Improving Anomaly Detection in E-commerce Websites: A Comparative Analysis of Parametric and Non-parametric Approaches

Master of Quantitative Economics, UCLA

Yujie Mao

Faculty Advisor: Zhipeng Liao

Date: May/31/2023

Table of Contents

Abstract

Anomaly detection is a critical task in various domains such as cybersecurity, online shopping, and healthcare. This paper evaluates the efficacy of parametric (logistic regression) and non-parametric methods (k-nearest neighbor, Random Forest, XGBoost) in anomaly detection, with a particular focus on e-commerce fraud. The analysis takes into account the challenge of class imbalance, employing three strategies: Random Under Sampling, Random Over Sampling, and Synthetic Minority Over-sampling Technique. However, it finds that despite balancing the data, models often overestimate risks, and the expected gains in model performance are not fully realized, underlining the complexity of class imbalance. Among the tested methods, non-parametric models outperform the logistic regression, with Random Forest demonstrating the highest sensitivity and specificity score. This study illuminates the strengths and limitations of machine learning techniques in predicting high-risk events in the presence of class imbalance.

1. Introduction

The exponential rise of the internet and eCommerce has completely revolutionized businesses across the globe. Electronic products are particularly prominent in this virtual marketplace, propelled by increased customer reliance on online platforms for their shopping needs. However, the surge in online transactions has also made eCommerce platforms vulnerable to illicit activities such as credit card fraud, money laundering, and other forms of deception. These nefarious activities can inflict serious damage on a business, from financial loss to compromised reputation, customer trust, and overall online shopping security.

Given these risks, the ability to detect anomalies is paramount. These anomalies often manifest as deviations from expected user behavior, indicating potential fraudulent activities. This study seeks to develop a predictive model that can effectively identify users on an eCommerce platform who have a high likelihood of engaging in such illegal activities. The successful application of advanced machine learning techniques in this model could substantially reduce human costs, ensuring robust business operations, customer protection, and a secure online shopping environment.

The class imbalance in the eCommerce fraud dataset, represented by an overwhelming 136,961 "normal" data instances compared to a mere 1,415 instances of fraud, forms a major impediment to this study. This massive disproportion, with fraud instances constituting just about 1.02% of the total data, introduces several complications that could detrimentally affect model performance.

The risk of biased model prediction is prevalent, as models trained on such data are more inclined to predict the majority class, i.e., "normal" instances. This occurs as models seek to minimize the overall error rate, which is primarily influenced by the majority class due to its sheer size. Consequently, the model could potentially demonstrate poor predictive performance for the minority class, in this case, instances of fraud.

Poor generalization is another pertinent issue. Models trained on such imbalanced data might deliver satisfactory performance on the training data but falter when faced with new, unseen data. This is a symptom of overfitting, where the model learns the characteristics of the majority class so well that it fails to discern the subtle patterns inherent in the minority class, consequently performing poorly when predicting anomalies.

High cardinality of categorical variables poses a significant challenge in machine learning applications. High cardinality refers to columns with values that are very uncommon or unique. In the context of categorical variables, it implies having a multitude of unique categories within a single feature. Generating dummy variable or one-hot encoding techniques to handle high cardinality categorical variable can be inefficient, leading to ineffective learning and increasing the computational resources needed for model training and scoring. High cardinality features could also inflate data size, extending model training and scoring time, which is particularly problematic when the need arises to score hundreds of millions of items as frequently as possible (Pawar, 2018). Appropriate feature engineering techniques should be adapted to avoid overfitting and ensure machine learning model's efficiency. Details for feature engineering for high cardinality variables will be further elaborated in the feature engineering section.

In the upcoming sections of the paper, the paper will delve into comprehensive details about the data, methodology, results, future works, and conclusion. Specifically, Section 2 describes the data and methodology, including descriptive statistics, data preprocessing techniques like feature engineering, standardization, normalization, and dataset division, and approaches to handle class imbalance. Subsequently, the section transitions to model development, detailing the application and parameter tuning of various machine learning models optimized for sensitivity. Section 2 concludes by providing a brief understanding of the confusion matrix and commonly used metrics for evaluating classification problems. In Section 3, findings from class imbalance corrections and the comparative performance analysis of various machine learning methods are discussed in detail. Section 4 looks at potential future work, including the calibration of probability predictions, utilization of one-hot encoding and the double lasso technique, and more data collection for feature engineering. Lastly, Section 5 concludes by synthesizing the key findings on model performance, class imbalance challenges, and the implications for e-commerce fraud detection.

2. *Data & Methodology*

2.1 *Descriptive Statistics*

The project has two csv files from eCommerce company: Fraud dataset and IP Address dataset. Fraud dataset contains 11 variables and 138376 rows. Each row in the Fraud dataset is a unique user's first transaction at the website. The data information provided by the IP address dataset allows us to map each transaction's IP address to a specific country. IP address dataset has 138846 rows and 3 columns.

Table 1: Descriptive Statistics for Selected Variables

Variable	Mean	Std.Dev	Min	Median	Max
user_id	200149.0322	115226.7815	2	200000.5	400000
purchase_value	36.9390	18.3211	9	35	154
age	33.1259	8.6236	18	33	76
ip_address	2154381000	1250563000	52093.5	2156471000	4294850000
class	0.0102	0.1006	0	0	1
interval_after_signup	5164377	3004642	1	5165848	10367970
signup_days_of_year	114.4124	66.3898	1	114	230
signup_seconds_of_day	43131.5367	24949.8508	0	43216	86398
purchase_days_of_year	174.1840	75.4106	1	174	350
purchase_seconds_of_day	43247.6002	24935.8299	0	43335.5	86399

2.2 Data Preprocessing

2.2.1 Feature Engineering

For every country stored in the IP address dataset contained an IP address range. If the IP address falls within the country's IP address range, this study will map the transaction's IP address from the fraud dataset to the corresponding country.

Several new features are engineered from categorical variable or timestamp in the existing dataset: The 'interval_after_signup' feature is constructed by calculating the time difference (in seconds) between the moment a user signs up and their first purchase. This is achieved by subtracting the signup timestamp from the purchase timestamp, which yields a time delta object. This object is subsequently converted to total seconds.

In addition, the 'signup_days_of_year' feature is created. This represents the specific day of the year that the user signed up. This is extracted directly from the signup timestamp.

The 'signup_seconds_of_day' feature is also constructed. It presents the exact time of user signup during the day but in terms of seconds. The total seconds are calculated by converting the hours and minutes into seconds from the signup timestamp and then adding these to the signup seconds.

Analogously, the 'purchase_days_of_year' and 'purchase_seconds_of_day' features are created to denote the day of the year of the purchase and the specific time of the purchase during the day, respectively. They are generated from the purchase timestamp in the same manner as their signup counterparts.

Transformation of categorical variables 'source' and 'browser' in the training set into binary variables through generating dummy variables. Additionally, the 'sex' variable is converted into a binary format where the male category is encoded as 1 and the female category as 0.

Next, an assumption was made that the more a device or an IP address is shared among multiple users, the higher the suspicion of fraudulent activities. To capture this, new features are created: 'n_dev_shared' and 'n_ip_shared', representing the number of times a particular device_id and an IP address appear in the training set, respectively. Another hypothesis is that fraudulent activities could be more common from countries with fewer visitors. To encapsulate this, a new feature 'n_country_shared' is constructed, which represents the frequency of each country in the training set.

After the generation of these new features, the original columns 'user_id', 'signup_time', 'purchase_time', 'device_id', 'ip_address', and 'country' are no longer necessary in their original form. Thus, these are removed from the dataset to prevent redundancy.

2.2.2 Standardization and Normalization

In this capstone project, Min-Max Normalization is applied to bring the newly engineered features ('n_dev_shared', 'n_ip_shared', 'n_country_shared') to the same scale, ranging from 0 to 1. This step is critical, especially for models that rely on feature scales such as logistic regression with regularization. Once the scaler is fitted, it is used to transform the same features in the training set, replacing the original values with the scaled ones. This transformed data is then used for model training.

Min-Max Normalization is favored over standardization in this capstone project due to its ability to confine values within a fixed range of [0,1], which is particularly beneficial for certain algorithms that do not operate well with negative numbers. Moreover, while standardization centers the distribution around the mean with a standard deviation of one, it doesn't bound values to a specific range, leading to values possibly extending into negative territories, which may not be suitable for the newly engineered features.

It's important to note that the same scaler fitted on the training data is later used to transform the corresponding features in the validation and test set. A new scaler is not fitted to prevent data leakage - using information from the test set in the model training process. The use of the same scaler ensures that the model evaluates its performance on data transformed in the same way as the training data.

2.2.3 Train, Validation, Test Split

The data is split into training, validation, and test sets using the 'train_test_split' function from the sklearn library. Data is then divided into a training set comprising 80% of the original dataset and a remainder set containing the remaining 20%. The remainder set is further divided into equal halves to form the validation and test sets. This resulted in a 80/10/10 split for the training, validation, and test sets, respectively.

2.3 Addressing Class Imbalance & Measurement

2.3.1 Three Common Class Balance Strategies

The task of mitigating the issues caused by class imbalance often involves strategies such as Random Under Sampling (RUS), Random Over Sampling (ROS), and Synthetic Minority Over-sampling Technique (SMOTE), all of which are implemented here using the default settings in the imbalanced-learn module. RUS is a technique aimed at balancing the class distribution by reducing the size of the majority class. It accomplishes this by randomly eliminating instances from the majority class until its size is equivalent to that of the minority class. On the other hand, ROS works by augmenting the size of the minority class through resampling. Instances from the minority class are randomly duplicated until the size of the minority class matches that of the majority class, leading to an artificially balanced dataset.

SMOTE, unlike RUS and ROS, generates synthetic instances of the minority class. For every instance in the minority class, it determines the 'k' nearest neighbors in the predictor space using the Euclidean distance, where 'k' is set to its default value of 5 in the imbalanced-learn module. It then computes the differences between the feature vector of the minority instance and its 'k' nearest neighbors. These differences, after being multiplied by a random number between 0 and 1, are added to the feature vector of the minority instance, thereby creating new synthetic data. This method introduces more variation into the minority class data, potentially mitigating overfitting risks in comparison with models trained on ROS data (Goorbergh et al., 2022).

2.3.2 Calibration Curve & Intercept

To gauge the effectiveness of RUS, ROS, and SMOTE in addressing class imbalance problems, inspiration is taken from the research by Goorbergh, Smeden, Timmerman, and Calster (2022). Their study investigated the influence of class imbalance corrections on model performance, focusing on aspects such as discrimination, calibration, and classification. Following their methodological approach, the calibration graph and calibration intercept are used as key measures for evaluating each class balancing strategy's effectiveness.

The calibration graph, also referred to as a reliability diagram, plays an essential role in predictive modeling and machine learning by visually illustrating the correlation between a model's predicted probabilities and actual outcomes. In a perfectly calibrated model, for instances predicted with a probability 'p', the proportion of those instances turning out to be actual positive outcomes should ideally also be 'p'. Therefore, a calibration curve that tracks the 45-degree line in a calibration graph is indicative of a well-calibrated model (Goorbergh et al., 2022).

The calibration intercept is another important concept. It is used to quantify whether the average probability estimates are too high, leading to overestimation (when the calibration intercept < 0), or too low, causing underestimation (when the calibration intercept > 0).

This capstone project is informed by the work of Stevens and Poppe (2019), which discussed the complexity and potential misinterpretation surrounding the calibration slope. In response to their findings, the project will concentrate on the calibration intercept and the use of calibration graphs for assessing model performance, rather than delving into the nuances of the calibration slope (Stevens & Poppe, 2019).

2.4 Model Development

The model development for this capstone project involves applying various machine learning models with their respective parameter tuning, specifically optimizing for sensitivity (recall) score.

A logistic regression model with both L1 and L2 regularization is first employed. The hyperparameters optimized in this model includes 'C', the inverse of regularization strength, where smaller values specify stronger regularization. For L1 regularization, 'C' is selected from a log scale range between 0.01 to 100, while for L2 regularization, 'C' is chosen from a list [0.01, 0.1, 1, 10, 100].

Next, a K-Nearest Neighbors (KNN) classifier is utilized, optimizing for the number of neighbors used in the computation, selecting from a range of 1 to 10. Following this, a Random Forest Classifier is employed, adjusting parameters such as 'max_depth' (the maximum depth of the tree, with choices of None, 5, or 15), 'n_estimators' (the number of trees in the forest, with choices of 10, 30, 50, or 100), and 'class_weight' (weights associated with classes, choosing from a list of weight dictionaries for class 1 [0.2, 1, 10]).

Finally, an XGBoost classifier is utilized, tuning hyperparameters like 'learning_rate' (the step size shrinkage used in the update to prevent overfitting, selecting from 0.001, 0.01, 0.1), 'n_estimators' (number of gradient boosted trees, choosing from 100, 500, or 1000), and 'max_depth' (maximum depth of a tree, selecting from 3, 5, or 7).

All models underwent grid search with 5-fold cross-validation for hyperparameter optimization with a focus on sensitivity, ensuring that the models are capable of correctly identifying the positive class. Confusion matrices and other performance metrics were calculated for the best models to further assess their performance. These models are then validated on a separate validation set and subsequently tested on a test set to assess their generalization ability. The Area Under the Curve (AUC) score is calculated for the final evaluation of the models' performance on the test set.

2.5 Model Evaluation Metrics

Conventional evaluation metrics: accuracy, can be deceptive when dealing with imbalanced data like our anomaly dataset. This is because a high accuracy score can be attained by merely predicting the majority class. Metrics like sensitivity, specificity, and F-1 score are more suitable for imbalanced data, as they emphasize the performance of the rare positive cases representing anomalies.

Table.2 Confusion Matrix

	Actual Negative	Actual Positive
Predicted Negative	True Negatives (TNs)	False Negatives (FNs)
Predicted Positive	False Positives (FPs)	True Positives (TPs)

The formulas for Precision, F-1 score, Sensitivity (Recall), Specificity are listed in the following:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Sensitivity\ (Recall) = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$Specificity = \frac{True\ Negatives}{True\ Negatives + False\ Negatives}$$

$$F1 = 2 \times \frac{Precision\ \times\ Recall}{Precision + Recall}$$

The Area Under the Curve (AUC) score is a performance measurement for classification models at various threshold settings. In simple terms, it tells us how well the model distinguishes between different classes - the higher the AUC score, the better the model is at predicting 0s as 0s and 1s as 1s. The major three metrics this study focused and used to evaluate model performance will be the sensitivity, specificity, and AUC score.

3. Results & Discussion

3.1 Class Imbalance Corrections

Figure 1. Calibration Curves: Logistic Regression across Different Balance Techniques

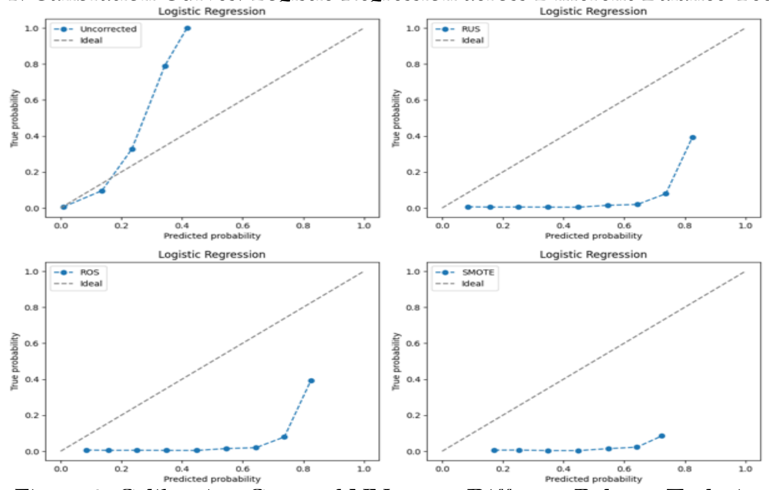


Figure 2. Calibration Curves: kNN across Different Balance Techniques

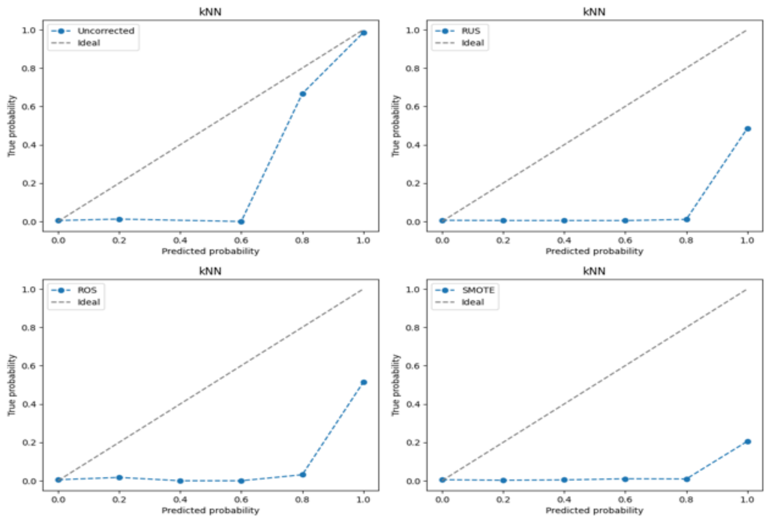


Figure 3. Calibration Curves: Random Forest across Different Balance Techniques

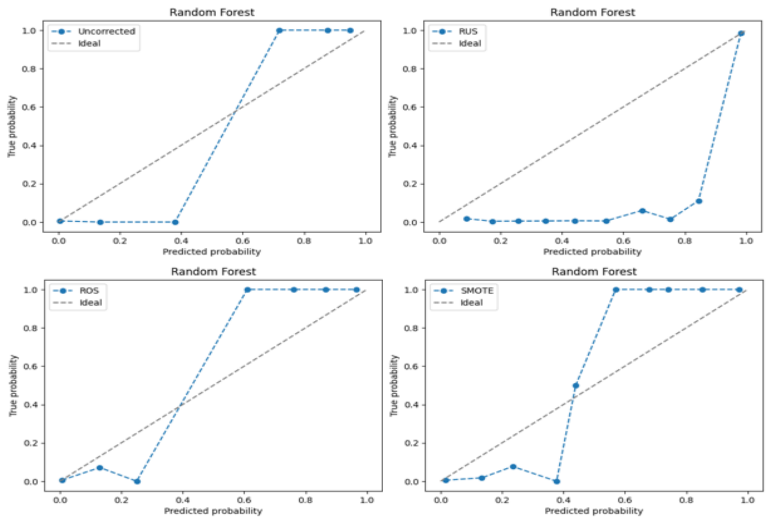
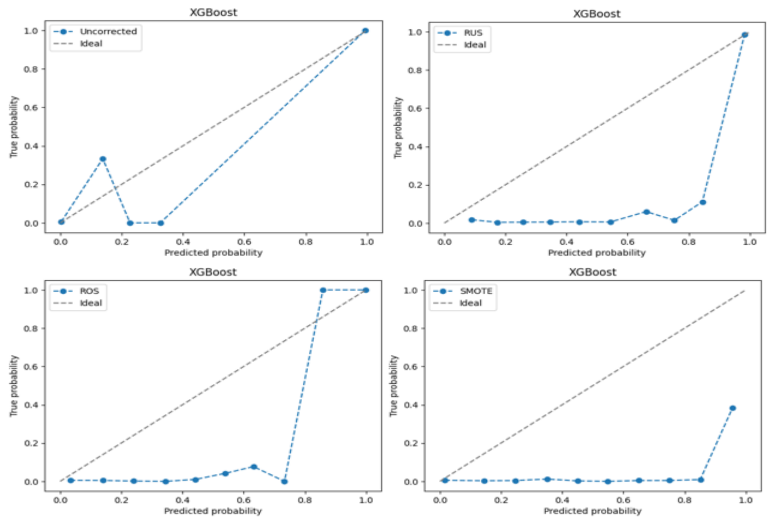


Figure 4. Calibration Curves: XGBoost across Different Balance Techniques



The calibration curves (Figure 1 – Figure 4) for the four machine learning models - Logistic Regression, k-Nearest Neighbors (kNN), Random Forest, and XGBoost - trained in the study, do not align with the ideal line in any of the cases. More specifically, Logistic Regression shifts from underestimating to overestimating after class balance (Figure 1). The kNN model consistently overestimates the probability of the positive class (Figure 2), while the Random Forest model seems to fluctuate around the ideal line without displaying a well-calibrated pattern (Figure 3). The XGBoost model overestimates the probabilities for both RUS and SMOTE (Figure 4).

Regarding the calibration intercepts as a measure of calibration error, the uncorrected models demonstrated a negative intercept, indicating a consistent tendency to underestimate the positive class. Specific calibration intercepts for uncorrected Logistic Regression, kNN, Random Forest, and XGBoost were -0.2531(-1.1146, 0.6084), -0.3637(-1.246, 0.5189), -0.2285(-0.9774, 0.5204), and -0.7454(-1.5508, 0.0600).

However, there are instances where it's impossible to calculate the calibration intercept. The calculation of calibration intercept requires the presence of both classes in the prediction. It is based on a comparison of predicted probabilities with actual class labels, which necessitates variety in these labels. If predictions consist of only one class, this comparison cannot be made, hence, the computation of the calibration intercept is not possible.

This situation arises for logistic regression model, kNN, XGBoost trained using RUS and SMOTE. Logistic regression under ROS also only had one class prediction result leading to an absence of calibration intercept and its corresponding 95% Confidence Interval. Calibration intercept for Random Forest model under RUS, ROS, SMOTE were -1.2024 (-1.5698, -0.8351), -0.1060 (-0.7242, 0.5120), -0.0617 (-0.5102, 0.3867) respectively. The calibration intercepts for ROS corrected kNN, XGBoost were -0.3636 (-1.2463, 0.5189) and -1.0021 (-1.4363, -0.5678).

Table 3 reveals a considerable disparity among algorithms when evaluating uncorrected models. Notably, Logistic Regression displays zero sensitivity, illustrating its ineffectiveness in identifying actual positive cases. Conversely, KNN, Random Forest, and XGBoost boast robust precision and specificity scores, although their sensitivity scores remain moderate.

Observations from RUS corrected models in Table 3 indicate a notable improvement in sensitivity across all algorithms, albeit at the expense of precision. This outcome aligns with Goorbergh et al.'s (2022) findings of a sensitivity-specificity trade-off.

The application of ROS to machine learning models, as demonstrated in Table 3, consistently enhances sensitivity, particularly for Logistic Regression. Intriguingly, Random Forest maintain identical precision, sensitivity, and specificity scores as their uncorrected counterparts.

When reviewing SMOTE-corrected models in Table 3, an increased sensitivity across all algorithms except Random Forest is noted. However, this improvement comes at the cost of reduced precision. Notably, the Random Forest model fails to identify any true positive cases, signaling potential challenges when applying SMOTE to certain algorithms.

In summary, the findings of this study are in alignment with those of Goorbergh et al. (2022), illuminating the intricate effect of imbalance correction methods on model calibration. Both investigations underscore the necessity for prudent application and interpretation of these methods in the setting of imbalanced data sets, with a specific focus on potential distortions in model calibration and their implications on decision-making.

Table 3. Comparative Performance Metrics of Imbalance Correction Methods on ML Algorithms

Algorithm	Correction Method	Precision	Sensitivity	F1-Score	Specificity
Logistic	Uncorrected	N/A	0.0	N/A	1.0
KNN	Uncorrected	0.9590	0.4861	0.6452	1.0
Random Forest	Uncorrected	1.0	0.4861	0.6542	1.0
XGBoost	Uncorrected	1.0	0.4861	0.6542	1.0
Logistic	RUS	0.0217	0.6389	0.0420	0.6972
KNN	RUS	0.0284	0.6111	0.0543	0.7802
Random Forest	RUS	0.2000	0.5208	0.2890	0.9781
XGBoost	RUS	0.0472	0.5625	0.0870	0.8805
Logistic	ROS	0.0216	0.6458	0.0418	0.6927
KNN	ROS	0.2664	0.5069	0.3493	0.9853
Random Forest	ROS	1.0	0.4861	0.6542	1.0
XGBoost	ROS	0.6577	0.5069	0.5725	0.9972
Logistic	SMOTE	0.0215	0.6458	0.0416	0.6907
KNN	SMOTE	0.0629	0.5556	0.1131	0.9130
Random Forest	SMOTE	0.0	0.0	N/A	0.9950
XGBoost	SMOTE	0.0646	0.5139	0.1148	0.9218

3.2 Model Results & Comparative Analysis

Logistic Regression

Logistic Regression is a parametric statistical model that works on several assumptions including the linearity of independent and dependent variables, independence of features, and the residuals following a logistic distribution. It excels in providing interpretable models since its coefficients can be transformed into odds ratios for assessing the effect of each predictor on the outcome likelihood. However, it may struggle with complex, non-linear relationships and high multicollinearity among predictors, a condition which can compromise its predictive capability.

Table 4: Performance Comparison of Logistic Regression Models

Model	Metric	Validation Sample	Test Sample
Benchmark	Precision	NaN	NaN
Benchmark	AUC	0.5	0.5
Benchmark	F-1	NaN	NaN
Benchmark	Sensitivity (Recall)	0	0
Benchmark	Specificity	1	1
Best Model (L1 Penalty)	Precision	0.6667	NaN
Best Model (L1 Penalty)	AUC	0.7689	0.5
Best Model (L1 Penalty)	F-1	0.0272	NaN
Best Model (L1 Penalty)	Sensitivity (Recall)	0.0139	0
Best Model (L1 Penalty)	Specificity	0.9999	NaN
Best Model (L2 Penalty)	Precision	NaN	NaN
Best Model (L2 Penalty)	AUC	0.7194	0.5
Best Model (L2 Penalty)	F-1	NaN	NaN
Best Model (L2 Penalty)	Sensitivity (Recall)	0	0
Best Model (L2 Penalty)	Specificity	1	1

In this anomaly detection task, despite its robust theoretical underpinnings, the logistic regression models did not perform well. As metrics illustrated in Table 4, the best regression model with L1 penalty delivered a sensitivity of 0.0139, a specificity of 0.9999 and an AUC score of 0.7689 on the validation set. The model's sensitivity is notably low, implying a poor ability to correctly identify positive cases.

Random Forest

Random Forest is a non-parametric ensemble learning method. Unlike Logistic Regression, it does not make any assumptions about the data distribution or the relationship form between predictors and outcome. It is particularly capable of handling high-dimensional data and capturing complex, non-linear relationships, and interactions among features. Random Forest has a strong resistance to overfitting and offers an effective method for dimensionality reduction. However, interpretability can be a challenge as it does not provide a straightforward measure of the effect of each predictor on the outcome.

Viewing performance metrics in Table 5, Random Forest shows a relative robust performance. The best model with cross-validation achieved a sensitivity of 0.4861, a specificity of 1 and an AUC score of 0.7368 on the validation set. On the testing set, the sensitivity and AUC score slightly increased to 0.5245 and 0.7622 respectively. Test Sample comparison results from Table 4 and Table 5 indicates Random Forest model generalizes better on unseen data.

Table 5: Performance Comparison of Random Forest Models

Model	Metric	Validation Sample	Test Sample
Benchmark	Precision	1	1
Benchmark	AUC	0.7431	0.7622
Benchmark	F-1	0.6542	0.6881
Benchmark	Sensitivity (Recall)	0.4861	0.5245
Benchmark	Specificity	1	1
Best Model (Cross-validation)	Precision	1	1
Best Model (Cross-validation)	AUC	0.7368	0.7622
Best Model (Cross-validation)	F-1	0.6542	0.6881
Best Model (Cross-validation)	Sensitivity (Recall)	0.4861	0.5245
Best Model (Cross-validation)	Specificity	1	1

<u>XGBoost</u>

Extreme Gradient Boosting (XGBoost) is another ensemble learning method. XGBoost is robust and efficient, capable of handling both numerical and categorical data, missing values, and preventing overfitting. However, similar to Random Forest, one of the major challenges with XGBoost is its lack of interpretability compared to Logistic Regression.

Performance Metrics in Table 6 proves XGBoost to be a powerful model like Random Forest. The best model with cross-validation showed a sensitivity of 0.4861, a specificity of 1, and an AUC score of 0.7554 on the validation set. Performance on the testing set improved, with the sensitivity and AUC score reaching 0.5245 and 0.7622 respectively. Comparing Tables 4 and Table 6, XGBoost exhibits uniform performance across validation and test samples, while Logistic Regression with L1 and L2 penalty does not display this consistency. This comparison further highlights the stronger generalization capability of XGBoost as compared to the parametric approach of Logistic Regression.

Table 6: Performance Comparison of XGBoost Models

Model	Metric	Validation Sample	Test Sample
Benchmark	Precision	1	1
Benchmark	AUC	0.7431	0.7622
Benchmark	F-1	0.6542	0.6881
Benchmark	Sensitivity (Recall)	0.4861	0.5243
Benchmark	Specificity	1	1
Best Model (Cross-validation)	Precision	1	1
Best Model (Cross-validation)	AUC	0.7554	0.7622
Best Model (Cross-validation)	F-1	0.6542	0.6881
Best Model (Cross-validation)	Sensitivity (Recall)	0.4861	0.5245
Best Model (Cross-validation)	Specificity	1	1

k-Nearest Neighbors (kNN)

k-Nearest Neighbors (kNN) is a type of instance-based learning algorithm. kNN is simple and makes no assumptions about the data, making it potentially useful where the data do not meet the assumptions of other methods. However, it can be computationally intensive during prediction and its performance can be significantly influenced by the choice of 'k' and the distance metric used.

Table 7: Performance Comparison of KNN Models

Model	Metric	Validation Sample	Test Sample
Benchmark (k = 5)	Precision	0.9589	0.0635
Benchmark (k = 5)	AUC	0.7429	0.7396
Benchmark (k = 5)	F-1	0.6452	0.1142
Benchmark (k = 5)	Sensitivity (Recall)	0.4861	0.5664
Benchmark (k = 5)	Specificity	0.9998	0.9128
Best Model (k = 1)	Precision	0.5145	0.5172
Best Model (k = 1)	AUC	0.7441	0.7597
Best Model (k = 1)	F-1	0.5035	0.5208
Best Model (k = 1)	Sensitivity (Recall)	0.4931	0.5245
Best Model (k = 1)	Specificity	0.9951	0.9949

From Table 7, the best kNN model with cross-validation presented a sensitivity of 0.4931, a specificity of 0.9951 and an AUC Score of 0.7441 on the validation set. Performance increased on the testing set, with the sensitivity, specificity, and AUC Score reaching 0.5245, 0.9949, and 0.7597 respectively, indicating a satisfactory generalization performance. A noteworthy observation for the benchmark model in Table 7 is the significant drop in precision on the test sample. This implies the sensitivity of the kNN model to the chosen 'k' and further stresses the importance of choosing appropriate performance metric for evaluating machine learning models.

To sum up, Random Forest, XGBoost and kNN demonstrated better generalization ability compared to Logistic Regression. Despite their lack of interpret ability compared to Logistic Regression, non-parametric methods presented more robust predictive performances. Both Random Forest and XGBoost show improved performance from the validation set to the test set, which suggests better adaptability to new, unseen data, a critical aspect of machine learning models.

4. Future Works

Raman (2018) describes the calibration of probability predictions as a process that can be achieved by fitting a sigmoid or isotonic regression model to the predictions of the base model. Given the findings suggesting that the calibration curve of the Random Forest model closely resembles a sigmoid function, this approach could potentially offer considerable benefits. This method can adjust the output probabilities of the Random Forest model, minimizing the discrepancy between predicted probabilities and observed frequencies.

However, it's vital to note that calibration isn't a panacea. While it serves as a potent tool for refining the reliability of a model's probability predictions, it doesn't inherently enhance the model's classification accuracy. Therefore, this process should be seen as a supplementary refinement procedure, not a primary strategy for improving model performance.

High cardinality often hinders the effectiveness of machine learning algorithms due to the sparsity and increased dimensionality it introduces. In the current models, the issue was addressed by converting the country variable into a frequency count. For future improvements, a different approach might involve the application of one-hot encoding to these high cardinality variables. This would create a binary vector representation for value each variable holds, potentially preserving more information than the frequency count method. While this approach would increase the dimensionality of the data, it could also improve the model's capacity to identify and learn new patterns.

However, the increased dimensionality following one-hot encoding can introduce the risk of overfitting and multicollinearity. Therefore, a complementary solution would be to apply the double lasso technique for variable selection post-encoding. This method is adept at handling high-dimensional data, efficiently selecting relevant features and discarding irrelevant ones, ultimately enhancing the model's performance and interpretability.

Collecting more data can invariably improve the accuracy and generalizability of the models. Larger datasets offer a more comprehensive view of the population and reduce overfitting. Beyond data collection, innovative feature engineering could bring additional value. This process entails creating new features from existing ones, often enabling the models to capture complex patterns and improve prediction performance.

5. Conclusion

The capstone project successfully developed several machine learning models to tackle the critical issue of fraud detection on e-commerce platforms. The best performing model is the Random Forest, which exhibited robust performance on the test set with a sensitivity of 0.5245, specificity of 1, and an AUC score of 0.7622. These metrics reflect the model's excellent capacity to balance both the detection of fraudulent activities and the minimization of false alarms.

Various popular class balancing techniques are examined, but these methods appeares to degrade the model's predictability and tends to overestimate fraudulent cases, underscoring the complexities inherent in handling imbalanced data.

Implications of this project are substantial for e-commerce platforms. Effective fraud detection mechanisms, like the ones developed here, are pivotal for maintaining user trust and operational integrity. By accurately identifying fraudulent transactions, platforms can protect both their users and themselves from financial losses and reputational damage. Moreover, machine learning models explored in this study provide a basis for understanding the characteristics of fraudulent activities, facilitating proactive measures to prevent fraud.

References

Pawar, A. (2018, December 4). Predicting Real-Time Availability of 200+ Million Grocery Items in US & Canada Stores. Tech at Instacart. Retrieved from https://tech.instacart.com/predicting-real-time-availability-of-200-million-grocery-items-in-us-canada-stores-61f43a16eafe

Raman, K. (2018, November 8). Model calibration with scikit-learn. Medium. Retrieved from https://medium.com/optima-blog/model-calibration-4d710a76c54

Rutjes, A. W., Reitsma, J. B., Coomarasamy, A., Khan, K. S., & Bossuyt, P. M. (2007). Evaluation of diagnostic tests when there is no gold standard. A review of methods. Health Technology Assessment, 11(50), iii, ix-51. Retrieved from https://www.sciencedirect.com/science/article/pii/S0895435619303579

van den Goorbergh, R., van Smeden, M., Timmerman, D., & Van Calster, B. (2022). The harm of class imbalance corrections for risk prediction models: illustration and simulation using logistic regression. Journal of the American Medical Informatics Association, 29(9), 1525–1534. https://doi.org/10.1093/jamia/ocac093