Homework 3

Jules Merigot (8488256)

October 26, 2022

First, let's load the titanic data and change the *survived* and *pclass* variables to factors.

```r
# loading the data
titanic_data <- read.csv(file = "C:/Users/jules/OneDrive/Desktop/homework-3/data/titanic.csv")
head(titanic_data)

titanic_data$survived <- factor(titanic_data$survived, labels = c("Yes", "No"))
titanic_data$pclass <- factor(titanic_data$pclass)

str(titanic_data)
```

Classification

Question 1

Let's set a seed, and randomly split the data to create a training and testing set.

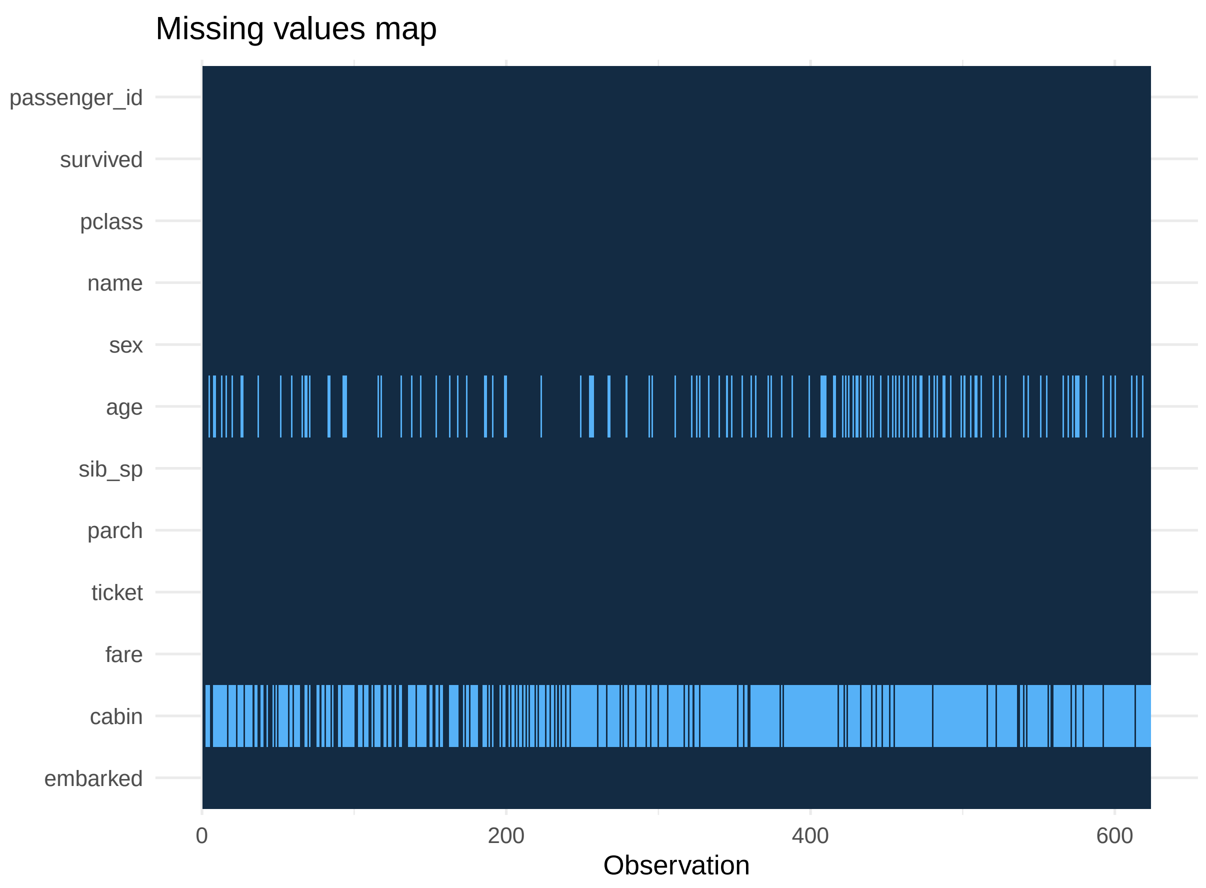
```r
# setting the seed
set.seed(8488)

titanic_split <- initial_split(titanic_data, prop=0.70, strata=survived)

titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
```

For splitting the data, I chose a proportion of 0.70 because it allows for more training data, while retaining enough data to be tested since there is a limited amount of observations. The training data has 623 observations while the testing data has 268 observations.

```r
#plot of missing values in the training data
missing_plot(titanic_train)
```



```r
# the number of missing values in the training data
sum(is.na(titanic_train))
```

```
## [1] 605
```

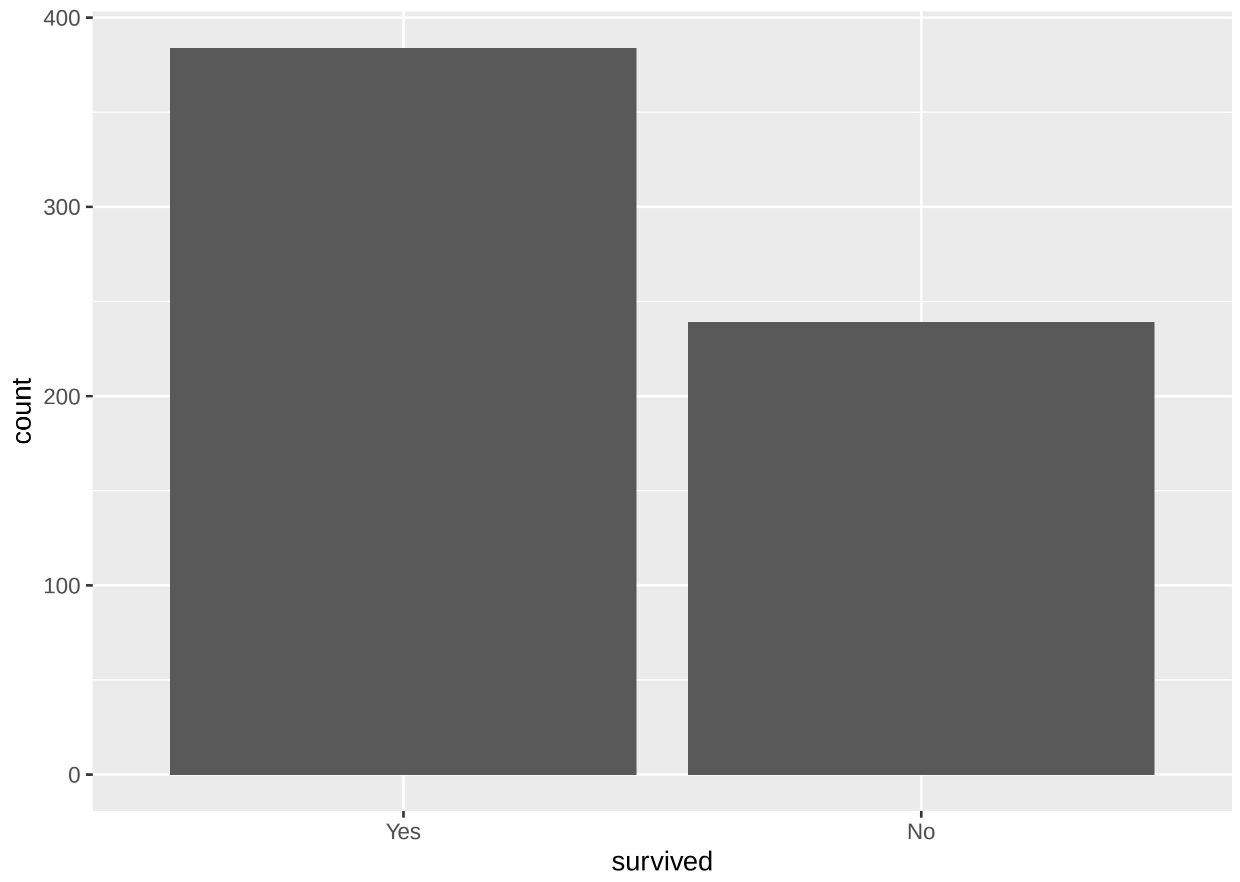
There is a good amount of missing data in the training data, 605 missing data values to be exact, particularly for the *age* and *cabin* variables, as can be seen in the plot above.

We want to use stratified sampling for this data because not only does it allow for less bias, but since we have less observations than the abalone dataset for example, stratified sampling allows for more precision on a smaller dataset, and thus a more precise sample in this case.

Question 2

Next, we explore the distribution of the outcome variables *survived*.

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



Using the above visualization of the distribution of the outcome variable *survived*, we can see that less people survived than people that perished on the Titanic. More than 300 (known) passengers lost their lives, while only a little more than 200 passengers survived.

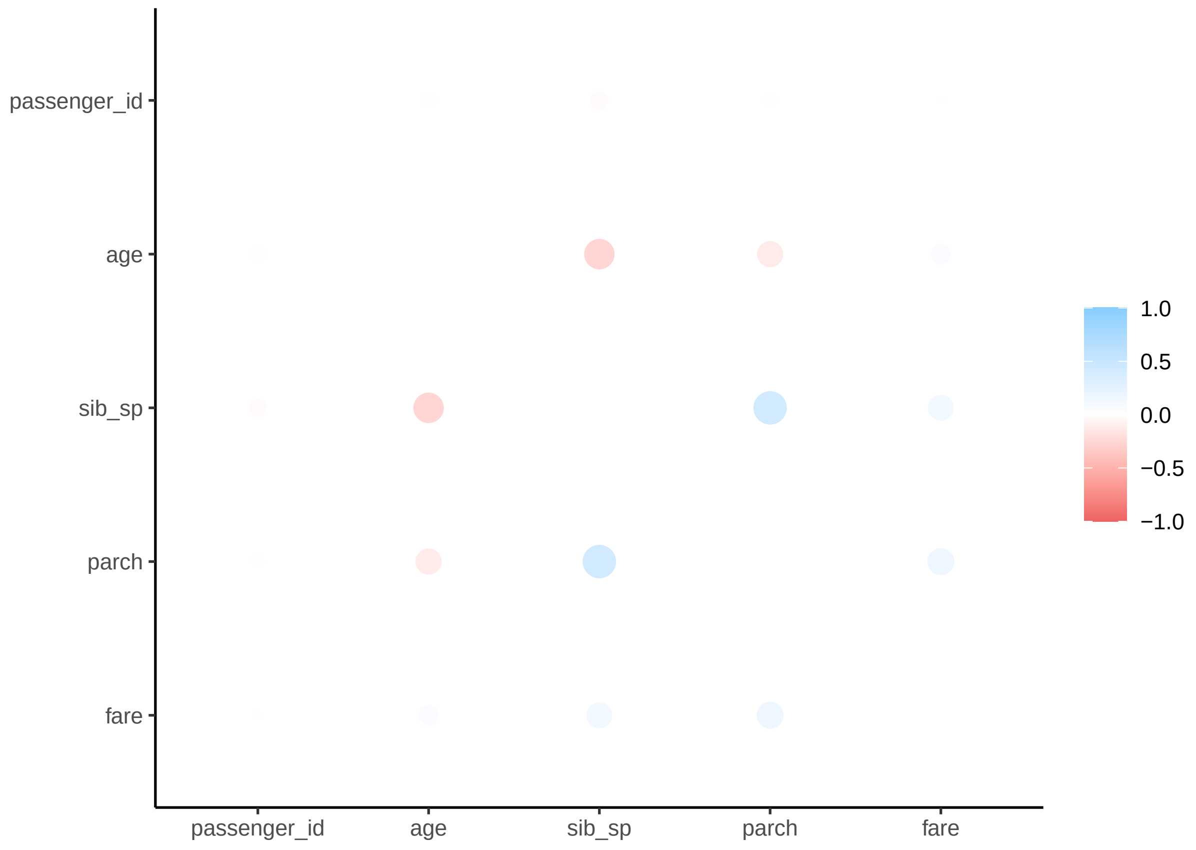
Question 3

We now create a correlation matrix and its visualization for all continuous variables.

```
cor_titanic <- titanic_train %>%
  select(where(is.numeric)) %>%
  correlate()
```

```
## Correlation computed with
## * Method: 'pearson'
## * Missing treated using: 'pairwise.complete.obs'
```

```
rplot(cor_titanic)
```



After making the correlation matrix above, there are some clear patterns that emerge, such as most variables being slightly negatively correlated with others, with some exceptions. *parch* and *sib_sp* have a positive correlation, which means that the number of siblings/spouses of a certain passenger is positively correlated with the number of children/parents of that passenger, which makes sense. Additionally, *sib_sp* and *age* are negatively correlated, which indicates that a passenger's age is negatively correlated with the number of siblings/spouses they have. This makes sense because younger passengers will tend to travel alone, and thus are less likely to have siblings or spouses.

Question 4

Let's now make our recipe, while accounting for missing values and creating the proper interactions.

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

titanic_recipe %>% prep() %>% juice()
```

```
## # A tibble: 623 x 10
##      age sib_sp parch  fare survived pclass_X2 pclass_X3 sex_m~1 sex_m~2 fare_~3
##    <dbl>  <int> <int> <dbl> <fct>        <dbl>     <dbl>   <dbl>   <dbl>   <dbl>
##  1  38        1     0 71.3  No               0         0       0     0     2709.
##  2  35        1     0 53.1  No               0         0       0     0     1858.
##  3  27        0     2 11.1  No               0         1       0     0      301.
##  4  14        1     0 30.1  No               1         0       0     0      421.
##  5  32.6      0     0 13    No               1         0       1    13      424.
##  6  34        0     0 13    No               1         0       1    13      442
##  7  28        0     0 35.5  No               0         0       1    35.5    994
##  8  25.7      0     0  7.88 No               0         1       0     0      202.
##  9  25.7      0     0  7.75 No               0         1       0     0      199.
## 10  14        1     0 11.2  No               0         1       0     0      157.
## # ... with 613 more rows, and abbreviated variable names 1: sex_male,
## #   2: sex_male_x_fare, 3: fare_x_age
```

Question 5

Making the logistic regression model, and printing the tidy output as an example.

```
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)      5.08      0.737       6.89   5.49e-12
##  2 age             -0.0707    0.0146     -4.85   1.21e- 6
##  3 sib_sp          -0.469     0.140      -3.34   8.29e- 4
##  4 parch            0.00333   0.153       0.0218 9.83e- 1
##  5 fare            -0.00417   0.0126     -0.330  7.41e- 1
##  6 pclass_X2       -1.27      0.413      -3.07   2.15e- 3
##  7 pclass_X3       -2.79      0.444      -6.27   3.54e-10
##  8 sex_male        -2.52      0.318      -7.92   2.46e-15
##  9 sex_male_x_fare -0.0126    0.00891    -1.41   1.58e- 1
## 10 fare_x_age       0.000418  0.000293    1.43   1.53e- 1
```

Question 6

Making a linear discriminant analysis model for classification using the “MASS” engine.

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

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

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

Question 7

Making a quadratic discriminant analysis model for classification using the "MASS" engine.

```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

Making a naive Bayes model for classification using the "klaR" engine, and setting the usekernel argument to FALSE.

```r
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

We now generate predictions of each of the four models using our training data, and we use the accuracy metric to asses their performance.

```r
library(yardstick)

log_predict <- predict(log_fit, titanic_train) %>%
  bind_cols(titanic_train$survived) %>%
  accuracy(truth = titanic_train$survived, estimate = .pred_class)
```

```
## New names:
## * `` -> `...2`
```

```r
log_predict

lda_predict <- predict(lda_fit, titanic_train) %>%
  bind_cols(titanic_train$survived) %>%
  accuracy(truth = titanic_train$survived, estimate = .pred_class)
```

```
## New names:
## * `` -> `...2`
```

```r
lda_predict

qda_predict <- predict(qda_fit, titanic_train) %>%
  bind_cols(titanic_train$survived) %>%
  accuracy(truth = titanic_train$survived, estimate = .pred_class)
```

```
## New names:
## * `` -> `...2`
```

```r
qda_predict

nb_predict <- predict(nb_fit, titanic_train) %>%
  bind_cols(titanic_train$survived) %>%
  accuracy(truth = titanic_train$survived, estimate = .pred_class)
```

```
## New names:
## * `` -> `...2`
```

```r
nb_predict
```

In order to compare the predictions and discover which model achieved the highest accuracy on the training data, we can make a table of the accuracy rates.

```r
accuracies <- c(log_predict$.estimate, lda_predict$.estimate,
                qda_predict$.estimate, nb_predict$.estimate)
models <- c("Logistic Regression", "LDA", "QDA", "Naive Bayes")
results <- tibble(accuracies = accuracies, models = models)
results %>%
  arrange(-accuracies)
```

```
## # A tibble: 4 x 2
##   accuracies models
##        <dbl> <chr>
## 1      0.828 Logistic Regression
## 2      0.815 LDA
## 3      0.811 QDA
## 4      0.795 Naive Bayes
```

As can be seen in the table above, the logistic regression model achieved the highest accuracy on the training data with an accuracy of 82.83%.

Question 10

Fitting the model with the highest training accuracy to the testing data. In this case, the logistic regression model.

```
test_model_acc <- predict(log_fit, titanic_test) %>%
  bind_cols(titanic_test$survived) %>%
  accuracy(truth = titanic_test$survived, estimate = .pred_class)
```

```
## New names:
## * `` -> `...2`
```

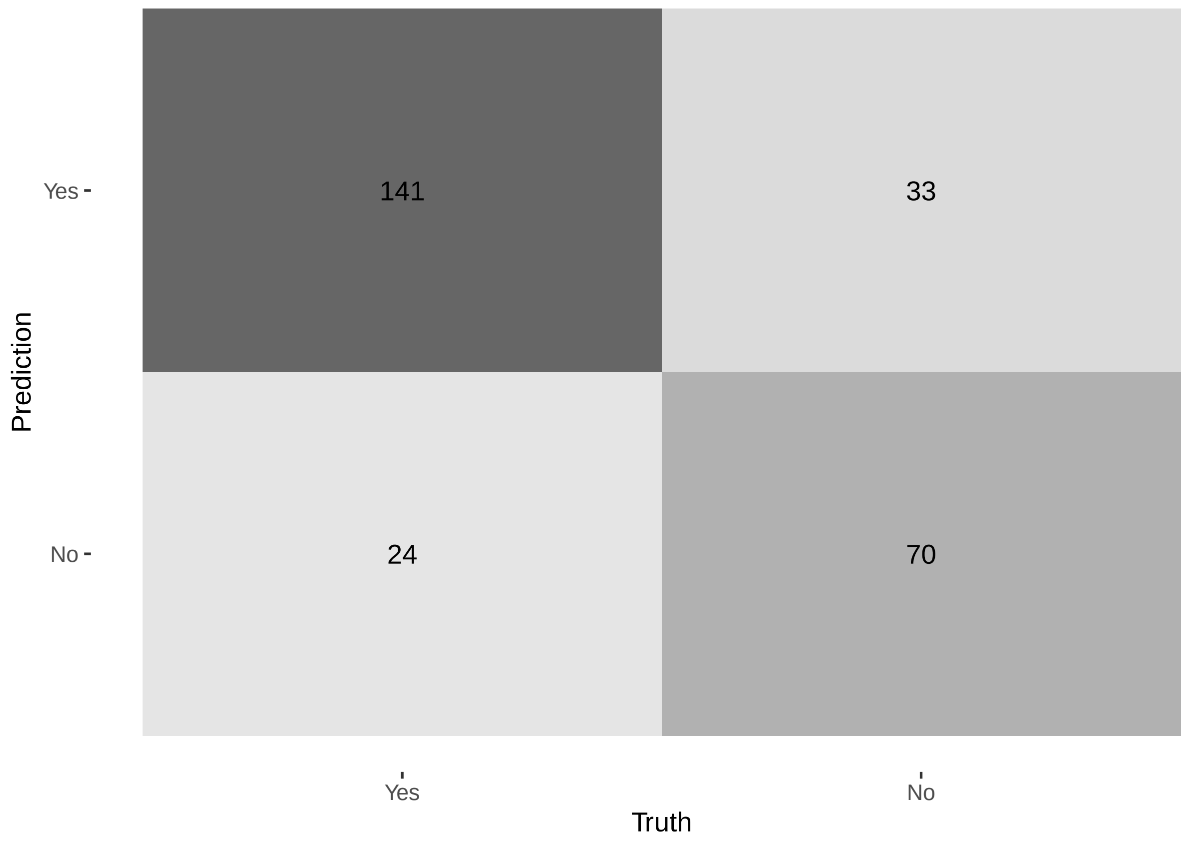
```
test_model_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.787
```

Fitting the logistic regression model on the testing data yields an accuracy of 78.73%, which is lower than for the training data, but is still rather accurate.

Now, to make a confusion matrix for the testing data and its visualization.

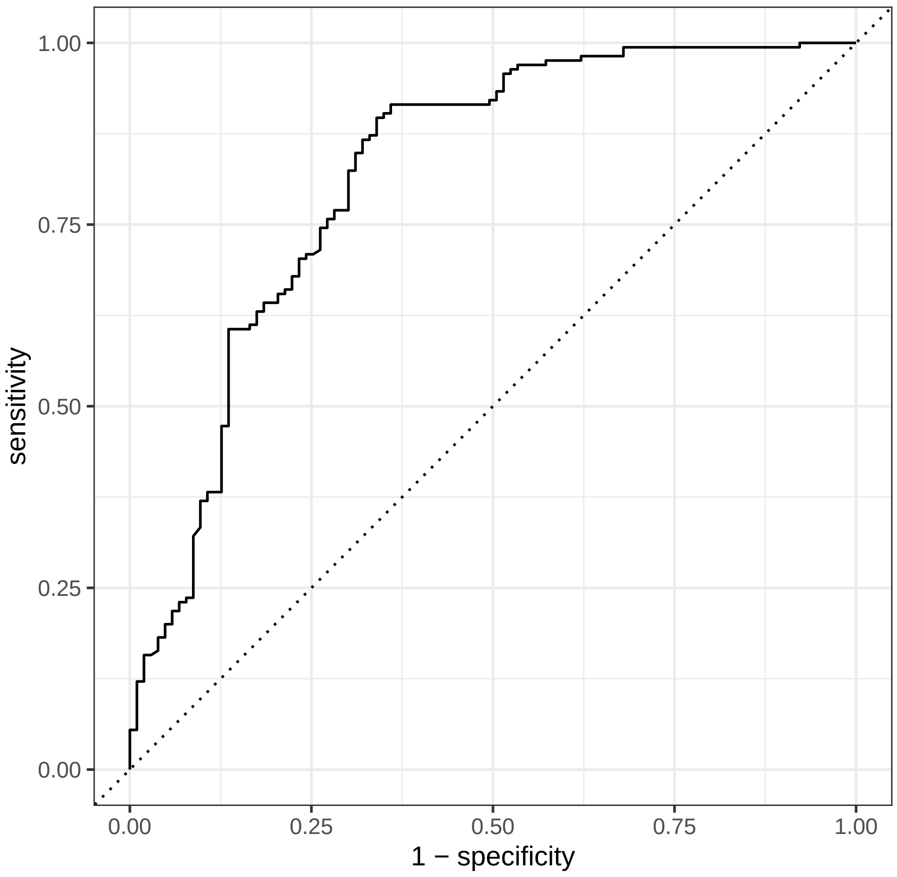
```
augment(log_fit, titanic_test) %>%
  conf_mat(truth = survived, estimate = .pred_class) %>%
  autoplot(type = "heatmap")
```



Plotting an ROC curve on the testing data and calculating the area under the curve (AUC).

```r
roc <- augment(log_fit, titanic_test)

# plotting the ROC curve
roc %>%
  roc_curve(survived, .pred_Yes) %>%
  autoplot()
```



```r
# calculating the AUC of the curve
roc %>%
  roc_auc(survived, .pred_Yes)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.818
```

As can be seen above, the AUC for the ROC curve on the testing data is 0.8184.

The model performed well on the testing data. With an accuracy of 0.7873 and an AUC of 0.8184, it can be said that the model is rather accurate. Since accuracy is weighted on a scale from 0 to 1, any value closer to 1 and above 0.75 is considered to be accurate and good. The training accuracy was slightly higher at 0.8283 than the testing accuracy at 0.7873 for the logistic regression model, which makes sense since the model was optimized for the training data in its earlier stages.