1. INTRODUCTION

1.1 Project Overview

The Online Payment Fraud Detection System is a Machine Learning-based web application designed to detect fraudulent online financial transactions. With the rapid growth of digital payment platforms such as mobile banking, online transfers, and e-commerce payments, the risk of fraudulent activities has significantly increased.

Fraud detection in financial transactions is a challenging task due to:

- Large volume of transactions
- Highly imbalanced datasets
- Evolving fraud patterns
- Complex transaction behavior

This project builds a predictive model using supervised machine learning algorithms that analyze transaction features such as transaction type, amount, sender balance, and receiver balance to classify transactions as fraudulent or legitimate.

The trained model is deployed using a Flask web application, allowing real-time fraud prediction through a user-friendly interface.

1.2 Purpose

The main objectives of this project are:

- To develop an automated fraud detection system.
- To classify online transactions as fraud or non-fraud.
- To reduce financial losses caused by fraudulent transactions.
- To provide real-time fraud prediction using a web application.
- To compare multiple machine learning algorithms and select the best-performing model.

---

2. IDEATION PHASE

2.1 Problem Statement

Online payment systems are widely used for financial transactions. However, the increasing number of online transactions has also led to an increase in fraudulent activities. Traditional rule-based systems are insufficient to detect complex fraud patterns.

Manual fraud detection is inefficient due to:

- High transaction volume

- Complex fraud patterns
- Time constraints
- Human error

Therefore, there is a need for an intelligent Machine Learning-based system that can automatically detect fraudulent transactions in real-time based on transaction data.

2.2 Empathy Map Canvas

Stakeholders:

- Banks
- Financial institutions
- Payment gateways
- Online users

What Users Need:

- Secure transactions
- Quick fraud detection
- Accurate classification
- Reduced financial loss

User Pain Points:

- Unauthorized transactions
- Account balance draining
- Delayed fraud detection
- Loss of trust in digital platforms

User Goals:

- Safe online payments
- Immediate fraud alerts
- Minimal false alarms

2.3 Brainstorming

Different approaches were considered:

1. Rule-based fraud detection
2. Threshold-based transaction filtering
3. Statistical anomaly detection
4. Supervised Machine Learning models
5. Ensemble learning techniques

After analysis, supervised Machine Learning algorithms were selected due to their ability to learn complex patterns from historical data.

---

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

1. User initiates an online transaction.
2. Transaction details are recorded.
3. System processes the transaction data.
4. ML model evaluates fraud probability.
5. Transaction is classified as fraud or legitimate.
6. Result is displayed instantly.

3.2 Solution Requirements

Functional Requirements:

- Accept transaction input via web form.
- Preprocess user input data.
- Load trained ML model.
- Predict fraud status.
- Display prediction result.
- Save model as .pkl file.

Non-Functional Requirements:

- High accuracy.
- Low response time.
- Secure backend processing.

- User-friendly interface.

3.3 Data Flow Diagram (Text Representation)

User Input
↓
Flask Backend
↓
Data Preprocessing
↓
MachineLearning Model
↓
Prediction Output
↓
Result Display

3.4 Technology Stack

Programming Language:

- Python

Machine Learning Libraries:

- Scikit-learn
- XG Boost

Data Handling:

- Pandas
- NumPy

Visualization:

- Matplotlib
- Seaborn

Backend Framework:

- Flask

Frontend:

- HTML
- CSS

Model Storage:

- Pickle

---

4. PROJECT DESIGN

4.1 Problem–Solution Fit

The problem of online payment fraud involves detecting suspicious transactions from a large volume of financial data. Fraud patterns are often complex, dynamic, and difficult to identify using traditional rule-based systems. Therefore, a Machine Learning-based approach is suitable for this problem. Machine Learning models can learn hidden patterns from historical transaction data and identify unusual behaviors such as full account balance draining, large transfer amounts, or abnormal transaction types. By analyzing features like transaction amount, sender balance, and receiver balance, the system can effectively distinguish between legitimate and fraudulent transactions. This makes Machine Learning an appropriate and scalable solution for fraud detection.

4.2 Proposed Solution

The proposed solution is a Machine Learning-powered fraud detection system integrated with a web application. Initially, the transaction dataset is pre-processed by handling missing values, encoding categorical variables, and managing outliers. The processed data is then divided into training and testing sets. Multiple machine learning algorithms such as Decision Tree, Random Forest, Extra Trees, and XG Boost are trained and evaluated using performance metrics like Accuracy, F1-score, and Confusion Matrix. The best-performing model is selected and saved using Pickle. This trained model is then integrated into a Flask-based web application, where users can input transaction details through an HTML interface. The system processes the input data, applies the trained model, and instantly displays whether the transaction is fraudulent or not.

4.3 Solution Architecture

The architecture of the system consists of three main components: frontend, backend, and machine learning model. The frontend is developed using HTML and CSS, providing a user-friendly interface where users can enter transaction details. The backend is built using Flask, which handles user input, processes the data, loads the trained model, and generates predictions. The machine learning model, trained during the development phase, is stored as a serialized file and loaded whenever a prediction request is made. When a user submits transaction details, the data flows from the frontend to the backend, where it is passed to the trained model for classification. The predicted result is then sent back to the frontend and displayed to the user. This architecture ensures real-time fraud detection with efficient processing and minimal delay.

Frontend (HTML Form)
↓
Flask Backend (Python)
↓
Load Trained ML Model

↓
Predict Fraud
↓
Display Result

]

---

5. PROJECT PLANNING & SCHEDULING

Development Phases:

1. Dataset Collection
2. Data Cleaning & Preprocessing
3. Feature Engineering
4. Model Training
5. Model Evaluation
6. Model Selection
7. Web Application Development
8. Integration & Testing
9. Final Deployment

---

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Functional Testing

Test cases performed:

- Valid transaction input
- Fraud transaction input
- Invalid input handling
- Edge case testing

All functionalities worked as expected.

6.2 Performance Testing

Performance Metrics Used:

- Accuracy Score
- F1 Score
- Precision
- Recall
- Confusion Matrix

Confusion matrix helps evaluate:

- True Positives (Correct Fraud Detection)
- False Positives
- False Negatives
- True Negatives

Due to class imbalance, F1-score was considered more reliable than accuracy.

---

7. RESULTS

The models trained:

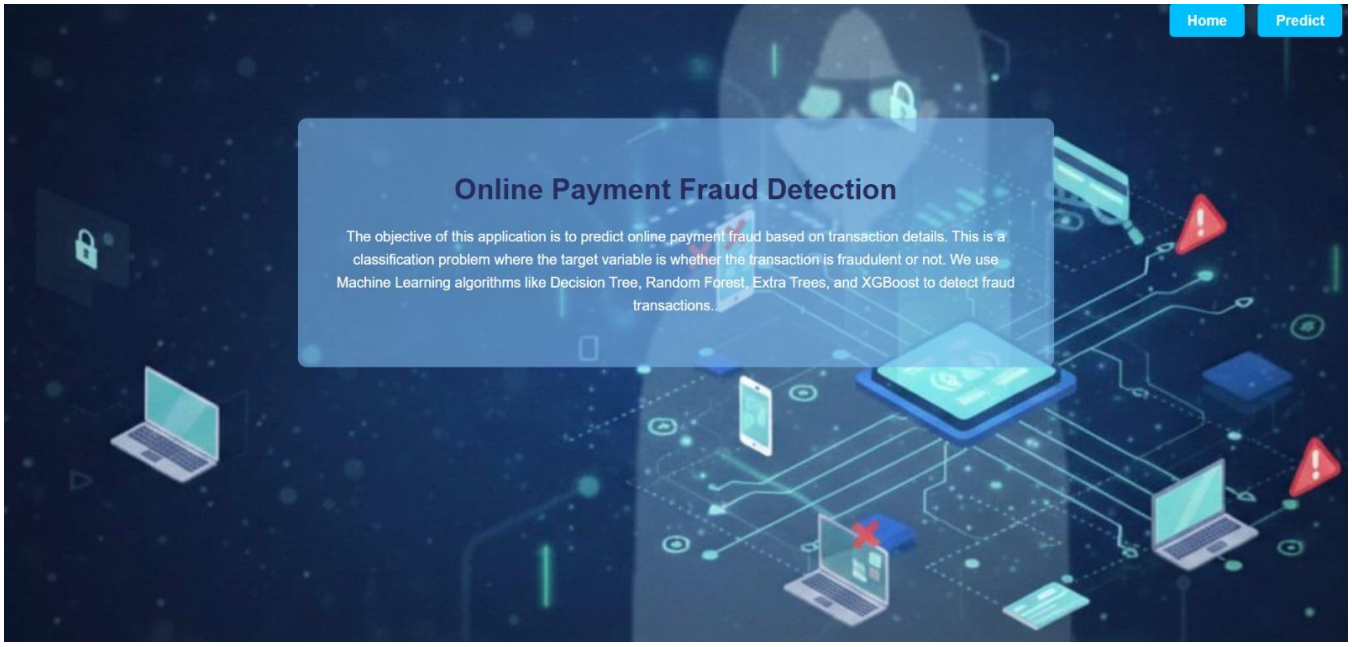
- Decision Tree
- Random Forest
- Extra Trees
- XG Boost

The best-performing model was selected based on accuracy and F1-score.

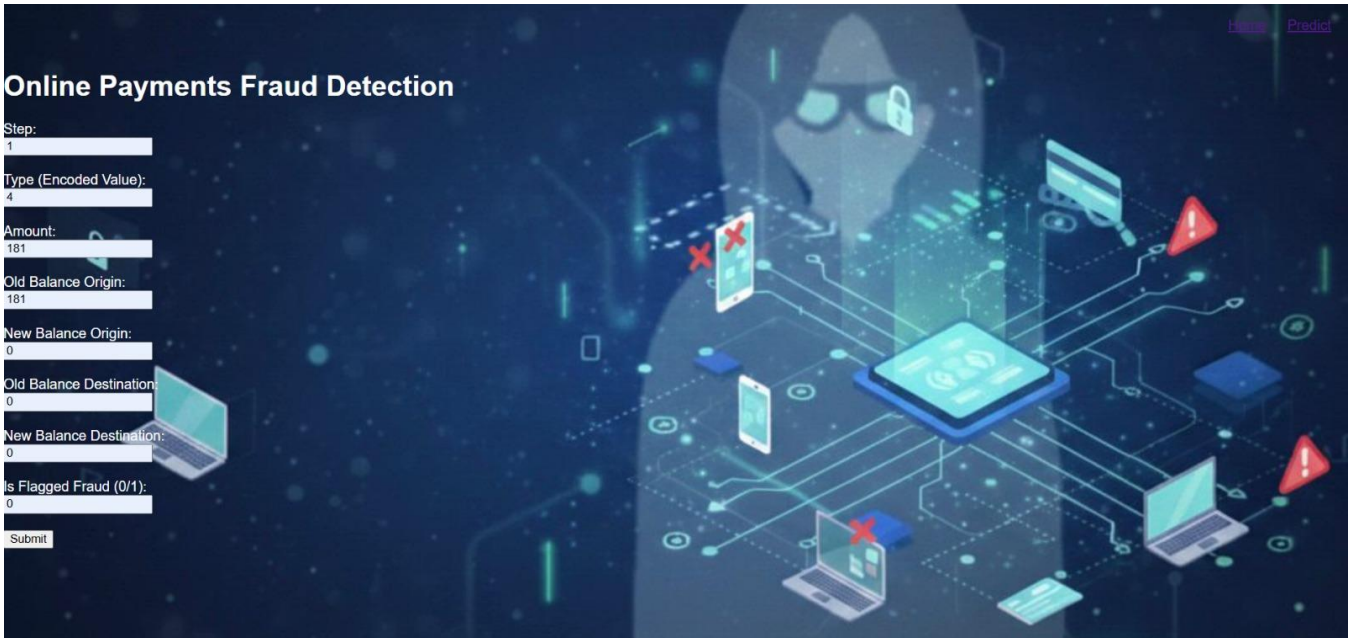
The model successfully classified fraudulent transactions with high performance and minimal false negatives.

7.1 Output Screenshots

- Home Page

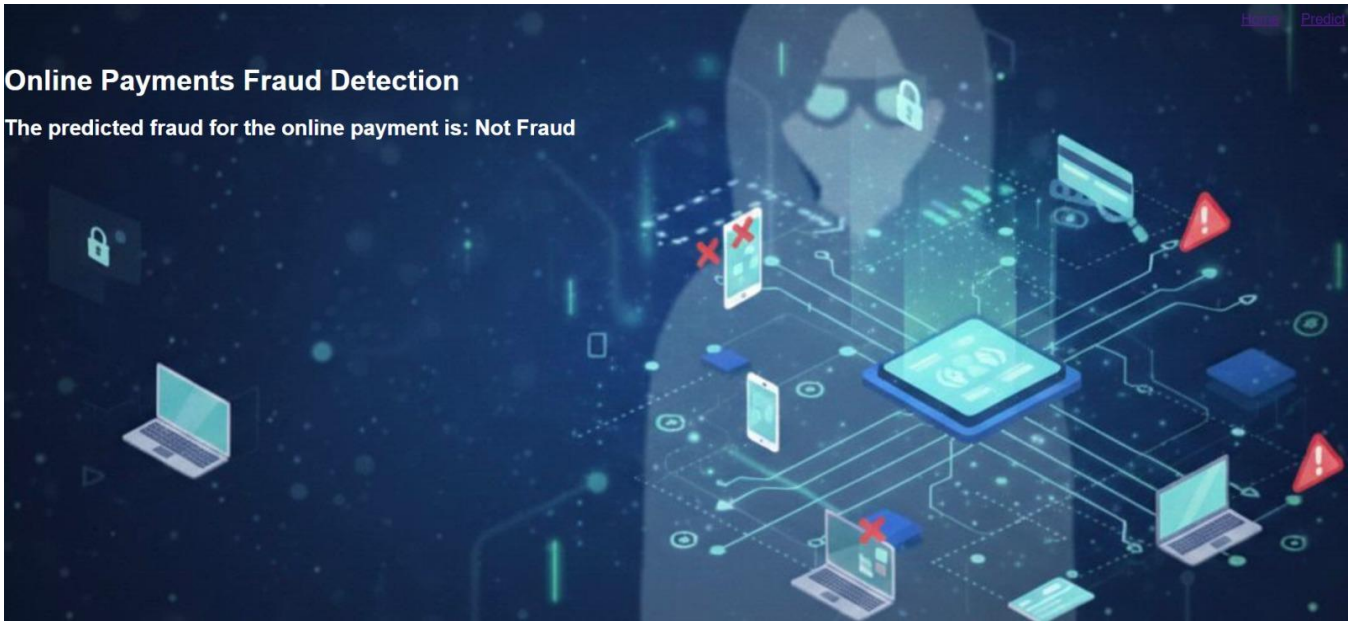


- Prediction Form



- Fraud Prediction Result

- Non-Fraud Prediction Result



---

8. ADVANTAGES & DISADVANTAGES

Advantages:

- Real-time fraud detection
- Automated decision-making
- Reduced manual effort
- Improved financial security
- High scalability

Disadvantages:

- Requires large dataset
- Imbalanced data challenges
- Model performance depends on data quality
- Requires regular retraining

---

9. CONCLUSION

The Online Payment Fraud Detection System successfully demonstrates the application of Machine Learning in detecting fraudulent financial transactions. By integrating multiple ML algorithms and deploying the best-performing model using Flask, the system provides real-time fraud prediction.

This project improves transaction security and reduces potential financial losses in digital payment systems.

---

10. FUTURE SCOPE

- Deploy system on cloud (AWS, Azure, Render)
- Implement SMOTE for better imbalance handling
- Add Deep Learning models (ANN)
- Add fraud probability score
- Develop API-based integration with banking systems
- Implement user authentication

---

11. APPENDIX

- Dataset Link: https://www.kaggle.com/datasets/rupakroy/online-payments-fraud-detection-dataset
- GitHub & Project Demo Link: https://github.com/VYashasweeni/Online-Payment-Fraud-Detection-using-ML & https://drive.google.com/file/d/1703W65DMMs0MdYuzV7-o9YETZtOmocPD/view?usp=sharing