Classification-Based Predictive Analysis of Loan Default

Esteban Gomez

01/17/2024

Table of Contents

Abstract

Loan defaults significantly impact the economy and lead to notable losses for banks, setting a precedent for the current lending guidelines to mitigate the damage they cause. Despite the rules put in place to navigate financial managers in their decisions to approve or deny borrowers, the number of details required to decide on just one customer's profile can be taxing for people alone. The project aims to develop a predictive model that classifies borrower applications as possible defaults. By analyzing borrower data such as income and debt-to-income ratios to detect early signs of potential defaults, our model will aid financial managers in making better-informed decisions.

The nature of predicting loan defaults is a difficult task since the failure of borrowers to honor their commitment is a rare occasion, leaving only a few cases for the model to study in comparison to the vast amounts of non-defaulting loans. Hence, although the long-term goal is to automate decisions, the current state of classification models in loan default are more effective as supplemental tools to help prevent losses and make informed decisions.

After preparing our data for analysis, we explore the data details and search for potential patterns among features that will assist us in predicting the class of interest. We test various models superficially to understand which models will work best, then utilize resampling techniques and hyperparameter optimization to maximize model performance.

As mentioned before, the nature of the data's class imbalance led most of the models tested to tend to predict that all loans are non-defaulted. This imbalance limits their ability to classify defaults despite the methods utilized to optimize performance. A great approach to using the final model is to raise red flags for financial managers to delve deeper into an applicant's profile for further analysis and help decide to approve or deny.

Introduction

Default poses a significant risk in the lending industry, emphasizing effective risk management's role in identifying potential losses for lenders. Consumers are also seriously affected by a reduction in credit score, hindering their ability to borrow in the future by being limited to higher interest rates or being denied altogether for an application of credit. Unsecured loans are an even greater risk since an asset does not back them, although lenders still hold a legal claim. To address the issue, we hope to provide lenders with a more robust approach to risk management and improve outcomes for both lenders and borrowers. LendingClub, an online lending platform that links individuals seeking personal loans with investors, has made its data available for unsecured loans, and we will utilize the data to achieve our task.

The project's primary objective is to examine the use of classification algorithms in predicting unsecured loan defaults. We conclude that classification models are best utilized as accessories to the application process rather than as a means to an end. It is important to note that predicting loan defaults is a complex task for machine learning models, and a combination of indicators and other financial analyses determines the decision to acquire, hold, or dispose of potential loans. The exploratory analysis, modeling, and evaluation are conducted entirely with Python libraries such as NumPy, Pandas, matplotlib, seaborn, and scikit-learn to facilitate the investigation. All scripting is documented in Jupyter Notebooks and is available for further inspection to integrate with any current business workflow.

Exploratory Data Analysis

LendingClub's data set contains over 38,000 rows of data. Some of the features in the data are pieces of information that the lender gathers after the loan's inception. In the end, we utilized 20 columns to build the model compared to the initial 36 columns. Of the features kept, only one column had about 3% of data missing, and the subsequent column with the highest proportion was missing less than 0.2%; therefore, we swiftly removed any missing data without cause for concern. It is worth noting that defaulted loans accounted for only 15% of the data. To achieve reasonable model performance, we must overcome the obstacle of class imbalance.

After collecting, organizing, and ensuring it was well defined, we explored the data to uncover relationships among features and their relation to the class we wanted to predict. We only managed to unravel weak relationships between the features and the default status of the loans. These weak correlations make it difficult for the model to classify decisively what class a loan belongs to. Figure 1 demonstrates the most robust distribution differences among numerical features and the loan class. Note that there are three boxplots in each row, where each row represents a numeric feature. Each column showcases a different way of viewing the boxplot by setting a separate limit for the outliers, with the rightmost column having no outliers. Utilizing the various perspectives is especially useful for highly skewed features. The blue boxplots are non-defaulted loans, and the orange is defaulted loans.

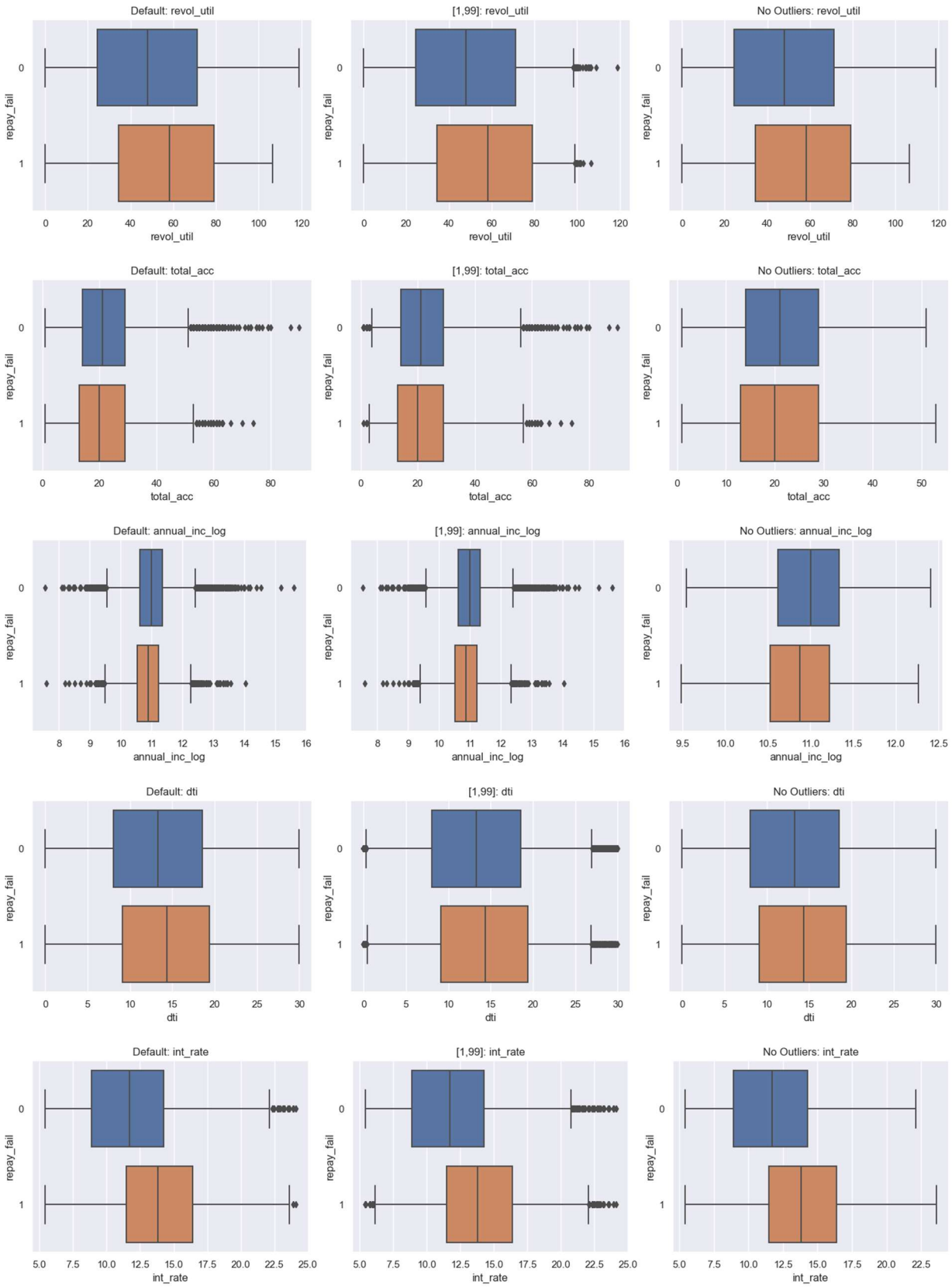

Figure 1. Numerical Distributions of defaulted and non-defaulted loans.

The numeric features in Figure 1 display the most apparent differences in distribution among the defaulted and non-defaulted loans, and even then, the differences are not significant. The rest of the numeric features have distributions that are even closer together. The categorical features follow the same pattern.

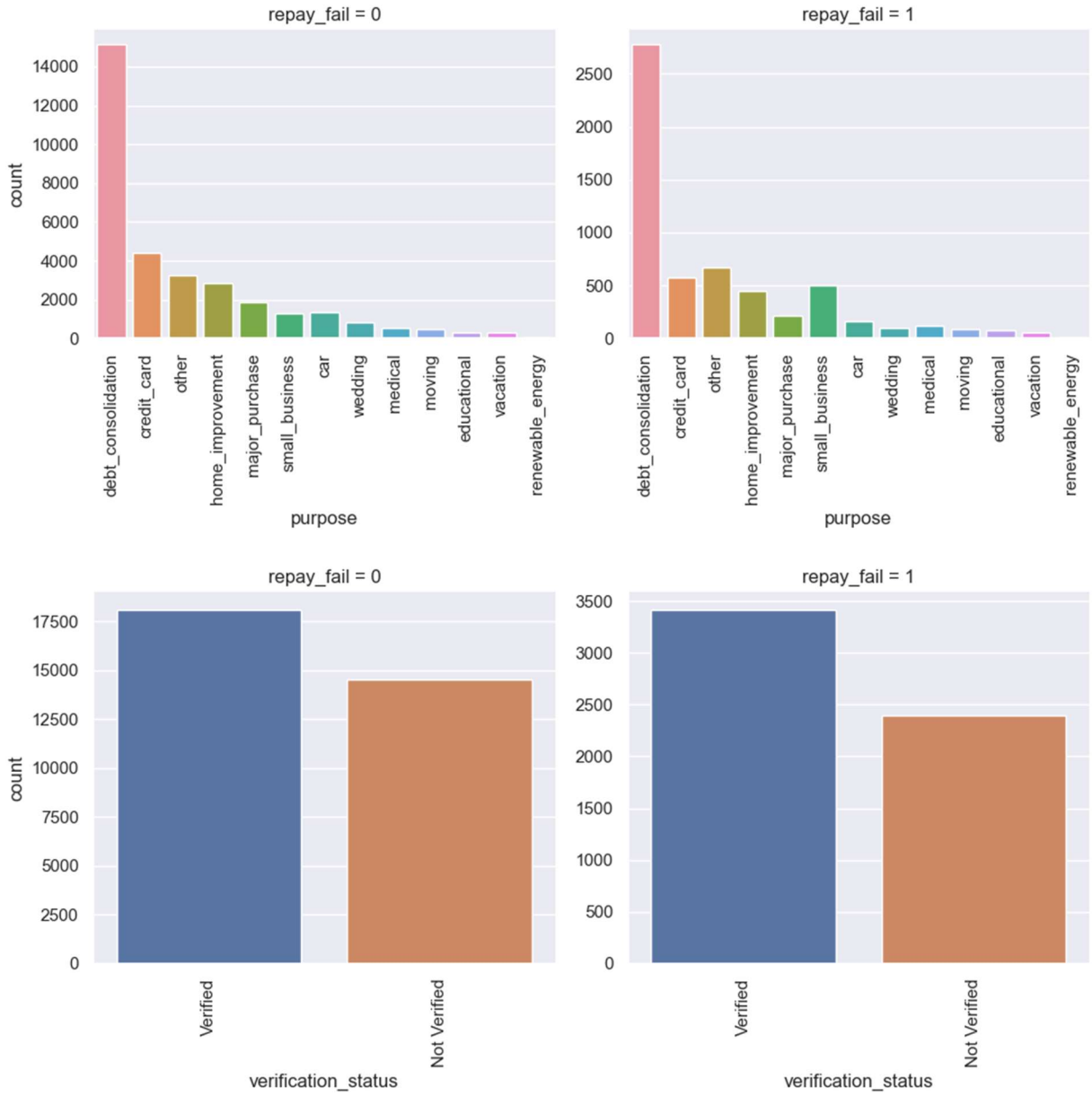

Figure 2. Categorical distributions of defaulted and non-defaulted loan

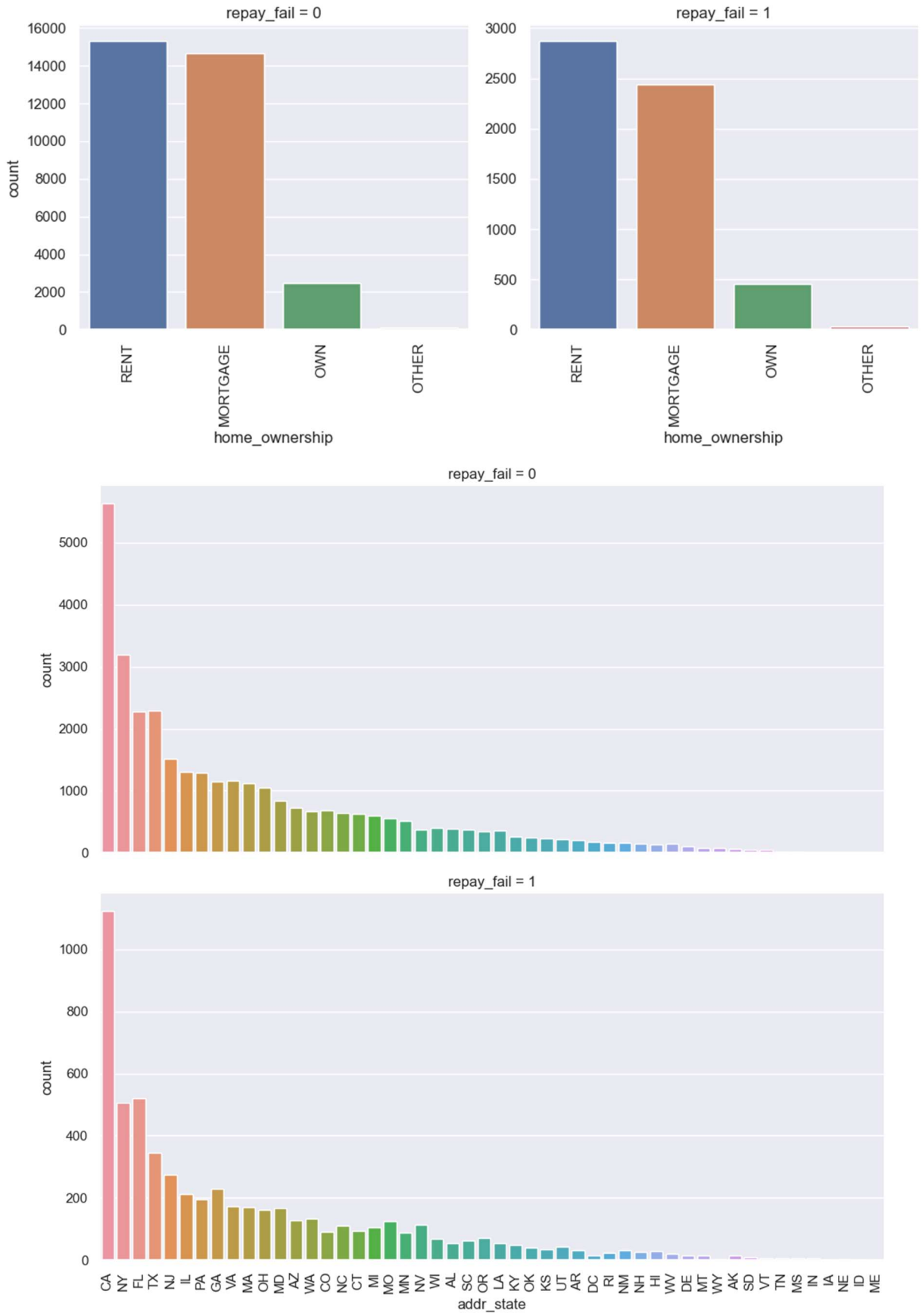

Figure 3. Categorical distributions of defaulted and non-defaulted loan

Figures 2 and 3 display the distributions among the defaulted and non-defaulted loans for categorical features. Like the numeric features, the differences between loan classes could be more apparent for the model to decipher clear patterns.

Modeling

Data Pre-Processing

To create the first instance of the model, we prepare the data by creating dummy variables for the categorical columns and splitting the entire set into training and test sets. One of our features contains the address state of the borrower, and since there are 50 states, that would create an undesirably sparse dataset for training the model when encoding the categorical features. We excluded the address state feature to test the model's predictive power without it.

The first round of training is to identify which models will be the most useful for our case. We started by constructing a superficial overview of results from various models with LazyClassifier. We aim to maximize the F1 score because of the class imbalance. The models that provided the highest F1 score and accuracy were XGBClassifier (XG Boost Classifier), BernoulliNB (Bernoulli Naïve Bayes), and LinearDiscriminantAnalysis (Linear Discriminant Analysis). The times it took the three models to provide results were also remarkably low, making for efficient options.

Model	Accuracy	F1 Score
XGBClassifier	84.20%	78.99%
BernoulliNB	84.10%	78.93%
LinearDiscriminantAnalysis	84.88%	78.75%

Table 1. LazyClassifier accuracy and F1 score results are sorted by F1 score and then accuracy.

The accuracy of all the models is about 85% because 15% of the loans are defaulted; therefore, the models propose that all the loans are non-defaulted to achieve a high yet deceiving accuracy. The deceitful score is a result of class imbalance.

Initial Fits

We delve deeper into the three models to test which will work best. We manually ran the training data through the models to compare the results.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	85.38%	98.21%	91.35%	67.73%
	Defaulted	35.68%	5.58%	9.65%	
BernoulliNB	Non-defaulted	84.88%%	100%	91.82%	60.13%
	Defaulted	0%	0%	0%	
LinearDiscriminantAnalysis	Non-defaulted	85.20%	99.47%	91.78%	70.91%
	Defaulted	50%	2.96%	5.60%	

Table 2. Model results from the initial fit of training data.

Table 2 shows the three models excellently performing on the non-defaulted class with high recall and precision while performing poorly on the defaulted class. The BernoulliNB model failed to identify any defaulted instances correctly. Among the three models, the LinearDiscriminantAnalysis demonstrates the highest overall ROC AUC, positioning itself as the basis for our final model. Most importantly, however, it is evident that the models struggle to identify any defaulted loans, leading us to tackle the class imbalance.

Oversampling

We use the oversampling technique to overcome the models' tendency to assume that all loans are non-defaulted. A subset of the data is created by resampling the minority class (defaulted loans) with replacement until there is an equal number of loans in the subgroup to the number of non-defaulted loans. Then, we feed the models the new training set with equal amounts of both classes to yield more accurate results.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	97.22%	82.11%	89.03%	92.11%
	Defaulted	46.36%	86.78%	60.43%	
BernoulliNB	Non-defaulted	88.47%	56.38%	68.87%	60.42%
	Defaulted	19.35%	58.75%	29.11%	
LinearDiscriminantAnalysis	Non-defaulted	91.14%	65.21%	76.02%	71.03%
	Defaulted	24.79%	64.39%	35.80%	

Table 3. Model results after fitting the oversampled data.

Table 3 has shifted the focus from LinearDiscriminantAnalysis to XGBClassifier as the basis for the final model, as it has performed remarkably well given the oversampled data. It is worth noting that oversampling the data has impacted the models' ability to recall defaulted loans. XGBClassifier, in particular, has now become the best at correctly identifying defaulted loans when it comes across one, although it was still only able to identify less than half. The next step is to tune the hyperparameters to improve the models' results.

Cross-Validation

The models have used the default hyperparameter settings to classify each loan's class. We now turn to optimizing the hyperparameters for each model and comparing the results again. The final results will determine which model to focus on optimizing.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	96.50%	79.42%	87.13%	90.17%
	Defaulted	42.04%	83.82%	55.99%	
BernoulliNB	Non-defaulted	88.45%	56.46%	68.93%	60.39%
	Defaulted	19.34%	58.61%	29.08%	
LinearDiscriminantAnalysis	Non-defaulted	91.21%	65.28%	76.10%	71.03%
	Defaulted	24.91%	64.67%	35.97%	

Table 4. Model results after cross-validation and fitting the oversampled data.

Surprisingly, cross-validation yielded slightly worse results than sticking with the default hyperparameters. However, the results still hold that the XGBClassifier was the best model. The next step is fine-tuning XGBClassifier's hyperparameters to achieve the best possible performance.

Optimizing XGBClassifier

Despite the undesirable results from cross-validation, we aim to optimize the model by adding more hyperparameter values to the parameter grid. The trade-off is that searching through a larger parameter grid is more computationally expensive. However, we can focus on tuning the hyperparameters that make the most significant difference in our target variable that we want to optimize, the F1 score. Optimizing the F1 score will provide the hyperparameter values to construct the best predictive model given the inherent class imbalance. Therefore, we test how the F1 score changes while changing only one hyperparameter and keeping the rest the same.

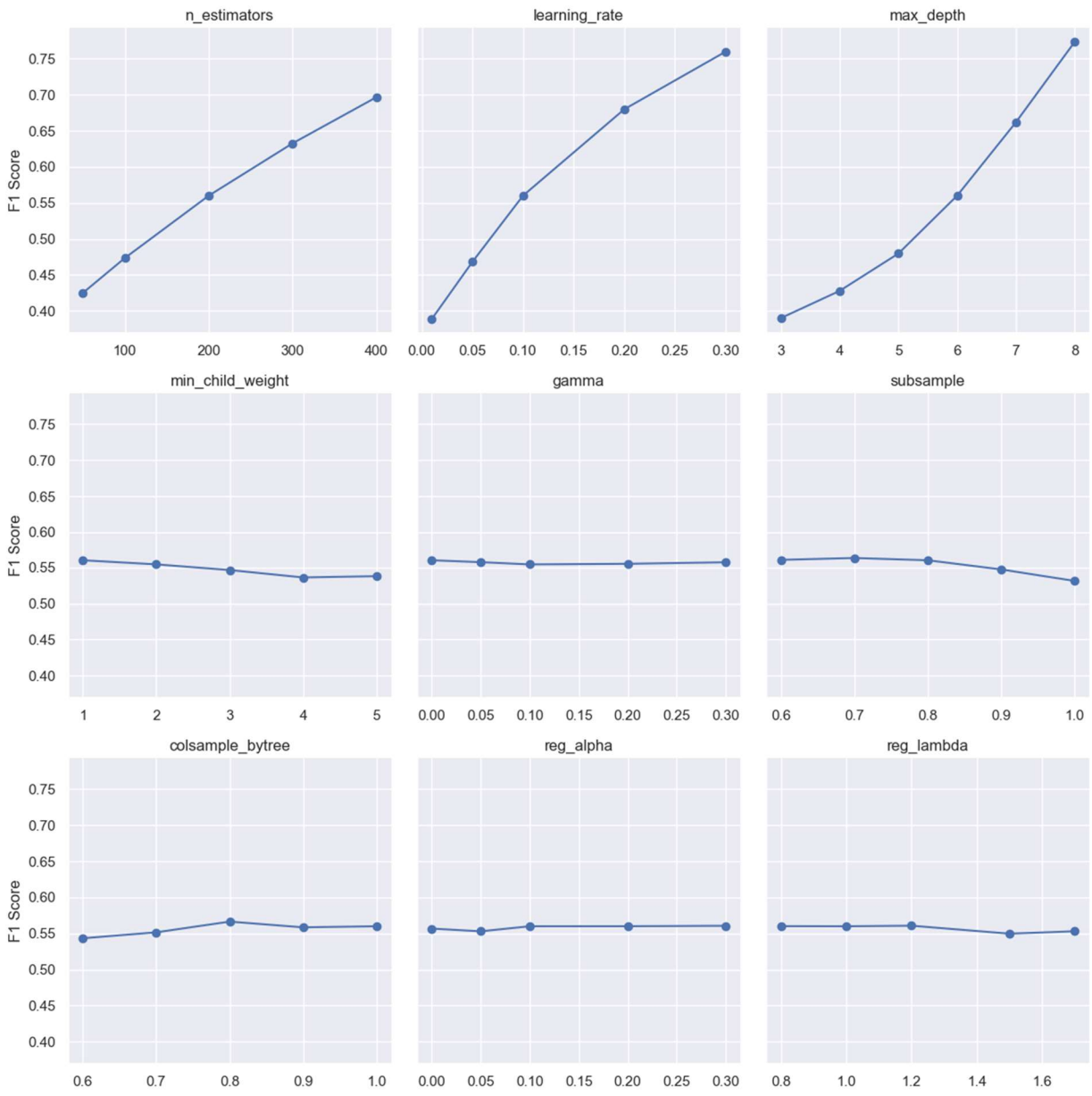

Figure 4. F1 score at different hyperparameter values.

Figure 4 shows that learning_rate, max_depth, and n_estimators are the most significant hyperparameters affecting the F1 score. Therefore, we optimized the XGBClassifier by using the rest of the optimal hyperparameters found previously and then performing a cross-validation with the three most significant hyperparameters to reach the next evolution of our model.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	99.85%	97.69%	98.76%	99.62%
	Defaulted	88.45%	99.17%	93.51%	

Table 5. Results after performing cross-validation with the three significant hyperparameters.

The results indicate that the model is likely overfitting the data since the scores are remarkably high. Typically, poor results on the test data indicate overfitting; however, the scores are exceptionally high in our case. The dataset we have used to construct the model is biased in testing its predictive ability.

Test on Unseen Data

We introduce an unseen dataset and make predictions to determine whether the model is overfitting the data. We obtained the unseen data from Coursera's Loan Default Prediction Challenge, which contains over 250,000 rows. The features on the unseen dataset do not precisely match the original dataset we used to construct the model. We converted the unseen dataset's relevant features to check with the features in the original dataset by renaming the columns, matching the datatypes, and formatting the columns to match the format of the original dataset. Any features not in the unseen dataset were imputed either through the mode or by calculating using the known features.

We could not impute a couple of features in the unseen dataset. Therefore, we retrained the model without the non-imputable features and tested its predictive ability by running it through the process above.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	99.85%	97.47%	98.65%	99.47%
	Defaulted	87.48%	99.17%	92.96%	

Table 6. Results after the removal of non-imputable features.

Removing the non-imputable features did not significantly affect the model's predictive ability on the test set. Therefore, we now move forward with testing on the unseen dataset.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	88.45%	95.25%	91.72%	52.03%
	Defaulted	12.90%	5.36%	7.57%	

Table 7. Results from making predictions on the unseen data.

Our suspicion was correct, as the model predicts that most loans in the new dataset will not default. However, we can utilize the latest dataset we obtained to create a more reliable version of the model.

Re-train with Unseen Data

We integrate some unseen data into training the model to construct a more robust model. To avoid class imbalance influencing the model's predictions, we undersample the unseen data by resampling the non-defaulted loans until there is an equal amount of non-defaulted loans as defaulted loans. Then, we split the data in half, added half to the original dataset, and then saved the other half for testing. That way, we can solely test the newest model on the unseen data without the scores being affected by the original dataset. The consolidated data, which now contains the actual data plus half of the undersampled unseen data, is split into a training set and a test set. We trained the model with the new training set, and then we tested the model with the latest test set. We run the model through the former cross-validation process to optimize its predictive ability.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	84.84%	79.00%	81.82%	91.39%
	Defaulted	80.18%	85.75%	82.87%	

Table 8. Results from the model trained with consolidated data.

Upon proceeding with the final iteration of training, the results indicate a more balanced model, with scores low enough to suggest that it is not overfitting all the data. However, we tested the model on the original test data and the unseen test data from the undersampled subset to know the truth behind the results.

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	99.57%	96.07%	97.79%	99.17%
	Defaulted	81.59%	97.66%	88.90%	

Table 9. Results from the model trained with consolidated data on the original data's test set.

Table 9 suggests that the model is still overfitting to the original dataset. Therefore, it continues to perform worse on the unseen data, and the scores from Table 8 average the overall model performance. To simulate how the model will perform in a real-world application, we feed it samples of 100 loans at a time and record its results. This process is a simulation of 100 loan applications being submitted at a time, and the model predicts which loans of those 100 will default.

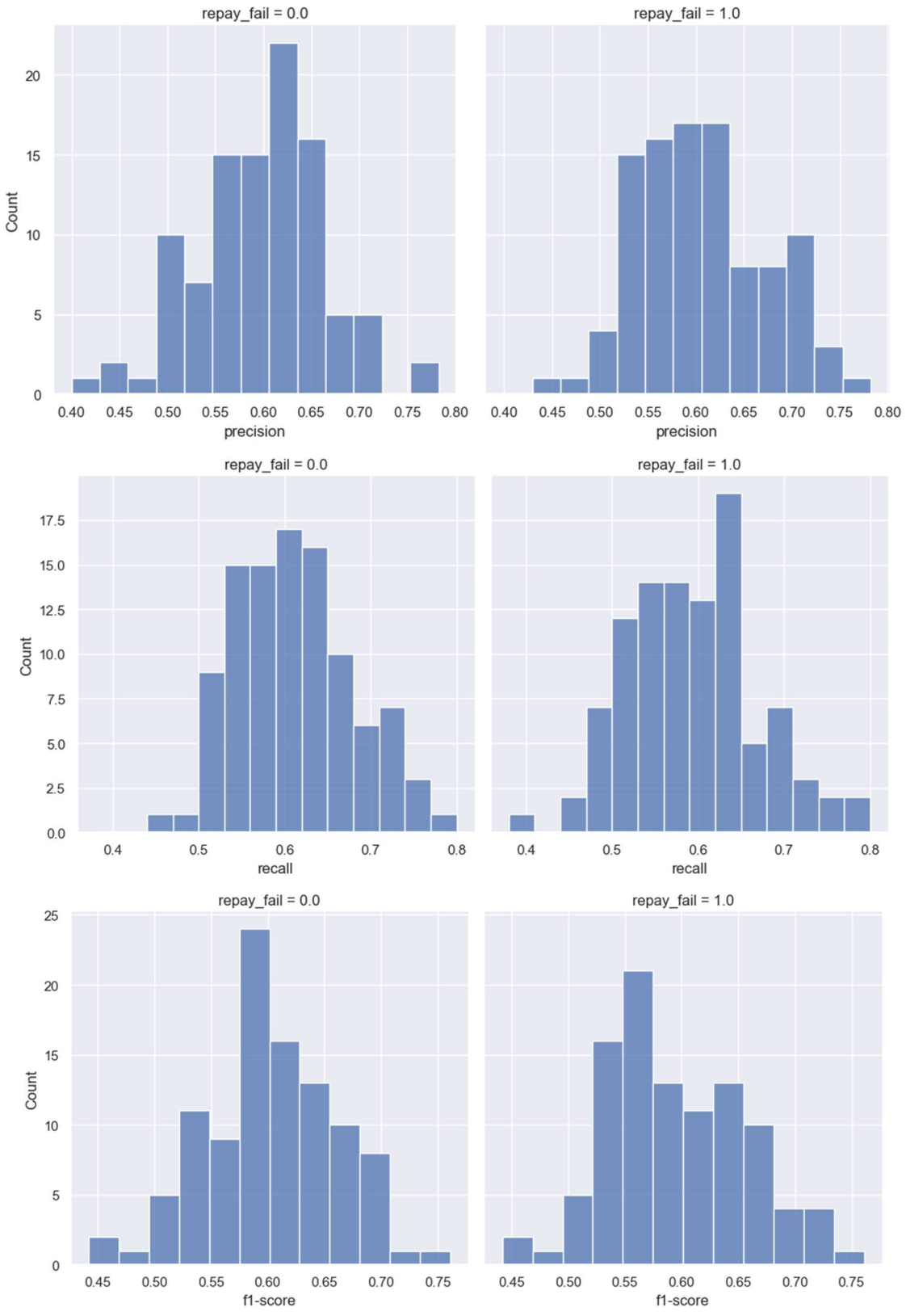

Figure 5. Distribution of results on unseen test data by samples of 100 loans.

We plotted the distributions of the results to visualize the model's performance on the unseen data. The model averages around a 60% score for the three scores of interest, precision, recall, and f1-score, regardless of the loan being defaulted or non-defaulted (Note: repay_fail = 1 means the loan is defaulted).

Model	Class	Precision	Recall	F1 Score	ROC AUC
XGBClassifier	Non-defaulted	60.18%	61.37%	60.77%	64.89%
	Defaulted	60.59%	59.40%	59.99%	

Table 10. Results from the model trained with consolidated data on the unseen data's test set.

As we noticed in the distributions, the model's performance averages around 60% for all the scores. Therefore, the model is still overfitting to the original data despite training with a subset of the unseen data. However, the unseen data performance improved tremendously, as the model can now decipher between more defaulted and non-defaulted loans in the unseen data.

Discussion

The XGBoost Classifier model has reached its peak performance by introducing an unseen dataset. Despite fine-tuning the hyperparameters and enhancing the training data, the model's performance has plateaued, showing no significant improvement. The plateau suggests that the model has grown increasingly complex, and we're now experiencing diminishing marginal returns regarding its predictive accuracy. While we could continue to refine the parameters and further augment the training data, the expected gains would be minimal compared to the effort needed for further improvement.

In retrospect, different techniques could be employed when constructing the model to improve results. One such method involves applying alternative sampling techniques like SMOTE or ADASYN instead of a simple oversampling or undersampling. Another powerful strategy is using ensemble methods, combining multiple models instead of relying on one model alone to do the trick. In addition, we can use L1 or L2 regularization to control overfitting by penalizing complexity in the model. Combining these methods can significantly increase the model's ability to accurately predict and forecast loan defaults.

Conclusion

In light of the importance of risk management in the lending sector, we created a model that can be utilized as an accessory when analyzing applicants' financial profiles. In combination with class imbalance, weak correlations between features and default hinder the model's predictive ability. Despite the obstacles, we improved the model's predictive ability through extensive hyperparameter optimization and resampling techniques. Unfortunately, our model overfitted the training data; however, we developed a final, more robust model by utilizing an unseen dataset. This improved model can now serve as a valuable aid to financial managers, helping to raise red flags on potential loan defaults.