Final Project – Loan Classification

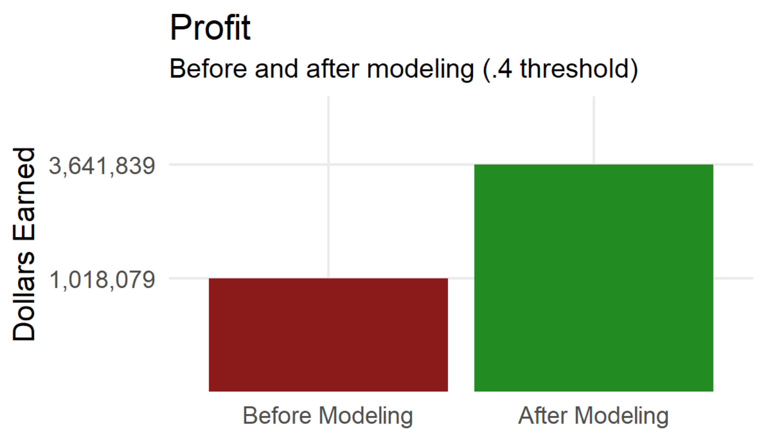
Deanna Schneider

April 4, 2018

Section 1 - Executive Summary

Banks need to constantly balance the risks and benefits in the lending business. Ideally, a bank would be able to predict loans that are most likely to end in default so that those loans could be denied. Banks tend to have a large amount of data about their customers and loans. The challenge is choosing the right data to use to most accurately predict loan defaults without unnecessarily denying loans that would succeed.

To that end, a model was developed that considered 18 factors, including such things as the grade of the loan, the loan term, and the time the customer has been continuously employed. Using these 18 factors, the model generates the probability that the loan will end in default and assigns a score between zero (worst) and one (best). The model can be tuned by picking a threshold level between zero and one, granting only those loans which exceed the threshold. As the threshold increases, more loans will be denied. Various thresholds were evaluated with a test data set that contained 21.9% bad loans. The threshold of .4 resulted in an increase in profits of $2,623,760 compared to the test data results that included no modeling.



Recommendations

The following steps are recommended:

-- Provide the data scientist with the most up-to-date, accurate data possible. There were multiple instances in the sample data that seemed suspicious. Working with the data scientist to validate all provided data could result in a more accurate model. Barring that:
-- Implement the suggested model with a threshold of .4. Carefully track the results of loans over the next one to two years. Review the accuracy of the model against any early defaults.
-- Increase the threshold slightly if too many defaulting loans are slipping through. In testing, the threshold could be increased to .45 before profits started to decrease.
-- If too many loans are being denied, decrease the threshold slightly. In testing, the .4 threshold resulted in 1,059 loans turned down in error, despite the increase in overall profit. Thresholds as low as .25 still resulted in increased profit of $1,441,031.40, with fewer denied loans (494), though the profit was

significantly less than at the recommended threshold.
-- After 3-5 years, remodel the data with the newest information about customers. As the bank denies more bad loans, the model may need to be shifted to fine-tune the predictors.

Section 2 - Introduction

Given a dataset of 50,000 loans (https://datascienceuwl.github.io/Project2018/TheData.html), data science methods will be used to build a logistic regression model to predict loans that will end in default. Exploratory data analysis will be used to determine relevant predictors. Multiple models will be fitted and tuned to improve the model diagnostics. Models will be evaluated for goodness of fit and predictive capability. Ultimately, the goal is to find a model that maximizes the profit for the bank, while minimizing the number of loans that will be rejected unnecessarily.

Section 3 - Preparing and Exploring the Data

Any good data science project begins with exploring the data - looking for inaccuracies, redundancies, potential predictor variables, variables that can be removed due to lack of predictive power, and determining an appropriate response variable. Each step, and the reason for the step, is articulated in this section.

Data Cleanup - Step One

For logistic regression, you need a response variable that has two states - the positive state and the negative state. In this model, the status of the loan is the response variable. However, the data set contains loans in progress. The first step is to set the positive ("Fully Paid") and negative ("Charged Off" or "Default") status and remove the rows that are in neither of those two states. This is an important first step so that additional decisions can be made based on just the applicable rows of data.

```r
#load libraries and data
pacman::p_load(caret,corrplot,pscl,purrr,tidyr,ggplot2,ROCR,ggcorrplot,ResourceSelection,gridExtra,rJava,glmulti,HH, pa
nder)
loans <- read.csv('loans50k.csv')

#Function to update the status of the loan
setStatus <- function(x){
  if(x[12] == "Default" | x[12] == "Charged Off") "Bad"
  else if (x[12] == "Fully Paid") "Good"
  else "Drop"
}
#apply the function
loans$status <- apply(loans, 1, setStatus)
loans <- subset(loans, status != "Drop")
```

Data Cleanup - Step Two - Dropping Unnecessary Columns

Some variables can be eliminated because they would not reasonably add predictive power:

Employment is a factor with 15853 levels. About half of the observations would have unique values in this factor. It is safe to assume it does not provide any meaningful predictive value.
State is a factor with 49 levels. While it is possible that there is a higher default rate in certain states, it does not make any sense from a business perspective to use state as a predictive variable. If this model is to be used at a place-based business, the likelihood that most applicants would come from a single state is

quite high.
"loadID" is a row ID and adds no meaning.

```
drops <- c("employment",  "loadID", "state")
loans <- loans[ , !(names(loans) %in% drops)]
```

Data Cleanup - Step Three - Dealing with Missing Values

Null values are always an issue when doing any kind of statistical analysis. Three parameters contain null values - revolRatio (15 rows), bcOpen (360 rows), and bcRatio (384 rows). Inspection reveals that there is extensive overlap in the columns that have missing values, meaning one of the key tenants of imputation (data must be missing at random) seems to be violated. Most of the missing data seems to be directly related to a credit card limit of zero, and the missing data in the bcOpen column has a 97% correlation with the missing data in the bcRatio column.

When the total credit limits on credit cards is zero, logically the total unused credit on credit cards should also be zero. Therefore, a rational approach to updating the missing data in bcOpen can be undertaken.

This approach does not account for bcRatio, a derived field based on other credit card limit and usage fields. The bcRatio variable is significantly correlated with bcOpen (and other columns). Given the number of variables we already have, dropping this derived data by column seems like a reasonable approach.

This leaves just 15 rows with missing values. This is less than half a percent of the overall dataset. A loss of that amount of data is an acceptable loss. Those rows will be deleted.

```
#rational approach to updating bcOpen
loans$bcOpen[is.na(loans$bcOpen) & loans$totalBcLim == 0] <- 0
#drop the bcRatio column
loans <- loans[ , !(names(loans) %in% c("bcRatio"))]
#drop remaining rows with nulls
complete_loans <- loans[complete.cases(loans), ]
```

Data Cleanup - Step Four - Consolidating Factors

Factor consolidation can reduce the overall number of predictors, and eliminate groups with too few observations.

Reason has a single large category (debt consolidation) and several small categories. All the small categories will be combined as an "other" category. (Note: a model with less consolidation of the reason category was made based on the reasons that appeared to have the most significance outside of debt consolidation - small_business and renewable_energy. But, that model had reduced accuracy.)
Verified has two categories that appear to have similar meanings (verified and source verified). They will be combined.
Length is the time continuously employed and has 12 levels, the largest of which has over 11000 observations. All the other levels are much smaller. Combining these into three groups (under 5 years, 5 to 10 years, 10+ years) will provide a more reasonable number of factors. This variable also has an n/a value, which one can assume means "unemployed" and is meaningful information. (Note: length was divided into a boolean representing employed or unemployed in a test model, but that level of consolidation reduced effectiveness of the model at predicting profitability.)
Grade was consolidated to 3 levels in testing, but it reduced overall accuracy. It will remain unconsolidated.

```
setLength <- function(x){
  col_id <- which( colnames(complete_loans)=="length" )
  if(x[col_id] == "< 1 year" | x[col_id] == "1 year" | x[col_id] == "2 years" | x[col_id] == "3 years" | x[col_id] == "4 years")  "un
der 5 years"
  else if (x[col_id] == "5 years" |  x[col_id] == "6 years" |  x[col_id] == "7 years" |  x[col_id] == "8 years" | x[col_id] == "9 year
s" |  x[col_id] == "10 years") "5 - 10 years"
  else x[col_id]
}
complete_loans$reason[complete_loans$reason != "debt_consolidation"] <- "other"
complete_loans$verified[complete_loans$verified == "Source Verified"] <- "Verified"
complete_loans$length <- apply(complete_loans, 1, setLength)
```

Step 5 - Further Consolidation

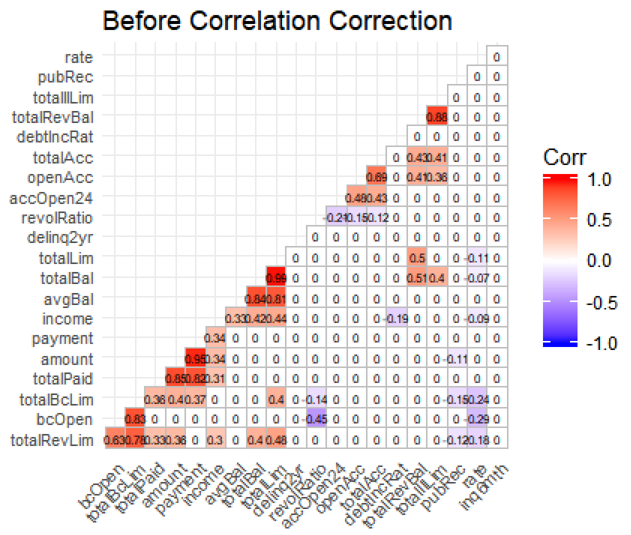
Collinearity

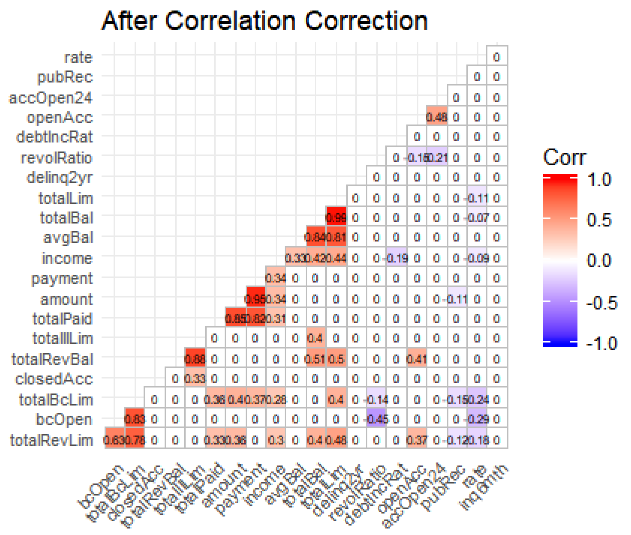
There appears to be some potential redundancy in the data, based on an analysis of the data terms and the correlation matrix below. Highly correlated data isn't necessarily an issue when the goal is to be able to do predictions. But if we can combine or split columns in obvious ways to make a more parsimonious model, we should.

Some correlation can be removed by subtracting the openAcc from the totalAcc and creating a new field called closedAcc (removing totalAcc). Splitting this predictor results in decreased correlations for both closedAcc and openAcc.

The highest remaining correlated predictors are totalBal and totalLim. Both seem like they could have predictive power, and neither has obvious ways to split or join. The total balance column has 97 instances where it is smaller than the total revolving balance (which one would assume should be smaller than a total balance). It is possible this is dirty data. In a real-world scenario, confirmation of the meaning of this column would be sought. Likewise, the totalLim column is not a total of the other limit columns or any obvious calculation of the other limit columns. Given the lack of clarity around these columns, they will be retained. Additional assessment of correlations can always be accomplished with VIF after model development, if necessary.

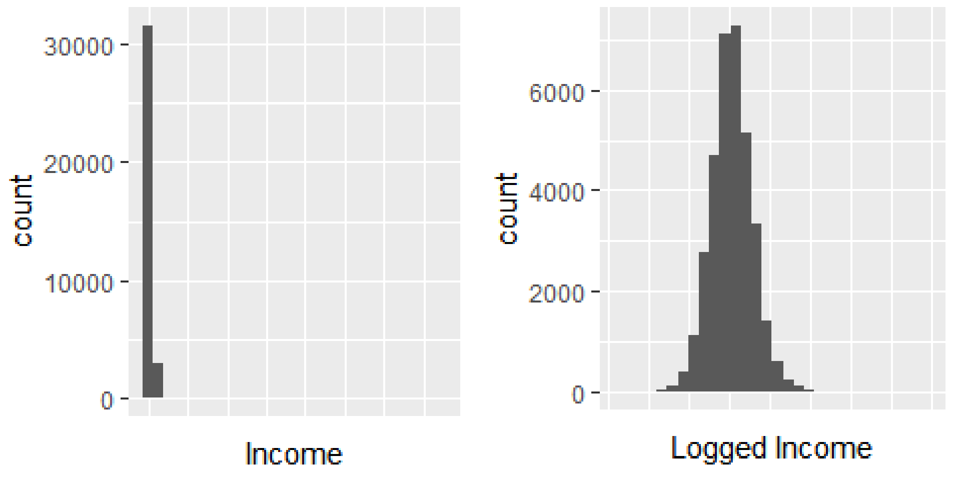
```
#make closedAcc
complete_loans$closedAcc <- complete_loans$totalAcc - complete_loans$openAcc
drops <- c("totalAcc")
complete_loans <- complete_loans[ , !(names(complete_loans) %in% drops)]
```





Skew

Most of the numeric predictors show skew. While there is no expectation of normality in the independent variables in logistic regression, there can be value in normalizing the data distributions. In this case, skewed data will not be corrected except in one instance - income. A log transformation creates a much more normalized version of income. Models run with and without the transformed income column show improvement with the transformation. A model with income's outliers removed was also run, but it did not improve the model. Transformations of other skewed columns resulted in models that lacked evidence of fit.



```
#transform income
complete_loans$logIncome <- log(complete_loans$income)
```

Data Cleanup - Step Six - Dividing the Data Set

To do predictive analytics, a training set and a test set are needed. In this case, 80% of the data will be the training set, with 20% reserved as a test set. The variable totalPaid (which is not a predictor, because it happens post-loan award) will be removed from the training set, but retained in the predictor set as a validator.

```
smp_size <- floor(0.8 * nrow(complete_loans))
train_ind <- sample(seq_len(nrow(complete_loans)), size = smp_size)

training_loans <- complete_loans[train_ind, !(names(complete_loans) == "totalPaid")]
test_loans <- complete_loans[-train_ind, ]
```

Section 4: First Model and Diagnostics

First models generally include all available variables, so that the data scientist has a sense of what can be accomplished with all the data at hand. First models are often too complicated, but act as a benchmark for future, improved models. The following first model was fit to the data.

```
training.full <- glm(status~., data=training_loans, family="binomial")
formula(training.full)

## status ~ amount + term + rate + payment + grade + length + home +
##     verified + reason + debtIncRat + delinq2yr + inq6mth + openAcc +
##     pubRec + revolRatio + totalBal + totalRevLim + accOpen24 +
##     avgBal + bcOpen + totalLim + totalRevBal + totalBcLim + totalIlLim +
##     closedAcc + logIncome
```

First Model Analysis

This model could use some improvement. The model includes 12 variables that do not show any statistical significance at the .05 level. The pseudo R-squared is just 0.108. The Hosmer-Lemeshow test gives a p-value of 0.647, however, indicating a lack of evidence that the model is not a good fit.

To assess the predictive capability of the first model, a confusion matrix was created based on the results of predicting the test data set.

```
getPredStatus <- function(model, threshold=.5){
  pred = predict(model, newdata=test_loans, type="response")
  pred.status <- cut(pred, breaks=c(-Inf, threshold, Inf),
                labels=c("Bad", "Good"))  #R will use bad = 0 and good = 1.
}

getCM <- function (model, threshold=.5 ){
  pred.status <- getPredStatus(model, threshold)
  cm <- caret::confusionMatrix(data=pred.status, test_loans$status, positive = "Bad")
  return(cm)
}

cf <- getCM(training.full)

addmargins(cf$table)

##           Reference
## Prediction  Bad Good  Sum
##       Bad   163  161  324
##       Good 1353 5251 6604
##       Sum  1516 5412 6928
```

This model has an overall accuracy rate of 78.1% with 10.8% of the bad loans correctly identified and 97% of the good loans correctly identified. This means that the model is much better at accurately predicting loans that will not default than it is at predicting loans that will default.

If the goal is to have a model that accurately predicts when someone will default on their loan so that the bank can avoid the costs of bad loans, then this model is not a very good model.

Section Five - Improved Model and Diagnostics

Balancing the data set

The current model suffers from a lack of ability to detect the event of concern due to a low percentage of the event in the dataset. Only 21.9% of training loans are bad loans. Oversampling to balance the training set will improve the sensitivity of the model.

```
bad_loans <- training_loans[which(training_loans$status == 'Bad'), ]
ind <- sample(nrow(bad_loans), 15588, replace=TRUE)
training_balanced <- rbind(training_loans, bad_loans[ind, ])
```

With a rebalanced dataset, another full, baseline model and confusion matrix were produced, using the same code previously shown and the dataset "training_balanced."

The new model has an overall accuracy rate of 64.9% and correctly identifies 66.9% of the bad loans and 64.3% of the good loans. Refining the model should improve those statistics.

Model Refinement - Automatic Model Selection

A standard practice with logistic regression is to use automatic model selection to find a better model. This approach automatically fits multiple models, looking for the best fit via the Akaike Information Criterion (or other measures). Two automatic model selections were utilized - a step approach and a genetic algorithm approach. Because we have many predictors, some of which have many levels, only first order terms were used in automatic modeling. Additionally, a small interaction model was hand-generated.

The stepped and genetic algorithm models each included one insignificant coefficient at the .01 level, so they were further reduced. A Chi-square anova was conducted with the parent model for each sub model, and there was not sufficient evidence to assert that the coefficients in the parent model that are not included in the child model are significant.

VIF was evaluated for the stepped model. One additional predictor was removed to lower the VIF scores. However, the Chi-square anova in this case indicated sufficient evidence that the removed coefficient varied significantly from zero, which means it added value to the model, and a model without it should not be used. This model also had a Hosmer-Lemeshow p-value of 0. This model is eliminated.

Each remaining model was evaluated for AIC and pseudo-R-squared. Predictions were generated and each model's accuracy, sensitivity and specificity were determined. Hosmer-Lemeshow tests were conducted and all models had insufficient evidence to suggest they don't fit except the interaction model, thus eliminating the interaction model.

Model	AIC	Pseudo R^2	Anova p-value	Hosmer-Lemeshow	Total Coefficients	Accuracy	Sensitivity	Specificity
Full	53056.99	0.117	NA	0.653	35	64.9	66.9	64.3
Stepped	53052.29	0.117	0.731	0.588	32	65.0	67.0	64.4
Stepped - VIF Corrected	53180.83	0.115	0.000	0.000	25	66.0	64.4	66.4
Stepped-Reduced	53054.01	0.117	0.053	0.698	31	65.2	67.2	64.6
Genetic	53052.90	0.117	NA	0.802	30	65.1	67.0	64.5
Genetic Reduced	53054.47	0.117	0.059	0.887	29	65.2	67.1	64.7
Interaction Model	55140.39	0.082	NA	0.000	33	64.9	58.2	66.7

Based on the output above, the recommendation is to use the genetic reduced model. The pseudo R-squared is the same for all the models that have evidence of fit (.117). The accuracy and specificity of the genetic reduced model are marginally higher than other models. The sensitivity is marginally lower than the stepped reduced model, but barely so. The AIC was slightly higher than the parent model, but the difference is minor.

The final chosen model is:

```r
glmulti.reduced <- glm(status ~ 1 + term + grade + length + home + verified + reason +
   amount  + payment + logIncome + debtIncRat + delinq2yr +
   inq6mth + pubRec + revolRatio + accOpen24 + totalLim + totalRevBal +
   totalBcLim + totalIlLim + closedAcc, data = training_balanced, family="binomial")
```

	Estimate	Std..Error	z.value	Pr...z..
(Intercept)	0.5327494	0.3461609	1.539022	0.1237988
term 60 months	-0.9473010	0.0616559	-15.364330	0.0000000

gradeB	-0.3184546	0.0410709	-7.753772	0.0000000
gradeC	-0.7182982	0.0426036	-16.860053	0.0000000
gradeD	-0.9500863	0.0487671	-19.482116	0.0000000
gradeE	-1.0246992	0.0584604	-17.528086	0.0000000
gradeF	-1.1240115	0.0816784	-13.761437	0.0000000
gradeG	-1.1691560	0.1437023	-8.135959	0.0000000
length5 - 10 years	-0.0493168	0.0284786	-1.731712	0.0833248
lengthn/a	-0.4990635	0.0487733	-10.232304	0.0000000
lengthunder 5 years	-0.0363511	0.0259816	-1.399109	0.1617802
homeOWN	-0.2331435	0.0369326	-6.312674	0.0000000
homeRENT	-0.3235804	0.0270912	-11.944111	0.0000000
verifiedVerified	-0.0836250	0.0251587	-3.323897	0.0008877
reasonother	-0.0591145	0.0218095	-2.710493	0.0067183
amount	0.0000369	0.0000095	3.880636	0.0001042
payment	-0.0015900	0.0002968	-5.356747	0.0000001
logIncome	0.1481256	0.0319512	4.636000	0.0000036
debtIncRat	-0.0213679	0.0015460	-13.821523	0.0000000
delinq2yr	-0.1202203	0.0118881	-10.112680	0.0000000
inq6mth	-0.0630381	0.0110882	-5.685130	0.0000000
pubRec	-0.0662557	0.0166850	-3.970972	0.0000716
revolRatio	-0.3828779	0.0522488	-7.327969	0.0000000
accOpen24	-0.0888285	0.0039457	-22.512797	0.0000000
totalLim	0.0000006	0.0000001	5.679707	0.0000000
totalRevBal	-0.0000022	0.0000006	-3.946161	0.0000794
totalBcLim	0.0000046	0.0000008	6.102826	0.0000000
totalIlLim	0.0000026	0.0000006	4.374985	0.0000121
closedAcc	0.0150435	0.0012876	11.683627	0.0000000

Section 6 - Tuning the Predictions and Profit Analysis

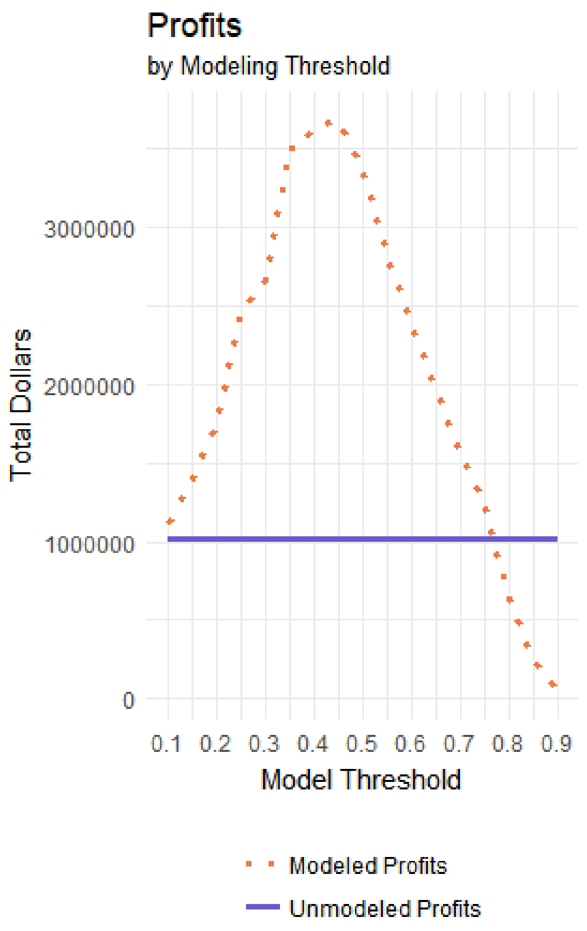
With the chosen formula in hand, further tuning can be accomplished by adjusting the probability threshold. The following results show the difference in profit at thresholds of .1 to .9. Note that the overall profit before modeling was $1,018,079.

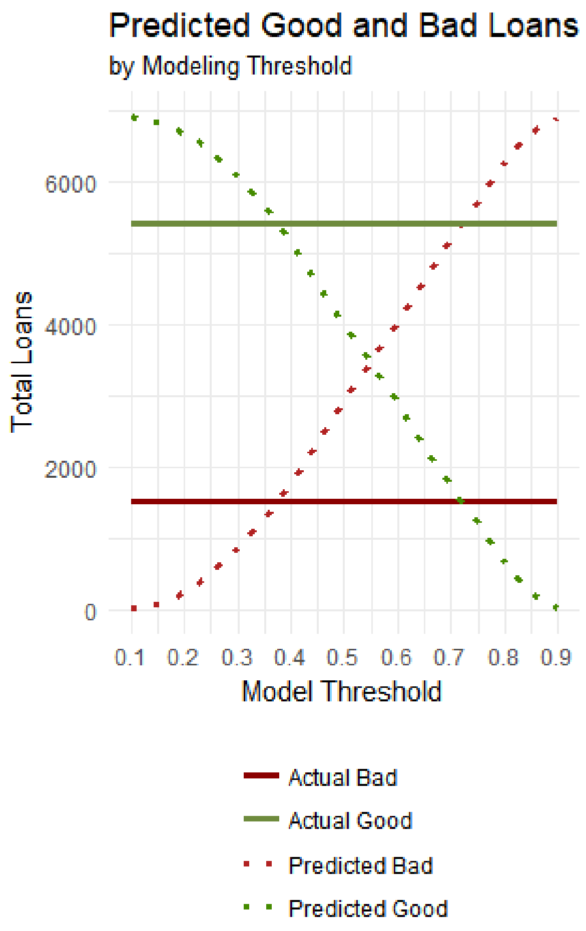
Threshold	Accuracy	Sensitivity	Specificity	Modeled Profits	Loans Granted	Loans Denied
0.100	78.2	0.7	99.9	1113755.97	6914	14
0.150	78.2	3.2	99.2	1384690.10	6837	91
0.200	78.1	8.0	97.7	1747786.95	6684	244
0.250	77.9	15.7	95.3	2459110.40	6434	494
0.300	76.7	25.3	91.1	2660297.76	6063	865
0.350	75.8	37.1	86.6	3501566.77	5639	1289
0.375	74.7	42.6	83.6	3520942.27	5396	1532
0.400	73.4	48.3	80.4	3641839.31	5137	1791
0.450	69.6	58.2	72.9	3684208.68	4577	2351
0.500	65.2	67.1	64.7	3356382.26	3998	2930
0.550	60.4	74.5	56.5	2824240.95	3442	3486
0.600	55.1	80.9	47.9	2385314.29	2882	4046
0.650	49.2	86.7	38.7	1962595.90	2297	4631

0.700	43.3	91.6	29.8	1569611.66	1740	5188
0.750	37.3	95.6	21.0	1235390.74	1200	5728
0.800	30.4	98.0	11.4	630837.24	648	6280
0.850	25.2	99.6	4.3	237719.51	240	6688
0.900	22.4	100.0	0.7	45826.01	37	6891

The table above shows that for this test set of data as the threshold increases:
-profits initially increase, then begin to decrease between the thresholds of .45 and .5
-more loans are denied
-fewer loans are granted.

The graphs below show the realized profit with modeling compared to the profit without modeling, and the trends in over and under predicting good and bad loans as threshold increases.





In this test data set, the point at which the modeled profit falls below the unmodeled profit is between the thresholds of .75 and .8. Below that, any threshold chosen will result in an increase in profit.

This model begins over-estimating bad loans at a threshold of approximately .375. Over-estimating bad loans is a lost profit opportunity and a customer relations pitfall. If a model such as this is used, it is extremely important to examine the model for any bias that may have ethical implications. For example, do minority populations tend to use specific types of credit such that their total revolving credit balance would have a greater impact on a loan application than the majority population? If so, then care needs to be taken in how this model is implemented.

If bias is accounted for and managed, then the bank needs to decide based on balancing maximized profit with their comfort level with falsely denying loans that would otherwise have turned out fine. A compromise position could be 0.4. At this threshold, we see an overall accuracy rate of 73.4%, a sensitivity of 48.3% and a specificity of 80.4%. In the test data, this threshold results in a profit of $2,623,760 more than was seen without any modeling, and 25.85% of the loans were denied, which is not too far off from the actual percentage of bad loans in the test data set (21.88%). While using this threshold results in fewer loans overall, the loans that are granted tend to be better quality loans, yielding higher overall profits for the bank. There were, however, 1,059 loans wrongfully turned down, which would have brought in $3,995,360. (Everything is a tradeoff.)

Section 7 - Results Summary

The final classification model chosen uses the following predictors: term, grade, length, home, verified, reason, amount, payment, income, debtIncRat, delinq2yr, inq6mth, pubRec, revolRatio, accOpen24, totalLim, totalRevBal, totalBcLim, totalIlLim, and closedAcc.

Additional work could be done on this model to look for interactions in the chosen terms, or possible quadratic effects, or to convert delinq2yr or inq6mth to factors and remodel. Ideally, we'd like to see a model that more accurately predicts both good and bad loans than this model does. But, despite its flaws, this model can still be used to generate additional profit.

To recap, the chosen threshold is 0.4, resulting in an increase in profits for the bank of $2,623,760, using the test data set. The threshold could always be adjusted. It will be important to continually monitor the threshold as circumstances change.