Credit Card Fraud Detection

MSA 8150 – Machine Learning

Team – Akshara Ravishankar, Dhanyatha Gandamalla, Nivethitha Pandi

May 1, 2025

Georgia State University | J. Mack Robinson College of Business

Robinson

DETECTING CREDIT CARD FRAUD USING MACHINE LEARNING

62 million Americans had fraudulent charges on their credit or debit cards in 2024, with unauthorized purchases exceeding $6.2 billion annually

PROBLEM STATEMENT

- Credit card fraud causes significant financial loss and **undermines trust in digital payments**.
- Financial losses for individuals and businesses, **damage to credit scores**, and **increased costs for card issuers**

IMPORTANCE

- Early fraud detection helps reduce financial losses, improves user trust in digital transactions, and enhances the overall security of financial systems.
- Automating fraud detection with machine learning enables real-time monitoring, minimizes manual effort, and scales efficiently with growing transaction volumes.

PROJECT SCOPE

The project focuses on supervised learning, specifically binary classification, using engineered features and ensemble models to efficiently detect credit fraud.

Exploratory Data Analysis

DATASET DESCRIPTION

OVERVIEW

- **Total Transactions:** 284,807
- **Total Features:** 31
- 30 predictors (V1 to V28, Amount, Time)
- 1 target (Class: 0 = Legitimate, 1 = Fraudulent)

CLASS DISTRIBUTION

- **Class Imbalance:**
- Legitimate (Class = 0): 99.83%
- Fraudulent (Class = 1): 0.17%

Highly imbalanced — fraud cases are rare and hard to detect.

FEATURE DESCRIPTION

- V1 to V28: PCA-transformed for privacy
- Amount: Raw transaction value
- Time: Seconds since first recorded transaction

DATA QUALITY

Missing Values: None

Outliers: Retained, as outliers could indicate fraud behaviour

Ready for modeling: Yes, no cleaning needed

PCA – Transformed features:

✓ The dataset contains only numerical input variables which are the result of a PCA transformation due to confidentiality issues. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'

Hidden Patterns in Transaction Timing & Spending

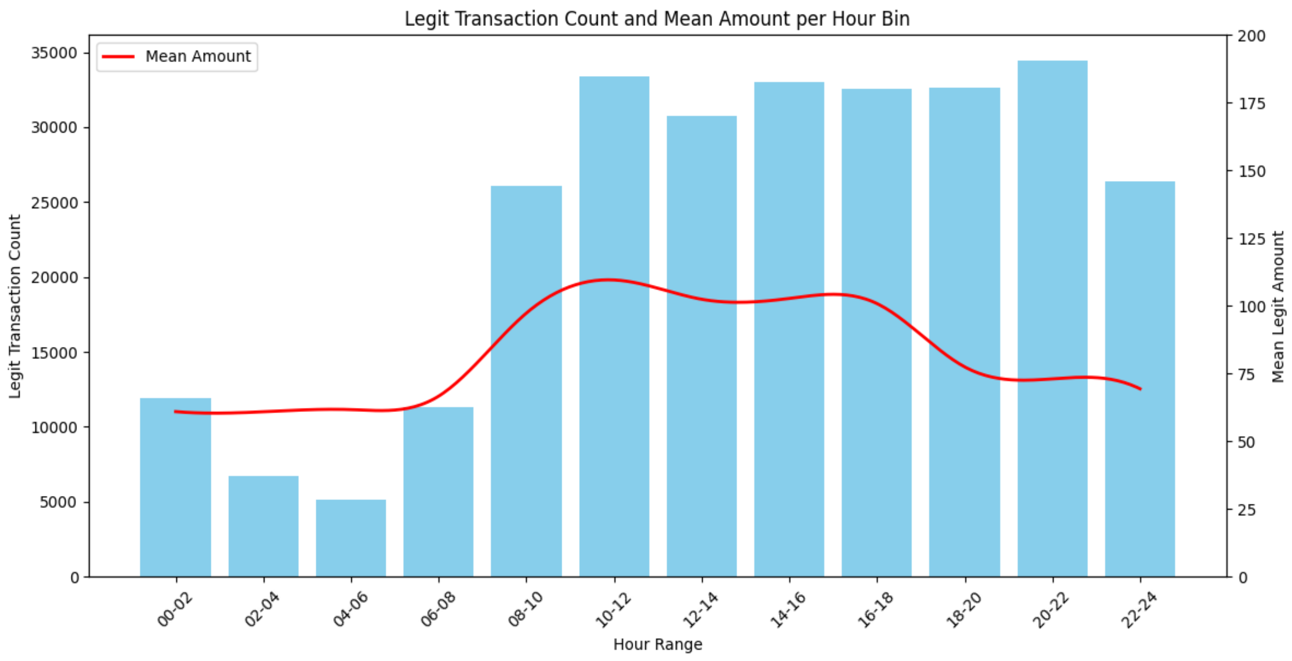
Legit (Non-Fraud) Transactions

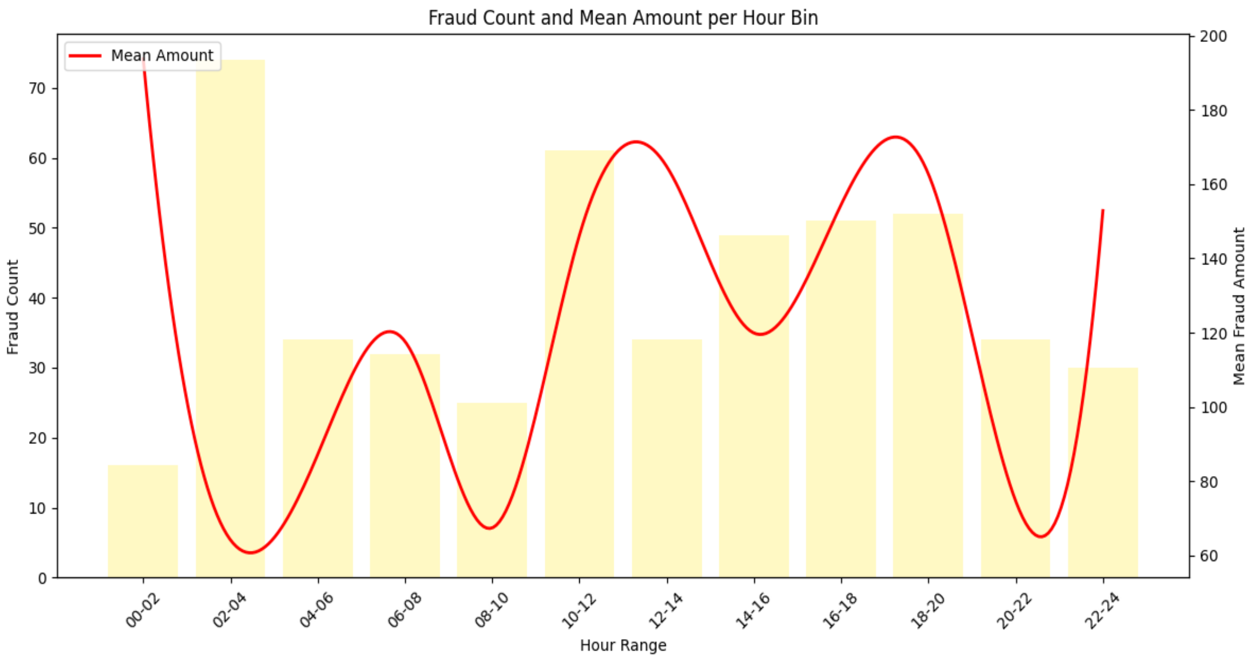
- Low activity during early hours (00:00–06:00), reflecting normal human/business activity cycles.
- $ Average amount is steady, ranging between $50 - $100

vs

Fraudulent Transactions

- Noticeable spikes during off-peak hours, such as 02:00–04:00, suggesting fraudsters may exploit quiet periods.
- $ Although average amount is generally low (<$70), fraud amounts fluctuate and are possibly intentionally varied to evade detection.





Feature Selection

Feature Engineering and Scaling Strategies

Changing Time → Hour

- Values under the given ‘Time’ column represented # of seconds in relative to the first transaction
- Converted continuous ‘Time’ into a cyclical behavioural signal
- Extracted ‘Hour’ from raw Time feature using:

Hour = (Time // 3600) % 24

Normalization (Min-max Scaling)

- Applied to the PCA transformed features (V1-V28)
- Ensures features are in a uniform [0, 1] range

RobustScaler

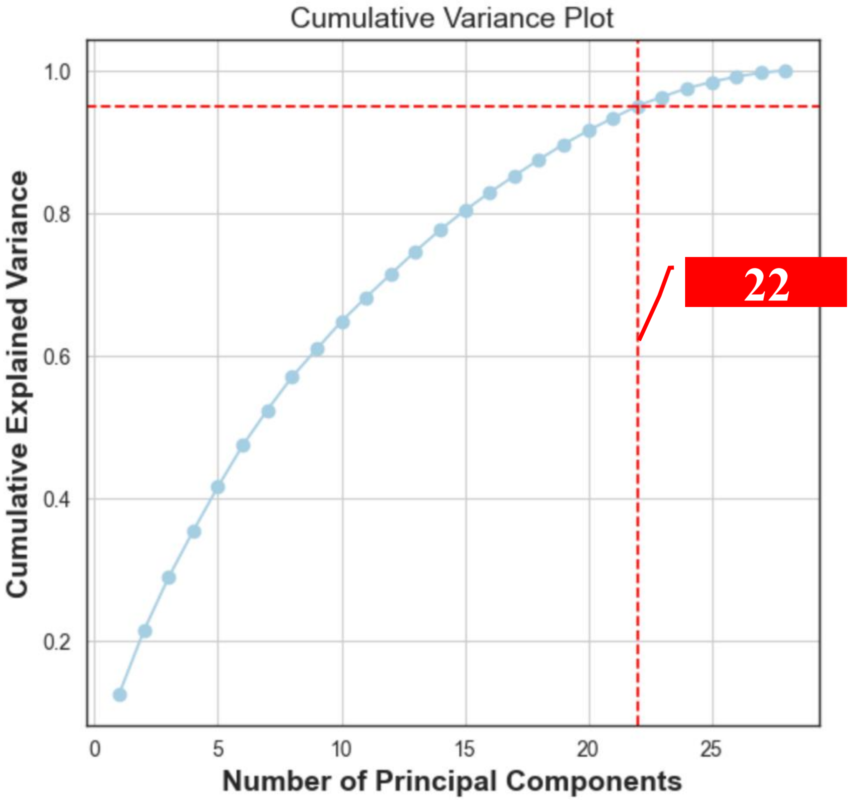
- Applied to the ‘Amount’ and ‘Hour’
- Preserves meaningful differences while avoiding distortion from extreme values
- Resistant to outliers using median and IQR
- Specifically important for our use case, as fraud instances could have extreme values, which will not be lost while scaling

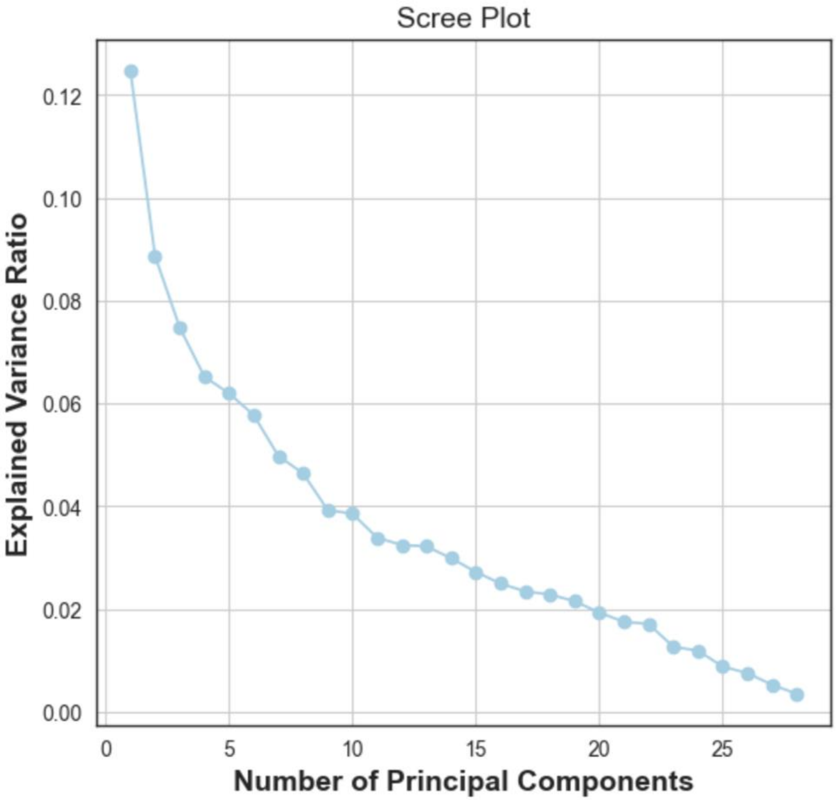
Feature Selection Using Explained Variance

- ✓ **Explained variance** refers to **how much "information" (variance)** each principal component carries
- ✓ We're working with **PCA-transformed features**, so explained variance is a **direct measure of information retained**
- ✓ Cumulative Explained Variance plot helps to select smallest number of components to capture most variance (95%)

Optimal number of components: 22

Total Variance of PCA transformed Features: 30.73





Modeling Experiments

Experiment 1 : Leveraging UnderSampling for Baseline Models

- Created a 1:1 balanced subset (**492 fraud, 492 legit**) by undersampling the majority class
- Evaluated different models using **Cross-validation** and F1-score

Model	Mean F1 Score
Logistic Regression	0.908
SVM	0.885
Random Forest	0.939
Gradient Boosting	0.943

- **Gradient Boosting** classifier trained and evaluated
- Outperforms simpler models in many tabular problems, robust to feature noise

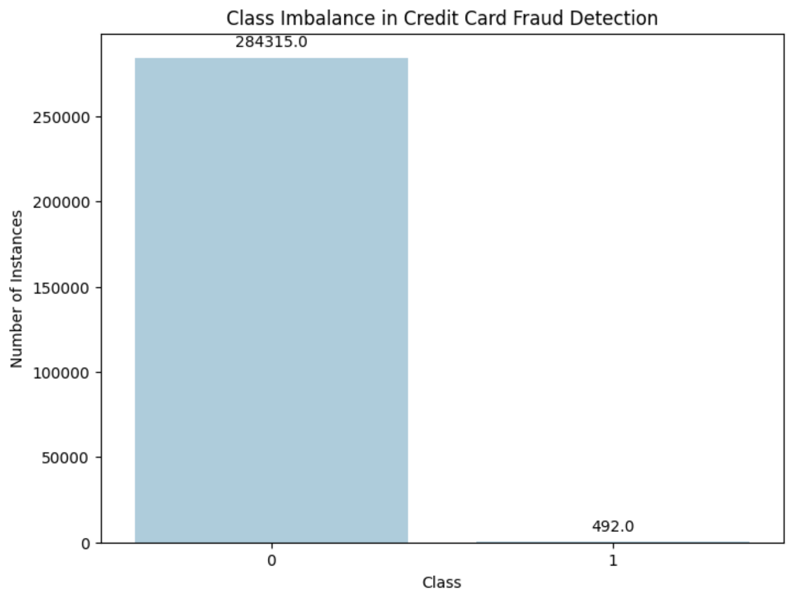
Evaluating Gradient Boost Outcomes

Classification Report:

	precision	recall	f1-score	support
0	0.91	0.98	0.94	87
1	0.98	0.93	0.95	110
accuracy			0.95	197
macro avg	0.95	0.95	0.95	197
weighted avg	0.95	0.95	0.95	197

- Precision-recall trade-off is very well balanced — rare for fraud tasks!
- Although, Good performance but model may not generalize well due to significant data loss

Resampling Alternative: Handling Class Imbalance Using SMOTE



Our dataset is highly imbalanced:

- There is **only 0.17% of fraud samples** in the dataset (284k non-fraud vs 492 fraud)
- Standard machine learning models tend to ignore the minority class, leading to poor generalization
- **Oversampling preferred** over downsampling in fraud detection as it retains the full information from the majority class

Why SMOTE for Oversampling?

✓ **Creates Synthetic Samples, Not Just Duplicate** - This avoids overfitting that happens with random oversampling

✓ **Preserves Decision Boundaries** - leads to more generalizable models

✓ **Fully Integrates with Pipelines and Cross-Validation** - works seamlessly with scikit-learn pipelines, allowing you to apply it only on training folds, preventing data leakage

Experiment 2 : SMOTE + Random Forest

Why Random Forest?

✓ Random Forest is an **ensemble of decision trees**, (using bagging) which naturally captures:

- Nonlinear relationships (like Fraudulent behavior)
- Interactions between features

Oversampling Technique	• Upsample fraud class (equally as non-fraud class) using SMOTE
Define Pipeline	• SMOTE for Oversampling minority class • Random Forest Classifier
Cross-Validation	• SMOTE generates synthetic data only in training splits • 5-fold Stratified Cross validation
Evaluation	• Validation performance evaluation by taking mean of metrics across all splits
Train & Test Evaluation	• Model is trained on complete Train dataset • Evaluating performance on test data

Validation Performance:

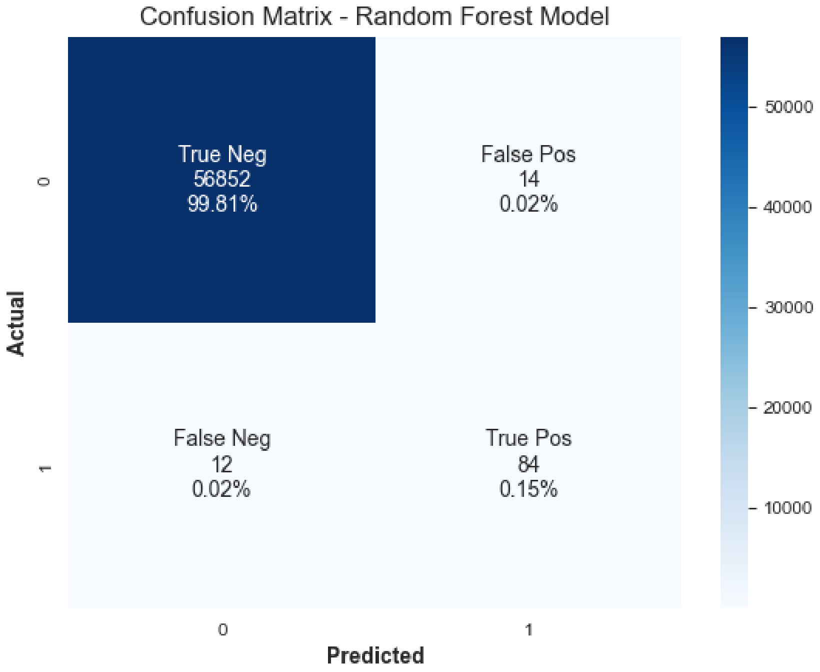
Accuracy:0.99
Precision: 0.88
Recall: 0.81
F1: 0.84
AUC: 0.91

Experiment 2 : SMOTE + Random Forest - Evaluation

High Performing model with Test Accuracy: 99.95%

Fraud Class Performance	
Precision: 0.86	Recall: 0.88
F1-Score: 0.87	Support: 96

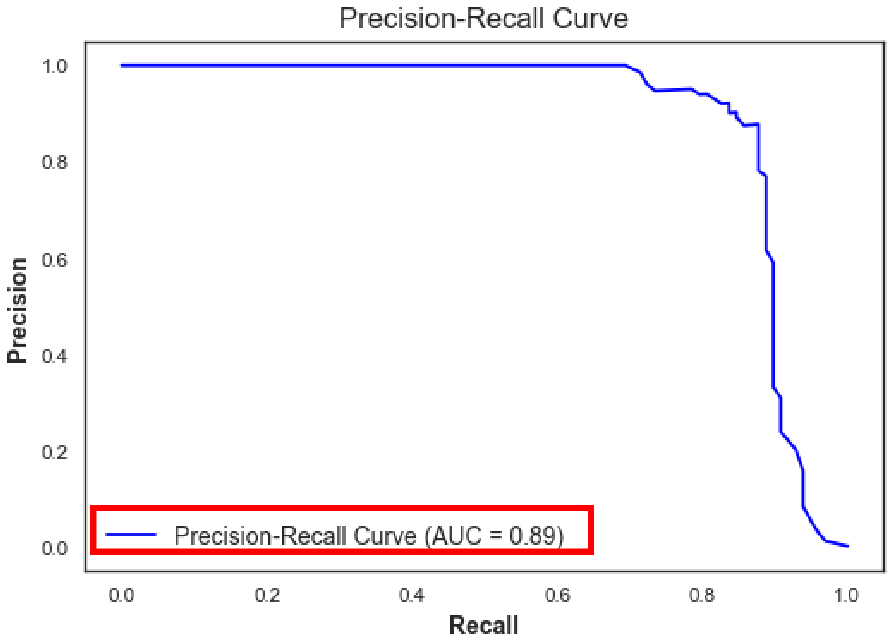
- Fraud Class is clearly distinguished from the majority normal class, with low False Negatives



Assessing Area under Precision-Recall (PR) Curve

Why AUC-PR Matters?

- In imbalanced data, ROC-AUC can be misleading. PR-AUC shows how well the model handles the **minority class**
- AUC = **0.89**, the PR curve shows **high precision is maintained even as recall increases**



Experiment 3 : RFE + SMOTE + XGBOOST

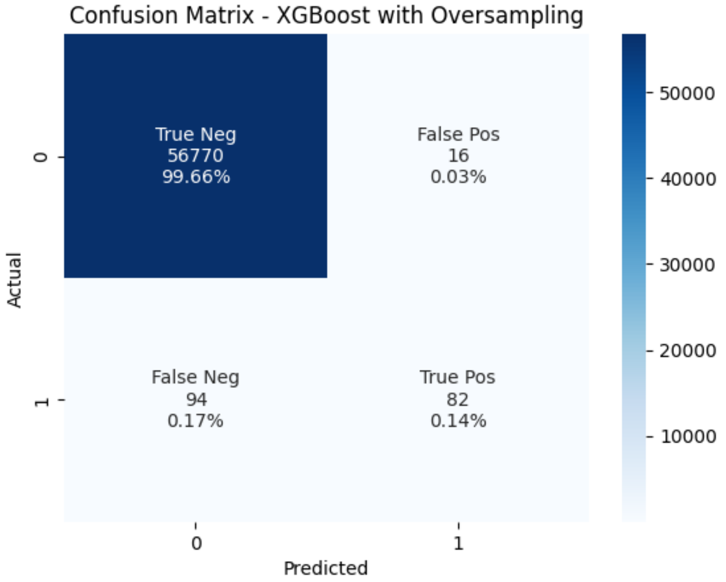
Alternative Feature Selection - RFECV

- RFECV recursively **removes the weakest features** (based on model importance) using Cross Validation
- Selected the optimal number of features based on **recall performance**

Model Pipeline

- Oversampled fraud cases using SMOTE until they reach **30% of the legit class**
- **XGBoost classifier** - powerful gradient boosting algorithm well-suited for tabular, imbalanced data, **Optimized for AUC-PR**

Evaluating XGBoost Outcomes



- Very high accuracy on legit transactions (99.66%)
- Also, high Falge Negatives → 94 frauds missed

Model is **too conservative** — it doesn't catch all frauds

Results

Georgia State University | J. MACK ROBINSON COLLEGE OF BUSINESS

Robinson

Assessing Models Using Recall

Why recall is important?

- The primary goal is to catch as many fraudulent transactions as possible
- Recall is the ratio of correctly identified frauds to the total actual frauds = TP / (TP + FN)
- Missing a fraud (false negative) = lost money, potential data breaches, legal issues

Establishing best performing model

Model	Recall (Fraud)	Precision (Fraud)	Accuracy (Overall)
Gradient Boosting + Undersampling	0.93	0.98	0.95
SMOTE + Random Forest	0.88	0.86	0.99
RFE + SMOTE + XGBoost	0.47	0.84	0.99

- **Smote + Random Forest** is the most balanced model
- Most suitable for real-world fraud detection, where recall is critical (you don't want to miss fraud) and false positives are tolerable

Analyzing Champion Model for Complexity, Interpretability

Complexity

- Random Forest is an **ensemble of decision trees**, which adds some complexity but remains **manageable**
- Compared to Gradient Boosting or deep models, it's **less sensitive to parameter choices**

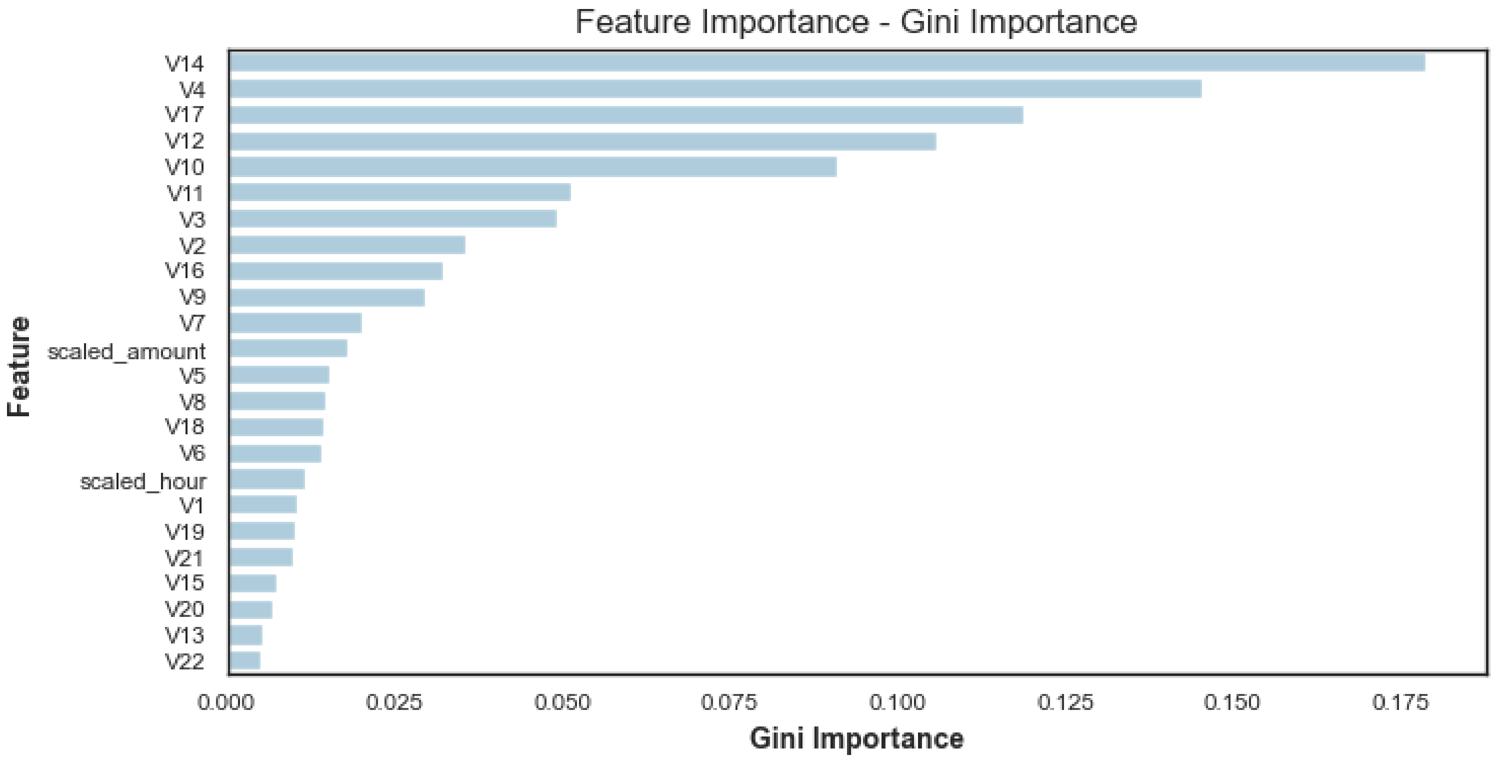
Interpretability

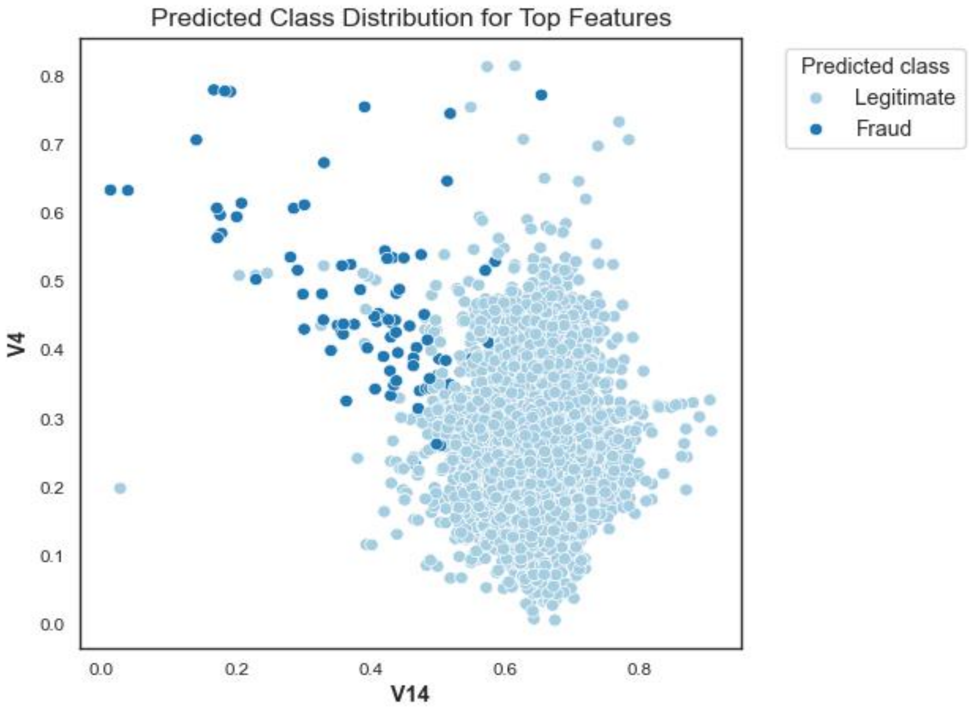
- Offers **feature importance**: Helps explain which inputs most influence predictions
- Easier to explain to stakeholders (vs. XGBoost or black-box neural networks)

Trade-Offs

- Random Forests can be slower to train/infer with many trees (vs. baseline models like logistic regression)
- Interpretation per individual prediction is possible but less straightforward than with simpler models

Visualizing Feature Importance





- Pros of RandomForest: Provides Feature Importance - We can identify the top features contributing to model
- V14 - Most influential in splitting fraud vs non-fraud

- We see that Legitimate samples have a cluster of values for the top features – **V14 and V4**.
- Fraud samples clearly separated, seem more scattered

Conclusion

KEY TAKEAWAYS

Business Value:

- The model successfully detects fraudulent transactions with high precision and recall, reducing potential financial losses and boosting trust in digital payment systems.

Key Insights:

- Feature engineering (like time-based patterns) and handling class imbalance (using SMOTE) were critical to improving detection accuracy.

Operational Benefits:

- Automating fraud detection minimizes manual review workload, accelerates response times, and allows scalable monitoring of large transaction volumes.

Future Improvements:

- Integrate real-time streaming data to enable live fraud detection.
- Explore advanced models (like neural-networks) to push performance further.
- Investigate cost-sensitive learning to balance between false positives and negatives more effectively.
- Perform continuous model retraining as fraud patterns evolve.

Thank You!