Homework 3

Yingshan Li (7937790)

April 19, 2022

Load the data

```
titanic <- read.csv(file = "titanic.csv" )
titanic1 <- titanic %>% mutate(survived = factor(survived, levels = c("Yes", "No"))) %>%
  mutate(pclass = factor(pclass))
```

Question 1

```
set.seed(2231)


titanic_split <- initial_split(titanic1, prop = 0.80, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
nrow(titanic_train)
```

```
## [1] 712
```

```
nrow(titanic_test)
```

```
## [1] 179
```

```
nrow(titanic1)
```

```
## [1] 891
```

```
712/891
```

```
## [1] 0.7991021
```

```
179/891
```

```
## [1] 0.2008979
```

There are approximately 80% of the observations in the training data set and 20% of the observations in the test data set, which correspond to the proportion we indicate in the initial_split() function.

check for missing data

```
table(is.na(titanic_train))
```

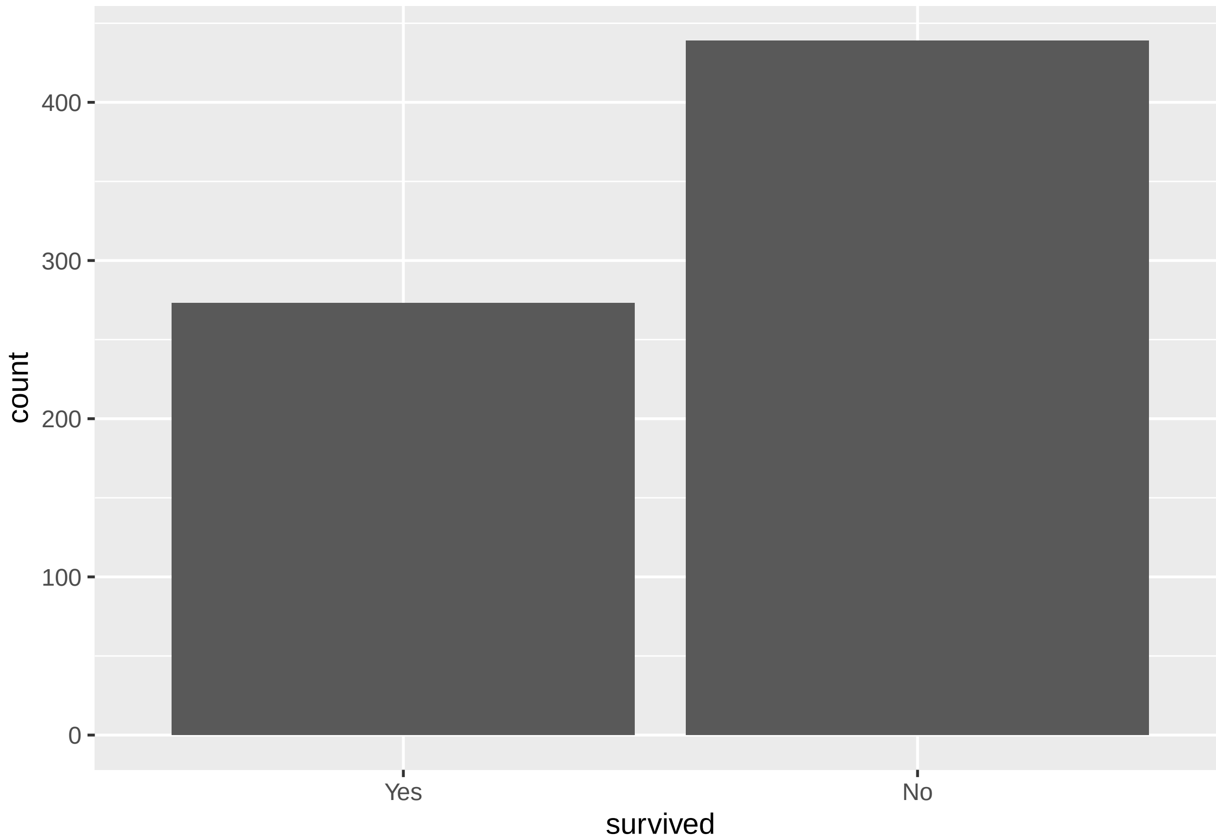
```
##
## FALSE  TRUE
##  7849   695
```

There are missing data in the data set, and most missing data are cabin and age.

Stratified sampling for this data make sure the distribution of survived or not survived is the same in both training and test data set.

Question 2

```
titanic_train %>%
  ggplot(aes(x = survived)) +
  geom_bar()
```



The number of not survived is obviously more than the number of survived, approximately a 40% - 60% split between Yes or No. Such difference is not significant to cause the problem of imbalance for our further analysis.

Question 3

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

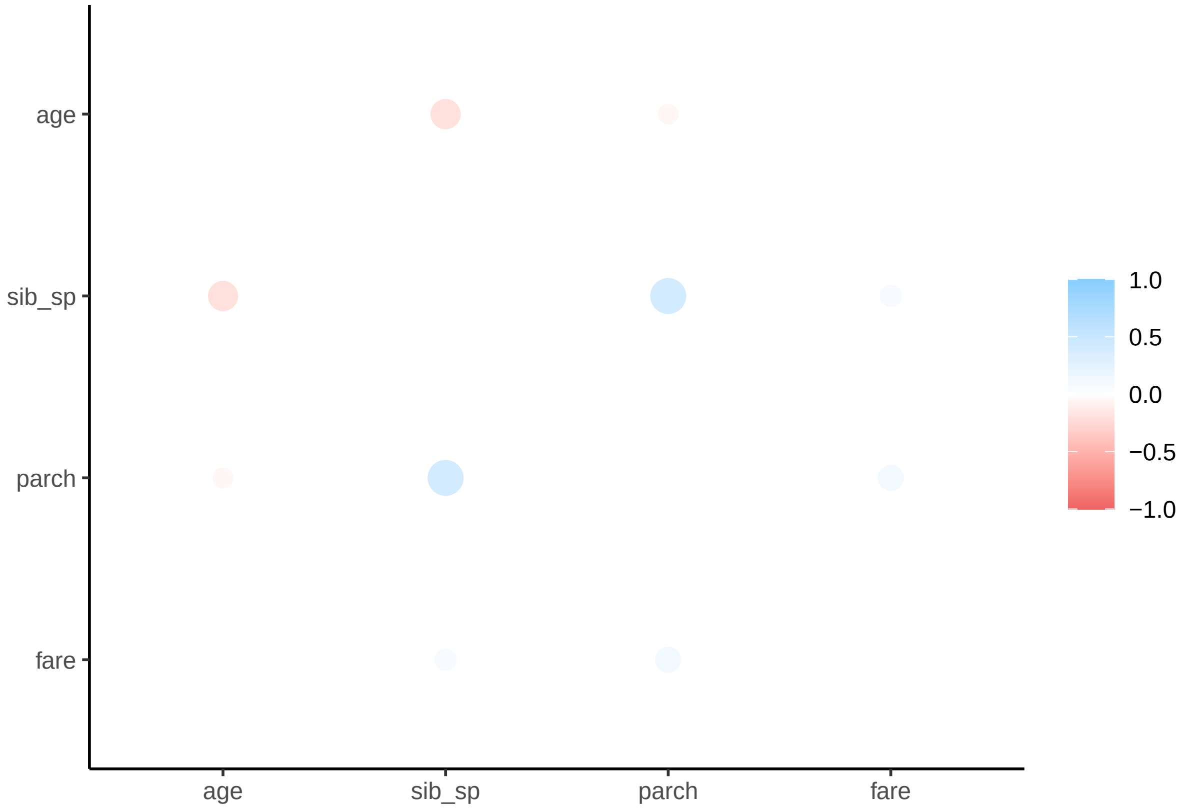
```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```

```
cor_titanic
```

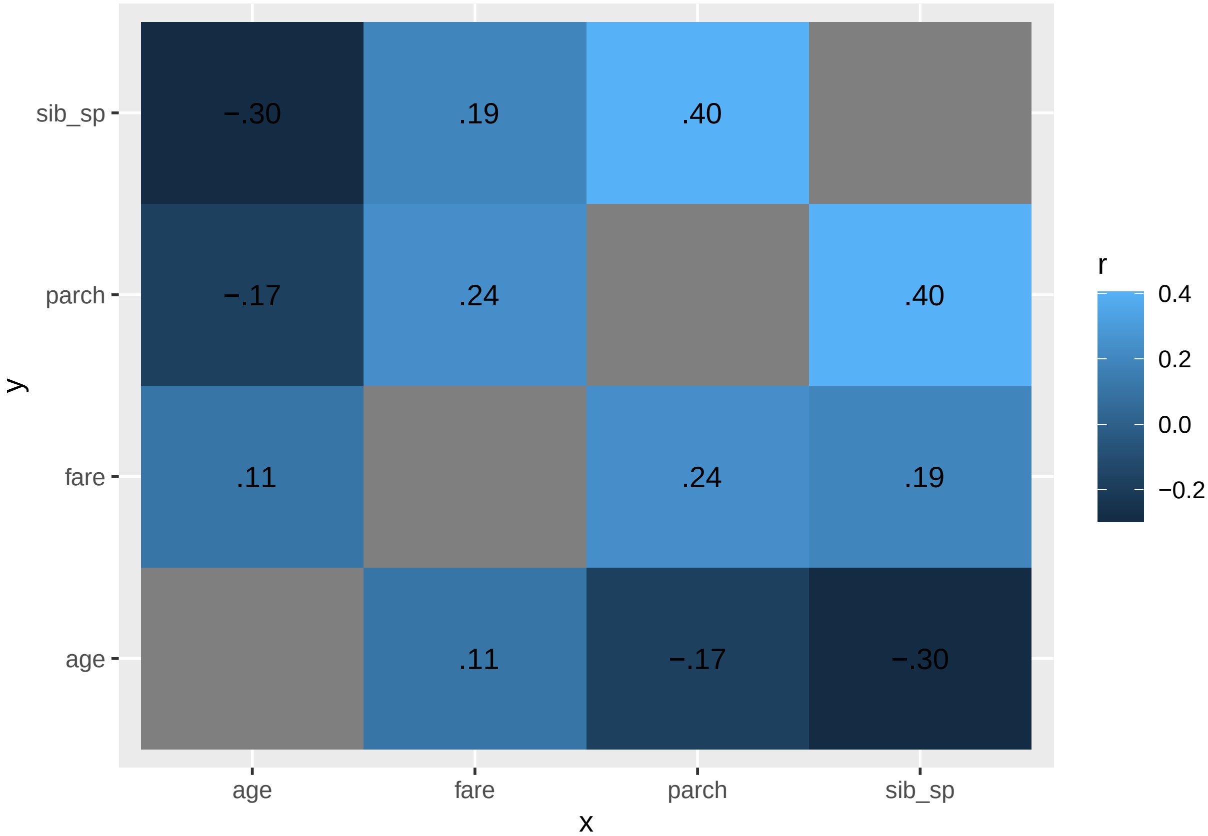
```
## # A tibble: 4 x 5
##   term      age sib_sp  parch   fare
##   <chr>   <dbl>  <dbl>  <dbl>  <dbl>
## 1 age    NA     -0.297 -0.174  0.110
## 2 sib_sp -0.297 NA      0.404  0.191
## 3 parch  -0.174  0.404 NA      0.237
## 4 fare    0.110  0.191  0.237 NA
```

```
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 plot, we can observe that the sib_sp and age are negatively correlated, sib_sp and parch are positively correlated.

Question 4 Create a recipe

```
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 + age:fare)
```

Question 5

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

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

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)         -4.34      0.618      -7.03  2.08e-12
##  2 age                  0.0579    0.0121      4.77  1.81e- 6
```

```
##   3 sib_sp           0.419     0.123        3.40  6.67e- 4
##   4 parch            0.107     0.126        0.852 3.94e- 1
##   5 fare             0.00513   0.00852      0.602 5.47e- 1
##   6 pclass_X2        1.07      0.345        3.10  1.96e- 3
##   7 pclass_X3        2.39      0.354        6.75  1.50e-11
##   8 sex_male         2.32      0.270        8.62  6.97e-18
##   9 sex_male_x_fare  0.00888   0.00628      1.41  1.58e- 1
## 10 fare_x_age      -0.000406  0.000202    -2.01  4.46e- 2
```

Question 6 LDA

```
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 QDA

```
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 naive Bayes model

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

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

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

Question 9

```
log_predict <- predict(log_fit, new_data = titanic_train, type = "prob")
```

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

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

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

```
titanic_train_predict <- bind_cols(log_predict, lda_predict, qda_predict, nb_predict)
```

```
## 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_predict
```

```
## # A tibble: 712 x 8
##    .pred_Yes...1 .pred_No...2 .pred_Yes...3 .pred_No...4 .pred_Yes...5
##            <dbl>        <dbl>         <dbl>        <dbl>         <dbl>
##  1        0.109         0.891        0.0733        0.927    0.0101
##  2        0.0832        0.917        0.0546        0.945    0.00884
##  3        0.116         0.884        0.0761        0.924    0.0114
##  4        0.331         0.669        0.274         0.726    0.107
##  5        0.106         0.894        0.0770        0.923    0.000273
##  6        0.171         0.829        0.112         0.888    0.0164
##  7        0.0284        0.972        0.0180        0.982    0.0134
##  8        0.758         0.242        0.801         0.199    0.595
##  9        0.0656        0.934        0.0506        0.949    0.00000327
## 10        0.521         0.479        0.602         0.398    0.00184
## # ... with 702 more rows, and 3 more variables: .pred_No...6 <dbl>,
## #   .pred_Yes...7 <dbl>, .pred_No...8 <dbl>
```

```
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         0.805
```

```
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         0.794
```

```
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         0.791
```

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

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
```

```
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.772
```

```
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      0.805 Logistic Regression
## 2      0.794 LDA
## 3      0.791 QDA
## 4      0.772 Naive Bayes
```

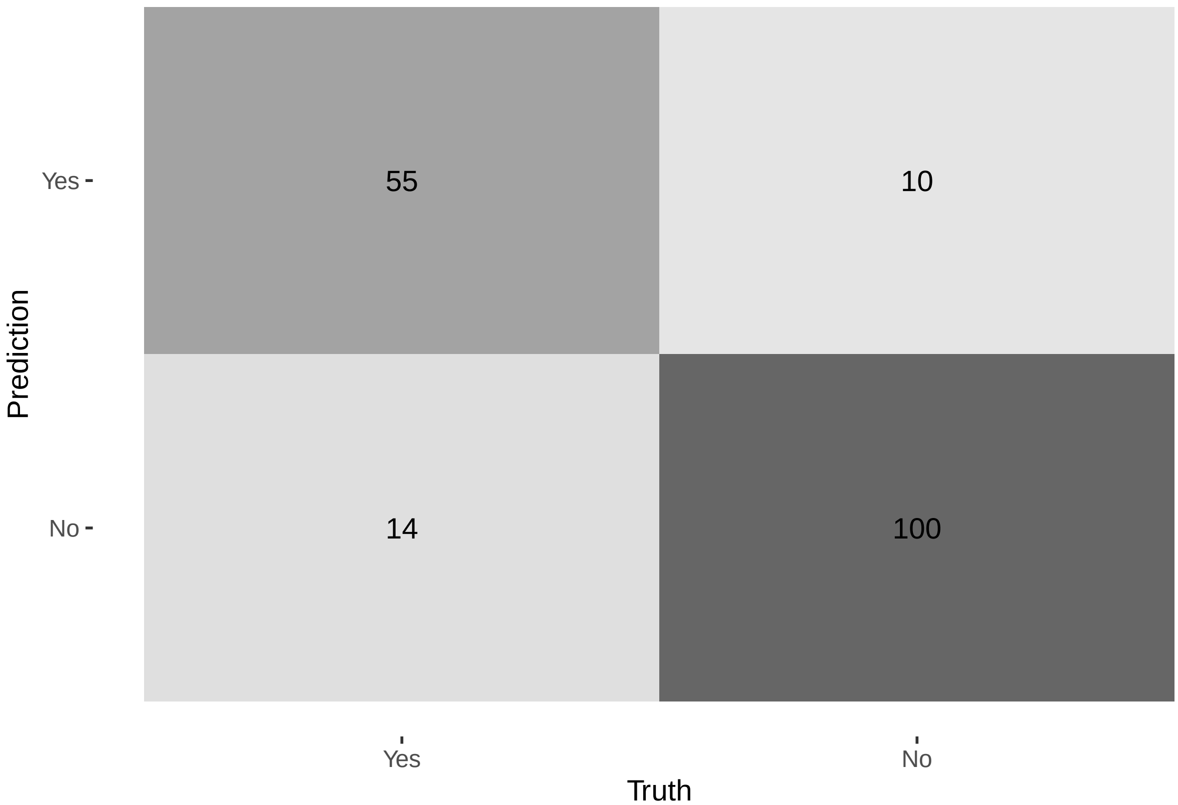
Logistic Regression achieved the highest accuracy on the training data.

Question 10

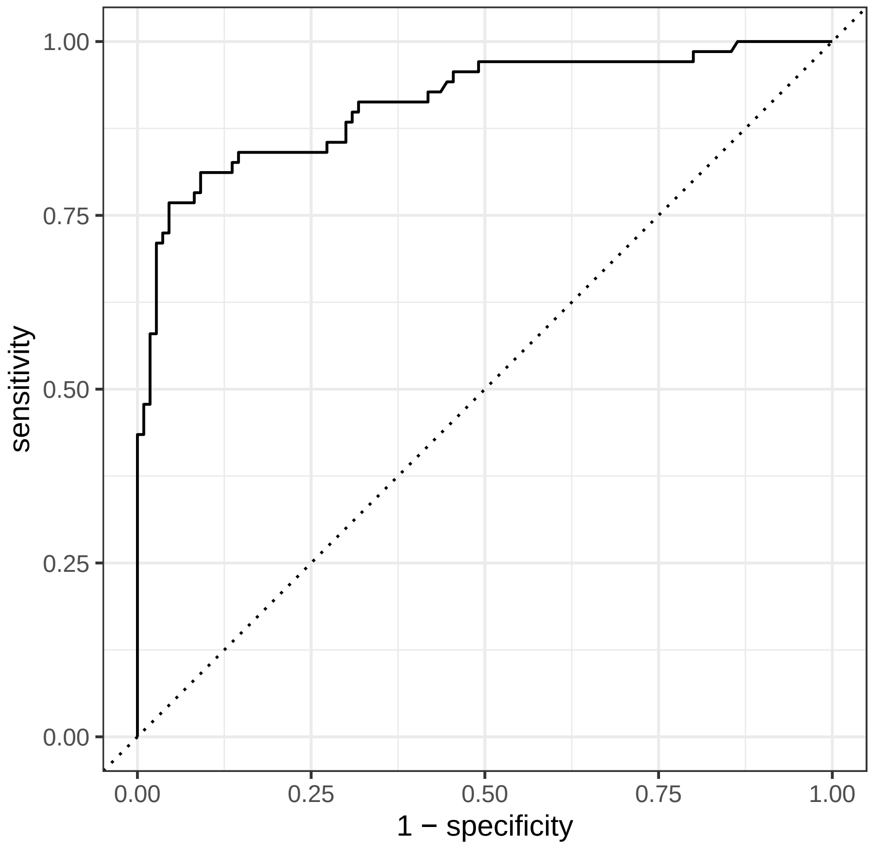
```
predict(log_fit, new_data = titanic_test, type = "prob")
```

```
## # A tibble: 179 x 2
##    .pred_Yes .pred_No
##        <dbl>    <dbl>
##  1     0.921   0.0792
##  2     0.589   0.411
##  3     0.467   0.533
##  4     0.623   0.377
##  5     0.255   0.745
##  6     0.624   0.376
##  7     0.441   0.559
##  8     0.440   0.560
##  9     0.674   0.326
## 10     0.804   0.196
## # ... with 169 more rows
```

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




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




```r
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         0.912
```

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

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.866
```

The accuracy pf the model on the testing data is approximately 86.59%, so thw model generally fits well on the tests data. The model performs well because the accuracy for training and testing data both exceed 80%. The accuracy rates are different for the two data sets, and the accuracy for testing data is slightly higher than the training accuracy, which might be due to the smaller sample size in the testing data.