Credit Card Fraud Analysis with Machine Learning

Key Words

Financial Fraud, Binary Classification, Class Imbalance, Anonymized Features, Random Forest, XGBoost, Oversampling, Undersampling, SMOTE, Recall, Precision, Data Leakage, imblearn, sklearn

Abstract

This project aims to use concepts from Data Mining Pipeline and Data Mining Methods to address the critical challenge of credit card fraud detection by developing and evaluating machine learning models to identify fraudulent transactions. My methodology involves comprehensive exploratory data analysis to understand the dataset's characteristics, followed by essential data preprocessing. I will compare 3 classification algorithms (Logistic Regression, Random Forest, XGBoost) and the 4 techniques (Random Oversampling, SMOTE, Random Undersampling, SMOTE + Tomek Links), in addition to baseline training data. Key findings are that the XGBoost model consistently performs well using F1-Score to rank. The Random Forest posts similar scores, but is significantly more costly to fit to the data. Both models perform the highest on the randomly oversampled training data.

The ultimate goal of credit card fraud detection is to minimize both financial losses due to undetected fraud and customer inconvenience from false alarms, thereby enhancing the security and trustworthiness of financial transactions.

Introduction

How can we effectively build a machine learning model to accurately identify fraudulent credit card transactions, minimizing false positives (legitimate transactions flagged as fraud) and false negatives (fraudulent transactions missed), given a highly imbalanced dataset of anonymized transaction data?

Credit card fraud leads to significant financial losses for banks, merchants, and consumers, eroding trust in financial systems. The nature of fraudulent transactions being rare events and often evolving in patterns makes their detection challenging using traditional rule-based systems.

One key existing limitation is the extreme class imbalance. Even with advanced sampling, models can still be biased towards the majority class. Achieving very high recall without low precision is an ongoing problem. In addition, financial institutions are reluctant to share raw data due to privacy regulations and competitive concerns.

My project's goal is to compare several methods of sampling techniques for helping models avoid majority class bias in datasets with large class imbalance.

Related Work

The dataset I am working with is extremely popular. There are many papers and Kaggle notebooks focused on it. Most notebooks and papers I've read so far include significant discussion around how they handle the imbalance of fraud. There also appears to be healthy debate between favoring simple models to more complex ones, I plan to explore this as well.

One issue that is often overlooked in many machine learning models created for this dataset is 'Data leakage.' Data leakage leads to models that perform worse than expected due to them being trained on information from outside the training set. This frequently occurs during data preprocessing or feature engineering steps. I will take care to handle scaling/normalizing after splitting the data into training and testing sets which should prevent the model learning the distribution of the test data.

Methodology

This project will involve several key data mining and machine learning techniques such as: Exploratory Data Analysis (EDA), Data Preprocessing, Machine Learning Models (Classification Algorithms), and Hyperparameter Tuning.

Data Understanding

From the data card on Kaggle we know that this dataset contains transactions made by credit cards in September 2013 over a 48 hour time period. I'll use fraud as the positive class. However, it only represents 0.17% of the data.

The features in this dataset, aside from 'Time' and 'Amount', have already been transformed by PCA by the provider of the data in order to maintain confidentiality. This limits the ability to interpret the models we create even further. It is possible to understand their statistical properties and relationship to the 'Class' variable, but I won't be able to infer meaning from them directly.

The most crucial aspect of the data is that it is highly imbalanced, therefore we'll need to take care with model building and evaluation.

First, I load the data and check the structure, column names, and initial values, and calculate descriptive statistics. Then, I conducted some analysis of the target variable 'Class' to confirm its imbalance. This includes visualizing the counts.

Next, I analyze the features, this includes the PCA transformed columns as well as the 'Amount' and 'Time' columns. This includes viewing the distributions of each, a correlation matrix, and outlier detection.

Fraud is more evenly distributed across the 48-hour time period than non-fraud, which peaks during working/daytime hours. It's also the case that fraud transactions are for smaller amounts.

From the correlation matrix we can see which features are correlated with 'Class'. These may be good candidates for models.

The last analysis is to further reduce the dimensionality using t-SNE on a subset of the data. This is to visually check if the fraudulent and non-fraudulent transactions have distinct clusters in lower-dimensional space. If so, I'll have some visual evidence that the classes are separable.

Data Preprocessing

The data has no missing values, but I did remove 1,081 duplicate rows.

I split the data into training and testing sets. I used stratification because of the extreme class imbalance. This ensures that the proportion of fraud to non-fraud is similar in both testing and training sets. If we didn't do this, it would be possible that none of the training data include any of the fraudulent transactions.

Since the 'Time' and 'Amount' columns are in their original scales, I applied a standard scaler to 'Time' due to its continuous, non-outlier prone nature, and a robust scaler to 'Amount' due to the presence of potential outliers in the transaction values. This scaling was done in order to keep the features with larger ranges from dominating distance calculations or gradient descent used by the models. This was applied to both training and testing sets, but separately in order to prevent data leakage.

The most critical step in preprocessing for this dataset is applying techniques to handle the class imbalance. If we didn't address the imbalance, the models would become biased towards the non-fraudulent transactions. We would achieve high accuracy just by predicting 'not fraud' on all of the data, but would have poor recall for the fraud transactions. I'll make use of the imbalanced-learn (imblearn) library to access these techniques.

Strategies to balance the data involve oversampling of the fraud labeled data, undersampling the non-fraud labeled data, or some combination of the two. There are model specific variable settings as well that are meant to handle class imbalance that I will discuss later. This will form the groundwork for my sampling comparison.

First, let's discuss oversampling the minority class. There are a couple of ways to do this, the simplest being randomly oversampling which just duplicates random instances of fraud labeled data. We'll use randomly oversampled data as a baseline. The original training dataset includes 378 instances of fraud and after oversampling the classes are equal sized at 226,602.

The next method is called Synthetic Minority Oversampleing Technique (SMOTE). Instead of copying datapoints to balance the classes, the SMOTE algorithm augments the data by creating new transactions that are similar, usually by just making small changes to existing data labeled as fraud. Again we end up with a balanced class count of 226,602 each.

Undersampling does the opposite. This technique throws data out until the classes are balanced. With this dataset, that means throwing out a lot of data, nearly all of it as this method ends up with balanced classes but only 378 each. This method is much more feasible with extremely large datasets where using a method like SMOTE would create an enormous dataset (instead of 2 days of credit card transactions, think on the order of months/years).

The last method I'd like to explore is one that combines the two methods called SMOTE + Tomek Links. This hybrid approach allows us to oversample using SMOTE and undersample using Tomek links. Tomek links are formed by two instances (nearest neighbors) where each belongs to a different class. Effectively removing instances that are near the decision boundary.

These are the 4 different ways of handling the class imbalance in addition to the baseline training data. Next, I will use these sets of training data across a few different models to see how they compare.

Baseline Model

I fit a logistic regression model from sklearn.linear_model on each of the training data sets as a baseline due to its simplicity and interpretability. I set the class weight parameter to 'balanced'. This setting makes it so that the model automatically adjusts weights inversely proportional to class frequencies.

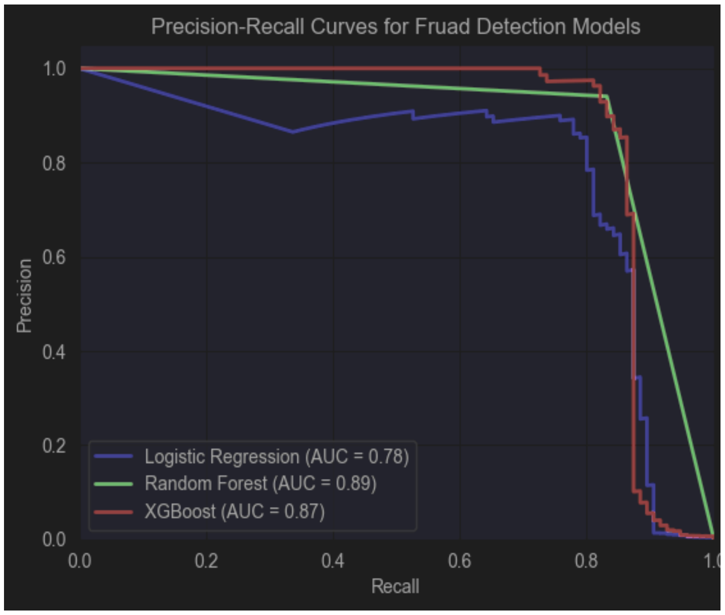
Complex Models

I fit a Random Forest ensemble model with 100 estimators as it can handle high dimensional data and has a 'class_weight' parameter for handling imbalanced data, just as the logistic regression model.

Then, I fitted an XGBoost Gradient Boosting Machine with 100 estimators and used the 'scale_pos_weight' parameter in order to explicitly bias the model towards improving recall for the minority class. The 'eval_metric' was set to 'logloss' as this is a classification problem.

Hyperparameter Tuning

Below is a plot that compares the performance of the three models in terms of their precision-recall trade-off. The closer a curve is to the top right corner, the better the model is. The higher the AUC score is for the curve, the better the model performs across different thresholds. The optimal decision threshold might be different depending on who is using the model. For example, a bank might want to avoid annoying customers and therefore prioritize very high precision. Law enforcement might prioritize very high recall to catch all possible fraud.



Data warehousing

I am working with one csv file. Warehousing data is certainly a key aspect of credit card data and I'm sure it is complex enough for its own detailed analysis. However, that has all been handled by the provider of this dataset and is beyond the scope of my project.

Evaluation

Given the highly imbalanced nature, even relatively simple models might achieve high accuracy due to the overwhelming number of non-fraudulent transactions. With that in mind, I focused on Recall and Precision metrics.

The baseline logistic regression model fit on the original training data yielded recall, precision, and F1-score of 0.91, 0.06, and 0.11, respectively. Recall is decent but precision is lower than desirable, this model flags a lot of non-fraudulent data as fraud. This model didn't seem to find any improvement related to how the training data was handled to balance the classes.

The Random Forest ensemble model resulted in recall, precision, and F1-score of 0.78, 0.97, and 0.87, respectively, when fit to the original training data (it does have model specific capability of handling class imbalance which explains the high performance). When fit to the random undersampled training data, recall increased to 0.91 but precision dropped off to 0.04. When fit to the SMOTE resampled data, recall improved to 0.83, precision lowered slightly to 0.94, and F1-score increased slightly to 0.88. The SMOTE + Tomek links method did not make any difference to the model performance. The best results came from the random oversampled training data producing a recall of 0.82, precision of 0.98, and F1-Score of 0.89.

The XGBoost model resulted in recall, precision, and F1-Score of 0.84, 0.92, and 0.88, respectively, when fit to the original training data. This model has the same capability of handling imbalanced classes on its own. I saw a similar drop in performance when fit to the random undersampled training data and increases in performance on the SMOTE resampled data. This algorithm, similar to the Random Forest model, performed the best on the random oversampled training data producing recall, precision, and F1-Score of 0.84, 0.93, and 0.88, respectively.

The XGBoost model is much faster to fit the training data compared to the Random Forest model, and their performance is so close that I would suggest this model to be the best performer.

The cost of misclassified data

A false positive in credit card fraud detection occurs when a transaction is labeled as fraudulent when it is actually a valid transaction. While a false positive doesn't involve direct financial loss from a fraudster, it does represent a missed revenue opportunity, damaged customer relationships, and significant operational expenditure.

A false negative occurs when a fraudulent transaction is labeled as valid. The cost of this transaction is the direct financial loss incurred from the undetected fraud in addition to the operational expenses to remedy the situation, and long term damage to customer relationships.

Discussion

While advanced sampling methods like SMOTE and SMOTE+Tomek Links are theoretically designed to improve performance on imbalanced datasets, these experimental results did not consistently demonstrate superior performance over simple random oversampling for the given models.

This could be due to this specific dataset, maybe the result of the PCA-transformed features. It could also be because the synthetic samples generated by SMOTE were too similar to each other.

As expected, the more complex models outperformed the baseline logistic regression model.

Limitations and Future Work

There are other, more advanced, strategies that I'd like to explore. A Neural Network might produce good results, or SVMs depending on their performance given the large dataset.

Over-reliance on synthetic data can also lead to overfitting if not handled carefully, and the synthetic data may not perfectly capture the complex, evolving patterns of real fraud.

Public access to credit card data is often limited due to its highly sensitive and proprietary nature. This Kaggle dataset has been anonymized and simplified. Without access to raw data and deep domain expertise, developing truly impactful insights is challenging.

One other interesting challenge with fraud data is the concept of 'concept drift'. This is due to fraudulent tactics evolving over time as fraudsters attempt to make their transactions look more 'normal'. This is something I would focus on if I had access to real time, non-anonymized data.

Conclusion

I used concepts from Data Mining Pipeline and Data Mining Methods to address the critical challenge of credit card fraud detection by developing and evaluating machine learning models to identify fraudulent transactions.

My methodology involved comprehensive exploratory data analysis to understand the dataset's characteristics, followed by essential data preprocessing.

I compared 3 classification algorithms using 4 different techniques to handle the data's class imbalance (in addition to baseline training data).

Key findings are that the XGBoost model consistently performs well using F1-Score to rank. The Random Forest posts similar scores, but is significantly more costly to fit to the data. Both models perform the highest on the randomly oversampled training data.

References

Bachmann, Janio Martinez (2019). Credit Fraud || Dealing with Imbalanced Datasets [Jupyter Notebook]. Kaggle. Retrieved from https://www.kaggle.com/code/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets/notebook

Preda, Gabriel (2021). Credit Card Fraud Detection Predictive Models [Jupyter Notebook]. Kaggle. Retrieved from https://www.kaggle.com/code/gpreda/credit-card-fraud-detection-predictive-models

Rutecki, Marcin (2023). Best techniques and metrics for Imbalanced Dataset [Jupyter Notebook]. Kaggle. Retrieved from https://www.kaggle.com/code/marcinrutecki/best-techniques-and-metrics-for-imbalanced-dataset

Brownlee, Jason. "SMOTE Oversampling for Imbalanced Classification." *Machine Learning mastery,* 3 July 2020, https://machinelearningmastery.com/smote-oversa mpling-for-imbalanced-classification/. Accessed 10 June 2025

Unit21**.** "False Positives: Causes, How to Calculate, & How to Reduce." *Unit21*, n.d., https://www.unit21.ai/fraud-aml-dictionary/false-positives. Accessed 9 June 2025.

Fletcher, David. “Understanding the True Cost of False Declines.” *ClearSale*, 31 Jan. 2020. Understanding the True Cost of False Declines Accessed 10 June 2025.

MIT CSAIL Data and Civil Society. "Imbalance, Outliers and Shift in Fraud Detection." *MIT CSAIL Data and Civil Society*, 2024, https://dcai.csail.mit.edu/2024/imbalance-outliers-shift/. Accessed 11 June 2025.

Appaji, Niranjan. "Balancing Act: Mastering Imbalanced Data with SMOTE and Tomek Link Strategies." *Medium*, 16 Aug. 2023, https://niranjanappaji.medium.com/balancing-act-mastering-imbalanced-data-with-smote-and-tomek-link-strategies-289f39597122. Accessed 9 June 2025.