Capstone 2 Project Report

Credit Card Fraud Detection

Introduction

This project focuses on detecting fraudulent credit card transactions using machine learning. Fraud detection is critical for reducing financial losses, protecting customers, and improving trust in financial systems. The objective of this project is to build and compare several models that can accurately identify fraudulent transactions, despite the extremely imbalanced nature of the dataset. The final goal is to select the most effective model and provide recommendations on how the results can support real-world fraud monitoring.

Data Wrangling

The dataset originally contained anonymized transaction features along with a binary variable indicating whether each transaction was fraudulent. Initial steps included loading the data, reviewing its structure, checking for missing values, correcting data types, and removing unnecessary columns.

No missing values were found, and all features were numeric except for the "Class" label. The "Time" variable was noted as a special case because it represents seconds from the start of data collection and does not behave like a typical continuous predictor. It was therefore handled carefully in later steps. After verification, the cleaned dataset was saved for exploration.

Data Dictionary

Column Name	Type	Description	Notes / Range
Time	Numeric	The number of seconds between this transaction and the first transaction in the dataset.	Ranges from 0 to about 172,000 seconds (around 2 days).
V1 – V28	Numeric (anonymized)	Features created using **Principal Component Analysis (PCA)** to hide sensitive details. Each represents patterns or combinations of original transaction data such as user behavior, location, or card usage.	Can be positive or negative values. Exact meanings are unknown.
Amount	Numeric (currency units)	The transaction amount in the original currency (e.g., Euros).	Ranges from very small to very large amounts. Often right-skewed (many small, few large).
Class	Categorical (0 or 1)	Target variable: **0 = normal transaction**, **1 = fraudulent transaction.**	Highly imbalanced — frauds make up less than 1% of the data.

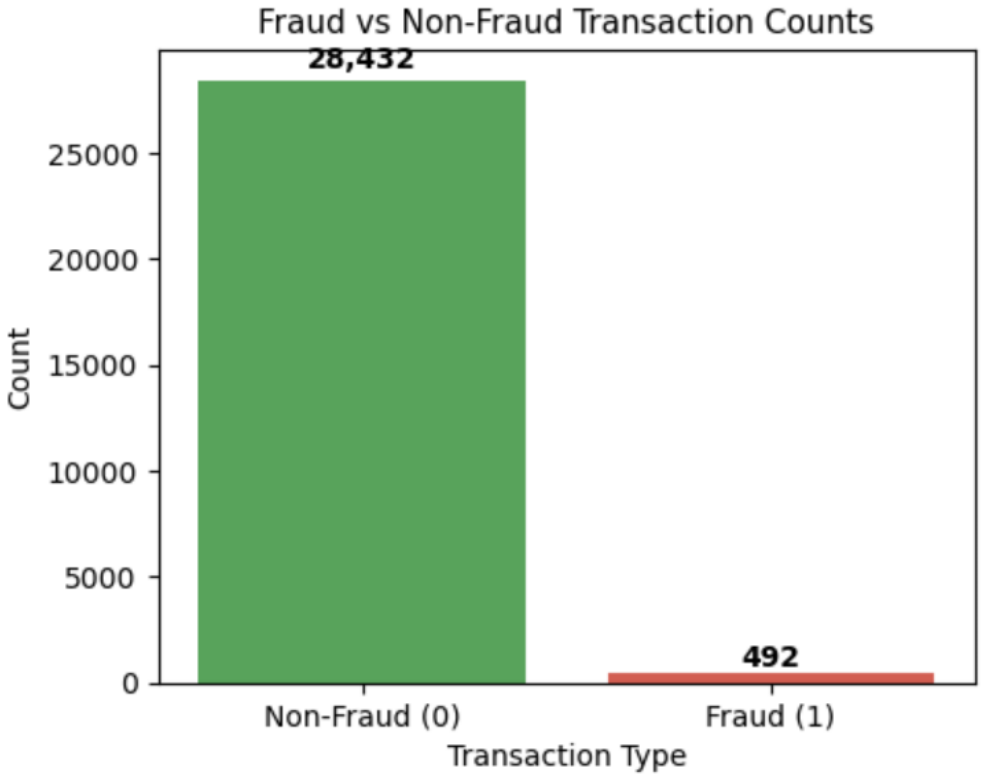
Figure 1. The dataset has 30 columns total — one for time, 28 PCA features, one for transaction amount, and one for class.

Exploratory Data Analysis

EDA focused on understanding the distribution of transactions and identifying patterns related to fraud. Visualizations showed that fraud cases make up less than 2% of all transactions, highlighting a significant class imbalance.

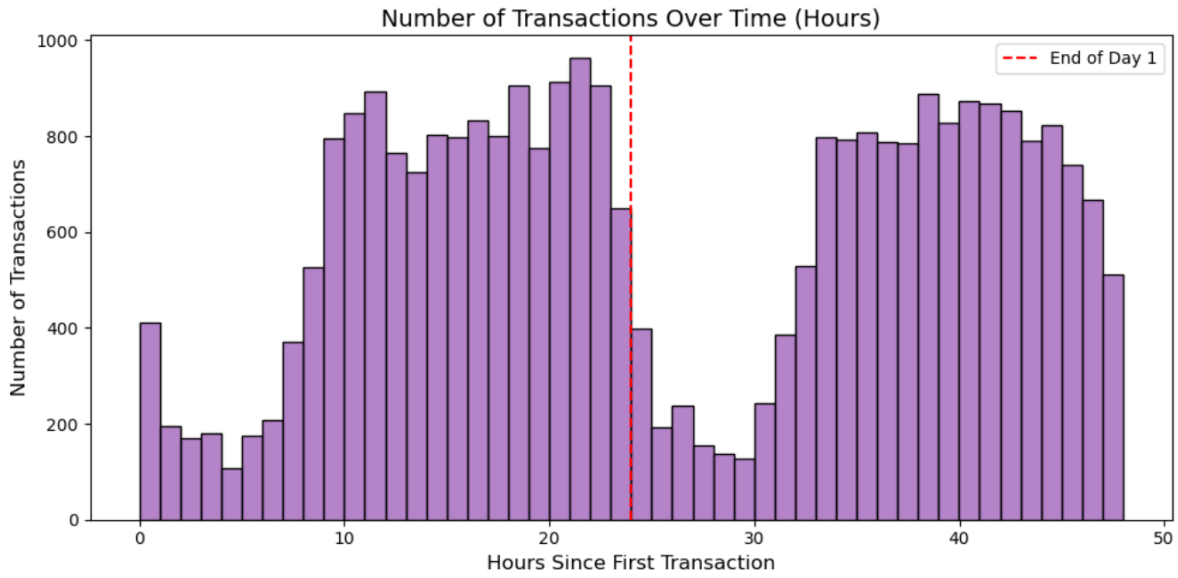
Transaction amounts were highly skewed, with most fraud occurring in lower to mid-range values. Correlation analysis showed limited linear relationships because features are PCA-transformed, but certain components still appeared more influential for fraud.

These findings guided later preprocessing steps and model selection. Figures from the EDA notebook help illustrate the imbalance and the patterns observed.

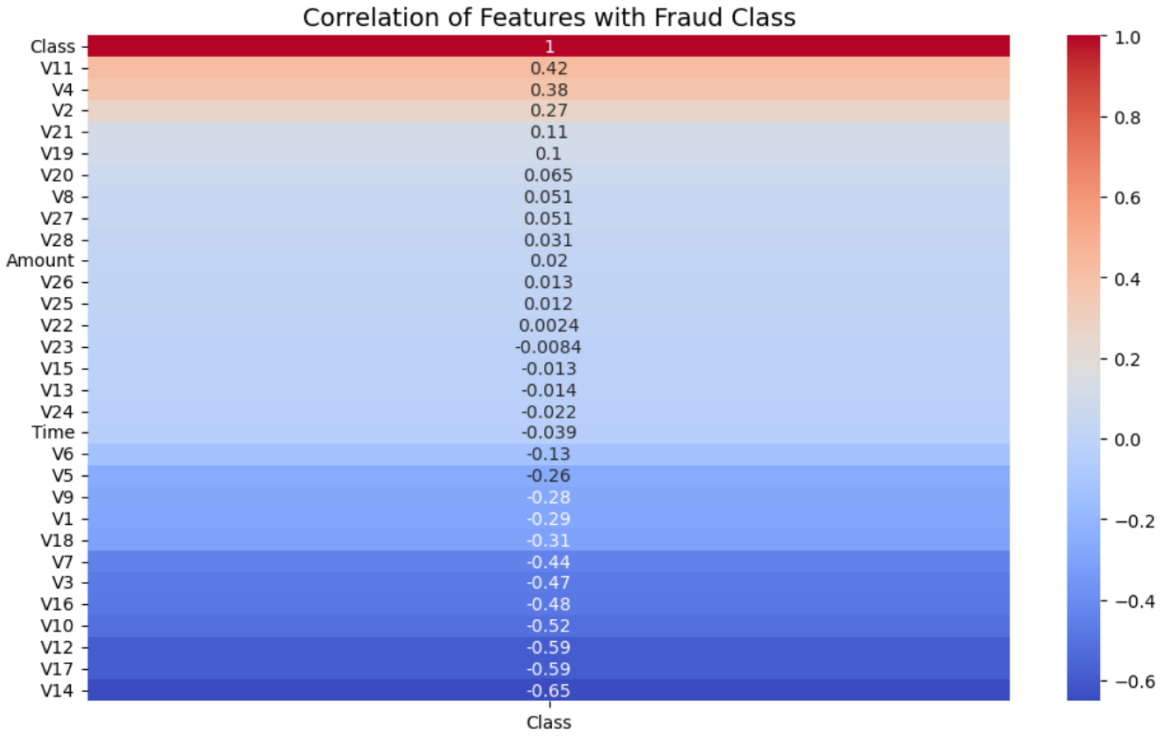



```
Percentage of fraudulent transactions: 1.7010%
```

Figure 2. Only about 1.7% of the transactions are fraud — almost all the others are normal. Because fraud is so rare, the model needs to be careful to catch those few fraud cases without wrongly flagging too many normal ones.



Fugure 3. Transactions are unevenly distributed across the two-day period. Peaks and dips suggest daily activity patterns (e.g., daytime vs. nighttime).




```
Top 10 features correlated with Class:
 Class    1.000000
V11       0.423192
V4        0.379510
V2        0.270344
V21       0.109878
V19       0.104997
V20       0.065156
V8        0.051324
V27       0.050756
V28       0.031038
Name: Class, dtype: float64
```

Figure 4. The features most strongly associated with fraud are V11, V4, and V2.

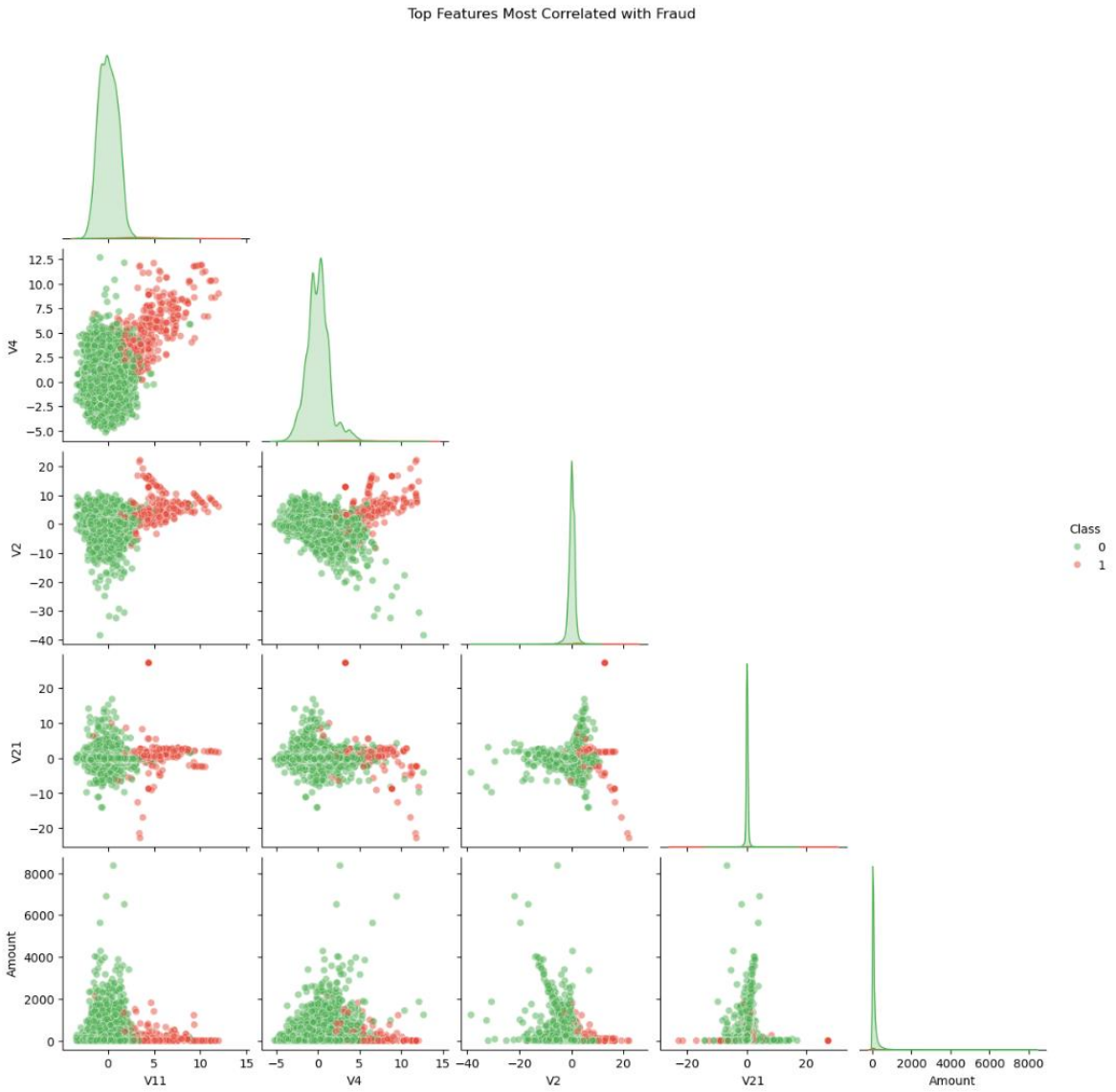


Figure 5. Fraudulent transactions tend to show up in different areas of the data compared to normal transactions, especially when looking at features like V11, V4, V2, and V21.

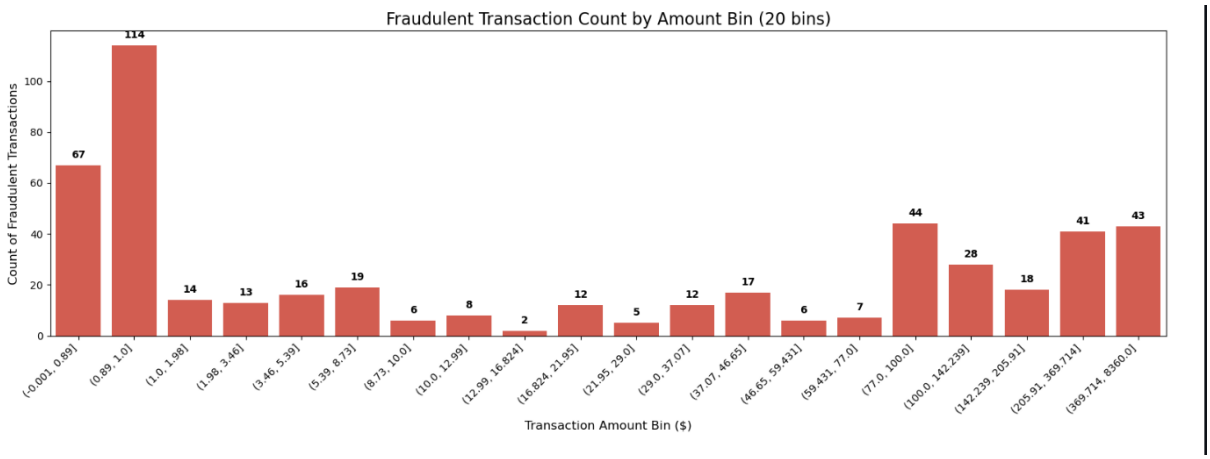


Figure 6. Most fraud happens with small to medium transactions, and very little occurs in very large transactions. Fraud can happen at any amount, so all transaction sizes should be considered. Grouping transaction amounts into categories could help the model spot fraud more easily.

Preprocessing

Preprocessing was designed to prepare the data for modeling while avoiding data leakage.

Train/Test Split

The dataset was split into training and testing sets first to ensure that transformations were applied only to training data.

Feature Scaling

A StandardScaler was applied to the numeric features after splitting. Scaling ensures that all variables contribute equally to the model and prevents models from being influenced by the magnitude of certain features.

Handling Class Imbalance

Because fraudulent transactions are extremely rare, SMOTE (Synthetic Minority Oversampling Technique) was used to balance the classes in the training set. SMOTE creates synthetic fraud samples to help the model learn patterns more effectively.

Final Preprocessed Dataset

After scaling and SMOTE, the training data was balanced, standardized, and ready for modeling. Visual checks confirmed the new class distribution.

```
Before SMOTE: Counter({0: 22745, 1: 394})
After SMOTE: Counter({0: 22745, 1: 22745})
```

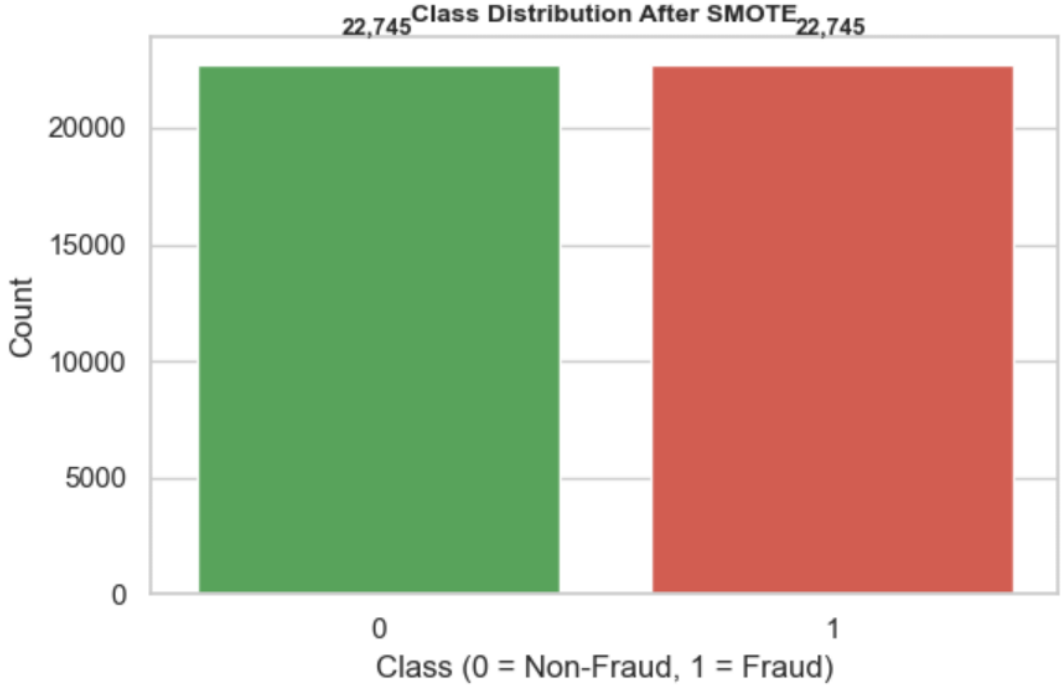


Figure 7. After applying SMOTE, we can now see it's clearly balanced.

Modeling

Three machine learning models were built and compared:

Logistic Regression

Served as the baseline model. It performed reasonably well but struggled to capture non-linear patterns, resulting in lower recall than more flexible models.

Random Forest Classifier

Performed better than Logistic Regression. It captured more complex patterns and delivered stronger recall and balanced metrics.

XGBoost Classifier

This model achieved the best overall performance. It produced the highest precision, recall, F1-score, and AUC, making it the most reliable at detecting fraud while minimizing false negatives. ROC curves and precision-recall curves confirmed its superiority across all evaluation areas.

Based on PR-AUC and Recall, the **Random Forest** model performs the best at detecting fraudulent transactions.

- Logistic Regression is simpler and faster but has lower Recall and PR-AUC.
- Random Forest captures more fraud cases with higher Recall and PR-AUC, making it suitable for imbalanced data.
- Precision is slightly lower for Random Forest, meaning some false positives occur, but this trade-off is acceptable in fraud detection where catching fraud is critical.

Next steps include using the chosen model for further fine-tuning, threshold adjustment, and possibly deployment for real-time fraud detection.

Recommendations

Based on the modeling results, the following actions are recommended:

1. **Adopt the XGBoost model** for fraud detection, as it consistently produced the strongest results.
2. **Adjust the decision threshold** depending on business needs—lowering the threshold increases fraud detection but may produce more alerts.
3. **Use insights from feature importance** to enhance rule-based monitoring and assist fraud analysts in reviewing suspicious activity.

Implementing these recommendations can significantly improve fraud detection accuracy and reduce financial risk.

Conclusion

This project successfully applied the full data science workflow to build a fraud detection model, moving from data wrangling to EDA, preprocessing, and modeling. After comparing multiple models, XGBoost emerged as the best-performing approach due to its strong recall,

precision, and overall predictive capability. The findings support practical recommendations that can help financial institutions strengthen fraud prevention and improve customer protection.