

Alejandro Rodriguez
CAP5619
Dr. Cuneyt Akcora
April 21st, 2025

Amazon Fraud Report

Chosen Datasets:

In this script, we are addressing the existing class imbalance in the two chosen datasets. For this we are testing two datasets: credit card (ccfraud) and twitterbot. These two were chosen solely for that reason, ccfraud has ratio of .18% and twitter with 33.10% of class ratio. There are multiple techniques to deal with class imbalance in a dataset (undersampling, oversampling and SMOTE) issue all of which are employed in this script.

Model Architecture

The model of choice is a feed-forward multi-layer perceptron (MLP) classifier to label fraudulent transactions and bot accounts. The architecture has two hidden layers (64 and 32) with ReLU activation separated by 0.2 dropout for regularization ending with a sigmoid output. It is using Adam to optimize backpropagation, a learning rate of $1e-3$ (.001) under binary-cross-entropy for loss calculation, with early-stopping for efficient convergence for our model. We established 50 epochs to give our code enough computing time to converge on the global minimum. In terms of our test train split we decided on going with 80-20 split which is the most common split in machine learning.

To mitigate extreme class imbalance, we apply the three techniques (oversampling, undersampling, and SMOTE). For comparison a baseline model was created for both datasets to gauge for metric improvements.

Credit Card Dataset Results:

The baseline model reports a ROC AUC of 96.73% with a recall of .075 and an f1-score of .82 for the minority class

SMOTE reports a ROC AUC of 94.95% with a recall of .79 and an f1-score of .79 for the minority class.

Undersampling reports a ROC AUC of 92.53% with a recall of .96 and an f1-score of .01 for the minority class.

Oversampling reports a ROC AUC of 94.38% with a recall of .79 and an f-score of .75 for the minority class.

SMOTE gave the best results, gave us a mix of high ROC AUC with a higher precision and somewhat similar f1-score. These results were also true for the twitterbot dataset, SMOTE performed better and addressed the class imbalance that we had for both datasets. Refer to [tables](#) at the end of report.

xAI Implementation:

For explainability in our model, we employed LIME to understand what features are most relevant for the model's predictions. For SMOTE which was our best performer LIME shows that the features that has the most predictive impact were the following: V19, V4, V7, V9, and V17 (credit card). For the twitter dataset SMOTE showed the features with the strongest predictive power were followers_count_log_scaled, favourites_count_log_scaled, geo_enabled, average_tweets_per_day_log_scaled, and account_age_scaled.

Conclusion:

Class imbalance proved to be the single biggest factor skewing our initial results. In the raw data, the model gravitated toward the majority class, producing deceptively high overall accuracy while missing a meaningful share of frauds and bots—precisely the observations that matter most in practice. Once we re-balanced the data, minority-class recall rose sharply (e.g., from 75 % to 79 % in ccfraud and from 74 % to 77 % in twitterbot with SMOTE) without materially sacrificing ROC-AUC, confirming that the network had previously under-exposed itself to rare patterns rather than lacking discriminative power. Explainability with LIME reinforced that conclusion: before resampling, the model's top features were dominated by broad, majority-oriented signals, whereas after SMOTE the same legitimate fraud and bot cues—V19 and V4 in credit-card transactions, or anomalous follower ratios and tweet frequency in Twitter accounts—gained salience. This shift shows that balancing techniques did not invent spurious correlations; they simply amplified domain-plausible signals that had been drowned out by majority noise, leading to a model that is both more vigilant and more interpretable where it counts.

Results:

Credit Card Dataset:

--- Evaluating Baseline Model ---

1774/1774  1s 281us/step

==== Baseline Model ====

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56651
1	0.91	0.75	0.82	95
accuracy			1.00	56746
macro avg	0.95	0.87	0.91	56746
weighted avg	1.00	1.00	1.00	56746

[[56644 7]

[24 71]]

ROC AUC: 0.9673808888959085

--- Evaluating UnderSampling Model ---

1774/1774  0s 224us/step

==== UnderSampling Model ====

	precision	recall	f1-score	support
0	1.00	0.36	0.53	56651
1	0.00	0.96	0.01	95
accuracy			0.36	56746
macro avg	0.50	0.66	0.27	56746
weighted avg	1.00	0.36	0.53	56746

[[20595 36056]

[4 91]]

ROC AUC: 0.9253079566579863

--- Evaluating SMOTE Model ---

1774/1774  0s 228us/step

==== SMOTE Model ====

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56651
1	0.78	0.79	0.79	95
accuracy			1.00	56746
macro avg	0.89	0.89	0.89	56746
weighted avg	1.00	1.00	1.00	56746

[[56630 21]

[20 75]]

ROC AUC: 0.9495869910783385

--- Evaluating OverSampling Model ---

1774/1774  0s 224us/step

==== OverSampling Model ====

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56651
1	0.72	0.79	0.75	95
accuracy			1.00	56746
macro avg	0.86	0.89	0.88	56746
weighted avg	1.00	1.00	1.00	56746

[[56622 29]

[20 75]]

ROC AUC: 0.9438878302886835


Twitterbot Dataset:

```

==== Baseline ====
      precision    recall  f1-score   support

     0       0.88       0.94       0.91       5003
     1       0.86       0.73       0.79       2485


 accuracy
macro avg       0.87       0.84       0.85       7488
weighted avg       0.87       0.87       0.87       7488

[[4697 306]
 [ 664 1821]]
ROC AUC: 0.9275033772493042
234/234  0s 258us/step

==== UnderSample ====
      precision    recall  f1-score   support

     0       0.89       0.91       0.90       5003
     1       0.81       0.76       0.79       2485

 accuracy
macro avg       0.85       0.84       0.84       7488
weighted avg       0.86       0.86       0.86       7488

[[4562 441]
 [ 584 1901]]
ROC AUC: 0.9200611624976723
234/234  0s 249us/step

==== SMOTE ====
      precision    recall  f1-score   support

     0       0.89       0.92       0.90       5003
     1       0.83       0.77       0.79       2485

 accuracy
macro avg       0.86       0.84       0.85       7488
...

[[4563 440]
 [ 571 1914]]
ROC AUC: 0.9211594170258408

==== OverSample ====
      precision    recall  f1-score   support

     0       0.89       0.91       0.90       5003
     1       0.81       0.77       0.79       2485

 accuracy
macro avg       0.85       0.84       0.85       7488
weighted avg       0.86       0.86       0.86       7488

[[4563 440]
 [ 571 1914]]
ROC AUC: 0.9211594170258408

```