September 22, 2022

LendingClub Data Analysis and Modeling

Capstone 2 Report, Meskerem Goshime

Springboard Data Science Program

[Project Files on Github]

"LendingClub is a US peer-to-peer lending company, headquartered in San Francisco, California." The company "enables borrowers to create unsecured personal loans between 1,000 and 40,000 dollars. The standard loan period is three years. Investors can search and browse the loan listings on Lending Club website and select loans that they want to invest in based on the information supplied about the borrower, amount of loan, loan grade, and loan purpose. Investors make money from interest. Lending Club makes money by charging borrowers an origination fee and investors a service fee." (source: https://www.kaggle.com/datasets/husainsb/lendingclub-issued-loans?select=lc_2016_2017.csv)

This study aims to help LendingClub to make informed lending decisions and investors to make informed investment decisions

The data made available on Kaggle is from 2007-2015 (as training data) and 2016-17 (as test data). The 2007-2015 data has 887,379 rows and 74 columns and the 2016-17 data has 759,339 rows and 72 columns.I decided to work with only 2007-2015 data and take out my training and test sets from this data.

I. Data Wrangling and Data Cleaning

A. Data Cleaning

I have listed some of the data cleaning I have done to prepare this data for analysis and modeling.

- ➔ The data did not have any duplicates.
- ➔ The data had a significant amount of missing values.
- ➔ I dropped columns with 50,000 or more missing values.
- ➔ After that, rows with missing values were dropped. This allowed me to keep the columns with a relatively small number of null values while making the data free of null values.

- ➔ The below pairs of columns had redundant data and I dropped one of the two columns.
    - ◆ Purpose and title
    - ◆ funded_amnt, funded_amnt_inv
- ➔ Null values in pub_rec columns were imputed with 0.
- ➔ I combined the values ANY and OTHER in the home_ownership column.
- ➔ In the loan status column I combined values that mean the same thing.
- ➔ I combined the values mortgage and own in the home_ownership column.
- ➔ Finally, based on my exploration of the data, I chose the columns that seem important for my analysis and dropped the rest. The columns I chose were the following:
    - ◆ loan_amnt, int_rate, grade, sub_grade, annual_inc, zip_code, dti, delinq_2yrs, inq_last_6mths, open_acc, collections_12_mths_ex_med, acc_now_delinq, term, emp_length, home_ownership, application_type, purpose, loan_status

B. Outliers Handling

The most significant outliers were in the Annual_inc column and the dti column.

For the annual income column, the 99.7 percentile was 379,557, but the maximum income was 9,500,000.

For the dti column, the 99.7 percentile was 39.06 while the max dti value was 380.53.

For both annual income and dti columns, I removed those few records that lie beyond the 99.7 percentile values of the respective columns.

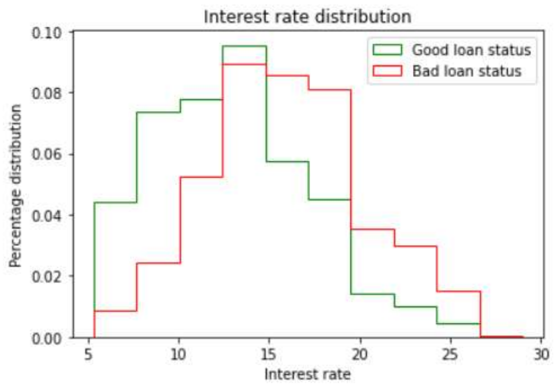
II. Exploratory Data Analysis

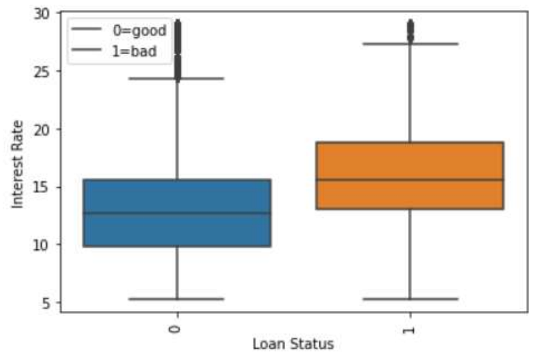
Here are some of the insights I got from the data in the Exploratory Data Analysis phase.

1. Interest rate of borrowers in bad loan status is significantly higher than the interest rate of borrowers in good loan status.

I selected those borrowers who are in bad loan status and calculated summary statistics of their interest rate. And then, I calculated summary statistics of interest rates for those good loan status. The below table shows the comparison between the two groups.

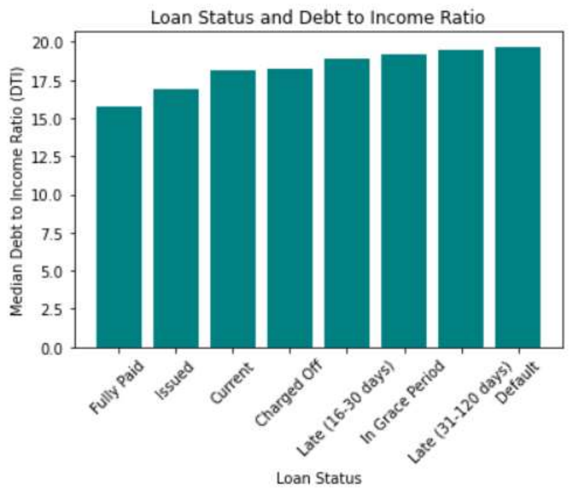
	Mean	Median	Standard deviation	5th, 25th, 50th, 75th, and 95th percentiles
Bad loan status	16.02	15.62	4.29	[8.9 13.05 15.62 18.85 23.76]
Good loan status	13.05	12.69	4.32	[6.62 9.76 12.69 15.61 20.53]



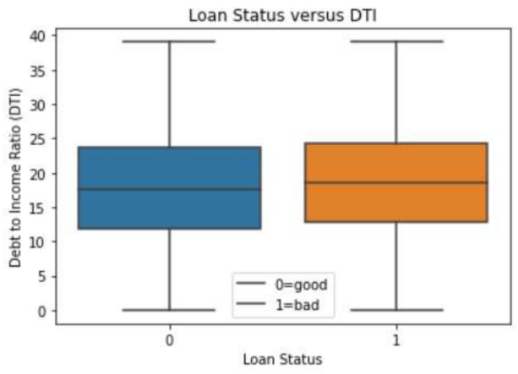


Interest rate appears to be an important factor affecting loan status. It seems borrowers with high interest rates struggle to pay back their debt and are likely to default on their loan. Therefore, it is in the best interest of the lender to keep the interest rate as low as possible.

2. The data shows that the bad loan statuses correspond with higher median dti value and the good loan statuses correspond to lower median dti value.



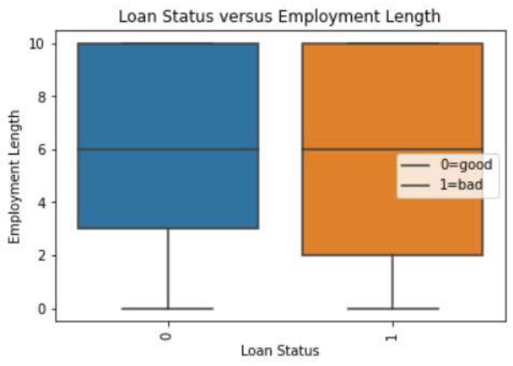
I then encoded loan statuses into two categories. Fully Paid, Issued, Current, and In Grace Period were categorized as good represented by 0. Charged Off , Late (16-30 days), Late (31-120 days) and Default were categorized as bad represented by 1. The below chart shows that those in the good category have a little lower DTI value than those in the bad category.



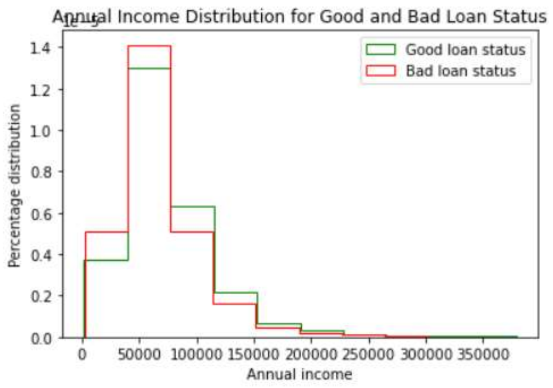
3. There is not a significant difference in employment length between those in bad loan status versus those in good status.

Summary statistics for employment length for those in good loan status versus those in bad loan status.

	Mean	Median	Standard Deviation	5th, 25th, 50th, 75th, and 95th percentiles
Bad loan status	5.80	6.0	3.64	[0. 2. 6. 10. 10.]
Good loan status	6.02	6.0	3.67	[0. 3. 6. 10. 10.]



4. **The data shows that those in bad loan status have lower annual income on average than those in good loan status.**



5. **Zip Code and bad loans**

These zip codes have 12% or more of their loans defaulting, which is the highest default ratio out of all other zip codes.

Zip Code	Bad Loans	Loan Count	Percentage in Bad Status
415xx	12	75	0.160
736xx	12	83	0.144
237xx	26	191	0.136
126xx	28	209	0.133
638xx	20	154	0.129
668xx	13	105	0.123

III. Preprocessing and Preparing the Data for Modeling

A. Target and Predictor Variables

- The data was split into the predictor variables (X) and the target variable (y).
- I chose the target variable to be the loan status column. I encoded the values as 0 and 1 as follows:

    - Current, Fully Paid, In Grace Period, and Issued became good loan status represented by 0.
    - Charged Off, Late (31-120 days), Late (16-30 days), Default became bad loan status represented by 1.
- The rest of the columns became predictor variables.

B. Some Categorical Columns Converted to Numeric Columns

I converted grade and subgrade columns from string to numeric values. This method is not perfect, since the distance between two consecutive letters is assumed to be equal. For example, the distance between A1 and A2 or the distance between A5 and B1 are assumed to be the same.

C. Standardizing Numeric Columns

The below columns were standardized using StandardScaler

> loan_amnt, int_rate, grade, sub_grade, annual_inc, dti, delinq_2yrs, inq_last_6mths, open_acc, collections_12_mths_ex_med, acc_now_delinq, term, emp_length

D. One Hot Encoding

The categorical columns were encoded using one-hot-encoding as follows.

- home_NONE, home_OTHER, home_OWN, home_RENT
- appl_INDIVIDUAL, appl_JOINT
- purpose_car, purpose_credit_card, purpose_debt_consolidation, purpose_educational, purpose_home_improvement, purpose_house, purpose_major_purchase, purpose_medical, purpose_moving, purpose_other, purpose_renewable_energy, purpose_small_business, purpose_vacation, purpose_wedding

F. Splitting the Data into Training Set and Test Set

The data was split into a training set (X_train, y_train) which is 80% of the data and a test set (X_test, y_test) which is 20% of the data.

IV. Modeling

The next step in the process was classifying and predicting loans to determine if they will be good loans or bad loans.

A. Feature Importance

I ran feature importance on a Random Forest Classifier and then chose the top 10 important features for the modeling. The top 10 important features were the following.

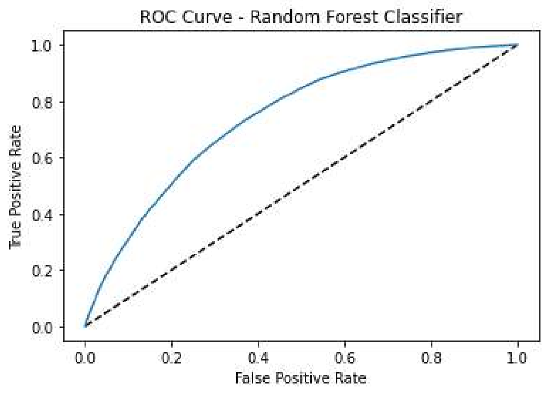
> dti, annual_inc, loan_amnt, int_rate, open_acc, emp_length, sub_grade, inq_last_6mths, delinq_2yrs, grade

B. First Model : Random Forest Classifier and Hyperparameter Tuning

I run the Random forest classifier with Grid Search CV to choose the best n_estimator value out of 100, 200 and 300. The Grid Search CV chose 300 as the best n_estimator value.

I also used manual tuning on the Random Forest model to choose from 50, 100, 300, 500, and 800. I was able to test more n_estimator values since the manual tuning took less time to run. With the manual tuning, I found 500 to perform well and run in a reasonable amount of time.

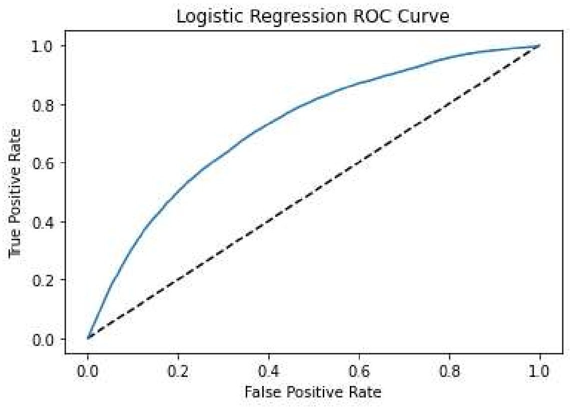
I finally ran the Random Forest Classifier with the chosen n_estimator value of 500. The ROC/AUC Score was 0.74. Please see the ROC curve below.



I also calculated the optimal threshold value as the highest tpr-fpr value and made predictions using the optimal threshold value. Please see the details in the Model Performance Comparison Table in **Section IV. E.** below.

C. Second Model : Logistic Regression

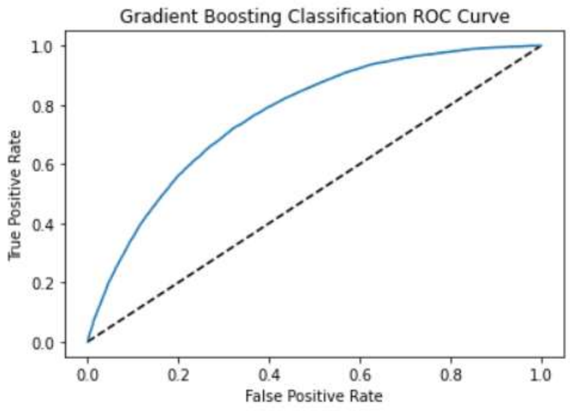
The second model I used was the Logistic Regression Model. The ROC/AUC Score was 0.72. Please see the ROC Curve below.



I also calculated the optimal threshold value where tpr-fpr is the highest. I then made predictions using the optimal threshold value. Please see the details in the Model Comparison Table in **Section IV. E.** below.

D. Third Model: Gradient Boosting

The third model I built was the Gradient Boosting Model. I performed manual tuning to choose the best n_estimators value among 50, 100, 200, 400, and 600, with the default max_depth value of 3. The performance seemed to increase steadily as the n_estimators value increased. However, the processing time for the model also increased with the n_estimators value. I also checked to see if the n_estimator value of 800 with max_depth value of 1 will perform well. In the end, I determined that n_estimator value of 600 and max_depth of 3 provides the best performance and reasonable amount of processing time. The ROC/AUC score for this model was 0.7668261420627009. Please see the ROC curve below.



I then calculated the optimal threshold value which came out to be 0.073607. I then made predictions using the optimal threshold value. Please see the details in the Model Comparison Table in **Section IV. E.** below.

E. Model Performance Comparison Table

	Random Forest	Logistic Regression	Gradient Boosting
ROC/AUC Score	0.74	0.72	0.77
Optimal threshold	0.072	0.065	0.074
F1 score with the optimal threshold value	Class 0 0.78 Class 1 0.22	Class 0 0.76 Class 1 0.21	Class 0 0.80 Class 1 0.24
Confusion matrix with the default 0.5 threshold value	[[152450 71] [11557 33]]	[[152520 1] [11590 0]]	[[152487 34] [11572 18]]
Confusion Matrix with the optimal threshold value	[[99246 53275] [3498 8092]]	[96218 56303] [3444 8146]	[[103940 48581] [3276 8314]]

The best performing model is the Gradient Boosting Model for the following reasons.

- ➔ It has the best ROC/AUC score and F1 Score.
- ➔ It minimizes the false positives while also keeping the false negatives low.

V. Challenges and Recommendations

What is challenging about this data is that it is highly imbalanced. The ratio of bad loans to good loans is about 7:100. The models tend to correctly classify the good loans, but have a hard time correctly identifying the bad loans.

The strategy I used was changing the threshold value from the default 0.5 to some optimal value. For each model, I calculated the optimal threshold as the value where tpr-fpr is the highest. I then used the new threshold value to make predictions for each model. This enabled the models to detect the bad loans better. However, it came at the cost of misclassifying some good loans as bad loans. However, this is a better approach since the cost of misclassifying bad loans is a lot higher than misclassifying good loans.

Other techniques like resampling may be applied to handle the imbalanced data.