Project Final Report

Teresa Bodart

04/25/2021

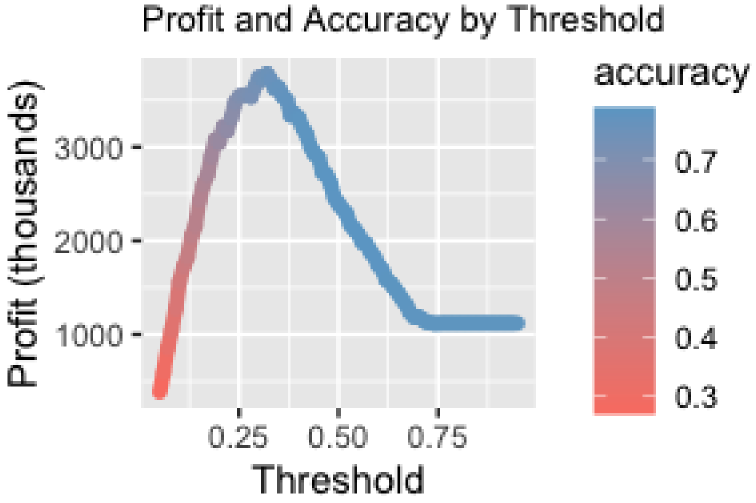
Section 1: Executive Summary

This report provides an analysis and evaluation of loan profitability by predicting which loan applicants are likely to default on their loans in the future. The methods of analysis include logistic regression modeling and optimization for both accuracy and profit, and can be stepped through in the accompanying R Markdown file.

From the performed analyses, this report concludes that the bank's profit can be optimized by:

- Utilizing a logistic regression model that incorporates 23 out of 30 available variables, from loan term to the applicant's number of credit checks in the past six months;
- Setting a classification threshold of 32%, representing the probability that the applicant defaults on their loan; and
- Accepting loan applications from applicants below the threshold and denying applicants above the threshold.

The graph below shows how this criteria maximizes profit while maintaining accuracy:



Compared to a model which accepts all loans, the profit is increased by 238%, and about 77% of all loan applications will be accepted The overall accuracy of this model is 75%, only 3% below a model optimized for accuracy rather than profit. Limitations include: missing data in certain variables that have consequently been left out of the model, but are logically accounted for by other variables.

Setup: Load Packages

```
library(ggplot2)
library(ggformula)
library(gridExtra)
library(dplyr)
library(tidyr)
library(readr)
library(knitr)
```

Section 2: Introduction

Given a dataset containing information on 50,000 loans with 30 variables, the goal is to use logistic regression to predict which applicants are likely to default on their loans. I will do this by preparing and cleaning the dataset, exploring and transforming the data to eliminate skew, creating and optimizing a logistic model, and reporting my findings.

Section 3: Preparing and Cleaning the Data

The goal of this section is to explore, clean, and prepare the data for further analysis. Here is a sample of what the original data looks like. It consists of 50000 rows x 32 columns.

loanID	amount	term	rate	payment	grade	employment	length
188861	8000	36 months	0.14	272.07	C	Warehouseman	3 years
517703	11000	36 months	0.10	354.89	B	Vice President	10+ years
268587	35000	36 months	0.15	1220.33	D	Owner/Attorney	10+ years
579902	20000	60 months	0.12	447.83	C	Analyst	2 years
617630	12000	60 months	0.12	266.88	B	medical technician	10+ years

Preparing the response variable `status` by filtering out loans that aren't "Fully Paid", "Charged Off", or "Default", and refactoring the column into two factors: "Good" and "Bad".

Eliminating variables not useful as predictors from analysis: `loanID` is irrelevant to predicting loan payment; there are over 15,000 different titles and 1,918 `NA` values in `employment` which do not provide more information than income; `pubRec`, which is the number of derogatory public records, seems to be making a subjective judgment on what is termed "derogatory" and is unethical to use in this analysis.

Removing redundant variables: `totalRevLim`, `totalIlLim`, and `totalBcLim` are all some subset or variation of `totalLim` in terms of total credit limits; `openAcc` is correlated with `totalAcc` with a correlation coefficient of 0.687, so it will also be removed.

```
data <- data %>% dplyr::select(!c('loanID', 'employment', 'pubRec',
                                  'totalRevLim', 'totalIlLim',
                                  'totalBcLim', 'openAcc'))
```

Converting categorical variables to factors when appropriate and lumping small or indistinguishable categories together. "Source Verified" and "Verified" combined as categories in `verified`. "wedding" and "renewable_energy" lumped into "other" in `reason` as they have such small counts (2 and 27, respectively).

There are 759 missing values in the dataset, 384 in `bcRatio`, 360 in `bcOpen`, and only 15 in `revolRatio`. As these `NA`s all overlap, the rows with missing values make up only 1.11% of the dataset. Removing these rows with missing values makes more sense than potentially biasing the analysis by imputing values.

Additionally, there are 1796 "n/a"s in `length`. This is a large number of rows, so we would not automatically exclude them. Looking at a barchart of `length` by `status`, the distribution of "Good" and "Bad" loans is similar in "n/a" to most other categories, so we will include the "n/a"s in our analysis, along the lines of an "other" category.

```
data <- data %>% tidyr::drop_na(revolRatio) %>%
  tidyr::drop_na(bcOpen) %>%
  tidyr::drop_na(bcRatio)
```

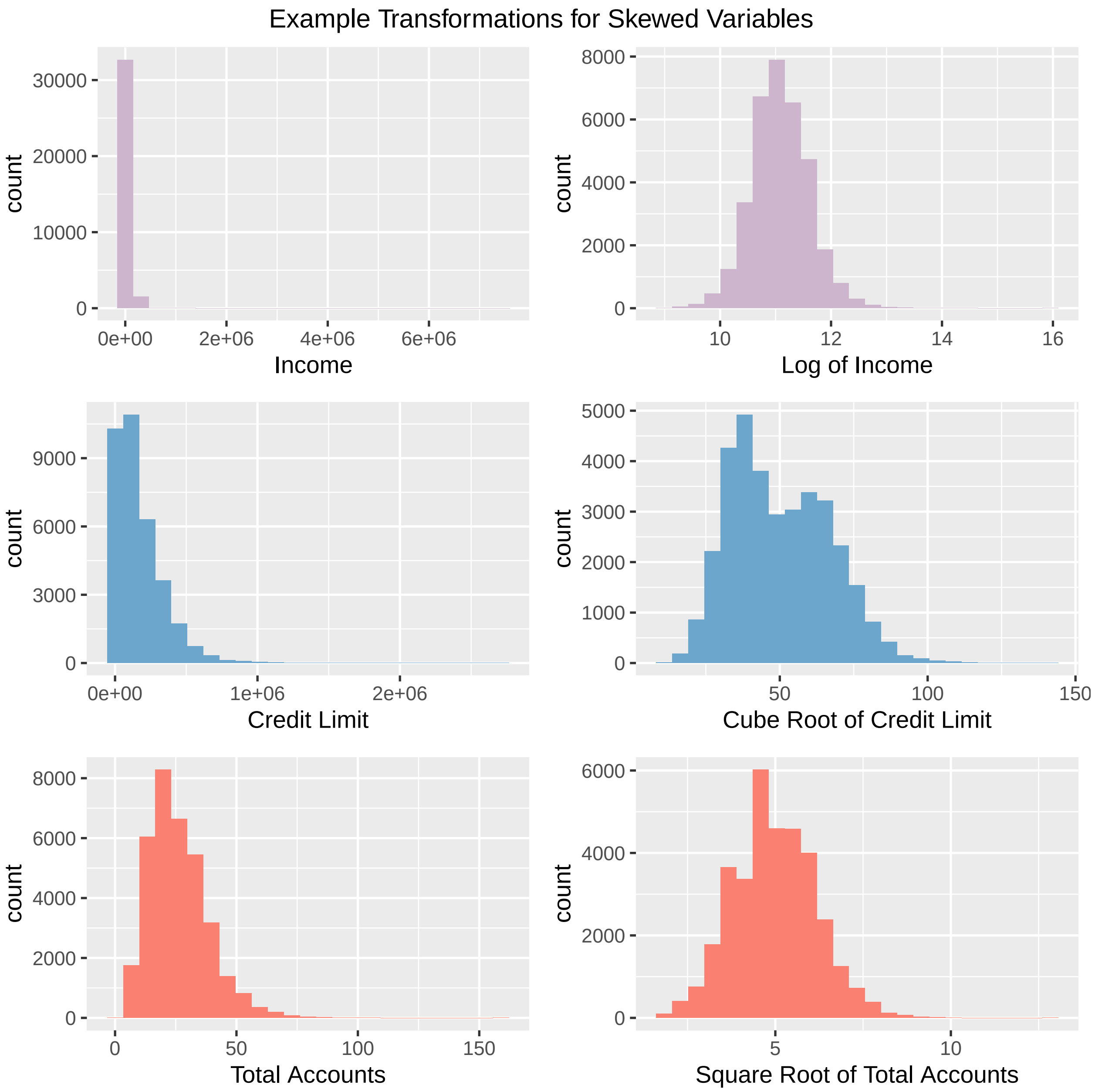
Section 4: Exploring and Transforming the Data

The purpose of this section is to explore and transform the dataset to prepare for the model. This involves creating graphs and visualizations of the data and variables.

The quantitative predictor variables are almost all right skewed, to varying degrees. The strongest skew is in `income` and `avgBal`. The next strongest skewed are `totalBal`, `totalLim`, and `totalRevBal`. The other variables with right skew that need transformation are `accOpen24` and `totalAcc`. The following transformations were applied in order to give the variable distributions a more normal shape and so that extreme values will have less influence on the model: logarithm to the most skewed, cube root the the next tier, and square root to the last tier.

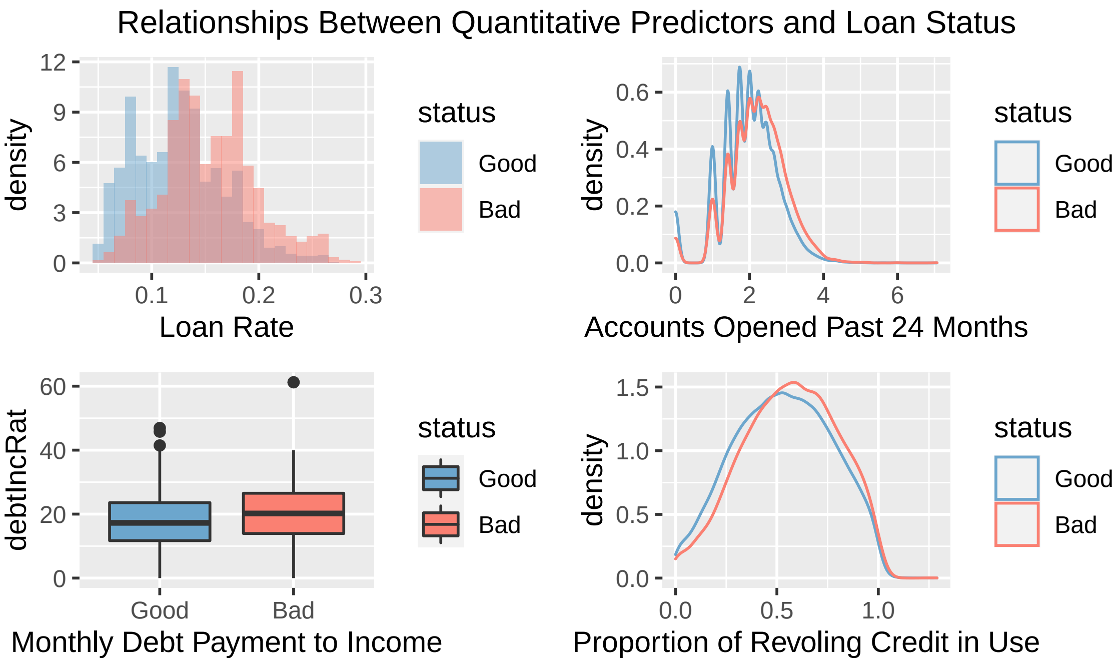
Examples from each tier of skewness, and how they were transformed:

```r
# Log transformation
h1 <- ggplot(data, aes(x=income)) +
  geom_histogram(bins=25, fill="thistle3") +
  labs(x="Income")
h2 <- ggplot(data, aes(x=log(income))) +
  geom_histogram(bins=25, fill="thistle3", size=.5) +
  labs(x="Log of Income")
# Cube root transformation
h3 <- ggplot(data, aes(x=totalLim)) +
  geom_histogram(bins=25, fill="skyblue3") +
  labs(x="Credit Limit")
h4 <- ggplot(data, aes(x=totalLim**(1/3))) +
  geom_histogram(bins=25, fill="skyblue3") +
  labs(x="Cube Root of Credit Limit")
# Square root transformation
h5 <- ggplot(data, aes(x=totalAcc)) +
  geom_histogram(bins=25, fill="salmon") +
  labs(x="Total Accounts")
h6 <- ggplot(data, aes(x=sqrt(totalAcc))) +
  geom_histogram(bins=25, fill="salmon") +
  labs(x="Square Root of Total Accounts")
grid.arrange(h1, h2, h3, h4, h5, h6, nrow=3,
             top="Example Transformations for Skewed Variables")
```

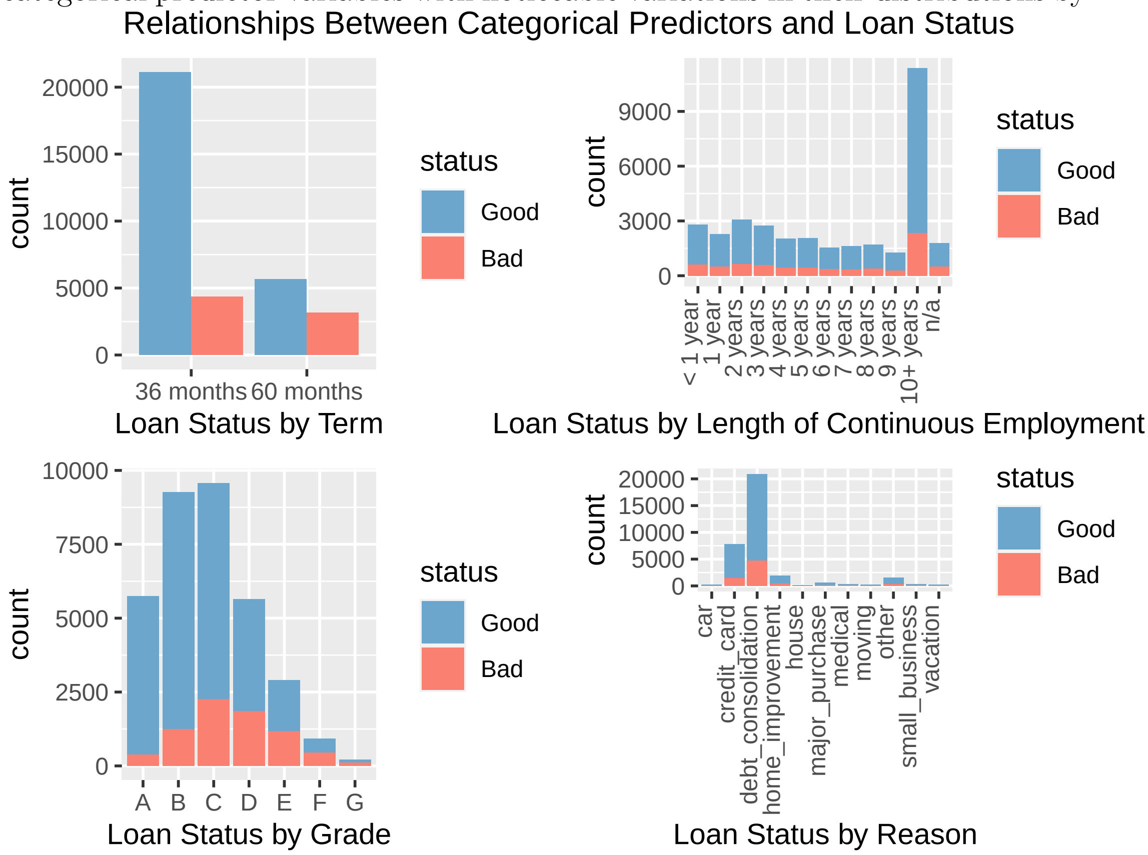


Example Transformations for Skewed Variables

Selection of quantitative predictor variables with noticeable variations in their distributions by loan status:



Selection of categorical predictor variables with noticeable variations in their distributions by loan status:



Of the quantitative predictors, `rate` has the strongest variation in behavior between "Good" and "Bad" loans. The categorical predictors have more obvious disparities between loan status. Specifically, the length of the loan `term` shows more "Good" loans at the 36 months level than at the 60 months level. Additionally, a continuous `length` of employment of 10+

years appears to be a good predictor of a “Good” loan.

Section 5: The Logistic Model

Creating two datasets, `train` and `test`, from the cleaned and prepared data using `sample` to randomly select 80% of the cases for `train` and the other 20% for `test`. Building logistic regression model from the `train` data, leaving out `totalPaid`, but including all other remaining variables as predictors.

```
set.seed(42)

sample_size = round(nrow(data)*.80)
index <- sample(seq(nrow(data)), size = sample_size)

train <- data[index, ]
test <- data[-index, ]

# Dropping 'totalPaid' from train data
train <- train %>% dplyr::select(!'totalPaid')

# Creating logistic regression model
model <- glm(status ~ ., data = train, family = 'binomial')
```

Using the test dataset along with the `model` and `predict()` to generate predicted statuses for each loan and to analyze the performance (accuracy) of the model. With the default threshold of 0.5 to classify the “Good” and “Bad” loans, creating a confusion matrix and determining the overall accuracy of the model (the percentage of correctly predicted outcomes), the percentage of actually good loans that are predicted as good, and the percentage of actually bad loans that are predicted as bad.

	Good=0	Bad=1	Sum
Good	5193	147	5340
Bad	1319	195	1514
Sum	6512	342	6854

Overall accuracy: 0.7861
Good loan accuracy: 0.7975
Bad loan accuracy: 0.5702

This model has an overall accuracy of 78.61%, which means that greater than 3/4 loans will be correctly classified as “Good” or “Bad”. This model is effective as compared to a strategy which might only correctly classify 50% of loans, or another strategy than simply accepts all loans. However, this model’s accuracy can be improved by varying the threshold it uses to classify a loan, which will be explored in the next section.

Section 6: Optimizing the Threshold for Accuracy

This section will investigate how changing the threshold affects the model predictions, and accuracy, when applied to the test data.

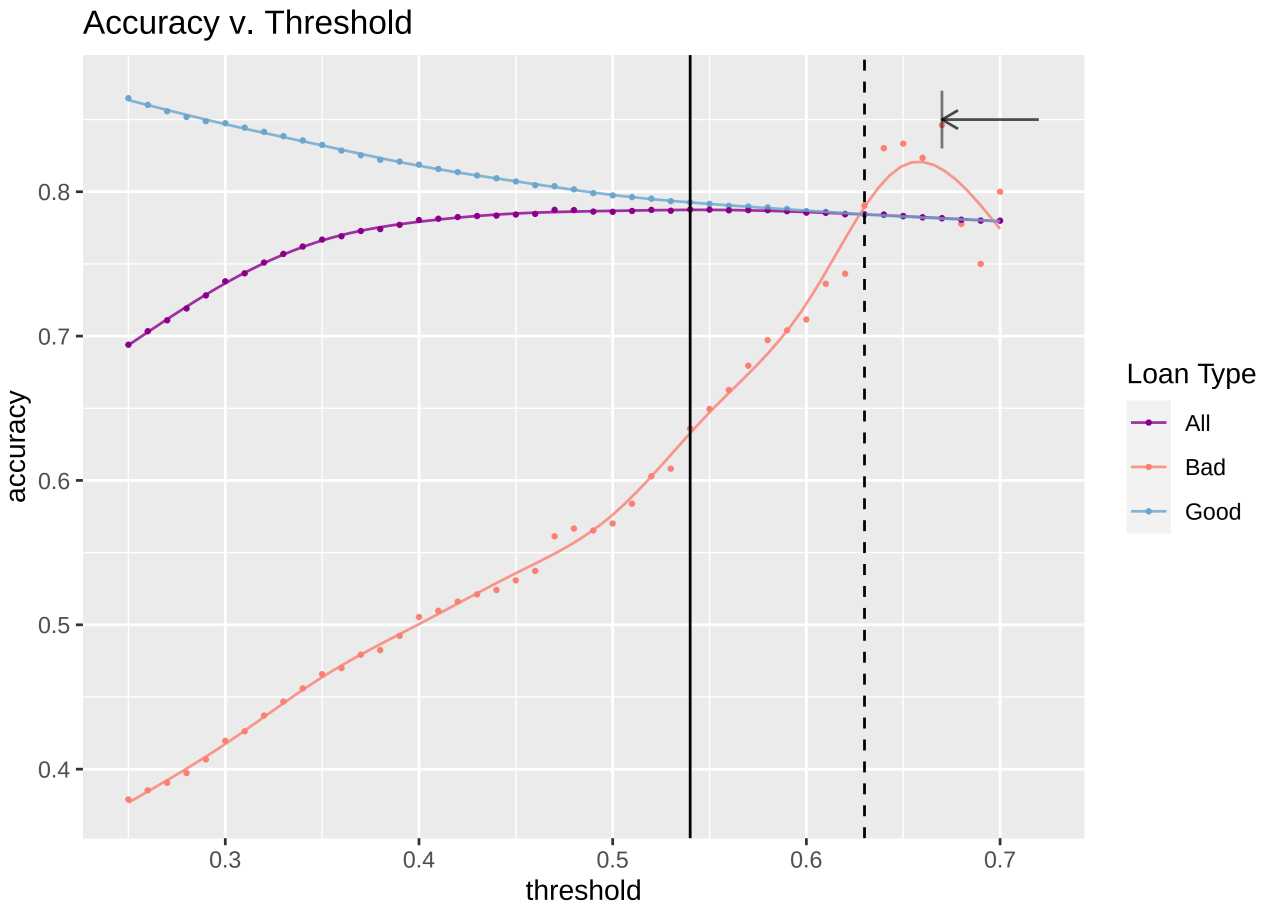
Experimenting with the classification threshold to change the proportions of loans that are predicted as good and bad.

```
thresholds <- seq(.25, .7, .01)
# Creating empty vectors to hold accuracies from different thresholds
acc.good <- c()
acc.bad <- c()
acc.all <- c()
for (val in thresholds) {
  pred_status <- cut(pred_prob, breaks=c(-Inf, val, Inf),
                labels=c("Good=0", "Bad=1"))  # Y=1 is "Bad" here
  cTab <- table(test$status, pred_status)
  cTab <- addmargins(cTab)

  good <- cTab[1,1] / cTab[3,1]
  bad <- cTab[2,2] / cTab[3,2]
  all <- (cTab[1,1] + cTab[2,2]) / cTab[3,3]
  # Store values
  acc.good <- c(acc.good, good)
  acc.bad <- c(acc.bad, bad)
  acc.all <- c(acc.all, all)
  }

acc.df <- data.frame(thresholds, acc.good, acc.bad, acc.all)
threshold_max_accuracy <- acc.df$thresholds[which.max(acc.df$acc.all)]
```

The table and plot below demonstrate how the proportion of loans correctly predicted varies.

Table 3: Proportion of loans correctly predicted (51 x 4)

thresholds	acc.good	acc.bad	acc.all
0.25	0.8647919	0.3789954	0.6940473
0.26	0.8601612	0.3853292	0.7033849
0.27	0.8557509	0.3905298	0.7109717
0.28	0.8518442	0.3973013	0.7191421
0.29	0.8488862	0.4067344	0.7281879
0.30	0.8474510	0.4196123	0.7379632

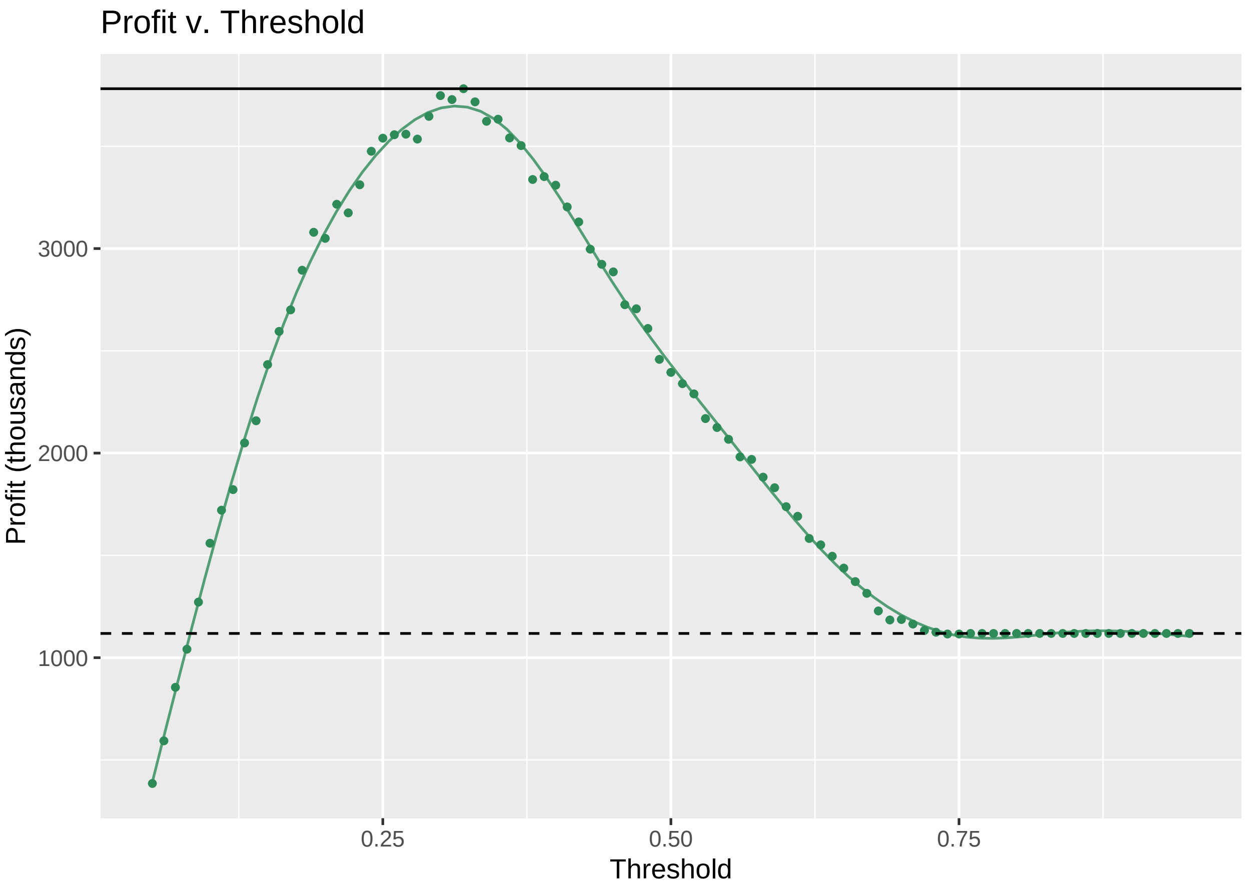


The maximum overall accuracy of the model on the test data is achieved at a 0.54 threshold. “Good” loans are predicted the most accurately at lower thresholds, whereas “Bad” loans are better predicted at the higher thresholds. However, the best overall accuracy is not at the threshold where the two lines meet, shown by the dashed black line on the plot. Instead, the most accurate threshold is slightly left, shown by the solid black line, because the majority of the loans are “Good” loans, as can be seen in the confusion matrix above.

Note on the plot: Higher thresholds mean that less loans are classified as “Bad”. What this means is that for thresholds over 0.67, the number of loans classified as “Bad” is very small (< 20 out of 6854). The arrow on the plot demonstrates the point where “Bad” loan accuracy starts to fall, but this is due to the small number of observations rather than a true decrease in accuracy. It can generally be assumed that “Bad” loan accuracy will increase at higher thresholds.

Section 7: Optimizing the Threshold for Profit

Repeating the threshold analysis of the previous section to find the value of the threshold that maximizes the profit. Applying the model to the test data and assuming the bank denies all of the loans that the model predicts as “Bad”.



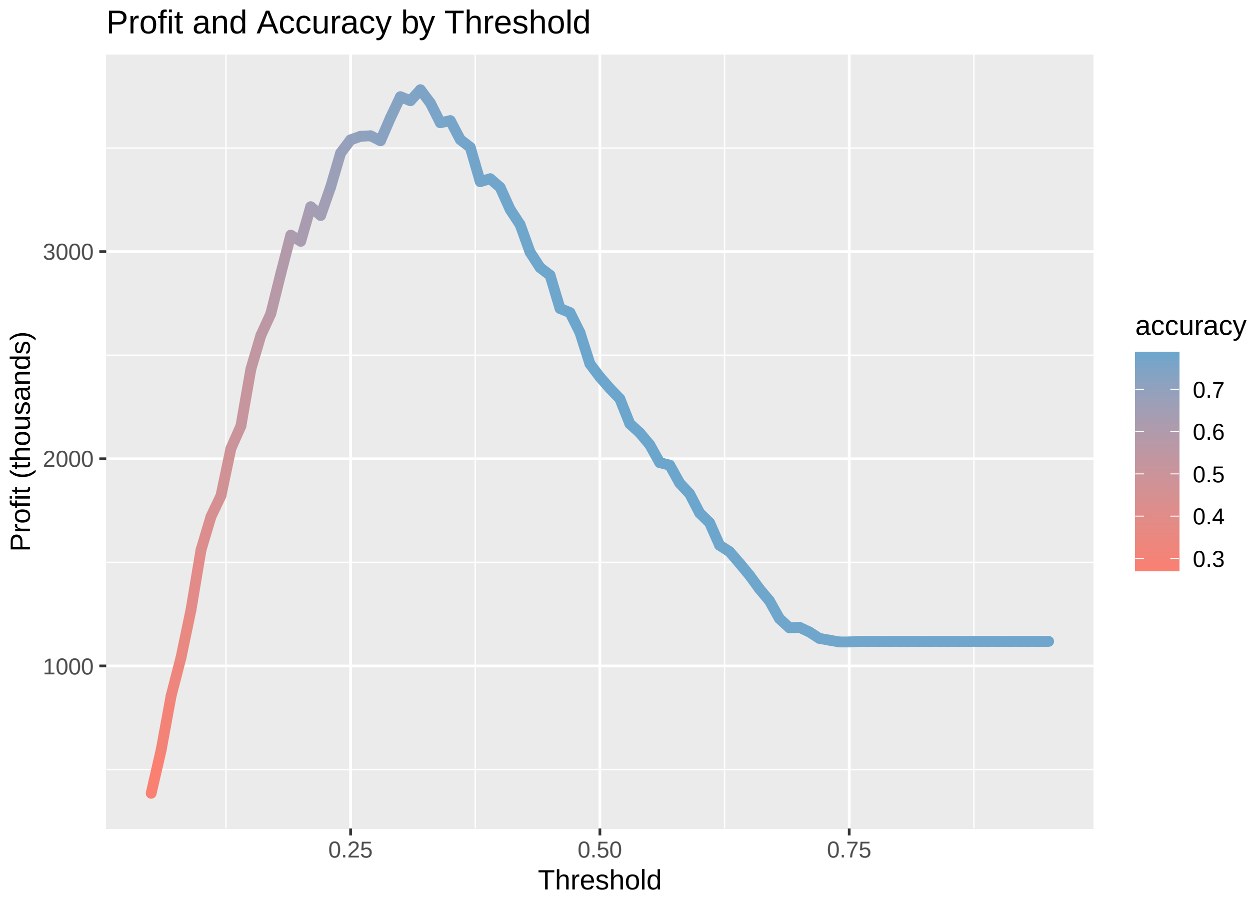
The total profit increases and then peaks at a threshold of 0.32 and profits of $3,781,349 (shown as a solid black line), before declining and eventually plateauing after a threshold of 0.76 and profits of $1,118,388.80, which is the same total profit a bank would make by accepting all loans (shown as a dashed black line).

The maximum percentage increase in profit that can be expected by deploying the model is 238.11%. Compared to the increase in profit from a perfect model that denies all of the truly bad loans, a 1004.96% increase, the perfect model would be a greater than 4x increase profit from our model.

For our model's best profit threshold, the overall accuracy is 75.09%; 84.15% of "Good" loans are correctly predicted, and 43.71% of "Bad" loans. The maximum profit threshold does not coincide with the maximum accuracy threshold, 0.32 and 0.54, respectively.

Section 8: Results Summary

The final classification model I suggest to the bank is based on maximizing profit rather than accuracy. With this assumption, the classification threshold to use is 0.32, where 0="Good" and 1="Bad" for loan status. This model is based on the predictor variables: `amount, term, rate, payment, grade, length, home, income, verified, reason, state, debtIncRat, delinq2yr, inq6mth, revolRatio, totalAcc, totalBal, accOpen24, avgBal, bcOpen, bcRatio, totalLim, totalRevBal.`



As the plot shows, there is certainly a tradeoff between accuracy and profit, but at the threshold for maximizing profit, the accuracy is still quite high. The overall profit of this model, based on the test data, is $3,781,349. The overall accuracy is 75.09%, the percent of “Good” loans correctly predicted is 84.15%, and the percent of “Bad” loans correctly predicted is 43.71%.