Module3\_Assignment2

parole <- read\_csv("C:/Users/cltma/OneDrive/Documents/BAN502/Module3/Module 3 Assignment 2/Module3\_Assignment2/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(race = as\_factor(race)) %>%  
 mutate(state = as\_factor(state)) %>%  
 mutate(crime = as\_factor(crime)) %>%  
 mutate(multiple.offenses =as\_factor(multiple.offenses)) %>%  
 mutate(violator = as\_factor(violator)) %>%  
 mutate(male = fct\_recode(male, "Female" = "0", "Male" = "1" )) %>%  
 mutate(race = fct\_recode(race, "Non-White" = "2", "White" = "1" )) %>%  
 mutate(state = fct\_recode(state, "Kentucky" = "2", "Louisiana" = "3", "Virginia" = "4", "Other" = "1")) %>%  
 mutate(crime = fct\_recode(crime, "Larceny" = "2", "Drug-Related Crime" = "3", "Driving-Related Crime" = "4", "All Other" = "1")) %>%  
 mutate(multiple.offenses = fct\_recode(multiple.offenses, "No" = "0", "Yes" = "1" )) %>%  
 mutate(violator = fct\_recode(violator, "No" = "0", "Yes" = "1" ))  
  
summary(parole)

## male race age state time.served   
## Female:130 White :389 Min. :18.40 Other :143 Min. :0.000   
## Male :545 Non-White:286 1st Qu.:25.35 Kentucky :120 1st Qu.:3.250   
## Median :33.70 Louisiana: 82 Median :4.400   
## Mean :34.51 Virginia :330 Mean :4.198   
## 3rd Qu.:42.55 3rd Qu.:5.200   
## Max. :67.00 Max. :6.000   
## max.sentence multiple.offenses crime violator   
## Min. : 1.00 No :313 All Other :315 No :597   
## 1st Qu.:12.00 Yes:362 Larceny :106 Yes: 78   
## Median :12.00 Drug-Related Crime :153   
## Mean :13.06 Driving-Related Crime:101   
## 3rd Qu.:15.00   
## Max. :18.00

glimpse(parole)

## Rows: 675  
## Columns: 9  
## $ male <fct> Male, Female, Male, Male, Male, Male, Male, Femal...  
## $ race <fct> White, White, Non-White, White, Non-White, Non-Wh...  
## $ age <dbl> 33.2, 39.7, 29.5, 22.4, 21.6, 46.7, 31.0, 24.6, 3...  
## $ state <fct> Other, Other, Other, Other, Other, Other, Other, ...  
## $ time.served <dbl> 5.5, 5.4, 5.6, 5.7, 5.4, 6.0, 6.0, 4.8, 4.5, 4.7,...  
## $ max.sentence <dbl> 18, 12, 12, 18, 12, 18, 18, 12, 13, 12, 12, 12, 1...  
## $ multiple.offenses <fct> No, No, No, No, No, No, No, No, No, No, Yes, No, ...  
## $ crime <fct> Driving-Related Crime, Drug-Related Crime, Drug-R...  
## $ violator <fct> No, No, No, No, No, No, No, No, No, No, No, No, N...

### Task 1

set.seed(12345)  
parole\_split = initial\_split(parole, prob = 0.70, strata = violator)  
train = training(parole\_split)  
test = testing(parole\_split)

### Task 2

#### Male

tMale = table(parole$violator, parole$male)  
prop.table(tMale, margin = 2)

##   
## Female Male  
## No 0.8923077 0.8825688  
## Yes 0.1076923 0.1174312

view(tMale)

Very little difference in the proportion of violators based upon gender.

#### Race

tRace = table(parole$violator, parole$race)  
prop.table(tRace, margin = 2)

##   
## White Non-White  
## No 0.90488432 0.85664336  
## Yes 0.09511568 0.14335664

view(tRace)

Non-White seems to make up a slightly smaller proportion of parole violators.

#### State

tState = table(parole$violator, parole$state)  
prop.table(tState, margin = 2)

##   
## Other Kentucky Louisiana Virginia  
## No 0.86013986 0.88333333 0.54878049 0.97878788  
## Yes 0.13986014 0.11666667 0.45121951 0.02121212

glimpse(tState)

## 'table' int [1:2, 1:4] 123 20 106 14 45 37 323 7  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:2] "No" "Yes"  
## ..$ : chr [1:4] "Other" "Kentucky" "Louisiana" "Virginia"

Louisiana seems to have a much higher proportion of parole violators compared to other states. However, the data set for Louisiana seems small compared to other states, so the proportion may be skewed with outliers.

#### Crime

tCrime = table(parole$violator, parole$crime)  
prop.table(tCrime, margin = 2)

##   
## All Other Larceny Drug-Related Crime Driving-Related Crime  
## No 0.87619048 0.87735849 0.87581699 0.93069307  
## Yes 0.12380952 0.12264151 0.12418301 0.06930693

view(tCrime)

The proportions broken down by type of crime committed seems to be be consistent across the various types of crime except for Driving-Related Crimes which are somewhat lower.

#### Multiple.Offenses

tMultiple.Offenses = table(parole$violator, parole$multiple.offenses)  
prop.table(tMultiple.Offenses, margin = 2)

##   
## No Yes  
## No 0.9201278 0.8535912  
## Yes 0.0798722 0.1464088

view(tMultiple.Offenses)

Multiple Offenses seems to cause the proportion to almost double.

Based on an analysis of proportions Multiple.Offenses, race and state seems to be good predictors. The proportions when moving from False Positive to True Positive is significant in each.

### Task 3

parole\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
parole\_recipe = recipe(violator ~ state, parole) %>%  
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

### Task 4

train\_male\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_male\_recipe = recipe(violator ~ male, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_male\_recipe) %>%   
 add\_model(train\_male\_model)  
  
train\_male\_fit = fit(logreg\_wf, train)

summary(train\_male\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.4981 -0.4981 -0.4981 -0.4942 2.0801   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.04122 0.33613 -6.073 1.26e-09 \*\*\*  
## male\_Male 0.01684 0.36890 0.046 0.964   
## ---  
## 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: 364.66 on 505 degrees of freedom  
## AIC: 368.66  
##   
## Number of Fisher Scoring iterations: 4

train\_race\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_race\_recipe = recipe(violator ~ race, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_race\_recipe) %>%   
 add\_model(train\_race\_model)  
  
train\_race\_fit = fit(logreg\_wf, train)

summary(train\_race\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.5304 -0.5304 -0.4708 -0.4708 2.1236   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.1440 0.1930 -11.111 <2e-16 \*\*\*  
## race\_Non.White 0.2538 0.2774 0.915 0.36   
## ---  
## 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: 363.83 on 505 degrees of freedom  
## AIC: 367.83  
##   
## Number of Fisher Scoring iterations: 4

train\_state\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_state\_recipe = recipe(violator ~ state, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_state\_recipe) %>%   
 add\_model(train\_state\_model)  
  
train\_state\_fit = fit(logreg\_wf, train)

summary(train\_state\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.0008 -0.5405 -0.2204 -0.2204 2.7312   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.84958 0.28751 -6.433 1.25e-10 \*\*\*  
## state\_Kentucky 0.09704 0.41584 0.233 0.815481   
## state\_Louisiana 1.41880 0.38226 3.712 0.000206 \*\*\*  
## state\_Virginia -1.85583 0.50341 -3.686 0.000227 \*\*\*  
## ---  
## 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: 300.70 on 503 degrees of freedom  
## AIC: 308.7  
##   
## Number of Fisher Scoring iterations: 6

train\_crime\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_crime\_recipe = recipe(violator ~ crime, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_crime\_recipe) %>%   
 add\_model(train\_crime\_model)  
  
train\_crime\_fit = fit(logreg\_wf, train)

summary(train\_crime\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.5216 -0.5216 -0.5200 -0.3616 2.3495   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.926009 0.198766 -9.690 <2e-16 \*\*\*  
## crime\_Larceny -0.006829 0.378913 -0.018 0.986   
## crime\_Drug.Related.Crime -0.030054 0.347904 -0.086 0.931   
## crime\_Driving.Related.Crime -0.768618 0.503010 -1.528 0.127   
## ---  
## 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: 361.73 on 503 degrees of freedom  
## AIC: 369.73  
##   
## Number of Fisher Scoring iterations: 5

train\_multiple.offenses\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_multiple.offenses\_recipe = recipe(violator ~ multiple.offenses, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_multiple.offenses\_recipe) %>%   
 add\_model(train\_multiple.offenses\_model)  
  
train\_multiple.offenses\_fit = fit(logreg\_wf, train)

summary(train\_multiple.offenses\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.5663 -0.5663 -0.4088 -0.4088 2.2466   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.4401 0.2392 -10.201 <2e-16 \*\*\*  
## multiple.offenses\_Yes 0.6909 0.2942 2.348 0.0189 \*   
## ---  
## 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: 358.85 on 505 degrees of freedom  
## AIC: 362.85  
##   
## Number of Fisher Scoring iterations: 5

State has the lowest AIC, so would be the preferable variable to use in the model if only using one variable.

### Task 5

train\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
train\_recipe = recipe(violator ~ state + race + multiple.offenses, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(train\_recipe) %>%   
 add\_model(train\_model)  
  
train\_fit = fit(logreg\_wf, train)

summary(train\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.2598 -0.4718 -0.2675 -0.2173 2.7414   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.5431 0.3579 -7.106 1.20e-12 \*\*\*  
## state\_Kentucky 0.4036 0.4470 0.903 0.367   
## state\_Louisiana 0.7135 0.4481 1.592 0.111   
## state\_Virginia -2.7907 0.5570 -5.010 5.43e-07 \*\*\*  
## race\_Non.White 0.4215 0.3527 1.195 0.232   
## multiple.offenses\_Yes 1.5998 0.3684 4.342 1.41e-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: 277.99 on 501 degrees of freedom  
## AIC: 289.99  
##   
## Number of Fisher Scoring iterations: 6

The variable with the most significant p-values are multiple offender = yes. The AIC score is lower including the multiple variables above rather than selecting just an individual variable.

### Task 6

newdata = data.frame(race = "White", state = "Louisiana", multiple.offenses = "Yes")  
predict(train\_fit, newdata, type="prob")

## # A tibble: 1 x 2  
## .pred\_No .pred\_Yes  
## <dbl> <dbl>  
## 1 0.557 0.443

newdata = data.frame(race = "Non-White", state = "Kentucky", multiple.offenses = "No")  
predict(train\_fit, newdata, type="prob")

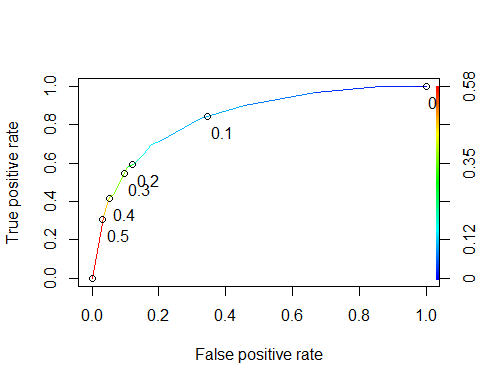
## # A tibble: 1 x 2  
## .pred\_No .pred\_Yes  
## <dbl> <dbl>  
## 1 0.848 0.152

### Task 7

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

## # A tibble: 6 x 1  
## .pred\_Yes  
## <dbl>  
## 1 0.0729  
## 2 0.0729  
## 3 0.0729  
## 4 0.107   
## 5 0.107   
## 6 0.0729

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] 0.834916

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 0.7118644  
## specificity 0.7968750  
## cutoff 0.1070172

### Task 8

t1 = table(train$violator,predictions > 0.1070172)  
t1

##   
## FALSE TRUE  
## No 368 80  
## Yes 18 41

#### Accuracy

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

## [1] 0.8067061

#### Sensitivity

41/(41+368)

## [1] 0.1002445

#### Specificity

80/(80+18)

## [1] 0.8163265

Incorrectly classifying a parolee could mean that the model could be inaccurate and skewed when making decisions on ways to prevent parole violations.

### Task 9

t1 = table(train$violator,predictions > 0.5)  
t1

##   
## FALSE TRUE  
## No 434 14  
## Yes 42 17

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

## [1] 0.8895464

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

##   
## FALSE TRUE  
## No 426 22  
## Yes 35 24

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

## [1] 0.887574

t1 = table(train$violator,predictions > 0.3)  
t1

##   
## FALSE TRUE  
## No 418 30  
## Yes 33 26

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

## [1] 0.8757396

t1 = table(train$violator,predictions > .54)  
t1

##   
## FALSE TRUE  
## No 434 14  
## Yes 42 17

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

## [1] 0.8895464

test\_model =   
 logistic\_reg(mode = "classification") %>%   
 set\_engine("glm")  
  
test\_recipe = recipe(violator ~ state + race + multiple.offenses, test) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(test\_recipe) %>%   
 add\_model(test\_model)  
  
test\_fit = fit(logreg\_wf, test)

summary(test\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.05006 -0.34730 -0.18450 -0.07472 2.85723   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.7780 0.6922 -4.013 5.99e-05 \*\*\*  
## state\_Kentucky -1.2869 1.1457 -1.123 0.261322   
## state\_Louisiana 0.7853 0.8831 0.889 0.373907   
## state\_Virginia -4.9570 1.4212 -3.488 0.000487 \*\*\*  
## race\_Non.White 2.1086 0.8736 2.414 0.015798 \*   
## multiple.offenses\_Yes 1.8551 0.8156 2.274 0.022941 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 118.59 on 167 degrees of freedom  
## Residual deviance: 64.53 on 162 degrees of freedom  
## AIC: 76.53  
##   
## Number of Fisher Scoring iterations: 7

predictions2 = predict(test\_fit, test, type="prob")[2]  
head(predictions2)

## # A tibble: 6 x 1  
## .pred\_Yes  
## <dbl>  
## 1 0.339   
## 2 0.0585  
## 3 0.339   
## 4 0.0585  
## 5 0.0585  
## 6 0.0585

t2 = table(test$violator,predictions2 > .54)  
t2

##   
## FALSE TRUE  
## No 148 1  
## Yes 12 7

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

## [1] 0.922619

The accuracy of the test data using the probability threshold of .54 is slightly higher at .922 versus accuracy of the training set at .8895.