DATA2020PROJECT

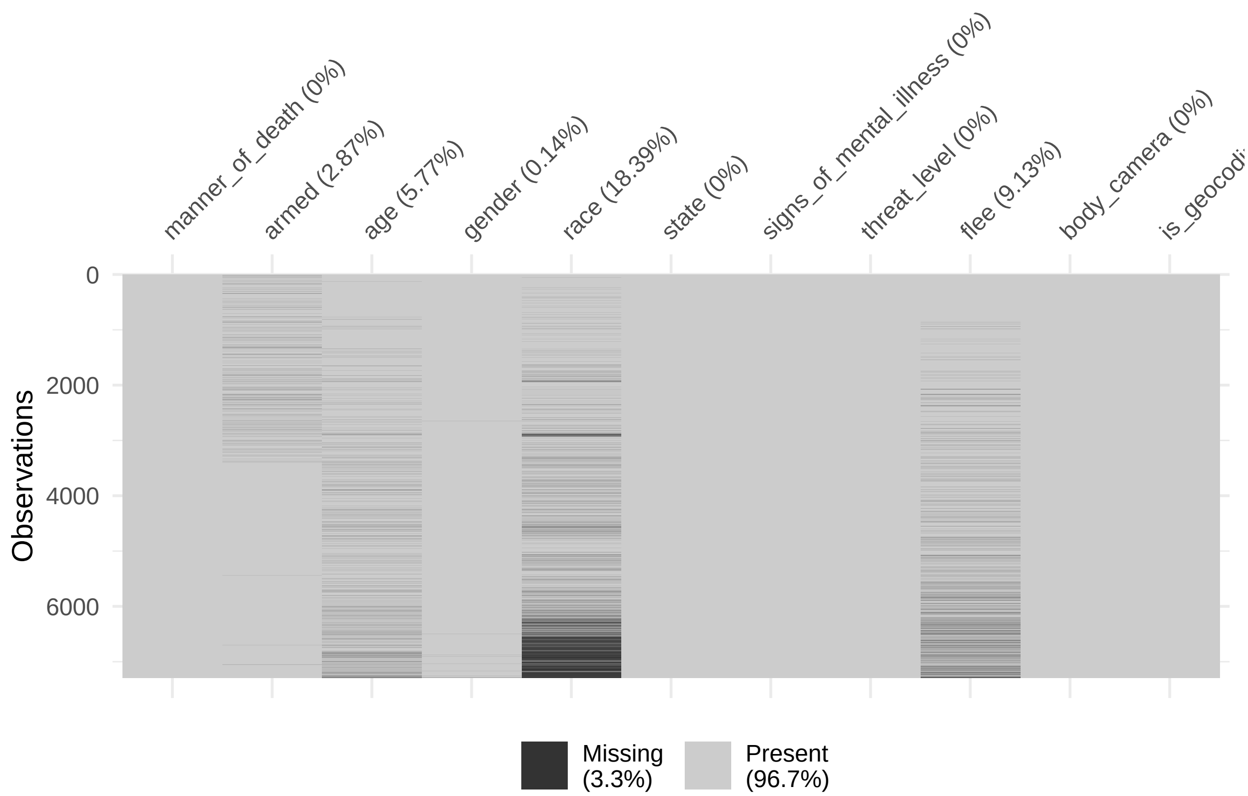
```r
fatal <- read_csv("~/Desktop/fatal-police-shootings-data.csv")
```

```
## Rows: 7291 Columns: 17
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (9): name, manner_of_death, armed, gender, race, city, state, threat_le...
## dbl  (4): id, age, longitude, latitude
## lgl  (3): signs_of_mental_illness, body_camera, is_geocoding_exact
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
fatal <- fatal %>%
  select(-c(id, name, date, longitude, latitude, city))
```

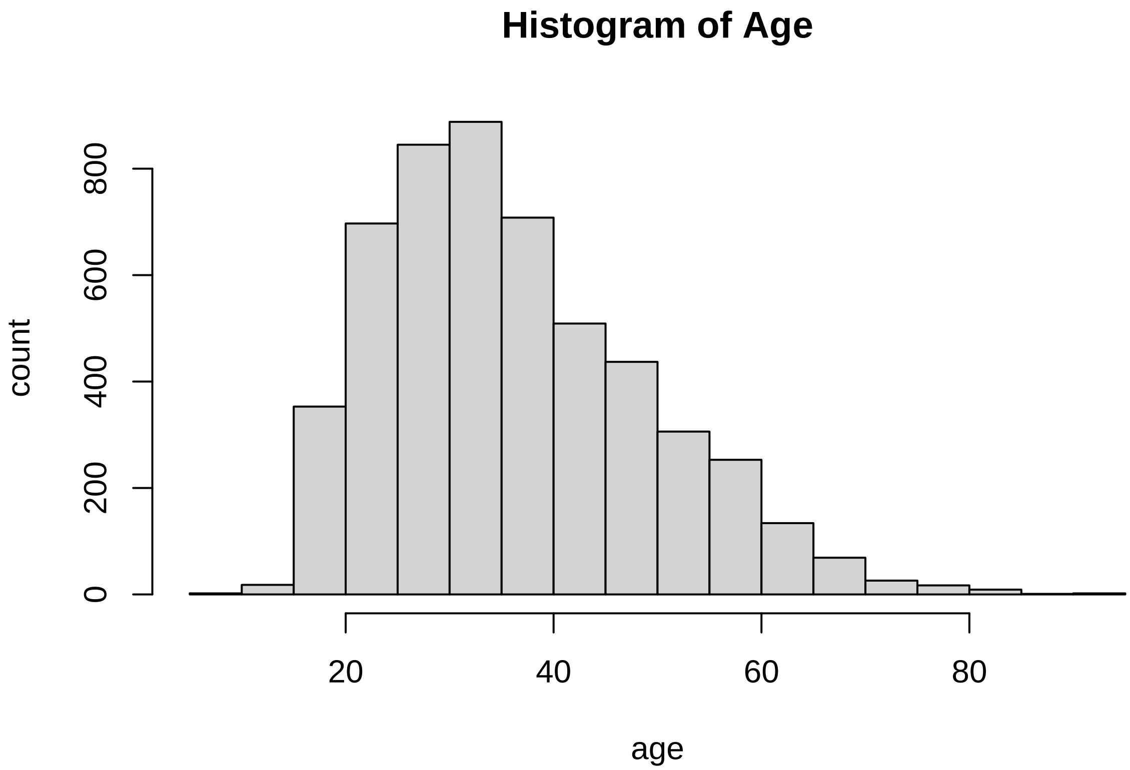
```r
vis_miss(fatal)
```

```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## Please use `gather()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```



```r
fatal = na.omit(fatal)
```

```r
hist(fatal$age, main = "Histogram of Age", xlab = 'age', ylab = 'count')
```

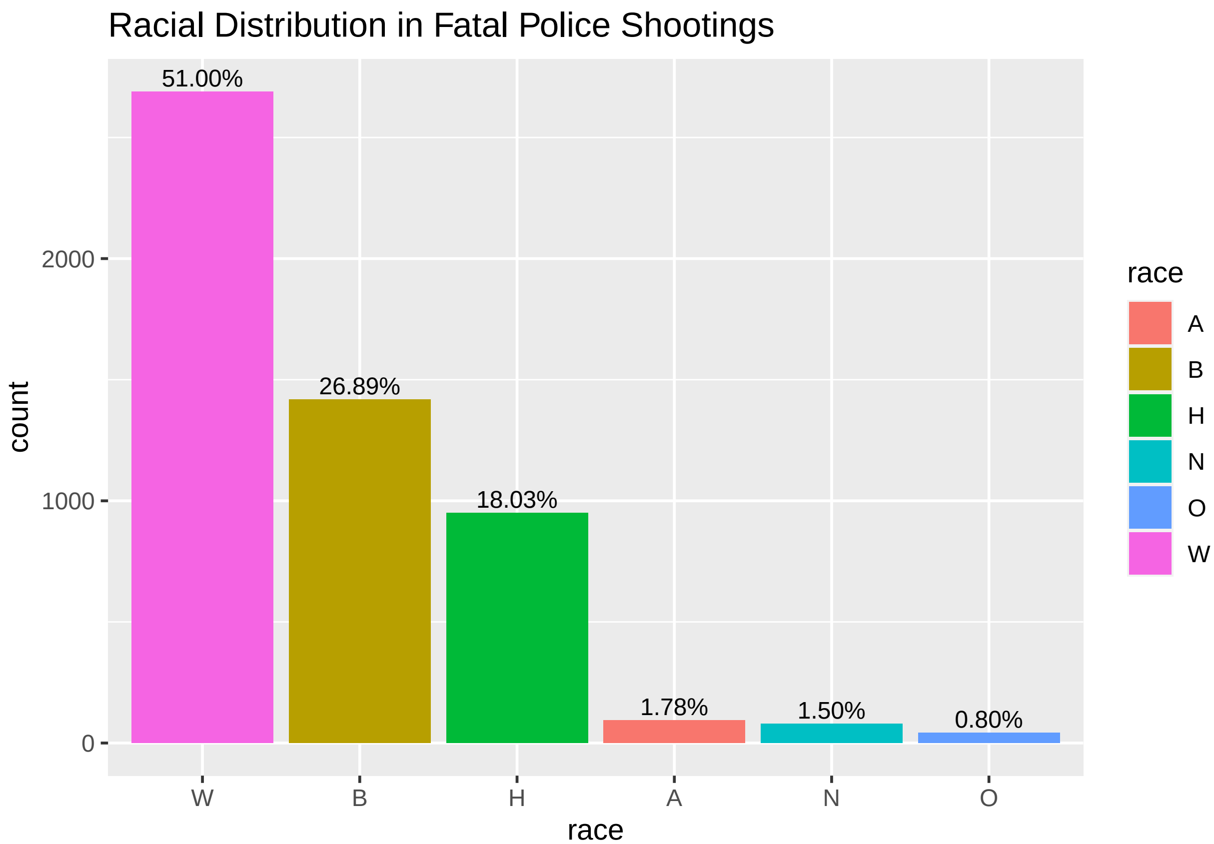



```r
count_race = as.data.frame(table(fatal$race))
colnames(count_race)[colnames(count_race) == "Var1"] <- "race"
colnames(count_race)[colnames(count_race) == "Freq"] <- "count"
race_percentage = numeric(6)
for (i in 1:6){
  race_percentage[i] = count_race$count[i]/sum(count_race$count)
}
count_race['race_percentage'] <- race_percentage
count_race
```

```
##   race count race_percentage
## 1    A    94     0.017823284
## 2    B  1418     0.268866136
## 3    H   951     0.180318544
## 4    N    79     0.014979143
## 5    O    42     0.007963595
## 6    W  2690     0.510049298
```

```r
ggplot(data = count_race, aes(x = reorder(race, -count),
                              y = count,
                              label = scales::percent(race_percentage),
                              fill = race)) +
  geom_bar(stat = 'identity') +
  geom_text(vjust = -0.3,
            size = 3) +
  ggtitle('Racial Distribution in Fatal Police Shootings') +
```

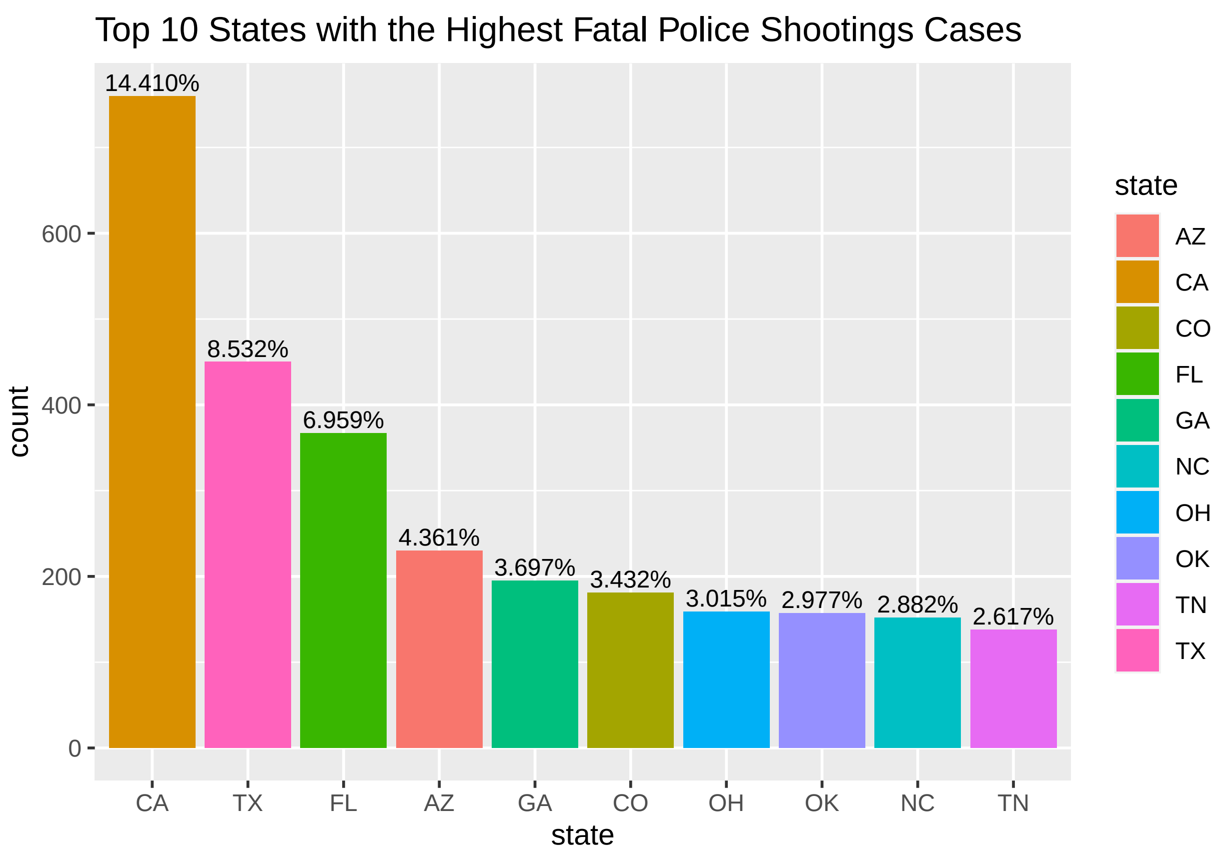
```
  labs(x = 'race', y = 'count')
```




```
count_state = as.data.frame(table(fatal$state))
colnames(count_state)[colnames(count_state) == "Var1"] <- "state"
colnames(count_state)[colnames(count_state) == "Freq"] <- "count"
count_state =count_state[order(-count_state$count),] # order by descending
                                                     # order() returns indices
count_state_10 = count_state[1:10,]
state_percentage = numeric(10)
for (i in 1:10){
  state_percentage[i] = count_state$count[i]/sum(count_state$count)
}
count_state_10['state_percentage'] <- state_percentage
sum(count_state_10$state_percentage)
```

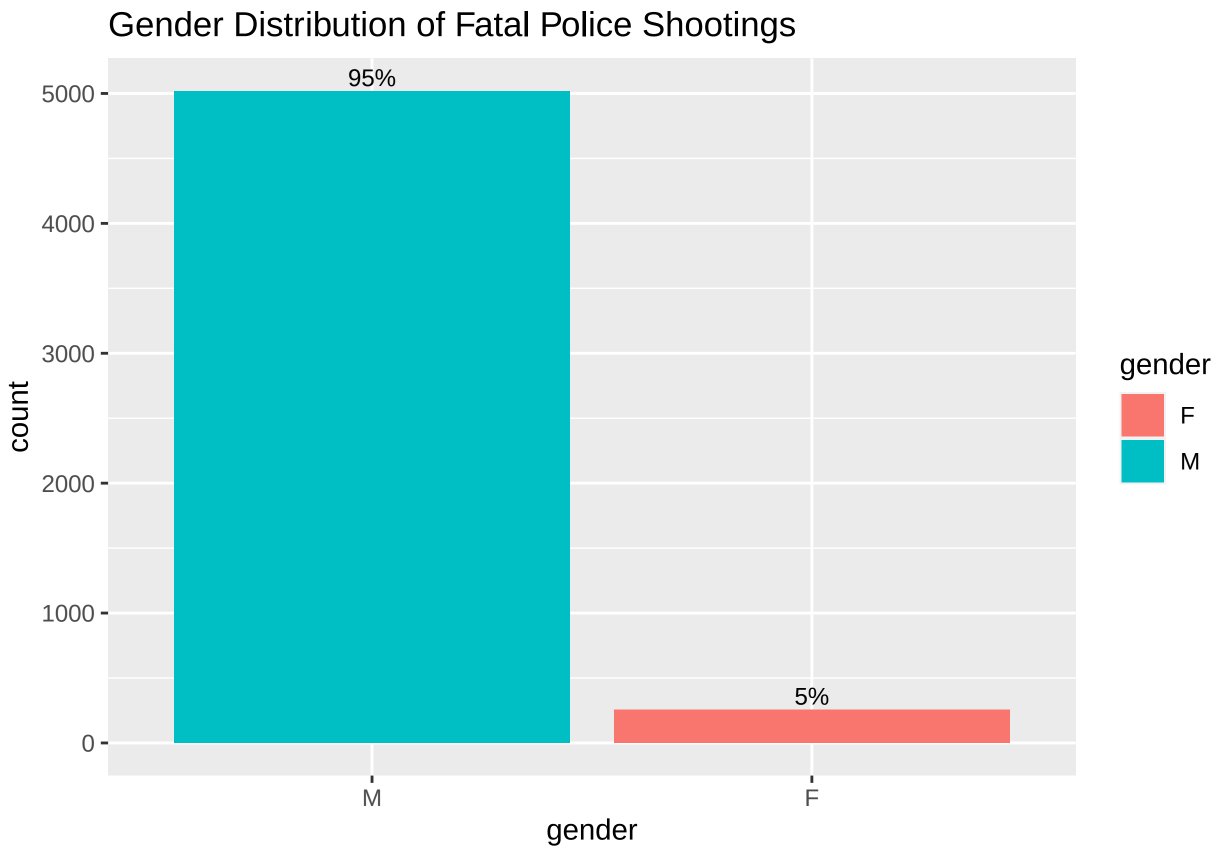
```
## [1] 0.5288206
```

```
ggplot(data = count_state_10, aes(x = reorder(state, -count),
                                  y = count,
                                  label = scales::percent(state_percentage),
                                  fill = state)) +
  geom_bar(stat = 'identity') +
  ggtitle('Top 10 States with the Highest Fatal Police Shootings Cases') +
  geom_text(vjust = -0.3,
            size = 3) +
  labs(x = 'state', y = 'count')
```



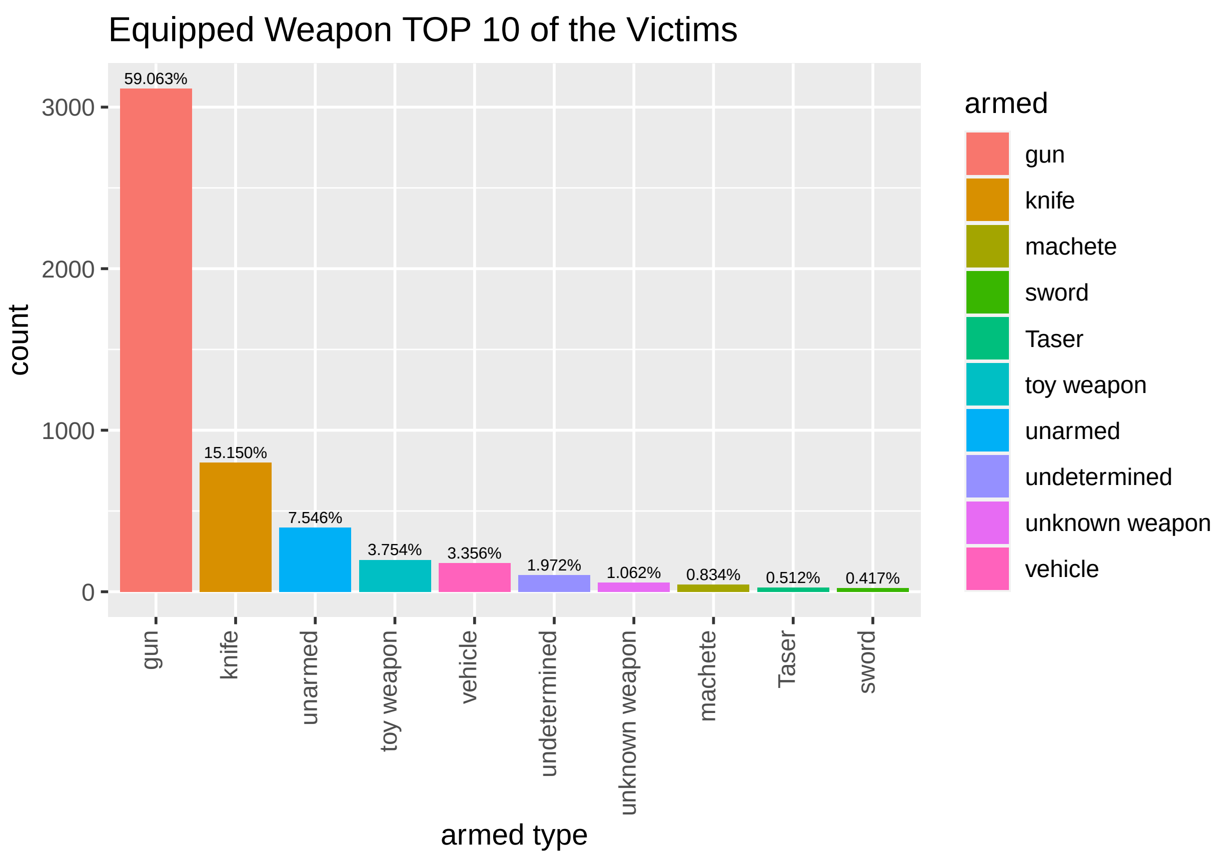

```
count_gender = as.data.frame(table(fatal$gender))
colnames(count_gender)[colnames(count_gender) == "Var1"] <- "gender"
colnames(count_gender)[colnames(count_gender) == "Freq"] <- "count"
count_gender =count_gender[order(-count_gender$count),]
count_gender_2 = count_gender[1:2,]
gender_percentage = numeric(2)
for (i in 1:2){
  gender_percentage[i] = count_gender$count[i]/sum(count_gender$count)
}
count_gender_2['gender_percentage'] <- gender_percentage
```

```
ggplot(data = count_gender_2, aes(x = reorder(gender, -count),
                                  y = count,
                                  label = scales::percent(gender_percentage),
                                  fill = gender)) +
  geom_bar(stat = 'identity') +
  ggtitle('Gender Distribution of Fatal Police Shootings') +
  geom_text(vjust = -0.3,
            size = 3) +
  labs(x = 'gender', y = 'count')
```




```
count_armed = as.data.frame(table(fatal$armed))
colnames(count_armed)[colnames(count_armed) == "Var1"] <- "armed"
colnames(count_armed)[colnames(count_armed) == "Freq"] <- "count"
count_armed =count_armed[order(-count_armed$count),]
count_armed_10 = count_armed[1:10,]
armed_percentage = numeric(10)
for (i in 1:10){
  armed_percentage[i] = count_armed$count[i]/sum(count_armed$count)
}
count_armed_10['armed_percentage'] <- armed_percentage
```

```
ggplot(data = count_armed_10, aes(x = reorder(armed, -count),
                                  y = count,
                                  label = scales::percent(armed_percentage),
                                  fill = armed)) +
  geom_bar(stat = 'identity') +
  ggtitle('Equipped Weapon TOP 10 of the Victims') +
  geom_text(vjust = -0.4,
            size = 2) +
  labs(x = 'armed type', y = 'count') +
  theme(axis.text.x = element_text(angle=90, hjust=1, vjust=0.1))
```




```r
fatal <- fatal %>%
  mutate(race = ifelse(race == 'B', 1, 0))
```

If race is black, encoded as 1. Otherwise, encoded as 0.

```r
fatal <- fatal %>%
    mutate(across(.cols=c(manner_of_death, armed, race, gender, state, signs_of_mental_illness, threat_
```

```r
fatal <- fatal %>%
    mutate(across(.cols=c(age), .fns = ~ (.x-mean(.x))/sd(.x)))
summary(fatal)
```

```
##          manner_of_death           armed            age            gender
##  shot              :4995   gun          :3115   Min.   :-2.4130   F: 255
##  shot and Tasered: 279   knife        : 799   1st Qu.:-0.7636   M:5019
##                            unarmed      : 398   Median :-0.1353
##                            toy weapon   : 198   Mean   : 0.0000
##                            vehicle      : 177   3rd Qu.: 0.6501
##                            undetermined: 104   Max.   : 4.2631
##                            (Other)      : 483
##  race         state       signs_of_mental_illness       threat_level
##  0:3856   CA      : 760   FALSE:4016                attack      :3481
##  1:1418   TX      : 450   TRUE :1258                other       :1673
##           FL      : 367                             undetermined: 120
##           AZ      : 230
##           GA      : 195
##           CO      : 181
##           (Other):3091
```

```
##           flee      body_camera  is_geocoding_exact
##   Car        : 803   FALSE:4512   FALSE:   8
##   Foot       : 777   TRUE : 762   TRUE :5266
##   Not fleeing:3483
##   Other      : 211
##
##
##
```

```
# help to converge
```

We only have one continuous variable which is age.

```
head(fatal)
```

```
## # A tibble: 6 x 11
##   manner_of_death  armed    age gender race  state signs_of_mental~ threat_level
##   <fct>            <fct>  <dbl> <fct>  <fct> <fct> <fct>            <fct>
## 1 shot             gun    1.28  M      0     WA    TRUE             attack
## 2 shot             gun    0.807 M      0     OR    FALSE            attack
## 3 shot and Tasered unar~ -1.08  M      0     KS    FALSE            other
## 4 shot             toy ~ -0.371 M      0     CA    TRUE             attack
## 5 shot             nail~  0.179 M      0     CO    FALSE            attack
## 6 shot             gun   -1.47  M      0     OK    FALSE            attack
## # ... with 3 more variables: flee <fct>, body_camera <fct>,
## #   is_geocoding_exact <fct>
```

```
length(unique(fatal$state))
```

```
## [1] 51
```

We have 51 groups.

```
# Split into test and train sets
set.seed(1)
samp.size = floor(0.8*nrow(fatal))
train.ind = sample(nrow(fatal), size = samp.size)
fatal.train = fatal[train.ind,]
fatal.test = fatal[-train.ind,]
dim(fatal.train)
```

```
## [1] 4219   11
```

```
dim(fatal.test)
```

```
## [1] 1055   11
```

```
model1 = glmer(race ~ manner_of_death + armed + gender + signs_of_mental_illness + threat_level + flee +
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## unable to evaluate scaled gradient
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge: degenerate Hessian with 21 negative eigenvalues
```

```
summary(model1)
```

```
## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from
## not positive definite or contains NA values: falling back to var-cov estimated from RX
```

```
## Warning in vcov.merMod(object, correlation = correlation, sigm = sig): variance-covariance matrix co
```

```
## not positive definite or contains NA values: falling back to var-cov estimated from RX
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: race ~ manner_of_death + armed + gender + signs_of_mental_illness +
##     threat_level + flee + body_camera + is_geocoding_exact +
##     age + (1 | state)
##    Data: fatal
## Control: glmerControl(optimizer = "optimx", optCtrl = list(method = "nlminb"),
##     nAGQ = 9)
##
##      AIC      BIC   logLik deviance df.resid
##   5421.4   6137.6  -2601.7   5203.4     5165
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.6114 -0.5729 -0.3625  0.6096  5.8912
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  state  (Intercept) 1.38     1.175
## Number of obs: 5274, groups:  state, 51
##
## Fixed effects:
##                                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           -17.53592 6540.06942  -0.003   0.9979
## manner_of_deathshot and Tasered         0.26568    0.16070   1.653   0.0983
## armedair pistol                        -0.90166 9228.75517   0.000   0.9999
## armedAirsoft pistol                    -0.86008 7774.80613   0.000   0.9999
## armedax                                14.35268 6540.06944   0.002   0.9982
## armedbarstool                          -0.83703 9240.83220   0.000   0.9999
## armedbaseball bat                      15.78076 6540.06938   0.002   0.9981
## armedbaseball bat and bottle           -0.10596 9241.38227   0.000   1.0000
## armedbaseball bat and fireplace poker  -0.20238 9243.04624   0.000   1.0000
## armedbaseball bat and knife            -0.36531 9250.73923   0.000   1.0000
## armedbaton                             18.09029 6540.06948   0.003   0.9978
## armedBB gun                            15.56056 6540.06941   0.002   0.9981
## armedBB gun and vehicle                -2.56780 9222.38587   0.000   0.9998
## armedbean-bag gun                      -0.76160 9227.60731   0.000   0.9999
## armedbeer bottle                       -0.53636 7999.95941   0.000   0.9999
## armedbinoculars                        33.98765 9252.46603   0.004   0.9971
## armedblunt object                      -0.18205 7108.21541   0.000   1.0000
## armedbottle                            15.28955 6540.06953   0.002   0.9981
## armedbow and arrow                      0.86223 9267.81441   0.000   0.9999
## armedbox cutter                        15.52969 6540.06939   0.002   0.9981
## armedbrick                             -1.12088 7502.65718   0.000   0.9999
## armedcar, knife and mace               -2.50646 9214.24803   0.000   0.9998
## armedcarjack                           -0.72575 9228.38797   0.000   0.9999
## armedchain                             18.09529 6540.06949   0.003   0.9978
## armedchain saw                          0.39584 9247.80336   0.000   1.0000
## armedchainsaw                          -1.28411 9248.56793   0.000   0.9999
## armedchair                             16.09242 6540.06943   0.002   0.9980
## armedcontractor's level                -0.78025 9226.61339   0.000   0.9999
```

```
## armedcordless drill                      0.37948 9255.36203   0.000   1.0000
## armedcrossbow                           -0.44269 7096.22863   0.000   1.0000
## armedcrowbar                            15.14402 6540.06948   0.002   0.9982
## armedfireworks                          -0.70221 9249.80523   0.000   0.9999
## armedflagpole                           34.82120 9237.81695   0.004   0.9970
## armedflashlight                         -0.66149 7920.18059   0.000   0.9999
## armedgarden tool                        -0.81054 9226.16491   0.000   0.9999
## armedglass shard                        -2.41124 7983.26752   0.000   0.9998
## armedgrenade                             1.66270 9279.68670   0.000   0.9999
## armedgun                                15.57795 6540.06935   0.002   0.9981
## armedgun and car                        15.53708 6540.06940   0.002   0.9981
## armedgun and knife                      15.48200 6540.06939   0.002   0.9981
## armedgun and machete                    -1.54514 7855.23581   0.000   0.9998
## armedgun and sword                      -0.99050 9245.52432   0.000   0.9999
## armedgun and vehicle                    15.33946 6540.06938   0.002   0.9981
## armedguns and explosives                17.55515 6540.06948   0.003   0.9979
## armedhammer                             14.40529 6540.06940   0.002   0.9982
## armedhand torch                         -1.19637 9218.74015   0.000   0.9999
## armedhatchet                            15.87049 6540.06940   0.002   0.9981
## armedhatchet and gun                    -0.32641 7951.68973   0.000   1.0000
## armedice pick                           -2.08644 9239.05668   0.000   0.9998
## armedincendiary device                   1.03402 9247.56088   0.000   0.9999
## armedknife                              15.28699 6540.06935   0.002   0.9981
## armedknife and vehicle                  32.13848 9223.02289   0.003   0.9972
## armedlawn mower blade                   14.98670 6540.06951   0.002   0.9982
## armedmachete                            15.45013 6540.06937   0.002   0.9981
## armedmachete and gun                    -1.15734 9240.94621   0.000   0.9999
## armedmeat cleaver                       17.11858 6540.06942   0.003   0.9979
## armedmetal hand tool                    16.73131 6540.06951   0.003   0.9980
## armedmetal object                       -0.88100 7392.66591   0.000   0.9999
## armedmetal pipe                         15.02814 6540.06939   0.002   0.9982
## armedmetal pole                         -0.48547 7211.76719   0.000   0.9999
## armedmetal rake                         -0.31010 9236.60738   0.000   1.0000
## armedmetal stick                        -0.15399 7500.81652   0.000   1.0000
## armedmicrophone                         -1.19672 9240.78978   0.000   0.9999
## armedmotorcycle                         32.02585 9228.83700   0.003   0.9972
## armednail gun                            0.30410 9238.55798   0.000   1.0000
## armedoar                                -1.09930 9232.00753   0.000   0.9999
## armedpellet gun                         17.70677 6540.06953   0.003   0.9978
## armedpen                                -1.18665 9215.36117   0.000   0.9999
## armedpepper spray                       34.20758 9228.79429   0.004   0.9970
## armedpick-axe                           -1.06610 7216.64378   0.000   0.9999
## armedpiece of wood                      14.42211 6540.06945   0.002   0.9982
## armedpipe                               -1.12402 6989.16550   0.000   0.9999
## armedpitchfork                           0.45203 7970.54602   0.000   1.0000
## armedpole                               17.50429 6540.06949   0.003   0.9979
## armedpole and knife                      0.41812 7956.64555   0.000   1.0000
## armedrailroad spikes                    32.17891 9257.67182   0.003   0.9972
## armedrock                               17.17761 6540.06941   0.003   0.9979
## armedsamurai sword                      -1.17948 7209.46405   0.000   0.9999
## armedscissors                           -0.87758 6805.67345   0.000   0.9999
## armedscrewdriver                        16.39150 6540.06938   0.003   0.9980
## armedsharp object                       14.96806 6540.06941   0.002   0.9982
## armedshovel                             -0.84952 6900.50872   0.000   0.9999
```

```
## armedspear                               -0.62353 7999.72596   0.000   0.9999
## armedstapler                             -2.06257 9216.31085   0.000   0.9998
## armedstraight edge razor                 15.26796 6540.06946   0.002   0.9981
## armedsword                               15.47963 6540.06938   0.002   0.9981
## armedTaser                               15.78645 6540.06937   0.002   0.9981
## armedtire iron                           -0.85978 7835.11105   0.000   0.9999
## armedtoy weapon                          15.24628 6540.06936   0.002   0.9981
## armedunarmed                             15.55104 6540.06936   0.002   0.9981
## armedundetermined                        15.28621 6540.06936   0.002   0.9981
## armedunknown weapon                      15.11358 6540.06937   0.002   0.9982
## armedvehicle                             15.73672 6540.06936   0.002   0.9981
## armedvehicle and gun                     14.80690 6540.06946   0.002   0.9982
## armedvehicle and machete                 32.94912 9276.47375   0.004   0.9972
## armedwalking stick                       -0.30901 9236.79585   0.000   1.0000
## armedwasp spray                          -0.95081 9238.62150   0.000   0.9999
## armedwrench                               0.55337 9236.90634   0.000   1.0000
## genderM                                   0.38449    0.18021   2.134   0.0329
## signs_of_mental_illnessTRUE              -0.62543    0.09262  -6.752 1.45e-11
## threat_levelother                        -0.05528    0.08290  -0.667   0.5049
## threat_levelundetermined                  0.40489    0.24456   1.656   0.0978
## fleeFoot                                  0.59594    0.12634   4.717 2.39e-06
## fleeNot fleeing                           0.15936    0.10677   1.493   0.1355
## fleeOther                                -0.12142    0.19363  -0.627   0.5306
## body_cameraTRUE                           0.71119    0.09639   7.378 1.60e-13
## is_geocoding_exactTRUE                    0.11067    0.88820   0.125   0.9008
## age                                      -0.48697    0.03928 -12.396  < 2e-16
##
## (Intercept)
## manner_of_deathshot and Tasered       .
## armedair pistol
## armedAirsoft pistol
## armedax
## armedbarstool
## armedbaseball bat
## armedbaseball bat and bottle
## armedbaseball bat and fireplace poker
## armedbaseball bat and knife
## armedbaton
## armedBB gun
## armedBB gun and vehicle
## armedbean-bag gun
## armedbeer bottle
## armedbinoculars
## armedblunt object
## armedbottle
## armedbow and arrow
## armedbox cutter
## armedbrick
## armedcar, knife and mace
## armedcarjack
## armedchain
## armedchain saw
## armedchainsaw
## armedchair
```

```
## armedcontractor's level
## armedcordless drill
## armedcrossbow
## armedcrowbar
## armedfireworks
## armedflagpole
## armedflashlight
## armedgarden tool
## armedglass shard
## armedgrenade
## armedgun
## armedgun and car
## armedgun and knife
## armedgun and machete
## armedgun and sword
## armedgun and vehicle
## armedguns and explosives
## armedhammer
## armedhand torch
## armedhatchet
## armedhatchet and gun
## armedice pick
## armedincendiary device
## armedknife
## armedknife and vehicle
## armedlawn mower blade
## armedmachete
## armedmachete and gun
## armedmeat cleaver
## armedmetal hand tool
## armedmetal object
## armedmetal pipe
## armedmetal pole
## armedmetal rake
## armedmetal stick
## armedmicrophone
## armedmotorcycle
## armednail gun
## armedoar
## armedpellet gun
## armedpen
## armedpepper spray
## armedpick-axe
## armedpiece of wood
## armedpipe
## armedpitchfork
## armedpole
## armedpole and knife
## armedrailroad spikes
## armedrock
## armedsamurai sword
## armedscissors
## armedscrewdriver
## armedsharp object
```

```
## armedshovel
## armedspear
## armedstapler
## armedstraight edge razor
## armedsword
## armedTaser
## armedtire iron
## armedtoy weapon
## armedunarmed
## armedundetermined
## armedunknown weapon
## armedvehicle
## armedvehicle and gun
## armedvehicle and machete
## armedwalking stick
## armedwasp spray
## armedwrench
## genderM                              *
## signs_of_mental_illnessTRUE          ***
## threat_levelother
## threat_levelundetermined             .
## fleeFoot                             ***
## fleeNot fleeing
## fleeOther
## body_cameraTRUE                      ***
## is_geocoding_exactTRUE
## age                                  ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation matrix not shown by default, as p = 108 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
## optimizer (optimx) convergence code: 0 (OK)
## unable to evaluate scaled gradient
## Model failed to converge: degenerate  Hessian with 21 negative eigenvalues
```

```
model2 = glmer(race ~ gender + signs_of_mental_illness  + flee + body_camera + age + (1|state), data = 
```

```
summary(model2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: race ~ gender + signs_of_mental_illness + flee + body_camera +
##     age + (1 | state)
##    Data: fatal
## Control: glmerControl(optimizer = "optimx", optCtrl = list(method = "nlminb"),
##     nAGQ = 9)
##
##      AIC      BIC   logLik deviance df.resid
##   5343.0   5402.2  -2662.5   5325.0     5265
##
## Scaled residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -2.8301 -0.5821 -0.3740  0.6516  6.3760
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  state  (Intercept) 1.384    1.177
## Number of obs: 5274, groups:  state, 51
##
## Fixed effects:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -1.87179    0.25993  -7.201 5.98e-13 ***
## genderM                     0.40222    0.17661   2.277   0.0228 *
## signs_of_mental_illnessTRUE -0.66275    0.08965  -7.393 1.44e-13 ***
## fleeFoot                    0.54883    0.12046   4.556 5.21e-06 ***
## fleeNot fleeing             0.09548    0.09962   0.958   0.3378
## fleeOther                  -0.12833    0.18946  -0.677   0.4982
## body_cameraTRUE             0.68673    0.09459   7.260 3.87e-13 ***
## age                        -0.48212    0.03864 -12.476  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) gendrM s___TR fleeFt flNtfl flOthr b_TRUE
## genderM     -0.648
## sgns___TRUE -0.040  0.027
## fleeFoot    -0.201 -0.062 -0.024
## fleeNotflng -0.275 -0.029 -0.149  0.644
## fleeOther   -0.126 -0.034  0.001  0.341  0.407
## bdy_cmrTRUE -0.071  0.013 -0.061 -0.023 -0.001 -0.026
## age          0.056 -0.007 -0.023  0.031 -0.103  0.021  0.015
```

```
lrtest(model1, model2)
```

```
## Likelihood ratio test
##
## Model 1: race ~ manner_of_death + armed + gender + signs_of_mental_illness +
##     threat_level + flee + body_camera + is_geocoding_exact +
##     age + (1 | state)
## Model 2: race ~ gender + signs_of_mental_illness + flee + body_camera +
##     age + (1 | state)
##   #Df  LogLik   Df  Chisq Pr(>Chisq)
## 1 109 -2601.7
## 2   9 -2662.5 -100 121.63    0.06972 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value is not significant on 5% significance level, we fail to reject the null and stick with the reduced model model 2.

```
model3 = glmer(race ~ gender + signs_of_mental_illness + flee + body_camera + age + (1|state) + gender::
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## unable to evaluate scaled gradient
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge: degenerate Hessian with 1 negative eigenvalues
```

```
summary(model3)
```

```
## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from f
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Warning in vcov.merMod(object, correlation = correlation, sigm = sig): variance-covariance matrix com
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: race ~ gender + signs_of_mental_illness + flee + body_camera +
##     age + (1 | state) + gender:flee
##    Data: fatal
## Control: glmerControl(optimizer = "optimx", optCtrl = list(method = "nlminb"),
##     nAGQ = 9)
##
##      AIC      BIC   logLik deviance df.resid
##   5340.4   5419.3  -2658.2   5316.4     5262
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.8109 -0.5805 -0.3729  0.6462  6.4507
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  state  (Intercept) 1.384    1.176
## Number of obs: 5274, groups:  state, 51
##
## Fixed effects:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -1.60488    0.39016  -4.113 3.90e-05 ***
## genderM                      0.11689    0.35974   0.325   0.7452
## signs_of_mental_illnessTRUE  -0.66530    0.08979  -7.410 1.26e-13 ***
## fleeFoot                    -1.54352    1.12517  -1.372   0.1701
## fleeNot fleeing             -0.08152    0.40391  -0.202   0.8401
## fleeOther                  -12.89646  368.52397  -0.035   0.9721
## body_cameraTRUE              0.69111    0.09458   7.307 2.73e-13 ***
## age                         -0.48248    0.03867 -12.478  < 2e-16 ***
## genderM:fleeFoot             2.12947    1.13187   1.881   0.0599 .
## genderM:fleeNot fleeing      0.19072    0.41600   0.458   0.6466
## genderM:fleeOther           12.82287  368.52402   0.035   0.9722
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) gendrM s___TR fleeFt flNtfl flOthr b_TRUE age    gndM:F
## genderM     -0.865
## sgns___TRUE -0.027  0.013
## fleeFoot    -0.275  0.300  0.008
## fleeNotflng -0.768  0.833 -0.041  0.267
## fleeOther   -0.001  0.001  0.000  0.000  0.001
## bdy_cmrTRUE -0.042  0.002 -0.062 -0.015  0.000  0.000
## age          0.032  0.001 -0.024  0.000 -0.023  0.000  0.015
## gendrM:flFt  0.274 -0.318 -0.011 -0.994 -0.266  0.000  0.012  0.003
```

```
## gndrM:flNtf  0.747 -0.865  0.004 -0.260 -0.969 -0.001  0.000 -0.002  0.276
## gndrM:flOth  0.001 -0.001  0.000  0.000 -0.001 -1.000  0.000  0.000  0.000
##             gnM:Nf
## genderM
## sgns___TRUE
## fleeFoot
## fleeNotflng
## fleeOther
## bdy_cmrTRUE
## age
## gendrM:flFt
## gndrM:flNtf
## gndrM:flOth  0.001
## optimizer (optimx) convergence code: 0 (OK)
## unable to evaluate scaled gradient
## Model failed to converge: degenerate  Hessian with 1 negative eigenvalues
```

```r
lrtest(model2, model3)
```

```
## Likelihood ratio test
##
## Model 1: race ~ gender + signs_of_mental_illness + flee + body_camera +
##     age + (1 | state)
## Model 2: race ~ gender + signs_of_mental_illness + flee + body_camera +
##     age + (1 | state) + gender:flee
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   9 -2662.5
## 2  12 -2658.2  3 8.6035    0.03506 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value is significant on 5% significance level, we reject the null and stick with the full model.

```r
random_effects = as.data.frame(ranef(model3))
random_effects = random_effects[order(-random_effects$condval),]
random_intercept_top10 = random_effects[1:10,]
random_intercept_top10
```

```
##    grpvar        term grp  condval    condsd
## 8   state (Intercept)  DC 2.597200 0.5367490
## 21  state (Intercept)  MD 1.856567 0.2368605
## 19  state (Intercept)  LA 1.699653 0.2036552
## 35  state (Intercept)  NY 1.403079 0.2141763
## 32  state (Intercept)  NJ 1.363625 0.2771099
## 15  state (Intercept)  IL 1.260613 0.1989739
## 46  state (Intercept)  VA 1.150454 0.2106795
## 11  state (Intercept)  GA 1.096554 0.1516108
## 25  state (Intercept)  MO 1.040313 0.1836298
## 39  state (Intercept)  PA 1.035038 0.2036082
```

```r
random_intercept_bottom10 = random_effects[order(random_effects$condval),][1:10,]
random_intercept_bottom10
```

```
##    grpvar        term grp   condval    condsd
## 33  state (Intercept)  NM -2.116190 0.4955930
## 27  state (Intercept)  MT -1.836926 0.7124800
## 14  state (Intercept)  ID -1.626104 0.6022155
```

```
## 29  state (Intercept)  ND -1.308097 0.8025432
## 42  state (Intercept)  SD -1.265575 0.8023424
## 4   state (Intercept)  AZ -1.248993 0.2397495
## 38  state (Intercept)  OR -1.192308 0.4081365
## 51  state (Intercept)  WY -1.188253 0.8121211
## 12  state (Intercept)  HI -1.175502 0.6467943
## 31  state (Intercept)  NH -1.144006 0.8166288
```

```r
DC_odds = exp(2.5972)
MD_odds = exp(1.856567)
NM_odds = exp(-2.116190)
DC_odds
```

```
## [1] 13.42609
```

```r
MD_odds
```

```
## [1] 6.401722
```

```r
NM_odds
```

```
## [1] 0.1204898
```

```r
probability_DC = DC_odds/(1+DC_odds)
probability_MD = MD_odds/(1+MD_odds)
probability_NM = NM_odds/(1+NM_odds)
probability_DC
```

```
## [1] 0.9306812
```

```r
probability_MD
```

```
## [1] 0.8648963
```

```r
probability_NM
```

```
## [1] 0.1075332
```

```r
comparison = 13.42609/0.1204898
comparison
```

```
## [1] 111.4293
```

If a victim is in Washington, D.C., the odds of the probability being black controlling all the other predictors is the highest and it is 13.42609, which is almost double the odds of the state Maryland (second highest) controlling all the other predictors. Moreover, when we compared the state with the highest random intercept(Washington, D.C.) and the state with the lowest random intercept(New Mexico), DC has almost 111 times higher odds for the victim being classified as black controlling all the other predictors.

```r
prob <- predict(model3, newdata = fatal.test, type = "response", allow.new.levels = TRUE)
pred <- ifelse(prob > 0.5, 1, 0)
actual = fatal.test$race
table(pred, actual)
```

```
##     actual
## pred   0   1
##    0 718 206
##    1  44  87
```

```r
precision = 87/(87 + 44)
recall = 87/(87 + 206)
```

```
precision
```

```
## [1] 0.6641221
```

```
recall
```

```
## [1] 0.2969283
```

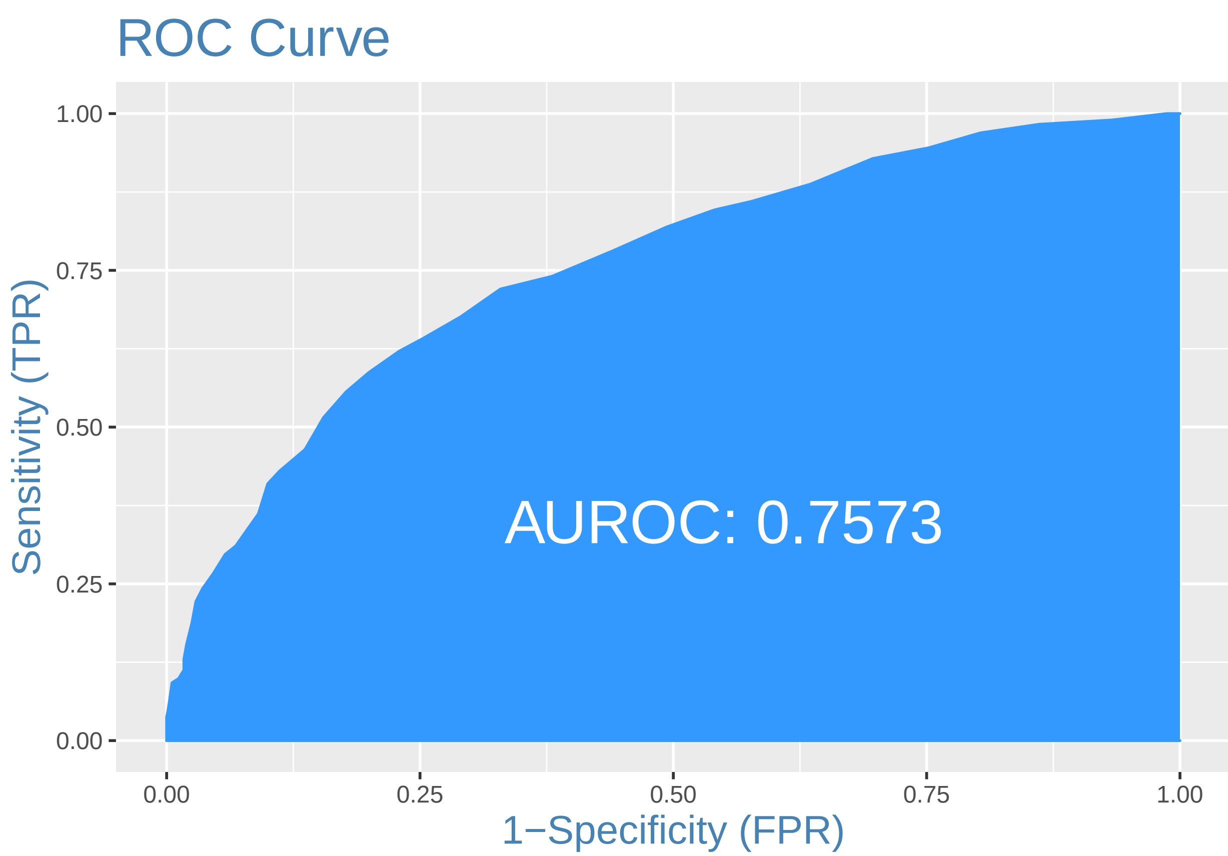
The precision of our model is 66.4% and the recall of our model is 29.7%.

```
sum(pred == fatal.test$race)/nrow(fatal.test)
```

```
## [1] 0.7630332
```

Accuracy for the model 2 is 76.3%.

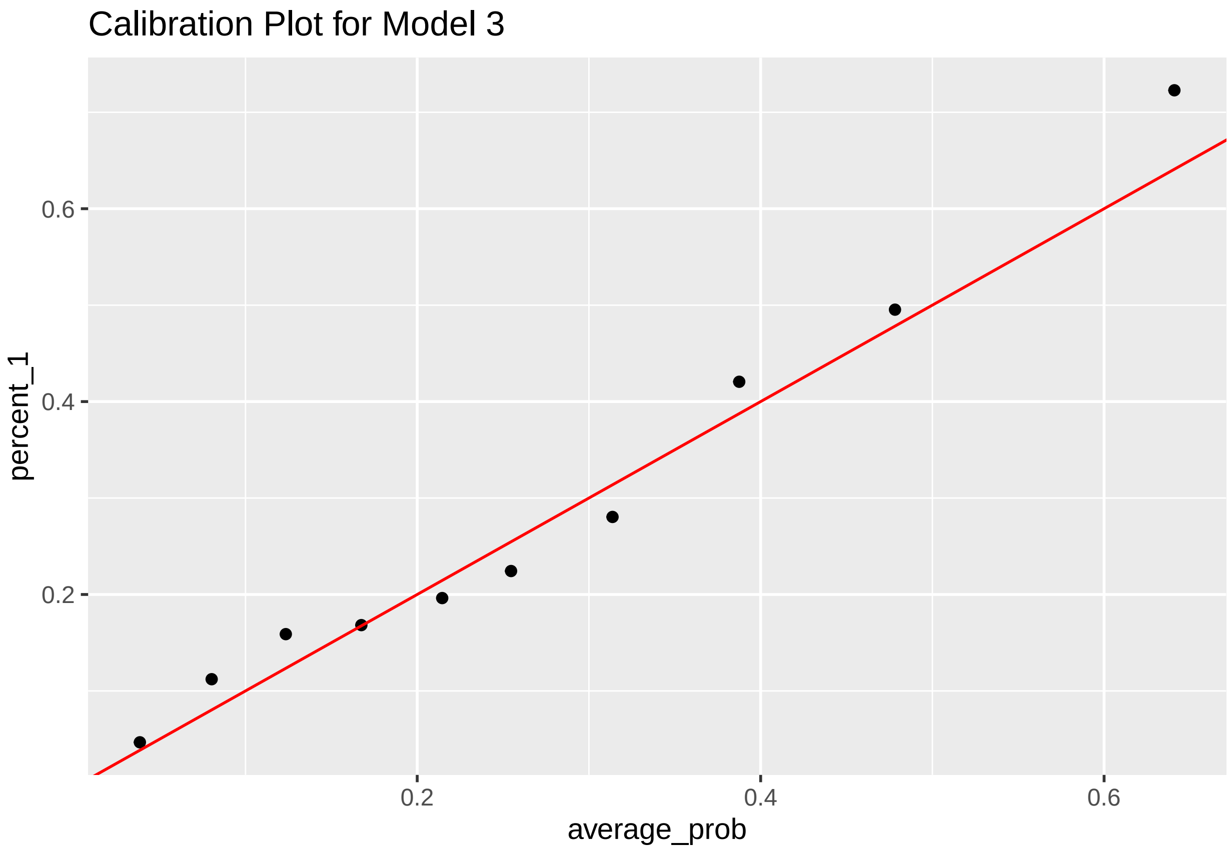
```
plotROC(fatal.test$race, prob, Show.labels=F)
```




```
group_size = ceiling(length(prob)/10)
ordering = order(prob) # order(prob) returns indices, not the actual probability
average_prob = numeric(10)
percent_1 = numeric(10)
for (i in 1:10){
  start = (i-1)*group_size + 1
  end = min(length(prob), start + group_size)
  average_prob[i] = mean(prob[ordering[start:end]])
  percent_1[i] = mean(fatal.test$race[ordering[start:end]] == 1)
}

ggplot()+
```

```
  geom_point(aes(x = average_prob, y = percent_1))+
  geom_abline(aes(slope = 1,intercept = 0), col="red") +
  ggtitle('Calibration Plot for Model 3')
```



From the calibration plot, we can see that the estimated distribution matches the actual distribution on the test data set.

Forward Stepwise Selection without Random Effects

```
model.null = glm(race~1, data = fatal.train, family = 'binomial')
model.full = glm(race~., data = fatal.train, family = 'binomial')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
model.forward = step(model.null, scope=list(lower = model.null,upper = model.full), direction="forward")
```

```
## Start:  AIC=4895.19
## race ~ 1
##
##                          Df Deviance    AIC
## + state                  50   4357.5 4459.5
## + age                     1   4751.5 4755.5
## + flee                    3   4826.7 4834.7
## + signs_of_mental_illness  1   4833.6 4837.6
## + body_camera             1   4864.2 4868.2
## + gender                  1   4880.7 4884.7
## + threat_level            2   4886.7 4892.7
```

```
## <none>                          4893.2 4895.2
## + is_geocoding_exact       1   4892.6 4896.6
## + manner_of_death          1   4893.1 4897.1
## + armed                   89   4759.1 4939.1
##
## Step:  AIC=4459.55
## race ~ state
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                           Df Deviance    AIC
## + age                      1   4197.7 4301.7
## + signs_of_mental_illness  1   4285.2 4389.2
## + flee                     3   4287.0 4395.0
## + body_camera              1   4315.1 4419.1
## + gender                   1   4342.3 4446.3
## + threat_level             2   4353.0 4459.0
## <none>                         4357.5 4459.5
## + is_geocoding_exact       1   4356.9 4460.9
## + manner_of_death          1   4357.4 4461.4
## + armed                   89   4237.4 4517.4
##
## Step:  AIC=4301.74
## race ~ state + age
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                           Df Deviance    AIC
## + signs_of_mental_illness  1   4142.3 4248.3
## + body_camera              1   4161.5 4267.5
## + flee                     3   4157.7 4267.7
## + gender                   1   4184.2 4290.2
## + threat_level             2   4192.9 4300.9
## <none>                         4197.7 4301.7
## + manner_of_death          1   4197.3 4303.3
## + is_geocoding_exact       1   4197.4 4303.4
## + armed                   89   4083.4 4365.4
##
## Step:  AIC=4248.27
## race ~ state + age + signs_of_mental_illness
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                      Df Deviance    AIC
## + body_camera         1   4102.1 4210.1
## + flee                3   4110.2 4222.2
## + gender              1   4131.3 4239.3
## <none>                    4142.3 4248.3
## + manner_of_death     1   4140.9 4248.9
## + threat_level        2   4139.4 4249.4
## + is_geocoding_exact  1   4141.8 4249.8
## + armed              89   4039.2 4323.2
##
## Step:  AIC=4210.14
## race ~ state + age + signs_of_mental_illness + body_camera
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                      Df Deviance    AIC
## + flee                3   4072.0 4186.0
## + gender              1   4090.7 4200.7
## <none>                    4102.1 4210.1
## + threat_level        2   4098.5 4210.5
## + manner_of_death     1   4101.2 4211.2
## + is_geocoding_exact  1   4101.8 4211.8
## + armed              89   3997.2 4283.2
##
## Step:  AIC=4185.98
## race ~ state + age + signs_of_mental_illness + body_camera +
##     flee
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                      Df Deviance    AIC
## + gender              1   4062.4 4178.4
## + threat_level        2   4067.5 4185.5
## <none>                    4072.0 4186.0
## + manner_of_death     1   4071.1 4187.1
## + is_geocoding_exact  1   4071.6 4187.6
## + armed              89   3968.3 4260.3
##
## Step:  AIC=4178.44
## race ~ state + age + signs_of_mental_illness + body_camera +
##     flee + gender
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                      Df Deviance    AIC
## + threat_level        2   4058.4 4178.4
## <none>                    4062.4 4178.4
## + manner_of_death     1   4061.7 4179.7
## + is_geocoding_exact  1   4062.0 4180.0
## + armed              89   3959.5 4253.5
##
## Step:  AIC=4178.37
## race ~ state + age + signs_of_mental_illness + body_camera +
##     flee + gender + threat_level
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                      Df Deviance    AIC
## <none>                    4058.4 4178.4
## + manner_of_death     1   4057.4 4179.4
## + is_geocoding_exact  1   4057.9 4179.9
## + armed              89   3957.8 4255.8
```

```
summary(model.forward)
```

```
##
## Call:
## glm(formula = race ~ state + age + signs_of_mental_illness +
##     body_camera + flee + gender + threat_level, family = "binomial",
##     data = fatal.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.0955  -0.7636  -0.4976   0.7952   3.1275
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -3.31434    0.78108  -4.243 2.20e-05 ***
## stateAL                       1.98852    0.78861   2.522 0.011683 *
## stateAR                       1.57622    0.80658   1.954 0.050677 .
## stateAZ                      -0.12701    0.79951  -0.159 0.873774
## stateCA                       1.07030    0.75487   1.418 0.156231
## stateCO                      -0.10267    0.81844  -0.125 0.900167
## stateCT                       1.04801    1.08904   0.962 0.335887
## stateDC                       5.24053    1.29921   4.034 5.49e-05 ***
## stateDE                       3.01623    1.03747   2.907 0.003646 **
## stateFL                       2.14091    0.75905   2.821 0.004795 **
## stateGA                       2.38519    0.76798   3.106 0.001898 **
## stateHI                     -13.68605  509.41528  -0.027 0.978566
## stateIA                       1.12304    0.88427   1.270 0.204076
## stateID                      -0.97299    1.26411  -0.770 0.441473
## stateIL                       2.52454    0.77869   3.242 0.001187 **
## stateIN                       1.74798    0.78544   2.225 0.026049 *
## stateKS                       0.58129    0.89014   0.653 0.513733
## stateKY                       1.05295    0.81027   1.300 0.193767
## stateLA                       3.03591    0.78384   3.873 0.000107 ***
## stateMA                       1.62244    0.86861   1.868 0.061783 .
## stateMD                       3.19093    0.79252   4.026 5.67e-05 ***
## stateME                      -0.10407    1.28345  -0.081 0.935376
## stateMI                       2.34778    0.79120   2.967 0.003004 **
## stateMN                       0.76439    0.83109   0.920 0.357709
## stateMO                       2.29581    0.77635   2.957 0.003105 **
## stateMS                       2.28297    0.80643   2.831 0.004641 **
## stateMT                     -14.04726  452.00721  -0.031 0.975208
## stateNC                       2.08134    0.77520   2.685 0.007255 **
## stateND                     -14.33662  812.65410  -0.018 0.985925
## stateNE                       0.79640    0.98256   0.811 0.417634
## stateNH                     -13.59144  628.31228  -0.022 0.982742
## stateNJ                       2.67435    0.80982   3.302 0.000959 ***
## stateNM                      -1.94081    1.25776  -1.543 0.122816
## stateNV                       0.66194    0.82753   0.800 0.423767
## stateNY                       2.56234    0.78751   3.254 0.001139 **
## stateOH                       2.29430    0.77148   2.974 0.002941 **
## stateOK                       1.02247    0.78323   1.305 0.191739
## stateOR                      -0.37599    0.95652  -0.393 0.694263
## statePA                       2.31189    0.78201   2.956 0.003113 **
## stateRI                       2.92816    1.87728   1.560 0.118810
## stateSC                       1.88302    0.79124   2.380 0.017321 *
## stateSD                     -13.68267  747.15212  -0.018 0.985389
## stateTN                       1.57305    0.78269   2.010 0.044453 *
## stateTX                       1.42021    0.75858   1.872 0.061180 .
## stateUT                       0.15064    0.86928   0.173 0.862417
## stateVA                       2.39138    0.78438   3.049 0.002298 **
## stateVT                     -13.88533  785.70638  -0.018 0.985900
## stateWA                       0.83598    0.79826   1.047 0.294984
## stateWI                       1.53673    0.80268   1.914 0.055557 .
## stateWV                       1.16234    0.86255   1.348 0.177800
```

```
## stateWY                      -13.79462  782.59472  -0.018 0.985937
## age                           -0.44596    0.04314 -10.338  < 2e-16 ***
## signs_of_mental_illnessTRUE  -0.67556    0.10172  -6.641 3.11e-11 ***
## body_cameraTRUE               0.67224    0.10590   6.348 2.19e-10 ***
## fleeFoot                      0.62141    0.13788   4.507 6.58e-06 ***
## fleeNot fleeing               0.09097    0.11343   0.802 0.422561
## fleeOther                    -0.06429    0.21016  -0.306 0.759687
## genderM                       0.61060    0.21077   2.897 0.003767 **
## threat_levelother            -0.12384    0.08647  -1.432 0.152102
## threat_levelundetermined      0.30471    0.24133   1.263 0.206729
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4893.2  on 4218  degrees of freedom
## Residual deviance: 4058.4  on 4159  degrees of freedom
## AIC: 4178.4
##
## Number of Fisher Scoring iterations: 15
```

```
forward_coefficients = as.data.frame(coef(model.forward))
colnames(forward_coefficients)[1] = "coefficient"
forward_coefficients['state'] <- rownames(forward_coefficients)
forward_coefficients = forward_coefficients[order(-forward_coefficients$coefficient),]
forward_coefficients_top10 = forward_coefficients[1:10,]
forward_coefficients_top10
```

```
##         coefficient   state
## stateDC    5.240528 stateDC
## stateMD    3.190929 stateMD
## stateLA    3.035911 stateLA
## stateDE    3.016233 stateDE
## stateRI    2.928156 stateRI
## stateNJ    2.674353 stateNJ
## stateNY    2.562339 stateNY
## stateIL    2.524541 stateIL
## stateVA    2.391383 stateVA
## stateGA    2.385193 stateGA
```

```
forward_coefficients = forward_coefficients[order(forward_coefficients$coefficient),]
forward_coefficients_bottom10 = forward_coefficients[1:10,]
forward_coefficients_bottom10
```

```
##             coefficient       state
## stateND     -14.3366207     stateND
## stateMT     -14.0472566     stateMT
## stateVT     -13.8853277     stateVT
## stateWY     -13.7946228     stateWY
## stateHI     -13.6860520     stateHI
## stateSD     -13.6826684     stateSD
## stateNH     -13.5914421     stateNH
## (Intercept)  -3.3143445 (Intercept)
## stateNM      -1.9408092     stateNM
## stateID      -0.9729923     stateID
```

We can see that for the forward stepwise selection model, state DC, MD, and LA also have the highest coefficients, which mean that these states have the highest odds/log odds when controlling all the other predictors to 0.

```
forward_DC = exp(-3.31434 + 5.240528)
forward_MD = exp(-3.31434 + 3.190929)
forward_ND = exp(-3.31434 - 14.3366207)
forward_DC
```

```
## [1] 6.863297
```

```
forward_MD
```

```
## [1] 0.8839003
```

```
forward_ND
```

```
## [1] 2.159162e-08
```

```
comparison_forward = 6.863297/2.159162e-08
comparison_forward
```

```
## [1] 317868553
```

```
probability_DC_forward = forward_DC/(1+forward_DC)
probability_MD_forward = forward_MD/(1+forward_MD)
probability_ND_forward = forward_ND/(1+forward_ND)
probability_DC_forward
```

```
## [1] 0.8728269
```

```
probability_MD_forward
```

```
## [1] 0.4691863
```

```
probability_ND_forward
```

```
## [1] 2.159162e-08
```

```
prob_forward <- predict(model.forward, newdata = fatal.test, type = "response")
pred_forward <- ifelse(prob_forward > 0.5, 1, 0)
actual = fatal.test$race
table(pred_forward, actual)
```

```
##             actual
## pred_forward   0   1
##            0 710 205
##            1  52  88
```

```
precision_forward = 88/(88 + 52)
recall_forward = 88/(88 + 205)
precision_forward
```

```
## [1] 0.6285714
```

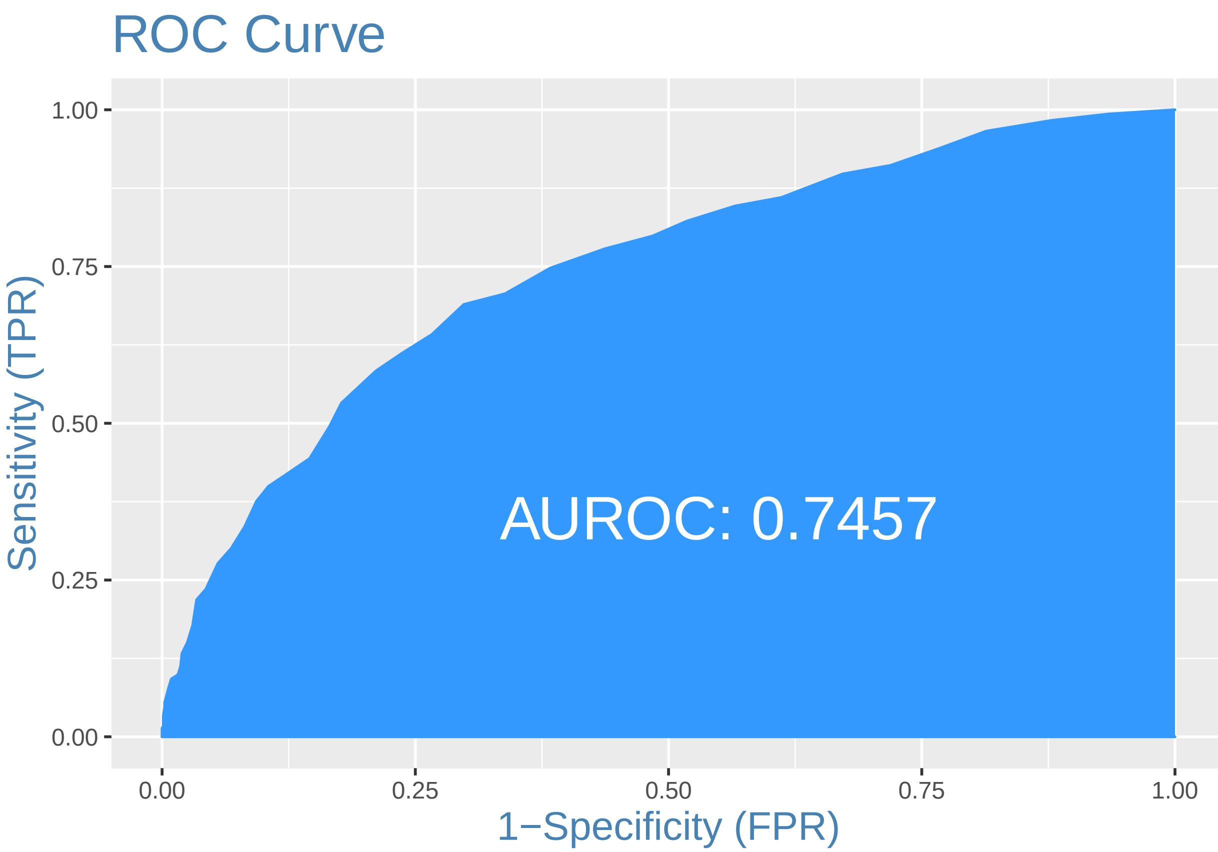
```
recall_forward
```

```
## [1] 0.3003413
```

```
sum(pred_forward == fatal.test$race)/nrow(fatal.test)
```

```
## [1] 0.7563981
```

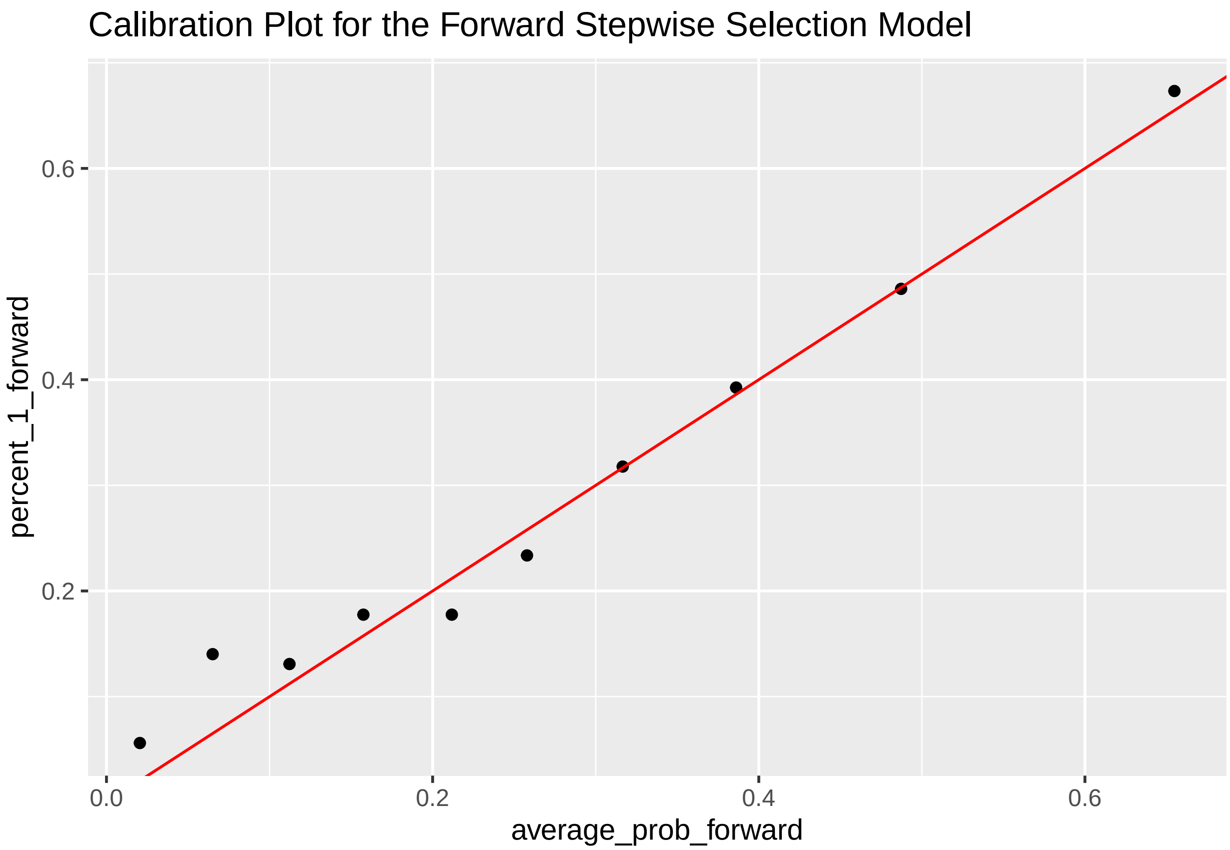
```
plotROC(fatal.test$race, prob_forward, Show.labels=F)
```



The accuracy and AUROC for the foward stepwise selection model is worse than the mixed effect model.

```
group_size_forward = ceiling(length(prob_forward)/10)
ordering_forward = order(prob_forward)
# order(prob_forward) returns indices, not the actual probability
average_prob_forward = numeric(10)
percent_1_forward = numeric(10)
for (i in 1:10){
  start = (i-1)*group_size_forward + 1
  end = min(length(prob_forward), start + group_size_forward)
  average_prob_forward[i] = mean(prob_forward[ordering_forward[start:end]])
  percent_1_forward[i] = mean(fatal.test$race[ordering_forward[start:end]] == 1)
}

ggplot()+
  geom_point(aes(x = average_prob_forward, y = percent_1_forward))+
  geom_abline(aes(slope = 1,intercept = 0), col="red") +
  ggtitle('Calibration Plot for the Forward Stepwise Selection Model')
```



The calibration plot for the forward stepwise selection model is not bad.

```
precision = c(0.6641221, 0.6285714)
recall = c(0.2969283, 0.3003413)
accuracy = c(0.7630332, 0.7563981)
AUROC = c(0.7573, 0.7457)
df = data.frame(percent(precision), percent(recall), percent(accuracy), percent(AUROC))
rownames(df)[1] = "mixed effects model"
rownames(df)[2] = "model using FSS"
colnames(df)[1] = "precision"
colnames(df)[2] = "recall"
colnames(df)[3] = "accuracy"
colnames(df)[4] = "AUROC"
df
```

```
##                      precision recall accuracy AUROC
## mixed effects model      66.4% 29.69%   76.30% 75.7%
## model using FSS          62.9% 30.03%   75.64% 74.6%
```