School of Information Technologies
Faculty of Engineering & IT

Predicting "Bad" Loans Using Lending Club Loan Data

Unit of Study: INFO3406, Introduction to Data Analytics

Assignment name: Project Stage 2

Tutorial time: Monday, 11am

Tutor name: Dr. Ali Anaissi

Student ID: 480490915

Student name: Ryan Dunham

Signed: Ryan Dunham **Date:** 26/10/18

Experimental Setup

My goal in this project is to create a classifier to predict "bad" loans based on the Lending Club loan database. A "bad" loan, is one in which the loan status is default, charged off, late (16-30) days, late (31-120) days or in grace period. On the other hand, a "good loan", is one in which the loan status is fully paid, issued, or current. My aim is to create a classifier that will be able to correctly predict whether or not a given loan will turn out to be "good" or "bad". In order to accomplish this task, I will construct a logistic regression classifier.

While my main focus in this experiment is to create an effective logistic regression model with resampled values, in order to test the effectiveness of this model, a simple logistic regression model will be used as a baseline comparison. As a result, my hypothesis is as follows:

H0 (Null Hypothesis): There is no change or difference between the two classifiers

HA(Alternative Hypothesis): There is a significant difference in performance between the two classifiers

While Classification F1 score is a common measure of effectiveness for classifier models, I will use both the F1 score and type II error for comparing the classifiers I have built. F1 Score is computed as 2*((precision x recall)/(precision +recall)) and essentially measures the harmonic average of the precision and recall. Type II error or false positive is the rejection of a true null hypothesis. In our case this would be falsely classifying a loan as "good".

Approach/Results

I began my modeling process with some preparation. First, I began by encoding my Target variables "good and "bad" using dummy variables 0 and 1 respectively. I then dropped the loan_status feature in my dataframe because it was used to build the target variables and would lead to overfitting if it was used in the modelling process. Also, when using the model in practice, this loan status is not available so it is of no use in this stage. Since I am building a logistic regression model, I must label encode the categorical features with <=2 possible values and hot encode features with >2 possible values. However, in order to keep the dimensions of the dataset down, I decided to drop categorical features with more than 2 possible values. By dropping these values, the amount of features in the modeling dataset is 35 as opposed to 150.

With the model preparation stage complete, my model is now ready to be trained. I begin by creating a test-train split of my dataset, allocating 80% to training and 20% to testing. A baseline logistic regression model with untuned parameters, unscaled values and non resampled data is trained and the results are as follows:

THE UNIVERSITY OF SYDNEY

```
                  Classification Report
              precision    recall  f1-score   support
           0       0.97      1.00      0.99    163859
           1       0.99      0.66      0.80     13617
   micro avg       0.97      0.97      0.97    177476
   macro avg       0.98      0.83      0.89    177476
weighted avg       0.97      0.97      0.97    177476
```

```
Confusion Matrix
Predicted   0      1
Actual
0        163802   57
1        4562    9055
```

According to the confusion matrix, the amount of false negatives, or the number of loans our model predicted as 'good' but actually are 'bad' is rather high (around 4562). The amount of false positives, or the number of loans our model predicted as 'bad' but turned out to be good' is very low at 57. Thus the recall score is around 0.66, which is problematic. While lending loan is hurt by false negatives, false positives do not have much of an impact, except that you would essentially turn down "good" loans. This low recall score caused by the high rate of false negatives might be a result of the class imbalance. Because there are so few "bad" loans in the dataset, approximately 7%, the model may be achieving high accuracy by predicting more loans as "good" purely as a result of this imbalance. While the model did achieve high accuracy, the accuracy may be simply reflecting the underlying class distribution**.**

In order to attempt to fix the class imbalance, I performed oversampling on the training set using SMOTE. Before resampling, I began by scaling the data using StandardScaler which transforms the data so that each features distribution will have a mean value of 0 and standard deviation of 1. Since our data is of many different units, standard scaling allows them to be compared by striping them of their units. After scaling the data, I then performed oversampling. SMOTE or synthetic minority oversampling technique works by creating synthetic samples from the minor class instead of creating copies. The algorithm selects two or more similar instances (using a distance measure) and perturbing an instance one attribute at a time by a random amount within the difference to the neighboring instances.

```
            Classification Report
        precision   recall  f1-score   support
        0      0.98      0.97      0.97    163859
        1      0.64      0.74      0.69     13617

  micro avg       0.95      0.95      0.95    177476
   macro avg       0.81      0.85      0.83    177476
weighted avg       0.95      0.95      0.95    177476
            Confusion Matrix
            Predicted   0      1
            Actual
            0        158194    5665
            1        3481    10136
```

Oversampling using SMOTE improved the recall score by approximately 8%, yet the F1 score decreased by 2%. While the rate of false negatives decreased which, the rate of false positives increased significantly. While this is not necessarily problematic in the case of loan default, would be "good" loans are essentially flagged as "bad" and would be turned away. However, it is more important to reduce the amount of false negatives, as they can hurt the company.

While oversampling and scaling the data improved the rate of false negatives or recall, the f1-score decreased. In order to increase the f1-score and attempt to reduce the rate of false negatives, I preformed undersampling. Undersampling essentially changes the dataset used to build the predictive model to have more balanced data by deleting instances from the over-represented class. I resampled my data, through undersampling and then trained a logistic regression classifier on the now balanced data, the results are as follows:

```
            Classification Report
        precision   recall  f1-score   support
        0      0.79      0.97      0.87     10834
        1      0.96      0.73      0.83     10691
micro avg       0.85      0.85      0.85     21525
macro avg       0.87      0.85      0.85     21525
weighted avg       0.87      0.85      0.85     21525
            Confusion Matrix
            Predicted   0      1
            Actual
            0        10534    300
            1        2864    7827
```

While the recall score did not improve, undersampling did solve the issue that oversampling created. Oversampling greatly increased the rate of false positives. While the average f1-score is lower than the two previous classifiers, the undersampled logistic regression classifier seems to offer the best of both worlds. While it improved the recall of the first model by 7%, it did not increase false positives like the oversampled classifier.

Limitations

One limitation of this dataset was the amount of features and sheer size of the overall dataset. The original dataset consisted of over 887,000 entries with 73 features. After reducing features through cleaning, I was left with around 35 features for modelling. While I attempted to reduce redundant or mostly null features, this dataset proved to be an immensely high dimensional and complex. I originally had planned to perform hyperparameter tuning to increase the performance of the classifier, however, the dataset was too high dimensional and the process was too computationally complex for my machine.

Not only was this dataset immense but it was largely imbalanced. Lending loan denies up to 90% of loan applications, as they focus on high-creditworthy borrowers to reduce risk. Essentially, this dataset consists of only approved applicants that have gone through a strict approval and screening process. As a result, a large class imbalance existed in the target variable. 93% of loans were “good” and only 7% of loans were “bad”. Class imbalance is especially problematic for learning algorithms as it is difficult to discern whether or not the high accuracy may be simply reflecting the underlying class distribution.

Conclusion

I would recommend my classifier as a solution to the problem of loan default for lending club. As stated before, lending club denies around 90% of potential borrowers and focuses on only high-credit worthy individuals. Despite this effort, still around 7% of these borrowers end up becoming “bad” loans that end up costing time and money. While my classifier was not perfect, it would be able to cut the amount of “bad loans” in half. This secondary monitoring/screening could be a second step to lending-clubs screening process, and would effectively reduce the amount of “bad” loans.

I have learned many lessons from conducting this project. Foremost, I learned about how difficult it is to work with large and high dimensional data sets. Not only are they difficult to grasp and understand, but they can be arduous to work with. Only the most efficient methods for analysis can be used, as inefficient loops used for cleaning could take hours. This high dimensionality caused problems for hyperparameter tuning, as it was too computationally expensive. While PCA could have been performed and the amount of features further reduced, this does not always result in improved performance. In addition, I learned alot from the target class imbalance in this dataset. While methods such as resampling or changing performance metrics can help solve an imbalance, the effects can still pervade the dataset.