Final Project Part 1 & 2

Jason Vander Weele

May 1, 2018

# Executive Summary

## Bank31 Loan Profitability Model

An analysis of how to increase Bank31 loan profitability was recently completed. The analysis used data modeling methods to determine how to increase likelihood of bank profit. The results of the analysis conclude that by implementing a data model that was developed the bank could increase its current loan profit by 153%.

To accomplish this, the model uses historic loan data for which the amount of profit is known. The historic data (such as debt to income ratio, total credit, etc.) is used to develop an algorithm that can be applied to new incoming loan applications. By applying the algorithm to new loan applications, bankers can then determine whether to approve or deny a bank loan. The banker would be informed by the algorithm whether the loan should or should not be made. The result of using this algorithm will be an increase in profit of the loans awarded.

We recommend implementation of the scoring model, which will result in:

* the ability to **predict** future loan **charge offs and defaults** – **before** awarding the loan
* an **increase** in loan **profit** by up to 153%
* a **decrease** in number of **loans** **awarded** by up to 21% (while still achieving an increase in profit)

The following should be considered regarding use of the model.

First, while the model is accurate today, it is limited in that it will change with time as the applicant pool shifts resulting in differences in loans awarded. For these reasons, it will need ongoing examination for continued accuracy.

Second, there is capability to adjust the model based on risk-preferences of Bank31 and its shareholders. While we are confident the model will increase profit, considerations for the acceptable ratio of potentially risky to non-risky loans can be incorporated. That is, we ask that Bank31 provide more guidance as to the acceptable levels of risk so that we may set the threshold for individual loan acceptance or rejection to meet the bank’s risk needs, accordingly.

Lastly, the model is always dependent on how ‘truthful’ information about the applicant is. That is, if there are errors or omissions relating to the credit history of an applicant, the model may produce erroneous results. It will be critical that correct and verifiable information is used as input to the model to ensure a great output.

# Section 1: Setup/Load Packages

library(ggplot2); library(gridExtra); library(GGally); library(HH); library(leaps); library(psych); library(scales)

# Section 2: Introduction

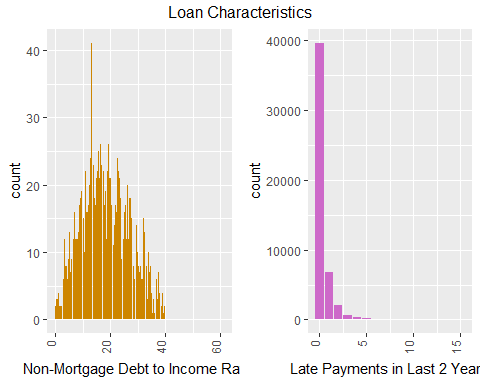
Bank31 is a small regional bank located in San Jose, CA. Over the years, Bank31 management has realized they hold key information in their historical data that could help drive better decisions on bank loans and result in stronger returns in their books. Bank31 has requested an investigation into their data. Specifically, Bank31 has asked for a loan prediction model to gain an understanding and method to determine if a customer applying for a loan is likely to default. The end goal is to develop a model to increase profit by making predictions about whether a loan to a customer will be good or bad by scoring individual applicants and assigning a minimum threshold to use in granting a loan.

The approach we will use is for the analysis is to use logistic regression to model any number of the 29 independent variables (excluding ID and totalPaid) to predict the dependent variable loan default. Prior to building the model we will perform some transformations, cleaning, and possible missing value imputation in preparing the 50,000 records provided. We will then split the data into training and test datasets before building our models.

# Section 3: Preparing and Exploring the Data

After loading the data, we take a look at some of the summary statistics to better understand the dataset we are working with. By doing so, we observe there are some missing value categories, such as in the *home* variable, *bcratio*, and *revolRatio*.

By examining the variable *state* we see most of the 50 states are represented with varying counts between them. For example, we see most of the records in our dataset come from California, whereas barely any come from North Dakota. This tells us that maybe there is some correlation for the size of a state compared with its population, but we can’t know for sure. Because it has so many categories we will transform this variable into region.



Above we see non-mortgage debt to income ratio which looks to be a fairly normal distribution with a mean of 18.7 percent. We see most people have zero late payments. This means as a predictor for bad loans we could imagine the category of ‘zero’ isn’t going to be very helpful to the model because so much of the data is represented by that category, and likely from both good and bad loan applicants. However, we will find out if our hunch is right later on.

We reviewed correlation within the dataset and observed some highly correlated variables. This means we expect the dataset contains some collinearity between variables. For example, performing a correlation function between *totalLim* and *totalBal* results in a correlation of .9858 meaning there is a strong positive linear relationship between these two variables. We can further understand this to mean that these two variables will provide overlapping explanations of the same variation in the response variable.

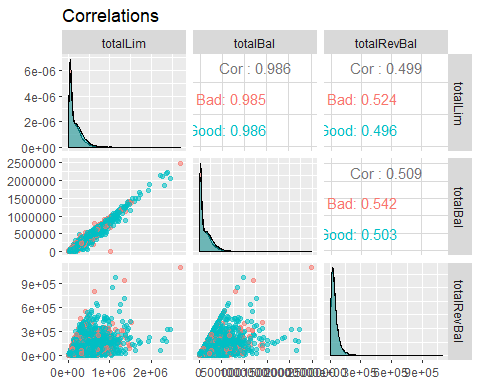
#Correlation between totalLim and totalBal:  
with(mydata,cor.test(totalLim, totalBal)$estimate)

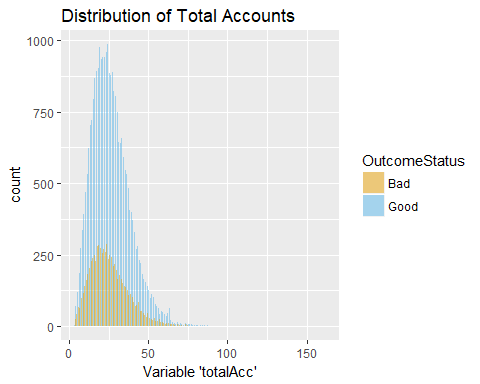
## cor   
## 0.9857649

We know that our prediction is going to be a variation of the loan *status* variable. In our case, we want a *Good* status to include “Fully Paid” and a *Bad* status to include “Charged Off” or “Default”. All other status types are considered to be unknown final outcomes and are therefore removed from the data in their entirety. Below is the number of records by status in the data, which we will transform to a factor variable having a value of Good or Bad as our dependent outcome variable for the logistic regression.

## Number of Records  
## 1  
## Charged Off 7579  
## Current 14532  
## Default 2  
## Fully Paid 27074  
## In Grace Period 261  
## Late (16-30 days) 102  
## Late (31-120 days) 449

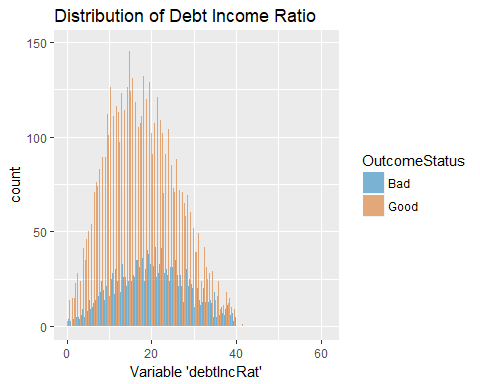
Now we can see correlation as it relates to good and bad loan outcomes. Notice *totalLim* and *totalBal* are highly correlated to both good and bad outcomes (.986 overall). Whereas there is only a moderate correlation (~.50) for *totalRevBal* and *totalBal* when compared to *totalLim*. This result can be observed in the plots below where the red color indicates bad loans and the blue color indicates good loans.





Above, we see the distribution of total accounts by loan outcome status. We can see there isn’t an obvious visual difference in good or bad outcomes based on the total account distributions. On its own, this variable wouldn’t appear to be a great predictor.

Below we see that the debt-to-income ratio appears to have a somewhat shifted distribution for good and bad loan outcomes. The peak for good loans is a little left of the peak for bad loans. This plot suggests that bad loans are given to people with slightly higher debt to income ratios. Perhaps this variable will be influential in the prediction model.



In our analysis, we found missing data. We determined that when both *bcOpen* and *bcRatio* fields are missing we would impute zero-values for because they are unused credit on credit cards and ratio of total credit card balance. We imputed missing values for *bcOpen* and *bcRatio* when both were NA and imputed zero into the records if one of the values was missing, as these appeared to be MAR values.

We recategorized *state* into *Region*. We also consolidated *reason* into larger buckets. After attempting some re-categorization for *employment* it was determined to remove. The *employment* variable simply listed or did not list the job title of the loan applicant.

We deleted rows where the variable *revolRatio* was missing.

Finally, after performing transformations into new variables with consolidated categories, we eliminated the variables: *status*, *state*, *reason*, *length*, and *employment*.

The procedures above successfully dealt with all missing values in the dataset as well as transforming categories and we have a resulting count of 34,618 rows.

Below we see the outcome of re-categorization of the *length* variable by grouping the smaller groups together. We decided to leave the ‘unknowns’ included because it is possible that having an unknown value for the length of time employed might be useful information to our model in some way. That is, it could be measuring something that is not yet obvious in our analysis.

## 0 to 3 years 4 to 7 years 8 to 10+ years Unknown   
## 10965 7347 14484 1822

Finally, below we split the total count of records having “good” and “bad” outcomes into a train (80%) and test (20%) dataset.

#Display outcome tables for each dataset  
table(train$OutcomeStatus)

##   
## Bad Good   
## 6095 21599

table(test$OutcomeStatus)

##   
## Bad Good   
## 1480 5444

# Section 4: First Model and Diagnostics

Now that we have cleaned our data and have a basic understanding of it, we can perform our first logistic regression model. Below is the output of that model, using the independent variables in our training data set. Note: we have further removed totalPaid and loadID as they are not to be used as predictors because we wouldn’t know totalPaid prior to issuing a loan and loadID is just a row identifier.

#Create the logistic regression model and display summary:  
loan.out <- glm(OutcomeStatus~., data=train, family="binomial")

Next, we apply our model using the predict function to our test data (which was kept blind from the model).

prediction <- predict(loan.out, test, type="response")

Upon review of the test predictions data, we find that our model’s median predicted value for the test data is .81. A review of the summary statistics, below, shows that the first quartile begins at .70. This means at least 75% of our outcomes are much greater than .50. This tells us that the balance in our dataset is more heavily skewed toward individuals who are going to be classified as “good” (above the .50 threshold). This isn’t surprising, because our original dataset has only 22% of cases being actual outcomes of ‘bad’ and thus we would expect to have many fewer prediction outcomes of bad than good.

summary(prediction)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1809 0.6996 0.8114 0.7816 0.8942 0.9923

#Get 1/0 for the contingency table  
test$OutcomeStatusPredicted <- ifelse(prediction>= .5, "Good", "Bad")  
  
threshhold <- 0.5 # Declare threshold .5  
predOutcomeStatus <- cut(prediction, breaks=c(-Inf, threshhold, Inf),   
 labels=c("Bad", "Good")) #Bad=0, Good=1  
  
cTab <- table(test$OutcomeStatus, predOutcomeStatus)   
addmargins(cTab)

## predOutcomeStatus  
## Bad Good Sum  
## Bad 179 1301 1480  
## Good 157 5287 5444  
## Sum 336 6588 6924

p <- sum(diag(cTab)) / sum(cTab) # compute the proportion of correct classifications  
print(paste('Proportion correctly predicted = ', round(p,2)))

## [1] "Proportion correctly predicted = 0.79"

The total proportion correctly predicted was 79%. This means that of all the test data (good and bad) loan results, our model was accurate in its prediction 79% of the time. However, because only about 22% of our data includes “bad” loan outcomes our logistic regression method is not going to be as good at predicting the “bad” outcomes as it is the “good” outcomes (as we previously explored). More balancing of the data is needed. We see evidence of this in our overall rate being 79% but our correct classification of “bad” loans (the ones we’re really interested in predicting) is only accurate for 179 of 1480 results. That is, the first logistic regression model only correctly predicted “bad” loans 12.1% of the time. Therefore, it is not recommended that we use this model at this time and we should continue developing other models and methods to be obtain greater accuracy.

# Section 5: Improved Model and Diagnostics

Here we begin with creating our balanced training set by resampling our bad outcomes.

#View our proportions of training data  
table(train$OutcomeStatus)

##   
## Bad Good   
## 6095 21599

prop.table(table(train$OutcomeStatus))

##   
## Bad Good   
## 0.2200838 0.7799162

#Create our resample 'badtoappend'  
set.seed(101)  
badrows <- train[which( train$OutcomeStatus == "Bad"),]  
summary(badrows$OutcomeStatus)

## Bad Good   
## 6095 0

badtoappend <- badrows[sample(nrow(badrows), size=15504, replace=TRUE),]

#Need to create a new train dataset that includes both Train and BadToAppend, for a total of 27694+15504=43198 rows  
mytrain <- rbind(train, badtoappend)

We split the dataset into a train (80%) and test (20%) set and then balanced the Good/Bad loans in the train set so that they each represent 50% of the training set. The method used to balance the train set was to extract the bad loans from our train set and sample with replacement (oversampling). Then, we appended the sample to our original train set representing 21,599 cases of each good and bad. The reason for creating a balanced data set to create bias so that our predictions work better for “Bad” loans. If we didn’t do this, our model would likely be better at predicting “Good” loans but not necessarily for “Bad” loans, since only 20% of the data it would have trained on would have had the “Bad” loan outcome. In other words, this balances our training set to ensure the logistic regression model is trained to correctly predict each category as opposed to just good or bad loans because differing frequency.

Now we are ready to create our new model from the training set and make and classify predictions.

#Create the logistic regression model  
loanBAL.out <- glm(OutcomeStatus~., data=mytrain, family="binomial")

#Create prediction values for test data  
predictionBAL <- predict(loanBAL.out, test, type="response")  
  
#Classify 1/0 for the contingency table  
test$OutcomeStatusPredictedBAL <- ifelse(predictionBAL>= .5, "Good", "Bad")  
threshhold <- 0.5 # Declare threshold .5  
predBALOutcomeStatus <- cut(predictionBAL, breaks=c(-Inf, threshhold, Inf),   
 labels=c("Bad", "Good")) #Bad=0, Good=1  
cTab <- table(test$OutcomeStatus, predBALOutcomeStatus)   
addmargins(cTab)

## predBALOutcomeStatus  
## Bad Good Sum  
## Bad 1001 479 1480  
## Good 1918 3526 5444  
## Sum 2919 4005 6924

p <- sum(diag(cTab)) / sum(cTab) # compute the proportion of correct classifications  
print(paste('Proportion correctly predicted = ', round(p,2)))

## [1] "Proportion correctly predicted = 0.65"

The total proportion correctly predicted was 65%. This means that of all the test data (good and bad) loan results, our model was accurate in its overall prediction 65% of the time. Our correct classification of “bad” loans (the ones we’re really interested in predicting) is accurate for 1001 of 1480 results. That is, after balancing the training data our logistic regression model correctly predicted “bad” loans 67.6% of the time. However, our incorrectly predicted “good” loans increased to 35.2%. Overall, this model does better predicting the bad loans and worse predicting the good loans. That is, we increase our failure to correctly predict good loans while increasing our accuracy in correctly predicting bad loans. Therefore, it is not recommended that we use this model at this time and we should continue developing other models and methods to be obtain greater accuracy.

To try to produce a more effective logistic regression model we ran a series of stepwise procedures: backward and forward. Neither of these methods significantly improved results when applied against the test data with a .5 threshold. Additionally, we used regsubsets method with a 10 variable maximum, only achieving a .14 R-squared. For the step procedure, the best model was using the backward method with an AIC of 52,981. There were 21 variables of importance in that model selection which are: amount + term + payment + grade + home + verified + reason + debtIncRat + delinq2yr + inq6mth + openAcc + revolRatio + totalAcc + totalRevLim + accOpen24 + totalLim + totalRevBal + totalBcLim + totalIlLim + Region + lengthtransformed

#Best Models:  
#Best backward:  
bestloanbackward.out <- glm(formula = OutcomeStatus ~ amount + term + payment + grade +   
 home + verified + reason + debtIncRat + delinq2yr + inq6mth +   
 openAcc + revolRatio + totalAcc + totalRevLim + accOpen24 +   
 totalLim + totalRevBal + totalBcLim + totalIlLim + Region +   
 lengthtransformed, family = "binomial", data = mytrain)  
  
#Best forward:  
bestloanforward.out <- glm(formula = OutcomeStatus ~ grade + debtIncRat + term + totalLim +   
 accOpen24 + Region + home + lengthtransformed + bcOpen +   
 totalAcc + delinq2yr + payment + amount + reason + verified +   
 inq6mth + revolRatio + openAcc + totalBal + totalIlLim +   
 totalRevBal + totalBcLim + totalRevLim, family = "binomial",   
 data = mytrain)  
  
  
extractAIC(bestloanbackward.out)

## [1] 44.00 52981.36

extractAIC(bestloanforward.out)

## [1] 46.00 52984.93

Below, we take our best automatic selection model and make predictions and classify the outcomes using the .5 threshold.

#AutoOut  
loanAUTO.out <- glm(formula = OutcomeStatus ~ amount + term + payment + grade +   
 home + verified + reason + debtIncRat + delinq2yr + inq6mth +   
 openAcc + revolRatio + totalAcc + totalRevLim + accOpen24 +   
 totalLim + totalRevBal + totalBcLim + totalIlLim + Region +   
 lengthtransformed, family = "binomial", data = mytrain)  
  
#Create prediction values for test data  
predictionAUTO <- predict(loanAUTO.out, test, type="response")  
  
#Get 1/0 for the contingency table  
test$OutcomeStatusPredictedAUTO <- ifelse(predictionAUTO>= .5, "Good", "Bad")  
  
threshhold <- 0.5 # Declare threshold .5  
predAUTOOutcomeStatus <- cut(predictionAUTO, breaks=c(-Inf, threshhold, Inf),   
 labels=c("Bad", "Good")) #Bad=0, Good=1  
  
cTab <- table(test$OutcomeStatus, predAUTOOutcomeStatus)   
addmargins(cTab)

## predAUTOOutcomeStatus  
## Bad Good Sum  
## Bad 1002 478 1480  
## Good 1913 3531 5444  
## Sum 2915 4009 6924

p <- sum(diag(cTab)) / sum(cTab) # compute the proportion of correct classifications  
print(paste('Proportion correctly predicted = ', round(p,2)))

## [1] "Proportion correctly predicted = 0.65"

When applied to the test data at the .5 threshold, the best step model total proportion correctly predicted was 65%. Our correct classification of “bad” loans is accurate for 1002 of 1480 results, or 67.7% of the time. Our incorrectly predicted “good” loans are 35.1%. Overall, this model with only 21 predictors did about the same as our linear regression model using all of the predictors. We did note previously that some variables were highly correlated with others and that would lead us to believe there is overlapping explanation in the predictor variables for the full model, and thus why our final model with reduced predictors would make sense.

# Section 6: Tuning the Predictions and Profit Analysis

We now adjust our classification threshold to .60 to see if there is an improvement, resulting in the below:

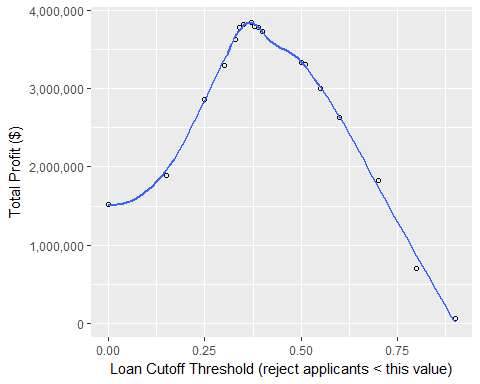
## predAUTOOutcomeStatus  
## Bad Good Sum  
## Bad 1213 267 1480  
## Good 2821 2623 5444  
## Sum 4034 2890 6924

## [1] "Proportion correctly predicted = 0.55"

By experimenting with the classification threshold to increase it from .5 to .6 or above, we can greatly increase the number of “bad loans” predicted to be bad. However, this is done at the expense of incorrectly predicting many of the “good loans” to be bad.

Continuing, we performed extensive experimentation with various classification thresholds, and found .37 is ideal for our model. The way we performed this was by picking an arbitrary threshold and modifying it slightly to observe the effect. Each time, we summed the profit to see the total profit at any given threshold. To do this, we started at the .50 threshold and went in each direction. We immediately noticed increasing the threshold, from .50 to .55, for example, resulted in a decrease in overall profit. However, when we decreased the threshold, profit increased. So, we continued decreasing until we observed a plateau in the total profit. This occurred at .37.

We see below that using a value of .37 results in the most profit.



A threshold of .37 is achieved with the following and we see the results of the predicted vs. actual outcome:

#Get 1/0 for the contingency table  
test$OutcomeStatusPredictedAUTO <- ifelse(predictionAUTO >= .37, "Good", "Bad") # Declare classification threshold (this is where we put in .25, .50, .60, .37, etc.)  
  
threshhold <- .37 # Declare cutoff threshold (this is where we put in .25, .50, .60, .37, etc.)  
predAUTOOutcomeStatus <- cut(predictionAUTO, breaks=c(-Inf, threshhold, Inf),   
 labels=c("Bad", "Good")) #Bad=0, Good=1  
  
cTab <- table(test$OutcomeStatus, predAUTOOutcomeStatus)   
addmargins(cTab)

## predAUTOOutcomeStatus  
## Bad Good Sum  
## Bad 626 854 1480  
## Good 819 4625 5444  
## Sum 1445 5479 6924

p <- sum(diag(cTab)) / sum(cTab) # compute the proportion of correct classifications  
print(paste('Proportion correctly predicted = ', round(p,2)))

## [1] "Proportion correctly predicted = 0.76"

#Profit for Loans Given  
sum(test$profit[which(test$OutcomeStatusPredictedAUTO=="Good")])

## [1] 3848457

#Number of Loans Given  
length(test$profit[which(test$OutcomeStatusPredictedAUTO=="Good")])

## [1] 5479

The result of experimenting with various thresholds is that the ideal threshold for maximizing profit is to refuse loans to any applicant scoring below .37. Anything above a .37 should be considered a ‘good’ loan for purposes of profit maximization. This means that while we know some individuals who do achieve above .37 will default, the profits, overall, will still be maximized by allowing individuals with at least that score to receive loans. That is, even though one person scoring above .37 may default (while still producing some revenue for the bank), there is more potential for profit reduction by misclassifying other applicants above that score because they may actually be good and we would miss those profits by rejecting them.

For example, we look at a few of the results of a various classification thresholds:

.00 Profit: $1,521,970 over 6,924 loans (This is the full dataset, without model)

.30 Profit: $3,290,391 over 6,084 loans

.37 Profit: $3,848,457 over 5,479 loans

.50 Profit: $3,338,390 over 4,009 loans

.70 Profit: $1,822,461 over 1,788 loans

It is important for us to set the rejection threshold. The reason it is important is because it allows us to have a ‘maximum effect’. Certainly just taking a cutoff value of .50 would be better than not doing a model at all. However, we have shown above that by adjusting this threshold we can actually ‘squeeze out’ more profit than we would have by just settling for a .50 threshold. That is, by changing our rejection threshold to be anything below .37 we increase our profit even more.

While the total profit will increase, we know that doing this (changing the rejection from .50 to .37) will also increase the number of bad loans. In terms of “risk” we must be clear that the risk of seeing more defaults increases as we decrease our rejection threshold. On the other hand, by setting the threshold higher we would potentially miss good loans.

# Section 7: Results Summary

To recap, the final model we used is a logistic regression model. The target variable was the transformed OutcomeStatus and the predictors were: amount + term + payment + grade + home + verified + reason + debtIncRat + delinq2yr + inq6mth + openAcc + revolRatio + totalAcc + totalRevLim + accOpen24 + totalLim + totalRevBal + totalBcLim + totalIlLim + Region + lengthtransformed.

Our final model threshold for good loans is any applicant scoring greater than .37 which results in overall model accuracy of 76% with 42% of bad loans being correctly classified and 85% of good loans correctly classified.

When we apply our model to our test dataset, we are able to correctly classify 631 out of 1,480 bad loans and 4,618 out of 5,444 good loans based on the confusion matrix with an overall accuracy of 76%.

By setting a threshold of .37 and using our best logistic regression model, we actually achieve an increase of 153% in profit while decreasing total number of loans overall by 21%.