SE 3007: Introduction to Machine Learning

A Case Study in Credit Card Fraud Detection

Our Investigation: From Raw Data to Actionable Insight

1. **The Problem:** Defining the high-stakes challenge of credit card fraud.

2. **Exploratory Data Analysis (EDA):** Uncovering the story hidden within the data.

3. **Data Preparation & Modeling:** Forging our tools and selecting a champion model.

4. **In-Depth Model Evaluation:** Scrutinizing performance with the right metrics.

5. **Model Interpretability:** Opening the black box to understand *why* a prediction is made.

6. **Conclusion & Future Work:** Summarizing findings and outlining next steps.

The Challenge: Detecting a Silent, Costly Threat

The Business Problem:

Credit card fraud is a significant financial threat, requiring detection models that are not only accurate but also operate in real-time without disrupting legitimate customer transactions.

The Machine Learning Goal:

To build a high-performance classification model that accurately distinguishes fraudulent transactions (`Class = 1`) from legitimate ones (`Class = 0`).

High Class Imbalance:

Fraudulent transactions are exceptionally rare.

Performance:

Detections must be near-instantaneous.

Cost of Errors:

False negatives result in direct financial loss, while false positives lead to poor customer experience.

The Raw Material: A Snapshot of European Transactions

Dataset Overview: The dataset comprises credit card transactions over a two-day period from September 2013 by European cardholders.

Features (30 total):

- `Time`, `Amount`, and the target variable `Class`.
- `V1` through `V28`: These are principal components derived from a PCA transformation to anonymize sensitive cardholder data. This means they are already scaled and uncorrelated.

Dataset Size:

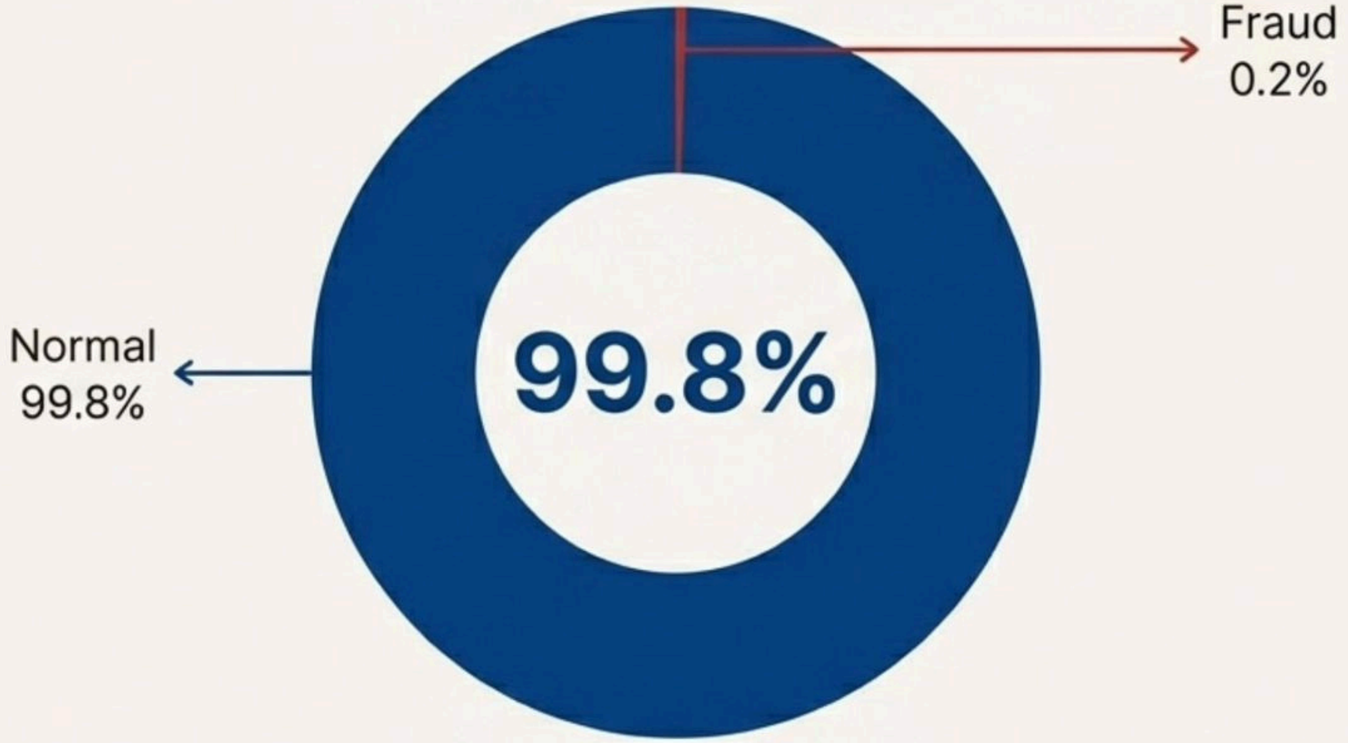
- 284,807 transactions.

Target Variable (`Class`):

- `1` for fraudulent transactions.
- `0` for normal transactions.

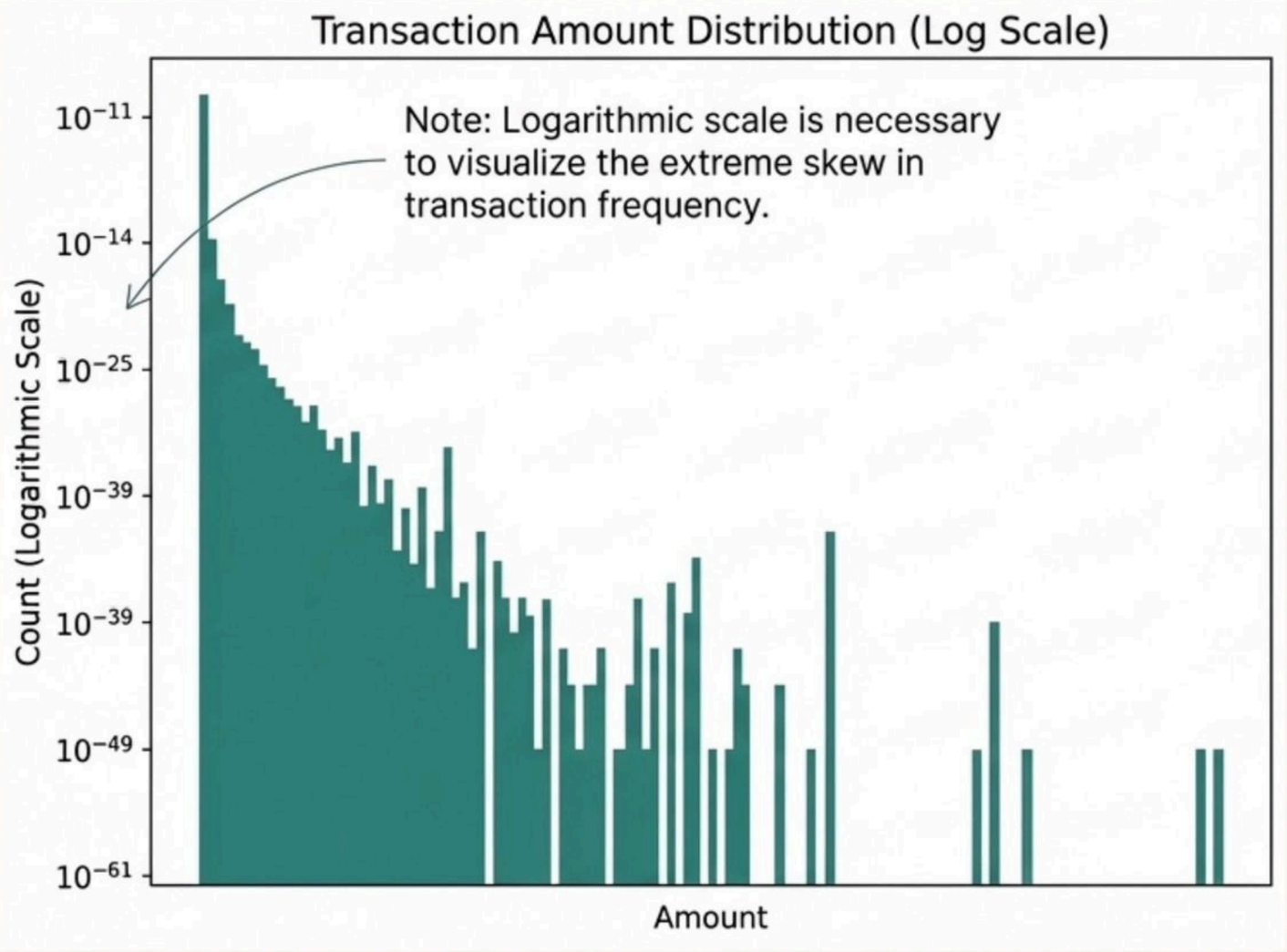
EDA: The Needle in the Haystack

Class Distribution: The first and most important observation is the severe class imbalance. Fraudulent transactions represent a tiny fraction of the data.



Critical Implication: A naive model predicting 'Normal' every time would achieve 99.8% accuracy. This makes **accuracy a fundamentally misleading metric** for this project. We must focus on metrics like Precision, Recall, and F1-Score.

EDA: Analyzing Transaction Values



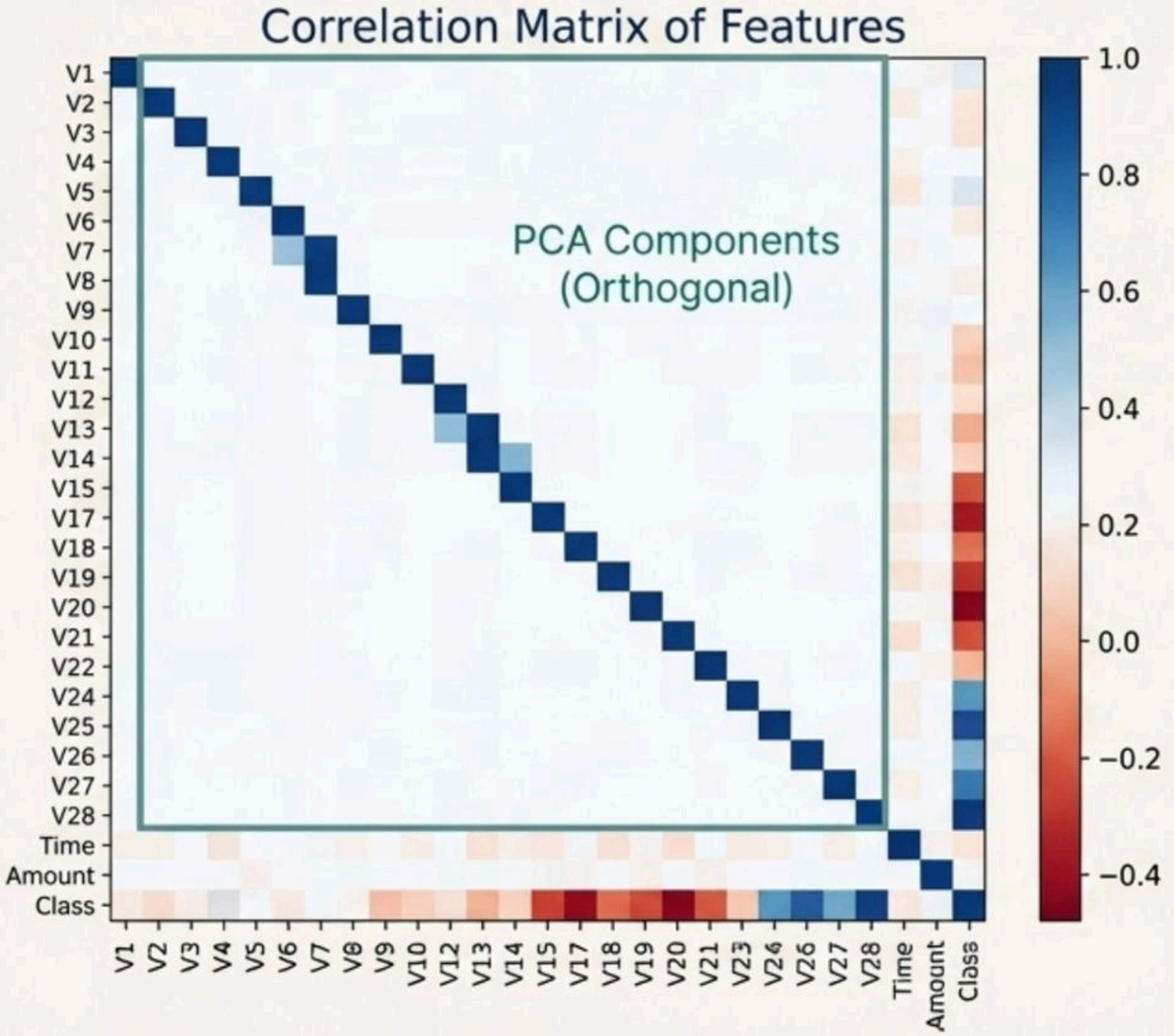
**Observations:

- The vast majority of transactions are for small amounts, with the distribution being highly skewed to the right.
- The y-axis uses a logarithmic scale to visualize the distribution, as the frequency of low-value transactions is orders of magnitude higher than high-value ones.

**Actionable Step:

The `Amount` and `Time` features have vastly different scales from the V-features. They will require scaling (e.g., using `StandardScaler`) during preprocessing to prevent them from disproportionately influencing the model.

EDA: Uncovering Feature Relationships



- **Analysis:**
  - **V1-V28 Features:** These features show virtually no correlation with one another. This is an expected outcome of the PCA transformation, which generates orthogonal (uncorrelated) components.
  - **Correlations with Target:** Several features show a notable correlation with the `Class` variable. For example, `V17`, `V14`, `V12`, and `V10` show a negative correlation, while `V11`, `V4`, and `V2` show a positive correlation. These are likely to be strong predictors of fraud.

Preparing the Data for Modeling

A disciplined preprocessing pipeline is essential for reliable results.

1. Feature Scaling

We applied `StandardScaler` to the `Amount` and `Time` columns. This normalizes their distributions, giving them equal weight relative to the V-features.

2. Train-Test Split

The data was split into an 80% training set and a 20% testing set. The model is evaluated on the completely unseen test set.

3. Handling Imbalance (Training Data Only)

We applied **SMOTE** (Synthetic Minority Over-sampling Technique) exclusively to the training data.

SMOTE synthesizes new minority class (fraud) samples, balancing the dataset for the model to learn from without contaminating the test set.

A Contest of Algorithms: Selecting the Best Model

We evaluated four robust classification algorithms to identify the top performer for this task.

Evaluation Metric:

Our primary metric is the **F1-Score**. It calculates the harmonic mean of Precision and Recall, providing a single, reliable measure for imbalanced classification performance.

Logistic Regression

A linear, highly interpretable baseline model.

Gradient Boosting

An ensemble method building decision trees sequentially.

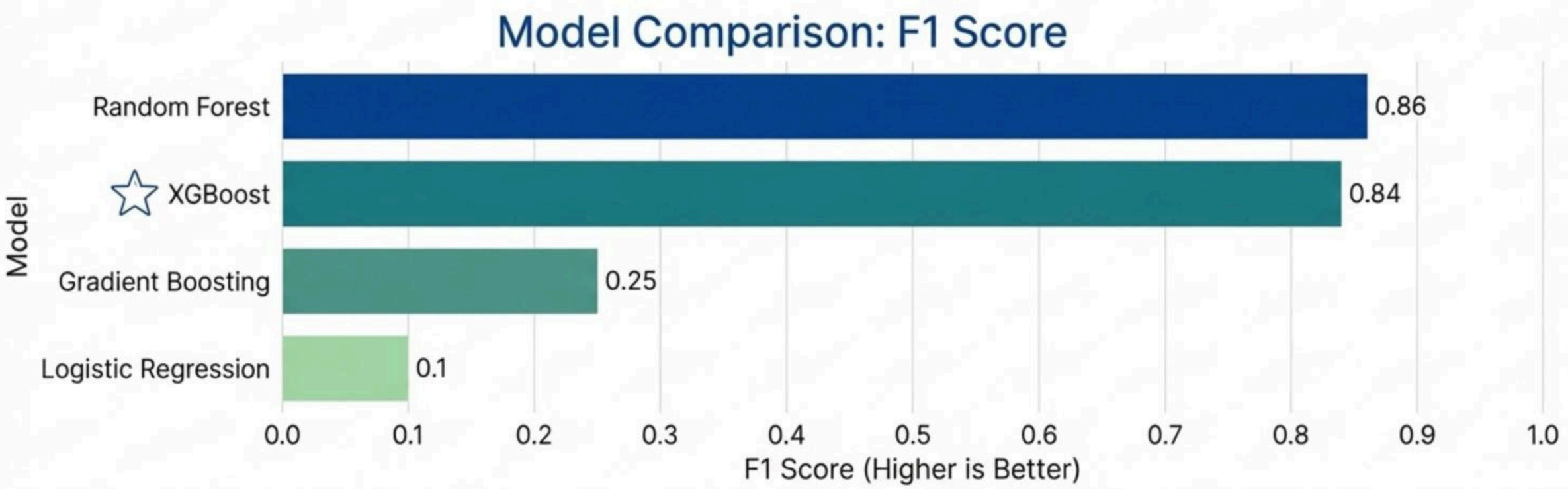
XGBoost

A highly optimized, high-performance implementation of gradient boosting.

Random Forest

An ensemble method building decision trees in parallel.

The Results: Random Forest Sets the Bar



Performance Summary:

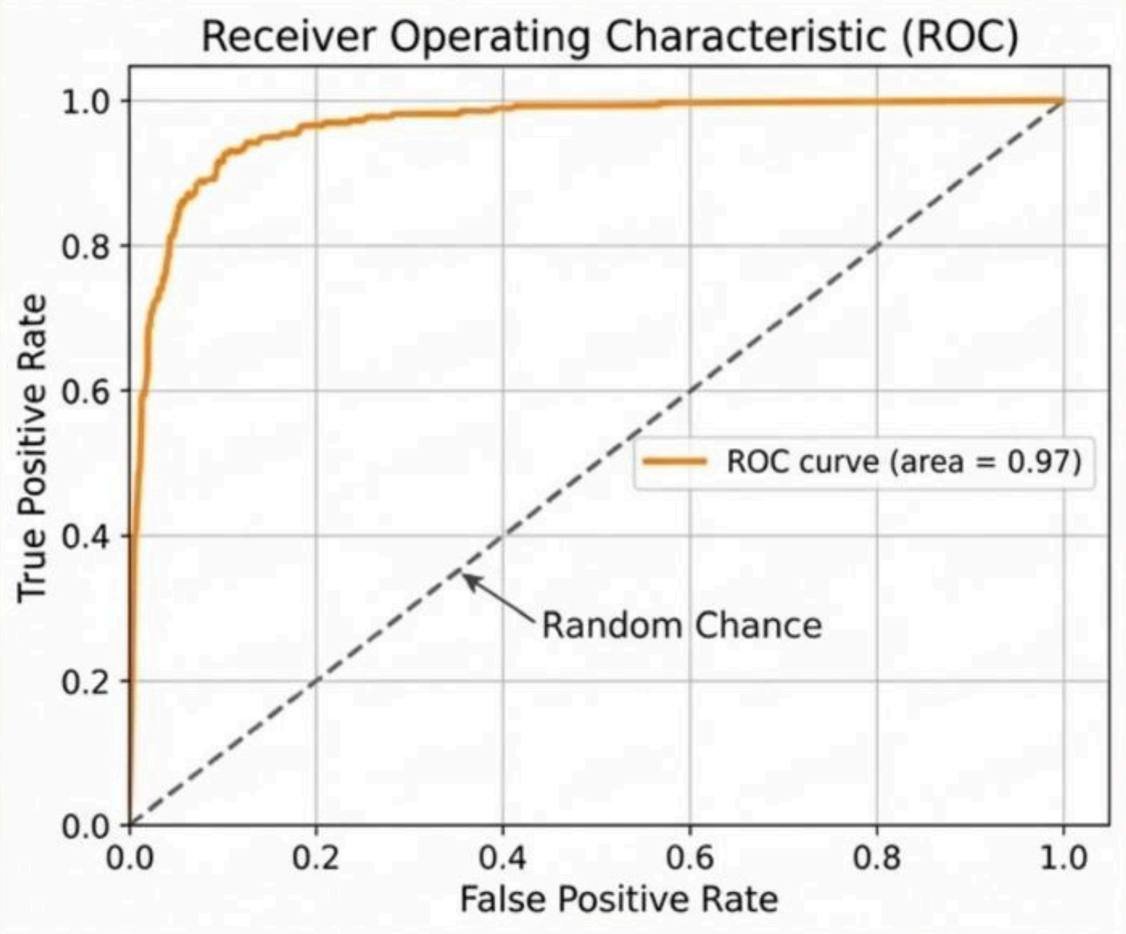
- **Random Forest** achieved the highest F1 Score, indicating the best balance between precision and recall on the unseen test data.
- **XGBoost** performed nearly as well, proving to be a powerful alternative.
- Both tree-based ensemble methods significantly outperformed the baseline Logistic Regression.

Our Choice for Deep Dive:

We will proceed with **XGBoost** for our in-depth analysis. Its performance is comparable to Random Forest, and it offers superior tools for model interpretability, such as SHAP.

Model Evaluation: The ROC Curve

The Receiver Operating Characteristic (ROC) curve illustrates the model's ability to separate fraudulent and normal transactions across all classification thresholds.

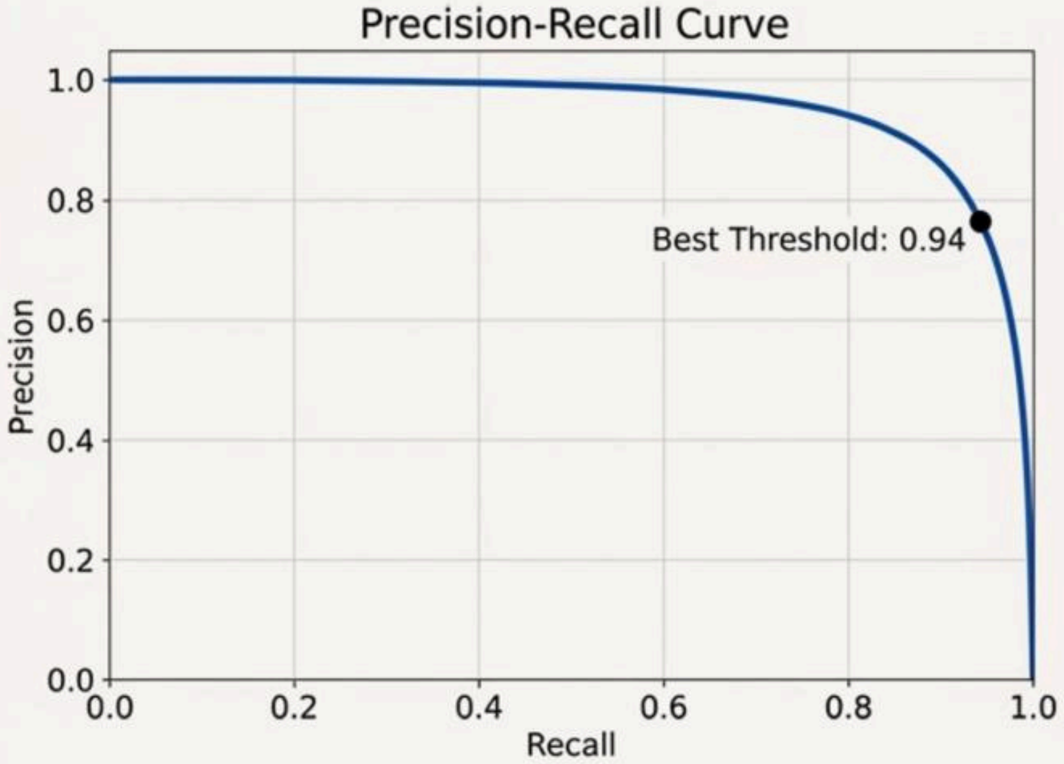


Interpretation:

- **Area Under the Curve (AUC) = 0.97.** A score this close to 1.0 indicates an excellent model with a high degree of separability.
- The curve’s position far from the diagonal dashed line (representing random chance) confirms the model’s strong predictive power.

Model Evaluation: The Precision-Recall Trade-off

For highly imbalanced datasets, the Precision-Recall (PR) curve provides more practical insight than the ROC curve.



Precision:
Of all transactions we flag as fraud, what percentage are *actually* fraud? (Measures the cost of False Positives).

Recall:
Of all actual fraudulent transactions, what percentage did we *catch*? (Measures the cost of False Negatives).

The curve shows that to increase Recall (catch more fraud), we must sometimes accept a lower Precision. The optimal threshold found (0.94) provides an excellent balance.

The Moment of Truth: Confusion Matrix Analysis (at Threshold=0.94)

The confusion matrix provides a detailed look at the model's performance on the 56,746 test transactions.

Breakdown of Predictions:

- **True Negatives (TN): 56,648** – Correctly identified normal transactions.
- **True Positives (TP): 74** – Correctly identified fraudulent transactions.
- **False Positives (FP): 3** - **Type I Error**: Legitimate transactions incorrectly flagged as fraud. *Business Impact: Potential customer friction.*
- **False Negatives (FN): 21** - **Type II Error**: Fraudulent transactions that were missed. *Business Impact: Direct financial loss.*

Final Scoresheet: Performance by the Numbers

The classification report summarizes the precision, recall, and F1-score for our chosen optimal threshold (0.94).

Classification Report (XGBoost on Test Set):

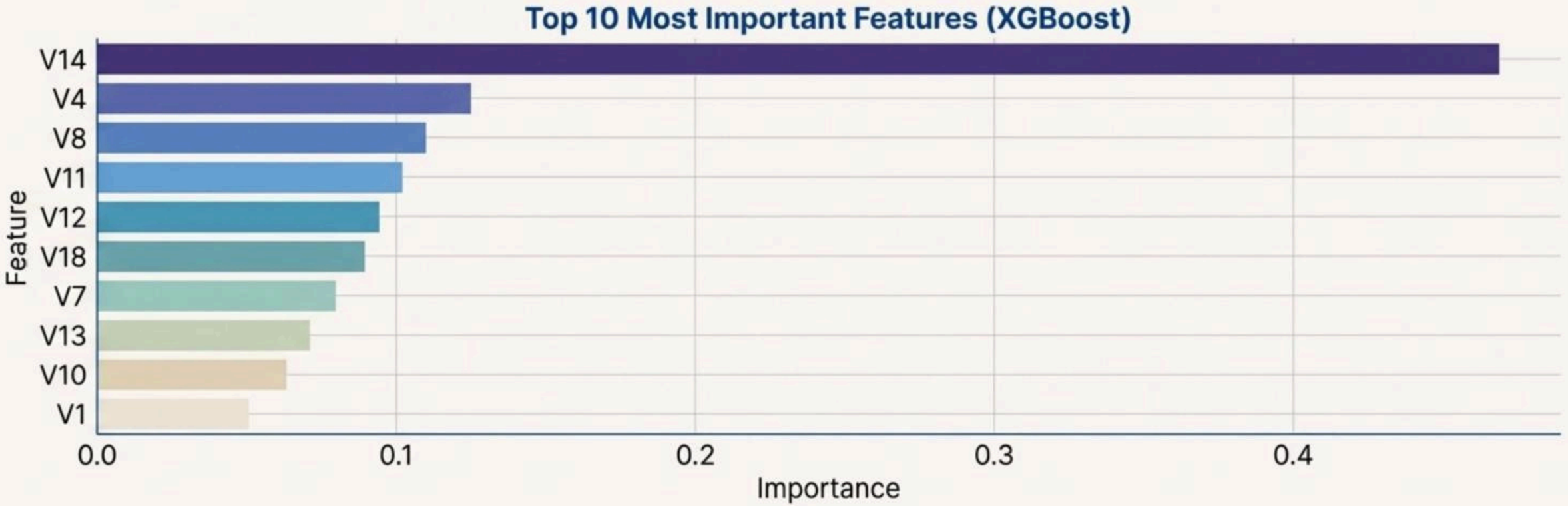
	precision	recall	f1-score	support
Normal (0)	1.00	1.00	1.00	56651
Fraud (1)	0.96	0.78	0.86	95
accuracy			1.00	56746
macro avg	0.98	0.89	0.93	56746
weighted avg	1.00	1.00	1.00	56746

Key Results for the Fraud Class:

- **Recall:** **0.78** — We successfully identified 78% of all actual fraud in the test set.
- **Precision:** **0.96** — When our model predicted fraud, it was correct 96% of the time.
- **F1-Score:** **0.86** — A strong overall score reflecting a healthy balance between precision and recall.

Inside the Model: Which Features Drive Predictions?

XGBoost can rank features based on their overall contribution to the model's performance.



Top 5 Most Important Features:

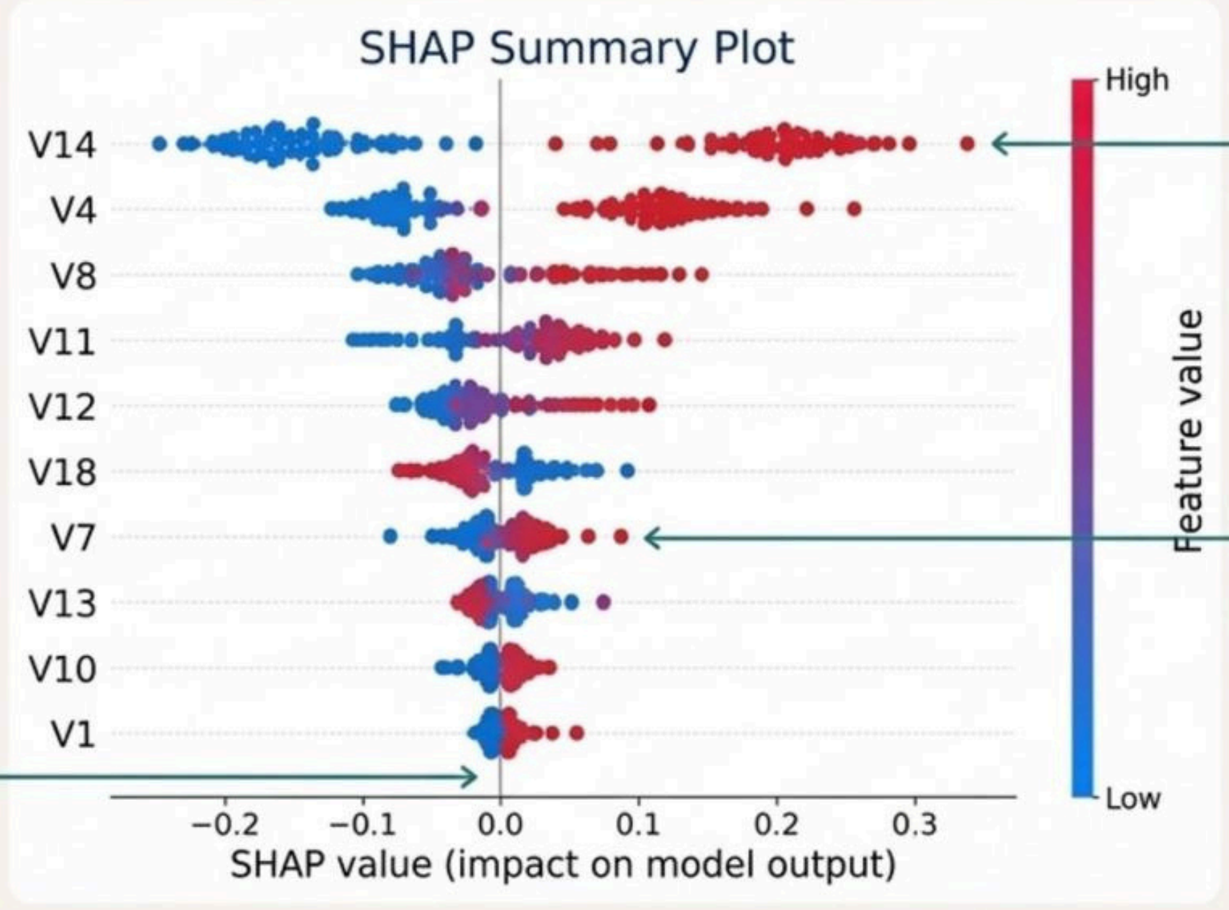
1. V14
2. V4
3. V8
4. V11
5. V12

Insight:

A small subset of the anonymized features are disproportionately important for identifying fraud. Feature **V14** is by far the most significant predictor.

Advanced Interpretability: Opening the Black Box with SHAP

To understand *why* individual predictions are made, we use SHAP (SHapley Additive exPlanations). It shows the impact of each feature on every single prediction.



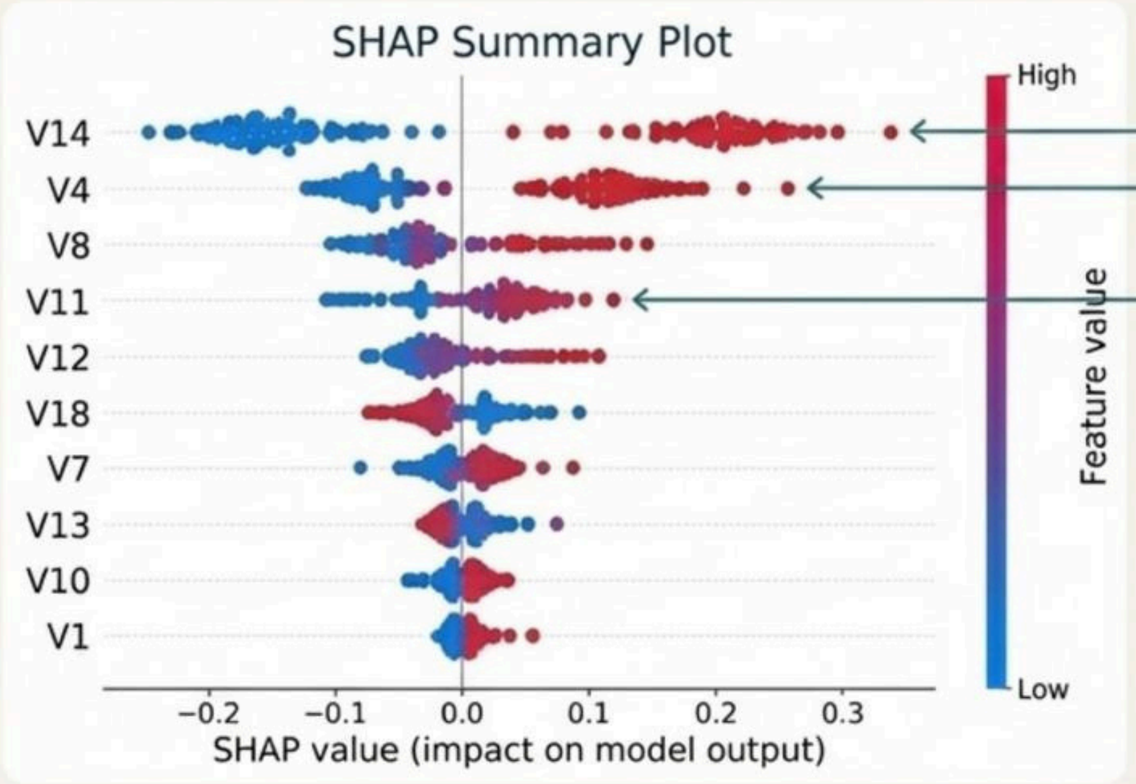
Features: Each row is a feature, ordered by importance.

Feature Value: Red indicates a high value for that feature, while blue indicates a low value.

Impact on Output: Positive SHAP values push the prediction towards 'Fraud' (1). Negative values push it towards 'Normal' (0).

Actionable Insights from SHAP Analysis

The SHAP plot reveals the specific data patterns that the model associates with fraud.



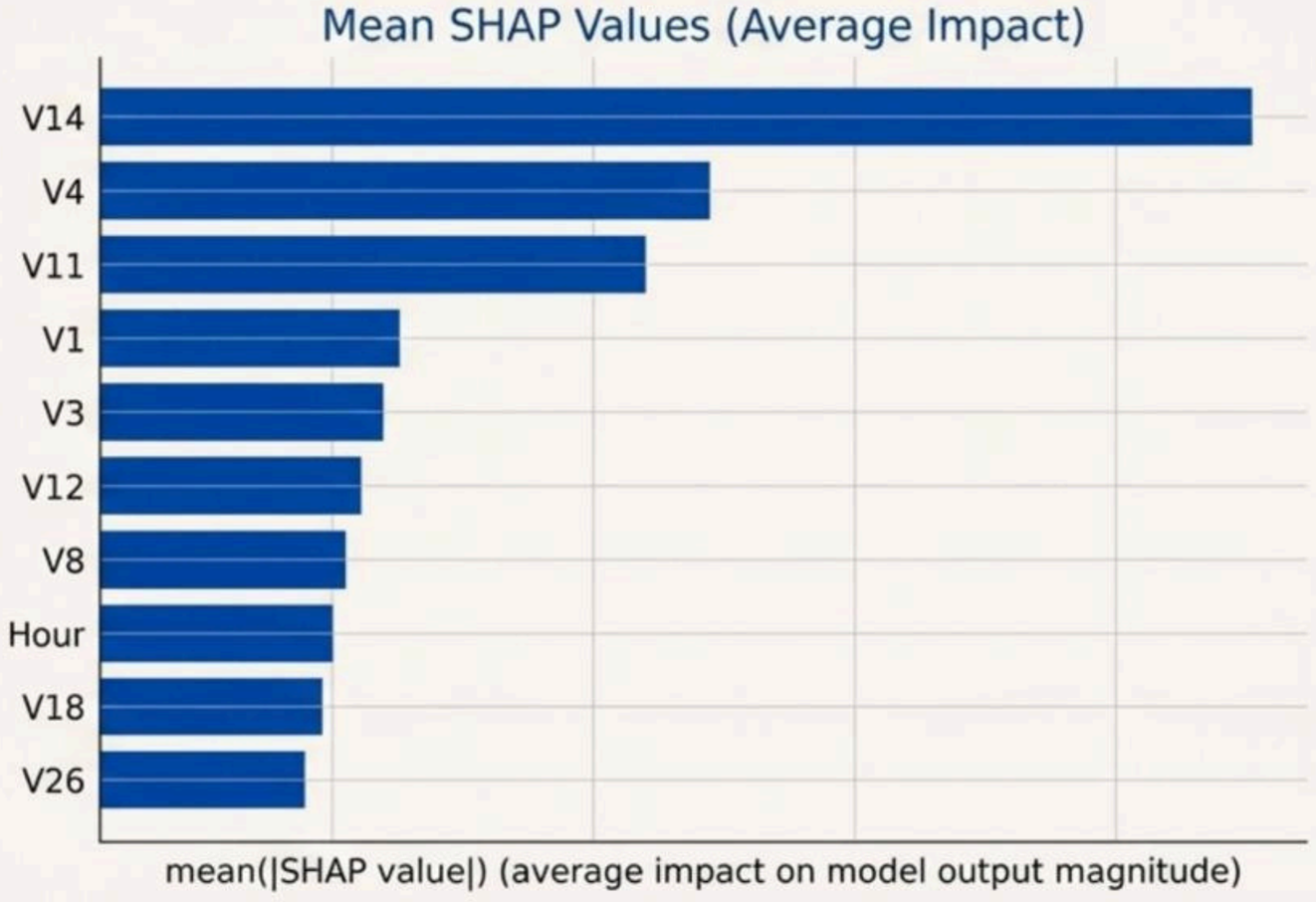
Key Fraud Indicators Learned by the Model:

- **V14:** Low values of V14 (blue dots on the left) have large positive SHAP values, strongly pushing the prediction towards **Fraud.**
- **V4:** High values of V4 (red dots) consistently have positive SHAP values, indicating a higher likelihood of **Fraud**.
- **V12:** Low values of V12 (blue dots) have a strong positive impact, also indicating **Fraud**.
- **V11:** High values of V11 (red dots) are another strong indicator of **Fraud**.

Even without knowing the original meaning of these features, we can identify the exact decision rules the model has learned.

A Different Perspective: Mean Feature Impact (SHAP)

This plot simplifies the SHAP analysis by showing the average absolute impact each feature has on the model's output magnitude. It confirms the feature importance hierarchy we saw earlier.



Interpretation:

- This view reinforces that `V14`, `V4`, and `V11` have the largest average influence on the model's predictions, either positive or negative.
- This aligns with the standard feature importance plot but is grounded in Shapley values, providing a more robust measure of importance.

Project Conclusion and Key Findings

Problem: We successfully tackled the challenge of credit card fraud detection within a highly imbalanced dataset where standard accuracy is an ineffective metric.

Solution: An XGBoost model was trained and tuned, achieving an F1-Score of **0.86** for the fraud class and an AUC of **0.97**, demonstrating excellent predictive power.

Methodology: The key to success was a rigorous process: detailed EDA, correct handling of class imbalance with SMOTE on the training set, and evaluation using appropriate metrics like the PR curve.

Insights: Advanced interpretability with SHAP moved beyond prediction to explanation, revealing the specific feature behaviors (e.g., low V14, high V4) that are strong indicators of fraud.

Future Work and Project Dependencies

A model is never truly “finished.” The next steps would focus on optimization and production readiness.

Next Steps:

1. **Advanced Hyperparameter Tuning:** Systematically optimize the XGBoost model using a framework like Optuna or GridSearch to potentially improve the F1-Score further.

2. **Deployment Strategy:** Package the preprocessing pipeline and trained model into a production-ready API for real-time inference.

3. **Performance Monitoring:** Implement a system to monitor the model for data drift and performance degradation over time, with a strategy for periodic retraining.

Project Dependencies:

```
pandas
numpy
scikit-learn
imbalanced-learn
xgboost
matplotlib
seaborn
joblib
streamlit
shap
```