module5

Andrew Estes

2023-02-09

1

Load the data set and perform any necessary cleaning. While the data set should be relatively "clean" already, you may want to think about which variables are numbers, which should be factors, which should be character, etc.

2

Decide on which variables you will consider for your model. There are likely more variables that you can easily use. Which do you believe might be useful? Write the code to select those variables, and in the text include a short explanation for why you chose the variables you did. (Note: you may decide later to only use some of these variables, but this step is about identifying the largest set of variable's you'll consider.)

```
pacman::p_load(knitr,
               tidymodels,
               tidyverse,
               ggplot2,
               leaps,
               car,
               caret,
               rpart,
               rpart.plot,
               randomForest,
               pROC,
               #MASS,
               #factoextra,
               #FactoMineR,
               #vcd,
               dplyr,
               hablar,
               cvms,
               ranger,
               readxl,
               lubridate,
               gridExtra,
               cowplot,
               patchwork,
               naivebayes,
               psych)
```

```r
orig.df <- read.csv("compas-scores-two-years-w-marital-bw-cleaned.csv")
#sapply(orig.df, class)

factorColumns <- c("sex", "age_cat", "race", "marital_status", "c_charge_degree", "score_text", "v_scor

timeColumns <- c("dob", "c_jail_in", "c_jail_out", "c_offense_date",
                 "c_arrest_date", "screening_date", "r_offense_date",
                 "r_jail_in", "r_jail_out", "vr_offense_date")

removeColumns <- c("in_custody", "out_custody", "r_case_number",
                   "c_case_number", "type_of_assessment",
                   "v_type_of_assessment", "vr_case_number",
                   "is_recid", "id", "event", "r_days_from_arrest",
                   "screening_date", "r_offense_date", "r_jail_in",
                   "r_jail_out", "r_charge_desc", "r_charge_degree",
                   "vr_charge_degree", "vr_charge_desc",
                   "vr_offense_date", "dob", "c_jail_in", "c_jail_out",
                   "c_offense_date", "c_arrest_date", "screening_date",
                   "r_offense_date", "r_jail_in", "r_jail_out",
                   "vr_offense_date", "is_violent_recid")

df <- orig.df
df <- orig.df %>%
  mutate_each_(funs(factor(.)), factorColumns) %>%
  mutate_at(vars(timeColumns), funs(as.Date(., "%Y-%m-%d"))) %>%
  mutate(two_year_recid = as.numeric(two_year_recid)) %>%
  dplyr::select(-c(removeColumns))
```

```
## Warning: `mutate_each_()` was deprecated in dplyr 0.7.0.
## i Please use `across()` instead.
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## i Please use a list of either functions or lambdas:
##
## # Simple named list: list(mean = mean, median = median)
##
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)
##
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(timeColumns)
##
##   # Now:
##   data %>% select(all_of(timeColumns))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## i Please use a list of either functions or lambdas:
```

```
##
## # Simple named list: list(mean = mean, median = median)
##
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)
##
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(removeColumns)
##
##   # Now:
##   data %>% select(all_of(removeColumns))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
#transforming logical variables to integers
#cols <- sapply(df, is.logical)

#removing all date columns and character columns
df <- df[, sapply(df, class) != "Date"]
df <- df[, !sapply(df, is.character)]

sapply(df, class)
```

```
##                    sex                    age                age_cat
##               "factor"              "integer"               "factor"
##                   race          juv_fel_count         juv_misd_count
##               "factor"              "integer"              "integer"
##        juv_other_count           priors_count         marital_status
##              "integer"              "integer"               "factor"
##       c_days_from_compas        c_charge_degree days_b_screening_arrest
##              "integer"               "factor"              "integer"
##           decile_score             score_text         v_decile_score
##              "integer"               "factor"              "integer"
##           v_score_text                  start                    end
##               "factor"              "integer"              "integer"
##          two_year_recid
##              "numeric"
```

The "in_custody" and "out_custody" variables because they were copies of the "c_jail_in" and "c_jail_out" variables.

All of the case numbers from the dataset were removed as well as that isn't pertinent without corresponding judicial information. And, relevant to this class, it helps with privacy concerns.

The assessments were removed because there was no difference in the type of assessment utilized.

The charge description remained as characters because there might be some NLP analysis that could be useful.

All the columns that were dates but classified as characters were re-characterized as dates. I think the dates could be interesting to analyze. Maybe there is a correlation between month and recidivism. But I deleted them in the interest of timeliness.

The factor classification was applied to the relevant columns, such as race, marital status, etc. I thought about changing the Decile information into a factor but decided to keep it as an integer as it is a numeric calculation.

And finally, all logical columns became numeric for the purposes of running a logistic regression on the two_year_recid.

After running the models, a bunch of columns had to be removed due to high correlation with the two_year_recid.

3

Create tables or graphs to visualize the values of your variables. Ideally, data visualizations should start to give you an idea of which variables are important and how they might relate to the response variable and each other. At minimum, you should do the following.

A) Create tables or graphs to understand the distributions of each individual variable. For example, are there the same number of African-American and Caucasian arrestees in the data set?

B) Create graphs that look at how each explanatory variable is related to the response variable two_year_recid.

C) Write a paragraph or two that points out any features of your tables or graphs that you believe are important. You don't need to describe every graph, but rather comment on aspects that might affect your analysis later.

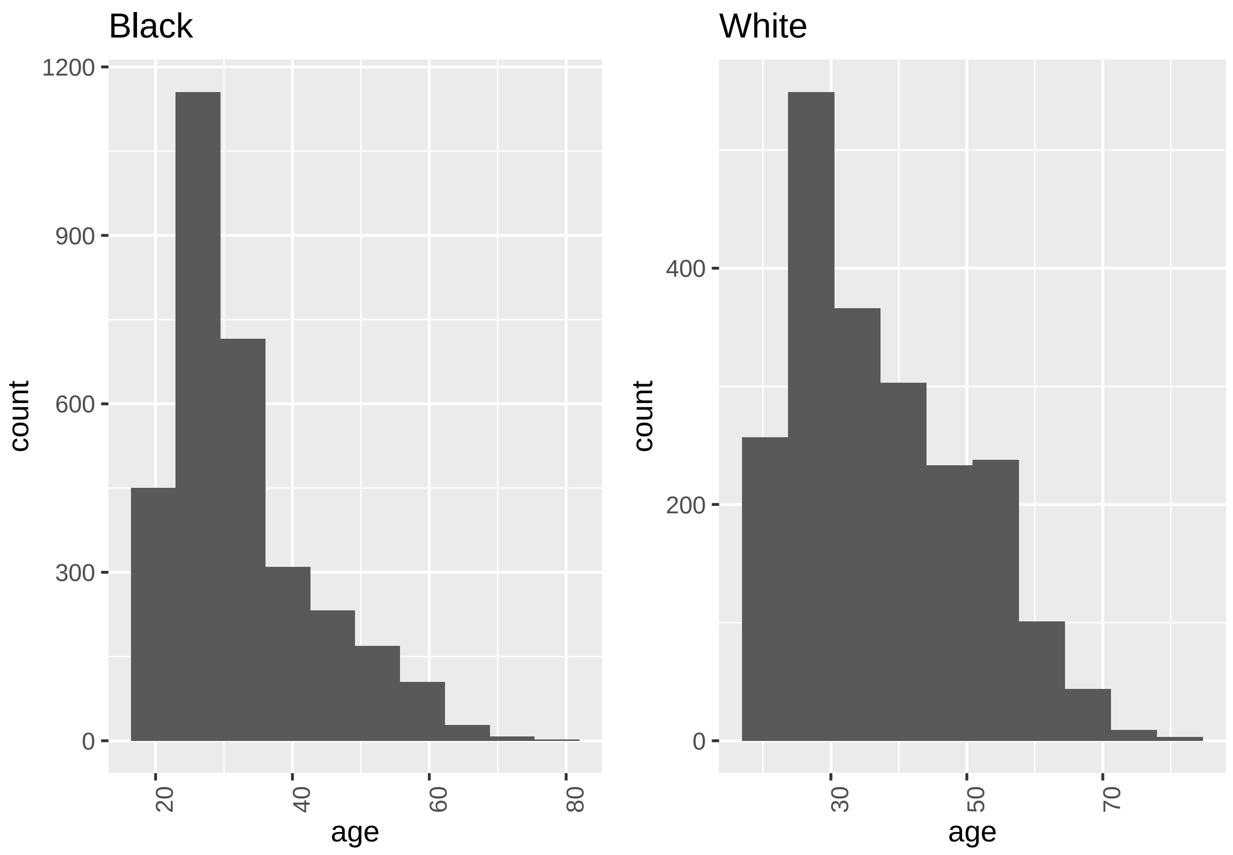
```r
#PCA doesn't work due to non-numerical variables
#FAMD allows for non-numerical variables but cannot handle NAs

#prepping the data for further exploration. this breakdown isn't necessarily needed but is nice to look
df_male <- df %>%
  filter(sex == 'Male')

df_female <- df %>%
  filter(sex == 'Female')

df_black <- df %>%
  filter(race == 'African-American')

df_white <- df %>%
  filter(race == 'Caucasian')

df_black_r <- df_black %>%
  filter(two_year_recid==TRUE)

df_white_r <- df_white %>%
  filter(two_year_recid==TRUE)

df_r <- df %>%
    filter(two_year_recid==TRUE)

################### PART A
#Showing the counts of data for a quick overview

counts <- matrix(c(
  count(df_black), sum(df_black$two_year_recid==TRUE), sum(df_black$sex=='Male'), sum(df_black$sex=='Fe

      count(df_white), sum(df_white$two_year_recid==TRUE), sum(df_white$sex=='Male'), sum(df_white$sex=

    count(df_black_r), NA, sum(df_black_r$sex=='Male'), sum(df_black_r$sex=='Female'), sum(df_black_r$c

      count(df_white_r), NA, sum(df_white_r$sex=='Male'), sum(df_white_r$sex=='Female'), sum(df_white_r$

    ncol=4)
```

```r
colnames(counts) <- c("Black", "White", "Black_Recided", "White_Recided")
rownames(counts) <- c("Total", "Recided", "Male", "Female", "Felony", "Misdemeanor", "Juvenile Felony",

kable(counts)
```

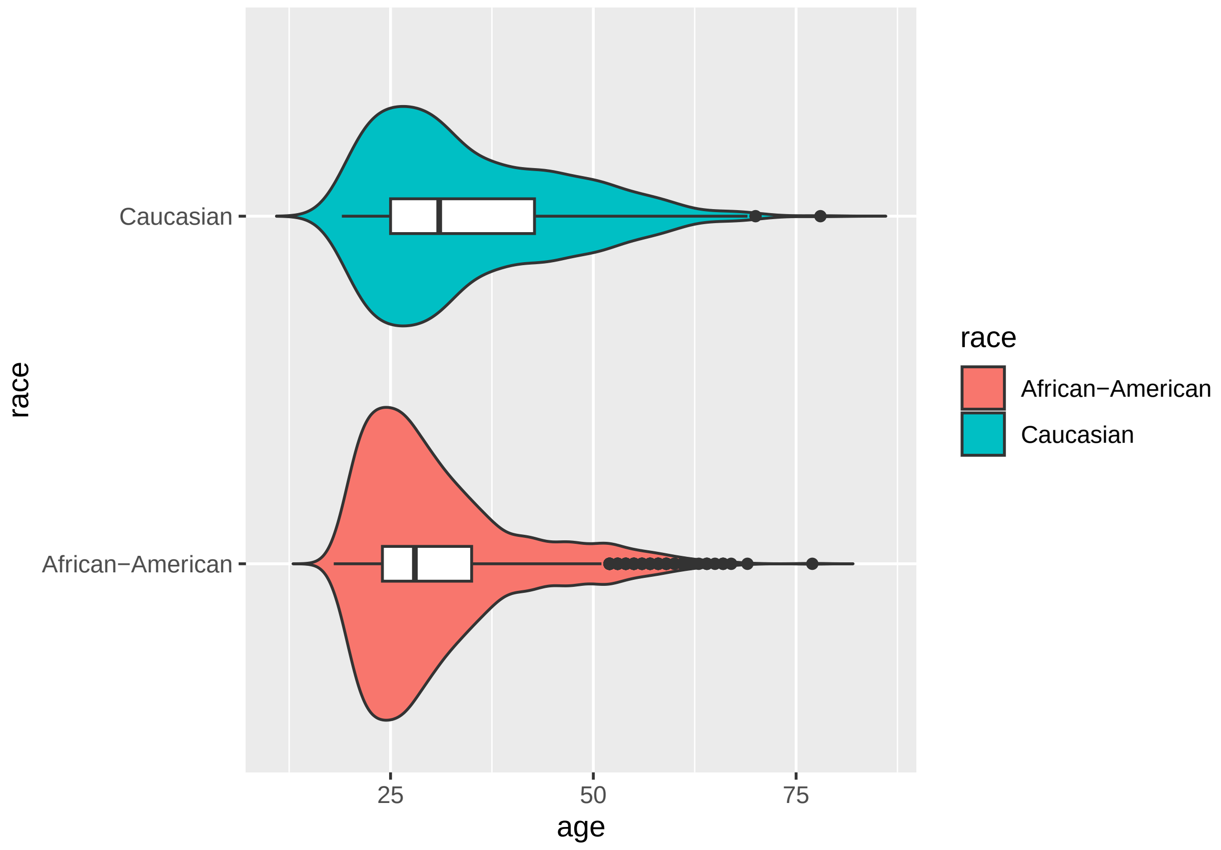
	Black	White	Black_Recided	White_Recided
Total	3175	2103	1661	822
Recided	1661	822	NA	NA
Male	2626	1621	1458	652
Female	549	482	203	170
Felony	2196	1244	1217	541
Misdemeanor	979	859	444	281
Juvenile Felony	271	53	208	38
Juvenile Misdemeanor	433	87	335	57
Juvenile Other	423	206	347	114

```r
################### PART B
#Showing the age breakdown by race and age
histogram_black_scaled <- ggplot(df_black, aes(x = age, fill = two_year_recid)) +
  geom_histogram(bins=10) +
  ylim(0, 1150) +
  xlim(10,80) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1), legend.position = "none") +
  ggtitle("Black Scaled")

histogram_white_scaled <- ggplot(df_white, aes(x = age, fill = two_year_recid)) +
  geom_histogram(bins=10) +
  ylim(0, 1150) +
  xlim(10,80) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ggtitle("White Scaled")

histogram_black <- ggplot(df_black, aes(x = age, fill = two_year_recid)) +
  geom_histogram(bins=10) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1), legend.position = "none") +
  ggtitle("Black")

histogram_white <- ggplot(df_white, aes(x = age, fill = two_year_recid)) +
  geom_histogram(bins=10) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ggtitle("White")

grid.arrange(histogram_black, histogram_white, ncol=2)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a ‘group‘ aesthetic or to convert a numerical
##   variable into a factor?
## The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



```
#another way to look at the age/race breakdown
violin <- ggplot(df_r, aes(x = age, y=race, fill=race)) +
  geom_violin(trim=FALSE)
violin + geom_boxplot(width=0.1, fill = "white")
```



```
#plotting different variables against the two_year_recid
marital <-
  ggplot(df, aes(x=factor(marital_status), fill=factor(two_year_recid))) +
  geom_histogram(stat="count") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1), legend.position = "none")
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## ‘binwidth‘, ‘bins‘, and ‘pad‘
```

```
marital_proportion <-
  ggplot(df, aes(x=factor(marital_status), fill=factor(two_year_recid))) +
  geom_histogram(stat = "count", position= "fill") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
## Warning in geom_histogram(stat = "count", position = "fill"): Ignoring unknown
## parameters: ‘binwidth‘, ‘bins‘, and ‘pad‘
```

```
decile <-
  ggplot(df, aes(x=factor(decile_score), fill=factor(two_year_recid))) +
  geom_bar(stat="count") +
  theme(legend.position = "none")

decile_proportion <-
  ggplot(df, aes(x=factor(decile_score), fill=factor(two_year_recid))) +
  geom_histogram(stat = "count", position= "fill")
```

```
## Warning in geom_histogram(stat = "count", position = "fill"): Ignoring unknown
## parameters: ‘binwidth‘, ‘bins‘, and ‘pad‘
```

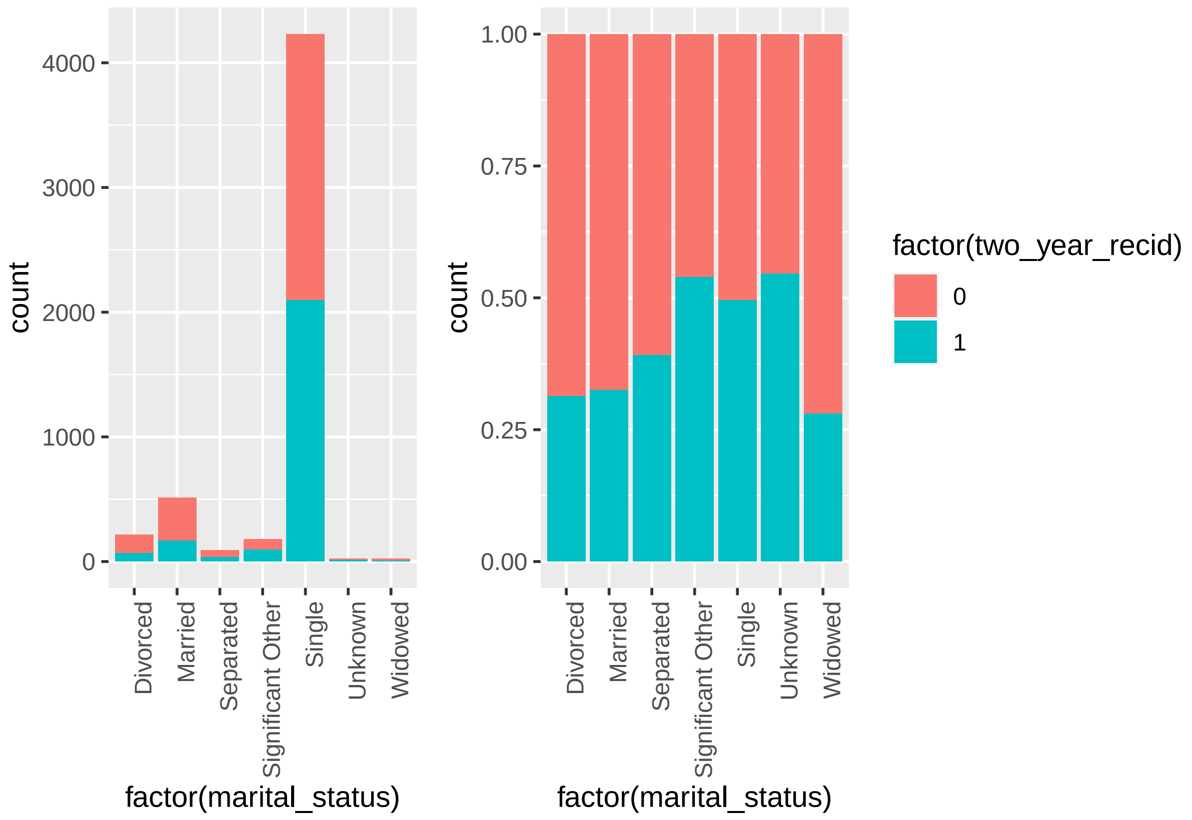
```
decile_text <-
  ggplot(df, aes(x=factor(score_text), fill=factor(two_year_recid))) +
  geom_histogram(stat = "count") +
  theme(legend.position = "none")
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## ‘binwidth‘, ‘bins‘, and ‘pad‘
```

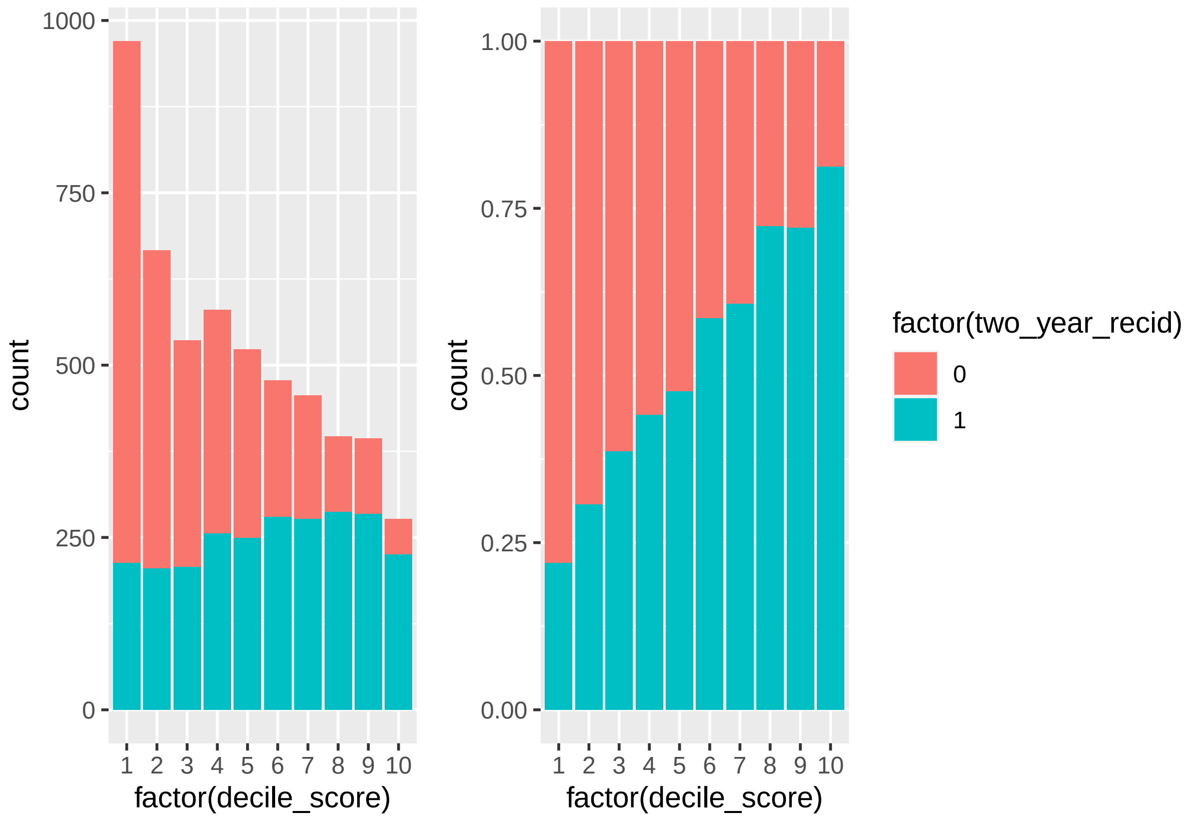
```
decile_text_proportion <-
  ggplot(df, aes(x=factor(score_text), fill=factor(two_year_recid))) +
  geom_histogram(stat = "count", position= "fill")
```

```
## Warning in geom_histogram(stat = "count", position = "fill"): Ignoring unknown
## parameters: ‘binwidth‘, ‘bins‘, and ‘pad‘
```

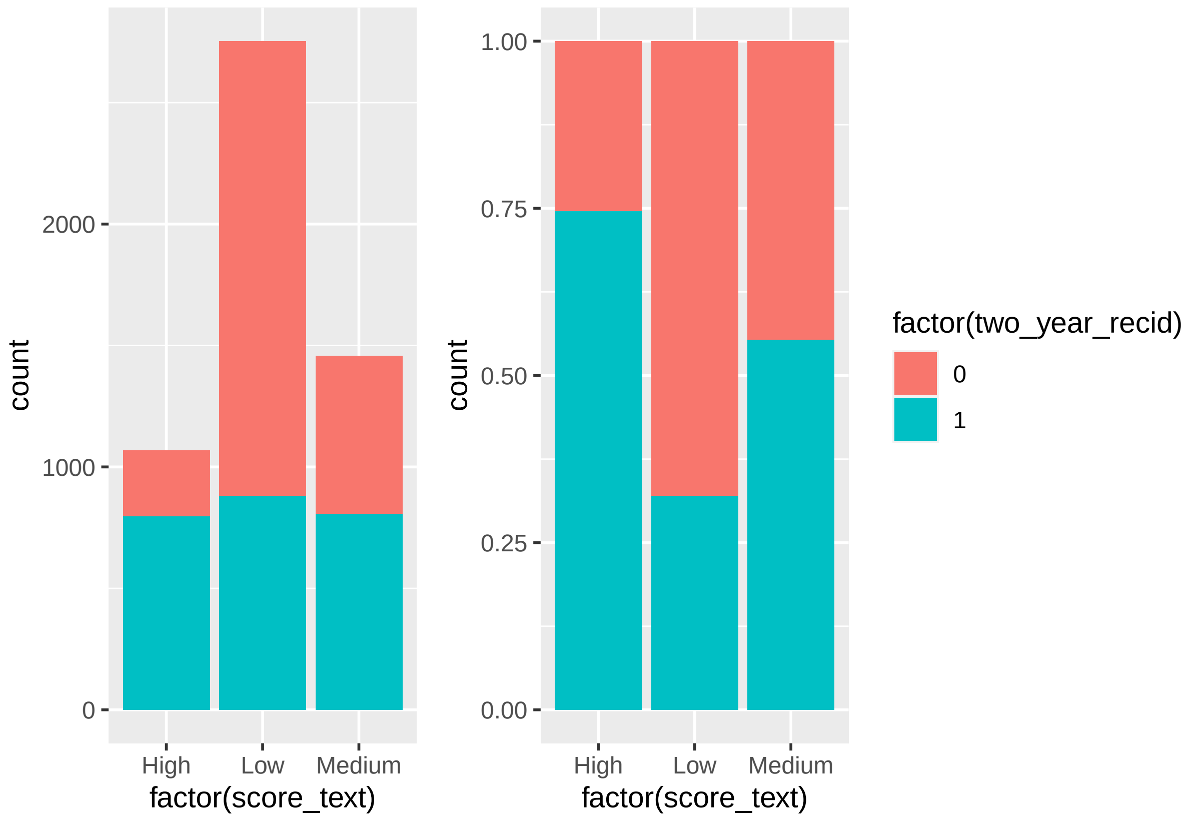
```
marital + marital_proportion
```



```
decile + decile_proportion
```

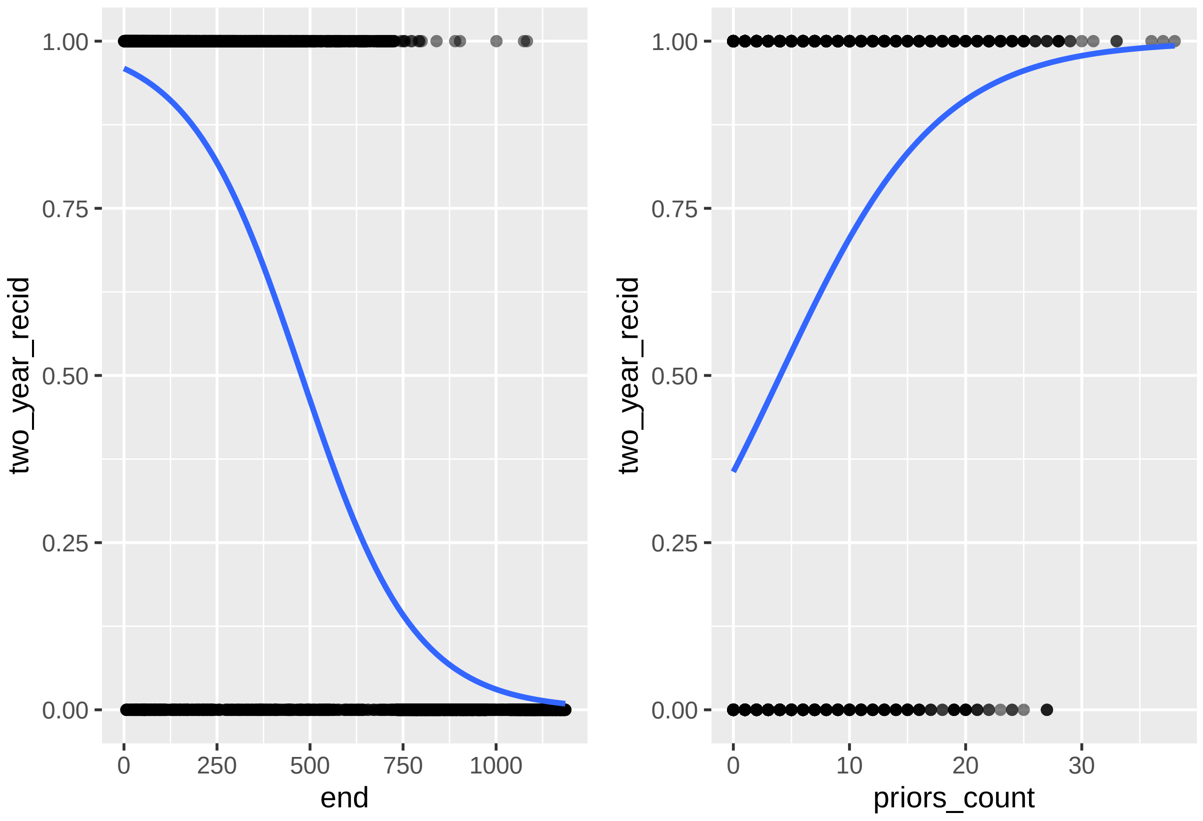


```
decile_text + decile_text_proportion
```



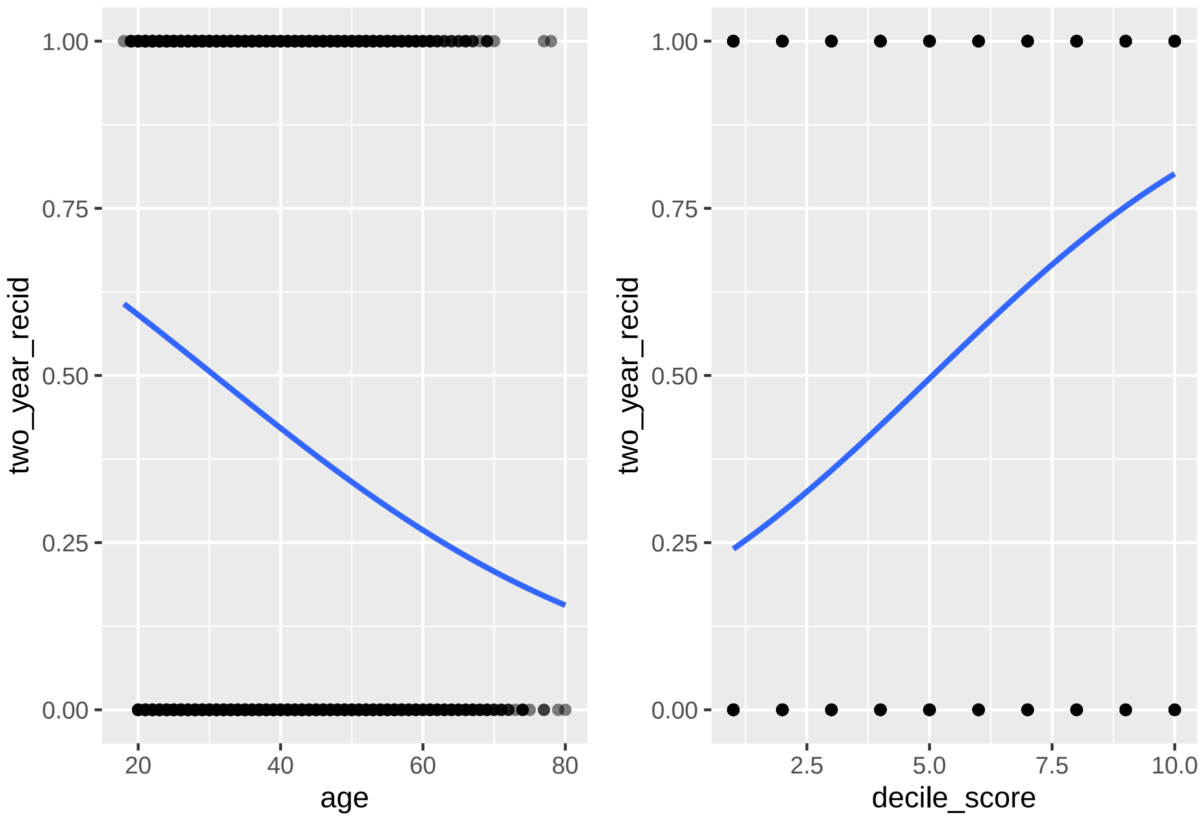
```
#showing overall impact of age, time out of jail, and likelihood of recividism
model_end <- ggplot(df, aes(x=end, y=two_year_recid)) +
  geom_point(alpha=.5) +
  stat_smooth(method="glm", se=FALSE, method.args = list(family=binomial))

model_priors <- ggplot(df, aes(x=priors_count, y=two_year_recid)) +
  geom_point(alpha=.5) +
  stat_smooth(method="glm", se=FALSE, method.args = list(family=binomial))

model_age <- ggplot(df, aes(x=age, y=two_year_recid)) +
  geom_point(alpha=.5) +
  stat_smooth(method="glm", se=FALSE, method.args = list(family=binomial))

model_decile <- ggplot(df, aes(x=decile_score, y=two_year_recid)) +
  geom_point(alpha=.5) +
  stat_smooth(method="glm", se=FALSE, method.args = list(family=binomial))

model_end + model_priors
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



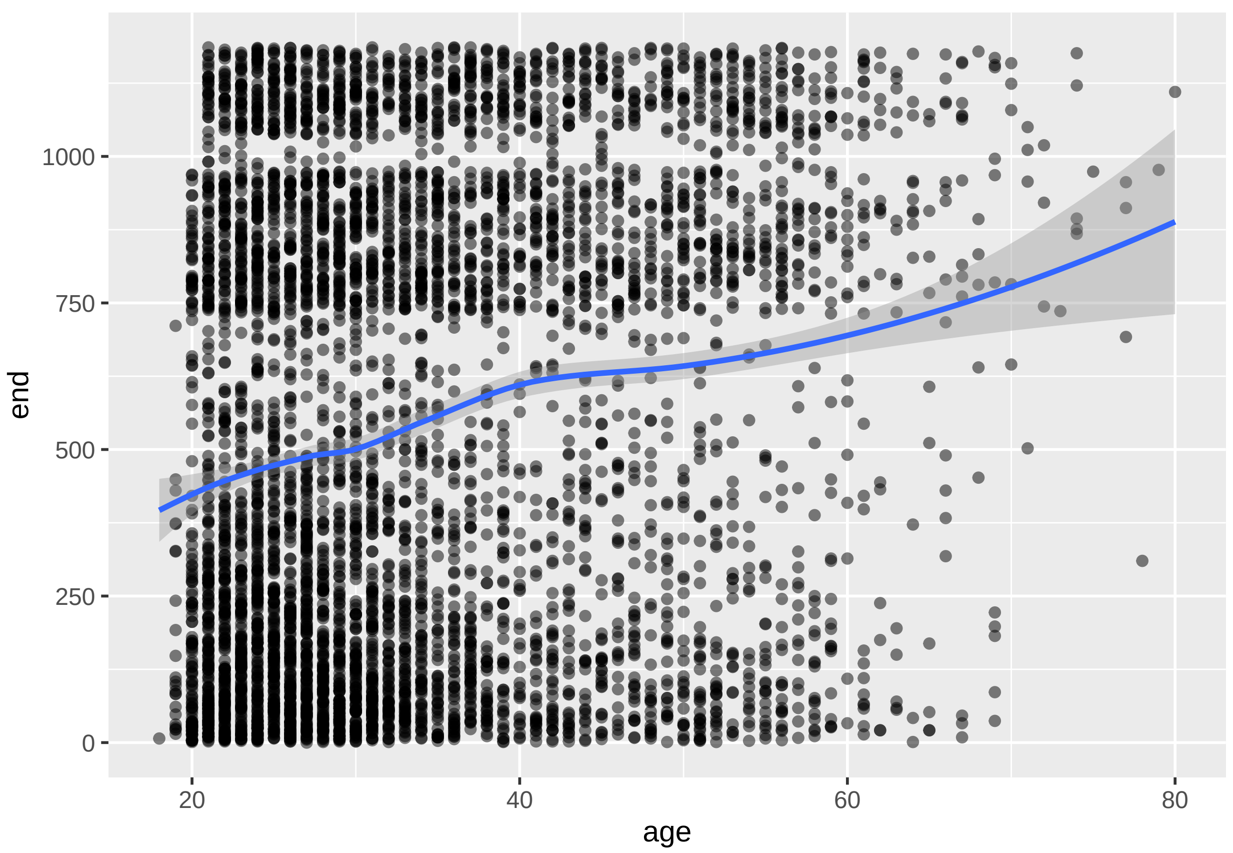
```
model_age + model_decile
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



```r
#showing the relation between age and staying out of jail
model_end_age <- ggplot(df, aes(x=age, y=end)) +
  geom_point(alpha=.5) +
  stat_smooth(method="loess", se=TRUE)
model_end_age
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



My final graph shows the relationship between age and staying out of jail. As expected there is a positive upwards relation, the older someone is, the less likely they get put back in jail. This linear model was created due to the output from prior logit models (age, time out of jail, priors, decile score).

I also ran the marital status and made it proportional. My expectation that married individuals would have the lowest rate of recidivism, but they were 3rd lowest behind widows and divorcees. The divorcee surprised me.

The other interesting graphical output was the age breakdown by race. The black concentration is closer to 25, while the white concentration is closer to 35. Further exploration showing the recidivism based upon the “age” and “end” variables should incorporate race since the distribution is quite different.

4

Create a training and test set from your original data. (You can decide to use cross-validation for model creation and assessment if you like, but it's not required for this assignment.)

5

Using your training set, create at least three predictive models that attempt to predict two_year_recid using some or all of your chosen predictor variables.

A) For each model, include a paragraph or so that explains why you chose that particular model, or those particular variables, and any other decisions you made that might be relevant. Ideally, your explanation might also describe how you hope the next model will improve on the previous model. You might consider overall choice of model (logistic regression, SVM, random forest, etc.) as well as choices made to balance classes or categories.

B) Write the code to make the model and to assess its overall accuracy and its sensitivity and specificity. You could write your own code, pull the numbers out of the confusionMatrix command output, or find other packages if need be. (Note: It should be relatively easy to match the accuracy of the Compas algorithm. We might take a minute to ponder what that means. It may be harder to get unbiased results for the two racial groups.)

```
#4
#finding NA values
colSums(is.na(df)) %>% as.data.frame()
```

```
##                           .
## sex                       0
## age                       0
## age_cat                   0
## race                      0
## juv_fel_count             0
## juv_misd_count            0
## juv_other_count           0
## priors_count              0
## marital_status            0
## c_days_from_compas        0
## c_charge_degree           0
## days_b_screening_arrest 0
## decile_score              0
## score_text                0
## v_decile_score            0
## v_score_text              0
## start                     0
## end                       0
## two_year_recid            0
```

```
df <- subset(df, select = -c(end)) #removing due to high correlation

#creating training/test datasets
set.seed(1234)
```

```r
split <- sample(1:nrow(df), 0.75*floor(nrow(df)))
split.train <- df[split, ]
split.test <- df[-split, ]

split.train.numeric <- split.train %>%
  mutate(two_year_recid = as.numeric(two_year_recid))
split.test.numeric <- split.train %>%
  mutate(two_year_recid = as.numeric(two_year_recid))

table(split.train$race) %>%
  kable()
```

Var1	Freq
African-American	2366
Caucasian	1592

```r
fit.stats <- function(fit, test){
  test$res <- test$two_year_recid - predict(fit, newdata = test)
  rbind(test %>% group_by(race) %>%
          summarize(Bias = -mean(res),
                    RMSE = sqrt(mean(res^2))),
        test %>%
          summarize(race = 'All',
                    Bias = -mean(res),
                    RMSE = sqrt(mean(res^2))))
}

# Log Loss as defined on the kaggle forum
LogLoss<-function(act, pred){
 eps = 1e-15;
 nr = length(pred)
 pred = matrix(sapply( pred, function(x) max(eps,x)), nrow = nr)
 pred = matrix(sapply( pred, function(x) min(1-eps,x)), nrow = nr)
 ll = sum(act*log(pred) + (1-act)*log(1-pred))
 ll = ll * -1/(length(act))
 return(ll);
 }
```

```r
#5 - model 1 - steppwise regression
#Entire model
full.model <- lm(as.numeric(two_year_recid) ~., data = split.train)
summary(full.model)
```

```
##
## Call:
## lm(formula = as.numeric(two_year_recid) ~ ., data = split.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0421 -0.3817 -0.1499  0.4289  1.1398
##
```

```
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        4.386e-01  1.246e-01   3.520 0.000436 ***
## sexMale                            7.290e-02  1.865e-02   3.909 9.41e-05 ***
## age                               -6.491e-03  1.543e-03  -4.207 2.64e-05 ***
## age_catGreater than 45             6.968e-02  3.484e-02   2.000 0.045597 *
## age_catLess than 25                1.302e-02  2.526e-02   0.516 0.606182
## raceCaucasian                      1.448e-02  1.573e-02   0.921 0.357255
## juv_fel_count                     -9.789e-03  1.861e-02  -0.526 0.598820
## juv_misd_count                    -2.666e-02  1.534e-02  -1.738 0.082281 .
## juv_other_count                    4.513e-02  1.574e-02   2.867 0.004170 **
## priors_count                       2.023e-02  1.944e-03  10.408  < 2e-16 ***
## marital_statusMarried             -3.491e-02  4.161e-02  -0.839 0.401471
## marital_statusSeparated            4.852e-02  6.573e-02   0.738 0.460451
## marital_statusSignificant Other    6.465e-02  5.340e-02   1.211 0.226123
## marital_statusSingle              -1.001e-02  3.709e-02  -0.270 0.787234
## marital_statusUnknown              1.303e-01  1.132e-01   1.151 0.249928
## marital_statusWidowed              1.343e-01  1.156e-01   1.162 0.245199
## c_days_from_compas                -4.243e-05  2.750e-05  -1.543 0.122886
## c_charge_degreeM                  -3.654e-02  1.569e-02  -2.329 0.019908 *
## days_b_screening_arrest            3.637e-03  1.442e-03   2.521 0.011726 *
## decile_score                       3.699e-02  8.670e-03   4.267 2.03e-05 ***
## score_textLow                     -1.507e-02  5.388e-02  -0.280 0.779767
## score_textMedium                  -9.541e-03  3.202e-02  -0.298 0.765755
## v_decile_score                    -1.536e-03  9.748e-03  -0.158 0.874819
## v_score_textLow                   -3.354e-02  5.812e-02  -0.577 0.563838
## v_score_textMedium                -1.927e-02  3.731e-02  -0.516 0.605583
## start                             -4.411e-05  1.412e-04  -0.312 0.754763
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4533 on 3932 degrees of freedom
## Multiple R-squared:  0.1801, Adjusted R-squared:  0.1749
## F-statistic: 34.55 on 25 and 3932 DF,  p-value: < 2.2e-16
```

```
stepAIC <- MASS::stepAIC
step.model <- stepAIC(full.model, direction = "both", trace = FALSE)
summary(step.model)
```

```
##
## Call:
## lm(formula = as.numeric(two_year_recid) ~ sex + age + age_cat +
##     juv_misd_count + juv_other_count + priors_count + c_days_from_compas +
##     c_charge_degree + days_b_screening_arrest + decile_score,
##     data = split.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0349 -0.3829 -0.1504  0.4269  1.1109
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                3.768e-01  5.434e-02   6.934 4.75e-12 ***
## sexMale                    7.090e-02  1.832e-02   3.870 0.000111 ***
## age                      -6.337e-03  1.431e-03  -4.427 9.81e-06 ***
## age_catGreater than 45    7.240e-02  3.467e-02   2.088 0.036842 *
## age_catLess than 25       1.711e-02  2.356e-02   0.726 0.467913
## juv_misd_count           -2.589e-02  1.526e-02  -1.697 0.089845 .
## juv_other_count           4.643e-02  1.570e-02   2.958 0.003111 **
## priors_count              1.983e-02  1.899e-03  10.443  < 2e-16 ***
## c_days_from_compas       -4.526e-05  2.741e-05  -1.652 0.098705 .
## c_charge_degreeM         -3.542e-02  1.556e-02  -2.276 0.022890 *
## days_b_screening_arrest   3.691e-03  1.436e-03   2.569 0.010221 *
## decile_score              3.958e-02  3.353e-03  11.804  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4531 on 3946 degrees of freedom
## Multiple R-squared:  0.1778, Adjusted R-squared:  0.1755
## F-statistic: 77.59 on 11 and 3946 DF,  p-value: < 2.2e-16
```

```
#creating confusion matrix
split.test$prob <- predict(step.model, split.test, type = "response")
split.test$recid <- ifelse(split.test$prob>=0.5, 1, 0)


library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.1.3

## Loading required package: survival

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following object is masked from 'package:psych':
##
##     describe

## The following object is masked from 'package:parsnip':
##
##     translate

## The following objects are masked from 'package:dplyr':
##
##     src, summarize
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

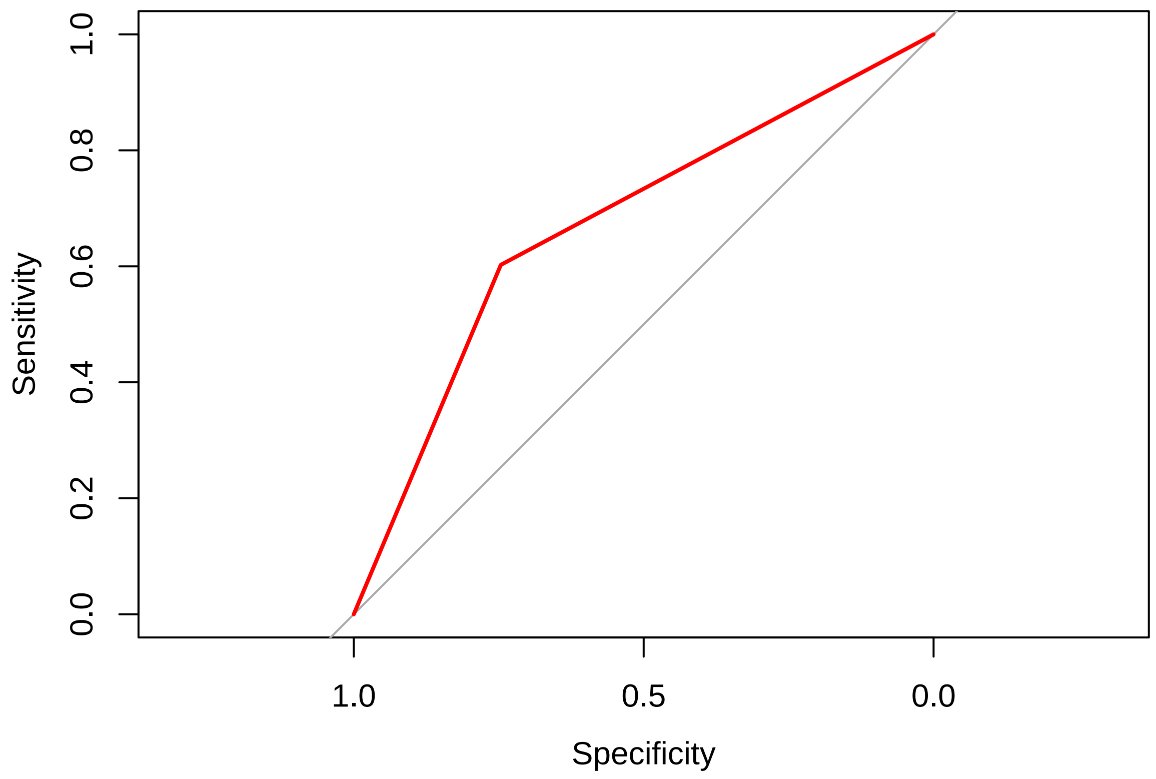
```
describe(split.test$race)
```

```
## split.test$race
##        n  missing distinct
##     1320        0        2
##
## Value      African-American        Caucasian
## Frequency               809              511
## Proportion            0.613            0.387
```

```
#list of races, 1,2,3
#dataframe

tab.step <- table(group=split.test$race, split.test$recid, split.test$two_year_recid)
tab.step2 <- table(split.test$recid, split.test$two_year_recid)

#plotting roc
plot(roc(split.test$two_year_recid, split.test$recid), col="red")
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

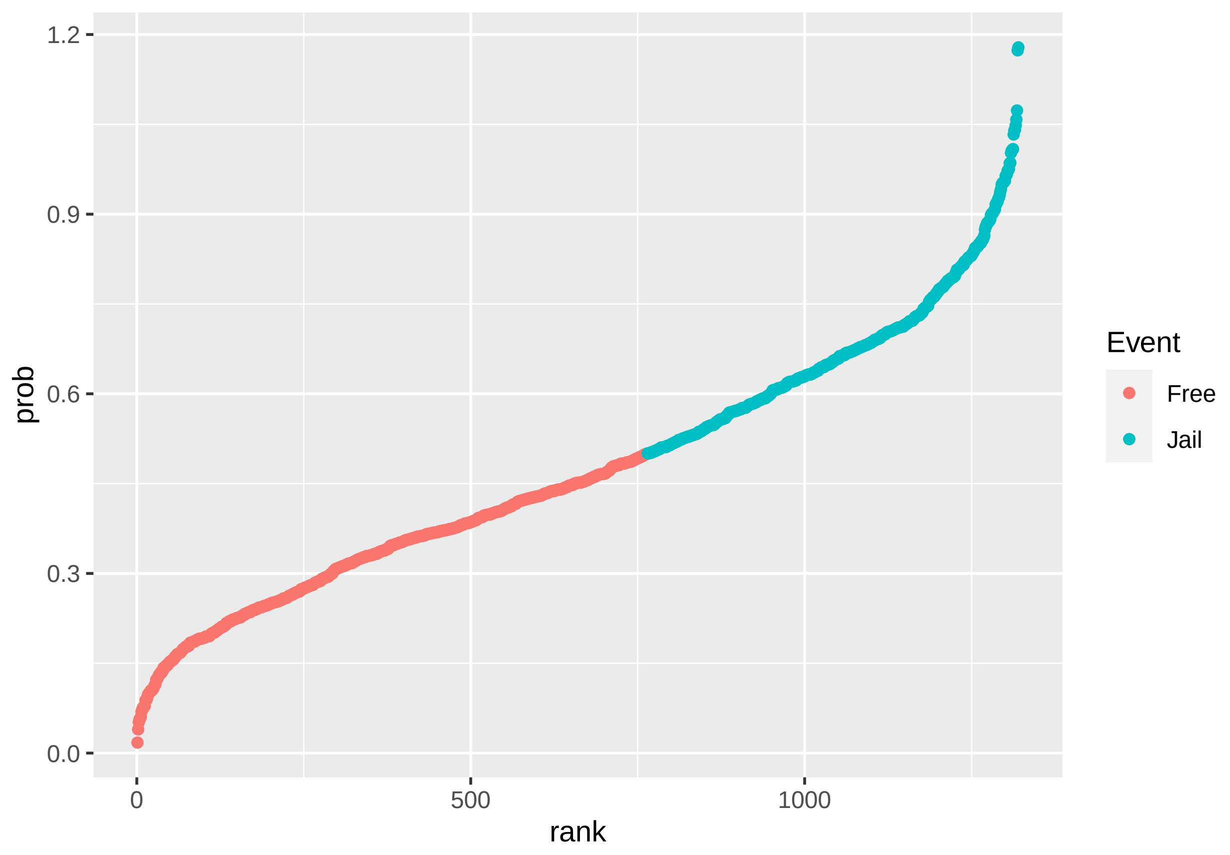


```r
auc(roc(split.test$two_year_recid, split.test$recid), col="red")
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## Area under the curve: 0.6744
```

```r
#test dataset, create new column,

#graphing output
p <- split.test %>%
  arrange(prob) %>%
  mutate(rank = rank(prob),
         Event = ifelse(prob >= 0.5, 'Jail', 'Free')) %>%
  ggplot(aes(rank, prob)) +
  geom_point(aes(color = Event)) +
  geom_smooth(method = 'glm',
              method.args = list(family = 'binomial'))

p
```

```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Computation failed in `stat_smooth()`
## Caused by error:
## ! y values must be 0 <= y <= 1
```



```
#using fit.stats function
#fit.stats(step.model, split.test)

#specificity and other stats
cm.step <- tab.step

accuracy.step <- sum(cm.step[1], cm.step[4]) / sum(cm.step[1:4])
precision.step <- cm.step[4] / sum(cm.step[4], cm.step[2])
sensitivity.step <- cm.step[4] / sum(cm.step[4], cm.step[3])
fscore.step <- (2 * (sensitivity.step * precision.step))/(sensitivity.step + precision.step)
specificity.step <- cm.step[1] / sum(cm.step[1], cm.step[2])
```

I ran the stepwise model first to determine what variables were important.

```r
#5 - model 2 - naive bayes (https://www.r-bloggers.com/2021/04/naive-bayes-classification-in-r/)
bayes.model <- naive_bayes(as.factor(two_year_recid) ~ ., data = split.train, usekernel = T)


full.model <- lm(as.numeric(two_year_recid) ~., data = split.train)

#plot(bayes_model)

p <- predict(bayes.model, split.train, type="prob")
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```r
#confusion matrix train data
p1 <- predict(bayes.model, split.train)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```r
tab1.bayes <- table(p1, split.train$two_year_recid)

#confusion matrix test data
p2 <- predict(bayes.model, split.test)
```

```
## Warning: predict.naive_bayes(): more features in the newdata are provided as
## there are probability tables in the object. Calculation is performed based on
## features to be found in the tables.
```

```r
tab.bayes <- table(p2, split.test$two_year_recid)
tab.bayes
```

```
##
## p2    0   1
##   0 596 392
##   1  90 242
```

```r
1 - sum(diag(tab.bayes)) / sum(tab.bayes)
```

```
## [1] 0.3651515
```

```r
#just need to figure out how to group by race
#fit.stats(bayes_model, split.test)

#specificity and other stats
cm.bayes <- tab.bayes
```

```r
accuracy.bayes <- sum(cm.bayes[1], cm.bayes[4]) / sum(cm.bayes[1:4])
precision.bayes <- cm.bayes[4] / sum(cm.bayes[4], cm.bayes[2])
sensitivity.bayes <- cm.bayes[4] / sum(cm.bayes[4], cm.bayes[3])
fscore.bayes <- (2 * (sensitivity.bayes * precision.bayes))/(sensitivity.bayes + precision.bayes)
specificity.bayes <- cm.bayes[1] / sum(cm.bayes)
specificity.bayes
```

```
## [1] 0.4515152
```

```
#5 - model 3 - linear discriminant analysis https://www.r-bloggers.com/2021/05/linear-discriminant-analys

lda <- MASS::lda
#install.packages("klaR")
library(klaR)
```

```
## Warning: package 'klaR' was built under R version 4.1.3

## Loading required package: MASS

## Warning: package 'MASS' was built under R version 4.1.3

##
## Attaching package: 'MASS'

## The following object is masked from 'package:patchwork':
##
##     area

## The following object is masked from 'package:dplyr':
##
##     select
```

```
lda.model <- lda(two_year_recid ~., split.train)
lda.predict <- predict(lda.model, split.train)
#partimat(two_year_recid ~., data = split.train, method = 'lda')

lda.predict.class <- predict(lda.model, split.test.numeric)$class
tab.lda <- table(lda.predict.class, split.test.numeric$two_year_recid)
tab.lda
```

```
##
## lda.predict.class    0    1
##                 0 1604  740
##                 1  505 1109
```

```
#fit.stats(lda.predict, split.test)

#specificity and other stats
cm.lda <- tab.lda

accuracy.lda <- sum(cm.lda[1], cm.lda[4]) / sum(cm.lda[1:4])
precision.lda <- cm.lda[4] / sum(cm.lda[4], cm.lda[2])
sensitivity.lda <- cm.lda[4] / sum(cm.lda[4], cm.lda[3])
fscore.lda <- (2 * (sensitivity.lda * precision.lda))/(sensitivity.lda + precision.lda)
specificity.lda <- cm.lda[1] / sum(cm.lda)
specificity.lda
```

```
## [1] 0.4052552
```

```r
#running logistic regression
logit <- glm(two_year_recid ~ ., data = split.train, family = "binomial")
```

```r
#prepping for table output
race.list <- list("African-American" = "African-American",
                  "Caucasian" = "Caucasian",
                  "All" = c("African-American", "Caucasian"))

detach(package:Hmisc)

display.stats <- function(test){
map_dfr(race.list, ~ test %>% filter(race %in% .x) %>%
          summarize(Accuracy = mean(two_year_recid==predict),
                    Sensitivity = sum(two_year_recid=="TRUE" & predict=="TRUE")/sum(two_year_recid=="TRU
                    Specificity = sum(two_year_recid=="FALSE" & predict=="FALSE")/sum(two_year_recid=="F
        .id="Race") %>%
  kable()
}


step.results <- split.test %>%
  mutate(predict = factor(predict(step.model, newdata=split.test, type="response") > .50)) %>%
  display.stats()

bayes.results <- split.test %>%
  mutate(predict = (predict(bayes.model, newdata=split.test, type="prob") > .50)) %>%
  display.stats()
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `predict = (...)`.
## Caused by warning:
## ! predict.naive_bayes(): more features in the newdata are provided as there are probability tables i
```

```r
logit.results <- split.test %>%
  mutate(predict = (predict(logit, newdata=split.test, type="response") > .50)) %>%
  display.stats()
```

6

Finally, summarize the results of your three models in an easy-to-read table, and comment on what you found as a result of this process.

```
step.results
```

Race	Accuracy	Sensitivity	Specificity
African-American	0	NaN	NaN
Caucasian	0	NaN	NaN
All	0	NaN	NaN

```
bayes.results
```

Race	Accuracy	Sensitivity	Specificity
African-American	0.5	NaN	NaN
Caucasian	0.5	NaN	NaN
All	0.5	NaN	NaN

```
#lda.results
logit.results
```

Race	Accuracy	Sensitivity	Specificity
African-American	0.6736712	NaN	NaN
Caucasian	0.6908023	NaN	NaN
All	0.6803030	NaN	NaN

A higher specific test means that there are few false positive results. In this model, a positive result is a prediction of recidivism. Our goal is to have a highly accurate model that doesn't discriminate. It is the opinion of author that it would be better to minimize the false positive rate at the expense of reducing accuracy or increasing false negative rates (falsely predicting the person will not go back to jail).

The damage done by predicting someone will go back to jail who does not go back to jail exceeds the damage done by predicting someone will remain out of jail, when in fact they do return.

I selected four models to evaluate the data. The first model was stepwise regression. The stepwise model had an 82% specificity overall, but only 74% specificity for blacks compared to the 91% specificity for whites. A 17 point difference in accuracy based on race is not ideal

The second model used was a naive bayes model. For some reason the display.stats function indicates a perfect specificity rate. However if calculated independently of the function, the overall specificity was 45%. This should be looked into further.

The third model was a linear-discriminant model. LDA is similar to naive Bayes, except it assumes the distribution is Guassian and similar in all classes. I could not find a way to output any data with the lda using the display.stats function. My multiple attempts are at the end of this page. Similar to the naive bayes model, the specificity was a paltry 44%.

The fourth and final model calculated was the simple multi-variate logistic regression. Like the stepwise model, it's specificity towards blacks was 74%. It performed worse than the stepwise model for whites, reducing the specificity from 91% to 89%. The overall accuracy of the logistic regression was slightly better than the stepwise model.

A fifth model considered was the KNN model. Since we are only classifying two groups, KNN would work as an unsupervised learning algorithm since it has the restriction of n groups <= 5. Prior to the office hours meeting, I was unable to get it to work.

In the interest of time, there are many flaws with the analysis. Removal of race as a predictor variable should be considered. Cross validation should be implemented. Better pre-processing of the data should occur. Further exploration of the models and thoughts on how to build upon them should also be evaluated. One such method could be gradient boosting the logistic regression.

```
#trying to get display.stats to work with the lda model
#issue: x unimplemented type 'list' in 'orderVector1'
#issue: no applicable method for 'mutate' applied to an object of class "c('double', 'numeric')"

split.test2 <- as.data.frame(lapply(split.test, unlist))

lda.results <- split.test2 %>%
  ungroup() %>%
  mutate(predict = factor(predict(lda.model, newdata=split.test2))) %>%
  display.stats()

lda.results0 <- split.test %>%
  mutate(predict = (predict(lda.model, newdata=split.test))) %>%
  display.stats()

lda.results1 <- split.test2 %>%
  unlist() %>%
  mutate(predict = (predict(lda.model, newdata=split.test))) %>%
  display.stats()

lda.results2 <- split.test %>%
  ungroup() %>%
  mutate(predict = (predict(lda.model, newdata=split.test))) %>%
  display.stats()

lda.results3 <- split.test %>%
  mutate(predict = (predict(lda.model, newdata=split.test))) %>%
  ungroup() %>%
  display.stats()

lda.results4 <- split.test %>%
  mutate(predict = (predict(lda.model, newdata=split.test, type="prob") > .50)) %>%
  ungroup() %>%
  display.stats()

lda.results5 <- split.test %>%
  mutate(predict = (predict(lda.model, newdata=split.test, type="response") > .50)) %>%
  ungroup() %>%
  display.stats()

lda.results6 <- split.test %>%
  ungroup() %>%
  mutate(predict = (predict(lda.model, newdata=split.test.numeric, type="response") > .50)) %>%
  display.stats()

lda.results7 <- split.test %>%
  unlist() %>%
  mutate(predict = (predict(lda.model, newdata=split.test, type="prob") > .50)) %>%
  display.stats()

lda.results8 <- split.test %>%
  unlist() %>%
  mutate(predict = factor(predict(lda.model, newdata=split.test, type="prob") > .50)) %>%
display.stats()
```