Date: 05/31/2021

BAN 502-802

Course: Predictive Analytics Author: Gregory Carmichael

Assignment: Module 3 Logistic Regression

Load required libraries

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3 v purrr 0.3.4  
## v tibble 3.1.1 v dplyr 1.0.5  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(tidymodels)

## -- Attaching packages -------------------------------------- tidymodels 0.1.3 --

## v broom 0.7.6 v rsample 0.1.0   
## v dials 0.0.9 v tune 0.1.5   
## v infer 0.5.4 v workflows 0.2.2   
## v modeldata 0.1.0 v workflowsets 0.0.2   
## v parsnip 0.1.5 v yardstick 0.0.8   
## v recipes 0.1.16

## -- Conflicts ----------------------------------------- tidymodels\_conflicts() --  
## x scales::discard() masks purrr::discard()  
## x dplyr::filter() masks stats::filter()  
## x recipes::fixed() masks stringr::fixed()  
## x dplyr::lag() masks stats::lag()  
## x yardstick::spec() masks readr::spec()  
## x recipes::step() masks stats::step()  
## \* Use tidymodels\_prefer() to resolve common conflicts.

library(e1071)

##   
## Attaching package: 'e1071'

## The following object is masked from 'package:tune':  
##   
## tune

## The following object is masked from 'package:rsample':  
##   
## permutations

library(ROCR)

#Prework task Part 1

Load data from the parole.csv file.

parole = read\_csv("parole.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## male = col\_double(),  
## race = col\_double(),  
## age = col\_double(),  
## state = col\_double(),  
## time.served = col\_double(),  
## max.sentence = col\_double(),  
## multiple.offenses = col\_double(),  
## crime = col\_double(),  
## violator = col\_double()  
## )

Structure and summary

str(parole)

## spec\_tbl\_df[,9] [675 x 9] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ male : num [1:675] 1 0 1 1 1 1 1 0 0 1 ...  
## $ race : num [1:675] 1 1 2 1 2 2 1 1 1 2 ...  
## $ age : num [1:675] 33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...  
## $ state : num [1:675] 1 1 1 1 1 1 1 1 1 1 ...  
## $ time.served : num [1:675] 5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...  
## $ max.sentence : num [1:675] 18 12 12 18 12 18 18 12 13 12 ...  
## $ multiple.offenses: num [1:675] 0 0 0 0 0 0 0 0 0 0 ...  
## $ crime : num [1:675] 4 3 3 1 1 4 3 1 3 2 ...  
## $ violator : num [1:675] 0 0 0 0 0 0 0 0 0 0 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. male = col\_double(),  
## .. race = col\_double(),  
## .. age = col\_double(),  
## .. state = col\_double(),  
## .. time.served = col\_double(),  
## .. max.sentence = col\_double(),  
## .. multiple.offenses = col\_double(),  
## .. crime = col\_double(),  
## .. violator = col\_double()  
## .. )

summary(parole)

## male race age state   
## Min. :0.0000 Min. :1.000 Min. :18.40 Min. :1.000   
## 1st Qu.:1.0000 1st Qu.:1.000 1st Qu.:25.35 1st Qu.:2.000   
## Median :1.0000 Median :1.000 Median :33.70 Median :3.000   
## Mean :0.8074 Mean :1.424 Mean :34.51 Mean :2.887   
## 3rd Qu.:1.0000 3rd Qu.:2.000 3rd Qu.:42.55 3rd Qu.:4.000   
## Max. :1.0000 Max. :2.000 Max. :67.00 Max. :4.000   
## time.served max.sentence multiple.offenses crime   
## Min. :0.000 Min. : 1.00 Min. :0.0000 Min. :1.000   
## 1st Qu.:3.250 1st Qu.:12.00 1st Qu.:0.0000 1st Qu.:1.000   
## Median :4.400 Median :12.00 Median :1.0000 Median :2.000   
## Mean :4.198 Mean :13.06 Mean :0.5363 Mean :2.059   
## 3rd Qu.:5.200 3rd Qu.:15.00 3rd Qu.:1.0000 3rd Qu.:3.000   
## Max. :6.000 Max. :18.00 Max. :1.0000 Max. :4.000   
## violator   
## Min. :0.0000   
## 1st Qu.:0.0000   
## Median :0.0000   
## Mean :0.1156   
## 3rd Qu.:0.0000   
## Max. :1.0000

Part 2

Factor conversion. Convert and recode.

parole = parole %>% mutate(male = as\_factor(male)) %>% # 1 if the parolee is male, 0 if female  
 mutate(male = fct\_recode(male, "Female" = "0", "Male" = "1"))  
  
parole = parole %>% mutate(race = as\_factor(race)) %>% # 1 if the parolee is white, 2 otherwise  
 mutate(race = fct\_recode(race, "white" = "1", "otherwise" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(state)) %>% # codes for the parolee’s state.  
 mutate(state = fct\_recode(state, "Other State" = "1", "Kentucky" = "2", "Louisiana" = "3", "Virginia" = "4"))  
  
parole = parole %>% mutate(multiple.offenses = as\_factor(multiple.offenses)) %>% # 1 if the parolee was incarcerated for multiple offenses, 0 otherwise  
 mutate(multiple.offenses = fct\_recode(multiple.offenses, "otherwise" = "0", "incarcerated" = "1"))  
  
parole = parole %>% mutate(crime = as\_factor(crime)) %>% # code for the parolee’s main crime leading to incarceration.  
 mutate(crime = fct\_recode(crime, "Other Crime" = "1", "larceny" = "2", "drug-related" = "3", "drivig related" = "4"))  
  
parole = parole %>% mutate(violator = as\_factor(violator)) %>% # 1 if the parolee violated the parole, and 0 if the parolee completed the parole without violation.  
 mutate(violator = fct\_recode(violator, "Completed parole without violation" = "0", "violated parole" = "1"))  
  
str(parole)

## spec\_tbl\_df[,9] [675 x 9] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ male : Factor w/ 2 levels "Female","Male": 2 1 2 2 2 2 2 1 1 2 ...  
## $ race : Factor w/ 2 levels "white","otherwise": 1 1 2 1 2 2 1 1 1 2 ...  
## $ age : num [1:675] 33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...  
## $ state : Factor w/ 4 levels "Other State",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ time.served : num [1:675] 5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...  
## $ max.sentence : num [1:675] 18 12 12 18 12 18 18 12 13 12 ...  
## $ multiple.offenses: Factor w/ 2 levels "otherwise","incarcerated": 1 1 1 1 1 1 1 1 1 1 ...  
## $ crime : Factor w/ 4 levels "Other Crime",..: 4 3 3 1 1 4 3 1 3 2 ...  
## $ violator : Factor w/ 2 levels "Completed parole without violation",..: 1 1 1 1 1 1 1 1 1 1 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. male = col\_double(),  
## .. race = col\_double(),  
## .. age = col\_double(),  
## .. state = col\_double(),  
## .. time.served = col\_double(),  
## .. max.sentence = col\_double(),  
## .. multiple.offenses = col\_double(),  
## .. crime = col\_double(),  
## .. violator = col\_double()  
## .. )

#TASK 1

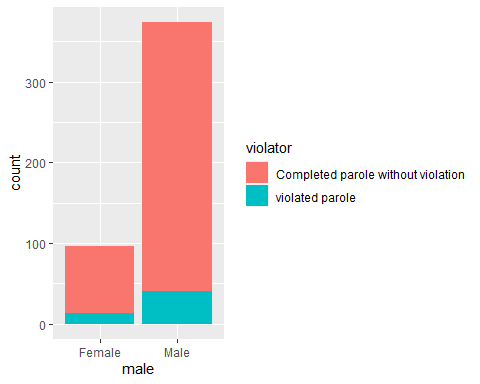
Split the data in ‘training’ and ‘testing’ sets. Training set will have 70% of the data.

set.seed(12345)  
parole\_split = initial\_split(parole, prop = 0.70, strata = violator)##Tidymodels function for splitting  
train = training(parole\_split)  
test = testing(parole\_split)

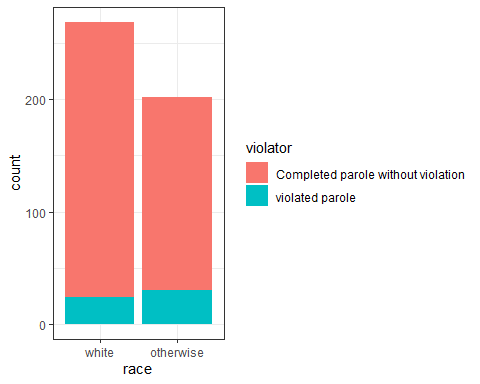
#TASK 2

Visualize using the training set (looking at relationship between ‘violator’ and the other variables)

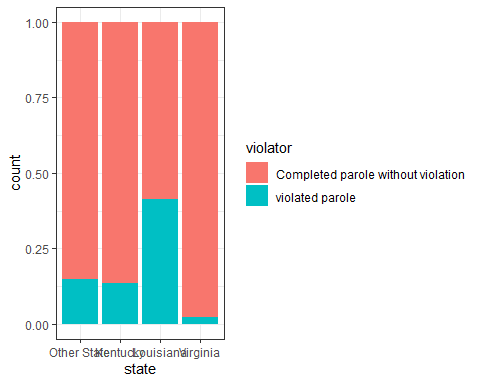
ggplot(train,aes(x=male, fill = violator)) + geom\_bar()



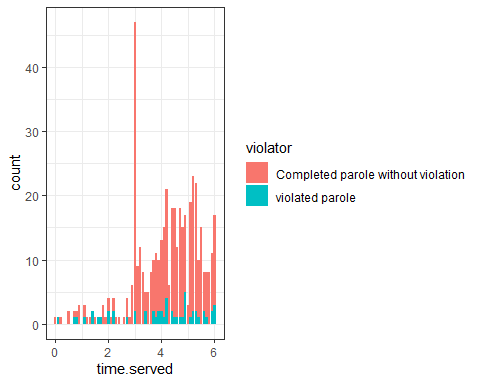
ggplot(train,aes(x=race, fill = violator)) + geom\_bar() + theme\_bw()



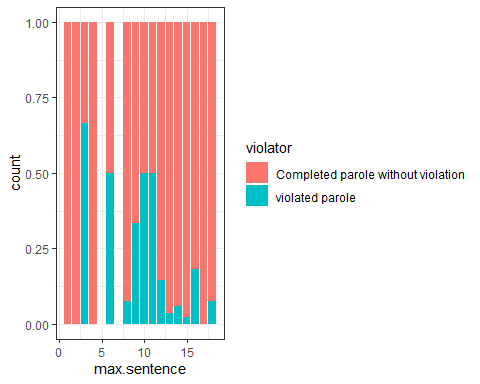
ggplot(train,aes(x=state, fill = violator)) + geom\_bar(position = 'fill') + theme\_bw()



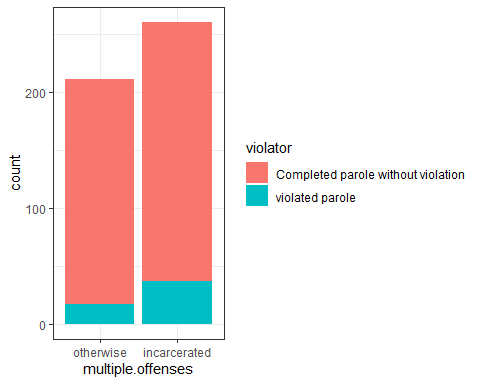
ggplot(train,aes(x=time.served, fill = violator)) + geom\_bar() + theme\_bw()



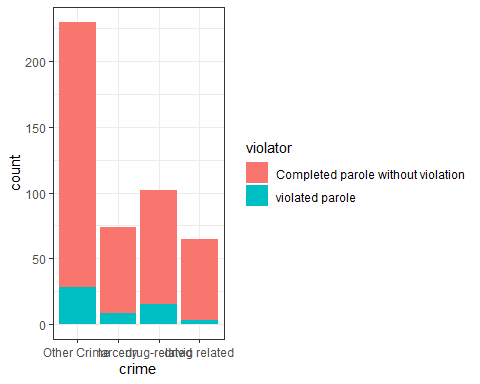
ggplot(train,aes(x=max.sentence, fill = violator)) + geom\_bar(position = 'fill') + theme\_bw()



ggplot(train,aes(x=multiple.offenses, fill = violator)) + geom\_bar() + theme\_bw()



ggplot(train,aes(x=crime, fill = violator)) + geom\_bar() + theme\_bw()

 Task 2 Explanation:

The following variables appear to stand out as most predictive of the response variable ‘violator’:

* male
* state
* max.sentence
* multiple.offenses

This data supports the notion that although there are more male and female parolees, more men than women appear to violate their parole. I also noticed that there is a higher rate of parolees from Louisiana who violate their parole. From what I gather, parolees who have been sentences to 3 months appear to be more prone to violate their parole. Lastly, there is a higher rate of parole violators who are incarcerated that have committed multiple offenses.

#TASK 3

From my observations, the ‘state’ variable appears to be the most predictive of the ‘violator’ variable. Outlined below is a logistic regression model using the ‘state’ variable to predict the ‘violator’ variable.

parole\_model =   
 logistic\_reg() %>% #note the use of logistic\_reg  
 set\_engine("glm") #standard logistic regression engine is glm(logistics regression)  
  
parole\_recipe = recipe(violator ~ state, parole) %>% ##We are predicting parole by the variable 'male'  
 step\_dummy(all\_nominal(), -all\_outcomes())   
  
logreg\_wf = workflow() %>%  
 add\_recipe(parole\_recipe) %>%   
 add\_model(parole\_model)  
  
parole\_fit = fit(logreg\_wf, parole)

summary(parole\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.0955 -0.4981 -0.2071 -0.2071 2.7760   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.8165 0.2411 -7.534 4.92e-14 \*\*\*  
## state\_Kentucky -0.2079 0.3728 -0.558 0.577   
## state\_Louisiana 1.6207 0.3277 4.946 7.58e-07 \*\*\*  
## state\_Virginia -2.0153 0.4517 -4.461 8.15e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 483.27 on 674 degrees of freedom  
## Residual deviance: 382.89 on 671 degrees of freedom  
## AIC: 390.89  
##   
## Number of Fisher Scoring iterations: 6

Comments on the quality of the model for TASK 3:

The ‘state’ variable has 4 levels representing the three names states and a ‘other state’ level. Note the negative coefficients for Kentucky and Virginia. This suggest that the probability that parolees from these two states will violate their parole decreases compared to parolees from Louisiana. The dropoff is more severe (larger coefficient) for the state of Virginia.

The AIC of this model (a measure of model quality) is 390.89. This smaller AIC value indicates that the quality of the model gets better. When looking at the p-values for the states, we can clearly see the states where parolees come from is significant since 3 of the 4 p-values is less than 0.05.

#TASK 4

Manually create the best model I can to make predictions. Is it the lecture where you talk about developing predicted probabilities for each row in a dataset? (edited)

predictions = predict(parole\_fit, train, type="prob") #develop predicted probabilities  
head(predictions)

## # A tibble: 6 x 2  
## `.pred\_Completed parole without violation` `.pred\_violated parole`  
## <dbl> <dbl>  
## 1 0.860 0.140  
## 2 0.860 0.140  
## 3 0.860 0.140  
## 4 0.860 0.140  
## 5 0.860 0.140  
## 6 0.860 0.140

predictions = predict(parole\_fit, train, type="prob")[2]## th [2] will pull just the second column, the YES  
head(predictions)

## # A tibble: 6 x 1  
## `.pred\_violated parole`  
## <dbl>  
## 1 0.140  
## 2 0.140  
## 3 0.140  
## 4 0.140  
## 5 0.140  
## 6 0.140

Comments on the quality of the model for TASK 4:

As you can see, the prediction from my model yields two values. Based on the my final assessment, this translates to a 7% chance for some or a 20% chance for the remaining that any of the parolees will violate there parole.

#TASK 5

Create a logistic regression model using the training set to predict “violator” using the variables state, multiple.offenses, and race.

parole\_model =   
 logistic\_reg() %>% #note the use of logistic\_reg and mode = "classification"  
 set\_engine("glm") #standard logistic regression engine is glm  
  
parole\_recipe = recipe(violator ~ state + multiple.offenses + race, train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(parole\_recipe) %>%   
 add\_model(parole\_model)  
  
parole\_fit = fit(logreg\_wf, train)

summary(parole\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.3572 -0.4013 -0.2705 -0.1557 2.9726   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.47873 0.36085 -6.869 6.46e-12 \*\*\*  
## state\_Kentucky -0.01418 0.46926 -0.030 0.97590   
## state\_Louisiana 0.11876 0.49950 0.238 0.81206   
## state\_Virginia -3.58422 0.63848 -5.614 1.98e-08 \*\*\*  
## multiple.offenses\_incarcerated 1.65689 0.39652 4.179 2.93e-05 \*\*\*  
## race\_otherwise 1.11646 0.39092 2.856 0.00429 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 335.47 on 470 degrees of freedom  
## Residual deviance: 244.52 on 465 degrees of freedom  
## AIC: 256.52  
##   
## Number of Fisher Scoring iterations: 6

Comments on the quality of the model for TASK 5:

The addition of the ‘race’ and ‘multiple.offenses’ variables has contributed to the improvement of this model rather than using ‘state’ by itself as a predictor of ‘violator’. Note the negative coefficients for Virginia, moving from -2.0153 to -3.58422. This model suggest that the probability is lower that parolees from Virginia will violate their parole compared to parolees from Louisiana and Kentucky.

The AIC of this model (a measure of model quality) is lower; going from 390.89 to 256.52. This smaller AIC value indicates that the quality of the model is better than selecting just ‘state’ by itself as a predictor of ‘violator’. Both ‘race’ and ‘multiple.offenses’ is not that significant, yielding positive coefficients. When looking at the p-values for the states, we can clearly see that Virginia and Kentucky are significant states where parolees come from.

#TASK 6

What is the predicted probability of parole violation of the two following parolees?

Parolee 1: Louisiana with multiple offenses and white race

newdata = data.frame(state = "Louisiana", multiple.offenses = "incarcerated", race = "white")  
predict(parole\_fit, newdata, type="prob")

## # A tibble: 1 x 2  
## `.pred\_Completed parole without violation` `.pred\_violated parole`  
## <dbl> <dbl>  
## 1 0.669 0.331

Parolee 2: Kentucky with no multiple offenses and other race

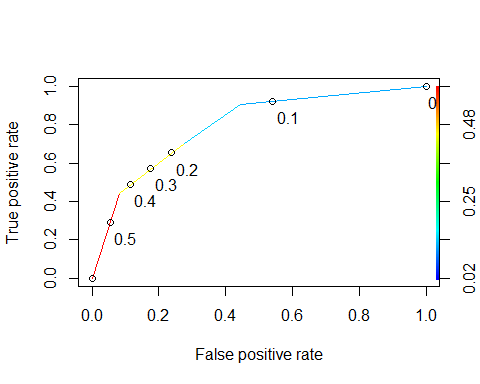
newdata = data.frame(state = "Kentucky", multiple.offenses = "otherwise", race = "otherwise")  
predict(parole\_fit, newdata, type="prob")

## # A tibble: 1 x 2  
## `.pred\_Completed parole without violation` `.pred\_violated parole`  
## <dbl> <dbl>  
## 1 0.798 0.202

#TASK 7

Develop an ROC curve and determine the probability threshold that best balances specificity and sensitivity (on the training set).

ROCRpred = prediction(predictions, train$violator)   
###Do not change the next two lines:  
ROCRperf = performance(ROCRpred, "tpr", "fpr")  
plot(ROCRperf, colorize=TRUE, print.cutoffs.at=seq(0,1,by=0.1), text.adj=c(-0.2,1.7))



#TASK 8

Part 1 What is the accuracy, sensitivity, and specificity of the model on the training set given the cutoff from Task 7?

#Determine threshold to balance sensitivity and specificity  
#DO NOT modify this code  
opt.cut = function(perf, pred){  
 cut.ind = mapply(FUN=function(x, y, p){  
 d = (x - 0)^2 + (y-1)^2  
 ind = which(d == min(d))  
 c(sensitivity = y[[ind]], specificity = 1-x[[ind]],   
 cutoff = p[[ind]])  
 }, perf@x.values, perf@y.values, pred@cutoffs)  
}  
print(opt.cut(ROCRperf, ROCRpred))

## [,1]  
## sensitivity 0.7037037  
## specificity 0.7242206  
## cutoff 0.1398601

Test thresholds to evaluate ‘accuracy’

#confusion matrix  
#The "No" and "Yes" represent the actual values  
#The "FALSE" and "TRUE" represent our predicted values  
t1 = table(train$violator,predictions > 0.1398601)  
t1

##   
## FALSE TRUE  
## Completed parole without violation 302 115  
## violated parole 16 38

Calculate accuracy

(t1[1,1]+t1[2,2])/nrow(train) #We are looking at the training dataframe.

## [1] 0.7218684

We are 72% accurate in putting people in the right group.

Calculate Sensitivity

38/(16+38)

## [1] 0.7037037

Calculate Specificity

302/(302+115)

## [1] 0.7242206

Part 2 What are the implications of incorrectly classifying a parolee?

If this model were used in a real situation, a false calculation would result in a parolee being approved for release, while a true calculation would result in a denial for release.

#TASK 9

Identify a probability threshold (via trial-and-error) that best maximizes accuracy on the training set.

Trying 0.5 as threshold

t1 = table(train$violator,predictions > 0.4)   
t1

##   
## FALSE TRUE  
## Completed parole without violation 383 34  
## violated parole 30 24

(t1[1,1]+t1[2,2])/nrow(train)

## [1] 0.8641189

t1 = table(train$violator,predictions > 0.3) #Trying 0.6 as threshold  
#t1  
(t1[1,1]+t1[2,2])/nrow(train)

## [1] 0.8641189

I will choose a probability threshold of 0.4

#TASK 10

Use your probability threshold from Task 9 to determine accuracy of the model on the testing set.

Part 1

predictionstest = predict(parole\_fit, test, type="prob")[2]## th [2] will pull just the second column, the YES  
head(predictions)

## # A tibble: 6 x 1  
## `.pred\_violated parole`  
## <dbl>  
## 1 0.140  
## 2 0.140  
## 3 0.140  
## 4 0.140  
## 5 0.140  
## 6 0.140

Part 2

t2 = table(test$violator,predictionstest > 0.4) #Trying 0.4 as threshold  
t2

##   
## FALSE TRUE  
## Completed parole without violation 175 5  
## violated parole 16 8

(t2[1,1]+t2[2,2])/nrow(test)

## [1] 0.8970588

Part 3

Calculate accuracy on the Test group

(t2[1,1]+t2[2,2])/nrow(test) #We are looking at the testing dataframe.

## [1] 0.8970588

We are near 90% accurate in putting people in the right group.