Phase-3 Submission Template

Student Name: Jayapratha.A

Register Number: 422723104047

Institution: V.R.S College of engineering and

Department: Computer science engineering

Date of Submission: 17-05-2025

GithubRepositorLink:https://github.com/Juiena-oss/Juiena.git

1. Problem Statement

Credit card fraud is a major concern in the financial industry, causing billions in losses each year. The objective is to develop a classification model that accurately distinguishes between legitimate and fraudulent transactions. Due to the rarity of fraud cases, this is an imbalanced binary classification problem.

2. Abstract

This project focuses on detecting fraudulent credit card transactions using machine learning. The dataset used is from Kaggle, containing anonymized transaction records with a highly imbalanced class distribution. After preprocessing and exploratory data analysis, various classification models were trained and evaluated, including Logistic Regression, Random Forest, and XGBoost. The final model was deployed using Streamlit for real-time predictions. The project demonstrates how ML can be applied to identify fraudulent activity and help financial institutions reduce risk.

3. System Requirements

Hardware:

Minimum 4 GB RAM

Dual-core processor

Software:

Python 3.8+

Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn, xgboost

IDE: Jupyter Notebook or Google Colab

4. Objectives

Detect fraudulent transactions accurately.

Handle imbalanced data effectively.

Evaluate model using precision, recall, and ROC-AUC.

Deploy a model for real-time fraud detection

5. Flowchart of Project Workflow

Include a flowchart like this:

Data Collection → Preprocessing → EDA → Feature Engineering → Modeling → Evaluation → Deployment

You can create this in draw.io or Canva and insert the image into the template.

6. Dataset Description

Source: Kaggle Credit Card Fraud Dataset

Type: Public, anonymized data

Size: 284,807 rows, 31 columns

Insert df.head() screensho

t here

7. Data Preprocessing

No missing values

Scaled features using StandardScaler

Handled class imbalance using SMOTE

Insert before/after screenshots

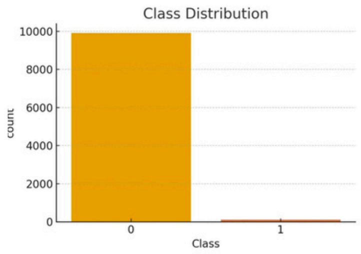
8. Exploratory Data Analysis (EDA)

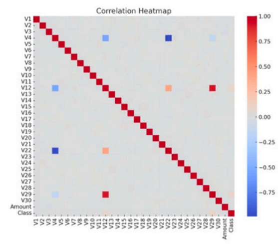
Class distribution shows heavy imbalance

Correlation heatmap shows most features are not correlated

Fraud cases often have low transaction amounts

Include plots like histograms, boxplots, heatmap





9. Feature Engineering

No new features created due to anonymized data

Selected most informative features based on importance from models

Applied PCA (if applicable)

10. Model Building

Tried: Logistic Regression, Random Forest, XGBoost

Chose XGBoost for its handling of imbalanced data

Insert screenshots of training output

11. Model Evaluation

Metrics: Precision, Recall, F1-score, AUC

XGBoost gave the best ROC-AUC

Include confusion matrix and ROC curve

12. Deployment

Platform: Streamlit Cloud

Link: [Insert your Streamlit app link]

Insert screenshot of UI and prediction output

13. Source code

Provide GitHub link to:

Preprocessing script

Model training and evaluation

Deployment app

Source code:

```
#Credit Card fraud detection
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve

from imblearn.over_sampling import SMOTE
from xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegression
```

```
# Load dataset
df = pd.read_csv("creditcard.csv")

# Check for missing values
print(df.isnull().sum())

# Feature scaling
scaler = StandardScaler()
df['Amount'] = scaler.fit_transform(df['Amount'].values.reshape(-1, 1))
df = df.drop(['Time'], axis=1)

# Split data
X = df.drop('Class', axis=1)
y = df['Class']

# Handle imbalance using SMOTE
sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X, y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2, random_state=42)

# Logistic Regression
log_model = LogisticRegression()
log_model.fit(X_train, y_train)
log_preds = log_model.predict(X_test)

print("Logistic Regression Report:")
print(classification_report(y_test, log_preds))

# XGBoost Classifier
```

```
xgb_model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
xgb_model.fit(X_train, y_train)
xgb_preds = xgb_model.predict(X_test)

print("XGBoost Classifier Report:")
print(classification_report(y_test, xgb_preds))

# ROC Curve
fpr, tpr, _ = roc_curve(y_test, xgb_model.predict_proba(X_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label="XGBoost (AUC = {:.2f})".format(roc_auc_score(y_test, xgb_preds)))
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```

14. Future scope

Incorporate real-time data ingestion

Apply deep learning models (e.g., Autoencoders)

Integrate feedback loop to improve model with new fraud patterns

15. Team Members and Roles

Jayapratha.A: Data Cleaning, EDA

Kamali.V: Feature Engineering, SMOTE, Model Training

Jayabharathi.N: Documentation, Visualizations

Jesima.J: Model Evaluation, Reporting