Credit Card Fraud Detection Project

1. Project Overview

The main objective of this project is to detect fraudulent credit card transactions using machine learning algorithms.
The focus is on building a model capable of identifying fraudulent activities while maintaining an **optimal trade-off between precision and recall**, minimizing both false positives and false negatives.

---

2. Dataset

- **Source:** Credit Card Fraud Detection Dataset (anonymized transactions)
- **Total Samples:** 284,807
- **Fraud Cases:** 492 (~0.17%)
- **Features:** 30 numeric features (V1–V28, Amount, Time)
- **Target Column:** *Class → 0 = Non-Fraud, 1 = Fraud*

The dataset was divided as follows:

Dataset Type	Number of Samples
Training Set	170,884
Validation Set	56,960
Test Set	56,960

After Applying SMOTE

Class	Before Balancing	After Balancing
0 (Non-Fraud)	170,579	170,579
1 (Fraud)	305	170,579

---

3. Data Preprocessing

The following preprocessing steps were performed before training:

1. **Data Cleaning:** Removed duplicates and handled missing values.
2. **Feature Scaling:** Scaled numerical columns using StandardScaler.
3. **Data Balancing:** Applied **SMOTE (Synthetic Minority Oversampling Technique)** to handle class imbalance also I used other approaches to balancing this data like (Oversampling , Undersampling)
4. **Data Splitting:** Divided the dataset into training, validation, and test subsets.

---

4. Model Training

- **Algorithm:** XGBoost Classifier
- **Reason for Selection:** XGBoost provides outstanding performance with imbalanced data, effectively handles non-linear relationships, and includes regularization to prevent overfitting.
- **Balanced Training:** Trained on SMOTE-balanced data.

- **Other Approaches Tested:** Oversampling and undersampling, but SMOTE produced the best stability and accuracy.

Training Configuration

Parameter	Value
max_depth	8
n_estimators	400
learning_rate	0.05
scale_pos_weight	2

---

5. Model Performance

Training Results

Metric	Value
F1-Score	**0.9999**
Precision	**0.9999**
Recall	**1.0000**
PR-AUC	**0.9999**
Best Threshold (Train)	**0.978**

Validation Results

Metric	Value
F1-Score	**0.8690**
Precision	**0.9359**
Recall	**0.8111**
PR-AUC	**0.8537**
Best Threshold (Validation)	**0.950**

Test Results (Using Validation Threshold = 0.950)

Metric	Value
F1-Score	**0.8718**
Precision	**0.8673**
Recall	**0.8763**
PR-AUC	**0.8701**

---

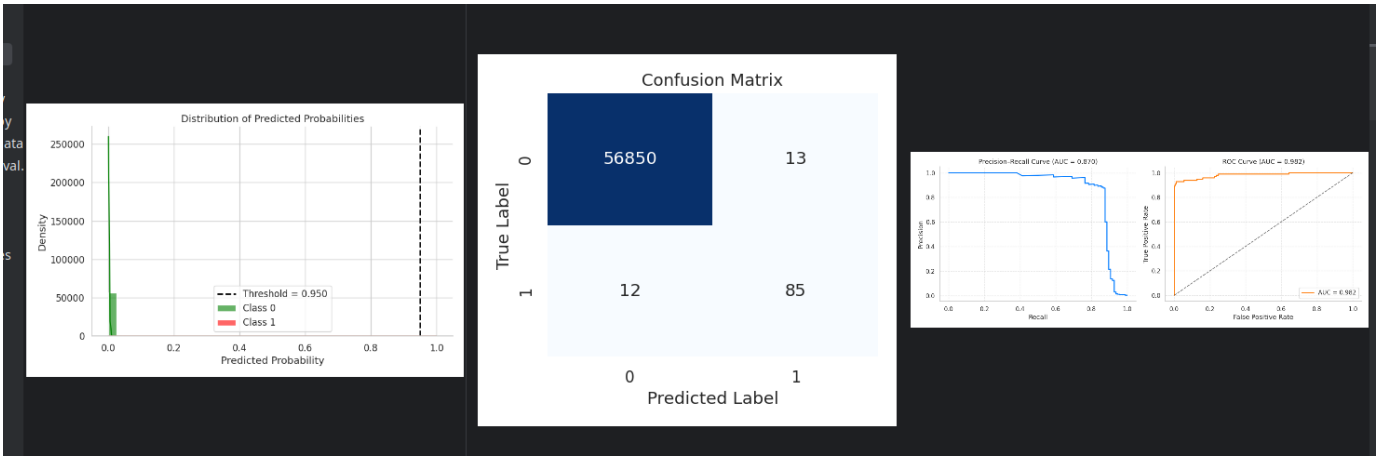
6. Visual Analysis

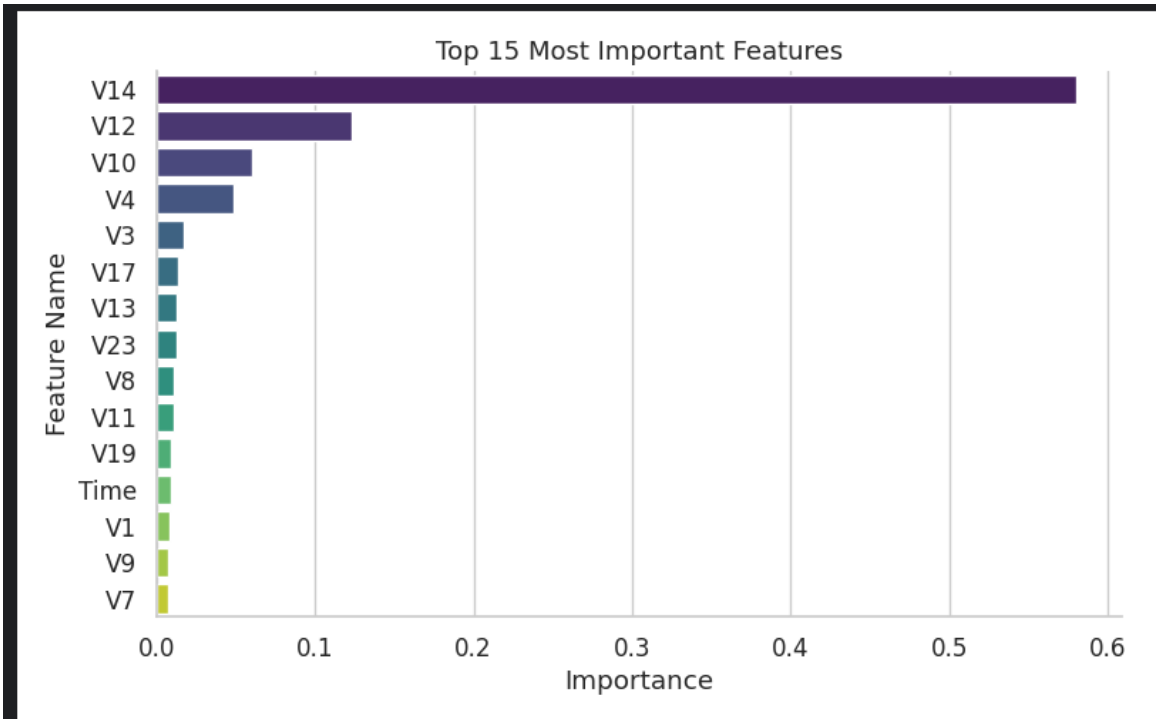
To better understand model performance and behavior, several visualizations were created:

1. **Precision–Recall Curve:** Demonstrates the balance between recall and precision.

2. **ROC Curve:** Plots the trade-off between true positive and false positive rates.
3. **Confusion Matrix:** Visual summary of prediction accuracy.
4. **Predicted Probability Distribution:** Shows probability separation between fraud and non-fraud cases.
5. **Feature Importance Plot:** Highlights which features had the most influence on predictions.

Top important features: V14, V12, and V10 — key indicators in identifying fraudulent activity.





---

7. Conclusion

The XGBoost model achieved **high precision (0.867)** and **strong recall (0.87)** on the test data, making it highly effective for fraud detection.
And my important matrices in this Project was **F1-Score (0.8718)**
Its robustness and ability to handle class imbalance made it the best-performing algorithm among those tested.

---

8. Technologies Used

Category	Tools / Libraries
Programming Language	Python 3.12
Libraries	XGBoost, Scikit-learn, Pandas, Matplotlib, Seaborn

Category	Tools / Libraries
Environment	PyCharm + Virtual Environment (.venv)
Hardware	CPU (CUDA GPU optional)

---

9. Author

Author: Ibrahem Ali