Phase-2

Student Name: Gokul S

Register Number: 410723106009

Institution: Dhanalakshmi College of Engineering

Department: Electronics and Communication Engineering

Date of Submission: 07-05-2025

Github Repository Link: https://github.com/Gokul7881/NM_Gokul

---

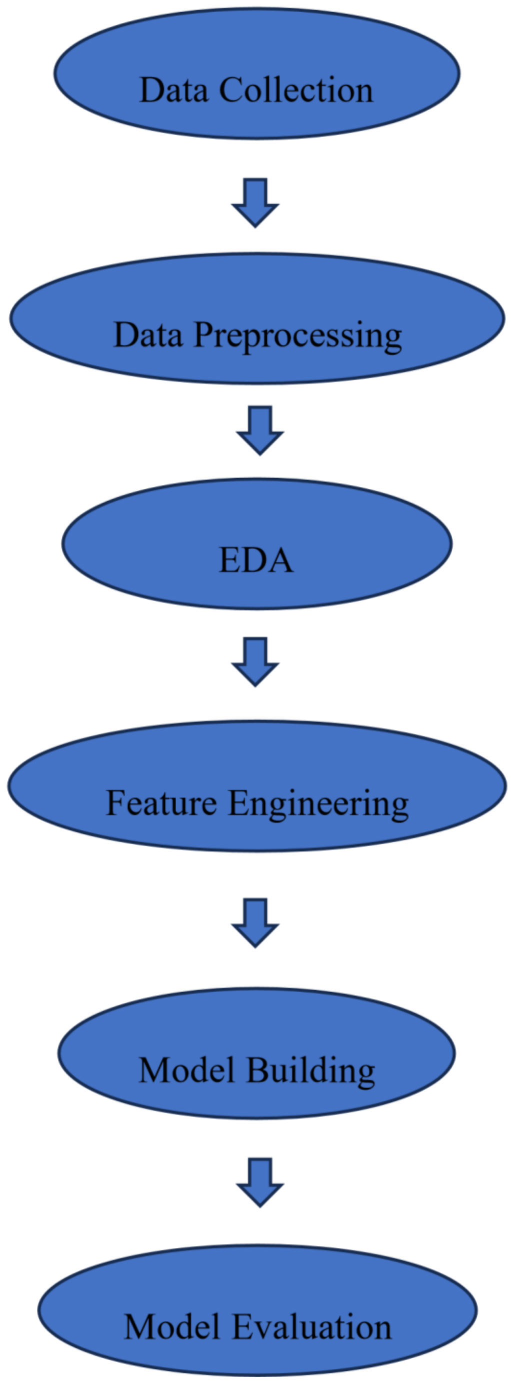
1. Problem Statement

Credit card fraud causes significant financial loss and erodes trust in digital transactions. Using AI-powered methods, especially classification models, we aim to identify fraudulent transactions in real time based on transaction patterns. This is a binary classification problem (fraud vs. not fraud). By automating fraud detection, we can significantly reduce manual monitoring and improve response time.

2. Project Objectives

1. Build models that can accurately classify transactions as fraudulent or legitimate.
2. Focus on maximizing recall to avoid false negatives (i.e., missing a fraud).
3. Ensure real-world applicability and interpretability (e.g., feature importance).
4. Adjust objectives based on EDA insights, especially dealing with class imbalance..

3. Flowchart of the Project Workflow



4. Data Description

Dataset Name and Origin:

https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

The dataset used is the "Credit Card Fraud Detection" dataset from Kaggle.

Type of Data: Structured, tabular data.

Number of Records and Features:

The dataset contains 284,807 transactions with 30 features including anonymized features V1 to V28, Time, Amount, and the target variable Class.

Static or Dynamic Dataset: Static dataset.

Target Variable: Class (0 = Not Fraud, 1 = Fraud).

5. Data Preprocessing

Missing Values: No missing values were found in the dataset.

Duplicate Records: Duplicate rows were checked and removed if present.

Outliers: Detected using boxplots; outliers in Amount were handled using log transformation.

```
from sklearn.preprocessing import StandardScaler
data['norm_amount'] =
StandardScaler().fit_transform(data['Amount'].values.reshape(-1,1))
data['norm_time'] = StandardScaler().fit_transform(data['Time'].values.reshape(-
1,1))
data.drop(['Amount', 'Time'], axis=1, inplace=True)
```

Data Types: All features are numeric. No conversion needed.
Encoding Categorical Variables: Not required as all features are already numerical.
Normalization: Amount and Time were scaled using **StandardScaler** to bring them on the same scale as V1–V28.

6. Exploratory Data Analysis (EDA)

Univariate Analysis

- Class is highly imbalanced: only **0.17%** of transactions are fraudulent.
- Distribution of Amount is right-skewed; normalization improves this.
- Features V1–V28 follow near-Gaussian distributions due to PCA transformation.

Bivariate/Multivariate Analysis

- **Correlation Matrix**: No highly correlated independent features.
- Fraudulent transactions show distinguishable patterns in features like V14, V17, V10, and V12.
- Scatter plots reveal that certain features have strong separability between fraud and non-fraud.

Insights Summary

- **Class imbalance** is critical; resampling is needed. •
- Features V14, V10, and V17 show strong influence on predicting fraud.

7. Feature Engineering

- **New Features**: ○ Created norm_time and norm_amount.
- **Feature Reduction**: ○ No dimensionality reduction applied due to prior PCA.
- **Domain Knowledge**:
    - Used statistical insights (e.g., top feature importance from models) to fine-tune.

8. Model Building **Models**

Selected:

- **Logistic Regression**: Simple baseline, interpretable.
- **Random Forest Classifier**: Robust to overfitting, handles imbalance well.

Justification:

Both models are well-suited for binary classification with imbalanced data.

Data Split:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y,
random_state=42)
```

Performance Metrics:

○ Accuracy, Precision, Recall, F1-Score, and AUC. ○
Special focus on **Recall** (fraud detection sensitivity).

9. Visualization of Results & Model Insights

Confusion Matrix: Helps analyze type I and II errors.

ROC Curve: Compared AUC for both models (Random Forest had higher AUC).

Feature Importance Plot (from Random Forest):

- Top predictors: V14, V10, V17, V12.

Conclusion:

- Random Forest outperformed Logistic Regression in recall and AUC.
- Visuals confirmed the model captures key fraudulent transaction patterns.

10. Tools and Technologies Used

- **Programming Language**: Python
- **IDE/Notebook**: Google Colab, Jupyter Notebook

- **Libraries**:

    ○ **Data Handling**: pandas, numpy ○

    Visualization: matplotlib, seaborn ○ **Modeling**:

    scikit-learn, imbalanced-learn ○ **Model**

    Evaluation: scikit-learn metrics, plotly

- **Version Control**: GitHub

11. Team Members and Contributions

NAME	ROLE	RESPONSIBLE
Inbarasu I	Member	Data Collection, Data Preprocessing
Deepak kumar S	Member	Feature Engineering
Hemanth D	Member	Exploratory Data Analysis (EDA),
Gokul S	Leader	Model Building, Model Evaluation