Predicting Default Risk of Lending Club Loans

Shunpo Chang
Stanford University
shunpoc@stanford.edu

Simon Dae-oong Kim
Stanford University
simonkim@stanford.edu

Genki Kondo
Stanford University
genki@stanford.edu

Abstract

Lending Club is a peer-to-peer lending company, the largest of its kind in the world with $11.1 billion originated loans. It is an online lending platform where borrowers are able to obtain loans and investors can purchase notes backed by payments based on loans. In this paper, we will attempt to predict the expected returns for loans to a given borrower. Since a loan default will result in a loss of both principal and interest, we will attempt to maximize our returns by predicting the probability of default of the borrower so as to help avoid investment in those high-risk notes.

I. Introduction

The 2-year U.S. Treasury bond yield hovers below 1%, so Lending Club offers an attractive alternative to bonds for steady investment income. Lending Club offers loans of various grades they assign that correspond to specific interest rates for investors. The higher the interest rate, the riskier the grade. The risk comes in the form of defaults - whenever a loan defaults, investors end up losing a portion of their investment. We believe that there is inherent variation between loans in a grade, and that we can use machine learning techniques to determine and avoid loans that are predicted to default.

The Lending Club dataset contains a comprehensive list of features that we can employ to train our model for prediction. The dataset includes detailed information for every loan issued by Lending Club from 2007 to 2015, including a borrower's annual incomes, zip codes, revolving balances, and purpose for borrowing. We will train and test a range of models in an attempt to identify the best performing model.

II. Related Work

Prior projects like "Predicting borrowers chance of defaulting on credit loans" [1] have set great examples of applying machine learning to improve loan default prediction in a Kaggle competition, and authors for "Predicting Probability of Loan Default" [2] have shown that Random Forest appeared to be the best performing model on the Kaggle data. However, despite of the early success using Random Forest for default prediction, real-world records often behaves differently from curated data, and a later study "Peer Lending Risk Predictor" [3] presented that a modified Logistic Regression model could outperform SVM, Naive Bayes, and even Random Forest on Lending Club data. The fact that Logistic Regression performance could be immensely improved by simply adding penalty factor on misclassification gave rise to our interest in fine tuning other not-yet optimized models, in particular, SVM and Naive Bayes, to continue the search for a better predictive model in the realm of loan default. Besides the difference in types of model that they focus on, the prior studies only used the out-of-the-box dataset from Kaggle or Lending Club, but research like "The sensitivity of the loss given default rate to systematic risk" [4] has shown the linkage between default rate and macroeconomic factors, so we have decided to add in census data, with info like regional median income, to train our models on a more holistic set of features.

III. Data

The raw Lending Club data contains 60 fields for each loan originated. However, not all of the fields are intuitively useful for our learning models, such as the loan ID and the month the last payment was received, and thus we removed such fields. We also removed fields for which greater than 10% of the loans were missing data for. Categorical features, such as address state (for example, California), were expanded into boolean columns, one column for each distinct value that the feature could take. Finally, we removed any loans that were missing data for any field (around 3% of the loans in our dataset).

To label the dataset, we classified any loan that defaulted, were charged off, or were late on payments was classified as negative examples, while we classified any loan that was fully paid or current was classified as positive examples.

III.I. Feature expansion

The Lending Club data contains a few fields which are not immediately usable in learning models, namely zip code and loan description. Since there are many zip codes, expanding them into boolean columns as we did for categorical features will not be effective. Instead, we joined census data with the Lending Club data. The description of the loan contains freeform text input by the loan requestor, and thus may contain keywords that correlate with defaulting or non-defaulting loans.

For census data by zip code, we use the "Median Household Income and Mean Household Income [2006-2010]"

dataset [5]. The dataset contains mean income, median income, and population by individual zip code. The Lending Club data contains the first three digits of the zip code for each loan. Thus, we calculated a population-weighted mean and median for each three digit zip code.

For loan descriptions, we use TF-IDF. TF-IDF allows us to determine any important words in each loan description while taking into account the number of times a word appears in the corpus so that words that occur frequently in general are weighted lower.

$$tf(t,d) = f_{t,d}$$
$$idf(t,D) = log\frac{N}{|\{d \in D : t \in d\}|}$$
$$tfidf(i,d,D) = tf(t,d) * idf(t,D)$$

$tf(t,d)$, or term frequency, is equal to the number of times term t occurs in document d. $idf(t,D)$, or inverse document frequency, is equal to the log of the total number of documents divided by the number of documents d in the set of all documents D that contains the term t. The TF-IDF score is the product of the term frequency and the inverse document frequency.

Before running TF-IDF on the loan descriptions, we removed any punctuation and html tags, tokenized and stemmed the text using the Porter stemming algorithm, then removed any English stop words. We split the corpus into two groups, one for defaulting loans and one for non-defaulting loans. For each group, we determined the unique words across all descriptions, calculated the TF-IDF score for each word for every document, then summed up the scores. Since we are looking for terms that occur in either defaulting or non-defaulting loans but not both, we normalized the TF-IDF scores for each group and chose words with the highest absolute difference in the normalized TF-IDF scores between the two groups. The top 3 words were "bills", "business", and "card". We chose the top 20 words, then created binary features for each of the words indicating whether that word is present in the loan's description.

IV. Methods

Given an imbalanced dataset, such as the Lending Club dataset where the rate of positive examples is about 85%, accuracy does not indicate the true performance of the model. The accuracy will depend on the overall default rate of the test data set. For example, in this case, a model that predicts any example to be non-defaulting would still achieve 85% accuracy. Furthermore, some models will bias toward classifications that occur more often, such as SVM and Logistic Regression. Instead, we will look at sensitivity, defined as:

$$\text{sensitivity} = \frac{TP}{TP + FN}$$

where TP is the number of true positives and FN is the number of false negatives. Sensitivity is the fraction of loans that are actually positive (non-default) that were predicted as positive by the model. Since the more positive-skewed the dataset the higher the sensitivity, we also need to look at specificity, defined as:

$$\text{specificity} = \frac{TN}{TN + FP}$$

where TN is the number of true negatives and FP is the number of false positives. Specificity is the fraction of loans that are actually negative (default) that were predicted as negative by the model. In order to combine both sensitivity and specificity, we will use the G-mean [6]:

$$G = \sqrt{\text{sensitivity} * \text{specificity}}$$

Also, for completion of performance metrics, we also looked at accuracy and precision:

$$\text{accuracy} = \frac{TN + TP}{N}$$

$$\text{precision} = \frac{TP}{TP + FP}$$

To establish performance, we train each model with the first 70% of the loans and test the trained models on the last 30% of the loans in our dataset.

After selecting the best predicting model based on specificity performance to optimize the detection of high-risk loans, we will calculate return on investment (ROI) by:

$$\text{ROI} = \frac{\text{Total payment received by investors}}{\text{Total amount committed by investors}} - 1$$

V. Logistic Regression

We try modeling with logistic regression with Newtonś method to learn more about the data features and get the basic performance of our prediction. To first get boundaries of iterations needed for Newton as well as understand predictive contribution from each data features, we trial-trained with a logistic classification on all features.

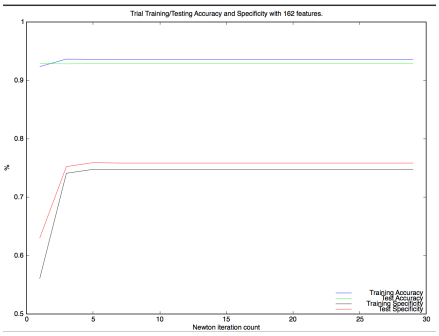
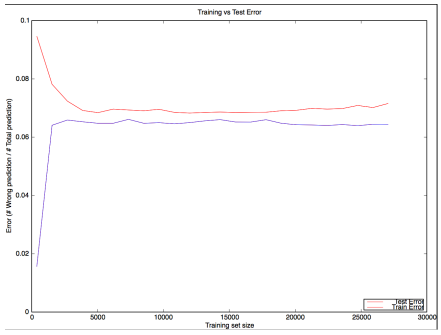


Figure 1: *Training vs. test Accuracy/Specificity by Newton iterations*

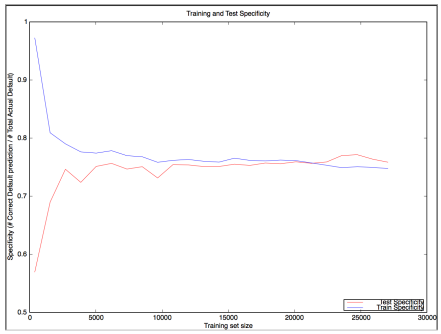
The training and test converges to an optimal solution within 5 iterations, and overall we reached a test accuracy of 92.9% and a test specificity of 75.8%.

V.I. Bias vs. variance

To see how the logistic model can be further improved, we ran a diagnostic by different sample size:



(a) *Training vs. test errors*



(b) *Training vs. test specificity*

Figure 2: *Stats by training sample size*

The test and training error converged quickly with a sample size >= 5,000, and we see that we may have a high bias problem as increasing sample size still resulted in a >5% test error. From the sensitivity chart, however, we see that sensitivity fluctuates with additional sample size, suggesting that the default prediction might potentially benefit from filtering on existing features even though test error has stabilized.

V.II. Feature Selection

Using ablative analysis on the logistic model, we found that for prediction on default rate (specificity), the credit score for the borrower is the most predicative of all features, followed by borrower population; while interest rate has negative impact as the number was subject to sporadic adjustment from Lending Club, and fields like loan description or borrower's lower fico range, where there are a lot of zero values, would worsen the default prediction. After we filtered out features that decreased our test specificity, such as *last_fico_range_low* , *installment*, *open_acc*, *desc*, *int_rate*, we managed to bump specificity from 75.8% to 77.1% without hurting overall test accuracy:

Table 1: *Performance of Logistic Model with feature selection*

Num Newton	Accu	Prec	Sens	Spec	G-mean
30 Iterations	92.8	96.6	95.1	**77.1**	85.7

VI. Naive Bayes

We used a Laplace smoothing factor of 1 and ran Naive Bayes using Gaussian, Bernoulli, and Multinomial probability distributions. For Bernoulli Naive Bayes, where we require boolean feature values, we binarized the features and values. Multinomial probability distributions take discrete feature values, so we rounded any decimal feature values to the closest integers.

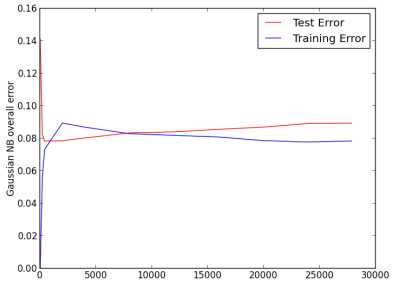
Table 2: *Performance of various Naive Bayes models*

Distribution	Accu	Prec	Sens	Spec	G-mean
Gaussian	91.1	96.6	92.7	**80.4**	**86.3**
Bernoulli	88.5	91.0	96.2	36.9	59.6
Multinomial	61.9	88.7	64.4	45.8	54.3

Gaussian Naive Bayes returns the most desirable performance on the test dataset. Bernoulli and Multinomial significantly underperformed Gaussian Naive Bayes.

VI.I. Bias vs. variance

In order to determine whether we are seeing high bias or high variance, we compare the training error to the test error for each case of Naive Bayes.



(a) *Gaussian NB training vs. test errors*

In an effort to reduce the error rate for Gaussian Naive Bayes, an additional feature of median income by zip code and descriptions was included in the dataset. The test error for Gaussian Naive Bayes decreased from 8.94% to 8.64% (by 0.3%). However, specificity decreased from 80.4% to 80.1% with the external dataset included.

Table 3: *Performance of Gaussian Naive Bayes with external dataset*

Distribution	Accu	Prec	Sens	Spec	G-mean
Gaussian	91.3	96.9	93.0	**80.1**	**86.3**

VII. Support Vector Machine

Since the training data is likely not linearly separable, and not guaranteed to be separable even in higher-dimensional feature spaces, we will use L1 regularization (soft margin SVM). For training data points $(x^{(i)}, y^{(i)})$, the model is the result of the optimization:

$$min_{\gamma,w,b} \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i$$
$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, ..., m$$

We first normalize the features by scaling the values of each feature to [-1, 1] using the same scaling factor for both the training and test data. This is necessary to prevent features with greater absolute numeric values to dominate those with smaller numeric values. Also, since the kernel values typically involve the inner products of feature vectors, normalizing the values prevents numeric problems such as overflows [7].

The performance of an SVM model depends on the kernel used, the parameters of the kernel, and the soft margin parameter C. We will attempt to optimize each of these.

VII.I. Selection of kernel

We investigate some commonly used kernels (linear, polynomial, Gaussian radial basis function, and sigmoid) and compare performance. We used LibSVM [8] with default settings (C-SVC, $C = 1$, $\gamma = 1/\#$ of features, $d = 3$), and trained the model with the first 70% of the loans and tested the models on the last 30% of the loans in our dataset.

$$\text{Linear: } K(x,z) = x^T z$$
$$\text{Polynomial: } K(x,z) = (\gamma(x^T z + 1))^d$$
$$\text{RBF: } K(x,z) = e^{-\gamma||x-z||^2}$$
$$\text{Sigmoid: } K(x,z) = tanh(\gamma x^T z + d)$$

Table 4: *Performance of SVM with various kernels*

Kernel	Accu	Prec	Sens	Spec	G-mean
Linear	**93.0**	**96.6**	95.3	**77.7**	**86.1**
Polynomial	**93.0**	94.5	97.6	61.8	77.7
RBF	92.8	93.9	98.0	57.5	75.1
Sigmoid	90.8	91.8	**98.2**	41.0	63.5

The baseline model is one which merely predicts every loan to be non-defaulting and will achieve an accuracy of 87.0%, which is the fraction of test data that are actually positive, so we see that SVM improves predictions.

VII.II. Bias vs. variance

We run SVM with a linear kernel with variable number of training examples, then compare the training error with the test error to determine whether we are likely to be encountering high bias or high variance in our SVM model with the dataset that we have.

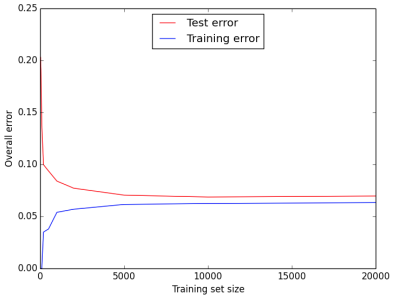


Figure 4: *Training vs. test error for SVM for various training set sizes*

The test and training errors converge quickly relative to the number of training examples available, and the gap between them is small, suggesting a high bias in the model. Thus we will increase the number of features by expanding zip code into census data, as well as identifying important words in the loan title and description.

Adding median income, mean income, and population fields extrapolated from the zip code, we see a minute 0.1% increase in precision and 0.2% increase in specificity, with all other performance metrics remaining the same. Adding the words selected via TF-IDF as boolean features, we see an increase across all performance metrics, including a 0.6% boost in G-mean.

VII.III. Soft margin parameter

We experimented with different values of the soft margin parameter C. We ran linear kernel SVM with $C = \{10^{-6}, 10^{-4}, 10^{-2}, 1, 10^2\}$. Ultimately, we found that using $C = 1$ yielded the best performance.

Table 5: *Performance of SVM with various values of C*

C	Accu	Prec	Sens	Spec	G-mean
10^{-6}	87.0	87.0	**100.0**	0.0	0.0
10^{-4}	87.0	87.0	**100.0**	0.0	0.0
10^{-2}	92.7	93.8	98.1	56.8	74.6
1	**93.7**	**96.9**	96.3	**78.0**	**86.7**
10^2	93.0	96.7	95.3	78.0	86.3

Table 6: *Performance of SVM with various optimizations*

Step	Accu	Prec	Sens	Spec	G-mean
Linear kernel	93.0	96.6	95.3	77.7	86.1
Add features	93.7	96.9	96.3	78.0	86.7
Tune C	93.7	96.9	96.3	78.0	86.7

VIII. Results

From the comparison of each model, we found that Naive Bayes with Gaussian performs the best with default prediction (80.1% sensitivity). We speculate that Naive Bayes with Gaussian is slightly better than the other models for 2 potential reasons:

- Naive Bayes model works well with independent feature sets [9], and the training features that we selected are possibly either independent or have evenly distributed dependencies.
- Some of the key features that we used, like credit scores and regional population, might be distributed in Gaussian, which would allow the Gaussian-assumption model to perform better.

To get the final improvement on the return, we will calculate ROI as mentioned in Method section. For the current return from Lending Club, the overall return received (including interest and late fee earned and principal recovered) by all investors divided by their initial principal is 10.1%. And If we apply the model prediction to avoid investing on the predicted-to-default note and calculate the return based on only predicted-positive loans, we can increase the return of investment of the test set from 10.1% to over **15.7%** (50% growth).

IX. Conclusion

From the comparison of multiple models, including Logistic Regression, SVM, and Naive Bayes and different fine-tuning mechanisms, we found that Naive Bayes with Guassian performs the best at predicting default rate (optimizing specificity). And by applying our best-performing model, we saw that the investment return from Lending Club can potentially grow by 50%. Some future work that could further improve the prediction includes:

- SVM has a degrading performance on highly imbalanced datasets [10], so we should experiment with balancing the datasets as best as possible to improve SVM prediction.
- We see high-bias problems across all models, so the predictions could benefit from more features, such as stock market or housing trend, so as to include more macroeconomic factors into the models.
- Not all default cases are the same, and some late payments could still be recovered later on, so instead of a binary classification, multinomial predictions could be employed to take into account the different types of default so as to make a more granular prediction.

References

[1] Liang, Junjie. "Predicting borrowers' chance of defaulting on credit loans."

[2] Pandey, Jitendra Nath. "Predicting Probability of Loan Default Stanford University, CS229 Project report Jitendra Nath Pandey, Maheshwaran Srinivasan."

[3] Tsai, Kevin, Sivagami Ramiah, and Sudhanshu Singh. "Peer Lending Risk Predictor."

[4] Caselli, Stefano, Stefano Gatti, and Francesca Querci. "The sensitivity of the loss given default rate to systematic risk: new empirical evidence on bank loans." Journal of Financial Services Research 34.1 (2008): 1-34.

[5] University of Michigan Population Studies Center, Institute for Social Research. Zip code characteristics: mean and median household income. Available at: http://www.psc.isr.umich.edu/dis/census/Features /tract2zip/index.html. Accessed December 1, 2015.

[6] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in Proceedings of the Fourteenth International Conference on Machine Learning. Morgan Kaufmann, 1997, pp. 179-186.

[7] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, A Practical Guide to Support Vector Classification, 2010. Available at: https://www.csie.ntu.edu.tw/ cjlin/papers/guide /guide.pdf

[8] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at: http://www.csie.ntu.edu.tw/ cjlin/libsvm

[9] Zhang, Harry. "The optimality of naive Bayes." AA 1.2 (2004): 3.

[10] He, He, and Ali Ghodsi. "Rare class classification by support vector machine." Pattern Recognition (ICPR), 2010 20th International Conference on. IEEE, 2010.