Predicting and simulating loan defaults and losses using neural networks

Liang Ping Koh
Stanford University
lpkoh@stanford.edu

Anika Mohindra
Stanford University
anikamo@stanford.edu

Diana Kim
Stanford University
dahea@stanford.edu

Murilo Moura
Stanford University
mmoura@stanford.edu

Wilhelm Bolin
Stanford University
wbolin@stanford.edu

Abstract

Understanding lending risk is essential for the flourishing of the credit industry, which serves as a tool of economic empowerment and growth. To this end, the project aims to use neural networks to predict if a loan will default, and given that it does, what the expected loss amount would be. The neural networks are trained on a dataset of roughly 150,000 equipment loans backed by the US Small Business Administration (SBA) between 1990 and 2014. The predictions are then used to simulate possible loss outcomes on a random portfolio of 500 loans.

1 Introduction

The credit industry enables economic activity and fuels economic growth, and is essential to the modern economy. However, lenders will not want to lend money when they do not understand risk; hence, being able to quantify the chance and loss of default is key for creditors to be willing to make loans. Traditional approaches may have relied on heuristic reasoning, coupled with simpler mathematical models, such as linear regression, to assess such risks. It may, however, be difficult to capture such complex phenomena with simple models, and the computation can often be laborious. The deluge of modern data and the development of machine learning techniques offers new techniques for understanding credit risk.

In this paper, we use two conventional, feed forward, fully connected neural networks to firstly, predict the probability of a loan defaulting in 1 year and 5 year's time, and secondly, predict the expected loss in the case of default. We will train these neural networks on a data set of 150,000 equipment loans backed by the US Small Business Administration (SBA) between 1990 and 2014. The input to our model will include categorical and numerical features from the data set, such as the state that the borrower of the loan is from and how much was borrowed, as well as features we have added ourselves, such as Housing Price Index data, age of the loan, etc.

Having built these two neural networks, we will randomly sample 500 loans, and then simulate several events of default/non default, and together with predicted loss, we will be able to simulate loss events on the portfolio of 500 loans, obtaining a distribution of total loss. This distribution of total loss will also be interpreted in the context of a 5 and 15 percent tranche.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

2.1 Deep Learning for Mortgage Risk

The work has developed a deep learning model of multi-period mortgage risk in order to to analyze an unseen dataset of origination and monthly performance records for mortgages across the United States between 1995 and 2014. The analysis of the paper is based on a nonlinear deep learning model of multi-period borrower state transitions that incorporates the influence on borrower behavior of a large number of loan and borrower-specific as well as economic and demographic variables at national, state, county and zip-code levels. We took note of the features used and found them useful in guiding our decisions of what to include.

2.2 Bankruptcy Prediction for Credit Risk Using Neural Networks: A Survey and New Results

The authors of the paper improved their accuracy in predicting corporate bankruptcies from 81.46% to 85.5% (out of sample) for their neural net system by engineering novel new features, such as stock price volatility and rate of change of cash flow per share. Our takeaway was to try to engineer our own features to assist in the prediction task.

2.3 A Neural Network Approach for Credit Risk Evaluation.

Using data from a bank in Italy, the authors used a feed-forward neural network with two hidden layers to predict whether a small business loan will default. This paper's network architecture, with two hidden layers, was able to achieve a test accuracy of 91.4%, prompting us to consider experimenting with different numbers of layers for our model. They developed an unconventional ad hoc network which we can consider as a future direction for the project.

2.4 "Should This Loan be Approved or Denied?": A Large Dataset with Class Assignment Guidelines

The authors used logistic regression to assess the risk of a loan and decide whether or not to make a loan. The dataset used was also from the U.S. Small Business Administration (SBA), making it particularly relevant for our task. We used it's results as a reference as to what features were promising to try in the training of our model.

3 Dataset and Features

3.1 Preliminary Data Analysis

The original data set contained 147,424 equipment loans backed by the US Small Business Administration (SBA) between 1990 and 2014. The data consisted of 30 covariates (features) describing the type of loan, amount and other information related to the issuing of the loan (see appendix for full list of features). Features consisted of both categorical and numerical types.

3.2 Data Cleaning

Before modeling, we had to clean the data. First, we removed variables that we believed to have little/limited predictive value. These variables are *Program*, *BorrName*, *BorrStreet*, *BorrZip*, *CDC_Name*, *CDC_Street*, *CDC_Zip*, *ThirdPartyLender_Name*, *subpgmdesc*, *DeliveryMethod* and *ProjectCounty*. These features were either simply irrelevant, such as name, or had too high a level of granularity to have predictive value, such as zip codes, or had identical values for every row, such as Delivery Method.

Second, we removed variables that had redundant information. For instance, *NaicsDescription* and *NaicsCode* had similar information describing the industry, so we removed *NaicsDescription*.

Third, we removed the features that were missing more than 30% of their values, as performance is closely tied to availability of data, and we did not want to include any variables that might have been useful, but due to the large number of null data points, would lead to a restricted dataset. Removed features include *ThirdPartyLender_City*, *ThirdPartyLender_State*, *ThirdPartyDollars* and *InitialInterestRate*.

Fourth, we had to deal with rows with a tolerable number of missing values. We took a different approach for quantitative variables and for categorical variables. For quantitative variables, we replaced missing values with "0." For categorical variables, we replaced missing values with "BLANK." That is, we treated "BLANK" as another category within the variable.

3.3 External Variables

In order to enhance our model, we added external explanatory variables.

The first variable we added is *UnemploymentRateBorrState*. This variable shows the unemployment rate in the year that the loan was granted and in the state that the borrower is located, for each of the loans. In *Deep Learning for Mortgage Risk*, state unemployment rates are found to "have the greatest explanatory power for borrower behavior among all variables studied." Moreover, "the delinquency rate increases with state unemployment." The paper also notes that "the important role of unemployment implies that the loan-to-loan correlation due to the exposure of borrowers to economic cycles can be substantial."

The second variable we added is *UnemploymentRateProjectState*. This variable also indicates the unemployment rate in the year that the loan was granted, but for the project state of each loan.

The third variable we added is *HPI-Borrower*, which gives the house price index in the borrower's state by year.

The fourth variable we added is *LogSP500*. This value is calculated as the log of daily return of S&P 500 based on the loan approval date. S&P 500 is an indicator of general market health and indicates the impact of economy on the loan default rate.

3.4 Exploring the Relationship between Variables

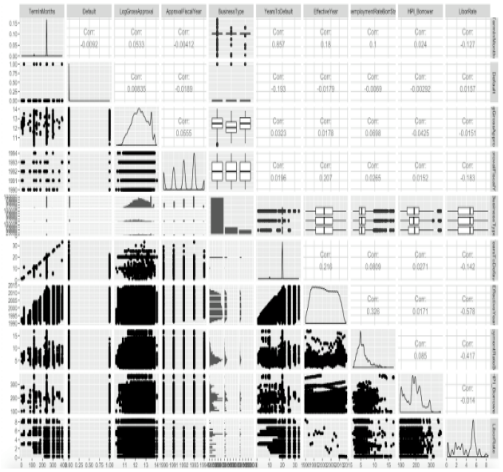

Figure 1: Analysis of data

We conducted preliminary data analysis primarily using R in order to better understand the relationship between different variables in the context of loan default probability.

We looked at the locations of the borrower and the lender. We observed that based on p-value the location of a borrower had a statistical significance. Mississippi, Georgia, and Florida had the highest probability of default. As mentioned previously, we added other state-level data such as HPI and unemployment rate that could more accurately capture state nuances. We also observed that loans with third party lenders have a higher probability of default, indicated by a positive coefficient of 1.26484. This makes sense given that people with low credit scores are more likely to need to borrow from third party lenders.

Another interesting observation was regarding approval fiscal year. We observed that loans issued between 2000 and 2008 had relatively higher probability of default. 2006, 2007, and 2008 especially

had the largest coefficients of 1.60311, 1.66008, and 1.41135 respectively. This shows that loans issued close to or during the financial crisis of 2008 have a higher probability of default.

We also conducted analysis with our external variables. All of our external variables: state unemployment rate, HPI of borrower, S&P 500, and LIBOR rate - seemed to have great statistical significance with small p-values. Unemployment rate was positively correlated with probability of default, as expected. Higher unemployment rate means a lower average income and people would have more difficulty to pay back the loans. HPI of the borrower and S&P 500 seem to have a negative correlation with the probability of default. This made sense because if HPI is high, a borrower would have more financial capacity to pay back and loan is less likely to default. If S&P 500 price index is high, it means the market is relatively strong, and the loan is less likely to default as well.

Finally, we observed that gross loan approval amount has a significant negative correlation with the default probability. This observation was in line with our learning from the class that borrowers with small loans often have the largest default rates.

3.5 Final processing

Our first prediction task is to predict whether a loan will default, in a year, and whether it will default in 5 years. We took our dataset, with the appropriate features removed and added, and expanded the dataset by entering an entry for every year a loan was active. For example, a loan active from 2004-2007 will have 4 entries in the new matrix. We then added a new column, saying whether the data will default in a year's (or five year's) time or not. For example, the previously mentioned loan, which say, defaults in 2007, will have label 1 for 2007 entry and 0 for the rest in the 1 year ahead dataset, and 1 for every year in the 5 year data set. When our neural net trains on this dataset, it will then train on a snapshot of a loan at annual frames in its lifetime, and predict whether the loan will default in a year's or five year's time based on the dataset we are working with. We have 1,492,371 entries in our datasets for 1 and 5 year default prediction. The dataset, however, is extremely unbalanced, with far fewer defaults than non defaults, and we re-balanced the dataset by splitting into default and non default rows, then sampling with replacement from each pool to generate a combined pool of 400,000 rows, half of which is default and half of which is non default. We then did a train-test split of 98-2.

Our second prediction task is to predict the amount lost when a loan defaults. For this, we restrict ourselves to look at the default cases. These are extremely limited. We have 7591 entries. We did a train-test split of 90-10.

We also added a variable to the data called *LoanYear*, which gives the effective age of the loan in years since issuance.

Finally, we performed a one-hot encoding on our categorical variables and normalized the numerical ones.

4 Model/Results/Discussion

4.1 Model overview

For our default prediction task, we took the two processed data sets (1 year look ahead and 5 year look ahead), and trained them using a baseline linear regression model, and a 4 layer neural network (non linear regression model). Our models had an input layer of 114 neurons, intermediate 'relu' activation, standard initialization, L2 regularization. We used an Adam optimizer, with a binary cross entropy loss function which is suitable for a binary classification task.

For our loss prediction task, we trained a 4 layer model on the dataset of only defaults. Our models had an input layer of 120 neurons, intermediate 'relu' activation, standard initialization, L2 regularization. We used an Adam optimizer, with a mean absolute error loss function which is suitable for predicting a quantity.

Here is the code for our models:

```
### baseline default model
baseline_default_model = Sequential()
baseline_default_model.add(Dense(1, input_dim = 114, kernel_initializer='normal', bias_initializer = 'zeros',
                                 kernel_regularizer = regularizers.l2(0.01)))

# Compile model
baseline_default_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

#train model
baseline_default_model.fit(X_trainsampled, y_trainsampled, epochs=20, batch_size=128, verbose = 1)
```

Figure 2: Baseline Model for default

```
#multilayer default model
multilayer_default_model = Sequential()
multilayer_default_model.add(Dense(90, input_dim=114, activation = 'relu', kernel_initializer='normal',
                                   bias_initializer = 'zeros', kernel_regularizer = regularizers.l2(0.005)))
multilayer_default_model.add(Dense(60, input_dim=90, activation = 'relu', kernel_initializer='normal',
                                   bias_initializer = 'zeros', kernel_regularizer = regularizers.l2(0.005)))
multilayer_default_model.add(Dense(25, input_dim=60, activation = 'relu', kernel_initializer='normal',
                                   bias_initializer = 'zeros', kernel_regularizer = regularizers.l2(0.005)))
multilayer_default_model.add(Dense(1, input_dim=25, kernel_initializer='normal',
                                   bias_initializer = 'zeros', kernel_regularizer = regularizers.l2(0.01)))

#Compile model
multilayer_default_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

#train model
multilayer_default_model.fit(X_trainsampled, y_trainsampled, epochs=100, batch_size=256, verbose = 1)
```

Figure 3: Multi-layer Model for default

```
#baseline loss model
baseline_loss_model = Sequential()
baseline_loss_model.add(Dense(100, input_dim= 120, activation = 'relu', kernel_initializer='normal', bias_initializer = 'zeros',
baseline_loss_model.add(Dense(80, input_dim=100, activation = 'relu', kernel_initializer='normal', bias_initializer = 'zeros', k
baseline_loss_model.add(Dense(20, input_dim=80, activation = 'relu', kernel_initializer='normal', bias_initializer = 'zeros', ke
baseline_loss_model.add(Dense(1, input_dim=20, activation = 'relu', kernel_initializer='normal', bias_initializer = 'zeros', ker

# Compile model
adam = keras.optimizers.Adam(lr=0.05, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.00001, amsgrad=False)
baseline_loss_model.compile(optimizer=adam, loss='mean_absolute_error', metrics = ['mae'])

#train model
baseline_loss_model.fit(X_train_loss, y_train_loss, epochs=100, batch_size=128, verbose = 1)
```

Figure 4: Baseline Model for loss

4.2 Feature overview

We experimented with which features to train on by including and excluding certain variables whose efficacy we wanted to test, and running training on a limited number of epochs to see how ROC would perform. We found that categorical variables with extremely high number of categories generally did not improve the model, even after an extensive training period, and often caused training times to be far longer. We have hence excluded *CDC_City*, *CDC_State* and *BorrCity*. We finally decided on these features, which we found to be most effective:

COLUMN_NUMERIC = [’GrossApproval’, ’TermInMonths’,’HPI_Borrower’, ’UnemploymentRate-BorrState’, ’LogSP500’, ’LiborRate’]

COLUMN_CATEGORICAL = [’BorrState’, ’ApprovalFiscalYear’, ’BusinessType’, ’LoanYear’, ’TPDBinary’]

We also included Approval Month for loss prediction task.

4.3 Hyperparameter tuning and final results

For the baseline model, we obtained an ROC of 0.72742 and 0.70056, for 1 year and 5 year dataset respectively. We then performed hyper parameter tuning on our model to try to improve our ROC score, with success. We did not change the type of regularization (L2), loss function (binary cross entropy), initialization method and optimizer (Adam), as we did not think this would be promising. What we did tune was the batch size (tried 64, 128, 256), the number of epochs (20, 50, 100), number of layers (2,4), and the number of neurons in the penultimate hidden layer, although these did not produce any significant changes. What was extremely important to our results, however, was the tuning of L2 regularization itself. Here are the results:

L2 regularization parameter	AUC
0.1	0.77357
0.01	0.80003
0.004	0.81906
0.002	0.83014
0.0005	0.82151

What we realized was that there was an optimum amount of L2 regularization, at 0.002, that gave us our best AUC, 0.82814. The interpretation of this is that there is noise in the large number of features that we have, and by having regularization, we prevent over-fitting to the noise. But at the same time, over regularizaton (0.0005) prevents us from adequately fitting to the data, lowering our AUC. Here is our ROC curve:

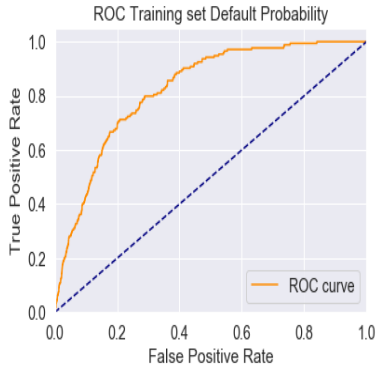

Figure 5: ROC Curve

As for the regression task, we found that with many epochs (up to 5000), relatively high starting learning rate (0.05) and learning rate decay, our model was able to converge indefinitely to an extremely low mean absolute error of 25264 on the training set, which is very significant considering the standard deviation on the dataset is 428290. However, when testing the actual error on the test set, we found that our error was 103805, telling us that what we were doing was over fitting to training data. So instead, we decided to employ regularization and early stopping, having an L2 parameter of 0.001 and only 50 epochs, but we were only able to bring the error down to 93578. We concluded, with TA's advice, that there were simply too few default data points for us to create a model that would generalize well.

5 Simulations

We randomly sampled 500 loans from the dataset, and then used our models to make predictions for chance of default in 1 year, chance of default in 5 years, and loss given default. Here are our distributions:

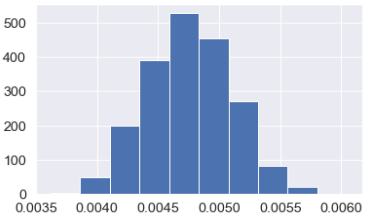

Figure 6: 1 year total loss

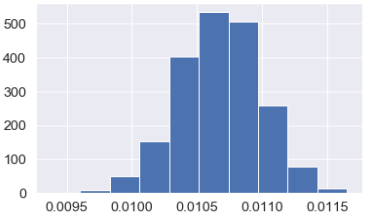

Figure 7: 5 year total loss

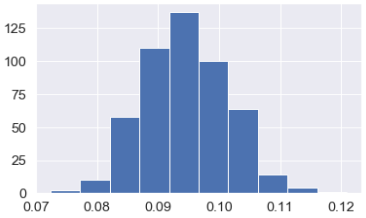

Figure 8: 1 year 0-5 tranche loss

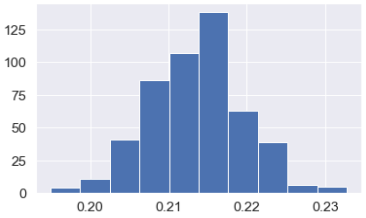

Figure 9: 5 year 0-5 tranche loss

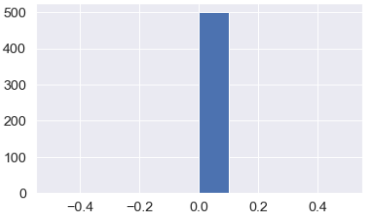

Figure 10: 1 year 5-15 tranche loss

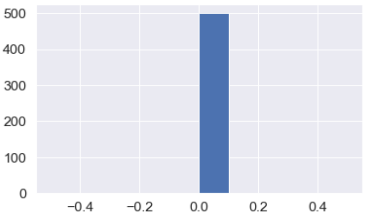

Figure 11: 5 year 5-15 tranche loss

VAR scores are as shown below:

Years	Percentile	VAR scores	Average VAR scores (95% C.I)
1	95	0.005336037353016124	0.00532556 +- 0.000145367
1	99	0.005568057474949305	0.00569943 +- 0.000123678
5	95	0.011187660573796223	0.0124356 +- 0.000118274
5	99	0.011391521000782345	0.0127482 +- 0.000112947

Figure for senior tranche will be the same as that for the 5-15 tranche. From a risk management perspective, you are at significant risk of losing a part of your portfolio if you are within the 0-5 tranche, especially over a 5 year horizon, and risk adverse investors should stay away from that. However, the 5-15 tranche offers a good degree of safety, and so does the senior tranche.

6 Conclusion/Future Work

In conclusion, we managed to build a reasonably effective model for default prediction in a year and five year's time, using standard approaches to training, but the most important hyper parameter to tune was the L2 regularization parameter, telling us that it was important to have just the right fit to the data for the model to generalize. For the loss prediction, there was a limit to what the regression could achieve, due to the lack of data.

Future directions that we could explore would be to gather more data, especially for default cases. Alternative network architectures could be considered for our task. For example, in Angelini et al., the neural net had input neurons being grouped into triplets and each triplet being connected to only one neuron in the first hidden layer, which led to better test set error (4.3% as opposed to 8.6%), suggesting that instead of fully connected layers, more restricted connections could perform better. Also, if we had textual description of the loan, we could use self trained or pre trained word embeddings to try and categorize loans by approximating qualitative information/sentiments about them through these models. Beyond just predicting the event of default, we could predict the time of default using more sophisticated models such as LSTM RNNs and hazard models, using a data input with time varying covariates.

7 Contributions

Liang Ping: Built and tuned both neural networks, generated simulations, major contributor to final report.
Anika: Built datasets for input into model, worked on and helped to debug the model and data, major contributor to final report.
Wilhelm Bolin: Helped find new features, Built datasets for input into model, worked on and helped to debug the model and data, contributed to final report.
Diana Kim: Kept track of group's progress as scribe, created presentation materials, helped find new features, performed data pre-analysis and visualization, contributed to final report.
Murilo: Helped find new features, performed data pre-analysis and visualization, contributed to final report.

References

[1] Angelini, Eliana, et al. (Nov. 2008) "A Neural Network Approach for Credit Risk Evaluation." *The Quarterly Review of Economics and Finance.*

[2] Min Li, Amy Mickel & Stanley Taylor (Apr. 2018) "Should This Loan be Approved or Denied?": A Large Dataset with Class Assignment Guidelines. *Journal of Statistics Education, Vol. 26, 2018, Issue 1*

[3] Atiya, Amir (Jul. 2001) "Bankruptcy Prediction for Credit Risk Using Neural Networks: A Survey and NewResults." *IEEE Transactions on Neural Networks.*

[4] Justin A. Sirignano, Apaar Sadhwani, Kay Giesecke (Sep. 2015) "Deep Learning for Mortgage Risk."

8 Appendix

Field Name	Definition
Program	Indicator of whether loan was approved under 7a/504 loan program
Name of borrower	Name of borrower
Borr Street	Borrower street address
BorrCity	Borrower city
BorrState	Borrower state
BorrZip	Borrower zip code
GrossApproval	Total loan amount
SBAGuaranteedApproval	Total loan amount
ApprovalDate	Date the loan was approved
ApprovalFiscalYear	Fiscal year the loan was approved
DeliveryMethod	"Specific delivery method loan was approved under. See SOP 50 10 5 for definitions and rules for each delivery method. 7(a) Delivery Methods: CA = Community Advantage CLP = Certified Lenders Program COMM EXPRS = Community Express (inactive) . DFP = Dealer Floor Plan (inactive) . DIRECT = Direct Loan (inactive) . EWCP = Export Working Capital Program . EXP CO GTY = Co-guaranty with Export-Import Bank (inactive) . EXPRES EXP = Export Express . GO LOANS = Gulf Opportunity Loan (inactive) . INTER TRDE = International Trade . OTH 7A = Other 7(a) Loan . PATRIOT EX = Patriot Express (inactive) . PLP = Preferred Lender Program . RLA = Rural Lender Advantage (inactive) . SBA EXPRES = SBA Express . SLA = Small Loan Advantage . USCAIP = US Community Adjustment and Investment Program . Y2K = Y2K Loan (inactive)"
subpgmdesc	Subprogram description - specific subprogram loan was approved under. See SOP 50 10 5 for definitions and rules for each subprogram.
InitialInterestRate	Initial interest rate - total interest rate (base rate plus spread) at time loan was approved
TermInMonths	Length of loan term
NaicsCode	North American Industry Classification System (NAICS) code
NaicsDescription	North American Industry Classification System (NAICS) description
FranchiseCode	Franchise Code
FranchiseName	Franchise Name (if applicable)
ProjectCounty	County where project occurs
ProjectState	State where project occurs
BusinessType	Borrower Business Type - Individual, Partnership, or Corporation
LoanStatus	"Current status of loan: PIF = Paid In Full, CHGOFF = Charged Off, CANCLD = Cancelled, EXEMPT = The status of loans that have been approved but have not been cancelled, paid in full, or charged off are exempt from disclosure under FOIA Exemption 4"
ChargeOffDate	Total loan balance charged off (includes guaranteed and non-guaranteed portion of loan)
RevolverStatus	Indicator of whether a loan is a term loan or revolving line of credit (0=Term, 1=Revolver)