#install.packages("e1071")  
#install.packages("ROCR")

library(tidyverse)

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

## ✓ ggplot2 3.3.2 ✓ purrr 0.3.4  
## ✓ tibble 3.0.4 ✓ dplyr 1.0.2  
## ✓ 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 0.7.2 ✓ 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)

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

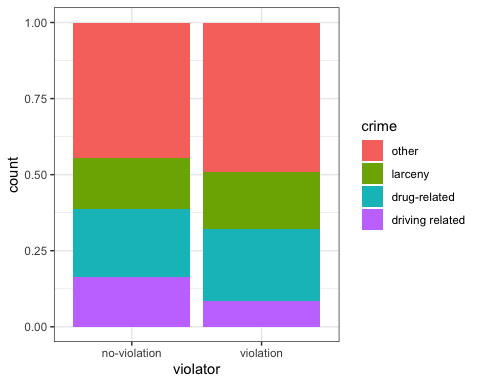
##   
## ── Column specification ────────────────────────────────────────────────────────  
## cols(  
## male = col\_double(),  
## race = col\_double(),  
## age = col\_double(),  
## state = col\_double(),  
## time.served = col\_double(),  
## max.sentence = col\_double(),  
## multiple.offenses = 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", "Louisiana" = "3", "Virginia" = "4", "other" = "1"))  
parole = parole %>% mutate(crime = as\_factor(crime)) %>%  
 mutate(crime = fct\_recode(crime, "larceny" = "2", "drug-related" = "3", "driving related" = "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, "violation" = "1", "no-violation" = "0"))

set.seed(12345)  
parole\_split = initial\_split(parole, prob = .70, strata = violator)  
train = training(parole\_split)  
test = testing(parole\_split)

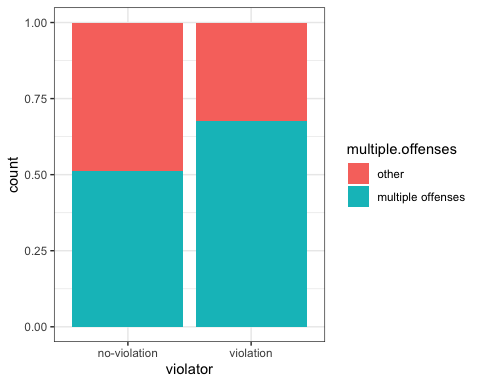
Violator and Crime

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



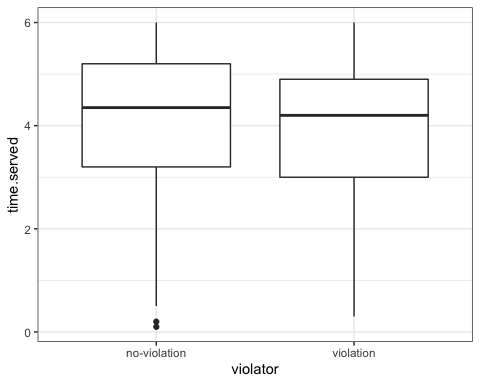
Violator and Multiple Offenses

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



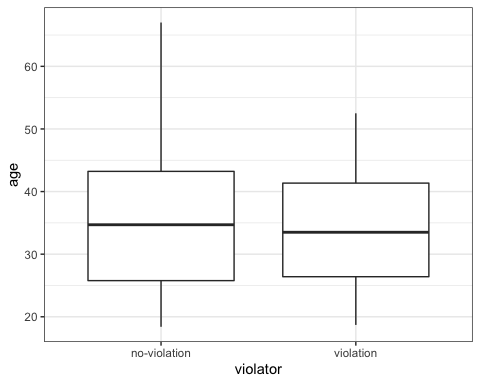
Violator and Time Served

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



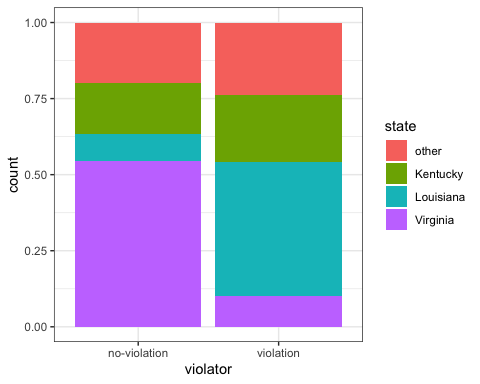
Violator and Age

ggplot(train, aes(x = violator, y = age)) + geom\_boxplot() + theme\_bw()

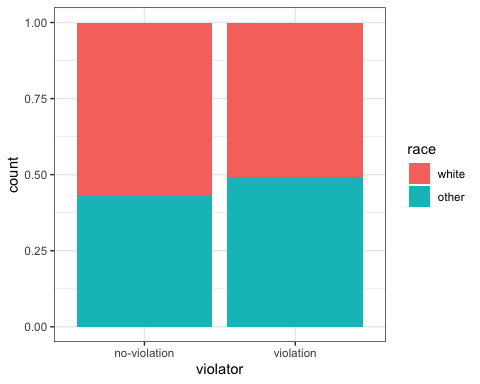


Violator and State

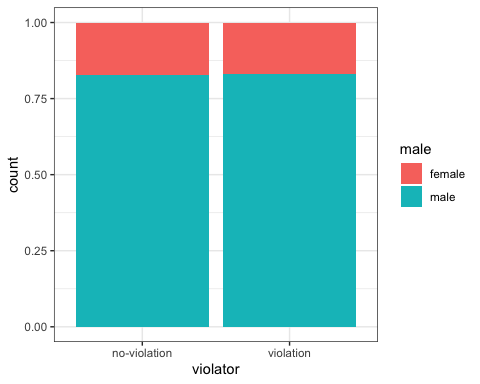
ggplot(train, aes(x = violator, fill = state)) + geom\_bar(position = "fill") + theme\_bw()

 Violator and Race

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

 Violator and Gender

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



In my opinion, the variable that is the most predictive of the response variable “violator” is whether or not the individual was charged with multiple offenses. All of the other graphs seem to be pretty similar, for example with the male or female and the race graphs, there do not seem to be much differences on whether or not the individual violated their parole. In the age and time served box plots, there are not huge differences in the median ages and timed served to show these would be the best predictors. The multiple offenses graph seemed to be the most predictive based on its differences.

parole\_model =  
 logistic\_reg() %>%  
 set\_engine("glm")  
  
parole\_recipe = recipe(violator ~ multiple.offenses, 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   
## -0.5627 -0.5627 -0.4080 -0.4080 2.2483   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.4441 0.2085 -11.722 < 2e-16 \*\*\*  
## multiple.offenses\_multiple.offenses 0.6810 0.2561 2.659 0.00783 \*\*   
## ---  
## 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: 475.81 on 673 degrees of freedom  
## AIC: 479.81  
##   
## Number of Fisher Scoring iterations: 5

The p value for multiple offenses is less than .05, which means this variable is significant to the prediction of whether or not the individual will violate parole. The AIC of this model is 479.81. We can compare this number to another model to see if the AIC goes down because the smaller the AIC, the better.

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

summary(violator\_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\_multiple.offenses 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

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

summary(violator\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.6402 -0.5889 -0.4501 -0.4095 2.5433   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.42156 0.29708 -8.151 3.61e-16 \*\*\*  
## multiple.offenses\_multiple.offenses 0.75748 0.30579 2.477 0.0132 \*   
## crime\_larceny -0.03805 0.38144 -0.100 0.9205   
## crime\_drug.related 0.18309 0.36086 0.507 0.6119   
## crime\_driving.related -0.77238 0.50502 -1.529 0.1262   
## ---  
## 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: 355.23 on 502 degrees of freedom  
## AIC: 365.23  
##   
## Number of Fisher Scoring iterations: 5

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

summary(violator\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.8566 -0.5580 -0.4403 -0.3728 2.4686   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.61025 0.53840 -2.991 0.00278 \*\*  
## time.served -0.19069 0.10834 -1.760 0.07840 .   
## multiple.offenses\_multiple.offenses 0.67528 0.30924 2.184 0.02899 \*   
## crime\_larceny -0.02551 0.38265 -0.067 0.94685   
## crime\_drug.related 0.21662 0.36284 0.597 0.55050   
## crime\_driving.related -0.77795 0.50538 -1.539 0.12372   
## ---  
## 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: 352.22 on 501 degrees of freedom  
## AIC: 364.22  
##   
## Number of Fisher Scoring iterations: 5

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

summary(violator\_fit$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.8564 -0.5583 -0.4404 -0.3730 2.4682   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.6058748 0.6953041 -2.310 0.0209 \*  
## time.served -0.1905584 0.1091063 -1.747 0.0807 .  
## age -0.0001408 0.0141539 -0.010 0.9921   
## multiple.offenses\_multiple.offenses 0.6750621 0.3100326 2.177 0.0295 \*  
## crime\_larceny -0.0253961 0.3828145 -0.066 0.9471   
## crime\_drug.related 0.2167464 0.3630607 0.597 0.5505   
## crime\_driving.related -0.7774399 0.5080065 -1.530 0.1259   
## ---  
## 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: 352.22 on 500 degrees of freedom  
## AIC: 366.22  
##   
## Number of Fisher Scoring iterations: 5

As I added more variables, the AIC was higher than analyzing just multiple offenses on the prediction of parole violation. Based on the model, age and crime are not significant since their p values were a lot higher than .05. Time served is also not significant since its p value is larger than .05.

violator2\_model =  
 logistic\_reg() %>%  
 set\_engine("glm")  
  
violator2\_recipe = recipe(violator ~ multiple.offenses + state + race, train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
logreg\_wf = workflow() %>%  
 add\_recipe(violator2\_recipe) %>%  
 add\_model(violator2\_model)  
  
violator2\_fit = fit(logreg\_wf, train)

summary(violator2\_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 \*\*\*  
## multiple.offenses\_multiple.offenses 1.5998 0.3684 4.342 1.41e-05 \*\*\*  
## 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\_other 0.4215 0.3527 1.195 0.232   
## ---  
## 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

Multiple offenses and the state of Virginia of significant variables to predict violation. Race and the states of Kentucky and Louisiana all have p values that are greater than .05 which makes them not significant. The AIC of this model is 289.99, which is better than previous AIC’s.

newdata = data.frame(state = "Louisiana", multiple.offenses = "multiple offenses", race = "white")  
predict(violator2\_fit, newdata, type = "prob")

## # A tibble: 1 x 2  
## `.pred\_no-violation` .pred\_violation  
## <dbl> <dbl>  
## 1 0.557 0.443

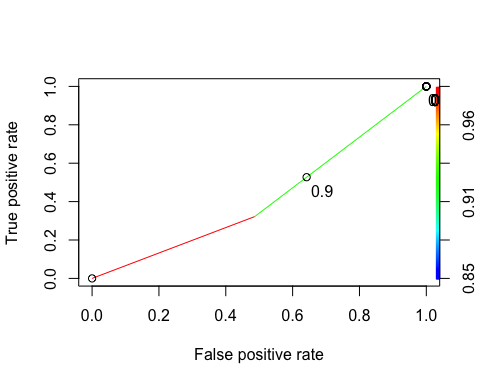
newdata = data.frame(state = "Kentucky", multiple.offenses = "other", race = "other")  
predict(violator2\_fit, newdata, type = "prob")

## # A tibble: 1 x 2  
## `.pred\_no-violation` .pred\_violation  
## <dbl> <dbl>  
## 1 0.848 0.152

predictions = predict(parole\_fit, train, type = "prob")[1]  
head(predictions)

## # A tibble: 6 x 1  
## `.pred\_no-violation`  
## <dbl>  
## 1 0.920  
## 2 0.920  
## 3 0.920  
## 4 0.920  
## 5 0.920  
## 6 0.920

ROCRpred = prediction(predictions, train$violator)  
ROCRperf = performance(ROCRpred, "tpr", "fpr")  
plot(ROCRperf, colorize = TRUE, print.cutoffs.at = seq(0,1,by=.1), text.adj=c(-0.2,1.7))



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 0.3220339  
## specificity 0.5133929  
## cutoff 0.9201278

t1 = table(train$violator, predictions > 0.9201278)  
t1

##   
## FALSE  
## no-violation 448  
## violation 59

Accuracy

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

Sensitivity

Specificity

Trail and Error

#t1 = table(train$violator,predictions > .5)  
#t1  
#(t1[1,1]+t1[2,2])/nrow(train)

#t1 = table(train$violator,predictions > .6)  
#t1  
#(t1[1,1]+t1[2,2])/nrow(train)

#t1 = table(test$violator,predictions > .5)  
#t1  
#(t1[1,1]+t1[2,2])/nrow(test)