Springboard--DSC

Capstone Project 2

Loan Application Approval

T. Hastings Reeves

April, 2021

[1] Introduction

The Dream Housing Finance Company's current process of approval for loan applications involves weighing specific criteria found in the application and deciding whether or not to approve the loan request. This requires a significant amount of company time and resources in reading through individual loan applications, where the company would rather have the few loan applications that are strong candidates for approval to search through. Thus, Dream Housing Finance Company is looking to automate the loan eligibility process (real-time) based on the customer detail provided in the loan application. This information includes but is not limited to gender, marital status, education, number of dependents, income, credit history and others.

Look to my github repository listed below for detailed notebooks on the implementation of datasets, provided in the repository, data wrangling/cleaning, exploratory data analysis, development of a baseline model, and further enhancement of models via multiple machine learning algorithms.

Github repo:

https://github.com/threeves91/Springboard/tree/main/Capstone%20Project%202

As you will see in the notebooks, and as I will outline here in this report, we can streamline the loan application approval and denial process by categorizing loan applications in a way that streamlines the application process and alleviates the workload of the loan officers.

[2] Approach

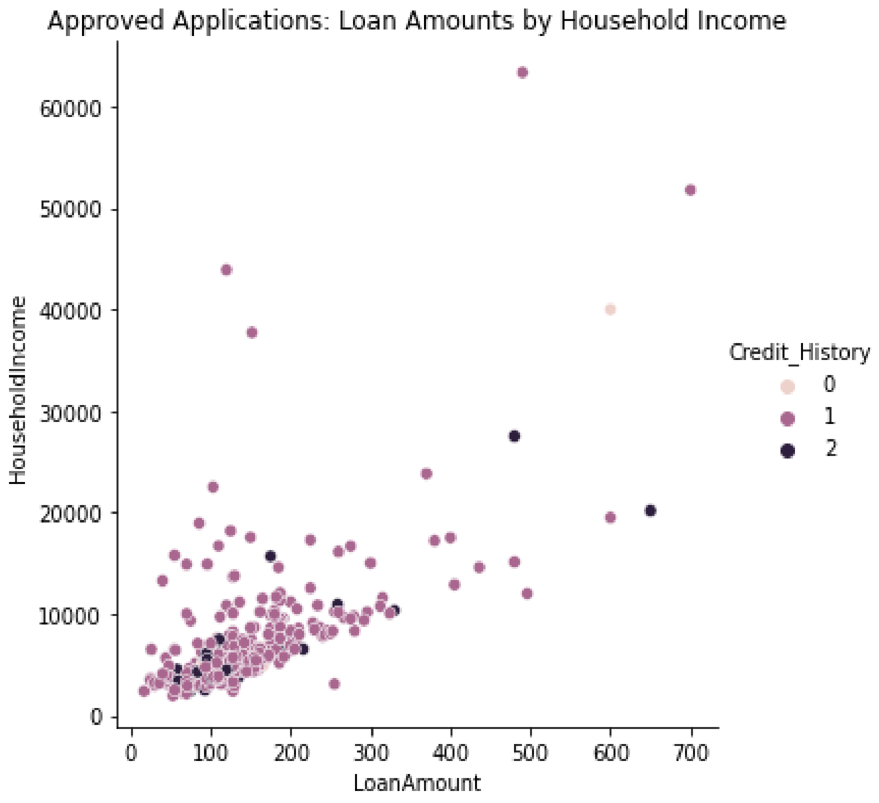
[2.1] Data Acquisition and Wrangling

This dataset was available on the following Kaggle website: https://datahack.analyticsvidhya.com/contest/practice-problem-loan-prediction-iii/#ProblemStatement
This dataset was relatively clean to begin with, not needing any significant cleaning of any kind. It was clear from the beginning that this would be a classification algorithm, that the target variable would be the Loan_Status and that there were minor issues with NaN values in several dependent variables, but nothing that would require more information from the stakeholders.

[2.2] Storytelling and Inferential Statistics

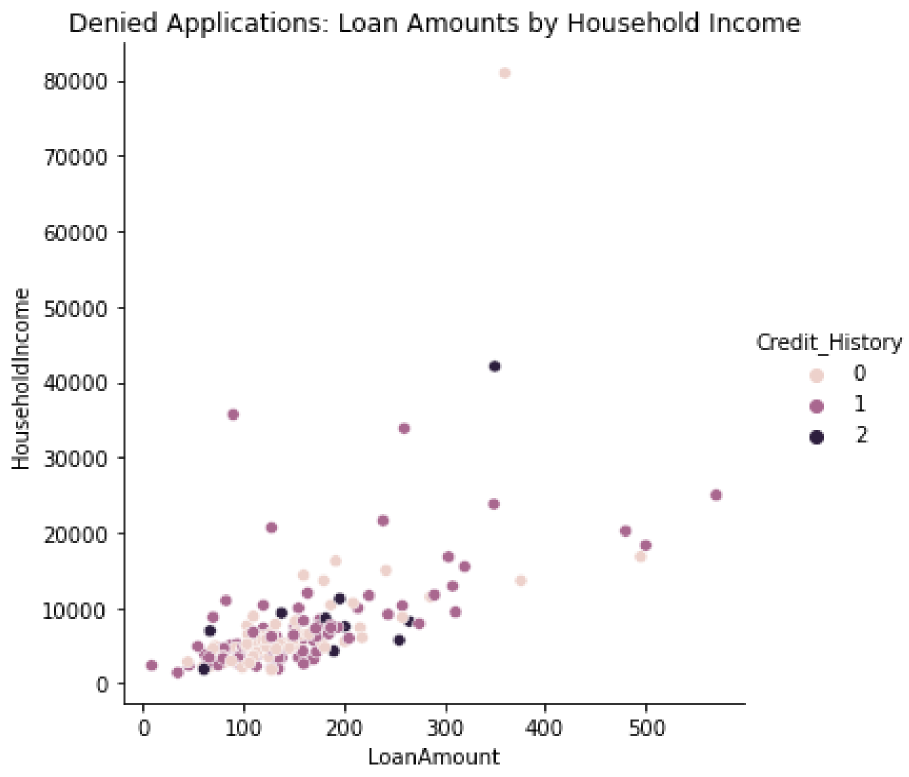
In my exploratory data analysis I sought to remedy any NaN values in any variables, combine the Coapplicant and Applicant Income variables into HouseholdIncome, and split the dataset into LA_Y and LA_N (Loan Application Approved or Not Approved) in order to observe the relationships between different variables based on their approval. For Example:

The following graph shows that very few applications that did not meet credit requirements were approved,



While the next graph indicates the opposite to be true, that of the Denied applications, most of the applications that did not meet the credit history requirements are in this category:

see next page



[2.3] **Baseline Modeling**

As I mentioned in the introduction, it was apparent that the baseline model I would attempt would involve a classification algorithm, of which I chose Logistic Regression. I attempted several different train/test splits and settled on a 70/30 split as my preferred model based on the classification report being slightly better for this model. The Accuracy rating was decent for all of the models, while the classification report told a different story. Ultimately it was determined that the recall performance metric for the minority class needed boosting, as it stood at .42 (marginally worse than a coin flip!). Below is the classification report for the baseline model with a 70/30 split and utilizing stratification to counter the imbalance in the dataset between the Y and N responses in the target variable.

[Training Classification Report]

```
         precision    recall  f1-score   support

N:          0.87      0.42      0.56       144

Y:          0.79      0.97      0.87       316

accuracy:                       0.80       460

macro avg:  0.83      0.69      0.72       460

Wghtd avg:  0.81      0.80      0.77       460
```

[Test Classification Report]

```
          precision    recall  f1-score   support

N:           1.00      0.44      0.61       48

Y:           0.80      1.00      0.89       106

accuracy:                        0.82       154

macro avg:   0.90      0.72      0.75       154

Wghtd avg:   0.86      0.82      0.80       154
```

[2.4] Extended Modeling

Ultimately, it was determined that the model’s performance could be attributed to the imbalance within the dataset. The ratio between approved and denied outcomes was nearly 3:1 which skewed the classification towards the majority class, leading to a high precision score in the majority class and a low recall score in the minority class. I utilized Random Oversampling of the Minority class, Random Undersampling of the Majority class, and Synthetic Minority Oversampling Technique(SMOTE) in an attempt to boost the recall metric of my Logistic Regression algorithm. I also utilized the same sampling techniques on a Random Forest algorithm as well as a Decision Tree algorithm to compare the algorithmic performance with the baseline model. Ultimately, the Random Undersampling of the Majority class for the Logistic Regression model and the SMOTE method for the Random Forest model showed the best performance metrics, with the Random Forest model being the preferred choice going forward. See following section for summary and comparative analysis of results.

[3] Findings

TEST SET CLASSIFICATION REPORT:

(Baseline model uses Training data CR due to overfitting as a result of categorical imbalance)

*** Best Model**

**** Performing Models**

MODEL:	Recall: Minority	Precision: Minority	F1: Minority	Recall: Majority	Precision: Majority	F1: Majority	AUC
Baseline Model: Logistic Regression(Training data)	0.42	0.87	0.56	0.97	0.79	0.87	0.744
Logistic Regression: Random Oversampling	0.55	0.27	0.36	0.31	0.6	0.41	0.738
Logistic Regression: Random Undersampling	0.6	0.49	0.54	0.71	0.8	0.75	0.741
**Logistic Regression: SMOTE	0.53	0.55	0.54	0.8	0.79	0.8	0.745
**Random Forest: Random Oversampling	0.53	0.65	0.58	0.87	0.8	0.83	0.744
Random Forest: Random Undersampling	0.62	0.52	0.57	0.74	0.81	0.77	0.738
*Random Forest: SMOTE	0.53	0.66	0.59	0.87	0.8	0.84	0.749
Decision Tree: Random Oversampling	0.55	0.55	0.55	0.8	0.8	0.8	0.674
Decision Tree: Random	0.55	0.39	0.46	0.61	0.75	0.67	0.579

Undersampling							
Decision Tree: SMOTE	**0.55**	**0.52**	**0.54**	**0.77**	**0.79**	**0.78**	**0.662**

With the goal of boosting the recall score of the minority class in mind, there were two issues that I found. First, the Random Undersampling technique would boost the recall score the most in each algorithm, while diminishing the precision score of the majority class. This would result in a lower AUC and diminished performance compared to the baseline model. Second, the Decision Tree algorithm as a whole did not perform as well as the baseline model.

The best model at boosting the recall of the minority class while maintaining a reasonable precision score in the majority class was the Random Forest-SMOTE. Here we see an increased performance in the AUC as well. The Random Forest algorithm as a whole seemed to perform better than the Logistic Regression algorithm after resampling techniques were applied. As we begin to expand our view from algorithmic analysis to real-world business problems some things are becoming clear. There is not enough data to ensure enough confidence in any of these models to produce a definitive approval or denial of every loan application. Were there 6,000,000 applications being used in this process as opposed to 600, the models accuracy and performance metric scores may increase in such a way that would allow one to put the loan applications in those two categories. However, there is enough positive performance with these models as is to suggest a confidence-based grouping of loan application results.

[4] **Conclusions and Recommendations**

We can streamline the loan application approval and denial process by categorizing loan applications in the following groups: definitely approved, tentatively approved, tentatively denied, definitely denied. With the definitely approved and definitely denied applications, that will decrease the workload of loan officers working through loan applications and allow them to focus on the tentatively denied and approved loans.

[5] **Future Work**

- If there is room for further exploration in these models, I would be interested in looking at the Random Undersampling techniques for both the Random Forest and Logistic Regression algorithms. With such a discrepancy in categories, the opportunity to create dozens or perhaps hundreds of undersampled models, average their classification reports and see if the performance metrics improve is there.

- Utilizing and ensemble method approach between the SMOTE resampling technique for the Random Forest and Logistic Regression algorithm would be interesting and could provide an enhanced model.
- Going back to the baseline model and using a clustering algorithm could also provide valuable insights and another way of looking at this problem.

[Consulted Resources]

- https://c3.ai/blog/using-binary-classification-metrics-to-maximize-enterprise-ais-potential/
- https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c
- https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/