PSTAT 131 Homework 3

Matthew Zhang

Spring 2022

Classification

For this assignment, we will be working with part of a Kaggle data set that was the subject of a machine learning competition and is often used for practicing ML models. The goal is classification; specifically, to predict which passengers would survive the Titanic shipwreck.

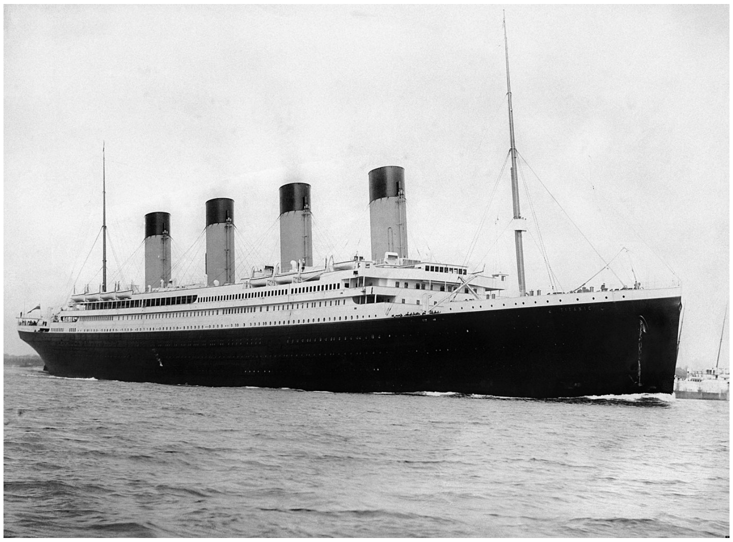


Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

Load the data from `data/titanic.csv` into *R* and familiarize yourself with the variables it contains using the codebook (`data/titanic_codebook.txt`).

Notice that `survived` and `pclass` should be changed to factors. When changing `survived` to a factor, you may want to reorder the factor so that *"Yes"* is the first level.

Make sure you load the `tidyverse` and `tidymodels`!

```
set.seed(123)
library(tidyverse)
library(tidymodels)
library(ggplot2)

titanic <- read.csv('data/titanic.csv')

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

head(titanic)
```

```
##   passenger_id survived pclass
## 1            1       No      3
## 2            2      Yes      1
## 3            3      Yes      3
## 4            4      Yes      1
## 5            5       No      3
## 6            6       No      3
##                                                  name    sex age sib_sp parch
## 1                             Braund, Mr. Owen Harris   male  22      1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38      1     0
## 3                              Heikkinen, Miss. Laina female  26      0     0
```

```
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35      1     0
## 5                            Allen, Mr. William Henry   male  35      0     0
## 6                                    Moran, Mr. James   male  NA      0     0
##             ticket    fare cabin embarked
## 1        A/5 21171  7.2500  <NA>        S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250  <NA>        S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500  <NA>        S
## 6           330877  8.4583  <NA>        Q
```

Remember that you'll need to set a seed at the beginning of the document to reproduce your results.

Question 1

Split the data, stratifying on the outcome variable, `survived.` You should choose the proportions to split the data into. Verify that the training and testing data sets have the appropriate number of observations. Take a look at the training data and note any potential issues, such as missing data.

```r
titanic_split <- initial_split(titanic, prop = 0.80,
                               strata = survived)

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

sum(is.na(titanic_train$survived))
```

```
## [1] 0
```

```r
head(titanic_train)
```

```
##    passenger_id survived pclass                           name  sex age sib_sp
## 5             5       No      3       Allen, Mr. William Henry male  35      0
## 6             6       No      3               Moran, Mr. James male  NA      0
## 7             7       No      1        McCarthy, Mr. Timothy J male  54      0
## 8             8       No      3 Palsson, Master. Gosta Leonard male   2      3
## 13           13       No      3 Saundercock, Mr. William Henry male  20      0
## 14           14       No      3    Andersson, Mr. Anders Johan male  39      1
##    parch    ticket    fare cabin embarked
## 5      0    373450  8.0500  <NA>        S
## 6      0    330877  8.4583  <NA>        Q
## 7      0     17463 51.8625   E46        S
## 8      1    349909 21.0750  <NA>        S
## 13     0 A/5. 2151  8.0500  <NA>        S
## 14     5    347082 31.2750  <NA>        S
```

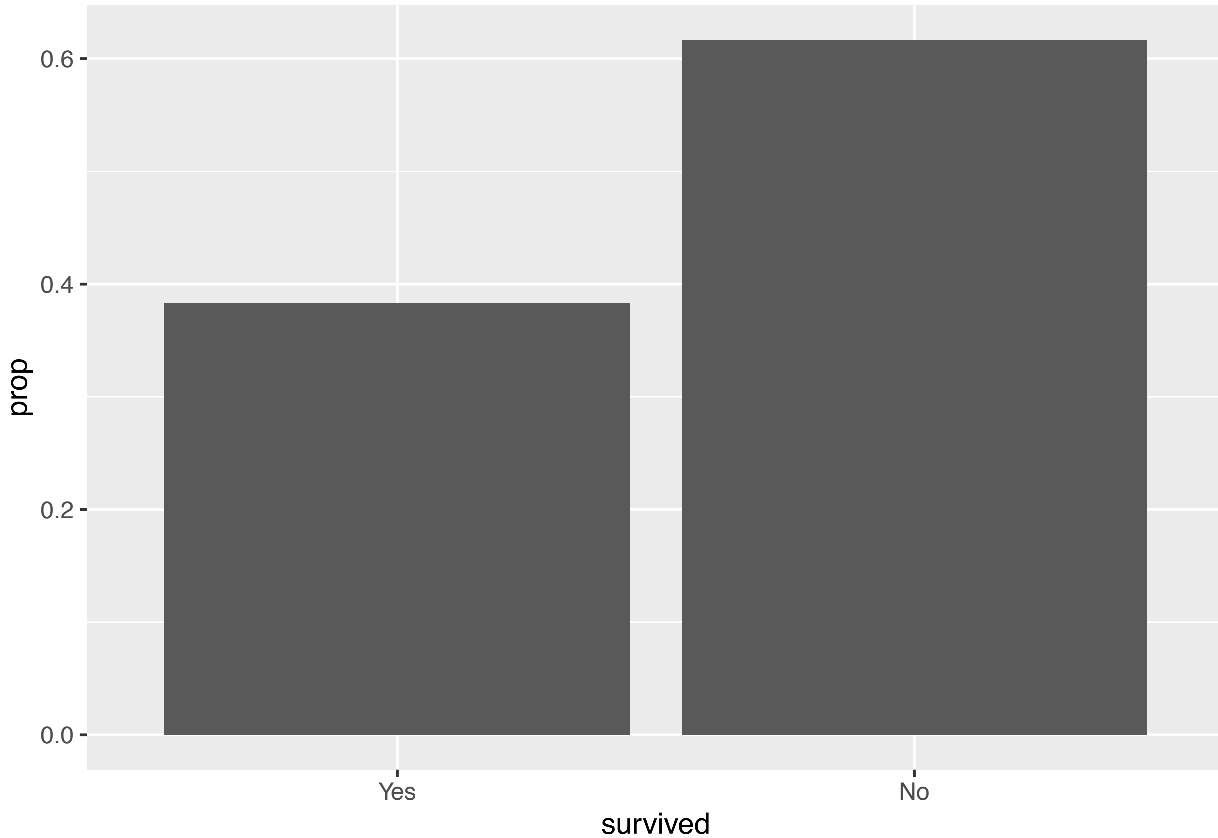
Why is it a good idea to use stratified sampling for this data?

Looking deeper into the data, it is apparent that there exists null values for the 'age' variable, while a large number of the values for 'cabin' are null values. This missing data could potentially cause issues further down the line in our model accuracy.
Stratified sampling helps divide the number of observations into a strata of smaller groups which is particularly useful for this data because of the variation in population on whether or not a person survived as a categorical variable.

Question 2

Using the **training** data set, explore/describe the distribution of the outcome variable `survived`.

```
titanic_train %>%
  ggplot(aes(x = survived)) +
  geom_bar(aes(y = ..prop.., group = 1))
```



According to the plot of the training data, we can observe that a majority of people did not survive at ~60%, while the rest of the population did survive at ~40%.

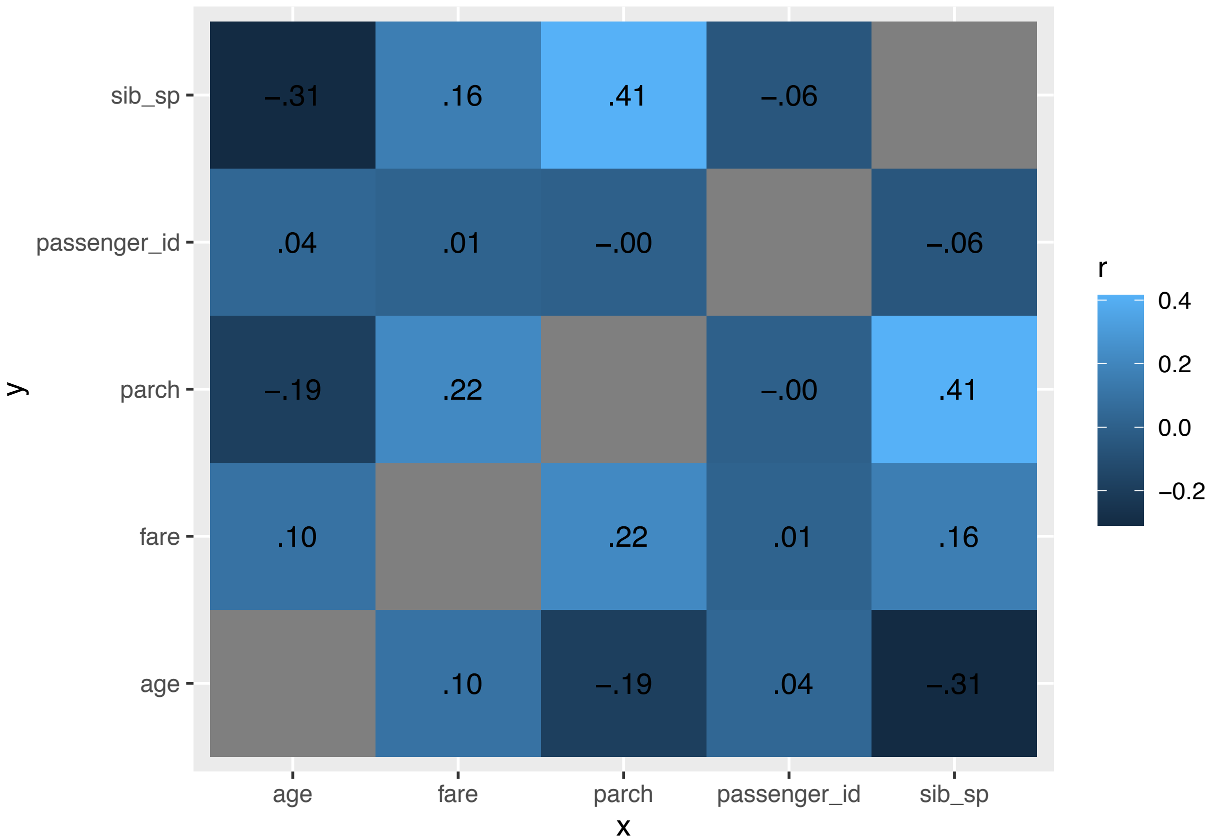
Question 3

Using the **training** data set, create a correlation matrix of all continuous variables. Create a visualization of the matrix, and describe any patterns you see. Are any predictors correlated with each other? Which ones, and in which direction?

```
# install.packages("corrr")
library(corrr)

corr_titanic <- titanic %>% dplyr::select(where(is.numeric)) %>% correlate()

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



By looking at the correlation matrix/heat map, we can see that most of the variables are not correlated with each other, determined by the value being below .5. The highest correlation value that exists is with the 'sib_sp' and 'parch' variables which is .41 while the others variable relationships are close to 0 with no correlation. Further, we can see that 'sib_sp' and 'age' have a fairly negative correlation at -.31.

Question 4

Using the **training** data, create a recipe predicting the outcome variable `survived`. Include the following predictors: ticket class, sex, age, number of siblings or spouses aboard, number of parents or children aboard, and passenger fare.

Recall that there were missing values for `age`. To deal with this, add an imputation step using `step_impute_linear()`. Next, use `step_dummy()` to **dummy** encode categorical predictors. Finally, include interactions between:

- Sex and passenger fare, and
- Age and passenger fare.

You'll need to investigate the `tidymodels` documentation to find the appropriate step functions to use.

```
?tidymodels
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(terms = ~ starts_with("sex"):fare) %>%
                          step_interact(terms = ~ age:fare)

titanic_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          6
##
## Operations:
##
## Linear regression imputation for age
## Dummy variables from all_nominal_predictors()
## Interactions with starts_with("sex"):fare
## Interactions with age:fare
```

Question 5

Specify a **logistic regression** model for classification using the `"glm"` engine. Then create a workflow. Add your model and the appropriate recipe. Finally, use `fit()` to apply your workflow to the **training** data.

Hint: Make sure to store the results of `fit()`. You'll need them later on.

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

log_wf <- workflow() %>%
  add_model(log_reg) %>%
  add_recipe(titanic_recipe)

log_fit <- fit(log_wf, titanic_train)
```

Question 6

Repeat Question 5, but this time specify a linear discriminant analysis model for classification using the `"MASS"` engine.

```r
# install.packages("discrim")
library(discrim)

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

lda_wf <- workflow() %>%
  add_model(lda_model) %>%
  add_recipe(titanic_recipe)

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

Question 7

Repeat Question 5, but this time specify a quadratic discriminant analysis model for classification using the `"MASS"` engine.

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

qda_wf <- workflow() %>%
  add_model(qda_model) %>%
  add_recipe(titanic_recipe)

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

Question 8

Repeat Question 5, but this time specify a naive Bayes model for classification using the `"klaR"` engine. Set the `usekernel` argument to `FALSE`.

```r
# install.packages("klaR")
library(klaR)

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

nb_wf <- workflow() %>%
  add_model(nb_model) %>%
  add_recipe(titanic_recipe)

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

Question 9

Now you've fit four different models to your training data.

Use `predict()` and `bind_cols()` to generate predictions using each of these 4 models and your **training** data. Then use the *accuracy* metric to assess the performance of each of the four models.

Which model achieved the highest accuracy on the training data?

```r
log_pred <- bind_cols(predict(log_fit, new_data = titanic_train), titanic_train %>% dplyr::select(survi
log_pred
```

```
## # A tibble: 712 x 2
##    .pred_class survived
##    <fct>       <fct>
##  1 No          No
##  2 No          No
##  3 No          No
##  4 No          No
##  5 No          No
##  6 No          No
##  7 Yes         No
##  8 No          No
##  9 No          No
## 10 Yes         No
## # ... with 702 more rows
```

```r
log_acc <- log_pred %>%
          accuracy(truth=survived, estimate = .pred_class)
log_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.812
```

```r
lda_pred <- bind_cols(predict(lda_fit, new_data = titanic_train), titanic_train %>% dplyr::select(survi
lda_pred
```

```
## # A tibble: 712 x 2
##    .pred_class survived
##    <fct>       <fct>
##  1 No          No
##  2 No          No
##  3 No          No
##  4 No          No
##  5 No          No
##  6 No          No
##  7 Yes         No
##  8 Yes         No
##  9 No          No
## 10 Yes         No
## # ... with 702 more rows
```

```r
lda_acc <- lda_pred %>%
          accuracy(truth=survived, estimate = .pred_class)
lda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.805
```

```r
qda_pred <- bind_cols(predict(qda_fit, new_data = titanic_train), titanic_train %>% dplyr::select(survi
qda_pred
```

```
## # A tibble: 712 x 2
##    .pred_class survived
##    <fct>       <fct>
##  1 No          No
##  2 No          No
##  3 No          No
##  4 No          No
##  5 No          No
##  6 No          No
##  7 No          No
##  8 No          No
##  9 No          No
## 10 No          No
## # ... with 702 more rows
```

```r
qda_acc <- qda_pred %>%
          accuracy(truth=survived, estimate = .pred_class)
```

```
qda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.764
```

```
nb_pred <- suppressWarnings(bind_cols(predict(nb_fit, new_data = titanic_train), titanic_train %>% dply
nb_pred
```

```
## # A tibble: 712 x 2
##    .pred_class survived
##    <fct>       <fct>
##  1 No          No
##  2 No          No
##  3 Yes         No
##  4 No          No
##  5 No          No
##  6 No          No
##  7 No          No
##  8 No          No
##  9 No          No
## 10 No          No
## # ... with 702 more rows
```

```
nb_acc <- nb_pred %>%
          accuracy(truth=survived, estimate = .pred_class)
nb_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.765
```

Looking at the accuracy of all the models, the logistic regression model performed the best with an accuracy of 82%.

Question 10

Fit the model with the highest training accuracy to the **testing** data. Report the accuracy of the model on the **testing** data.

Again using the **testing** data, create a confusion matrix and visualize it. Plot an ROC curve and calculate the area under it (AUC).

How did the model perform? Compare its training and testing accuracies. If the values differ, why do you think this is so?

```
bind_cols(predict(log_fit, new_data = titanic_test), titanic_test %>% dplyr::select(survived))
```

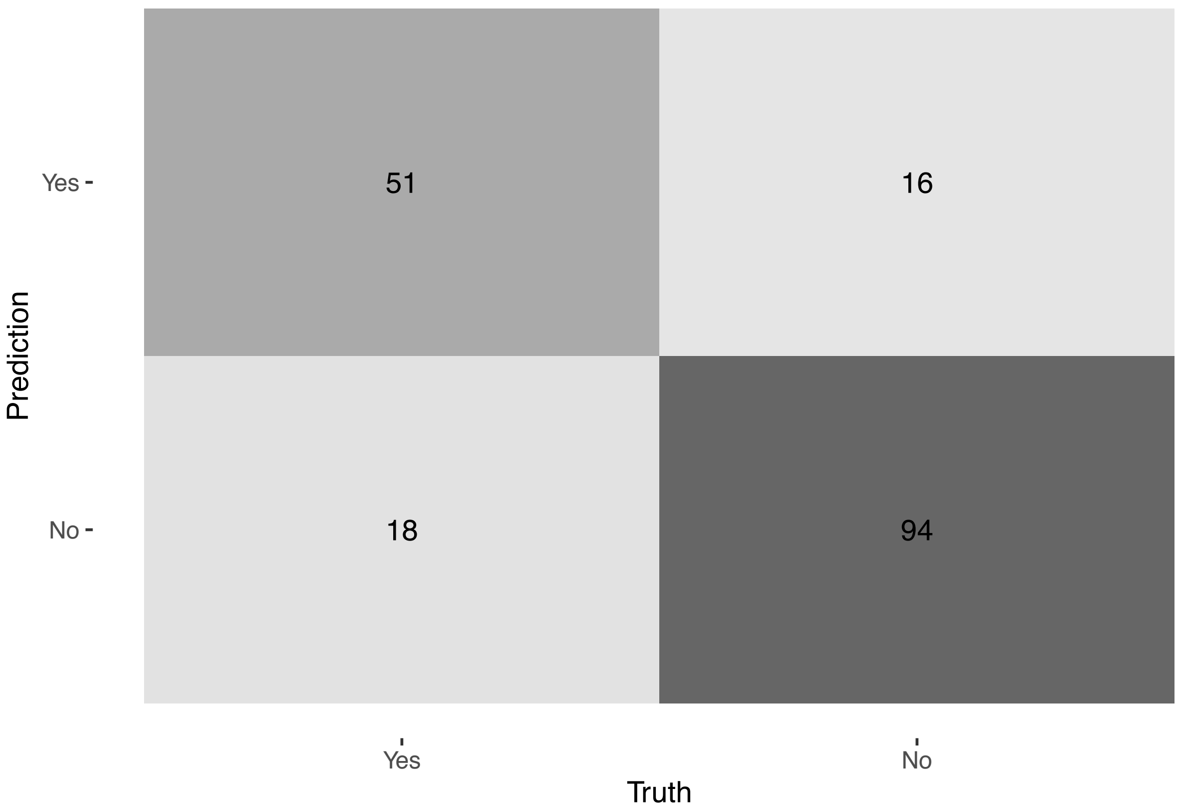
```
## # A tibble: 179 x 2
##    .pred_class survived
##    <fct>       <fct>
##  1 No          No
##  2 Yes         Yes
##  3 Yes         Yes
##  4 Yes         Yes
##  5 No          No
```

```
##   6 No          No
##   7 Yes         Yes
##   8 No          No
##   9 Yes         Yes
##  10 Yes         No
## # ... with 169 more rows
```

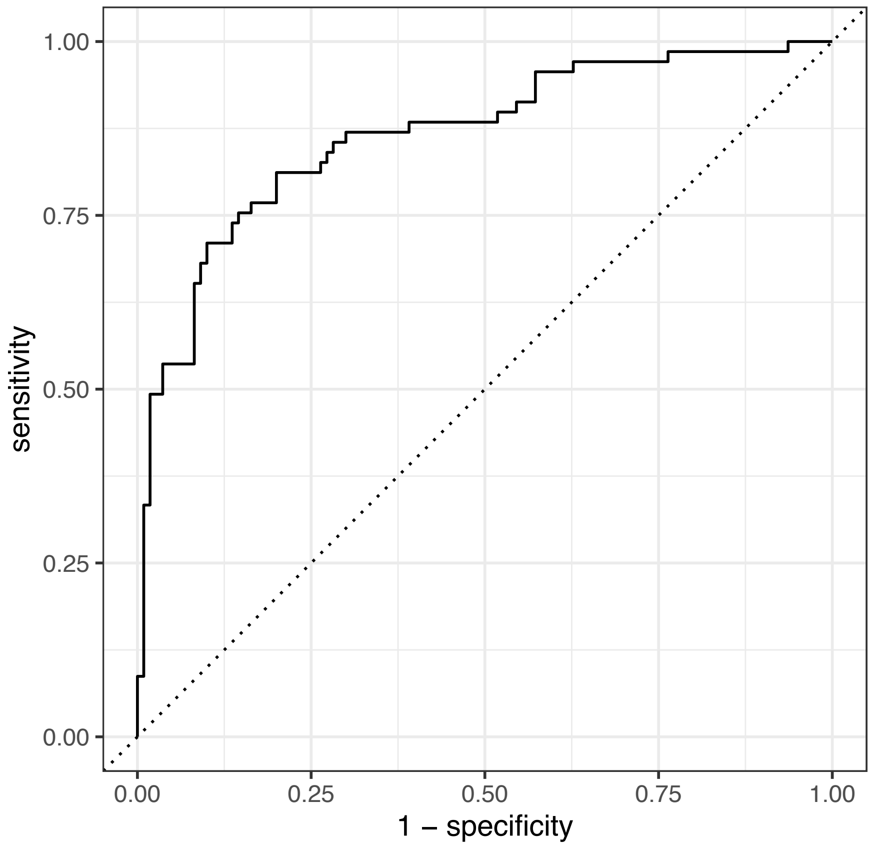
```r
bind_cols(predict(log_fit, new_data = titanic_test), titanic_test %>%
dplyr::select(survived)) %>% accuracy(truth=survived, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.810
```

```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()
```




```
pROC::auc(augment(log_fit, new_data = titanic_test)$survived, augment(log_fit, new_data = titanic_test)$
```

```
## Area under the curve: 0.8652
```

Looking at the results, we can see the AUC is almost 86% while the training and testing accuracies remain very similar at around 81%, both results indicating positive model performance and that the model was accurate in its predictions.