Evaluating the Performance of Machine Learning Models for Credit Card Fraud Detection Using SMOTE

Kyle Dennis
DS 480
Quinnipiac University
Spring 2025

Abstract

Fraud detection remains a significant challenge across various industries, especially in the financial domain, where companies lose millions annually. The biggest roadblock is the data itself- most online transactions are legitimate, making it challenging to predict fraud. This study investigates which of the three machine learning models—logistic regression, random forest, and gradient boosting—best predicts credit card fraud and how SMOTE (Synthetic Minority Over-sampling Technique) affects model performance. Using a publicly available credit card dataset with significant class imbalance, I used preprocessing techniques, specifically SMOTE oversampling, to enhance fraud detection across these models. In real-world applications, most transactions are classified as legitimate since fraudulent transactions are rare. Therefore, analyzing model accuracy alone does not provide a sufficient assessment. I used several evaluation metrics—accuracy, precision, recall, F1-score, and PR-AUC—to evaluate model performance. These metrics provided insight into how well the models identified fraud when classifying new transactions. The results demonstrated that random forest models best predicted credit card fraud in the test set. Random forests had the most balanced performance, while gradient boosting identified the most fraud. Logistic regression had the lowest false alarm rate. SMOTE oversampling decreased overall accuracy but significantly increased the number of fraudulent transactions caught. These results demonstrate that model selection and oversampling can address different goals of fraud detection systems. Some oversampled models detect more fraud, while others generalize better and produce fewer false alarms.

Credit card companies can leverage the power of these models to save millions on losses each year while protecting their customers.

Introduction

In 2026, credit card fraud worldwide is estimated to reach 43 billion dollars (Rej). A significant contribution to this problem is the increasing number of online purchases. To help optimize online purchasing and maximize customer convenience, customer credit card data is stored in databases. These databases, however, are constantly at risk of being exposed to data breaches. These breaches are a significant reason why fraud is so prevalent. Current estimates suggest up to 80% of all credit cards in circulation have been compromised by some hack or data breach (Rej). As a result, credit card companies face huge losses each year while private customer data is exposed.

Due to the growing nature of this issue, many credit card companies have deployed machine learning models to help identify fraud. For example, JPMorgan Chase, a multi-billion-dollar corporation, invests heavily in fraud detection. Every year, they monitor about $1.1 trillion in online payments. They use both supervised and unsupervised machine learning techniques to detect fraud. This approach resulted in over a 50% reduction in credit card fraud in the last five years within the company (Singh). The models implemented by these companies have a significant impact. They are trained on large datasets and are used to determine whether new online transactions are likely to be fraudulent. For example, when an unusual transaction happens, the cardholder's credit card company could flag it using their real-time

fraud detection models. If the models are accurate and deployed at scale, they could effectively prevent fraudulent activity and protect customers.

Despite the promising potential of these models, there is a significant challenge to achieving high accuracy. This challenge relates to the nature of credit card data. Current estimates suggest that less than one percent of credit card transactions are fraudulent ("Credit Card Fraud Statistics for 2025"), while the remaining transactions are legitimate. In other words, the data tends to be highly "imbalanced." Machine learning models trained on imbalanced data tend to be biased towards the majority class. This bias leads to models not training properly- they are less likely to pick up on the underlying patterns of fraudulent transactions ("Imbalanced Datasets").

This paper aims to answer the following questions: 1) Which machine learning model best predicts credit card fraud? 2) How does SMOTE oversampling affect model performance? The answers to these questions can help provide insight into how well machine learning models classify real transactions. SMOTE in particular is worth investigating because it has the potential to mitigate the data imbalance problem. Using credit card data, I prototyped three classification models—logistic regression, random forest, and gradient boosting. The dataset contains 284,807 real credit card transactions made by European cardholders in 2013 (*Credit Card Fraud Detection*), with only a small percentage of fraudulent transactions (approximately 0.172%). I oversampled the dataset's minority class using the SMOTE technique (likebupt et al.) to address class imbalance. I created three training sets, including two oversampled sets generated with SMOTE, which had higher rates of fraudulent transactions. This approach helped balance the

minority class. After data preprocessing, I fit each classifier on the training sets. For model evaluation, I analyzed several accuracy metrics.

Methods

Dataset

The dataset used in this study contains credit card transactions made in September 2013 by European cardholders. This dataset was collected and analyzed through a research collaboration between Worldline and the Machine Learning Group of ULB, focusing on big data mining and fraud detection. The dataset contains 284,807 real credit card transactions, with only 492 fraudulent transactions—approximately 0.172% of the total, indicating a significant imbalance (*Credit Card Fraud Detection*).

The features in this dataset I chose as predictors for all models were **"Time," "Amount," "V1,"** and **"V2."** The **"Time"** feature indicates the number of seconds between a given transaction and the first transaction in the dataset. The **"Amount"** feature represents the transaction amount. I chose these particular features because they are the only features not transformed by a Principal Component Analysis (PCA) and are thus directly interpretable. Additionally, I selected them because they are commonly collected by credit card companies, making them good candidates for predicting fraud. The **"V1"** and **"V2"** features represent the first two principal components in the dataset (*Credit Card Fraud Detection*). I included these two components since I believed they were enough to capture substantial variance in the original dataset. I only used this set of features for model fitting to maintain consistency and

simplicity across models. I did not explore feature selection in depth since the primary goal of this paper is to evaluate the performance of different models under similar conditions.

I used the feature **"Class"** as the target variable for the models. This binary variable indicates whether a transaction was fraudulent (1) or legitimate (0) (*Credit Card Fraud Detection*). The model evaluation metrics discussed later assessed the accuracy with which the models predicted this target variable.

Data Preprocessing

After importing the dataset, I checked for null values and duplicate transactions. No null values were present, but I identified 1,081 duplicate transactions. I kept these transactions in the dataset since they could represent recurrent payments or potentially fraudulent activity. I scaled the dataset before splitting it into training sets to avoid performance issues. Specifically, I scaled the features **"Time"** and **"Amount."** I did not scale **"V1"** and **"V2"** further since they are principal components and already scaled.

I split the scaled data into a training set (80% of the dataset) and a test set (the remaining 20%). I performed a stratified split to ensure the test set contained an adequate number of fraudulent transactions. Given the extreme class imbalance in the original dataset, this process was crucial for model evaluation. Without a stratified split, there would have likely been insufficient fraudulent transactions in the test set for model evaluation. I performed

preprocessing steps using the Pandas library (*Pandas*) and the remaining steps using Scikit-Learn (*Scikit-Learn*).

To address the extreme class imbalance in the dataset, I applied SMOTE (Synthetic Minority Over-sampling Technique) to the training set. SMOTE generates artificial data points from the minority class by considering "neighbors" in the feature space- data points close to each other. New data points are then randomly generated between these neighbors to resemble new data points similar to the neighbors (Maklin). In this context, the goal was to simulate new fraudulent transactions similar to the fraudulent transactions in the dataset. This process resulted in more balanced training sets, containing much higher proportions of fraudulent data points than the original training set.

The original training set after the initial split contained 227,845 total transactions. Of those transactions, only 394 belonged to the fraudulent class, about 0.1729% of the total set. I oversampled this training set to create two additional training sets. After the first round of SMOTE oversampling, there were 295,686 transactions in the newly created training set, with 68,235 total fraudulent transactions- a huge increase from the original 492 fraudulent transactions. The new fraud class in this set made up 30% of the majority class, about 23% of the entire training set. After the second round of SMOTE oversampling, there were 386,666 total transactions, with 159,215 total fraudulent transactions- a significant increase from the original 492 fraudulent transactions. The new fraud class in this set made up 70% of the majority class, about 41% of the entire training set. This procedure resulted in three training sets: the original set, the first SMOTE oversampled set, and the second SMOTE oversampled

set. The SMOTE training sets are much more balanced, which significantly affected model performance. I performed SMOTE oversampling using the Imbalanced-Learn library (*Imbalanced-Learn Documentation*).

Model Fitting

I fit logistic regression, random forest, and gradient boosting models on each training set. The predictor subset mentioned earlier was the only subset of features used for model training to enable direct comparisons between models. I made this decision to provide a baseline comparison between model types. Additionally, I did not tune the models after training. That is, I used the default hyperparameters of the models. I trained the models using Scikit-Learn (*Scikit-Learn*).

Model Evaluation

After model training, I used the models to classify transactions in the test set as fraudulent or non-fraudulent. The test set contained 56,962 total transactions. Of these transactions, 98 were fraudulent, which preserved the original class distribution (0.172%). To evaluate model performance on the test set, I used a variety of classification metrics: overall accuracy, average cross-validated accuracy, precision, recall, and F1-score. These metrics are explained below ("Classification: Accuracy, Recall, Precision, and Related Metrics"; *F1 Score in Machine Learning*).

- **Accuracy** is the proportion of correct classifications:

$$\textbf{Accuracy} = \frac{correct\ classifications}{total\ classifications} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision** is the proportion of all the model's positive classifications that are actually positive:

$$\textbf{Precision} = \frac{correctly\ classified\ positives}{all\ positive\ classifications} = \frac{TP}{TP + FP}$$

- **Recall** is the proportion of all actual positives that were classified as positives:

$$\textbf{Recall} = \frac{correctly\ classified\ positives}{all\ actual\ positives} = \frac{TP}{TP + FN}$$

- **F1-score** is the harmonic mean of precision and recall:

$$\textbf{F1-score} = \frac{2 * precision * recall}{precision + recall}$$

Above, **TP** and **TN** represent true positive and true negative classifications, while **FP** and **FN** represent false positive and false negative classifications. I calculated these metrics for each model using Scikit-Learn (*Scikit-Learn*). For average cross-validated accuracy, I used the complete dataset for each model type. For this step, using Scikit-Learn, I performed stratified K-fold cross-validation with five folds (5 training/test set splits).

The final metric I used for model evaluation was the Area Under the Curve (AUC) of the Precision-Recall Curve (PR Curve). A PR Curve represents a classifier's precision and recall values at different classification thresholds. The x-axis is recall in this curve, and the y-axis is precision ("Precision-Recall Curve - ML"). I generated a PR Curve for each of the nine classifiers and AUC values for each. I displayed the AUC values in the plots. In addition, I created a point plot of these AUC values, categorized by model type and by training set. I used Matplotlib, Seaborn, NumPy, and Scikit-Learn for these steps (*Matplotlib*; Waskom; *NumPy; Scikit-Learn*).

Results

In total, nine models were fit. These include the binary classification models logistic regression, random forest, gradient boosting, and corresponding training sets. The training sets include the following: the original training set (80% of the total dataset), the first SMOTE set (oversampled set with new fraud distribution- 30% of the majority class), and the second SMOTE set (oversampled set with higher fraud distribution- 70% of the majority class).

Table 1 presents the computed accuracy metrics of these models, including overall accuracy, precision, recall, and f1-score (derived from test set predictions). The base models (those trained on the original set) had comparable overall accuracies (rounded to 1.00). The model with the highest precision was the base logistic regression model (0.85). The model with the highest recall was the second SMOTE-trained gradient boosting model (0.80). Lastly, the model with the best F1 score was the base random forest model (0.42).

As the fraud class proportion increased in the training sets, overall accuracy, precision, and f1 scores all decreased while recall increased. There was a notable tradeoff between high recall and high precision.

Table 1: General accuracy metrics (accuracy, precision, recall, and f1 score) for all nine models. Note- values were rounded to the nearest hundredth place.

Model	Training Set	Accuracy	Precision	Recall	F1 Score
Logistic Regression	Original	1.00	0.85	0.11	0.20
Logistic Regression	First SMOTE	0.97	0.03	0.50	0.06
Logistic Regression	Second SMOTE	0.93	0.02	0.68	0.04
Random Forest	Original	1.00	0.80	0.29	0.42
Random Forest	First SMOTE	0.99	0.13	0.47	0.21
Random Forest	Second SMOTE	0.99	0.11	0.48	0.18
Gradient Boosting	Original	1.00	0.46	0.23	0.31
Gradient Boosting	First SMOTE	0.96	0.03	0.71	0.06
Gradient Boosting	Second SMOTE	0.94	0.02	0.80	0.04

Table 2 presents the average cross-validated accuracies across all three binary classifiers. Although these values are similar, random forest had the highest average accuracy (99.87%).

Table 2: Average Cross-Validated Accuracies for each classifier.

Model	Logistic Regression	Random Forest	Gradient Boosting
Average Cross-Validated Accuracy	99.84%	99.87%	99.82%

Figure 1 illustrates the computed PR-AUC (Precision Recall- Area Under the Curve) values for each model. The random forest models had the highest PR-AUC values by a significant margin (all above 0.25) compared to the other model types. Also, there was more variability in PR-AUC values in the random forest model group compared to the other groups- the two SMOTE-trained random forest models were similar in PR-AUC values, but the base random forest model PR-AUC was significantly higher. In other groups, there is much less variation between the base model and the SMOTE-sampled model PR-AUC values.

Figure 1. A point plot of computed PR-AUC values (y-axis) for each model type (x-axis) and their corresponding training sets (categorized by color- see legend).

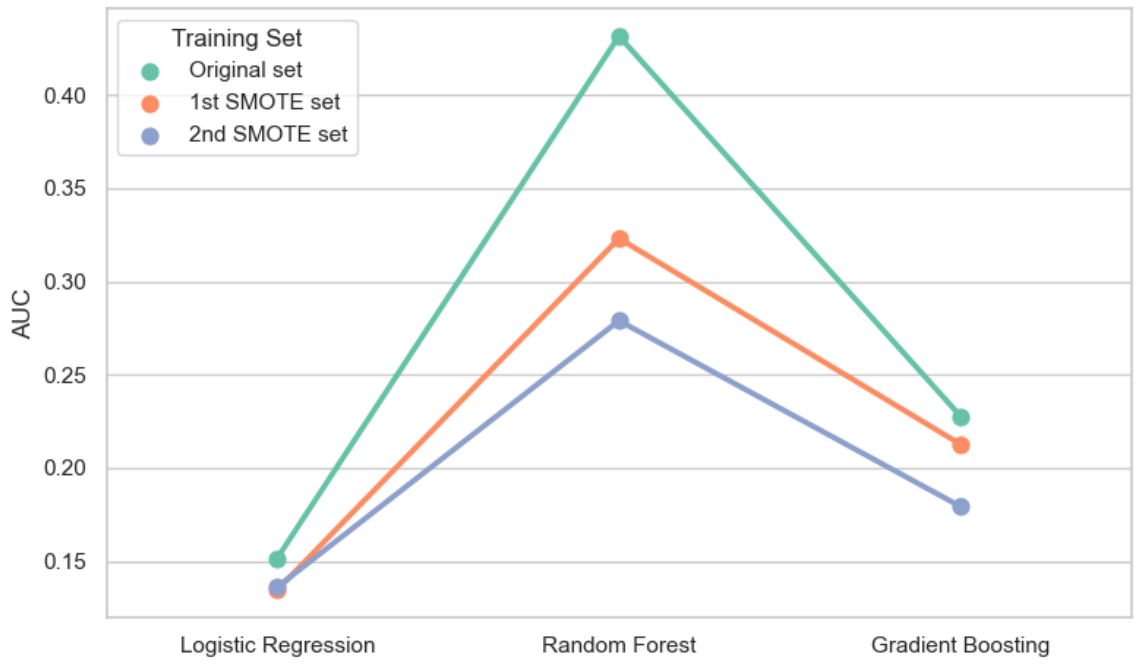

Figure 2 illustrates the Precision-Recall Curve of the random forest model trained on the original set. This model had the highest AUC value out of all the models (0.43). This curve, along with the other PR curves plotted, demonstrates that as recall rates increase, precision rates drop quickly. This pattern was consistent across all PR Curves created.

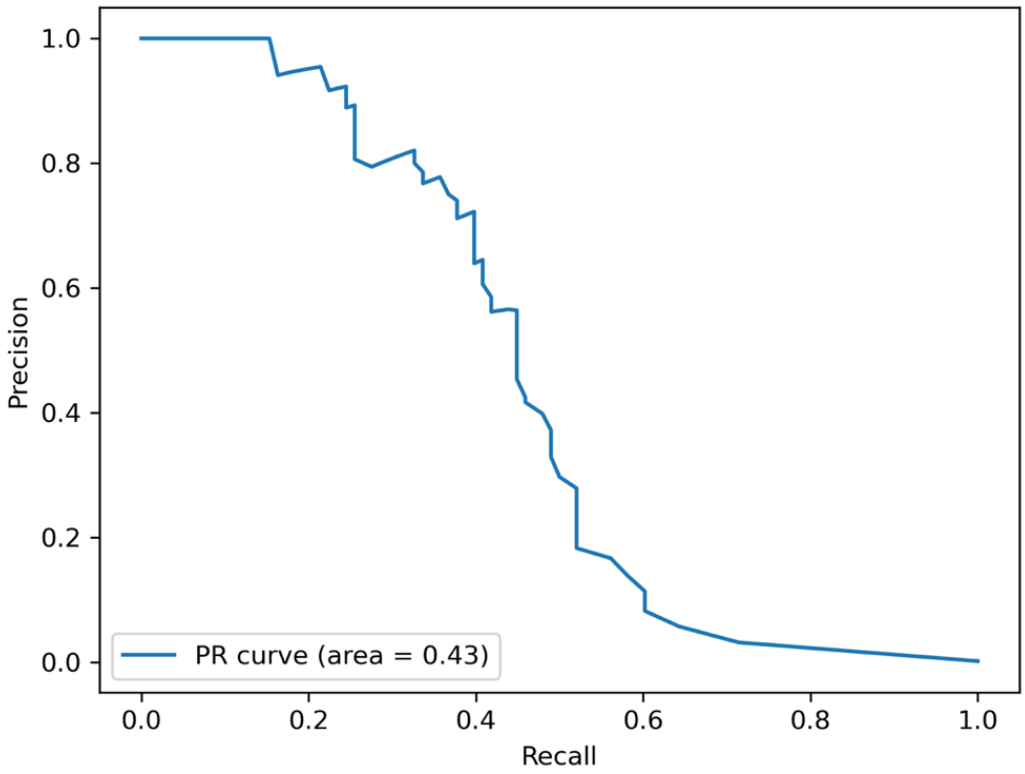

Figure 2. A Precision-Recall Curve of the random forest model trained on the original set.

Discussion

While the base logistic regression model had the highest precision (0.85), and the second SMOTE oversampled gradient boosting model had the highest recall (0.80), the base random forest model had the best F1 score (0.42) and PR-AUC (0.43), indicating the best overall balance between precision and recall for fraud detection. That is, the model correctly classified many fraudulent transactions in the test set without sacrificing too many incorrect fraudulent classifications. This insight is best captured by the F1 Score column in **Table 1**, and by **Figures 1** and **2**, where the base random forest model had the best PR-AUC. The next best-performing model overall was the base gradient boosting model, followed by the base logistic regression

model. These findings are consistent with recent literature, which also leveraged SMOTE to evaluate fraud prediction performance using these models and found random forest maintained the best balance, followed by the other two models (Ruchita et al.).

The random forest models, as a group, best-predicted fraud at different classification thresholds. This performance is illustrated in **Figure 1** and partially in **Figure 2** (I only displayed one PR Curve in this report). These results suggest random forest models have the potential to be the most accurate model type after tuning (for this data). The next best-performing model group at different classification thresholds was gradient boosting, followed by logistic regression. In **Figure 1**, the random forest group had the most variance across training sets. These results suggest that SMOTE sampling affected the random forest models the most in terms of PR-AUC.

In terms of average performance on new test sets, random forest models had the best performance. This performance is illustrated in **Table 2**, which contains average cross-validated accuracies. Although the average cross-validated accuracy for random forest was the highest (99.87%), it was not significantly higher than the other models. This is expected, given the data is highly imbalanced.

The gradient boosting model group detected the most fraud in the test set- especially the models trained on the SMOTE sets. This is illustrated in the recall column of **Table 1**. These results imply gradient boosting models are best at flagging fraud in the chosen test set. An important observation is that SMOTE sampling across all model groups significantly helped models flag more fraudulent transactions in the test set. However, this came at a significant tradeoff. As the fraud proportion increased in training sets from SMOTE oversampling, recall

rates increased while precision rates dropped quickly. This is seen in **Figure 2**, which illustrates a sharp drop-off in precision rates as recall rates increased. This drop-off was consistent across all Precision-Recall Curves plotted (note that I only showed one plot in this report). These results suggest that it is very difficult to build a model that flags many fraudulent transactions without incorrectly flagging many legitimate ones as fraudulent. If having a lot of false alarms in a fraud detection system is a problem for a credit card company, perhaps SMOTE sampling should be used carefully.

Although the logistic regression models did not catch as much fraud in the test set compared to other models, the base model had the highest precision (0.85), indicating a low false alarm rate. In addition, the logistic regression models took the least amount of time to train (under a minute each on my machine). Computational time could be essential in fraud detection systems deployed to production environments. Given that random forest and boosting models are ensemble models, they are expected to take significantly longer to train and tune. This issue grows as datasets become very large, which can be common, given how many credit card transactions occur daily- estimated to be over 147 million ("Number of Credit Card Transactions per Second & Year"). If a company has limited computational resources, logistic regression might be a favorable model.

Maintaining the right balance in fraud detection systems is vital. Since credit card companies can lose millions from fraud, they aim to minimize these losses by flagging as much fraud as possible. However, this should not be the only goal because false alarms can be a problem- they could lead to customer dissatisfaction and potential churn. In this study, I

assumed the "best" models exhibited this balance. Out of the metrics computed, the F1 score, and PR-AUC best quantify this balance.

The results of this study suggest that random forest models best predict credit card fraud, followed by gradient boosting and then logistic regression (out of the model types considered). SMOTE sampling resulted in a significant increase in fraud being caught in the test set. However, this had a critical tradeoff- while SMOTE oversampled models flagged more fraud, they were more likely to flag legitimate transactions as fraudulent.

In real-world applications, model selection and sampling techniques can address the specific goals of credit card companies. For example, one company might want to use machine learning models to flag credit card fraud while minimizing incorrectly flagged transactions. Random forest models could be used to optimize this balance. Another company might want to flag the most fraud, even if it leads to more false alarms. In this case, gradient boosting models coupled with SMOTE sampling can be useful tools for this goal. Lastly, if a company has limited resources and/or wants a low false alarm rate, logistic regression is a viable option.

In this study, my main findings were limited to predictions on a single test set. Future research should evaluate the metrics from **Table 1** and **Figure 1** using cross-validation. This would better estimate model performance on new transactions since there would be multiple test sets. Also, experimentation with different sampling strategies could be useful. Instead of just oversampling the minority class, undersampling the majority class or a mix of both can be done. A future research question could be: what is the optimal fraud proportion in training sets for peak model performance? In addition, future work should explore the tradeoffs of using simple models, like the ones used in this study, compared to advanced models, such as deep

neural networks. Also, feature selection should be explored to reveal the best predictors for credit card fraud. Lastly, model tuning should be addressed. The results found in this study were the baseline model results. They are subject to change after hyperparameter tuning. This process would likely yield new insights that are worth investigating. In future work, these processes can be carried out to develop the most accurate performing model possible for fraud detection when deployed to production.

References

- Rej, Matt. *Credit Card Fraud Statistics (2025)*. 9 Dec. 2024, https://merchantcostconsulting.com/lower-credit-card-processing-fees/credit-card-fraud-statistics/.
- Singh, Peeyush. "Guide to Detect Credit Card Fraud with Machine Learning." *Appinventiv*, 7 Jan. 2025, https://appinventiv.com/blog/credit-card-fraud-detection-using-machine-learning/.
- "Credit Card Fraud Statistics for 2025." *WalletHub*, https://wallethub.com/edu/cc/credit-card-fraud-statistics/25725. Accessed 6 March 2025.
- "Datasets: Imbalanced Datasets | Machine Learning." *Google for Developers*, https://developers.google.com/machine-learning/crash-course/overfitting/imbalanced-datasets. Accessed 7 March 2025.
- *Credit Card Fraud Detection*. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. Accessed 3 February 2025.
- likebupt. *SMOTE - Azure Machine Learning*. https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/smote?view=azureml-api-2. Accessed 24 April 2025.

- *Pandas - Python Data Analysis Library*. https://pandas.pydata.org/. Accessed 3 February 2025.
- *NumPy*. https://numpy.org/. Accessed 3 May 2025.
- *Matplotlib — Visualization with Python*. https://matplotlib.org/. Accessed 3 February 2025.
- Waskom, Michael. "Seaborn: Statistical Data Visualization." *Journal of Open Source Software*, vol. 6, no. 60, Apr. 2021, p. 3021. *DOI.org (Crossref)*, https://doi.org/10.21105/joss.03021.
- *Scikit-Learn: Machine Learning in Python — Scikit-Learn 1.6.1 Documentation*. https://scikit-learn.org/stable/. Accessed 3 February 2025.
- Maklin, Cory. "Synthetic Minority Over-Sampling Technique (SMOTE)." *Medium*, 14 May 2022, https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c.
- *Imbalanced-Learn Documentation — Version 0.13.0*. https://imbalanced-learn.org/stable/. Accessed 3 February 2025.
- "Classification: Accuracy, Recall, Precision, and Related Metrics | Machine Learning." *Google for Developers*, https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall. Accessed 10 April 2025.
- "Precision-Recall Curve - ML." *GeeksforGeeks*, 19 July 2019, https://www.geeksforgeeks.org/precision-recall-curve-ml/.
- *F1 Score in Machine Learning: Intro & Calculation*. https://www.v7labs.com/blog/f1-score-guide. Accessed 10 April 2025.
- Ruchita, M., et al. *"Leveraging Smote and Random Forest for Improved Credit Card Fraud Detection." 2024 International Conference on Sustainable Communication Networks and Application (ICSCNA)*, 2024, Theni, India, pp. 795–800. IEEE, doi:10.1109/ICSCNA63714.2024.10864103.
- "Number of Credit Card Transactions per Second & Year: 2024 Data." *Capital One Shopping*, https://capitaloneshopping.com/research/number-of-credit-card-transactions/. Accessed 1 May 2025.