Predicting Loan Default using Feature Selection and Ensemble of Classifiers Final Report for CIS419

Jeff Barg JEFFBARG@WHARTON.UPENN.EDU
Paul Asselin PASSELIN@SEAS.UPENN.EDU
Karthic Thangarasu KARTHICT@SEAS.UPENN.EDU
Akshay Chandramouli AKSHAYCH@SEAS.UPENN.EDU

Abstract

In this paper we present a predictor for loan defaults. Our data was obtained from Kaggle.com (Kan, 2016), a popular page for data science. We performed feature engineering to add attributes that we believed may be conducive to a stronger predictor and then performed feature selection to narrow down from the list of 70+ attributes to obtain a list of 'Golden Features'. Using these 'Golden Features' we performed supervised training on an a group of regression, ensemble and probabilistic classifiers to create an ensemble which polls from this group of classifiers to predict whether any given loan will default or not.

1. Introduction

Lending Club is the worlds largest online marketplace connecting investors and borrowers. Using an online platform for peer-to-peer lending, LendingClub is transforming the banking system by operating at low costs, "making credit more affordable and investing more rewarding." (LendingClub, 2016)

1.1. Goals

The goal of this analysis is to determine whether a borrower will default on their loan, using both traditional features (income, FICO score, interest rate) and nontraditional features (inflation rate, location). We would like to see if these features that we will engineer are in fact essential, and provide lenders with potentially more information when deciding upon a loan.

University of Pennsylvania, CIS 419/519 Course Project.

1.2. Motivation

Analyzing the credit-worthiness of a borrower is an essential step in the loan-making process which has been going on for hundreds of years to varying degrees. Having funded $\$20,687,488,911$ (LendingClub, 2016) in loans to this date, the LendingClub is a clear force in the loan market. Machine learning algorithms can be used to assist institutions in accurately predicting the riskiness of borrowers based on loan book data. We aim to apply a novel approach that that uses feature engineering to include data from external, macroeconomic, factors as well as an ensemble of classifier techniques which will hopefully boost predictive power.

2. Data Collection and Preprocessing

2.1. Data Overview

Through Kaggle, we obtained the Lending Club's loan book portfolio, a collection of $887,379$ loan cases with an initial collection of 77 different features.

Table 1 shows the breakdown of loan status'.

2.2. Data Preprocessing

We took a number of preprocessing steps in order to clean up our data set.

Since any loan that is "Current" has no associated default or no default label, we will be holding out this portion of the data and making predictions on it using our final classifier. We have also gone ahead and removed all instances that aren't "Fully paid", "Charged Off", or "Default". These are the only instances that we can train our classifier with because the outcome is certain and known. For any other instance, for example a "Late" loan, there is no definitive answer as to whether that loan will default or not and thus it has been ignored when training our predictors. A "default" loan becomes "charged off" and written off the balance sheet after a period of 30 days so they have been binned together. We changed many of the class labels to numeric

labels (Such as the grade and subgrade fields, which we converted to a 0 to 5 scale).

As we have narrowed down our selection of data to "Fully paid", "Charged off", and "Default", we then decided to binarize our data. All "Fully paid" loans are our positive instances and all loans that are "Charged off" or "Default" are our negative instances. Any loan that is "Default" becomes written off from the balance sheet and is charged off after an initial period of 30 days. We thus restricted the possible values for an end-state and assigned 1 for a "good" outcome and 0 for a "bad" outcome (Charged-off or Default).

Additionally, we had to remove certain attributes from our data set to remove skew. Features such as "Late Fees", "Interest Received to Date" and "Principal Received to Date" are features that are not available to us when the loan is 'Current'. Removing these features ensures that feature selection picks attributes that are available for on-going loans.

3. Feature Selection and Engineering

3.1. Feature Engineering

In addition to the 77 current features in a training instance, we decided to engineer other features that we believe may add to the predictive power of our classifier. We correlated the time of the loan to the inflation rate at the time of loan disbursal as well as the federal interest rate level to engineer features that reflect the state of the full economy. The price level and interest rate both reflect the macro-economic state of the economy. Using the Zip Code of each loan, we expanded our data to include the median income for the location of the loan (UMichigan, 2016) as well as the difference (measured as a percentage) between median income and applicant's income to add location and applicant-specific factors. Features such as these reflect the economic performance of the specific area of the applicant and the applicant's relative economic position with that location. We hope that these features will add insight when during the learning process.

Additionally, we decided to remove the "title" field given in our loan book. The "title" field differs from our "purpose" field in that it's a stream of text while the purpose field is already binned into several categories such as "debt_consolidation", "credit_card", "home_improvement", etc. However, the title and the purpose are essentially the same and title wouldn't give us any additional insight. The top 10 loan purposes can be seen in figure 1.

Figure 1. Loanbook breakdown by purpose.

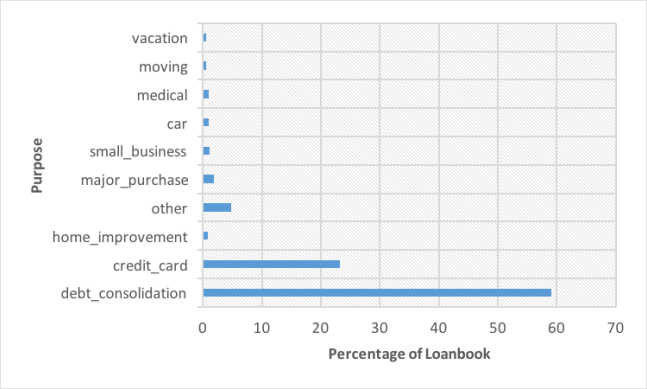

Table 1. Loanbook breakdown

STATUS	FREQUENCY	PERCENTAGE
CURRENT	601,779	67.8
FULLY PAID	207,723	23.4
CHARGED OFF	45,248	5.1
LATE (31-120 DAYS)	11,591	1.3
ISSUED	8,460	1.0
IN GRACE PERIOD	6,253	0.7
LATE (16-30 DAYS)	2,357	0.3
DEFAULT	1,219	0.1
NOT MEETING CREDIT POLICY	2,749	0.3

3.2. Feature Selection

Using the initial and engineered features, we then performed feature selection to obtain a list of "Golden Features".

We used a greedy forward process algorithm that incorporated the Binomial Generalized Linear Model (Binomial GLM) and used the Area-Under-Curve (AUC) (Nichol, 2016) approach to extract a list of 23 "Golden Features". We looked at the AUC of for different pairs of features to see if together they were valuable. We looked at feature pairs which had AUC's greater than 0.5 "as this threshold is the equivalent to random guessing" (Nichol, 2016).

4. Method

We created an ensemble classifier to maximize the predictive power of our approach. We measured the performance of each individual classifier using k-cross fold validation and obtaining performance measures. However, as the data is heavily imbalanced and positively skewed (due to the relatively small number of defaults) we needed a performance measure that goes further than just accuracy, recall, or precision. We needed to take into account the balance of each

class.

We used the geometric mean (G-Mean) (Kub) :

$$\sqrt{\frac{TN}{TN+FP}+\frac{TP}{TP+FN}}$$

"The goal is to maximize the accuracy on each of the two classes while keeping these accuracies balanced" (Kub). This gave us a much better indication to the performance of each classifier. We used this metric as well as accuracy to measure the performance of our predictors. By cutting out all the active loans, we have already added a lot more balance to our data set than was initially present. We gave great consideration to creating a 50-50 split within the data, creating a new data set with all the default cases and randomly picking just as many no-default cases to create a balanced data set but we voted against it. We decided that having more information would be better than creating a balance by severely reducing our training set.

Testing throughout is done by training classifier on 80% of data and testing on 20% and then performing cross validation.

5. Regression Techniques

We seek to add a classifier that uses regression. The two classifiers we looked at were MARS (Multivariate Adaptive Regression Splines) and Penalized Logistic regression

5.1. MARS

We used Orange's 'earth' package (Orange, 2016) utilize the MARS classifier. We chose the MARS classifier because of its high computational scalability and it's ability to handle missing and mixed data. This very flexibility model is the result of the optimization of (Anonymous):

$$\sum_{i=1}^{k} c_i B_i(x)$$

where $B_i(x)$ is a basis transformation of x. The model is essentially a weighted sum of basis functions. Each basis takes the form of a constant, a hinge function, or a product of hinge functions (Anonymous). The algorithm follows two procedures: a forward pass and a backward pass.The forward greedily adds the pair of basis functions that reduces residual error the most and continues until the residual error is too small or the maximum number of terms is reached. As we are minimizing residual error, the forward pass is certain to overfit the data. For this, a backwards pass is needed to generalize the model. At each step, we "prune and removes the least effective term until the best function subset is found" (Salling, 2013). Subsets are compared using General Cross Validation function (GVC) (Salling, 2013) given by:

Table 2. MARS results

TYPE	TESTING ACCURACY	G-MEAN
MARS	87.3	82.4
LOGISTIC REGRESSION	85.6	79.2

$$GCV(\lambda) = \frac{\sum_{i=1}^{N}(y_i - f_\lambda(x))^2}{1 - M(\lambda)/N^2}$$

.

where

$$M(\lambda) = \lambda + \frac{penalty(\lambda - 1)}{2}$$

where λ is the number of MARS parameters, N is the number of training instances and the penalty can be specified. Essentially this formula looks at the trade off between the number of parameters and the residual error (Salling, 2013).

the results of this classifier can be seen in table 2. We tuned the degree parameter and and obtained optimal results for degree = 3.

5.2. Penalized Logistic Regression

L1 Penalized logistic regression looks to minimize (GLM, 2016):

$$\|W\|_1 + C\sum_{i=1}^{n} log(exp(-y_i(X_i^T w + c)) + 1)$$

which incorporates a penalty parameter when maximizing the log likelihood of any given estimate. This is to "shrink the estimates of the regression coefficients towards zero relative to the maximum likelihood estimates. The purpose of this shrinkage is to prevent overfit arising due to either collinearity of the covariates or high-dimensionality" (Jelle Goeman). Tuning across a set of C = $10^-4, 10^-2, 1, 10^2, 10^4$ yielded an optimal parameter of C = 1. Results can be seen in table 2. MARS appears to be a superior classifier and was added to the final classifier.

6. Ensemble Learning

We look at three different ensemble methods to use: 2 boosting algorithms and one bagging algorithm. The clas-

Table 3. Boosting Results

TYPE	TESTING ACCURACY	G-MEAN
ADABOOST	83.2	70.2
XG BOOST	84.7	71.5
RANDOM FOREST	67.9	45.3

Table 4. Naive Bayes Results

KERNEL	TESTING ACCURACY	G-MEAN
MULTINOMIAL	57.6	52.9
BERNOULLI	79.3	60.2
GAUSSIAN	84.8	80.1

Table 5. Naive Bayes Results

TYPE	TESTING ACCURACY	G-MEAN
ENSEMBLE CLASSIFIER	88.2	80.9

sifiers for the boosting algorithms each used the decision stump as a base learner with 1,000 estimators used. can be seen in table 3.

6.1. Adaptive Boosting

Described as the best off the shelf classifier, the AdaBoost algorithm was used to benchmark against the XG Boosting algorithm.

6.2. XG Boosting

Like all Boosting algorithms, XGBoosting creates an ensemble of weak learners to obtain a strong learner. While AdaBoost identifies misclassifications by the weight of each instance, regular Gradient Boosting identifies misclassifications by the gradient of each instance (Li). However, for regular gradient boosting, each learner is built in series in order to compute a gradient, preform gradient descent, and minimize the loss function. This is computationally expensive and thus Extreme Gradient (XG) Boosting was used. With XG Boosted trees, the entire tree itself is built in "parallel fashion" (Steinweg-Woods, 2016) rather than sequentially and gives a speed advantage. We used the XGBoost API provided on their website (XGB).

6.3. Random Forest

As a versatile bagging algorithm, random forest can be used for a variety of decision boundaries and was used here using its default parameters. However, due to the high imbalance in our data set, its low performance and very low G-Mean is not surprising as each tree is learning uneven patterns and not generalizing well, essentially overfitting. Due to the XGBoosting's better performance and its clear advantage in speed, it was added to our final predictor instead of AdaBoost and Random Forest.

7. Naive Bayes

Having already explored regression techniques, and ensemble methods we also wanted to use a classifier that utilizes probabilistic classifiers. The results of Gaussian, Multinomial, and Bernoulli are seen in table 4. While we didn't assume independence, the Gaussian classifier was a good predictor in this case, clearly outperforming Multinomial and Bernoulli in terms of both accuracy and balance. The Gaussian classifier was added to our final ensemble.

8. Final Classifier

Our final classifier was an ensemble of MARS, XG Boosting, and Naive Bayes using the Gaussian Kernel. We believe this is a very balanced ensemble as it is very diverse in using a regression technique, an ensemble technique and a probabilistic technique.

8.1. Its general performance

We used a simple polling technique to determine the final prediction of "default" or "no-default" where majority vote ruled. The final performance of this classifier can be seen in table 5. This classifier had an overall better testing performance but didn't have a significantly larger G-mean than any of our individual classifiers. This means that our final classifier has a similar imbalance level as its individual predictors.

8.2. Applying it to the list of "Current" Loans

After training our classifier on the full training instance, we let it predict default or no-default to the collection of held out loans that are currently active. The overall default rate predicted was 2.7%.

9. Further Work

We considered using text classification over the user-provided loan description to see whether we could determine whether the terms used in the loan description would give us good insight into whether the loan would be successful or not. We however ended up deciding that this would likely not improve our accuracy or precision much due to the small and eratic nature of these descriptions

(some were well written and several sentences, whereas some were merely a sentence). In the future, we could consider possibly parsing, trimming, and tokenizing each string and then performing TF-IDF to compute values for each description and then performing feature selection with this modified field.

10. Conclusion

While we greatly trimmed our initial data set in order to binarize and pre-process the data, we felt that the classifiers we trained and used were adequate. Whilst dealing with data imbalance was difficult, using the G-mean became a good metric to measure performance alongside accuracy. Keeping this in mind, our classifier obtained a 2.7% default rate on LendingClub's active loans. This seems plausible as the average default rate hovers around 3% (Gustke, 2014).

While we initially started with a dataset of over $880,000$ cases, we were forced to remove approximately 70% of instances as they were not labeled. Thus we performed supervised learning on a significantly smaller data set and used it to predict outcomes on a much larger, probably sparser data set: a cause for concern. This is an error which could not be avoided given these circumstances but must be noted when evaluating and using the final model.

The general erratic nature of loans, and even more sporadic nature of online loans, means that our classifiers will generally overfit to the data. This along with the uneven balance already present in the data set (the sparse amount of actual default loans) means that our classifier is most likely to under-report the default rate. This could be the case in our example. Furthermore, the 3% is quoted from regular, commercial banks. LendingClub is an online peer-to-peer bank where money is being spread out across portfolio to minimize risk, a different infrastructure with its own unique default rate that will most likely be different than that of a commercial bank. Regardless, due to the generally good testing accuracy and G-Mean we obtained from our classifiers using cross validation, we have faith that our diverse and versatile classifier is suitable for predicting loan outcomes.

Acknowledgments

Grateful to Dr.Eaton and TA's for a wonderful semester.

References

Xgboost. URL http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdff.

Generalized linear models, 2016. URL https://scikit-learn.org/stable/modules/linearmodel.html.

Anonymous. Multivariate adaptive regression splines. URL https://en.wikipedia.org/wiki/Multivariate_adaptive_regression_splines.

Gustke, Constance. Back to banking basics. http://www.bankrate.com/finance/cd/bank-loan-funds.aspx, 2014.

Jelle Goeman, Rosa Meijer, Nimisha Chaturvedi.

Kan, Wendy. Lending club loan data, 2016. URL https://www.kaggle.com/wendykan/lending-club-loan-data.

LendingClub. Lendingclub.com, 2016. URL https://www.lendingclub.com.

Li, Cheng.

Nichol, Kiri. Predicting loan defaults, 2016.

Orange. orangecontrib.earth 0.1.4, 2016. URL https://pypi.python.org/pypi/orangecontrib.earth/0.1.4.

Salling, Jack. Multivariate adaptive splines, 2013. URL http://www.lans.ece.utexas.edu/courses/ee380l_ese/2013/mars.pdf.

Steinweg-Woods, Jesse. A guide to gradient boosted trees with xgboost in python, 2016. URL https://jessesw.com/XG-Boost/.

UMichigan. Zip code characteristics: Mean and median household income, 2016.