Homework03

Contents


```
set.seed(231)

data = read.csv('data/titanic.csv')

data$survived = factor(data$survived, levels = c('Yes', 'No'))
data$pclass = factor(data$pclass)
levels(data$survived)
```

```
## [1] "Yes" "No"
```

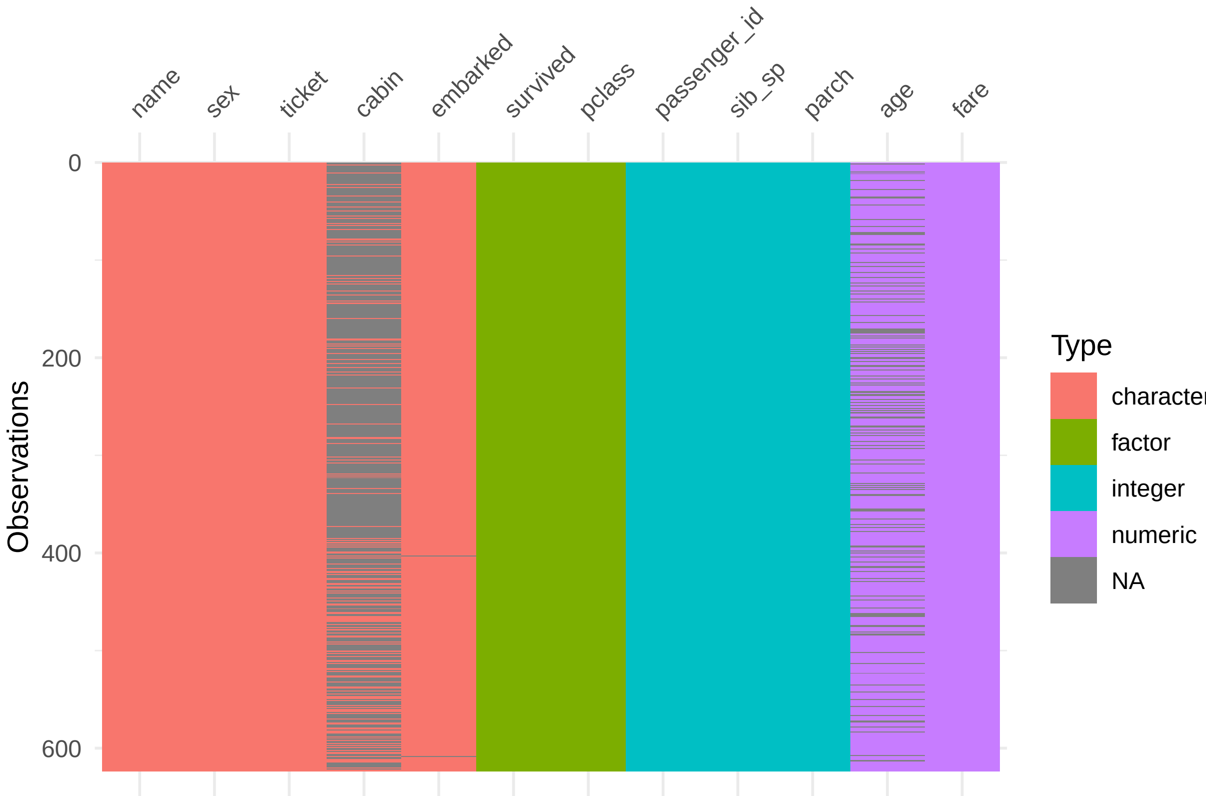
Question 1

```
data_split <- initial_split(data, prop = 0.70,
                                strata = survived)

data_train <- training(data_split)
data_test <- testing(data_split)
```

```
print(c(dim(data_train), dim(data_test)))
```

```
## [1] 623  12 268  12
```

```
vis_dat(data_train)
```

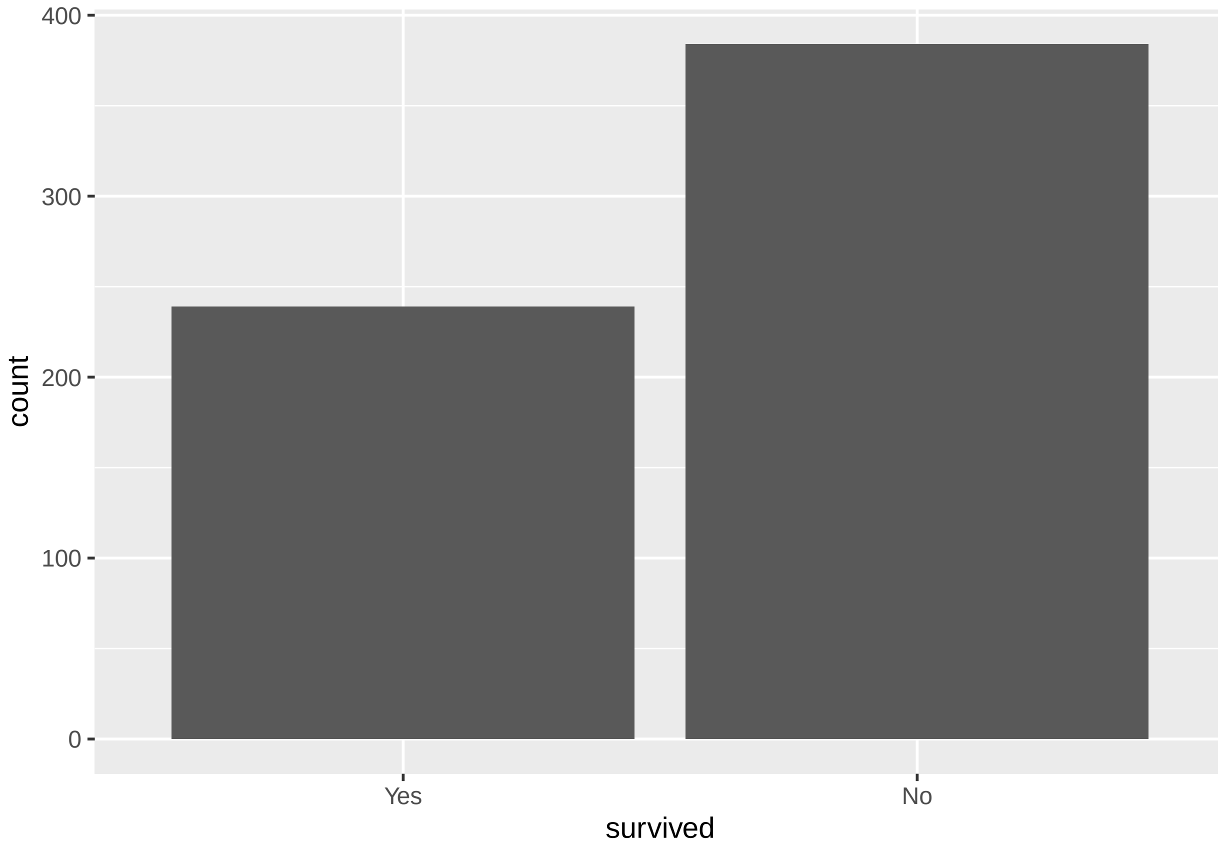


'age' and 'cabin' variables have NA. Categorical variables need to shift as dummy variables.

If we do not target variable, 'survived' will be able to lean on one side whether train or test set. If it's like this, it could be hard to train models.

Question 2

```
ggplot(data_train, aes(x = survived)) +
        geom_bar()
```



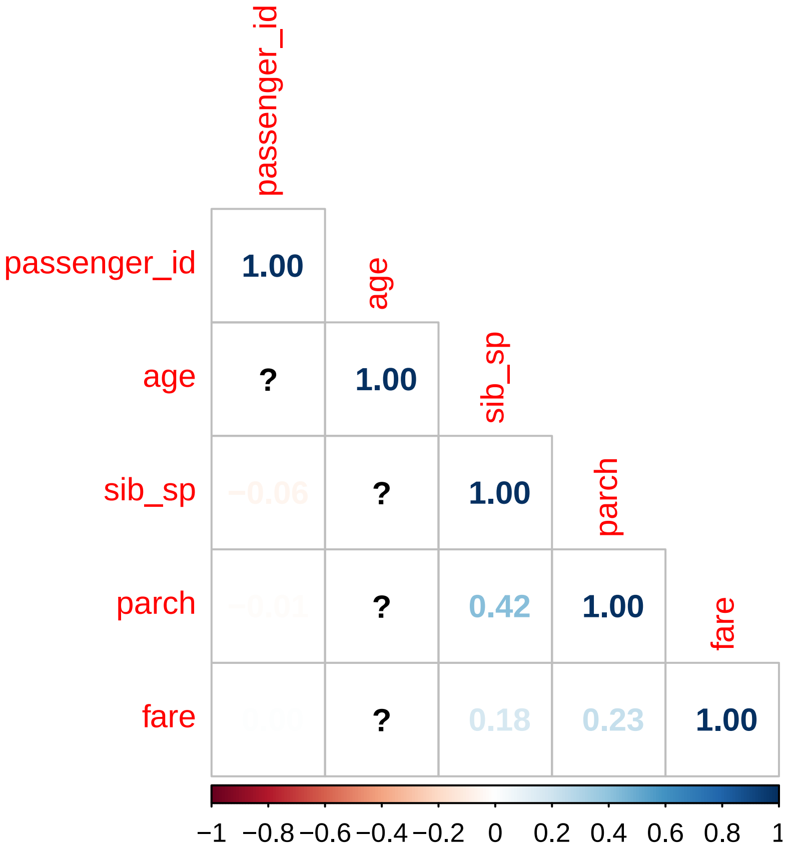

```
print(c(sum(data_train$survived == 'Yes')/length(data_train$survived),
        sum(data_train$survived == 'No')/length(data_train$survived)))
```

```
## [1] 0.3836276 0.6163724
```

Survivors are slightly less than non-survivors.

Question 3

```
M = cor(data_train %>% dplyr::select(where(is.numeric)))
corrplot(M, method = "number", type = "lower")
```



'parch', 'sib_sp' and 'fare', 'parch' variables are negatively correlated. 'age' NA have to be deal.

Question 4

```
titanic_recipe = recipe(survived ~ pclass + sex + age + sib_sp + parch + fare, data = data_train) %>%
  step_impute_linear(age) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(terms = ~ 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, data_train)
```

Question 6

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

```
lda_wkflow = workflow() %>%
  add_model(lda_mod) %>%
  add_recipe(titanic_recipe)

lda_fit = fit(lda_wkflow, data_train)
```

Question 7

```
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, data_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, data_train)
```

Question 9

```
r1 = predict(log_fit, new_data = data_train, type = "prob")
r2 = predict(lda_fit, new_data = data_train, type = "prob")
r3 = predict(qda_fit, new_data = data_train, type = "prob")
r4 = predict(nb_fit, new_data = data_train, type = "prob")

train_results = bind_cols(r1, r2, r3, r4)
```

```
## 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`
## * `.pred_No` -> `.pred_No...6`
## * `.pred_Yes` -> `.pred_Yes...7`
## * `.pred_No` -> `.pred_No...8`
```

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

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

qda_acc = augment(qda_fit, new_data = data_train) %>%
  accuracy(truth = survived, estimate = .pred_class)

nb_acc = augment(nb_fit, new_data = data_train) %>%
  accuracy(truth = survived, estimate = .pred_class)

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      0.812 Logistic Regression
## 2      0.801 LDA
## 3      0.785 QDA
## 4      0.770 Naive Bayes
```

Logistic Regression is the highest training accuracy.

Question 10

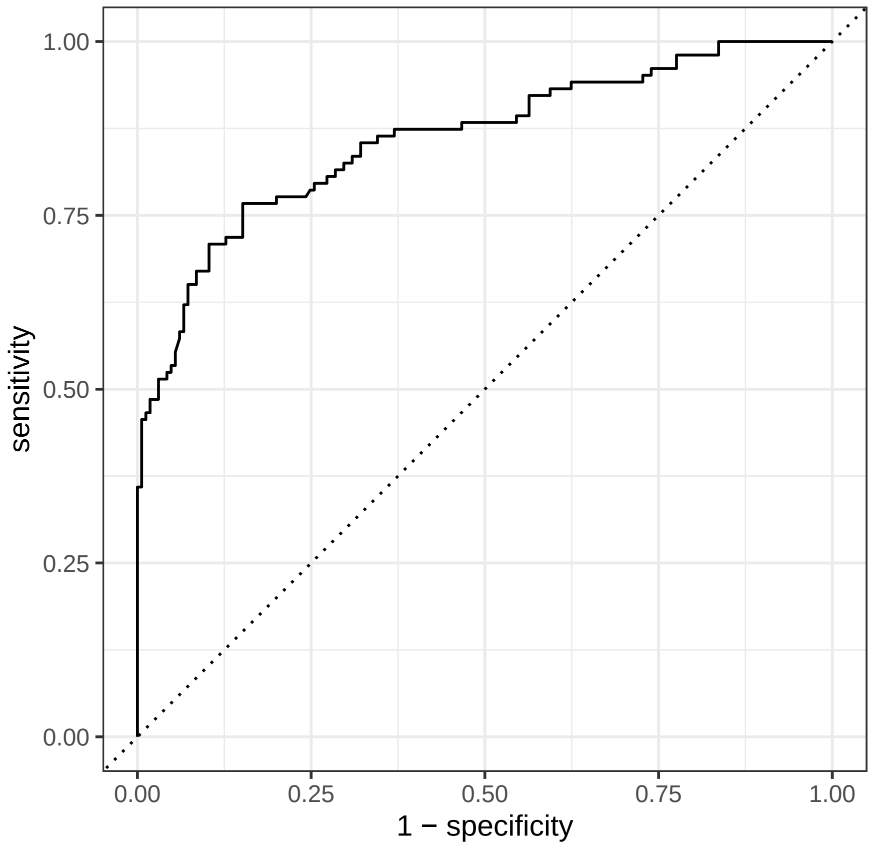
```r
#predict(log_fit, new_data = data_test, type = "prob")

multi_metric <- metric_set(accuracy, sensitivity, specificity)
augment(log_fit, new_data = data_test) %>%
  multi_metric(truth = survived, estimate = .pred_class)
```

```
## # A tibble: 3 x 3
##   .metric     .estimator .estimate
##   <chr>       <chr>          <dbl>
## 1 accuracy    binary         0.821
## 2 sensitivity binary         0.709
## 3 specificity binary         0.891
```

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

```
##           Truth
## Prediction Yes  No
##        Yes  73  18
##        No   30 147
```

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




```r
augment(log_fit, new_data = data_test) %>%
  roc_auc(truth = survived, estimate = .pred_Yes)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.861
```

The test accuracy of logistic model is 0.82. It is quite a similar result as the train set. It could be consider that there is no over-fit issue on the train set and fitted well to the test set.

Question 11

$p = \frac{e^z}{1+e^z}$
$= 1 - \frac{1}{1+e^z}$
$1 - p = \frac{1}{1+e^z}$
$1 + e^z = \frac{1}{1-p}$
$e^z = \frac{1}{1-p} - \frac{1-p}{1-p}$
$e^z = \frac{p}{1-p}$
$z = ln(\frac{p}{1-p})$

Question 12

$ln(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1$
$ln(\frac{p(y=1)}{p(y=0)}) = \beta_0 + \beta_1 x_1$

$ln(\frac{p(y=1|x_1)}{p(y=0|x_1)}) = \beta_0 + \beta_1 x_1$
$ln(\frac{p(y=1|x_1+2)}{p(y=0|x_1+2)}) = \beta_0 + \beta_1 (x_1 + 2)$

$ln(\frac{p(y=1|x_1+2)}{p(y=0|x_1+2)}) - ln(\frac{p(y=1|x_1)}{p(y=0|x_1)}) = 2\beta_1$

The odds ratio will increase as $e^{2\beta_1}$ if x_1 increase by two.

When β_1 is negative, and if x_1 approaches ∞, p approaches to 0. However, if x_1 approaches $-\infty$, p approaches to 1.