

Securing Transactions with ML

Credit Card Fraud Detection Analysis with Logistic Regression & Deep Learning

The Motivation

- **Critical Problem:** Credit card fraud results in significant financial liabilities and decreases consumer confidence, directly impacting a company's reputation.
 - We want to maintain consumer trust and protect company assets.
- **Time Sensitivity:** Fast and effective response times are needed to minimize financial loss.
- ML models provide the ability for quick pattern recognition in big transaction datasets.



Background

- Dataset: Credit card transactions made by European cardholders in September 2013 (over 2 days, 284, 807 transactions)
- Fraud cases: 492 (0.172%), making the dataset highly imbalanced
- Features:
 - V1-V28: Principal components derived from PCA for confidentiality
 - Amount: Transaction value
 - Class: Binary target variable

Data Summary: creditcard.csv

284k

TOTAL TRANSACTIONS

30

INPUT FEATURES (V1-V28)

0.17%

FRAUD RATE (CLASS 1)

Proposed Solution

1. Logistic Regression

- Simplistic linear classification provides a fast and easy foundation for classification by drawing linear boundaries.
- Includes L2 regularization to keep the weights small and prevent overfitting.
- **Expected Result:** Expected to yield high recall (identifying most fraud) but likely low precision (high rate of false alarms) due to model simplicity.

2. Neural Network

- **Architecture:** A Keras Deep Learning model.
 - Features two hidden layers employing the ReLU activation function.
 - Final output layer uses the sigmoid activation function for binary classification.
- **Expected Result:** Leads to fewer false alarms (high precision) while maintaining strong recall, indicating a better balance between catching fraud and minimizing noise.

Implementation Stack

Pre-Processing

- **Scaling:** StandardScaler applied to the 'Amount' feature to normalize its range
- **Cleaning:** The 'Time' column was dropped as it was deemed non-predictive.
- **Splitting:** Used StratifiedShuffleSplit (80/20) to ensure the 0.17% fraud ratio is preserved across training and test sets.

Logistic Regression

- **Solver:** lbfgs was chosen for its memory efficiency.
- **Weights:** class_weight='balanced' was utilized, which automatically adjusts weights inversely proportional to class frequencies to handle imbalance.
- **Iterations:** max_iter=1000 was set to ensure model convergence.

Neural Network

- **Architecture:**
- **Input Layer:**
 - Dense(32, activation='relu')
 - Hidden Layer: Dense(16, activation='relu')
- **Output Layer:**
 - Dense(1, activation='sigmoid')
- **Training:**
 - Optimizer: Adam (lr = 0.001)
 - Parameters: 10 Epochs, Batch size of 256.

Demo



Link: <https://youtu.be/O41fVjY96qk>

Model Evaluation: Raw Data & Metrics

1. Logistic Regression

- **Key Metrics:**
 - Error: 0.797%
 - Accuracy: 99.20%
 - Precision: 16.54%
 - Recall: 89.80%
 - F1 - Score: 0.279

	Predicted 0	Predicted 1
Actual 1	56420	444
Actual 0	10	88

2. Neural Network

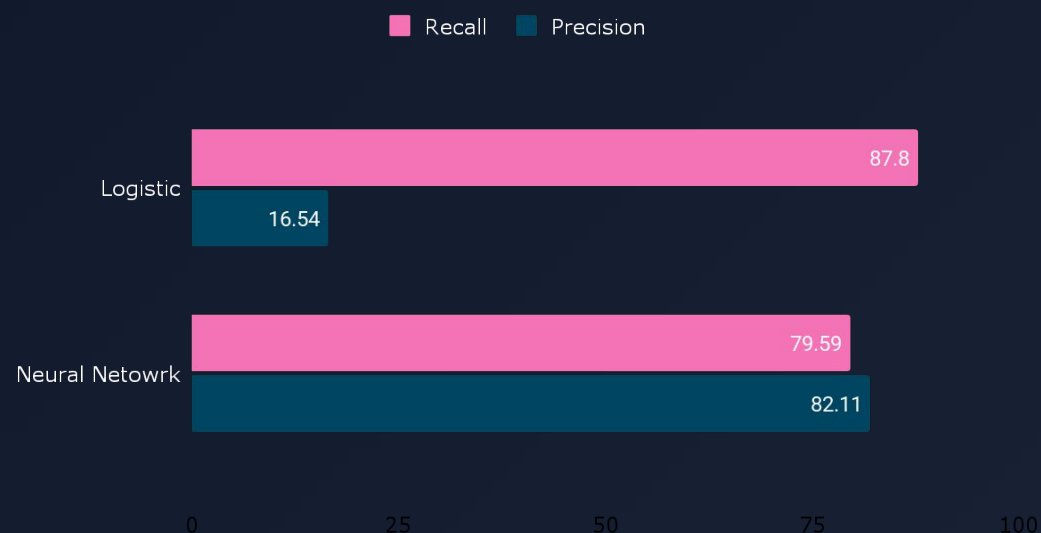
- **Key Metrics:**
 - Error: 0.065%
 - Accuracy: 99.94%
 - Precision: 82.11%
 - Recall: 79.59%
 - F1 - Score: 80.83%

	Predicted 0	Predicted 1
Actual 1	56487	17
Actual 0	20	78

Experimental Results & Analysis

Performance

Performance

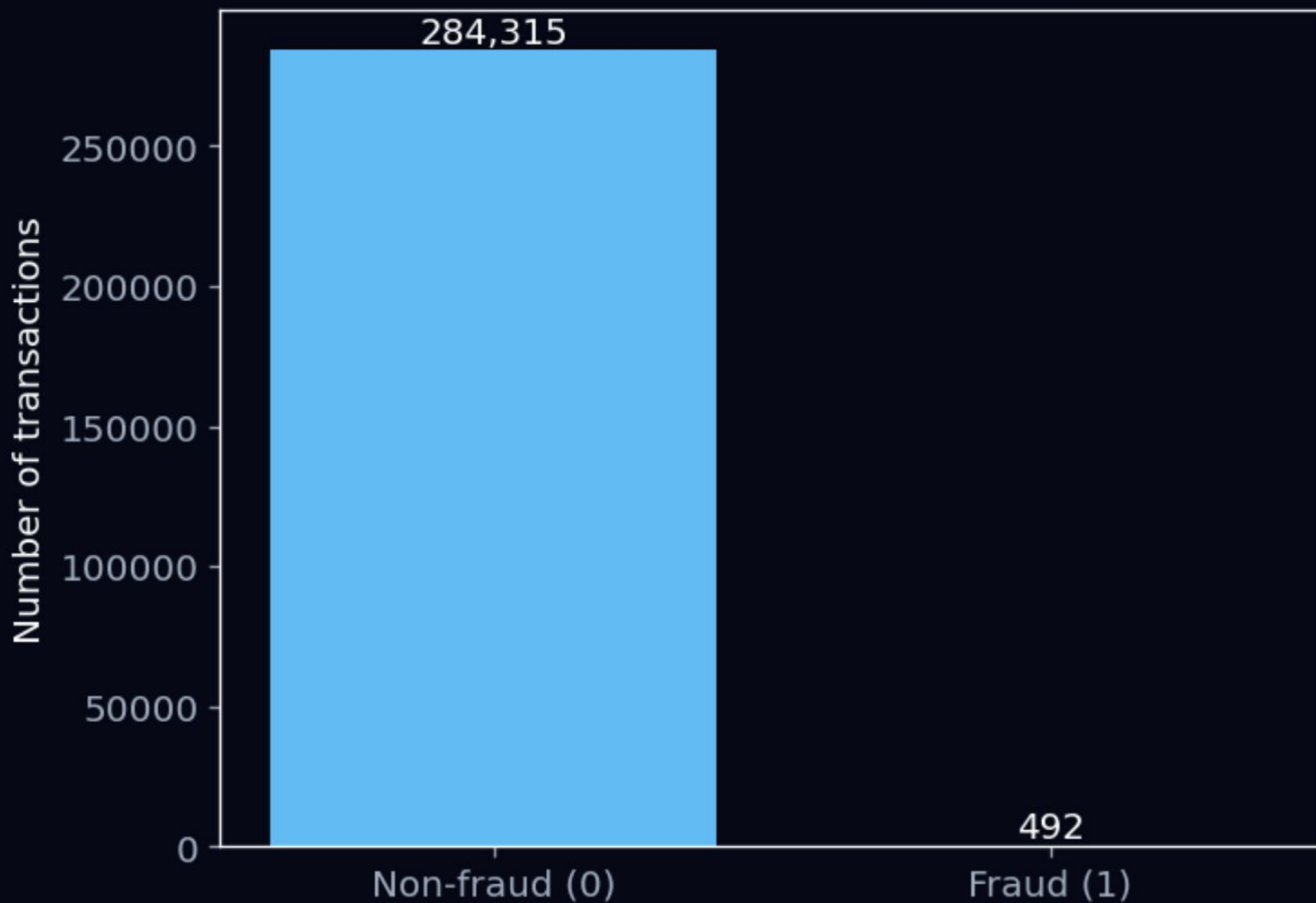


Model Analysis & Impact

- **Logistic Regression**
 - The model is very at detecting fraud - it catches most fraudulent transactions
 - However, it incorrectly flags many legitimate transactions as fraud
 - Logistic regression is linear so it struggles with the nonlinear structure of the fraud data
 - This class imbalance forces the model to be biased toward detecting fraud but it also causes many false alarms.
- **Neural Network**
 - Can capture nonlinear patterns in the data that the logistic regression couldn't
 - Reduced false alarms greatly
 - Still catches most fraud cases.
 - Overall improvements show that the NN is a better fit than LR.

Data Visualization

Class Balance: Fraud vs Non-Fraud



- Less than 0.2% of data consists of fraud cases
- Dumb model that always predicted “non-fraud” would be ~99.8% accurate
- Focus on precision, recall, and F1, not just accuracy

Confusion Matrices: Model Comparison

**Logistic Regression
(threshold = 0.8)**

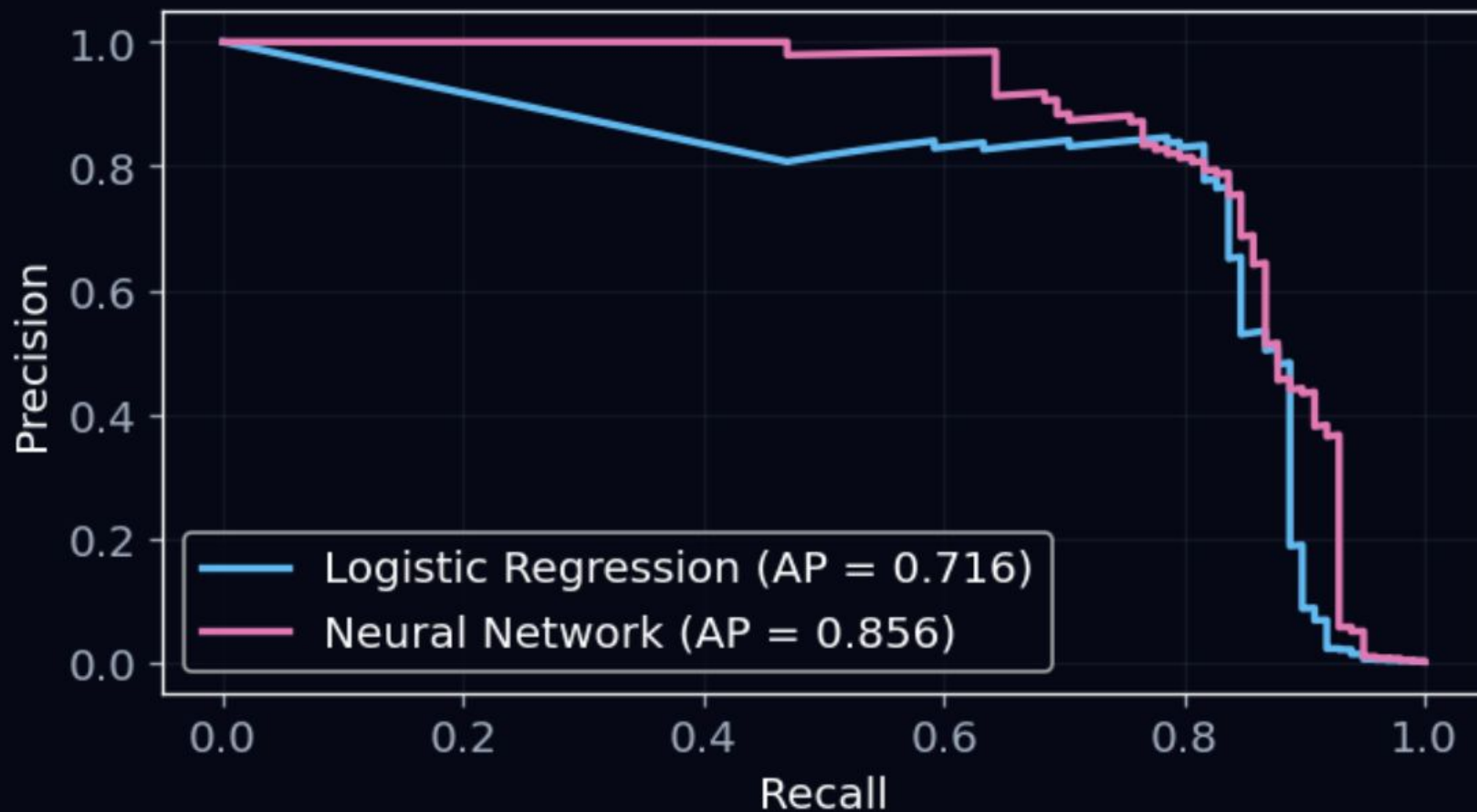
True label	Predicted label	
	0	1
0	56420	444
1	10	88

**Neural Network
(threshold = 0.5)**

True label	Predicted label	
	0	1
0	56847	17
1	20	78

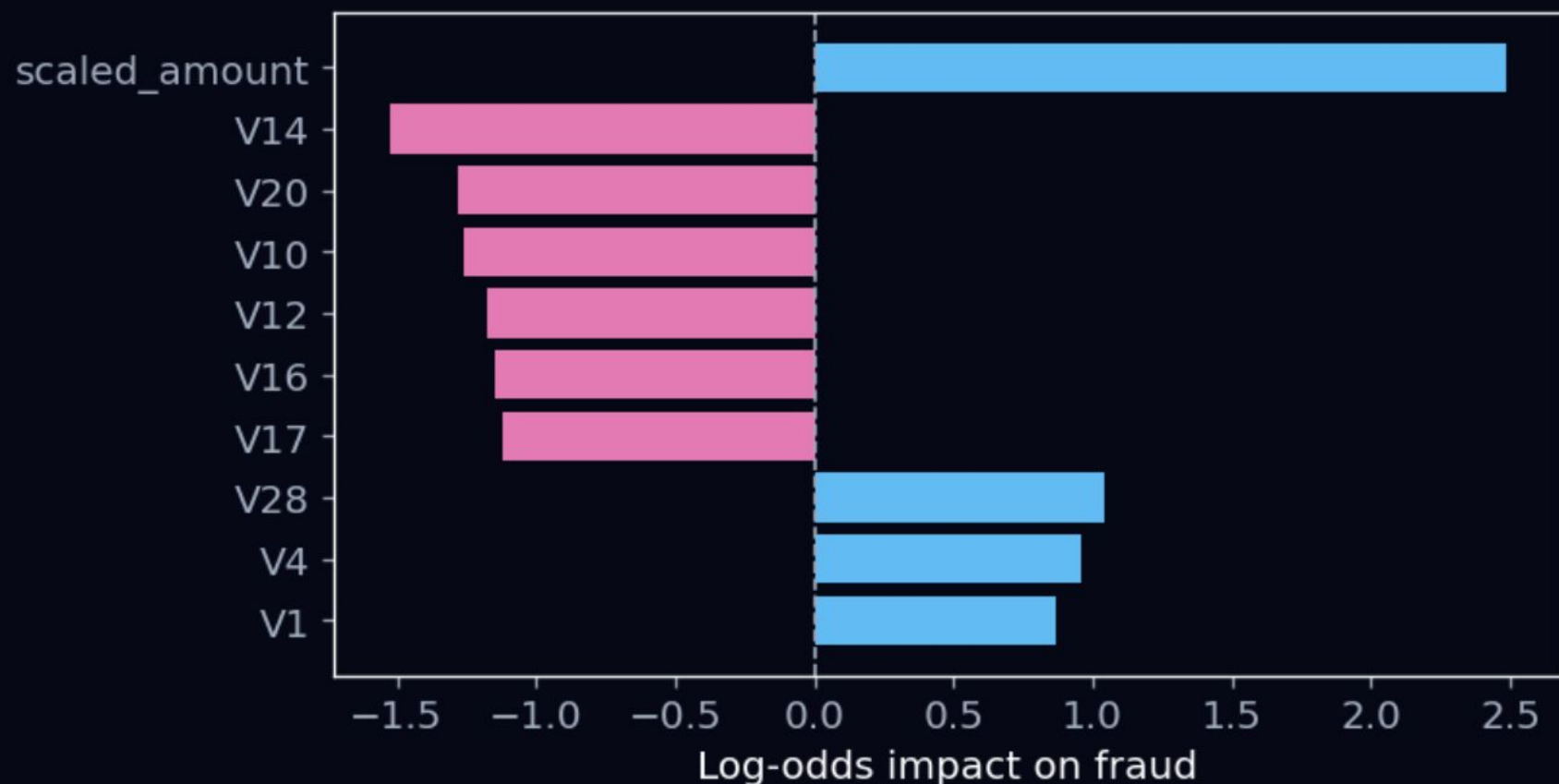
- LR model is cautious about identifying frauds, but misses many cases of actual fraud
- NN model has fewer false alarms but lets a few more fraudulent cases slip through

Precision-Recall Curves (Imbalanced Fraud Detection)

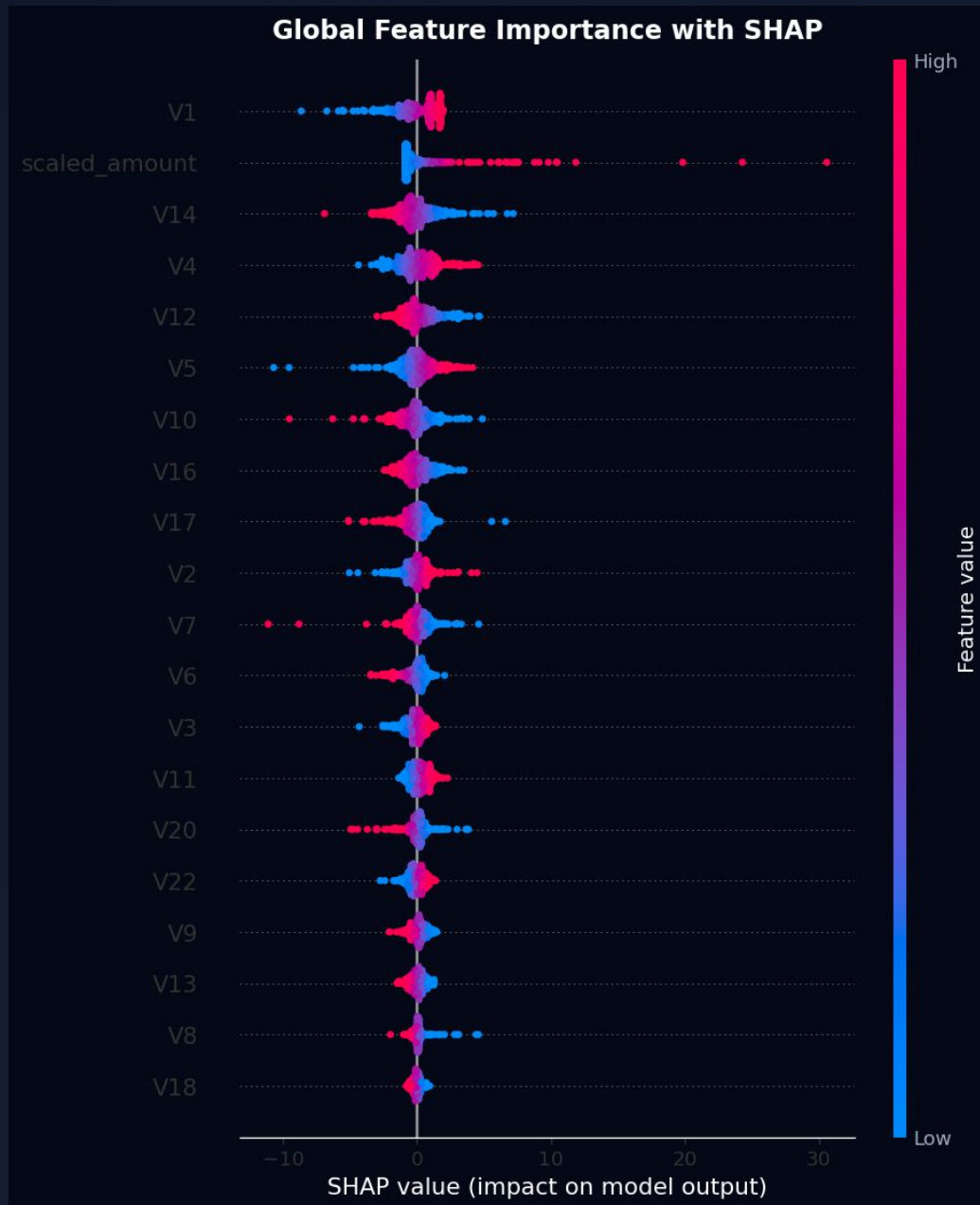


- NN curve sits above LR for the most part
- AP score for NN is higher, so NN gives us a stronger precision-recall tradeoff

Top 10 Logistic Regression Feature Weights



- Small set of features has big influence on model's fraud decision
- Model is looking at real signals, not making predictions based on noise



- For V4, pink dots on the right mean high V4 values increase model's fraud scores, blue dots on the left mean low V4 values lower it
- For V10/V14, pink dots on the left mean high values of both push prediction towards non-fraud, vice-versa
- Explainable AI >> looking at WHY a model predicts fraud, necessary for transparency in the finance industry

Future Work

Hyperparameter Tuning: Optimize the Neural Network (Layers, Neurons, Dropout) to boost recall without losing precision.

Ensemble Methods: Test XGBoost or LightGBM, which often perform exceptionally well on tabular data.

SMOTE: Apply Synthetic Minority Over-sampling Technique to training data to handle imbalance better than simple class weights.

Project Work Breakdown

Samriddhi: Code and Presentation

Suhani: Data Visualization and Presentation

Altaf: Model Design and Presentation

Nishitha: Model Analysis and Presentation

Thank you!

Links

Code: https://colab.research.google.com/drive/1sMVlgn14lskEm_cgFsqEKDuSZ3Nrr3hv#scrollTo=7e709ba6

Video Demo: <https://www.youtube.com/watch?v=O41fVjY96qk>