HW3

Xilong Li (3467966)

2022-04-18

```
#Note: ALL of the codes in this homework are cited from lab03!

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidymodels)
```

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

```
## v broom        0.7.12     v rsample      0.1.1
## v dials        0.1.0      v tune         0.2.0
## v infer        1.0.0      v workflows    0.2.6
## v modeldata    0.1.1      v workflowsets 0.2.1
## v parsnip      0.2.1      v yardstick    0.0.9
## v recipes      0.2.0
```

```
## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/
```

```
library(discrim)
```

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

```
library(poissonreg)
library(corrr)
library(klaR) # for naive bayes
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

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

```
tidymodels_prefer()
Titanic <- read.csv("titanic.csv")
Titanic$survived <- as.factor(Titanic$survived)
Titanic$pclass <- as.character(Titanic$pclass)
Titanic$pclass <- as.factor(Titanic$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
```

Question 1:

```
set.seed(2216)

titan_split <- initial_split(Titanic, prop = 0.80,
                             strata = survived)
titan_train <- training(titan_split)
titan_test <- testing(titan_split)
c(nrow(titan_train),nrow(titan_test),nrow(Titanic))
```

```
## [1] 712 179 891
```

```
head(titan_train)
```

```
##    passenger_id survived pclass                                 name    sex age
## 1             1       No      3              Braund, Mr. Owen Harris   male  22
## 5             5       No      3             Allen, Mr. William Henry   male  35
## 7             7       No      1              McCarthy, Mr. Timothy J   male  54
## 13           13       No      3       Saundercock, Mr. William Henry   male  20
## 14           14       No      3          Andersson, Mr. Anders Johan   male  39
## 15           15       No      3 Vestrom, Miss. Hulda Amanda Adolfina female  14
##    sib_sp parch    ticket    fare cabin embarked
## 1       1     0 A/5 21171  7.2500  <NA>        S
## 5       0     0    373450  8.0500  <NA>        S
## 7       0     0     17463 51.8625   E46        S
## 13      0     0 A/5. 2151  8.0500  <NA>        S
## 14      1     5    347082 31.2750  <NA>        S
## 15      0     0    350406  7.8542  <NA>        S
```

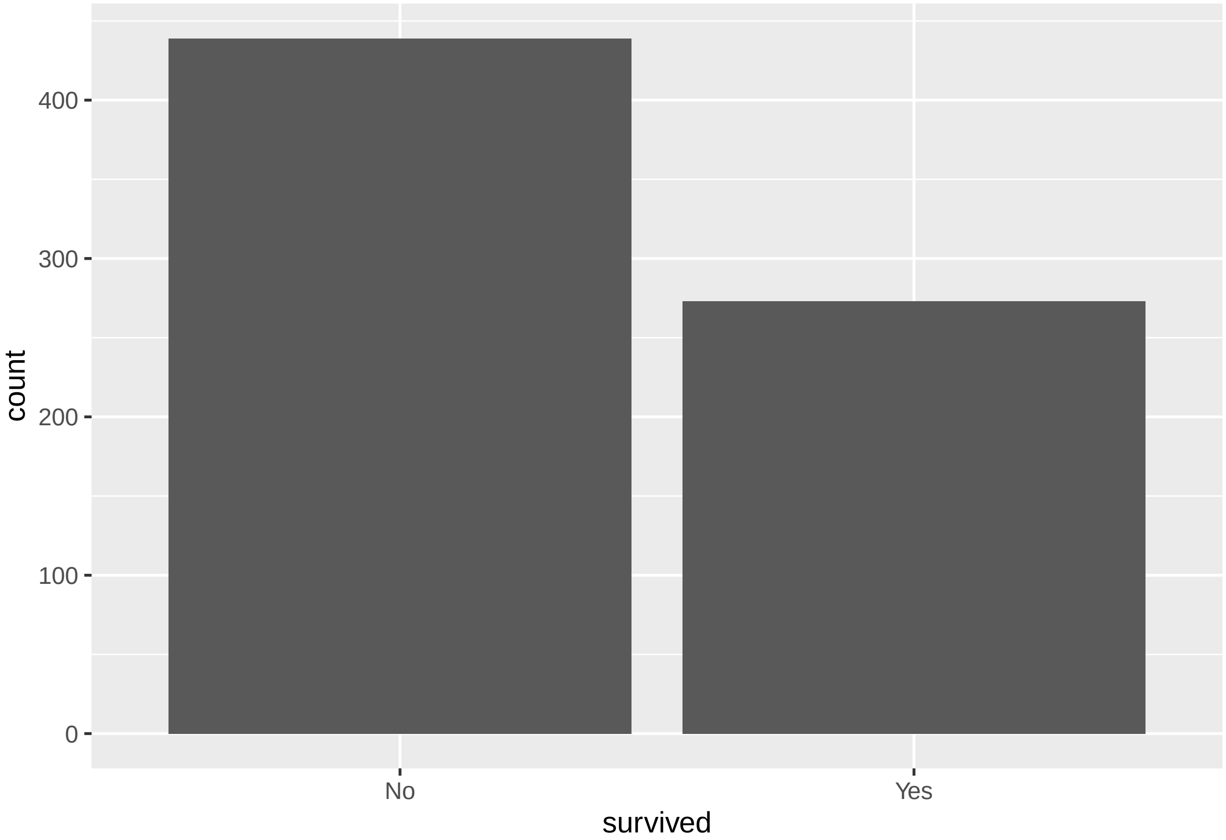
```
sum(is.na(titan_train$survived))
```

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

Therefore, as I have checked, there are no missing data on the column of “survived” in the training data, while there are indeed some missing data in other columns of the training data.
It is important to use stratified sampling in this data, because the result we want to predict is categorical parameter. Therefore, we need also to proportionally split the data based on the stratification.
We can also notice a possible problem which is that the column “ticket” has very untidy values, which might cause a problem during the training.

Question 2:

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



Therefore, as the graph above has shown, in the training data, there are more people who did not survive than those who did survive.

Question 3:

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

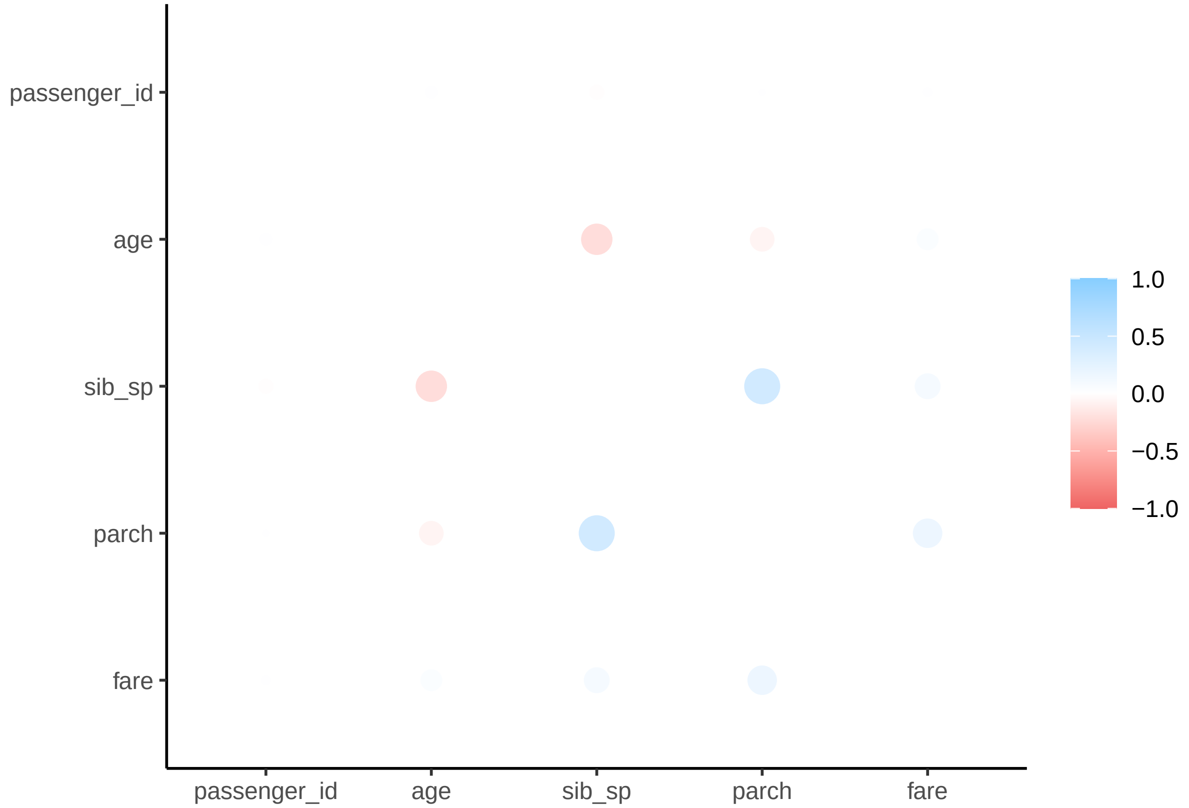
```
#install.packages("corrr")
library(corrr)

cor_titan <- titan_train %>%
  select (-c(survived,pclass,sex,embarked,name,ticket,cabin)) %>%
  correlate()
```

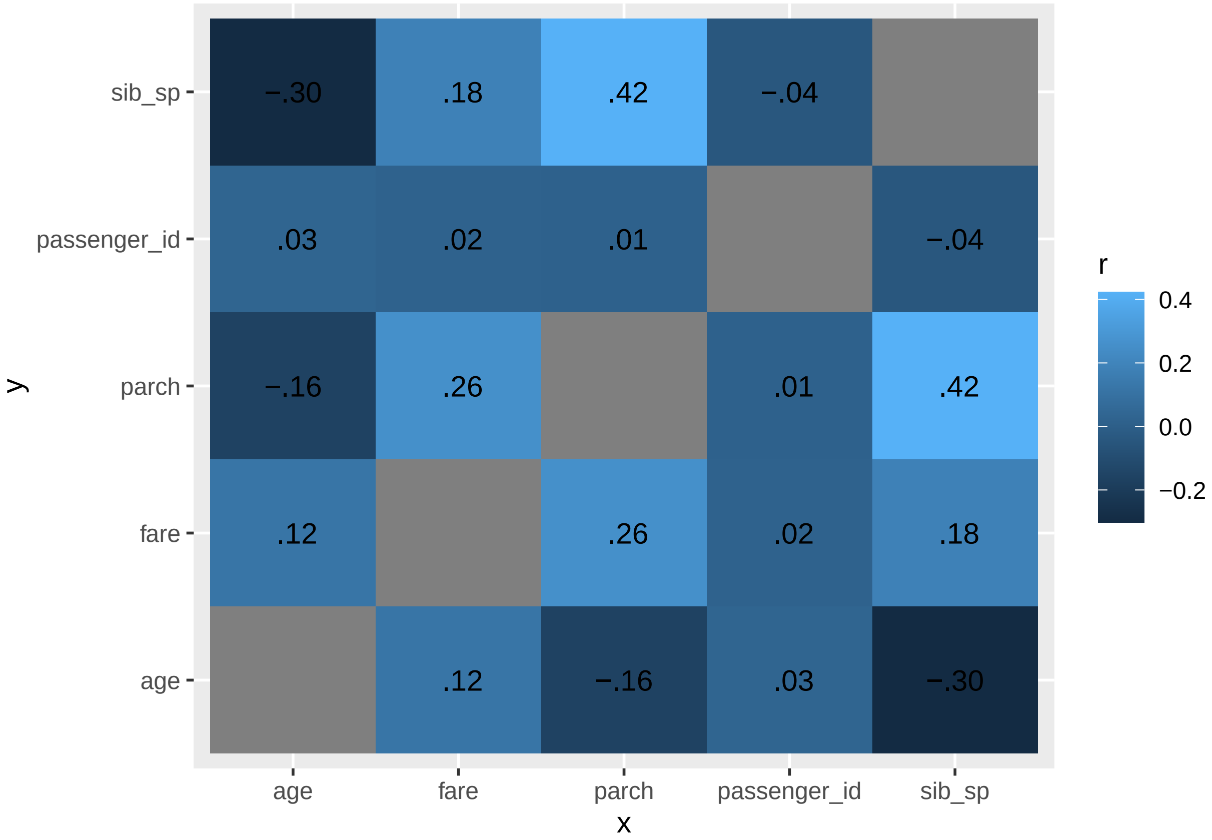
```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```

```
rplot(cor_titan)
```

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



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



As the graph shown above:
1) age has negative correlation with sib_sp; 2) age has slightly negative correlation with parch; 3) sib_sp has positive correlation with parch; 4) sib_sp has slightly positive correlation with fare; 5) parch has slightly positive correlation with fare;

Question 4:

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

titan_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:

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

log_wkflow <- workflow() %>%
  add_model(log_reg) %>%
  add_recipe(titan_recipe)

log_fit <- fit(log_wkflow, titan_train)
```

Question 6:

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

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

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

Question 7:

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

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

qda_fit <- fit(qda_wkflow, titan_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(titan_recipe)

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

Question 9:

```
# calculating the prediction accuracy of each model:
head(predict(log_fit, new_data = titan_train, type = "prob"))
```

```
## # A tibble: 6 x 2
##   .pred_No .pred_Yes
##      <dbl>     <dbl>
## 1    0.901    0.0986
## 2    0.917    0.0834
## 3    0.705    0.295
## 4    0.822    0.178
## 5    0.965    0.0352
## 6    0.221    0.779
```

```
log_reg_acc <- augment(log_fit, new_data = titan_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
augment(log_fit, new_data = titan_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction  No Yes
##        No  392  85
##        Yes  47 188
```

```
head(predict(lda_fit, new_data = titan_train, type = "prob"))
```

```
## # A tibble: 6 x 2
##   .pred_No .pred_Yes
##      <dbl>     <dbl>
## 1    0.937    0.0632
## 2    0.950    0.0499
## 3    0.765    0.235
## 4    0.892    0.108
## 5    0.978    0.0223
## 6    0.171    0.829
```

```
lda_acc <- augment(lda_fit, new_data = titan_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
augment(log_fit, new_data = titan_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction  No Yes
##        No  392  85
##        Yes  47 188
```

```
head(predict(qda_fit, new_data = titan_train, type = "prob"))
```

```
## # A tibble: 6 x 2
##   .pred_No .pred_Yes
##      <dbl>     <dbl>
## 1    0.994   0.00642
## 2    0.995   0.00530
## 3    0.937   0.0633
## 4    0.989   0.0111
## 5    0.975   0.0251
## 6    0.348   0.652
```

```
qda_acc <- augment(qda_fit, new_data = titan_train) %>%
  accuracy(truth = survived, estimate = .pred_class)
augment(log_fit, new_data = titan_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction  No Yes
##        No  392  85
##        Yes  47 188
```

```
head(predict(nb_fit, new_data = titan_train, type = "prob"))
```

```
## # A tibble: 6 x 2
##   .pred_No .pred_Yes
##      <dbl>     <dbl>
## 1    0.985   0.0151
## 2    0.984   0.0157
## 3    0.627   0.373
## 4    0.982   0.0176
## 5    0.994   0.00553
## 6    0.541   0.459
```

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

```
##            Truth
## Prediction  No Yes
##        No  392  85
##        Yes  47 188
```

```
# Summarizing the accuracy of each model:

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      0.802 LDA
## 3      0.799 QDA
## 4      0.772 Naive Bayes
```

Therefore, as it is shown above, the logistic regression model has the highest accuracy, and thus I will choose the logistic regression model as the best prediction.

Question 10:

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

```
## # A tibble: 179 x 2
##    .pred_No .pred_Yes
##       <dbl>     <dbl>
##  1   0.0864    0.914
##  2   0.883     0.117
##  3   0.913     0.0873
##  4   0.432     0.568
##  5   0.736     0.264
##  6   0.755     0.245
##  7   0.949     0.0510
##  8   0.831     0.169
##  9   0.965     0.0352
## 10   0.225     0.775
## # ... with 169 more rows
```

```
augment(log_fit, new_data = titan_test) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

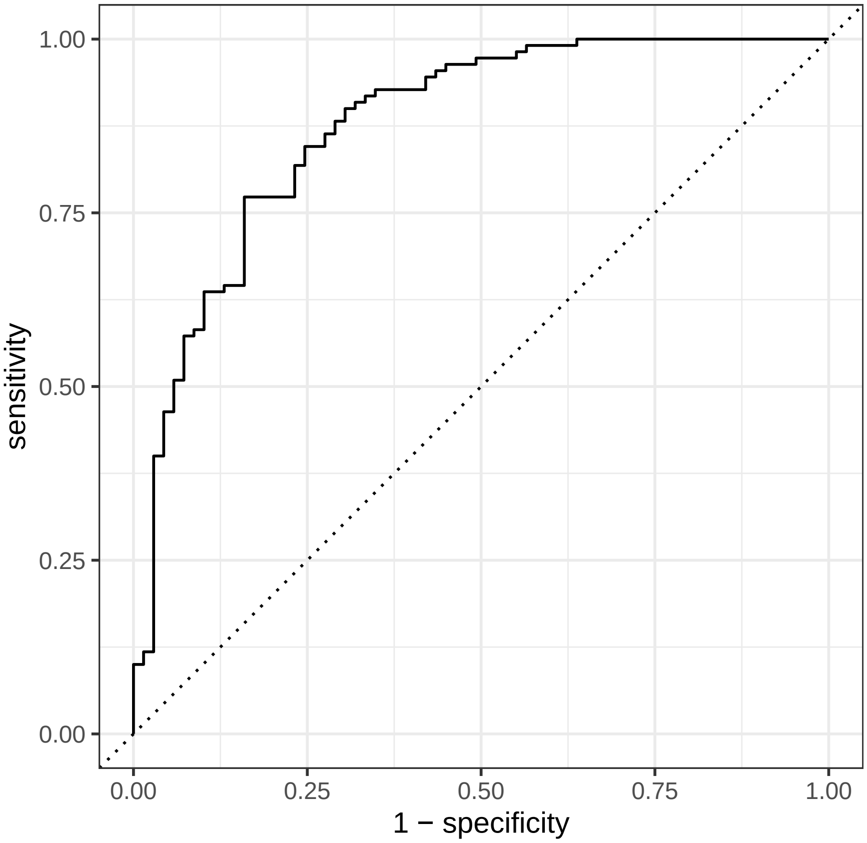
```
##            Truth
## Prediction No Yes
##        No  94  19
##        Yes 16  50
```

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

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

```
## # A tibble: 3 x 3
##   .metric     .estimator .estimate
##   <chr>       <chr>          <dbl>
## 1 accuracy    binary         0.804
## 2 sensitivity binary         0.855
## 3 specificity binary         0.725
```

```
augment(log_fit, new_data = titan_test) %>%
  roc_curve(survived, .pred_No) %>%
  autoplot()
```


sensitivity
1 − specificity