Module3\_Assignment2

MTracka

5/31/2021

Load libraries

library(tidyverse)

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

## v ggplot2 3.3.3 v purrr 0.3.4  
## v tibble 3.1.2 v dplyr 1.0.6  
## 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)  
library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

Read-in data into “parole” dataset.

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()  
## )

Convert given numerical variables to factors.

parole = parole %>% mutate(male = as\_factor(male))  
parole = parole %>% mutate(race = as\_factor(race))  
parole = parole %>% mutate(state = as\_factor(state))  
parole = parole %>% mutate(crime = as\_factor(crime))  
parole = parole %>% mutate(multiple.offenses = as\_factor(multiple.offenses))  
parole = parole %>% mutate(violator = as\_factor(violator))

Recode factor levels

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

###### Task 1.

Split the data into training and testing sets.

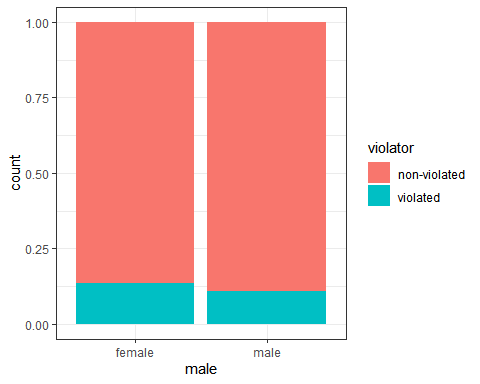
set.seed(12345)  
 parole\_split = initial\_split(parole, prop = 0.70, strata = violator)  
 train = training(parole\_split)  
 test = testing(parole\_split)

###### Task2.

Use visualization/tables to identify which variables are most predictive of response “violator”

Variable “male”

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

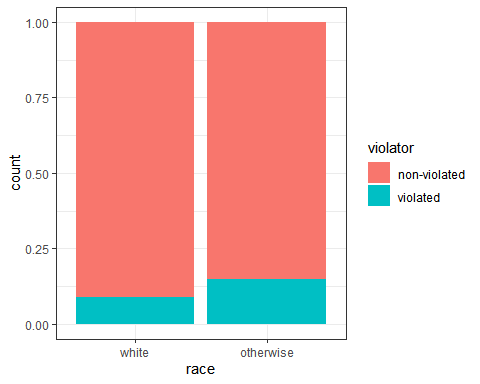


t1 = table(train$violator, train$male)  
prop.table(t1, margin = 2)

##   
## female male  
## non-violated 0.8659794 0.8903743  
## violated 0.1340206 0.1096257

Variable “race”

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

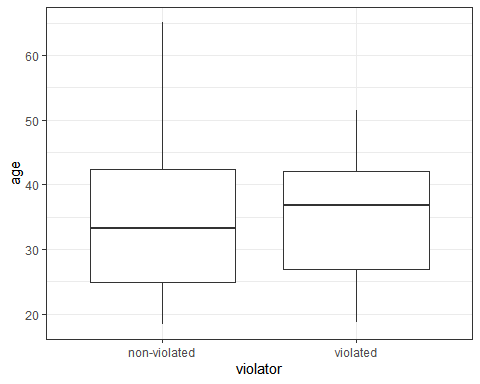


t2 = table(train$violator, train$race)  
prop.table(t2, margin = 2)

##   
## white otherwise  
## non-violated 0.91078067 0.85148515  
## violated 0.08921933 0.14851485

Variable “age”

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

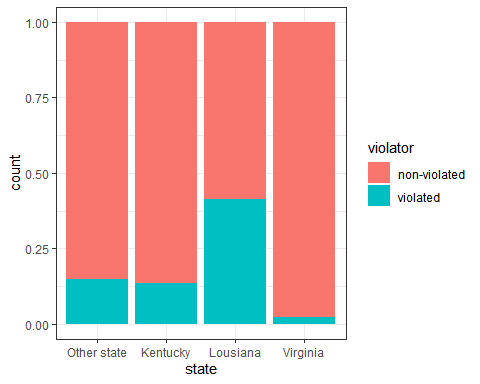


t3 = table(train$violator, train$age)  
prop.table(t3, margin = 2)

##   
## 18.4 18.5 18.7 18.8 19.1 19.2  
## non-violated 1.0000000 1.0000000 0.5000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000 0.0000000  
##   
## 19.3 19.4 19.5 19.6 19.7 19.9  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.6666667  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333  
##   
## 20 20.2 20.3 20.4 20.5 20.6  
## non-violated 1.0000000 0.7500000 0.5000000 1.0000000 1.0000000 0.6666667  
## violated 0.0000000 0.2500000 0.5000000 0.0000000 0.0000000 0.3333333  
##   
## 20.7 20.8 20.9 21.1 21.2 21.3  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 21.4 21.5 21.6 21.7 21.8 21.9  
## non-violated 1.0000000 1.0000000 0.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000  
##   
## 22 22.1 22.2 22.3 22.4 22.5  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 0.5000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.5000000 0.0000000  
##   
## 22.6 22.8 23 23.1 23.2 23.3  
## non-violated 1.0000000 0.6666667 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 23.4 23.6 23.7 23.8 24 24.2  
## non-violated 1.0000000 0.7500000 1.0000000 1.0000000 1.0000000 0.8333333  
## violated 0.0000000 0.2500000 0.0000000 0.0000000 0.0000000 0.1666667  
##   
## 24.4 24.5 24.6 24.7 24.8 24.9  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 25 25.1 25.3 25.4 25.5 25.6  
## non-violated 1.0000000 1.0000000 0.7500000 1.0000000 1.0000000 0.8333333  
## violated 0.0000000 0.0000000 0.2500000 0.0000000 0.0000000 0.1666667  
##   
## 25.7 25.8 25.9 26.4 26.5 26.6  
## non-violated 1.0000000 0.5000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 26.8 26.9 27 27.1 27.2 27.3  
## non-violated 0.5000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 27.4 27.5 27.6 27.7 27.8 27.9  
## non-violated 0.0000000 0.5000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 1.0000000 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 28 28.1 28.2 28.3 28.4 28.5  
## non-violated 1.0000000 0.6666667 1.0000000 1.0000000 0.5000000 1.0000000  
## violated 0.0000000 0.3333333 0.0000000 0.0000000 0.5000000 0.0000000  
##   
## 28.7 28.8 28.9 29.1 29.2 29.5  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.6666667  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.3333333  
##   
## 29.6 29.7 29.9 30 30.1 30.2  
## non-violated 1.0000000 1.0000000 0.3333333 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.6666667 0.0000000 0.0000000 0.0000000  
##   
## 30.3 30.4 30.7 30.8 31 31.1  
## non-violated 1.0000000 1.0000000 0.5000000 0.5000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.5000000 0.5000000 0.0000000 0.0000000  
##   
## 31.2 31.3 31.4 31.5 31.6 31.8  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 32 32.1 32.2 32.4 32.7 32.8  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 32.9 33 33.2 33.3 33.4 33.5  
## non-violated 1.0000000 1.0000000 1.0000000 0.0000000 1.0000000 0.5000000  
## violated 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.5000000  
##   
## 33.6 33.7 33.8 33.9 34 34.1  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000  
##   
## 34.2 34.3 34.4 34.5 34.6 34.7  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 34.8 34.9 35 35.1 35.2 35.4  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 35.5 35.6 35.8 35.9 36.1 36.2  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 36.3 36.5 36.6 36.7 37 37.2  
## non-violated 1.0000000 0.7500000 1.0000000 1.0000000 1.0000000 0.5000000  
## violated 0.0000000 0.2500000 0.0000000 0.0000000 0.0000000 0.5000000  
##   
## 37.3 37.4 37.5 37.6 37.8 38  
## non-violated 0.5000000 0.5000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.5000000 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 38.1 38.2 38.3 38.4 38.5 38.6  
## non-violated 1.0000000 1.0000000 0.5000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000 0.0000000  
##   
## 38.7 38.9 39 39.1 39.2 39.6  
## non-violated 0.5000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 39.7 39.8 40 40.1 40.3 40.4  
## non-violated 0.3333333 0.6666667 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.6666667 0.3333333 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 40.8 40.9 41 41.1 41.2 41.3  
## non-violated 1.0000000 1.0000000 1.0000000 0.6666667 1.0000000 0.5000000  
## violated 0.0000000 0.0000000 0.0000000 0.3333333 0.0000000 0.5000000  
##   
## 41.4 41.6 41.7 41.9 42 42.1  
## non-violated 0.5000000 1.0000000 0.6666667 1.0000000 1.0000000 0.0000000  
## violated 0.5000000 0.0000000 0.3333333 0.0000000 0.0000000 1.0000000  
##   
## 42.3 42.4 42.5 43 43.1 43.2  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 43.3 43.4 43.5 43.6 43.8 44  
## non-violated 1.0000000 1.0000000 1.0000000 0.5000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000  
##   
## 44.1 44.2 44.3 44.4 44.5 44.6  
## non-violated 0.7500000 1.0000000 1.0000000 0.0000000 1.0000000 1.0000000  
## violated 0.2500000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000  
##   
## 44.7 44.8 44.9 45 45.1 45.4  
## non-violated 0.5000000 1.0000000 1.0000000 0.5000000 1.0000000 1.0000000  
## violated 0.5000000 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000  
##   
## 45.6 45.8 45.9 46 46.1 46.3  
## non-violated 1.0000000 0.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 46.4 46.6 46.7 46.8 46.9 47  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000  
##   
## 47.1 47.5 47.8 48 48.2 48.4  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 0.5000000 0.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.5000000 1.0000000  
##   
## 48.5 48.7 48.8 48.9 49 49.3  
## non-violated 1.0000000 1.0000000 0.5000000 1.0000000 1.0000000 0.0000000  
## violated 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000 1.0000000  
##   
## 49.9 50.2 50.6 50.9 51 51.1  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 51.3 51.4 51.8 52.1 53 53.5  
## non-violated 1.0000000 0.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 53.9 54.1 54.4 54.8 54.9 55  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
##   
## 56.5 56.8 58.5 59.4 61.6 65.1  
## non-violated 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000  
## violated 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000

Variable “state”

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

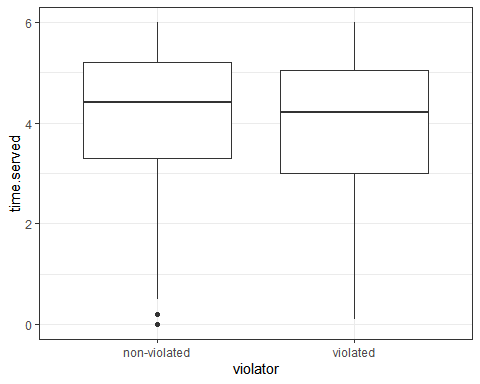


t4 = table(train$violator, train$state)  
prop.table(t4, margin = 2)

##   
## Other state Kentucky Lousiana Virginia  
## non-violated 0.85263158 0.86419753 0.58620690 0.97890295  
## violated 0.14736842 0.13580247 0.41379310 0.02109705

Variable “time.served”

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

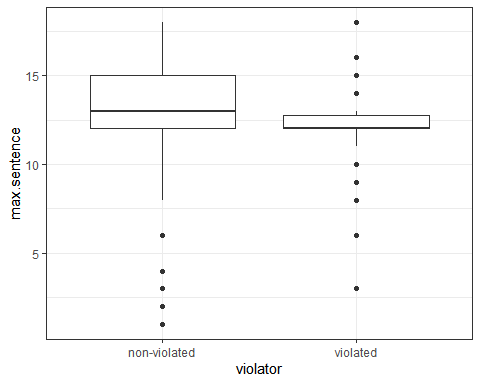


t5 = table(train$violator, train$time.served)  
prop.table(t5, margin = 2)

##   
## 0 0.1 0.2 0.5 0.7  
## non-violated 1.00000000 0.00000000 1.00000000 1.00000000 0.50000000  
## violated 0.00000000 1.00000000 0.00000000 0.00000000 0.50000000  
##   
## 0.8 0.9 1.1 1.2 1.3  
## non-violated 0.50000000 1.00000000 0.66666667 1.00000000 1.00000000  
## violated 0.50000000 0.00000000 0.33333333 0.00000000 0.00000000  
##   
## 1.4 1.5 1.6 1.7 1.8  
## non-violated 0.00000000 1.00000000 0.00000000 0.00000000 1.00000000  
## violated 1.00000000 0.00000000 1.00000000 1.00000000 0.00000000  
##   
## 1.9 2 2.1 2.2 2.3  
## non-violated 1.00000000 0.50000000 1.00000000 0.50000000 1.00000000  
## violated 0.00000000 0.50000000 0.00000000 0.50000000 0.00000000  
##   
## 2.4 2.6 2.7 2.8 2.9  
## non-violated 1.00000000 1.00000000 0.75000000 1.00000000 1.00000000  
## violated 0.00000000 0.00000000 0.25000000 0.00000000 0.00000000  
##   
## 3 3.1 3.2 3.3 3.4  
## non-violated 0.95744681 1.00000000 1.00000000 1.00000000 0.60000000  
## violated 0.04255319 0.00000000 0.00000000 0.00000000 0.40000000  
##   
## 3.5 3.6 3.7 3.8 3.9  
## non-violated 1.00000000 1.00000000 0.80000000 0.90909091 0.80000000  
## violated 0.00000000 0.00000000 0.20000000 0.09090909 0.20000000  
##   
## 4 4.1 4.2 4.3 4.4  
## non-violated 0.84615385 0.93333333 0.80952381 1.00000000 0.88888889  
## violated 0.15384615 0.06666667 0.19047619 0.00000000 0.11111111  
##   
## 4.5 4.6 4.7 4.8 4.9  
## non-violated 0.94444444 0.91666667 0.94444444 0.93333333 0.70588235  
## violated 0.05555556 0.08333333 0.05555556 0.06666667 0.29411765  
##   
## 5 5.1 5.2 5.3 5.4  
## non-violated 1.00000000 0.94736842 0.91304348 0.90909091 0.90000000  
## violated 0.00000000 0.05263158 0.08695652 0.09090909 0.10000000  
##   
## 5.5 5.6 5.7 5.8 5.9  
## non-violated 1.00000000 0.75000000 0.87500000 1.00000000 0.81818182  
## violated 0.00000000 0.25000000 0.12500000 0.00000000 0.18181818  
##   
## 6  
## non-violated 0.82352941  
## violated 0.17647059

Variable “max.sentence”

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

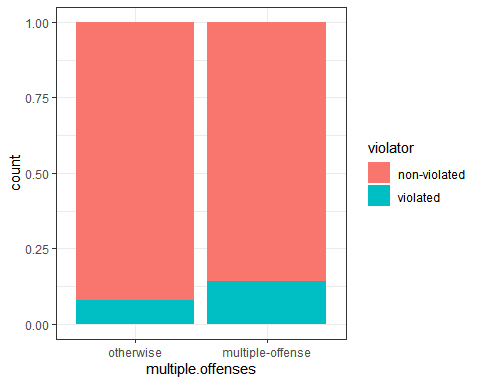


t6 = table(train$violator, train$max.sentence)  
prop.table(t6, margin = 2)

##   
## 1 2 3 4 6  
## non-violated 1.00000000 1.00000000 0.33333333 1.00000000 0.50000000  
## violated 0.00000000 0.00000000 0.66666667 0.00000000 0.50000000  
##   
## 8 9 10 11 12  
## non-violated 0.92307692 0.66666667 0.50000000 0.50000000 0.85567010  
## violated 0.07692308 0.33333333 0.50000000 0.50000000 0.14432990  
##   
## 13 14 15 16 17  
## non-violated 0.96363636 0.94000000 0.97619048 0.81818182 1.00000000  
## violated 0.03636364 0.06000000 0.02380952 0.18181818 0.00000000  
##   
## 18  
## non-violated 0.92307692  
## violated 0.07692308

Variable “multiple.offenses”

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

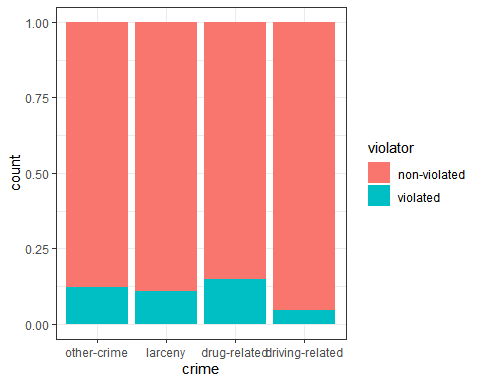


t7 = table(train$violator, train$multiple.offenses)  
prop.table(t7, margin = 2)

##   
## otherwise multiple-offense  
## non-violated 0.91943128 0.85769231  
## violated 0.08056872 0.14230769

Variable “crime”

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



t8 = table(train$violator, train$crime)  
prop.table(t8, margin = 2)

##   
## other-crime larceny drug-related driving-related  
## non-violated 0.87826087 0.89189189 0.85294118 0.95384615  
## violated 0.12173913 0.10810811 0.14705882 0.04615385

Selection of variable: From the all examined variables “state” “max.sentence” and “time.served” seems to be most predictive of the parole violation. The variable “state” seems to be most predictive of violator and will be further examined.

###### Task 3.

# Logistic regresion model using “state” variable.

train\_model =   
 logistic\_reg() %>%  
 set\_engine("glm")

Variable “state”

train\_recipe = recipe(violator ~ state, 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   
## -1.0335 -0.5403 -0.2065 -0.2065 2.7780   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.75539 0.28944 -6.065 1.32e-09 \*\*\*  
## state\_Kentucky -0.09521 0.43471 -0.219 0.826636   
## state\_Lousiana 1.40709 0.39351 3.576 0.000349 \*\*\*  
## state\_Virginia -2.08191 0.53672 -3.879 0.000105 \*\*\*  
## ---  
## 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: 270.95 on 467 degrees of freedom  
## AIC: 278.95  
##   
## Number of Fisher Scoring iterations: 6

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

## # A tibble: 6 x 2  
## `.pred\_non-violated` .pred\_violated  
## <dbl> <dbl>  
## 1 0.853 0.147  
## 2 0.853 0.147  
## 3 0.853 0.147  
## 4 0.853 0.147  
## 5 0.853 0.147  
## 6 0.853 0.147

Variable “max.sentence”

train\_recipe\_max.sent = recipe(violator ~ max.sentence, train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())

logreg\_wf\_max.sent = workflow() %>%  
 add\_recipe(train\_recipe\_max.sent) %>%  
 add\_model(train\_model)

train\_fit\_max.sent = fit(logreg\_wf\_max.sent, train)  
summary(train\_fit\_max.sent$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.9614 -0.5106 -0.4801 -0.3981 2.3743   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.40106 0.53661 -0.747 0.45483   
## max.sentence -0.13089 0.04272 -3.064 0.00219 \*\*  
## ---  
## 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: 326.56 on 469 degrees of freedom  
## AIC: 330.56  
##   
## Number of Fisher Scoring iterations: 5

Variable “time.served”

train\_recipe\_time.serv = recipe(violator ~ time.served, train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())

logreg\_wf\_time.serv = workflow() %>%  
 add\_recipe(train\_recipe\_time.serv) %>%  
 add\_model(train\_model)

train\_fit\_time.serv = fit(logreg\_wf\_time.serv, train)  
summary(train\_fit\_time.serv$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.7113 -0.5176 -0.4697 -0.4359 2.2400   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.2453 0.4591 -2.712 0.00668 \*\*  
## time.served -0.1964 0.1104 -1.780 0.07509 .   
## ---  
## 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: 332.41 on 469 degrees of freedom  
## AIC: 336.41  
##   
## Number of Fisher Scoring iterations: 4

Model with 3 variables “state” “max.sentence” “time.served”

# Evaluate quality of the model.

# Comparing AIC of “state” variable model to other models presented above, it has the lowest value. The lowest AIC value represents better quality of the model.

###### Task 4.

train\_recipe\_all\_var = recipe(violator ~., train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())

logreg\_wf\_all\_var = workflow() %>%  
 add\_recipe(train\_recipe\_all\_var) %>%  
 add\_model(train\_model)

train\_fit\_all\_var = fit(logreg\_wf\_all\_var, train)  
summary(train\_fit\_all\_var$fit$fit$fit)

##   
## Call:  
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6354 -0.3931 -0.2624 -0.1370 2.9521   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.959828 1.201122 -2.464 0.01373 \*   
## age 0.007477 0.016999 0.440 0.66004   
## time.served -0.099097 0.119169 -0.832 0.40565   
## max.sentence 0.066046 0.054472 1.212 0.22533   
## male\_male -0.178372 0.412252 -0.433 0.66525   
## race\_otherwise 1.165290 0.405637 2.873 0.00407 \*\*   
## state\_Kentucky 0.014750 0.501692 0.029 0.97655   
## state\_Lousiana 0.238848 0.555305 0.430 0.66711   
## state\_Virginia -3.771945 0.667998 -5.647 1.64e-08 \*\*\*  
## multiple.offenses\_multiple.offense 1.634887 0.398672 4.101 4.12e-05 \*\*\*  
## crime\_larceny 0.412647 0.515017 0.801 0.42300   
## crime\_drug.related -0.151590 0.415229 -0.365 0.71506   
## crime\_driving.related -0.717667 0.690246 -1.040 0.29847   
## ---  
## 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: 239.68 on 458 degrees of freedom  
## AIC: 265.68  
##   
## Number of Fisher Scoring iterations: 6

# Which variables are significant

###### Task 5.

train\_recipe\_3\_var = recipe(violator ~ state + multiple.offenses + race, train) %>%  
step\_dummy(all\_nominal(), -all\_outcomes())

logreg\_wf\_3\_var = workflow() %>%  
 add\_recipe(train\_recipe\_3\_var) %>%  
 add\_model(train\_model)

train\_fit\_3\_var = fit(logreg\_wf\_3\_var, train)  
summary(train\_fit\_3\_var$fit$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\_Lousiana 0.11876 0.49950 0.238 0.81206   
## state\_Virginia -3.58422 0.63848 -5.614 1.98e-08 \*\*\*  
## multiple.offenses\_multiple.offense 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

# This model seems to perform better than previous model as its AIC is 256.52 which is lower than models with all variables and only state variable. The significant variables are “state\_Virginia”, “multiple.offenses\_multiple.offense” and “race\_otherwise” based on the Pr(>|z|) values.

###### Task 6.

# Probability of parole violation.

Parolee1 = data.frame(state = "Lousiana", multiple.offenses = "multiple-offense", race = "white")  
predict(train\_fit\_3\_var, Parolee1, type ="prob")

## # A tibble: 1 x 2  
## `.pred\_non-violated` .pred\_violated  
## <dbl> <dbl>  
## 1 0.669 0.331

# .pred\_non-violated .pred\_violated

<dbl> <dbl>

# 0.669 0.331

# The probability of parole violation is 0.331

Parolee2 = data.frame(state = "Kentucky", multiple.offenses = "otherwise", race = "otherwise")  
predict(train\_fit\_3\_var, Parolee2, type ="prob")

## # A tibble: 1 x 2  
## `.pred\_non-violated` .pred\_violated  
## <dbl> <dbl>  
## 1 0.798 0.202

# The probability of parole violation is 0.202

###### Task 7

# ROC curve and probability threshold on training set.

# Create prediction and extract 2nd column(.pred\_violated)

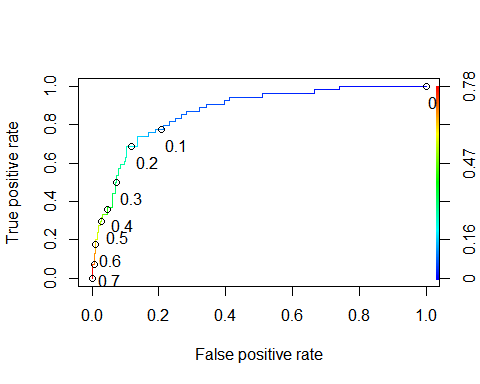
predictionsROCR = predict(train\_fit\_all\_var, train,type = "prob")[2]  
head(predictionsROCR)

## # A tibble: 6 x 1  
## .pred\_violated  
## <dbl>  
## 1 0.0491  
## 2 0.159   
## 3 0.148   
## 4 0.0784  
## 5 0.0788  
## 6 0.266

# Threshold selection

ROCRpred = prediction(predictionsROCR, 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] 0.8690825

# Determine threshold to balance sensitivity and specificity

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 0.7777778  
## specificity 0.8105516  
## cutoff 0.1190439

# The probability threshold that best balances specificity and sensitivity is 0.1190439

###### Task 8.

predictions2 = predict(train\_fit\_all\_var, train, type = "prob")

tab1 = table(train$violator, predictionsROCR > 0.1190439)  
tab1

##   
## FALSE TRUE  
## non-violated 338 79  
## violated 12 42

# Accuracy

(tab1[1,1]+tab1[2,2])/nrow(train)

## [1] 0.8067941

# Accuracy is 0.8067941

# Sensitivity

42/(12+42)

## [1] 0.7777778

# Sensitivity is 0.7777778

# Specificity

338/(338+79)

## [1] 0.8105516

# Specificity is 0.8105516

# Implications of incorrectly clasifying parolee can lead to violating of the term of the parolee. Can lead to increased crime and lack of control.

###### Task 9.

tab1 = table(train$violator, predictionsROCR > 0.5)  
tab1

##   
## FALSE TRUE  
## non-violated 406 11  
## violated 38 16

(tab1[1,1]+tab1[2,2])/nrow(train)

## [1] 0.895966

tab1 = table(train$violator, predictionsROCR > 0.6)  
tab1

##   
## FALSE TRUE  
## non-violated 413 4  
## violated 45 9

(tab1[1,1]+tab1[2,2])/nrow(train)

## [1] 0.895966

tab1 = table(train$violator, predictionsROCR > 0.2)  
tab1

##   
## FALSE TRUE  
## non-violated 368 49  
## violated 17 37

(tab1[1,1]+tab1[2,2])/nrow(train)

## [1] 0.8598726

tab1 = table(train$violator, predictionsROCR > 0.7)  
tab1

##   
## FALSE TRUE  
## non-violated 415 2  
## violated 50 4

(tab1[1,1]+tab1[2,2])/nrow(train)

## [1] 0.8895966

# The most desirable probability threshold to maximize accuracy(0.895966) is between 0.5 and 0.6

###### Task 10.

# Accuracy for threshold of 0.5 is the same as for 0.6. However threshold of 0.5 has specificity 0.97 and sensitivity 0.30 and threshold 0.6 has specificity 0.99 and sensitivity 0.17. The threshold of 0.5 seems to be best selection for this model.