Module3Assignment 2

JeKolby Worthy

2/8/2021

# Module 3 Assignment 2

## Classification with Logistic Regression Assignment

### Je’Kolby Worthy

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──

## ✓ ggplot2 3.3.3 ✓ purrr 0.3.4  
## ✓ tibble 3.0.5 ✓ dplyr 1.0.3  
## ✓ tidyr 1.1.2 ✓ stringr 1.4.0  
## ✓ readr 1.4.0 ✓ forcats 0.5.0

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

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 0.1.2 ──

## ✓ broom 0.7.3 ✓ recipes 0.1.15  
## ✓ dials 0.0.9 ✓ rsample 0.0.8   
## ✓ infer 0.5.4 ✓ tune 0.1.2   
## ✓ modeldata 0.1.0 ✓ workflows 0.2.1   
## ✓ parsnip 0.1.5 ✓ yardstick 0.0.7

## ── 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()

library(e1071)

##   
## Attaching package: 'e1071'

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

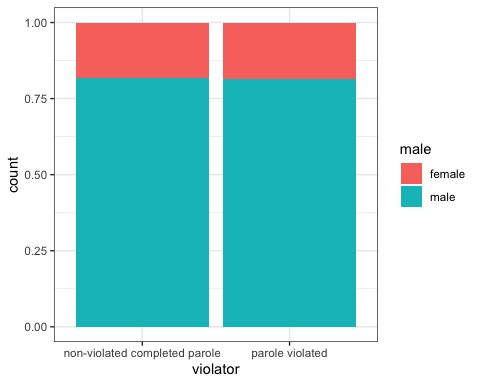
library(ROCR)  
parole <- read\_csv("parole.csv")

##   
## ── 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()  
## )

parole = parole %>% mutate(male = as\_factor(male)) %>%  
mutate(male = fct\_recode(male, "male" = "1", "female" = "0" ))  
  
parole = parole %>% mutate(race = as\_factor(race)) %>%  
mutate(race = fct\_recode(race, "white" = "1", "other" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(state)) %>%  
mutate(state = fct\_recode(state, "Kentucky" = "2", "Louisana" = "3", "Virginia" = "4", "other" = "1"))  
  
parole = parole %>% mutate(crime = as\_factor(crime)) %>%  
mutate(crime = fct\_recode(crime, "larceny" = "2", "drug" = "3", "driving" = "4", "other" = "1"))  
  
parole = parole %>% mutate(multiple.offenses = as\_factor(multiple.offenses)) %>%  
mutate(multiple.offenses = fct\_recode(multiple.offenses, "multiple.offenses" = "1", "other" = "0"))  
  
parole = parole %>% mutate(violator = as\_factor(violator)) %>%  
mutate(violator = fct\_recode(violator, "parole violated" = "1", "non-violated completed parole" = "0"))

set.seed(1235)  
parole\_split = initial\_split(parole, prob = 0.70, strata = violator)  
train = training(parole\_split)  
test = testing(parole\_split)

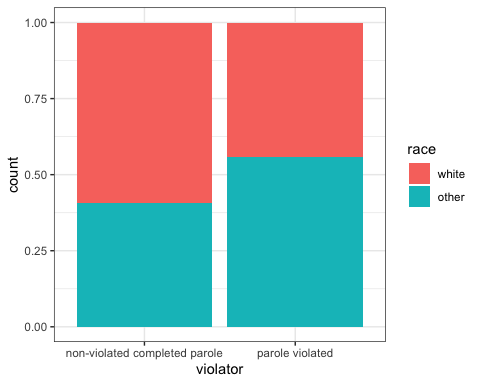
ggplot(train, aes(x=violator, fill=male)) + geom\_bar(position = "fill") + theme\_bw()



t1 = table(train$violator, train$male)  
prop.table(t1, margin = 2)

##   
## female male  
## non-violated completed parole 0.8804348 0.8843373  
## parole violated 0.1195652 0.1156627

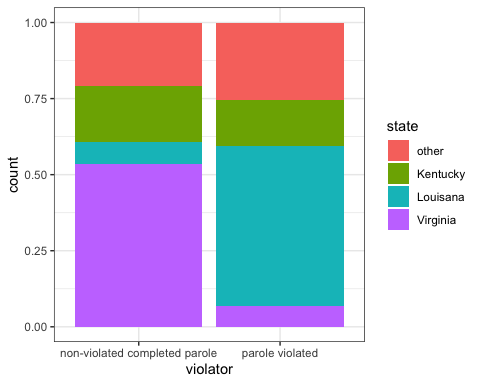
ggplot(train,aes(x=violator, fill=race)) + geom\_bar(position = "fill") + theme\_bw()



t2 = table(train$violator, train$race)  
prop.table(t2, margin = 2)

##   
## white other  
## non-violated completed parole 0.91065292 0.84722222  
## parole violated 0.08934708 0.15277778

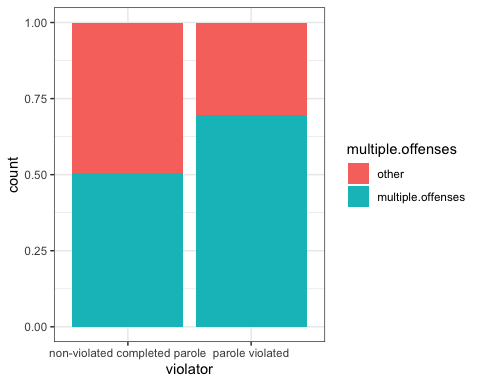
ggplot(train,aes(x=violator, fill=state)) + geom\_bar(position = "fill") + theme\_bw()



t3 = table(train$violator, train$state)  
prop.table(t3, margin = 2)

##   
## other Kentucky Louisana Virginia  
## non-violated completed parole 0.86238532 0.90109890 0.51562500 0.98353909  
## parole violated 0.13761468 0.09890110 0.48437500 0.01646091

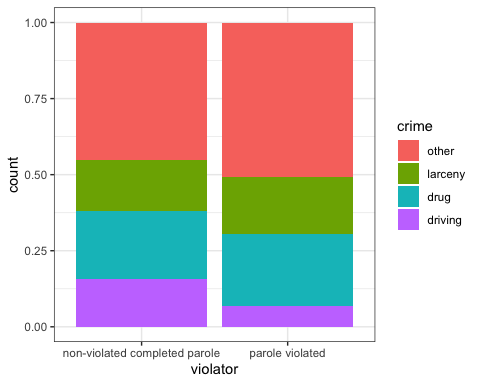
ggplot(train,aes(x=violator, fill=multiple.offenses)) + geom\_bar(position = "fill") + theme\_bw()



t4 = table(train$violator, train$multiple.offenses)  
prop.table(t4, margin = 2)

##   
## other multiple.offenses  
## non-violated completed parole 0.92468619 0.84701493  
## parole violated 0.07531381 0.15298507

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



t5 = table(train$violator, train$crime)  
prop.table(t5, margin = 2)

##   
## other larceny drug driving  
## non-violated completed parole 0.87068966 0.87209302 0.87719298 0.94666667  
## parole violated 0.12931034 0.12790698 0.12280702 0.05333333

**The state, multiple.offenses, and race tend to be the most predictive of the variable “violator”. According to these graphs and tables I have developed, has help me came to my final conclusion, the table was very helpful in comparing the values between each variable.**

parole\_model =  
 logistic\_reg() %>%  
 set\_engine("glm")  
  
parole\_recipe = recipe(violator ~ state, 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   
## -1.1510 -0.5003 -0.1822 -0.1822 2.8659   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.8352 0.2780 -6.601 4.09e-11 \*\*\*  
## state\_Kentucky -0.3743 0.4479 -0.836 0.403   
## state\_Louisana 1.7727 0.3740 4.740 2.14e-06 \*\*\*  
## state\_Virginia -2.2549 0.5757 -3.917 8.96e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 364.67 on 506 degrees of freedom  
## Residual deviance: 275.51 on 503 degrees of freedom  
## AIC: 283.51  
##   
## Number of Fisher Scoring iterations: 6

**According to this model, Louisiana citizens are more likely to break their parole because it has a coefficient of 1.77277**.

parole\_model =   
 logistic\_reg() %>%  
 set\_engine("glm")  
  
parole\_recipe = recipe(violator ~ ., train)%>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(parole\_recipe)%>%  
 add\_model(parole\_model)  
  
parole\_fit2 = fit(logreg\_wf, train)

options(scipen = 999)  
summary(parole\_fit2$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7086 -0.3963 -0.2346 -0.1361 3.0706   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -1.8445546 1.1918347 -1.548 0.121704  
## age 0.0007745 0.0159165 0.049 0.961188  
## time.served -0.1550656 0.1144754 -1.355 0.175553  
## max.sentence 0.0015986 0.0528721 0.030 0.975880  
## male\_male 0.0266361 0.4366176 0.061 0.951355  
## race\_other 0.8134586 0.3794727 2.144 0.032061  
## state\_Kentucky 0.1053799 0.4931568 0.214 0.830793  
## state\_Louisana 0.7875505 0.5064116 1.555 0.119908  
## state\_Virginia -3.4179765 0.6521725 -5.241 0.00000016  
## multiple.offenses\_multiple.offenses 1.4428102 0.3812041 3.785 0.000154  
## crime\_larceny 0.1984878 0.4939011 0.402 0.687774  
## crime\_drug -0.5072724 0.4195012 -1.209 0.226575  
## crime\_driving -0.2718154 0.6255115 -0.435 0.663890  
##   
## (Intercept)   
## age   
## time.served   
## max.sentence   
## male\_male   
## race\_other \*   
## state\_Kentucky   
## state\_Louisana   
## state\_Virginia \*\*\*  
## multiple.offenses\_multiple.offenses \*\*\*  
## crime\_larceny   
## crime\_drug   
## crime\_driving   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 364.67 on 506 degrees of freedom  
## Residual deviance: 250.13 on 494 degrees of freedom  
## AIC: 276.13  
##   
## Number of Fisher Scoring iterations: 7

options(scipen = 0)

**Based on this model, the most relevant variables appear to be the state and multiple offenses. This model’s AIC is 276, which is significantly higher than the previous model’s AIC of 283. The highest present coefficient is the variable for multiple offenses, which makes sense because this was a variable previously found out to be predictive of a violator of parole.As those coefficients are high, the state of Louisiana and race tend to play a key vital role in deciding who is likely to break their parole.**

parole\_model =  
 logistic\_reg() %>%  
 set\_engine("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\_fit3 = fit(logreg\_wf, train)

options(scipen = 999)  
summary(parole\_fit3$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.4443 -0.3827 -0.2325 -0.1605 2.9523   
##   
## Coefficients:  
## Estimate Std. Error z value  
## (Intercept) -2.57716 0.35685 -7.222  
## state\_Kentucky -0.03427 0.47224 -0.073  
## state\_Louisana 0.93125 0.44651 2.086  
## state\_Virginia -3.27453 0.63250 -5.177  
## multiple.offenses\_multiple.offenses 1.50655 0.37380 4.030  
## race\_other 0.74794 0.37312 2.005  
## Pr(>|z|)   
## (Intercept) 0.000000000000512 \*\*\*  
## state\_Kentucky 0.942   
## state\_Louisana 0.037 \*   
## state\_Virginia 0.000000225350216 \*\*\*  
## multiple.offenses\_multiple.offenses 0.000055690668583 \*\*\*  
## race\_other 0.045 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 364.67 on 506 degrees of freedom  
## Residual deviance: 254.42 on 501 degrees of freedom  
## AIC: 266.42  
##   
## Number of Fisher Scoring iterations: 6

options(scipen = 0)

**This model, with an AIC score of 266.42 and 3 significant variables, is an improvement from the previous model. Yet again, multiple offenses are considered significant.**

What is the predicted probability of parole violation of the two following parolees? **The Parolee1 probability of parole violation is 47%**

newdatal = data.frame(state= "Louisana", multiple.offenses= "multiple offenses", race="white")  
predict(parole\_fit3, newdatal, type= "prob")

## Warning: Novel levels found in column 'multiple.offenses': 'multiple offenses'.  
## The levels have been removed, and values have been coerced to 'NA'.

## Warning: There are new levels in a factor: NA

## # A tibble: 1 x 2  
## `.pred\_non-violated completed parole` `.pred\_parole violated`  
## <dbl> <dbl>  
## 1 NA NA

**The Parolee2 probability of parole violation is 13%**

newdata2 = data.frame(state= "Kentucky", multiple.offenses= "other", race= "other")  
predict(parole\_fit3, newdata2, type= "prob")

## # A tibble: 1 x 2  
## `.pred\_non-violated completed parole` `.pred\_parole violated`  
## <dbl> <dbl>  
## 1 0.866 0.134

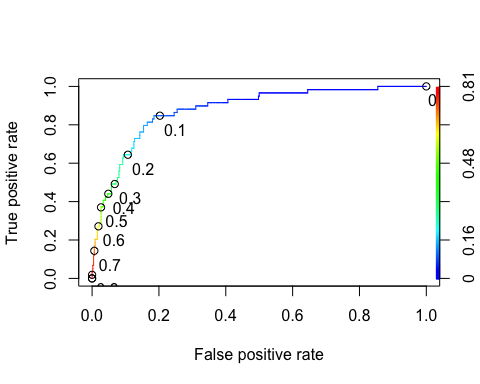
predictions = predict(parole\_fit2, train, type= "prob")  
head(predictions)

## # A tibble: 6 x 2  
## `.pred\_non-violated completed parole` `.pred\_parole violated`  
## <dbl> <dbl>  
## 1 0.947 0.0527  
## 2 0.958 0.0415  
## 3 0.912 0.0880  
## 4 0.859 0.141   
## 5 0.895 0.105   
## 6 0.961 0.0391

predictions = predict(parole\_fit2, train, type = "prob")[2]  
head(predictions)

## # A tibble: 6 x 1  
## `.pred\_parole violated`  
## <dbl>  
## 1 0.0527  
## 2 0.0415  
## 3 0.0880  
## 4 0.141   
## 5 0.105   
## 6 0.0391

ROCRpred = prediction(predictions, train$violator)  
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))



as.numeric(performance(ROCRpred,"auc")@y.values)

## [1] 0.8787833

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 0.8474576  
## specificity 0.8147321  
## cutoff 0.1116835

accur = table(train$violator, predictions > 0.1116835)  
accur

##   
## FALSE TRUE  
## non-violated completed parole 365 83  
## parole violated 10 49

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

## [1] 0.816568

**The accuracy is 82%, the sensitivity is 77% and the specificity is 82% . In the comparable sense of health care and insurance, encouraging potential offenders to reenter society and trusting that they will not commit another crime makes this similar in certain ways. Classifying anyone wrongly will result in a longer parole for them**.

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

##   
## FALSE TRUE  
## non-violated completed parole 427 21  
## parole violated 33 26

accur = table(train$violator,predictions > 0.6)  
accur

##   
## FALSE TRUE  
## non-violated completed parole 440 8  
## parole violated 43 16

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

## [1] 0.8994083

accur = table(train$violator, predictions > 0.5)  
accur

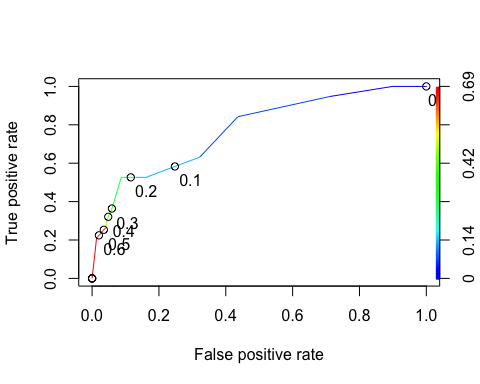
##   
## FALSE TRUE  
## non-violated completed parole 436 12  
## parole violated 38 21

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

## [1] 0.9013807

**The 0.5 probability threshold best maximizes accuracy after trial and error.**

predictions = predict(parole\_fit3, test, type="prob")[2]  
ROCRpred = prediction(predictions, test$violator)  
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))



as.numeric(performance(ROCRpred,"auc")@y.values)

## [1] 0.7723419

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 0.84210526  
## specificity 0.56375839  
## cutoff 0.06840645

t6 = table(test$violator, predictions > 0.5)  
(t6[1,1]+t6[2,2])/nrow(test)

## [1] 0.8988095