Alec Engel
June 25, 2021

Introduction

The question at hand here is to determine what customer characteristics play a role in a home loan being a good or bad credit risk for the bank. We want to run several cross-validation models to see what variables can be strongly correlated with our response variable of "bad". These cross-validation methods will allow us to split our dataset into training, validation and test portions. The split allows for better predictions of future outcomes with reduction in potential overfitting and removing unnecessary model noise amongst highly correlated independent variables. We will next take a look at our variables using a benchmark model of nominal logistic, followed by elastic net, varying boosted neural network, random forest, and boosted tree models.

Predictor variables consist of loan details, potential customer's finance backgrounds and working history that may cause increase or decrease in the amount of riskiness to borrow a home equity loan. The most important variables in this dataset are the binary response variable of bad or good risk; along with customer predictor variables of debt-to-income percentage, reason for loan, the number of recent credit inquiries, number of delinquent trade lines, number of deregulatory reports, and type of job along with number of years working. As we will elaborate later, a customer's reason for the loan as well as their debt-to-income percentage will have highly unique information as it relates to the correlation of loan riskiness for the bank.

In order to narrow down our predictor variables, we ran several statistical models and judged our outcomes. Statistical methods include cross-validation tactics of nominal logistic as well as a generalized adaptive elastic net (EN), two random forests (RF), four customized boosted neural network models (NN), and two boosted tree models. The latter methods of boosted neural networks and boosted trees produced extremely powerful and telling models that create a solid narrative when dealing with our nonlinear data. Neural networks use hidden layer nodes to help model very complex variable relations between a model's inputs and outputs. Transformation is then applied and used to provide stronger model predictions. With neural networks, it is also important that we include a validation portion of our dataset as one drawback is that they can overfit if validation is not present. Each neural network performed here is also being boosted which allows a model to be built sequentially by collecting and correcting residual errors. By placing heavier weight on these errors as each iteration is run, we get a more accurate prediction and avoid making the same model mistakes twice. Boosted trees are similar in that they learn from past mistakes. Sequenced trees adapt and learn from past tree data by giving more weight to worst mistakes of a previous tree. This cuts out the probability of heavy tree

correlation and the result is a strong final prediction with the average of each boosted tree in the sequence.

Analysis and Model Comparison

We have first created a cross-validation column which splits our data into training (60%), validation (20%) and testing (20%) sets to allow our model to gain predictive attributes and score our area under curve (AUC) and misclassification rates for how we believe future data will look. As previously mentioned, this is an important step and validation must be included for our neural network models to tell the model when to stop running and prevent overfitting. We then created each model with both inclusion and exclusion of informative data. This inclusion of informative missing data helps strengthen our model's predictability and gives us some insights into why customers may leave specific variables blank. By including missing data, our predictive model estimates perform at higher rates as we will see below. Higher model performance is a result of our models filling in missing values with averages on continuous variables and creating a new level on categorical variables. With each model produced, we will take a look at predictor formulas as a dataset column so that we can easily perform model comparisons.

For the bank's good or bad credit risk binary response variable, we started with a nominal logistic model but quickly found that although assessing penalties on our variables, this model often times leads to overfitting or misuse of highly correlated variables. Adaptive elastic net was then performed to see if penalizing less on some specific variables would improve our model but it did little to do so. While using all variables, there are often noisy and unnecessary data variables that are accounted for. Our more advanced random forest model was then implemented and performed much better while taking a nonlinear approach versus the linear and more biased methods we first used. Our random forest eliminated tree correlation risk and emphasized better variables to use for future model predictability.

We then began with boosted models of boosted neural networks and boosted trees while taking several different transformation approaches. I felt that these boosted models would outperform the first four models we had produced. For neural networks, we first used one layer of transformation with TangentH(3), NLinear(3), NGaussian(3) while including 40 model runs and squared penalties. For comparison, we then used the same layers of transformation, but an absolute penalty and 50 layers. TangentH transformation function uses a similar "S" shape as a logarithmic function while Linear is similar to a linear regression model and Gaussian transformation function is similar to a normal (bell

shaped) distribution. Taking the squared value of our penalties for each model run in this dataset seemed to produce better predictability numbers than the absolute value of our errors. For boosted trees, we stick with JMP defaults but made sure to include multiple fits over split learning rates along with suppressing multithreading and adding our random seed 123 for reproducibility. This final boosted tree model produced 189 layers and 15 trees in the background. To get the clearest picture and better predictive abilities, we take the most stock in our boosted tree model. Although more complex, this advanced model is our best bet for accurately predicting the risk involved with our bank providing a customer with a home equity loan.

Model comparison results can be seen below. First with the training sets, we see all ten models having relatively high performance. All ten are above 0.8 AUC with small misclassification rates. These however are our training sub sets of data and we want to focus more towards our validation and testing metrics. Moving to validation and test notations, we see consistency with our training data but more strength given to our models with informative missing data included. Although having a slightly larger misclassification rate versus our neural network model, the boosted tree has a slight edge in AUC value which is our more desired calculation. Therefore, we will go with our more complex model of boosted tree that's AUC is nearly 95% and misclassification rate is only 0.08. Although I will dig into each model to investigate strongest column contribution variables, I want to choose our predominant and strongest model of boosted tree for predicting our binary dependent variable.

Measures of Fit for BAD

Validation	Creator	.2 .4 .6 .8	Entropy RSquare	Generalized RSquare	Mean -Log p	RASE	Mean Abs Dev	Misclassification Rate	N	AUC
Training	Fit Nominal Logistic		0.2249	0.2815	0.2397	0.2531	0.1302	0.0820	2012	0.8027
Training	Fit Generalized Adaptive Elastic Net		0.2249	0.2815	0.2397	0.2531	0.1303	0.0820	2012	0.8027
Training	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.6976	0.7961	0.154	0.2078	0.1088	0.0576	3576	0.9816
Training	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.7325	0.7899	0.0827	0.1432	0.0620	0.0263	2012	0.9946
Training	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(50)		0.7084	0.8045	0.1485	0.2023	0.1069	0.0537	3576	0.9839
Training	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(34)		0.7565	0.8102	0.0753	0.1374	0.0550	0.0244	2012	0.9935
Training	Bootstrap Forest		0.7596	0.8432	0.1224	0.1777	0.0984	0.0403	3576	0.9955
Training	Bootstrap Forest		0.6197	0.7325	0.1936	0.2290	0.1529	0.0677	3576	0.9884
Training	Boosted Tree		0.8443	0.9028	0.0793	0.1324	0.0642	0.0218	3576	0.9988
Training	Boosted Tree		0.3373	0.4550	0.3374	0.3110	0.2528	0.1342	3576	0.9160
Validation	Fit Nominal Logistic		0.2434	0.2930	0.1961	0.2276	0.1117	0.0663	679	0.8166
Validation	Fit Generalized Adaptive Elastic Net		0.2432	0.2928	0.1962	0.2277	0.1117	0.0663	679	0.8164
Validation	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.6205	0.7282	0.1839	0.2332	0.1196	0.0822	1192	0.9648
Validation	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.5762	0.6383	0.1099	0.1664	0.0678	0.0353	679	0.9508
Validation	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(50)		0.6144	0.7229	0.1868	0.2329	0.1201	0.0747	1192	0.9610
Validation	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(34)		0.5638	0.6265	0.1131	0.1700	0.0651	0.0368	679	0.9439
Validation	Bootstrap Forest		0.5488	0.6646	0.2186	0.2568	0.1370	0.0973	1192	0.9503
Validation	Bootstrap Forest		0.4043	0.5223	0.2886	0.2931	0.1946	0.1242	1192	0.9120
Validation	Boosted Tree		0.6020	0.7122	0.1928	0.2326	0.1087	0.0721	1192	0.9476
Validation	Boosted Tree		0.2878	0.3921	0.345	0.3134	0.2553	0.1233	1192	0.8723
Test	Fit Nominal Logistic		0.2127	0.2682	0.2473	0.2577	0.1310	0.0817	673	0.7861
Test	Fit Generalized Adaptive Elastic Net		0.2128	0.2683	0.2473	0.2577	0.1310	0.0817	673	0.7862
Test	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.5657	0.6803	0.2109	0.2459	0.1286	0.0797	1192	0.9451
Test	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(40)		0.4693	0.5474	0.1667	0.2071	0.0855	0.0475	673	0.9057
Test	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(50)		0.5374	0.6544	0.2247	0.2545	0.1335	0.0856	1192	0.9373
Test	Neural Model NTanH(3)NLinear(3)NGaussian(3)NBoost(34)		0.4329	0.5105	0.1782	0.2072	0.0803	0.0505	673	0.8834
Test	Bootstrap Forest		0.5023	0.6213	0.2417	0.2706	0.1486	0.1032	1192	0.9361
Test	Bootstrap Forest		0.4019	0.5201	0.2905	0.2918	0.1957	0.1124	1192	0.9075
Test	Boosted Tree		0.5815	0.6944	0.2032	0.2418	0.1144	0.0805	1192	0.9474
Test	Boosted Tree		0.2586	0.3575	0.36	0.3238	0.2613	0.1435	1192	0.8607

Interpretation

Per below, we can see that roughly six or seven column contribution variables were leaned on to maximize our overall model performance on predictive test data. These top parameter estimates in our boosted tree model are reason for loan, number of recent credit inquiries, number of delinquent trade lines, type and years on job and number of derogatory reports. We can see that in our column contributions not including informative missing data, debt to income percentage basically becomes our sole column contributor of predictability with nearly 98% explanatory power. This goes to show that when leaving loan value, reason for loan, job details and number of trade lines blank, we can still get a productive model using solely someone's debt to income ratio.

Focusing on our model including informative missing data however, there is obviously more weight tied between the reason for a loan, the number of recent credit inquiries and times delinquency. Per our model, a customer's loan reason will have explanatory power (including non-linear variable interactions) of over 30%. So, when determining a whether a customer will be a good or bad risk for providing a home equity loan. The first thing that needs to be deciphered is why they are requesting a loan and if they already have open delinquency lines of credit. This makes a lot of sense as a second mortgage to cover a person's debt may be much riskier than an initial loan to pay off a home. Also, someone one with open lines of delinquency would obviously be a bad risk. For these reasons, risk is better evaluated for potential bank loans.

Column Contributions

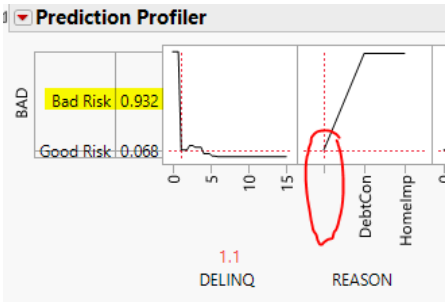
Term	Number of Splits	G^2	Portion
REASON	369	265737.113	0.3014
NINQ	491	115668.39	0.1312
DELINQ	412	100634.749	0.1142
YOJ	271	92497.0839	0.1049
DEROG	332	89832.7985	0.1019
JOB	293	83963.0059	0.0952
CLNO	226	77012.2133	0.0874
LOAN	90	23484.2404	0.0266
DEBTINC	90	12110.2979	0.0137
CLAGE	105	9623.50669	0.0109
VALUE	74	6508.60665	0.0074
MORTDUE	82	4500.13786	0.0051

Column Contributions

Term	Number of Splits	G^2	Portion
DEBTINC	768	346915.188	0.9770
DELINQ	9	5501.78801	0.0155
DEROG	3	1341.76135	0.0038
CLAGE	8	551.575886	0.0016
LOAN	4	386.265433	0.0011
MORTDUE	1	293.800394	0.0008
NINQ	2	82.5923298	0.0002
VALUE	0	0	0.0000
REASON	0	0	0.0000
JOB	0	0	0.0000
YOJ	0	0	0.0000
CLNO	0	0	0.0000

Reason and delinquency count will be our most important parameter estimates as they show both high estimate numbers as well as low standard errors. For reason, we see that as the classification of someone goes from manually created level (leaving blank) to debtcon/home loan, the risk simultaneously changes from bad risk to good risk. This is telling because potential customers that leave their reason for loan blank are much riskier than those that fill out the reason for obtaining their loan.

Our bank should not trust anyone asking for a loan who can't classify why they are asking for that home equity loan.



A similar notion can be used with a person's delinquency count. We see a steep change from good risk to bad once a customer has any greater than zero delinquent trade lines. Both of these relationships with bad loan risk help us to accurately predict the bank's level of avoiding bad risk loans. These factors all make complete sense along with debt-to-income percentage because they are all strong determining factors for whether the bank should trust or accept the risk that a customer will pay off their loan.