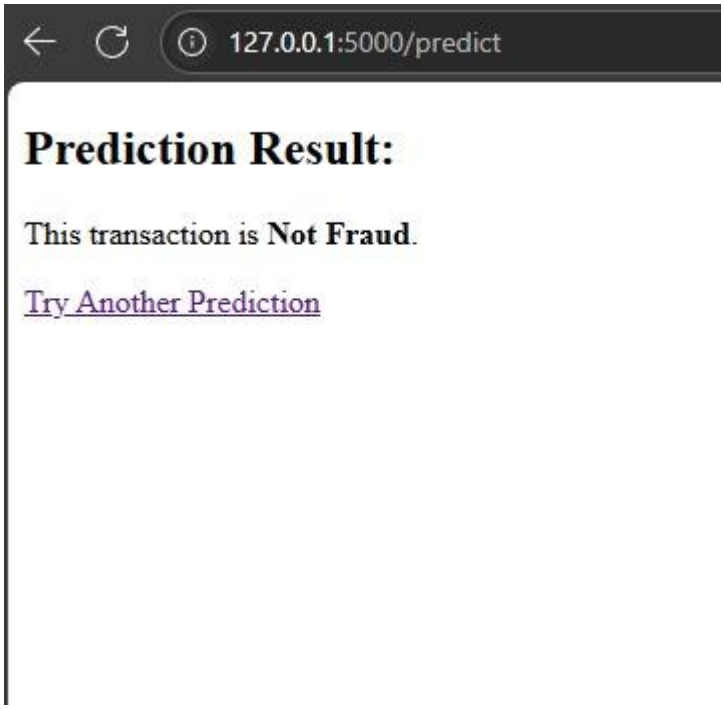

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
(venv) PS C:\Users\minig\OneDrive\Desktop\online_payments_fraud_detection> python train_model.py
>>
Accuracy: 0.9997155259940087
Classification Report:
precision recall f1-score support
0 1.00 1.00 1.00 1270904
1 0.89 0.88 0.89 1620
accuracy 1.00 1272524
macro avg 0.95 0.94 0.94 1272524
weighted avg 1.00 1.00 1.00 1272524
Model saved to flask/payments.pkl


File Edit Selection View Go Run
online_payments_fraud_detection
EXPLORER
ONLINE_PAYME...
__pycache__
data
fg.csv U
PS_20174392719_14... M
flask
payments.pkl
online_payments_fraud_det...
templates
exit.html
home.html
result.html
training
ONLINE PAYMENTS FRAU...
payments copy.pkl
payments.pkl
venv
.gitignore
app.py
data_preprocessing.py
output_links.txt U
procedure.txt M
requirements.txt
train_model.py M
OUTLINE
TIMELINE
MONGO RUNNER
output_links.txt U
train_model.py M
app.py
train_model.py > ...
You, 4 minutes ago | 1 author (You)
1 import pandas as pd
2 import numpy as np
3 from sklearn.model_selection import train_test_split
4 from sklearn.tree import DecisionTreeClassifier
5 from sklearn.metrics import accuracy_score, classification_report
6 import pickle
7 import os
8 import sys
9 sys.path.append(os.path.abspath('../training')) # adjust path if needed
10
11 from data_preprocessing import preprocess_data
12
13
14 # Load the dataset
15 df = pd.read_csv('data/PS_20174392719_1491204439457_log.csv')
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
(venv) PS C:\Users\minig\OneDrive\Desktop\online_payments_fraud_detection> python train_model.py
>>
Accuracy: 0.9997155259940087
Classification Report:
precision recall f1-score support
0 1.00 1.00 1.00 1270904
1 0.89 0.88 0.89 1620
accuracy 1.00 1272524
macro avg 0.95 0.94 0.94 1272524
weighted avg 1.00 1.00 1.00 1272524
Model saved to flask/payments.pkl
(venv) PS C:\Users\minig\OneDrive\Desktop\online_payments_fraud_detection> python app.py


127.0.0.1:5000
Online Payments Fraud Detection
1
Transaction Type: TRANSFER
181.0
181.00
0.0
0.0
0.0
1
Predict


127.0.0.1:5000/predict
Prediction Result:
This transaction is Fraud.
Try Another Prediction


127.0.0.1:5000
Online Payments Fraud Detection
1
Transaction Type: PAYMENT
9839.64
170136.0
160296.36
0.0
0.0
0
Predict



(All available in final_project_code_templates)

7. Advantages & Disadvantages

Advantages

- **High accuracy & recall**
- **Fast inference and computational efficiency**
- **Easy to interpret and explain**

Disadvantages

- **Potential overfitting if overly deep**
- **Less robust to very novel fraud cases**
- **Single-tree model limits ensemble robustness**

8. Conclusion

The project successfully developed a real-time fraud detection model with 99.97% accuracy using a Decision Tree. It balances performance with interpretability and is integrated into a functional web app for demonstration.

9. Future Scope

- **Incorporate ensemble methods (e.g., Random Forest, XGBoost) for robustness**
- **Use real payment gateway APIs for live transaction testing**
- **Deploy as a scalable microservice (e.g., Docker + cloud server)**
- **Add a feedback loop for model retraining with new fraud data**

10. Appendix

10.1. Source Code

- **data_preprocessing.py**
- **train_model.py**
- **app.py (Flask front-end)**
- **requirements.txt**
  (All available in final_project_code_templates)

10.2. GitHub & Project Demo Link

- **GitHub: https://github.com/SHAMINIG2006/final_project_code_templates**
- **App Demo/ Drive Folder: https://1drv.ms/v/c/7f020714fe6065f8/Ea0rfVOUM0pNkz9x0O2E_5gBGR0SCBx_4yk_XSsdeBJo-g?e=KsHw8g**