

**Student Name: P. Prakash**

**✔ Register Number: 622423243017**

**✔ Institution: Salem college of engineering and technology**

**✔ Department: Artificial Intelligent and data Science**

**✔ Date of Submission: 19.05.2025**

**✔ GitHub Repository Link:** [***https://github.com/prakash2323/Prakash.git***](https://github.com/prakash2323/Prakash.git)

**Project Title:**

**Guarding Transactions – AI-Powered Credit Card Fraud Detection and Prevention**

### **PHASE-3:**

### **1. Problem Statement**

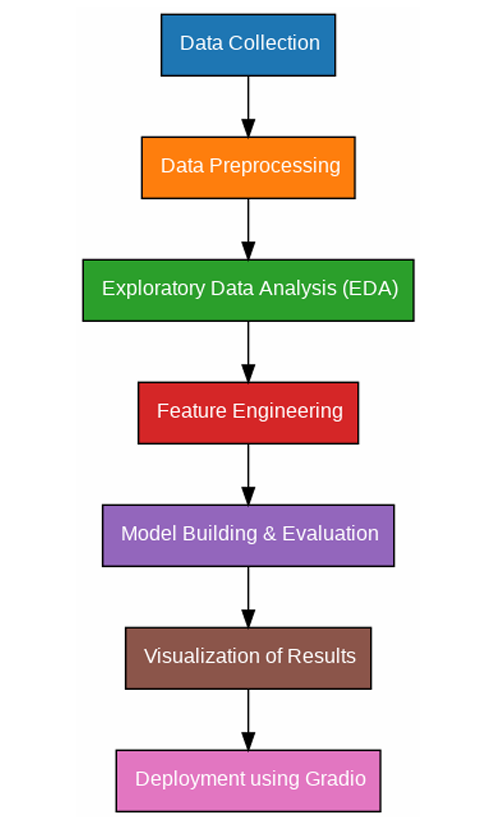
With the rapid growth of digital transactions, **credit card fraud** has become a major threat to both consumers and financial institutions. Fraudulent activities not only cause financial losses but also undermine trust in digital payments.

This project aims to build a **machine learning-powered fraud detection system** that can **identify and prevent suspicious transactions in real-time**. The system uses **AI algorithms** to learn patterns of genuine and fraudulent behaviors based on transaction data, geography, time, and user habits.

### **2. Project Objectives**

* Develop an **AI-based fraud detection model** using supervised machine learning.
* Analyze and extract features such as **transaction amount, time, location, device** behavior, etc.
* Mitigate **false positives** while improving fraud detection **accuracy** and **recall**.
* Explore the impact of **imbalanced datasets** and apply techniques like **SMOTE** or **undersampling**.
* Deploy a **Gradio-based interface** for users or analysts to test transaction records.

### **3. Flowchart of the Project Workflow**



### **4. Data Description**

* **Dataset**: Credit Card Fraud Detection Dataset (e.g., from Kaggle or anonymized bank data)
* **Source**: Kaggle - European cardholder dataset
* **Records**: ~284,000 transactions, 492 fraud cases (~0.17%)
* **Features**:
  + Time, Amount, anonymized variables (V1 to V28), Class
* **Target Variable**:
  + Class: 0 (Legit), 1 (Fraud)
* **Type**: Structured, tabular, anonymized data

### **5. Data Preprocessing**

* **Missing Values Handling**: Confirm dataset integrity.
* **Feature Scaling**: Normalize Amount and Time using StandardScaler.
* **Imbalanced Class Handling**:
  + Applied **SMOTE** (Synthetic Minority Oversampling Technique).
* **Train-Test Split**: Stratified split to maintain class distribution.
* **Outlier Removal (Optional)**: Isolation Forest or IQR-based trimming.

### **6. Exploratory Data Analysis (EDA)**

**Univariate Analysis**:

* Distribution of transaction amount and time
* Histogram of Amount by class

**Bivariate Analysis**:

* Correlation heatmap of PCA components
* Boxplots of feature importance across fraud vs legitimate cases

**Key Insights**:

* Fraudulent transactions often have **lower amounts**
* **Certain PCA components (V10, V12, V14)** are strongly associated with fraud

### **7. Feature Engineering**

* **Temporal Features**: Hour of transaction, time gaps between transactions
* **User Patterns**: Frequency of transactions per card per hour
* **Statistical Aggregates**: Rolling mean, std of transaction values
* **Anomaly Scores**: Isolation forest or One-Class SVM outputs as meta-features

### **8. Model Building**

**Algorithms Used**:

* Logistic Regression (Baseline)
* Random Forest
* XGBoost (Best performance)
* Optional: Autoencoder or LSTM for sequence anomaly detection

**Evaluation Metrics**:

* **F1-Score**, **Precision**, **Recall** (Important for imbalance)
* ROC-AUC Curve
* Confusion Matrix analysis

**Validation**:

* Stratified K-Fold Cross Validation

### **9. Visualization of Results & Model Insights**

* **Feature Importance**: Plotted using SHAP values
* **Confusion Matrix**: To show trade-off between False Positives and False Negatives
* **ROC Curves**: To compare models
* **Precision-Recall Curves**: Effective under class imbalance

### **10. Tools and Technologies Used**

|  |  |
| --- | --- |
| **Category** | **Tools/Libs Used** |
| Programming | Python 3 |
| Libraries | pandas, numpy, matplotlib, seaborn |
| ML Models | scikit-learn, xgboost, imbalanced-learn |
| NLP (optional) | nltk, spacy for metadata |
| Deployment | Gradio for interactive interface |
|  | Google Colab / Jupyter Notebook |

**12.Source Code**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix

# Step 1: Load the dataset from CSV

file\_path = "creditcard\_sample.csv" # Make sure this path is correct

df = pd.read\_csv(file\_path)

# Step 2: Inspect class distribution (fraud vs legitimate)

print("Class distribution:\n", df['Class'].value\_counts())

# Step 3: Split features and labels

X = df.drop("Class", axis=1)

y = df["Class"]

# Step 4: Split dataset into train and test sets (stratified to maintain class balance)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=42, stratify=y

)

# Step 5: Train Random Forest model

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

# Step 6: Predict on test set

y\_pred = model.predict(X\_test)

# Step 7: Evaluate model performance

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred, zero\_division=0))

print("\nConfusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

# Step 8: Example - check a single transaction

def check\_transaction(transaction):

pred = model.predict([transaction])

return "Fraudulent" if pred[0] == 1 else "Legitimate"

# Example transaction from test set

sample\_tx = X\_test.iloc[0].values

result = check\_transaction(sample\_tx)

print(f"\nSample transaction prediction: {result}")

### **13. Team Members and Contributions**

* **P.Prakash** – Data preprocessing, EDA
* **P.Prakash** – Feature engineering, modeling
* **Rithik** – Evaluation, SHAP analysis, Gradio UI

Would you like this as a downloadable PDF or formatted for GitHub README?