HW3

```r
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 0.2.0 --

## v broom        0.7.12     v recipes      0.2.0
## v dials        0.1.0      v rsample      0.1.1
## v dplyr        1.0.8      v tibble       3.1.6
## v ggplot2      3.3.5      v tidyr        1.2.0
## v infer        1.0.0      v tune         0.2.0
## v modeldata    0.1.1      v workflows    0.2.6
## v parsnip      0.2.1      v workflowsets 0.2.1
## v purrr        0.3.4      v yardstick    0.0.9

## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Search for functions across packages at https://www.tidymodels.org/find/
```

```r
library(ISLR)
library(ISLR2)
```

```
##
## Attaching package: 'ISLR2'

## The following objects are masked from 'package:ISLR':
##
##     Auto, Credit
```

```r
library(discrim)
```

```
##
## Attaching package: 'discrim'

## The following object is masked from 'package:dials':
##
##     smoothness
```

```r
library(poissonreg)
library(corrr)
library(klaR)
```

```
## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:ISLR2':
##
##     Boston

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

```
library(dplyr)
library(MASS)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
tidymodels_prefer()

library(readr)
titanic <- read_csv("~/Downloads/homework-3/data/titanic.csv")
```

```
## Rows: 891 Columns: 12

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): survived, name, sex, ticket, cabin, embarked
## dbl (6): passenger_id, pclass, age, sib_sp, parch, fare
##
## 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.
```

```
View(titanic)
```

Question 1

```r
titanic$survived <- factor(titanic$survived, levels = c("Yes","No"))
titanic$pclass <- as.factor(titanic$pclass)

set.seed(891)

titanic_split <- initial_split(titanic, prop = 0.8, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
```

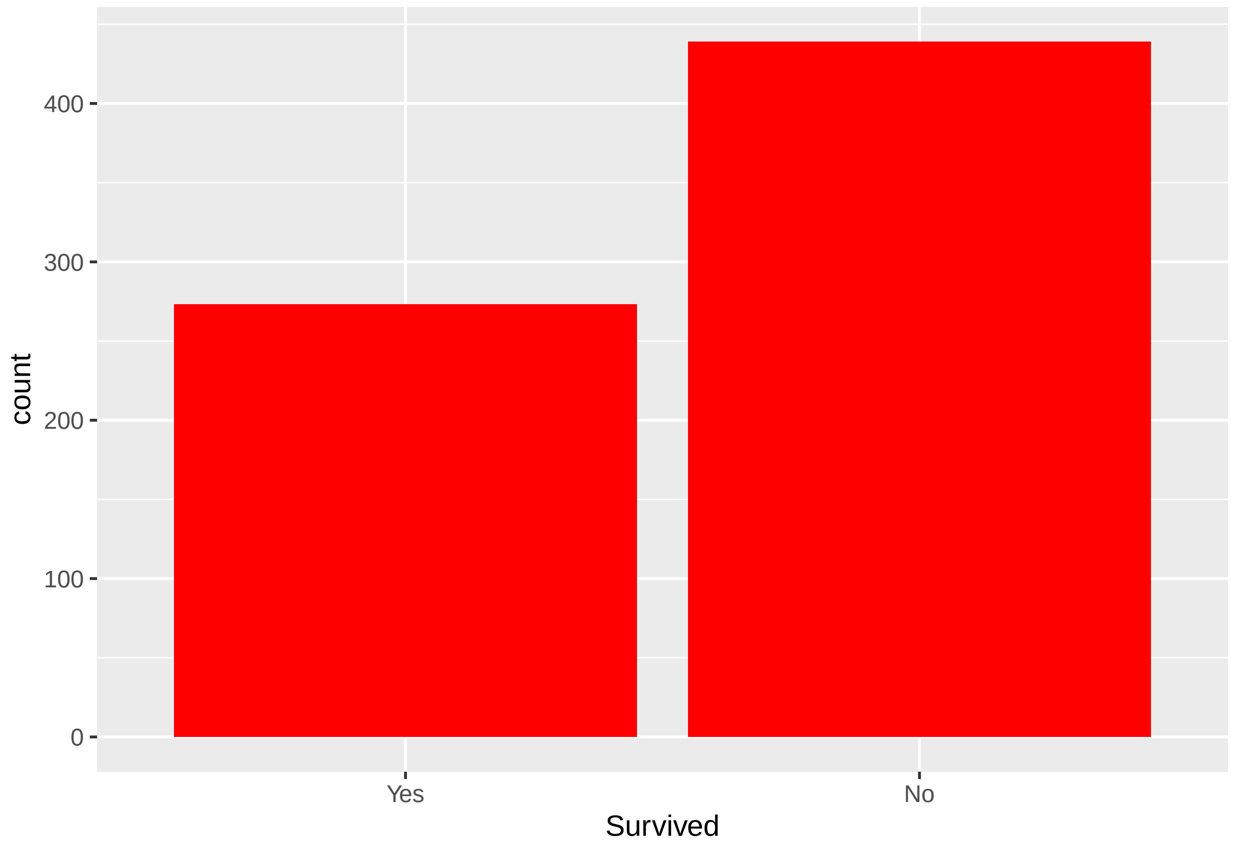
Training set observations: 712 Testing set observations: 179

Some variables have missing data, which is why we will used stratified sampling.

It is a good idea to use stratified sampling for this data because it provides better coverage of the population as the researchers have control over the subgroups to ensure all of them are represented in the sampling.

Question 2

```r
library(ggplot2)
ggplot(titanic_train, aes(x=survived)) +
  geom_bar(fill='red') +  labs(x='Survived')
```



The variable survived has two levels: Yes and No. From the distribution of the training data set, we can see that more individuals died than survived. The count of those who survived is a little less than 300, while the count for those who passed is greater than 400.

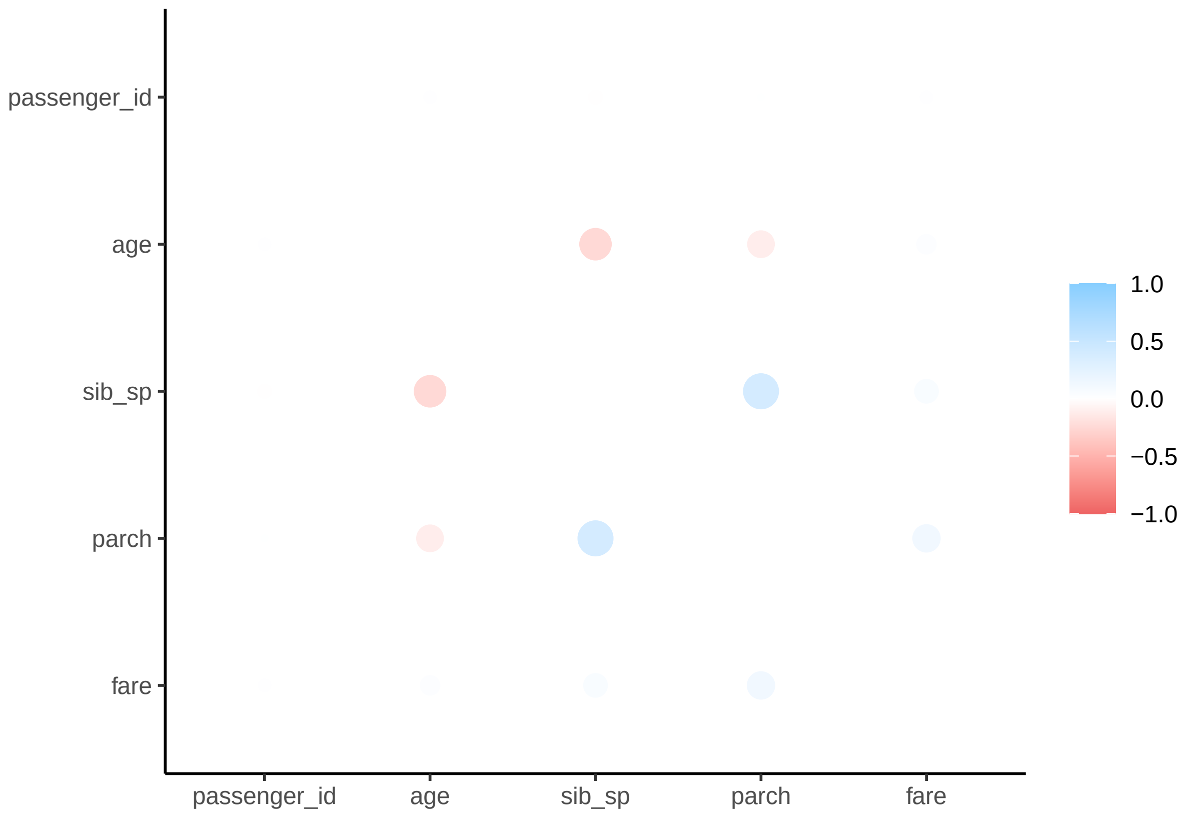
Question 3

```
cor_titanic <- titanic_train %>%
  select(passenger_id,age,sib_sp,parch,fare) %>%
  correlate()
```

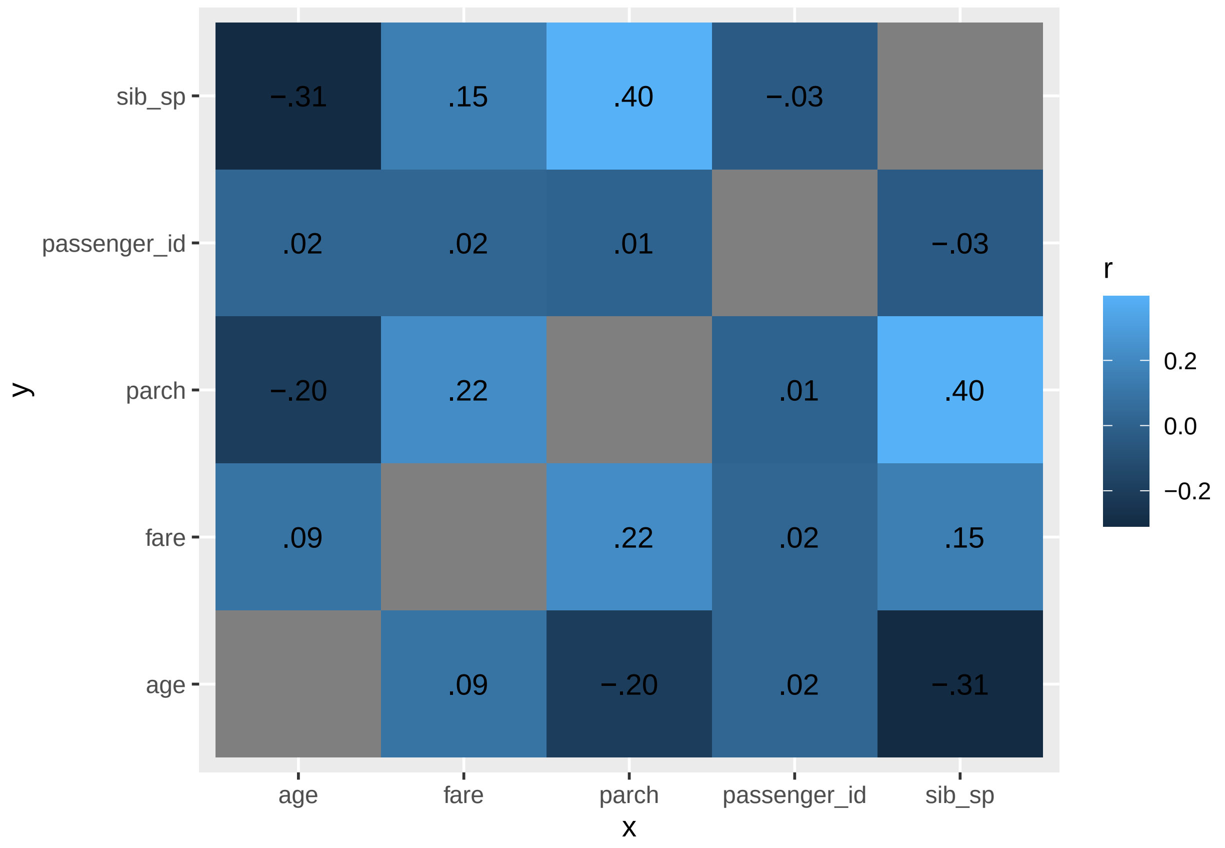
```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```

```
rplot(cor_titanic)
```

```
## Don't know how to automatically pick scale for object of type noquote. Defaulting to continuous.
```



```
cor_titanic %>%
  stretch() %>%
  ggplot(aes(x, y, fill = r)) +
  geom_tile() +
  geom_text(aes(label = as.character(fashion(r))))
```



From the two matrices, we can see that only sib_sp and parch have much of any correlation to one another in a positive direction, and it's only about .40.

Question 4

```
titanic_recipe <- recipe(survived ~ pclass + sex + age + sib_sp + parch + fare, data = titanic_train) %
  step_impute_linear(age) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(~starts_with("sex"):fare) %>%
  step_interact(~age:fare) %>%
  step_normalize(all_predictors())
```

Question 5

```
log_reg <- logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

log_wkflow <- workflow() %>%
  add_model(log_reg) %>%
  add_recipe(titanic_recipe)
```

```r
log_fit <- fit(log_wkflow, titanic_train)

log_fit %>%
  tidy()
```

```
## # A tibble: 10 x 5
##    term            estimate std.error statistic  p.value
##    <chr>              <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)        0.637     0.110     5.79  7.21e- 9
##  2 age                0.919     0.170     5.39  6.98e- 8
##  3 sib_sp             0.426     0.129     3.31  9.28e- 4
##  4 parch              0.124     0.110     1.13  2.60e- 1
##  5 fare               0.264     0.416     0.633 5.26e- 1
##  6 pclass_X2          0.484     0.139     3.47  5.14e- 4
##  7 pclass_X3          1.21      0.175     6.94  3.95e-12
##  8 sex_male           1.23      0.132     9.35  9.17e-21
##  9 sex_male_x_fare    0.202     0.232     0.872 3.83e- 1
## 10 age_x_fare        -0.667     0.346    -1.93  5.42e- 2
```

Question 6

```r
lda_mod <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

lda_wkflow <- workflow() %>%
  add_model(lda_mod) %>%
  add_recipe(titanic_recipe)

lda_fit <- fit(lda_wkflow, titanic_train)
```

Question 7

```r
qda_mod <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")

qda_wkflow <- workflow() %>%
  add_model(qda_mod) %>%
  add_recipe(titanic_recipe)

qda_fit <- fit(qda_wkflow, titanic_train)
```

Question 8

```
nb_mod <- naive_Bayes() %>%
  set_mode("classification") %>%
  set_engine("klaR") %>%
  set_args(usekernel = FALSE)

nb_wkflow <- workflow() %>%
  add_model(nb_mod) %>%
  add_recipe(titanic_recipe)

nb_fit <- fit(nb_wkflow, titanic_train)
```

Question 9

```
#predictions
log_p <- predict(log_fit, new_data = titanic_train, type = "prob")
log_p
```

```
## # A tibble: 712 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1    0.0685    0.931
##  2    0.127     0.873
##  3    0.163     0.837
##  4    0.0188    0.981
##  5    0.798     0.202
##  6    0.485     0.515
##  7    0.210     0.790
##  8    0.0972    0.903
##  9    0.0973    0.903
## 10    0.395     0.605
## # ... with 702 more rows
```

```
lda_p <- predict(lda_fit, new_data = titanic_train, type = "prob")
lda_p
```

```
## # A tibble: 712 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1    0.0428    0.957
##  2    0.0797    0.920
##  3    0.101     0.899
##  4    0.0112    0.989
##  5    0.851     0.149
##  6    0.585     0.415
##  7    0.154     0.846
##  8    0.0601    0.940
##  9    0.0602    0.940
## 10    0.320     0.680
## # ... with 702 more rows
```

```
qda_p <- predict(qda_fit, new_data = titanic_train, type = "prob")
qda_p
```

```
## # A tibble: 712 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1  0.00375     0.996
##  2  0.000396    1.00
##  3  0.0102      0.990
##  4  0.000115    1.00
##  5  0.529       0.471
##  6  0.292       0.708
##  7  0.00847     0.992
##  8  0.00546     0.995
##  9  0.00549     0.995
## 10  0.150       0.850
## # ... with 702 more rows
```

```
nb_p <- predict(nb_fit, new_data = titanic_train, type = "prob")
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 60

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 61

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 90

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 101

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 112

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 166

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 168

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 180
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 190

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 194

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 215

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 259

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 266

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 276

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 296

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 311

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 321

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 362

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 384

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 397

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 412

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 422

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 436

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 469

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 493
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 504

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 510

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 525

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 529

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 546

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 557

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 577

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 646

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 651

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 670

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 672

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 682
```

```
nb_p
```

```
## # A tibble: 712 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1  0.0122      0.988
##  2  0.000743    0.999
##  3  0.0155      0.984
##  4  0.00223     0.998
##  5  0.436       0.564
##  6  0.353       0.647
##  7  0.118       0.882
##  8  0.0136      0.986
##  9  0.0134      0.987
## 10  0.0492      0.951
## # ... with 702 more rows
```

```r
#accuracies
log_reg_acc <- augment(log_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
log_reg_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.815
```

```r
lda_acc <- augment(lda_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
lda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.798
```

```r
qda_acc <- augment(qda_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
qda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.777
```

```r
nb_acc <- augment(nb_fit, new_data = titanic_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 60
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 61
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 90
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 101
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 112
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 166

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 168

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 180

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 190

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 194

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 215

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 259

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 266

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 276

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 296

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 311

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 321

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 362

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 384

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 397

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 412

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 422
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 436

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 469

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 493

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 504

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 510

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 525

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 529

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 546

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 557

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 577

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 646

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 651

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 670

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 672

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 682

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 4

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 60
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 61

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 90

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 101

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 112

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 166

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 168

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 180

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 190

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 194

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 215

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 259

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 266

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 276

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 296

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 311

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 321

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 362
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 384

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 397

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 412

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 422

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 436

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 469

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 493

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 504

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 510

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 525

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 529

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 546

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 557

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 577

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 646

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 651

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 670
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 672

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 682
```

```
nb_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.774
```

```
#through bind_col
titanic_train_res <- bind_cols(log_p, lda_p, qda_p, nb_p, titanic_train %>% select(survived))
```

```
## New names:
## * .pred_Yes -> .pred_Yes...1
## * .pred_No -> .pred_No...2
## * .pred_Yes -> .pred_Yes...3
## * .pred_No -> .pred_No...4
## * .pred_Yes -> .pred_Yes...5
## * ...
```

```
titanic_train_res %>%
  head()
```

```
## # A tibble: 6 x 9
##   .pred_Yes...1 .pred_No...2 .pred_Yes...3 .pred_No...4 .pred_Yes...5
##           <dbl>        <dbl>         <dbl>        <dbl>         <dbl>
## 1        0.0685        0.931        0.0428        0.957      0.00375
## 2        0.127         0.873        0.0797        0.920      0.000396
## 3        0.163         0.837        0.101         0.899      0.0102
## 4        0.0188        0.981        0.0112        0.989      0.000115
## 5        0.798         0.202        0.851         0.149      0.529
## 6        0.485         0.515        0.585         0.415      0.292
## # ... with 4 more variables: .pred_No...6 <dbl>, .pred_Yes...7 <dbl>,
## #   .pred_No...8 <dbl>, survived <fct>
```

```
#comparing model performance through accuracy test
accuracies <- c(log_reg_acc$.estimate, lda_acc$.estimate,
                nb_acc$.estimate, qda_acc$.estimate)
models <- c("Logistic Regression", "LDA", "Naive Bayes", "QDA")
results <- tibble(accuracies = accuracies, models = models)
results %>%
  arrange(-accuracies)
```

```
## # A tibble: 4 x 2
##   accuracies models
##        <dbl> <chr>
## 1      0.815 Logistic Regression
## 2      0.798 LDA
## 3      0.777 QDA
## 4      0.774 Naive Bayes
```

The model that achieved the highest accuracy was the Logistic Regression model.

Question 10

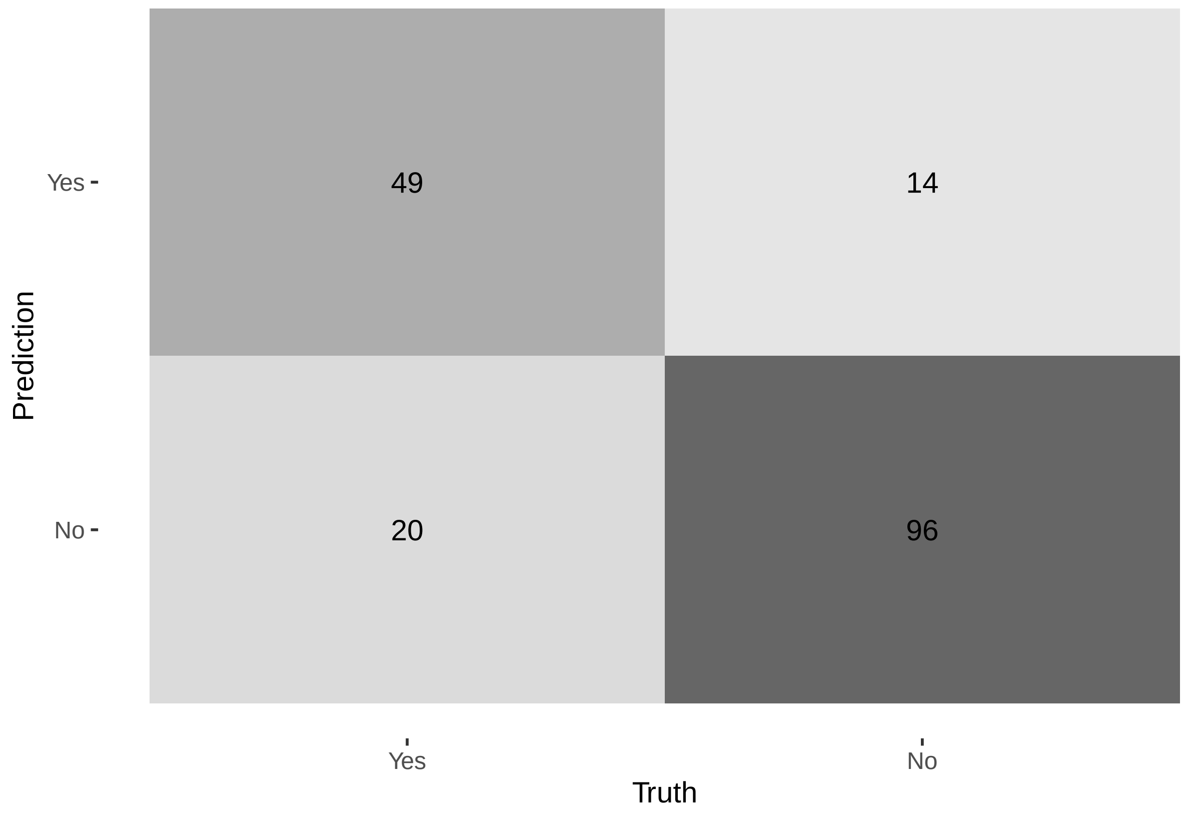
```r
predict(log_fit, new_data = titanic_test, type = "prob")
```

```
## # A tibble: 179 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1    0.104     0.896
##  2    0.645     0.355
##  3    0.0973    0.903
##  4    0.264     0.736
##  5    0.809     0.191
##  6    0.520     0.480
##  7    0.0830    0.917
##  8    0.230     0.770
##  9    0.599     0.401
## 10    0.240     0.760
## # ... with 169 more rows
```

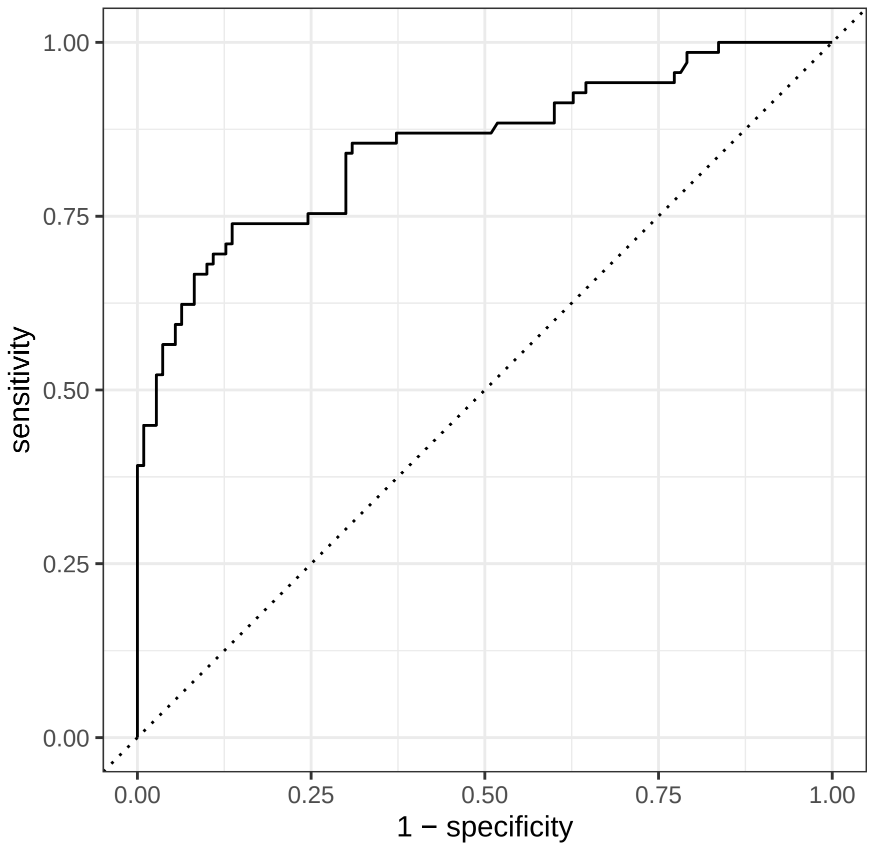
```r
multi_metric <- metric_set(accuracy, sensitivity, specificity)

augment(log_fit, new_data = titanic_test) %>%
  multi_metric(truth = survived, estimate = .pred_class)
```

```
## # A tibble: 3 x 3
##   .metric     .estimator .estimate
##   <chr>       <chr>          <dbl>
## 1 accuracy    binary         0.810
## 2 sensitivity binary         0.710
## 3 specificity binary         0.873
```

```r
augment(log_fit, new_data = titanic_test) %>%
  conf_mat(truth = survived, estimate = .pred_class) %>%
  autoplot(type = "heatmap")
```



```r
augment(log_fit, new_data = titanic_test) %>%
  roc_curve(survived, .pred_Yes) %>%
  autoplot()
```



```
augment(log_fit, new_data = titanic_test) %>%
  roc_auc(survived,.pred_Yes)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.851
```

In terms of model performance, the AUC from the ROC curve gives us a .85 estimate. This indicates that the performance essentially earned a “B” if we were to interpret it as a letter grade.

In comparison to the training data, the former had an accuracy about .814, while the testing data had an accuracy about .810. The two values are basically the same, but still slightly differ, and that is because the test accuracy should not be higher than the train as the model is optimized for the latter.