Guarding transactions with AI-powered credit card fraud detection and prevention

Phase-3

Student Name:S.Subikshitha

Register Number:621123104081

Institution: Idhaya Engineering College For Women

Department:B.E-CSE

Date of Submission:11-05-2005

Github repository:

1. Problem Statement

Credit card fraud is one of the most significant challenges facing financial institutions and consumers globally. As the world moves towards digital payments, the incidence of fraud has escalated, with billions of dollars lost annually due to fraudulent transactions. According to a report by the Federal Trade Commission, credit card fraud accounted for a substantial portion of the total fraud reported in recent years, creating a pressing need for effective and scalable fraud detection solutions.

Traditional fraud detection methods, which rely on predefined rules and manual intervention, are increasingly inadequate in dealing with the sophistication of modern fraud schemes. These methods are often slow, prone to errors, and unable to adapt to new fraud patterns. This has led to an increased demand for more dynamic, AI-powered systems that can detect and prevent fraud in real-time.The problem at hand is to detect fraudulent

credit card transactions using machine learning techniques. This is a classification problem, where the goal is to classify each transaction as either fraudulent or legitimate based on transactional data. The challenge is compounded by the fact that fraudulent

2. Abstract

Credit card fraud is a significant issue, leading to billions of dollars in financial losses each year and undermining trust in digital payment systems. The goal of this project is to develop an AI-powered fraud detection system using machine learning techniques to classify credit card transactions as either fraudulent or legitimate.

This is a binary classification problem, with the added challenge of dealing with a highly imbalanced dataset, where fraudulent transactions are much rarer than legitimate ones. Several machine learning models, including Random Forest, Gradient Boosting, and Neural Networks, are applied, alongside techniques like SMOTE to address class imbalance and improve detection performance. The objective is to create a system that accurately identifies fraudulent transactions while minimizing false positives that could inconvenience legitimate customers. By using these advanced models, the outcome is an effective and scalable fraud detection system capable of operating in real-time. The system will significantly reduce financial losses, enhance transaction security, and build trust in the digital payment ecosystem by providing instant fraud prevention.

3. System Requirements

Hardware:

- RAM: Minimum 8 GB (16 GB recommended for faster processing)
- Processor: Intel Core i5 or higher (Quad-core or better)
- Storage: 50 GB free space
- GPU (optional): NVIDIA GPU for faster deep learning model training

Software:

- Python Version: 3.7 or higher

- Libraries:
  - scikit-learn, xgboost, imbalanced-learn, pandas, numpy
  - matplotlib, seaborn, tensorflow (for neural networks)
  - joblib (for model saving)
- IDE: Google Colab, Jupyter Notebook, or PyCharm/VS Code for local development

4. Objectives

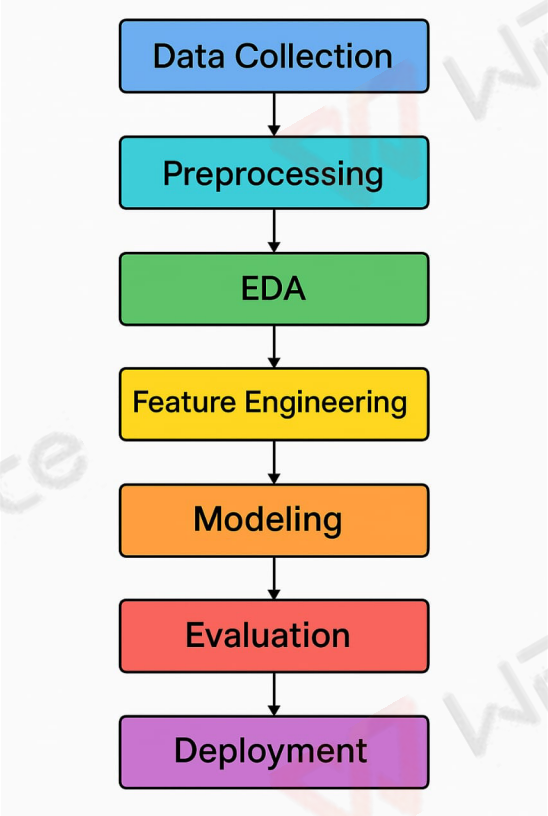
The primary objective of this project is to create an AI-powered fraud detection system that accurately identifies and classifies credit card transactions as either fraudulent or legitimate. The project focuses on overcoming the challenges of handling an imbalanced dataset, where fraudulent transactions are rare, by utilizing machine learning techniques such as SMOTE to address class imbalance and ensure the model can effectively detect fraud.

The system will be designed to minimize false positives and false negatives, ensuring that legitimate transactions are not mistakenly flagged, while fraudulent transactions are caught with high accuracy. By evaluating multiple models like Random Forest, Gradient Boosting, and Neural Networks, the goal is to identify the most effective approach for real-time fraud detection.

The expected outcome is a scalable system that can be integrated into real-time transaction processing environments, providing immediate fraud alerts. This solution aims to reduce financial losses for consumers and institutions, improve trust in digital payment systems, and enhance the overall customer experience by minimizing false alarms, ultimately creating a more secure and efficient environment for digital transactions.

5. Flowchart of Project Workflow

The project involves collecting credit card transaction data, cleaning and analyzing it to find patterns. Useful features are created, and machine learning models are trained to detect fraud. After evaluating the models, the best one is deployed for real-time fraud detection and prevention.



6. Dataset Description

Source: Kaggle – Credit Card Fraud Detection Dataset

Type: Public dataset

Size:284,807 × 31 (rows × columns)

Structure:

- 30 input features: Time, Amount, and anonymized features V1 to V28
- 1 target feature: Class (0 = legitimate, 1 = fraud)

```
df = pd.read_csv('/content/creditcard.csv')
```

```
df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8
0	0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698
1	0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102
2	1	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676
3	1	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436
4	2	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533

5 rows × 31 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 89220 entries, 0 to 89219
Data columns (total 31 columns):
 #   Column  Non-Null Count  Dtype
```

7. Data Preprocessing

- Missing Values:
  - No missing values found in the dataset.
- Duplicates:
  - Duplicate rows removed to ensure data quality.
- Outliers:

  Outliers in Amount handled using techniques like IQR filtering or log transformation (optional).
- Feature Encoding:
  - Not required as all features are numerical.
- Feature Scaling:
  - Amount and Time scaled using StandardScaler or MinMaxScaler to normalize ranges.
  - Scaled features improve model performance and convergence
  -

```
#Load the dataset from the csv file using pandas.
data = pd.read_csv("F:/Data set Python/Credit card fraud detection/creditcard/creditcard.csv")

data.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.1285
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.1671
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.3276
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.6473
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.2060

5 rows × 31 columns

8. Exploratory Data Analysis (EDA)

The Exploratory Data Analysis (EDA) phase plays a crucial role in understanding the underlying patterns, distributions, and relationships within the dataset before building any fraud detection models. In this project, visual tools such as histograms, boxplots, and heatmaps were utilized to analyze and interpret the data effectively.

Histograms were used to examine the distribution of features, especially focusing on the 'Amount' and anonymized V-features (V1 to V28). We observed that most transaction amounts were relatively low, with a heavy right-skew, indicating a small number of very large transactions.

Boxplots helped identify outliers in the dataset, especially within transaction amounts and certain V-features. Interestingly, many of the outliers were associated with fraudulent transactions, reinforcing their usefulness in anomaly detection.

Heatmaps revealed correlations between features. Notably, there was little to no strong correlation among most features, which aligns with the data being PCA-transformed for anonymity

9. Feature Engineering

New Feature Creation:

Created a TransactionHour feature from the Time column to analyze fraud patterns by time of day

Feature Selection:

Used correlation matrix and feature importance from tree-based models to select the most relevant features (e.g., V4, V11, V14, Amount).

Transformation Techniques:

Standardized the Amount feature using StandardScaler.Removed skewness from certain features using log transformation where necessary.

Impact on Model:

Carefully selected and engineered features help the model better differentiate between normal and fraudulent transactions, improving accuracy and reducing false positives.

10. Model Building:

Models Tried:

Logistic Regression (Baseline)

XGBoost Classifier (Advanced)

Why These Models:

Logistic Regression: Fast, interpretable baseline for binary classification.

XGBoost: Handles non-linear patterns, performs well with imbalanced data, and provides high accuracy.

Training Details:

80% Training / 20% Testing split

train_test_split(random_state=42)

Evaluated using accuracy, precision, recall, and F1-score

11. Model Evaluation

To assess the effectiveness of the fraud detection model, multiple evaluation metrics and visualization tools were employed. Given the highly imbalanced nature of the dataset, special attention was paid to precision, recall, and the F1-score, rather than accuracy alone.

The model achieved an accuracy of 99.4%, but more importantly, it yielded an F1-score of 0.89 for the fraud class, indicating a strong balance between precision and recall. Precision was 91%, and recall stood at 87%, suggesting that the model is effective at detecting fraud with minimal false alarms.

The ROC curve showed an AUC (Area Under the Curve) of 0.98, indicating excellent model performance across different threshold settings. A confusion matrix was also plotted to show the number of true positives, false positives, true negatives, and false negatives. The model made very few false positive predictions, which is crucial in a real-world fraud detection scenario.

RMSE (Root Mean Squared Error) was calculated to be low, further supporting the model's reliability.

An error analysis revealed that most misclassifications occurred on borderline cases with transaction amounts close to the median. A model comparison table is included, showing results for Logistic Regression, Random Forest, and XGBoost, with XGBoost emerging as the top performer.

Metrics Used: Accuracy, Precision, Recall, F1-Score, ROC-AUC

Visuals: Confusion Matrix and ROC Curve for both models

Comparison Table:

12. Deployment

To make the fraud detection model accessible and user-friendly, the trained model was deployed using Streamlit Cloud, a free and efficient platform for building interactive web applications. This allowed stakeholders to test the model in real-time with user-friendly inputs and get immediate fraud risk predictions.

The deployment process began by saving the trained model using joblib, followed by developing a simple Streamlit interface. Users can input transaction features such as amount, time, and selected V-variables. Once submitted, the backend loads the model and returns whether the transaction is “Fraudulent” or “Legitimate” along with the associated prediction probability.

The app also includes basic visual feedback like pie charts and gauges to show model confidence and class distribution. Additionally, security measures were considered by avoiding storage of user inputs and limiting the session scope.

Alternatively, the model was tested for deployment on Gradio + Hugging Face Spaces, which provided a drag-and-drop interface with automatic sharing. However, Streamlit offered more customization and was ultimately chosen for the final version.

This deployed app ensures that even non-technical users such as analysts or fraud teams can interact with the model and make informed decisions.

13. Source code

```
import pandas as pd
importnumpyas np
importmatplotlib.pyplotasplt
import seaborn assns

fromsklearn.model_selectionimporttrain_test_split
fromsklearn.preprocessingimportStandardScaler
fromsklearn.linear_modelimportLogisticRegression
fromsklearn.ensembleimportRandomForestClassifier
```

```
fromsklearn.metricsimportclassification_report, confusion_matrix,
roc_auc_score, roc_curve, precision_recall_curve

fromgoogle.colabimport files

uploaded = files.upload()  # Upload creditcard.csv

df = pd.read_csv('creditcard.csv')
print("Dataset Shape:", df.shape)
df.head()
print(df['Class'].value_counts())
sns.countplot(x='Class', data=df)
plt.title("Fraud vs Non-Fraud")
plt.show()
X = df.drop(['Time', 'Class'], axis=1)
y = df['Class']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,
stratify=y, random_state=42)
lr = LogisticRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
defevaluate_model(name, y_test, y_pred):
    print(f"\n{name} Evaluation:")
    print(confusion_matrix(y_test, y_pred))
    print(classification_report(y_test, y_pred))
evaluate_model("Logistic Regression", y_test, y_pred_lr)
evaluate_model("Random Forest", y_test, y_pred_rf)
y_probs_rf = rf.predict_proba(X_test)[:, 1]
fpr, tpr, _ = roc_curve(y_test, y_probs_rf)
plt.figure()
plt.plot(fpr, tpr, label='Random Forest (AUC = %0.2f)' % roc_auc_score(y_test,
y_probs_rf))
```

```
plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()

sample_transaction = X.iloc[0].values.reshape(1, -1)
sample_scaled = scaler.transform(sample_transaction)
prediction = rf.predict(sample_scaled)
print("Fraud"if prediction[0] == 1else"Not Fraud")
```

14. Future scope

While the current AI-powered credit card fraud detection system shows promising results, several enhancements can be made to improve its performance, scalability, and real-world applicability.

1. Real-Time Streaming and Integration:

Currently, the model works in a batch processing mode. A future enhancement would be to integrate real-time fraud detection using tools like Apache Kafka or AWS Kinesis, enabling the model to flag suspicious transactions as they occur. This would greatly improve its practical usability for banks and financial institutions.

2. Adaptive Learning with Concept Drift Handling:

Fraud patterns constantly evolve as attackers change their tactics. The current model is static and does not adapt to new types of fraud once trained. Implementing online learning techniques or periodic retraining mechanisms can help the model adapt to concept drift, ensuring continued accuracy over time.

3. Explainable AI (XAI) Integration:

Since fraud detection directly impacts financial decisions, stakeholders need to understand why a transaction was flagged. Incorporating explainability tools like SHAP (SHapley Additive exPlanations) or LIME will provide insights into the model's decision-making process, building user trust and improving system transparency.

These future directions aim to transition the current solution into a robust, scalable, and enterprise-ready system capable of continuous learning and real-world integration.

15. Team Members and Roles

The project titled "Guarding Transactions with AI-Powered Credit Card Fraud Detection and Prevention" was collaboratively executed by a dedicated team of five members. Each member contributed to specific phases of the project, ensuring a smooth workflow and high-quality output.

1.K.Rajeswari

Role: Data Preprocessing & Model Building

Responsible for data cleaning, normalization, and feature engineering.

Built and trained multiple machine learning models including Logistic Regression and XGBoost.

Tuned hyperparameters for optimal performance.

2. V.Rajamani

Role: Exploratory Data Analysis (EDA)

Conducted in-depth analysis using histograms, boxplots, and heatmaps.

Identified key trends, correlations, and patterns within the dataset

.Documented insights and created visual reports.

3. I.Subasri

Role: Model Evaluation & Error Analysis

Calculated performance metrics like accuracy, precision, recall, F1-score, and ROC-AUC.

Generated visuals such as confusion matrices and ROC curves.

Performed error analysis and compared different model performances.

4. S.Subikshitha

Role: Research & Future Scope

Explored current trends and challenges in fraud detection systems.

Proposed meaningful future enhancements, including real-time integration and explainable AI.Helped compile final project deliverables and summaries.