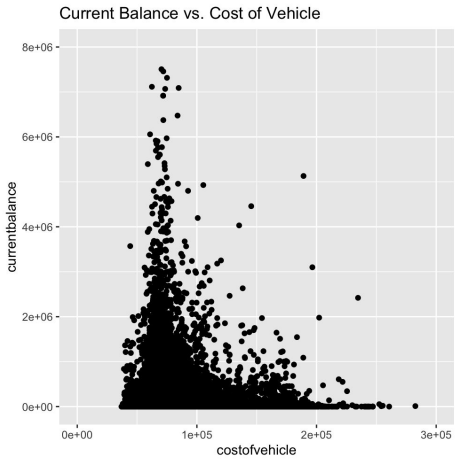
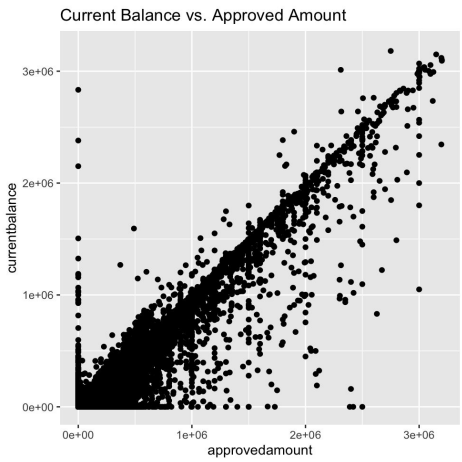
Dana DiVincenzo
3/2/21

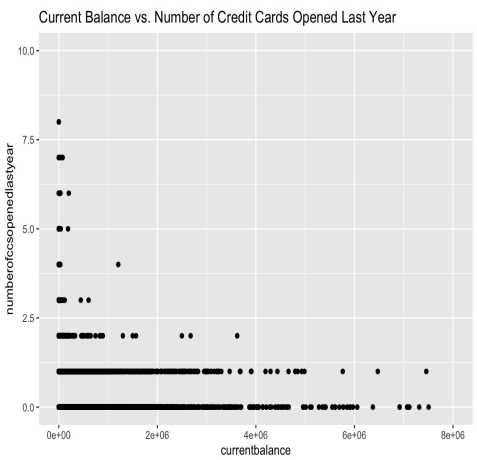
Electric Vehicle Repossession Model

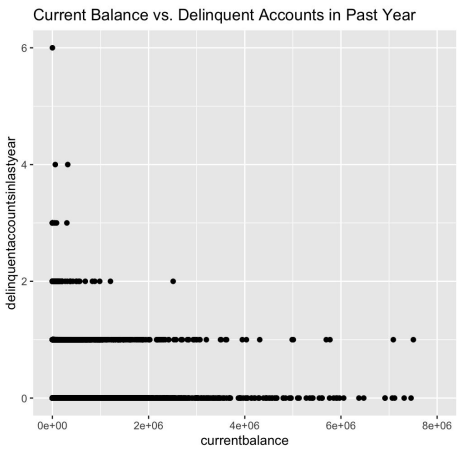
Electric vehicles are increasing in popularity, but are typically expensive. Because of this, many electric vehicle customers choose to lease their vehicle rather than purchase. However, customers who fail to complete their monthly payments will have their vehicles repossessed by the company. In this report, I analyze a dataset for a popular electric vehicle company, which contains information about thousands of their customers and whether or not the customer's vehicle was repossessed. The goal of this analysis is to better understand the factors that may lead to a vehicle repossession, and to create a predictive model that the company can use to decide if a potential customer is likely to have their vehicle repossessed. Using this insight, the company can minimize the amount of approved customers who have their vehicles repossessed, and also minimize the amount of unapproved customers who would not have had their vehicles repossessed, to maximize lease profits.

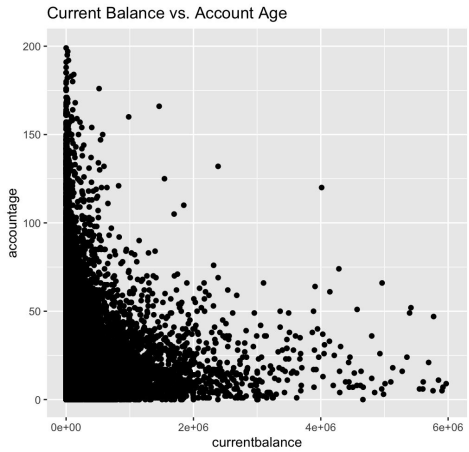
To start, I inspected some of the relationships within the data to gain insight to what may be most important in my model. I first created visualizations which compared a customer's current balance left on their loan to other related income factors about that customer. These visualizations are directly below.

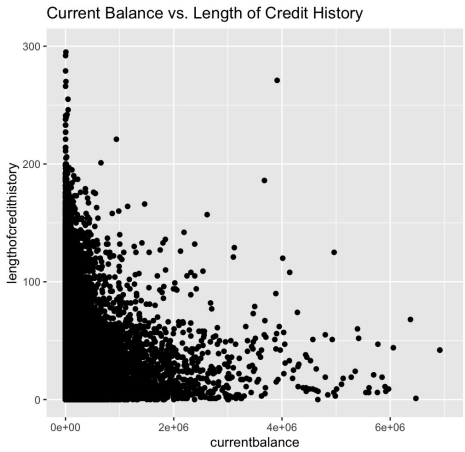


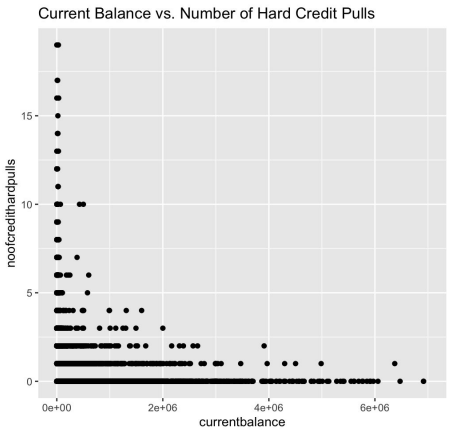


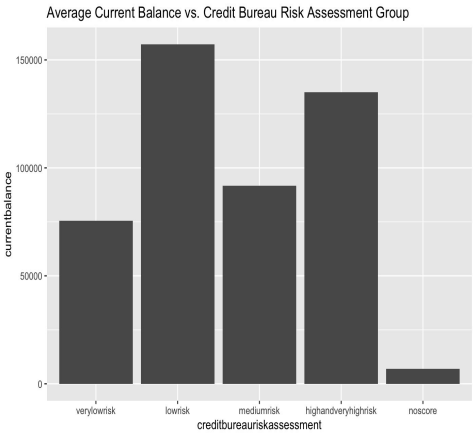


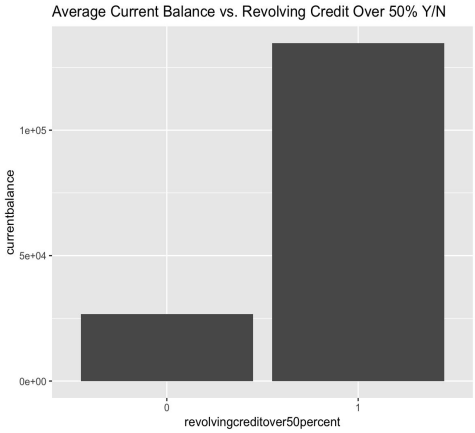


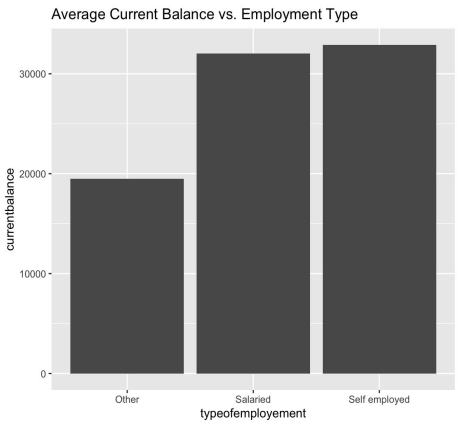


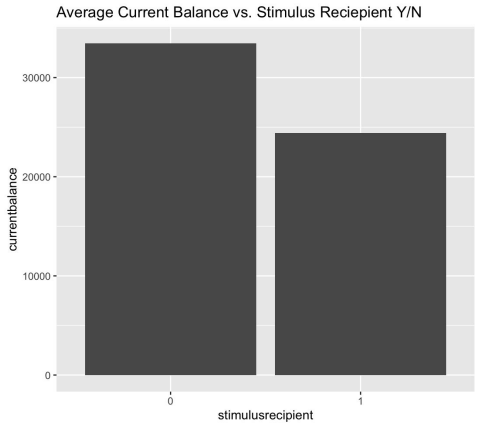








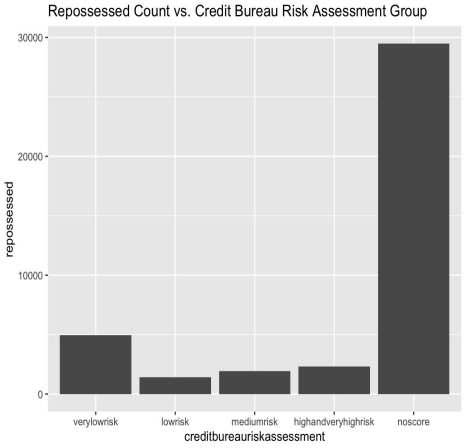


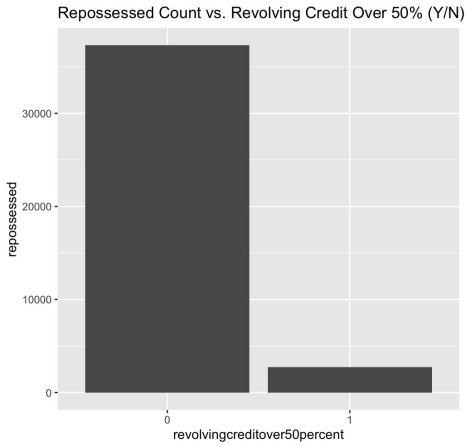


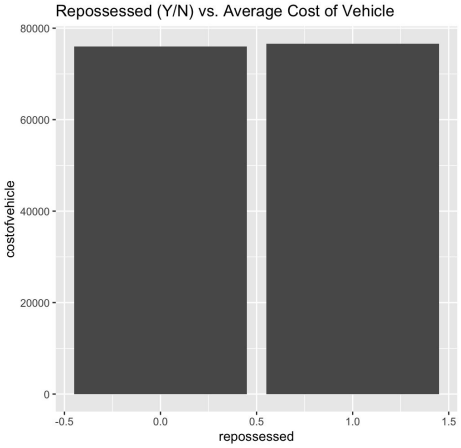
We can see that for variables shown with a scatter plot, only “Approved Amount,” the amount that was approved for the loan when financing the car, has a positive relationship with the current balance. All other scatter plot variables had a negative relationship with the current balance (Number of credit cards

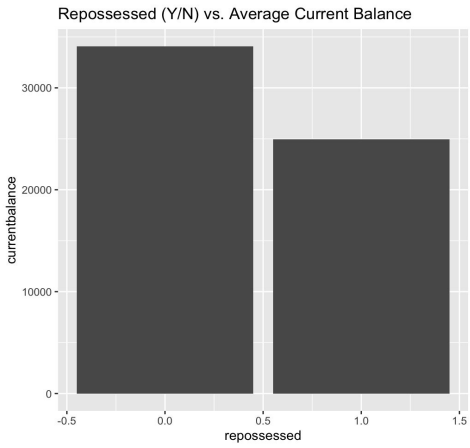
opened in the last year, delinquent accounts reported in the last year, the age of the account, the length of the customer's credit history, and the number of hard credit pulls on the customer's account.)

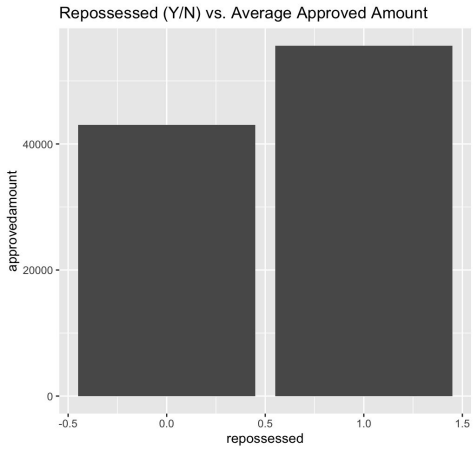
I then created plots which showed the repossessed variable against the factors that jumped out to me as possibly being important. These visualizations are directly below.

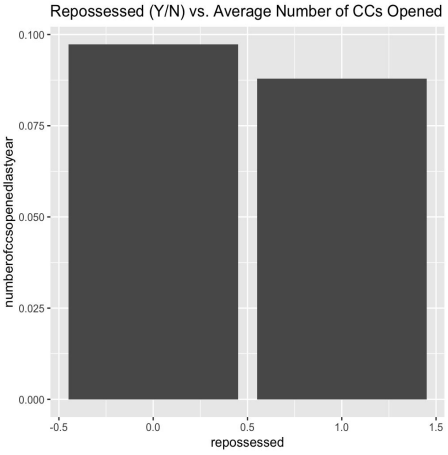


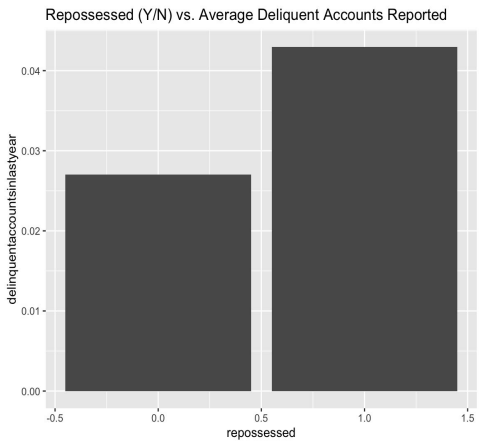


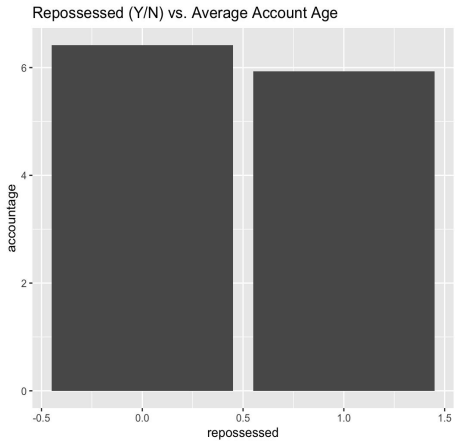


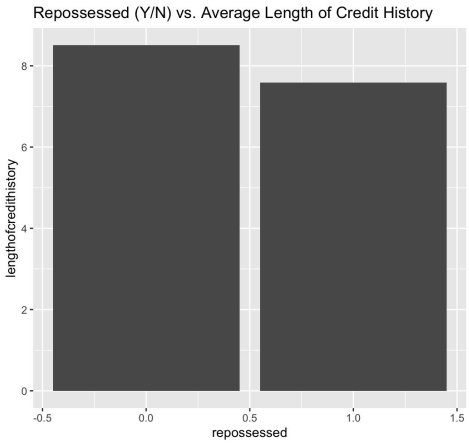


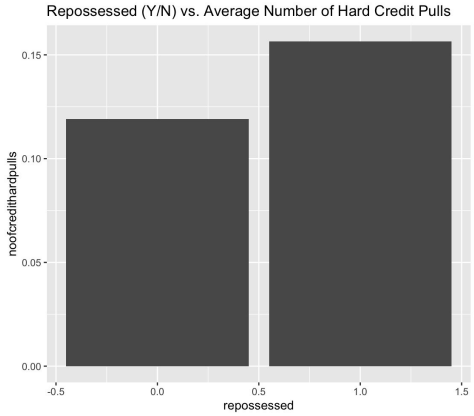












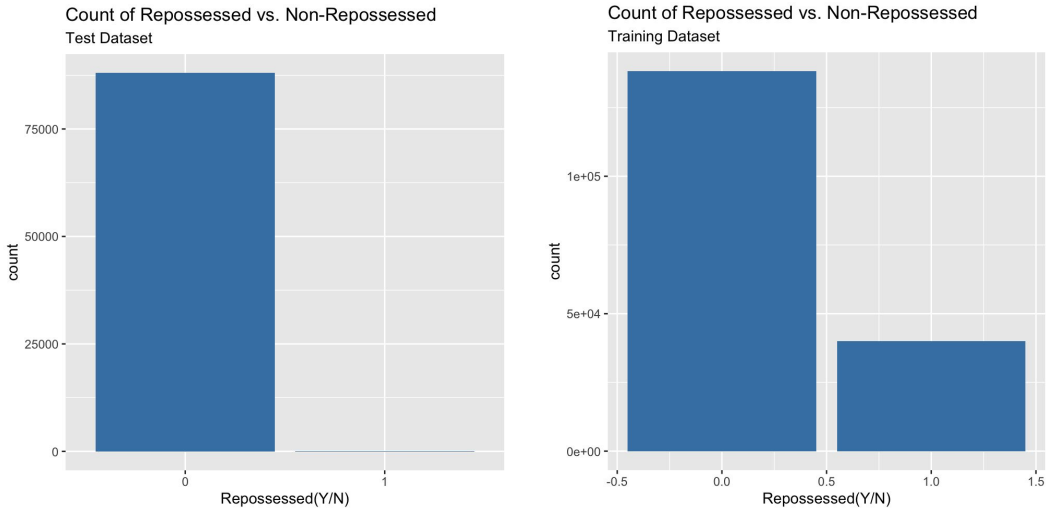
In the first graph above, we can see which of the groups of credit bureau risk assessment have the highest frequency of repossessions in the dataset. The “No score” group leads by far, with the “very low risk” group in second. The “low risk” group has the lowest count of repossessions. The second graph shows which of the groups of revolving credit over 50% (Y/N) have the highest frequency of repossessions. The “Yes” group has significantly fewer repossessions than the “No” group. All of the other graphs show the average of each variable for repossessed customers and non-repossessed customers. The variables with the biggest difference between the repossessed and non-repossessed groups are average current balance, average approved amount, average delinquent account reported, and average number of hard credit pulls. These variables may end up being some of the most significant in the predictive model.

Next, I created a logistic regression model to predict which customers are at risk of having their vehicle repossessed. After some tuning, I ended up with a model that uses the cost of the leased vehicle, the type of employment of the buyer, the reported level of credit risk, if the customer was using more than

50 percent of their available credit, the current balance left on the loan, the number of credit cards the customer opened in the last year, the number of reported delinquent accounts in the last year, the length of the customer's credit history, and the number of hard pulls on the credit check of the buyer, in order to predict the repossession risk. The model was found to be 77.59% accurate when tested on a sample of the dataset which hid the repossession results from the model until the predictions were complete. Cross validation was completed as well to ensure accuracy and stability of the model.

A summary of the model and an inspection of the odds ratio of the coefficients showed which of the variables in the model were significant, and what their effect would be on the likelihood of a repossession. All variables in the model were significant, except the customer's type of employment = salaried. Significant variables that increase the likelihood of repossession are: type of employment = self employed, credit bureau risk assessment score = low risk or medium risk or high and very high risk or no score (compared to low risk), using more than 50 percent of their available credit = yes, an increase in the number of delinquent accounts in the past year, and an increased number of credit hard pulls. The variable that decreases the likelihood of repossession is having an increased number of credit cards opened in the past year. The cost of the vehicle, the current balance remaining of the loan, and the length of the customers credit history all are significant to the model but do not change the likelihood of repossession.

Using my model, I was able to take a testing dataset which contained the data of potential customers and predict whether or not they would end up having their vehicle repossessed. The threshold of my model was set to be .5, so a customer who had a 50% or greater risk of having their vehicle repossessed was marked as a yes for repossession. Customers with less than a 50% risk of having their vehicle repossessed were marked as a no. This dataset will be attached along with the report. The model predicted that out of 88113 customers in the testing dataset, 50 would have their vehicle repossessed. This is a very different non-repossessed to repossessed ratio when compared to the customers in the original dataset. The visualization of that difference can be seen directly below.



Although the model had an acceptable level of accuracy and was able to make predictions, it still has limitations. One limitation of the model is that because there is no payoff formula, we cannot know which threshold to use to maximize payoff. A risk threshold of 50% was used because of the lack of information in this regard. However, if the company loses a significant amount of money when an approved customer's vehicle is repossessed, and does not lose a lot of money by turning away customers who would not have had a repossession, it would make sense to lower the risk threshold. It would also make sense to raise the threshold in the opposite scenario, where a repossession of an approved customer's vehicle costs the company little, but turning away any customers is a significant monetary loss. Unfortunately, because we do not know the costs and gains of true positives, true negatives, false positives, and false negatives, we can only speculate what the threshold should be. Another limitation is that there is demographic information we did not have access to that may increase the predictive power of the model. For example, the gender, age, and income of a customer may play a huge part in whether or not their vehicle gets repossessed. Because we did not have that information, our model may not be as accurate and as powerful as we need.

Code

```
library(ggplot2)
library(dplyr)
library(corrplot)
library(readr)
library(psych)
library(MASS)
library(tidyverse)


## READING IN THE TRAINING DATA AND MANIPULATING AND NEEDED

# dataframe containing the training data
traindf<-read.csv("/Users/dana/Downloads/Training dataset - electricvehicleloansTRAINdataset.csv")

# inspecting the data
head(traindf)
summary(traindf)
str(traindf, give.attr = FALSE) #displays string representation of data in each col

# changing dob to date type
traindf$dob <- gsub("(.*)/(..)$", "\\1/19\\2", traindf$dob)
traindf$dob <- as.Date(traindf$dob, "%d/%m/%Y")

# changing financedon to date type
traindf$financedon <- gsub("(.*)/(..)$", "\\1/19\\2", traindf$financedon)
traindf$financedon <- as.Date(traindf$financedon, "%d/%m/%Y")

head(traindf)

# formatting accountage into months
traindf$accountage <- gsub("([0-9]{1,2})yrs ([0-9]{1,2})mon", "\\1*12+\\2", traindf$accountage)
traindf$accountage <- sapply(sapply(traindf$accountage, FUN = function(x) parse(text = x), simplify =
TRUE, USE.NAMES = FALSE), eval)

#formatting lengthofcredithistory into months
# could have just done a substring, go back 2 digits from years
traindf$lengthofcredithistory <- gsub("([0-9]{1,2})yrs ([0-9]{1,2})mon", "\\1*12+\\2",
traindf$lengthofcredithistory)
traindf$lengthofcredithistory <- sapply(sapply(traindf$lengthofcredithistory, FUN = function(x)
parse(text = x), simplify = TRUE, USE.NAMES = FALSE), eval)

head(traindf)
```

```
str(traindf)
summary(traindf)

# removing any rows that do not contain a 0 or 1 in revolvingcreditover50percent and waspriorcustomer
unique(traindf[,"revolvingcreditover50percent"])
backup <- traindf[traindf$revolvingcreditover50percent == 0 | traindf$revolvingcreditover50percent ==
1,]
unique(backup[,"revolvingcreditover50percent"])

unique(backup[,"waspriorcustomer"])
traindf <- backup[backup$waspriorcustomer == 0 | backup$waspriorcustomer == 1,]
unique(traindf[,"waspriorcustomer"])

str(traindf, give.attr = FALSE) #displays string representation of data in each col

# making all non-continuous variables into factor types, besides the dependent variable
traindf$showroomid <- as.factor(traindf$showroomid)
traindf$modelversioncode <- as.factor(traindf$modelversioncode)
traindf$typeofemployement <- as.factor(traindf$typeofemployement)
traindf$residencestate <- as.factor(traindf$residencestate)
traindf$stimulusrecipient <- as.factor(traindf$stimulusrecipient)
traindf$creditbureauriskassessment <- as.factor(traindf$creditbureauriskassessment)
traindf$waspriorcustomer <- as.factor(traindf$waspriorcustomer)
traindf$revolvingcreditover50percent <- as.factor(traindf$revolvingcreditover50percent)

str(traindf, give.attr = FALSE) #displays string representation of data in each col

library(RColorBrewer)
# this makes a histogram showing the count of how many customers have or have not
# had their car repossessed in the dataset (1 means yes, 0 means no)
# lets make this graph look pretty
ggplot(traindf, aes(x= repossessed)) + geom_histogram(stat = "count") + labs(title = "Count of
Repossessed vs. Non-Repossessed",
                                                    subtitle = "Training Dataset", x = "Repossessed(Y/N)") +
geom_bar(fill = "steelblue")


# combining levels of creditbureauriskassessment into 5 groups
levels(traindf$creditbureauriskassessment) <- list(verylowrisk = c("A-Very Low Risk", "D-Very Low
Risk", "C-Very Low Risk", "B-Very Low Risk"),
                                   lowrisk = c("E-Low Risk", "F-Low Risk", "G-Low Risk"),
                                   mediumrisk = c("I-Medium Risk", "H-Medium Risk"),
                                   highandveryhighrisk = c("K-High Risk", "J-High Risk", "L-Very High
Risk", "M-Very High Risk"),
```

```
                                    noscore = c("No Bureau History Available", "Not Scored: Not Enough
Info available on the customer",
                                                "Not Scored: No Activity seen on the customer (Inactive)", "Not
Scored: Sufficient History Not Available",
                                                "Not Scored: No Updates available in last 36 months", "Not
Scored: Only a Guarantor"))

traindf$typeofemployement <- sub("^$", "Other", traindf$typeofemployement)

unique(traindf[,"creditbureauriskassessment"])
str(traindf, give.attr = FALSE) #displays string representation of data in each col

### MAKING VISUALIZATIONS

# making a scatterplot of currentbalance vs costofvehicle
ggplot(traindf, aes(x=costofvehicle, y=currentbalance)) + geom_point() +xlim(0, 300000) + ylim(0,
8000000) + labs(title = "Current Balance vs. Cost of Vehicle")

# making a scatterplot of currentbalance vs approvedamount
ggplot(traindf, aes(x=approvedamount, y=currentbalance)) + geom_point() +xlim(0, 3200000) + ylim(0,
3200000) + labs(title = "Current Balance vs. Approved Amount")

#making a scatterplot of currentbalance vs numberofccsopenedlastyear
ggplot(traindf, aes(x=currentbalance, y=numberofccsopenedlastyear)) + geom_point() +xlim(0, 8000000)
+ ylim(0,10) + labs(title = "Current Balance vs. Number of CCs Opened Last Year")

# #making a scatterplot of currentbalance vs delinquentaccountsinlastyear
ggplot(traindf, aes(x=currentbalance, y=delinquentaccountsinlastyear)) + geom_point() +xlim(0,
8000000) + labs(title = "Current Balance vs. Delinquent Accounts in Past Year")

# #making a scatterplot of currentbalance vs accountage
ggplot(traindf, aes(x=currentbalance, y=accountage)) + geom_point() +xlim(0, 6000000) + ylim(0, 200) +
labs(title = "Current Balance vs. Account Age")

# #making a scatterplot of currentbalance vs lengthofcredithistory
ggplot(traindf, aes(x=currentbalance, y=lengthofcredithistory)) + geom_point() +xlim(0, 7000000) +
ylim(0, 300) + labs(title = "Current Balance vs. Length of Credit History")

# #making a scatterplot of currentbalance vs noofcredithardpulls
ggplot(traindf, aes(x=currentbalance, y=noofcredithardpulls)) + geom_point() +xlim(0, 7000000) +
labs(title = "Current Balance vs. Number of Hard Credit Pulls")

# making a bar graph which shows the average currentbalance for each category of
creditbureauriskassessment
```

```
ggplot(traindf) + aes(x = creditbureauriskassessment, y = currentbalance) + stat_summary(geom = "bar", fun.y = "mean") + labs(title = "Average Current Balance vs. Credit Bureau Risk Assessment Group")

# making a bar graph which shows the average currentbalance for each category of revolvingcreditover50percent
ggplot(traindf) + aes(x = revolvingcreditover50percent, y = currentbalance) + stat_summary(geom = "bar", fun.y = "mean") + labs(title = "Average Current Balance vs. Revolving Credit Over 50% Y/N")

# making a bar graph which shows the average currentbalance for each category of typeofemployement
ggplot(traindf) + aes(x = typeofemployement, y = currentbalance) + stat_summary(geom = "bar", fun.y = "mean") + labs(title = "Average Current Balance vs. Employment Type")

# making a bar graph which shows the average currentbalance for each category of stimulusrecipient
ggplot(traindf) + aes(x = stimulusrecipient, y = currentbalance) + stat_summary(geom = "bar", fun.y = "mean") + labs(title = "Average Current Balance vs. Stimulus Reciepient Y/N")


# make graphs comparing repossessed with the variables I think will be important
# costofvehicle (int)
# creditbureauriskassessment (factor)
# revolvingcreditover50percent (factor)
# currentbalance (int)
# approvedamount (int)
# numberofccsopenedlastyear (int)
# delinquentaccountsinlastyear (int)
# accountage (int)
# lengthofcredithistory (int)
# noofcredithardpulls (int)

# bar graph to show the number of repossessed in each category of creditbureauriskassessment
ggplot(data=traindf) +
  aes(x=creditbureauriskassessment, y=repossessed) +
  geom_bar(stat="identity") + labs(title = "Repossessed Count vs. Credit Bureau Risk Assessment Group")

# bar graph to show the number of repossessed in each category of revolvingcreditover50percent
ggplot(data=traindf) +
  aes(x=revolvingcreditover50percent, y=repossessed) +
  geom_bar(stat="identity") + labs(title = "Repossessed Count vs. Revolving Credit Over 50% (Y/N)")


# bar chart to show the average cost of vehicle for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=costofvehicle) +
```

```
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Cost of
Vehicle")

# bar chart to show the average currentbalance for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=currentbalance) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Current
Balance")

# bar chart to show the average approvedamount for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=approvedamount) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Approved
Amount")

# bar chart to show the average numberofccsopenedlastyear for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=numberofccsopenedlastyear) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Number of
CCs Opened")

# bar chart to show the average delinquentaccountsinlastyear for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=delinquentaccountsinlastyear) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Deliquent
Accounts Reported")

# bar chart to show the average accountage for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=accountage) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Account
Age")

# bar chart to show the average lengthofcredithistory for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=lengthofcredithistory) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Length of
Credit History")

# bar chart to show the average noofcredithardpulls for repossessed and non-repossessed cars
ggplot(data=traindf) +
  aes(x=repossessed, y=noofcredithardpulls) +
  geom_bar(stat = "summary", fun.y = "mean") + labs(title = "Repossessed (Y/N) vs. Average Number of
Hard Credit Pulls")
```

MAKING AND ADJUSTING THE MODEL

```
# Making the initial regression model with every independent variable
trainmodel <- glm(repossessed ~ costofvehicle + showroomid + modelversioncode + dob +
                    typeofemployement + financedon + residencestate + stimulusrecipient +
creditbureauriskassessment
                  + noofcarsleasedprior + waspriorcustomer +revolvingcreditover50percent +
                    currentbalance + approvedamount + numberofccsopenedlastyear +
delinquentaccountsinlastyear
                  + accountage +  lengthofcredithistory + noofcredithardpulls, family = binomial,
                     data = traindf)

# way too many things are in this model
# especially becayse some of the factors like "showroom id" and "stateid" have so many levels
summary(trainmodel)

# stepAIC performs stepwise model selection by AIC
# this model still had some unnecessary variables in it
trainAIC <- stepAIC(trainmodel, trace=0)
summary(trainAIC)

# making a new model based on the trainAIC model and what I think is important
# took out tid, showroomid, modelversioncode, dob, fincancedon
trainAICadj <- glm(repossessed ~ costofvehicle +
                    typeofemployement + stimulusrecipient + creditbureauriskassessment
                  + noofcarsleasedprior +revolvingcreditover50percent +
                    currentbalance + approvedamount + numberofccsopenedlastyear +
delinquentaccountsinlastyear
                  + accountage +  lengthofcredithistory + noofcredithardpulls, family = binomial,
                  data = traindf)

# this model seems better, but I believe I could still remove some more variables without compromising
accuracy
summary(trainAICadj)

library(SDMTools)

# now let's test the accuracy of the models above and see if we need to adjust anything
# doing it on the AICadj model
traindf$predAICadj <- predict(trainAICadj, type = "response", na.action = pass)


#building the confusion matrix
```

```
# confusion.matrix Calculates a cross-tabulation of observed and predicted classes with associated
statistics
confMatrixModelAICadj <- confusion.matrix(traindf$repossessed,traindf$predAICadj, threshold = .5)
confMatrixModelAICadj

#calculating the accuracy of the model
# accuracy was found to be .7753949
# lets take out more variables and see what happens
accuracyAICadj <- sum(diag(confMatrixModelAICadj)) / sum(confMatrixModelAICadj)
accuracyAICadj

# removing noofcarsleasedprior and approvedamount from model trainAICadj and trying again
trainAICadj2 <- glm(repossessed ~ costofvehicle +
                      typeofemployement + stimulusrecipient + creditbureauriskassessment
                    +revolvingcreditover50percent +
                      currentbalance + numberofccsopenedlastyear + delinquentaccountsinlastyear
                    + accountage +  lengthofcredithistory + noofcredithardpulls, family = binomial,
                    data = traindf)

# this doesnt affect the accuracy or R2 much
summary(trainAICadj2)

traindf$predAICadj2 <- predict(trainAICadj2, type = "response", na.action = na.exclude)

#building the confusion matrix
# confusion.matrix Calculates a cross-tabulation of observed and predicted classes with associated
statistics
confMatrixModelAICadj2 <- confusion.matrix(traindf$repossessed,traindf$predAICadj2, threshold = .5)
confMatrixModelAICadj2

#calculating the accuracy of the model
# accuracy was found to be 0.7753669, which is almost exactly the same as before, but now we are using
less variables
accuracyAICadj2 <- sum(diag(confMatrixModelFull2)) / sum(confMatrixModelFull2)
accuracyAICadj2

# removing stimulusrecipient and accountage from trainAICadj2
trainAICadj3 <- glm(formula = repossessed ~ costofvehicle +
                      typeofemployement +
                      creditbureauriskassessment  +
                      revolvingcreditover50percent + currentbalance  +
                      numberofccsopenedlastyear + delinquentaccountsinlastyear +
                      lengthofcredithistory + noofcredithardpulls,
                    family = binomial, data = traindf)
```

```
summary(trainAICadj3)

traindf$predAICadj3 <- predict(trainAICadj3, type = "response", na.action = na.exclude)

#building the confusion matrix
# confusion.matrix Calculates a cross-tabulation of observed and predicted classes with associated
statistics
confMatrixModelAICadj3 <- confusion.matrix(traindf$repossessed,traindf$predAICadj3, threshold = .5)
confMatrixModelAICadj3

#calculating the accuracy of the model
# accuracy was found to be 0.7754286, which is even better than the last one
# this will be our final model
accuracyAICadj3 <- sum(diag(confMatrixModelAICadj3)) / sum(confMatrixModelAICadj3)
accuracyAICadj3

head(traindf)
# this finds the coefficients of the model in terms of odds (the exp() makes it in terms of odds)
# so as each variable increases, the odds of the dependent variable increase by the associated coeff
# round(2) rounds it to 2 decimal places
coefsexp <- coef(trainAICadj3) %>% exp() %>% round(2)
coefsexp

library(MASS)
library(poliscidata)

# these are nice to look at but not that relevent
logregR2(trainAICadj) #R^2 is .011
logregR2(trainAICadj2) #R^2 is .011
logregR2(trainAICadj3) #R^2 is .009
logregR2(trainmodel) #R^2 is .025
logregR2(trainAIC) #R^2 is .025


######## NOW LETS ADD IN THE TEST DATASET - WILL HAVE TO EDIT THE DATAFRAME
THE SAME WAY AS BEFORE


# dataframe containing the training data
testdf<-read.csv("/Users/dana/Downloads/testing data - electricvehicleloanstestingdataset.csv")
head(testdf)
# inspecting the data
```

```
# changing dob to date type
testdf$dob <- gsub("(.*)/(..)$", "\\1/19\\2", testdf$dob)
testdf$dob <- as.Date(testdf$dob, "%d/%m/%Y")

# changing financedon to date type
testdf$financedon <- gsub("(.*)/(..)$", "\\1/19\\2", testdf$financedon)
testdf$financedon <- as.Date(testdf$financedon, "%d/%m/%Y")

# formatting accountage into months
testdf$accountage <- gsub("([0-9]{1,2})yrs ([0-9]{1,2})mon", "\\1*12+\\2", testdf$accountage)
testdf$accountage <- sapply(sapply(testdf$accountage, FUN = function(x) parse(text = x), simplify =
TRUE, USE.NAMES = FALSE), eval)

#formatting lengthofcredithistory into months
# could have just done a substring, go back 2 digits from years
testdf$lengthofcredithistory <- gsub("([0-9]{1,2})yrs ([0-9]{1,2})mon", "\\1*12+\\2",
testdf$lengthofcredithistory)
testdf$lengthofcredithistory <- sapply(sapply(testdf$lengthofcredithistory, FUN = function(x) parse(text =
x), simplify = TRUE, USE.NAMES = FALSE), eval)

head(testdf)
str(testdf)
summary(testdf)

# removing any rows that do not contain a 0 or 1 in revolvingcreditover50percent and waspriorcustomer
unique(testdf[,"revolvingcreditover50percent"])
backup <- testdf[testdf$revolvingcreditover50percent == 0 | testdf$revolvingcreditover50percent == 1,]
unique(backup[,"revolvingcreditover50percent"])

unique(backup[,"waspriorcustomer"])
testdf <- backup[backup$waspriorcustomer == 0 | backup$waspriorcustomer == 1,]
unique(testdf[,"waspriorcustomer"])

str(testdf, give.attr = FALSE) #displays string representation of data in each col

# making all non-continuous variables into factor types, besides the dependent variable
testdf$showroomid <- as.factor(testdf$showroomid)
testdf$modelversioncode <- as.factor(testdf$modelversioncode)
testdf$typeofemployement <- as.factor(testdf$typeofemployement)
testdf$residencestate <- as.factor(testdf$residencestate)
testdf$stimulusrecipient <- as.factor(testdf$stimulusrecipient)
testdf$creditbureauriskassessment <- as.factor(testdf$creditbureauriskassessment)
testdf$waspriorcustomer <- as.factor(testdf$waspriorcustomer)
testdf$revolvingcreditover50percent <- as.factor(testdf$revolvingcreditover50percent)
```

```
levels(testdf$creditbureauriskassessment) <- list(verylowrisk = c("A-Very Low Risk", "D-Very Low Risk", "C-Very Low Risk", "B-Very Low Risk"),
                                                  lowrisk = c("E-Low Risk", "F-Low Risk", "G-Low Risk"),
                                                  mediumrisk = c("I-Medium Risk", "H-Medium Risk"),
                                                  highandveryhighrisk = c("K-High Risk", "J-High Risk", "L-Very High Risk", "M-Very High Risk"),
                                                  noscore = c("No Bureau History Available", "Not Scored: Not Enough Info available on the customer",
                                                              "Not Scored: No Activity seen on the customer (Inactive)", "Not Scored: Sufficient History Not Available",
                                                              "Not Scored: No Updates available in last 36 months", "Not Scored: Only a Guarantor"))
testdf$typeofemployement <- sub("^$", "Other", testdf$typeofemployement)

str(testdf, give.attr = FALSE) #displays string representation of data in each col

# now making a prediction column for repossessed using trainAICadj3 and the testdf


#prediction <- predict(trainAICadj3, newdata = testdf, type = "response")

# I need to set threshold = .5, so those with a value of above .5 will be predicted to get repossessed
# we dont have any type of formula that would allow us to find the best threshold to maximize profit
# (eg. we don't know what the cost is of a false positive and we don't know the payoff of a true positive, etc)

# Here Im making the predictions and setting the threshold at once
testdf$repossessedpred <- as.factor(ifelse(predict(trainAICadj3, newdata = testdf,
type="response")>=0.5,"1","0"))
head(testdf, n =10)

# comparing the two prediction lines to make sure the code worked, it did
table(testdf$repossessed == 1)
# need to cross validate - not sure if I do this here or on my training model

# why is this so low? Probably because the threshold needs to be lower?
# he says to talk about the similarities and differences, this could be a good example
ggplot(testdf, aes(x= repossessedpred)) + geom_histogram(stat = "count") + labs(title = "Count of Repossessed vs. Non-Repossessed",
                                                                                 subtitle = "Test Dataset", x = "Repossessed(Y/N)") +
geom_bar(fill = "steelblue")

###### DOING THE CROSSVALIDATION ON THE TRAINING DATASET
```

```
trainformula <- as.formula(summary(trainAICadj3)$call)
trainformula

set.seed(534381)

traindf$isTrain <- rbinom(nrow(traindf), 1, 0.7)
train <- subset(traindf, traindf$isTrain == 1)
test <- subset(traindf, traindf$isTrain  == 0)

# Use the saved formula and run the new model (from our previous tuning) on the test data and
# predict using the coefficients from this result with the test dataset
# trainlogit makes a model, like before, using our saved formula, and the train dataset
trainlogit <- glm(trainformula, family = binomial, data = train)

#adding a new column to the test dataset called predNew
#predicting based on new model we just created, testing on created test dataset which is a subset of traindf
test$predrep <- predict(trainlogit, type = "response", newdata = test) # Prediction

# we found the accuracy to be .7758714
# this is good, our accuracy on the whole model was 0.7754286
confMatrixModelTT <- confusion.matrix(test$repossessed, test$predrep, threshold = 0.5)
sum(diag(confMatrixModelTT)) / sum(confMatrixModelTT)  # the sum of the diagonals of the matrix
divided by the sum

library(boot)

# a function that returns the accuracy
costAcc <- function(r, pi = 0) {
  cm <- confusion.matrix(r, pi, threshold = 0.5) # creates a confusion matrix
  acc <- sum(diag(cm)) / sum(cm) # returning the accuracy, sum of diagonals of matrix divided by the
sum
  return(acc)
}

# Step 8 - Recalculate the cross validated accuracy with the cv.glm function and review
set.seed(534381)

# This function calculates the estimated K-fold cross-validation prediction error for generalized linear
models.
# costAcc function is the accuracy we calculated above
# K is The number of groups into which the data should be split to estimate the cross-validation prediction
error
# $delta[1] is what the output is on the first round of running the glm = 0.7754061
```

```
### the results for our training dataset are very compatable verifying the stability of our chosen model
# A vector of length two. The first component is the raw cross-validation estimate of prediction error.
# The second component is the adjusted cross-validation estimate.
cv.glm(traindf, trainAICadj3, cost = costAcc, K = 10)$delta[1]

#### Saving the test file with the new predicted column

head(testdf)

write.csv(testdf,"/Users/dana/Downloads/MKTG768 - Assignment1 Prediction.csv", row.names =
FALSE)
```