Loan Risk Management Through Machine Learning Techniques

University of Michigan

School of Information

Authors

HoJoon Kim, Master of Science in Information

Sarah Fringer, Master of Science in Data Science

Sai Samarit, Bachelor of Science in Information

Introduction

The loan service providers historically depended on credit scores to judge the risk of loan customers. Yet there are still some loans that get charged off as customers fail to repay the loans. On the brightside, a lot of information, both quantitative and qualitative, are collected when a person applies for a loan. This project aims to utilize this information to create a machine learning model to mitigate the risk of providing loans to customers from the perspective of the loan service providers. Information available at the point of loan application will be utilized to create a binary classification model that predicts whether customers will fully pay back their loan or fail to do so and get their loans charged off.

Keyword: Machine Learning, Loan Industry, Imbalance classification, Cost Sensitive Learning

Methods

Dataset

The data for this project was provided from the Lending Club (LC). The Lending Club is currently a technology built company, who focuses on connecting borrows and investors through an online marketplace [3]. Clients can qualify for personal loans, auto loan refinancing, small business loans, etc, while the Lending Club makes credit affordable and investing rewarding. The Lending Clubs offers publicly available data, regarding information and details on mostly all of the loans issued by the company [2]. The data provided online from the Lending Club encompasses Q1-Q3 data of 2020 and can be dated back to 2007. We will be using the entirety of 2019 data from the website to train and validate our machine learning model. Basic specification of the data can be found below.

Total Rows : 518,115 Rows
Features : 150 Features

The information given for each loan consists of all the details of the loan at the time of their issuance, along with more information regarding the latest status of the loan, how much principal has been paid off, the interest accumulated, etc. To be more specific, there are around 150 associated features of data given for each loan. Therefore, a large portion of this project dealt with data engineering, preprocessing the data, and feature selection.

Preprocessing The Data

To begin, as mentioned before the goal of this project was to find a model that best predicts whether a loan has been Fully Paid or Charged Off. Therefore, the first task of the data cleaning process was to look at our target variable. In this case, our target variable was the *Loan Status* feature. Initially, this feature contained 9 different categories. The categories were as follows: Current, Fully Paid, Charged Off, Late (31-120 days), In Grace Period, Late (16-30 days), Does not meet the credit policy (Status either: Fully Paid/Charged Off), and Default. For the purpose of this project, we decided to drop all rows besides those with loans that were either Fully Paid or Charged Off. We discovered that the distribution between Fully Paid loans and Charged Off loans was, then, very imbalanced which would later cause a problem with our model. Our solution to this problem is further discussed in a separate section below.

As stated before, the data contained around 150 features per loan. Although after reading through the data dictionary provided by the Lending Club, we realized that not all of the features were necessarily useful for our purpose of the project. Hence, our next step was to drop features that were either unnecessary, useless, or nonessential for this project. The first thing we did to drop unwanted features was to disregard those that had more than 50% of its data missing. We discovered that around 40% of the features had more than half missing values. After these features were dropped, we decided to dismiss features that were nonexistent at the time of the loan application. We did this in regards to what our model was trying to predict. Again, we wanted to find a model that best predicts the status of a loan (Fully Paid/Charged Off). Hence, features that had no connection to whether a loan was granted were dropped. We discovered which features we wanted to drop based on research, background knowledge, and reading through and understanding each feature in the data dictionary. From here, we decided to drop about 70 features, which we found unbeneficial to our project.

Cleaning the Data

The next steps consisted of inspecting the remaining features individually, while looking at their respective distribution, data-type, correlations, and overall importance. This particular step of the data cleaning process took quite a bit of time as we conducted an EDA on each of the remaining variables and performed transformations or modifications if necessary. A few examples of these transformations were as follows: (1) Converting the *emp_length* feature from an object to a float (2) Applying a log-transformation on the *annual_inc* feature to account for a large range of variation (3) Creating a new variable called *fico_score* which was the mean value of the *fico_low* and *fico_high* features (4) Converting our target_variable, *loan_status,* into a binary indicator with 0=Fully Paid and 1=Charged Off. Again, these were just a few example modifications done on these features, but we performed similar tasks on the remaining, contending features.

After cleaning and exploring each of the potential necessary features, we decided to look at the correlation and relationship between each of them. We did this by conducting a heat map between each of the remaining features to see if any features were linearly dependent with one another, as well as looking at the correlation with the target variable. From here, we decided to remove features that had a high linear correlation with one another, such as *installment, total_acc, pub_rec_backruptcies.* We then had about 20 remaining features that we could include in our model, but we wanted to further narrow this down to about 10-15 features. To do this, we included feature importance techniques in our baseline models to determine which were the best features to include.

Oversampling minor class through SMOTE (Synthetic Minority Oversampling Technique)

As noted before, the loan dataset had a heavy imbalance of classes in our target variable. The majority of borrowers are found to pay back their loans, which leads to scarcity of 'charged off' instances. Classification algorithms are built upon the premise that there is an equal distribution of classes within the target variable [2]. Running these particular algorithms on an imbalanced dataset, like ours, leads to biases toward the majority class. Hence, leading toward the minority class being misrepresented in the final model [2]. For this project, we are particularly interested in the minority class which is "Charged Off" loans, which are loans that do not get fully paid by the borrowers. Through the use of sampling techniques, there are two different directions we can take to overcome this imbalance problem. One direction is to undersample the majority class, while the other direction is to oversample the minority class. For this project, we decided to oversample the minority class because we wanted to protect the amount of information that the majority class had. Instead of simply oversampling the same data points from the minority class, we decided to use Synthetic Minority Oversampling Technique described by Nitesh Chawla 2002 paper[1]. This technique chooses a random example from the minority class, then selects the k nearest neighbors for that specific point. Once the neighbor is randomly selected, a synthetic example is created at a randomly selected point between the two examples in the feature space[1]. This preprocessing technique resolved our imbalance issue, to some extent, where we could then measure the performance of different models more thoroughly. Additional measures will be discussed in the Analysis section.

Models

Multiple classifiers will be used at their default parameters to carry out the preliminary modeling on the data. Some models will serve the role as baseline models for comparison purposes, while other ensemble models will be further tuned depending on the initial performance, in which one of them will be used as the final model. The models that we will explore are Naive Bayes, Logistic Regression, Decision Tree, Random Forest, Ada Boost, and Gradient Boost classifiers.

Code Structure

There are three code files that each serve different purposes in the project

- EDA.ipynb : Exploratory Data Analysis
- DataCleaning.ipynb : Data Cleaning and Preprocessing
- ModelingEvaluation.ipynb : Modeling and Evaluation

Summary of Algorithms

Preprocessing
- Pandas
- SMOTE from Imblearn
- Train Test Split from Sci-kit Learn
- Standard Scaler from Sci-kit Learn

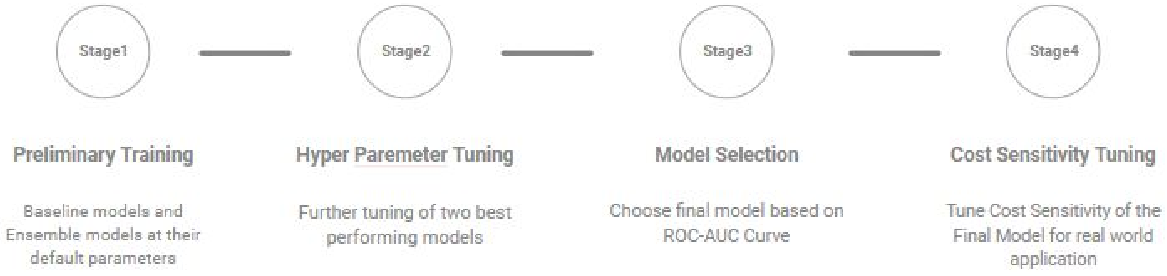
Modeling
- Naive Bayes, Logistic Regression, Decision Tree, Random Forest, Ada Boost, and Gradient Boost from Sci-kit Learn

Evaluation
- ROC Curves, Classification Report, Confusion Matrix from Sci-kit Learn

Evaluation and Analysis

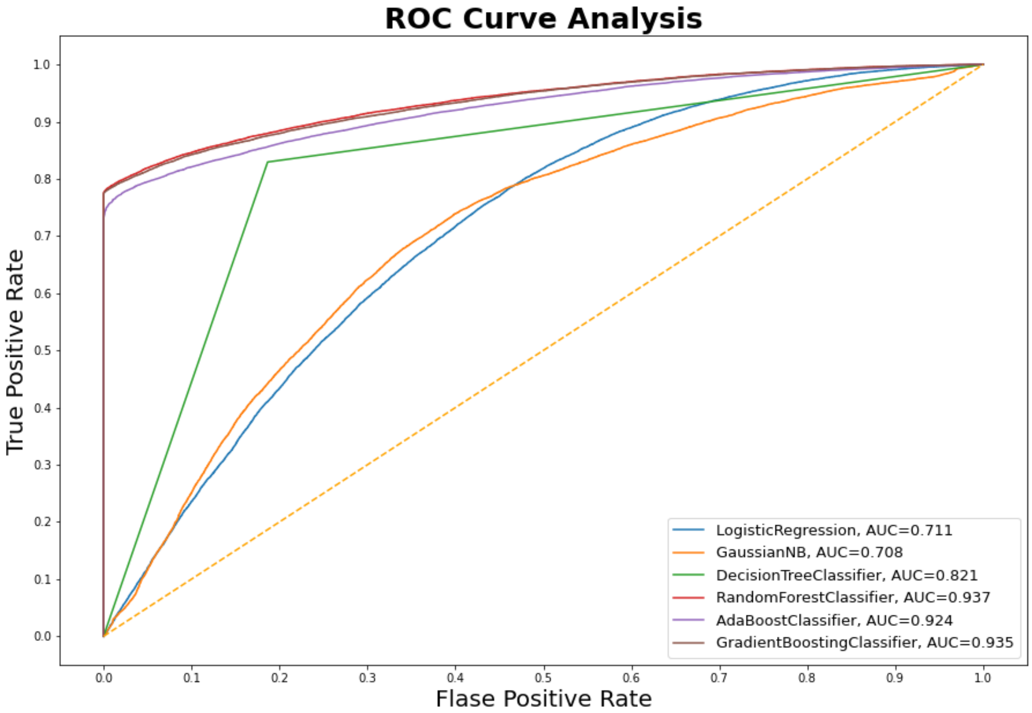
Study Design / Evaluation Methods

We decided to run our preprocessed data through several models at their base parameters, and depending on the performance based on Receiver Operating Characteristic - Area under the Curve score (ROC-AUC Score), we tuned the two best performing models to train our dataset.
We also used classification reports and confusion matrices for further evaluation and analysis. We, then, chose one single final model and tuned its cost sensitivity to make the model more appropriate for real world applications in the loan industry domain.


Stage1
Stage2
Stage3
Stage4
Preliminary Training
Baseline models and Ensemble models at their default parameters
Hyper Paremeter Tuning
Further tuning of two best performing models
Model Selection
Choose final model based on ROC-AUC Curve
Cost Sensitivity Tuning
Tune Cost Sensitivity of the Final Model for real world application

Results

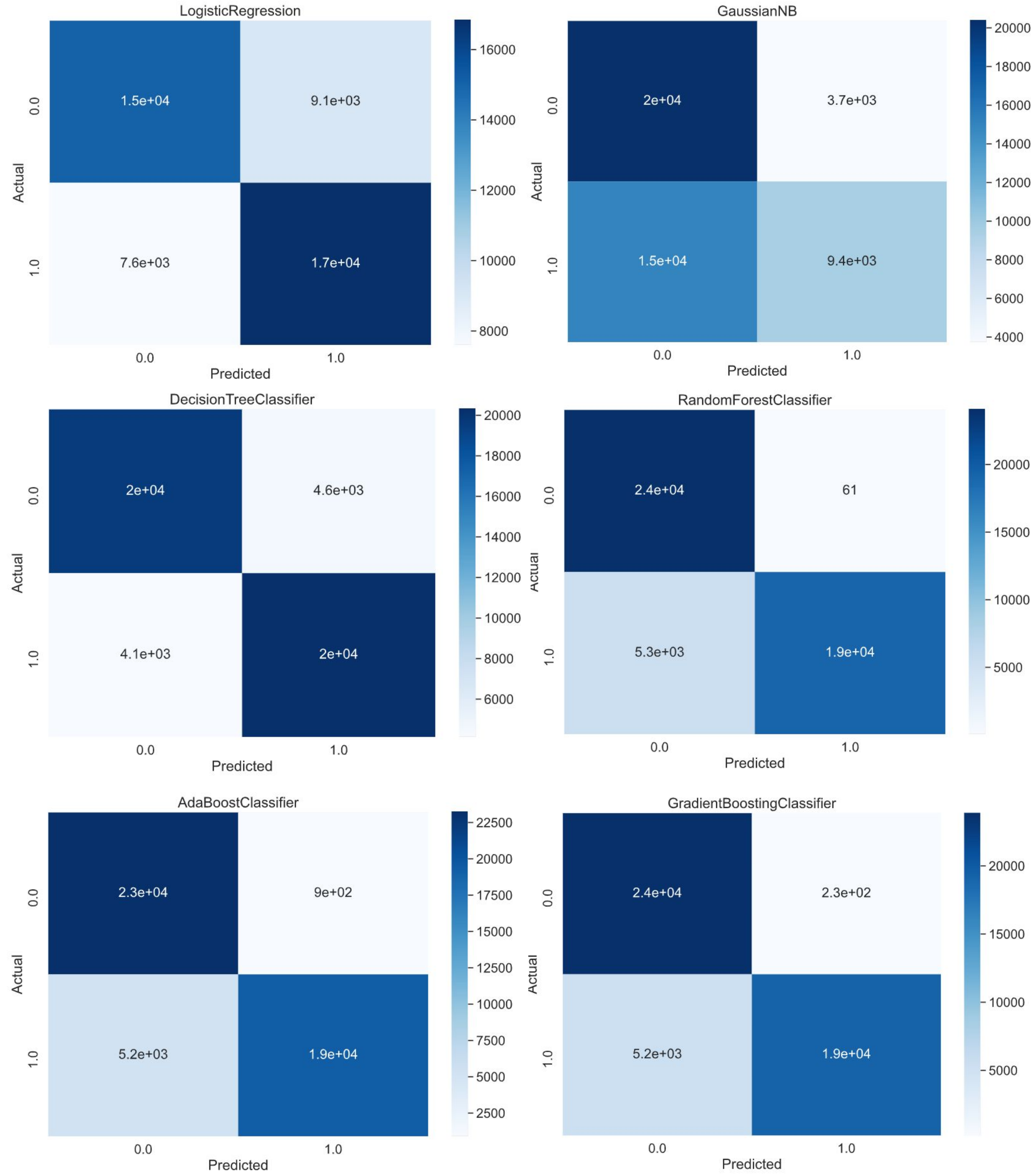
ROC Curve



Classification Report

Classifier	Class	Precision	Recall	F-1 Score
Logistic Regression	Fully Paid	0.664	0.624	0.643
	Charged Off	0.65	0.689	0.669
Gaussian NB	Fully Paid	0.575	0.845	0.685
	Charged Off	0.715	0.385	0.5
Decision Tree	Fully Paid	0.827	0.814	0.821
	Charged Off	0.82	0.832	0.826
Random Forest	Fully Paid	0.818	0.997	0.899
	Charged Off	0.996	0.781	0.876
AdaBoost	Fully Paid	0.817	0.962	0.884
	Charged Off	0.955	0.788	0.863
Gradient Boost	Fully Paid	0.82	0.99	0.897
	Charged Off	0.988	0.786	0.876

Confusion Matrix (0 = Fully Paid, 1 = Charged Off)



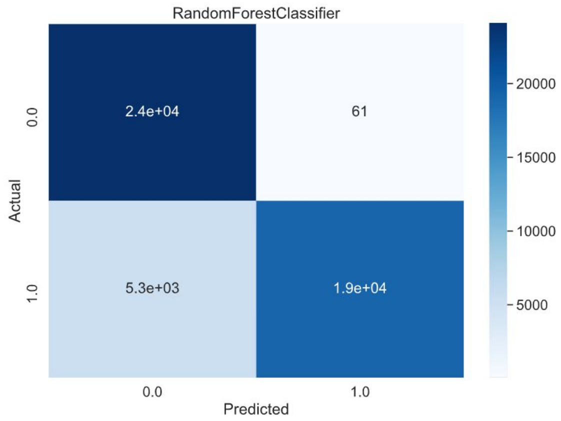
Final Model Classification Report with Cost Sensitivity Tuning

Classifier	Class	Precision	Recall	F-1 Score
Random Forest (Untuned)	Fully Paid	0.818	0.997	0.899
	Charged Off	0.996	0.781	0.876
Random Forest (Tuned)	Fully Paid	0.863	0.774	0.816
	Charged Off	0.798	0.879	0.836

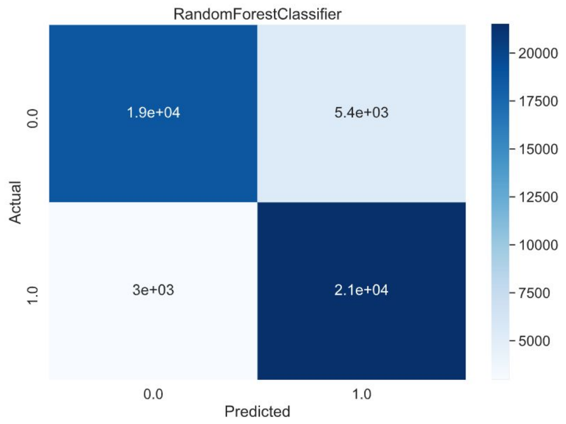
Final Model Confusion Matrix with Cost Sensitivity Tuning

(0 = Fully Paid, 1 = Charged Off)

Untuned



Tuned



Modeling Conclusion

As represented in the ROC-AUC graph, a random forest proved to be our best performing model with a ROC-AUC score of 0.937. We tuned its respective parameters using a grid search. This is a significant improvement in performance from our baseline model, which was a Naive Bayes classifier displaying a score of 0.708. When we look at the classification report, the recall score of the majority class is much higher than the minority class. This is because of the disproportionate amount of information the models have on the majority class compared to minority class. Our best solution for the purpose of this project was to increase the recall of the minority class ('Charged off'). This was achieved through further tuning of our final model (Random Forest). We used the cost sensitivity parameter of the model to increase the cost of providing a wrong prediction for the minority class. The recall of the minority class has, therefore, increased from 0.781 to 0.879. Although, there is a trade off in decreasing the recall of the 'fully paid' class. But, for the purpose of this project we needed to prioritize recall of the 'minority class'. Overall, our final model was Random Forest with the following parameters:

max_depth=50, max_features=10, min_samples_leaf=1, min_samples_split=15, n_estimators=500, class_weight = {0:0.05, 1:0.95} .

Discussion and Conclusion

The motivation for this study came from having high quality public data provided by the Lending Club platform. We wanted to learn more and understand the problems faced by loan service providers, today, while also using the power of machine learning to predict potentially bad borrowers that can cost significant loss to loan business entities.

For this project, we considered related work in a theoretical sense rather than gaining domain knowledge on the loan industry. We researched how to overcome imbalanced classification problems, leading us to apply well-renowned practices such as Synthetic Minority Oversampling Technique and Cost sensitivity tuning to our project in order to solve our problem. Although there is not a single universal solution to the imbalance classification problem, we learned that the context of the domain is heavily important regarding which measures to prioritize to make the models more applicable in a practical setting. In addition, there were several more takeaways that we gained from this project. We learned that not all features are suitable for machine learning. Although we had an overwhelming number of features (150) in the beginning state of our data, we used various statistical techniques and domain knowledge research to preprocess our data to be more suitable for machine learning. We also gained some practical experience in the engineering side of machine learning. Hence, we quickly realized that we had a limited amount of computing power, with our significantly large dataset. We then learned that algorithms like Decision Trees and Random Forest computed much faster than Support Vector Machine Classifier. Therefore, when you have limited computing power and a project timeline, one does not have the freedom to choose the most sophisticated algorithm, which in turn would lead to a better performance.

Again, our best performing model was Random Forest. We chose our final model after tuning our hyper parameters, based on the performance of the ROC-AUC score. However, we carried out a deeper analysis using the classification report and confusion matrix. Our model was prioritizing the majority class which was the ‘fully paid’ class of loans. This was not an ideal answer to our problem statement. For loan companies, it is much more costly to accept loan requests from bad borrowers compared to rejecting loan offers from good borrowers. We incorporated this domain knowledge into our final model through cost sensitivity tuning, which increased the recall score of the minority class; the class of interest for this project. We hope that this model can add another layer of protection or insight for loan service companies to mitigate the risks of loans, while successfully identifying bad borrowers from the pool of loan applications.

For future work, we want to work and collaborate with real domain experts from the loan industry to further tune the model. Our cost sensitivity tuning was based on a well educated assumption that it is more costly to accept a loan from a bad borrower, but we did not know by how much exactly. Therefore, it serves a purpose of displaying our ability to increase the recall of the minority class in our model, but it does not guarantee that our model will maximize the return to a loan serve provider. Hence by working with a real domain expert from the loan industry, we would hope to learn the exact cost ratio between providing a loan to a bad borrower vs. not providing a loan to a good borrower. We could then encode this knowledge into our model, which would significantly increase its ability to maximize the profit or return to the loan service provider. In addition, we also want to explore a stronger engineering environment for tuning our models. A stronger environment would encompass a wider grid for parameter tuning, while allowing us to explore more taxing models.

Work Cited List

[1] Chawla, N. V., et al. “SMOTE: Synthetic Minority Over-Sampling Technique.” *Journal of Artificial Intelligence Research*, vol. 16, 2002, pp. 321–357., doi:10.1613/jair.953.

[2] Fernández Alberto, et al. *Learning from Imbalanced Data Sets*. Springer Science+Business Media, 2018.

[3] “Loan Statistics.” *LendingClub*, www.lendingclub.com/statistics/additional-statistics.

[4] “Press Release” *LendingClub*, www.ir.lendingclub.com/news/news-details/2020/LendingClub-Expands-LCX-Platform/default.aspx