Homework 3

Yixin Wang

Contents

Classification

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`!

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

```
library(tidyverse)
library(tidymodels)
library(klaR)
library(corrplot)
library(discrim)
library(poissonreg)
library(corrr)
tidymodels_prefer()

set.seed(3435)

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

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.

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

```
titanic_split <- titanic %>%
  initial_split(strata = survived, prop = 0.7)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
dim(titanic_train)
```

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

```
dim(titanic_test)
```

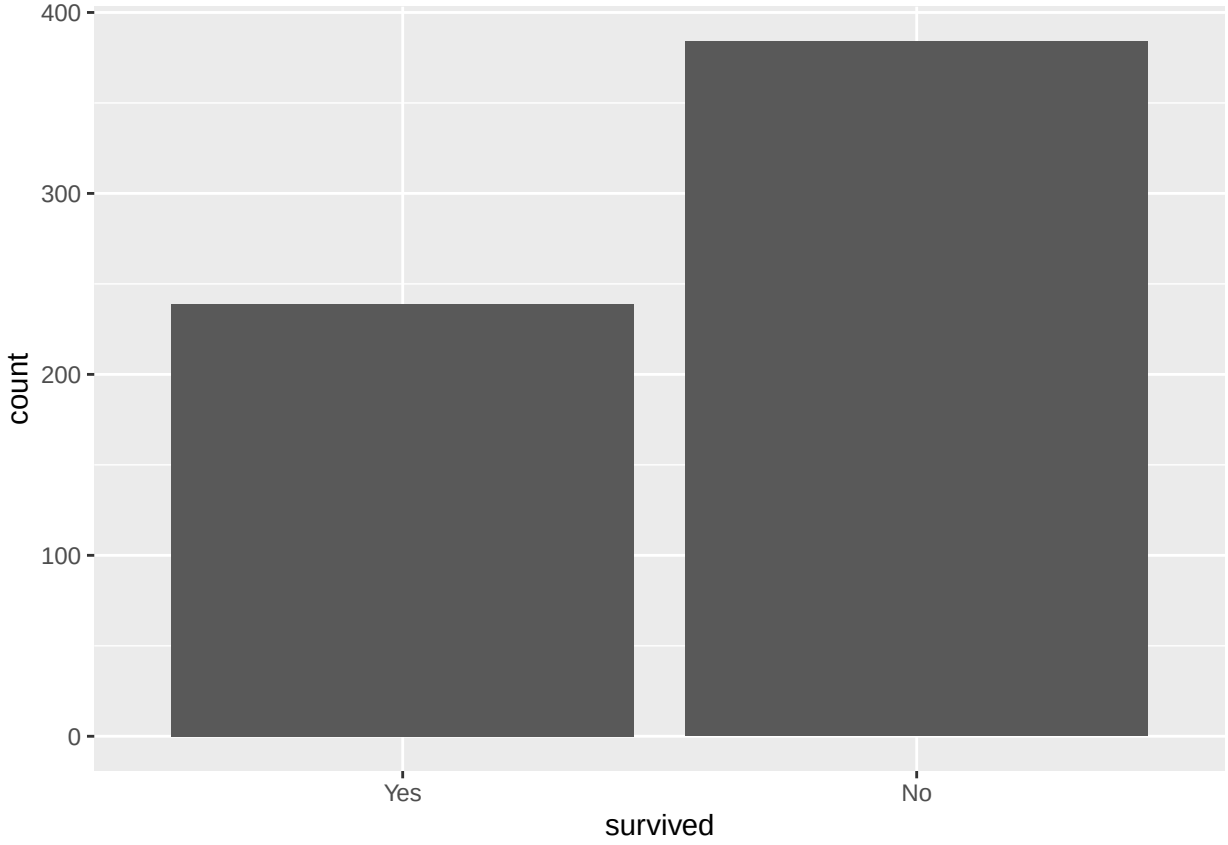
```
## [1] 268  12
```

Each dataset has approximately the right observations. 623 is approximately 70% of the data set, which contains 891 samples. It is a good idea to use stratified sampling because the outcome is imbalanced.

Question 2

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

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

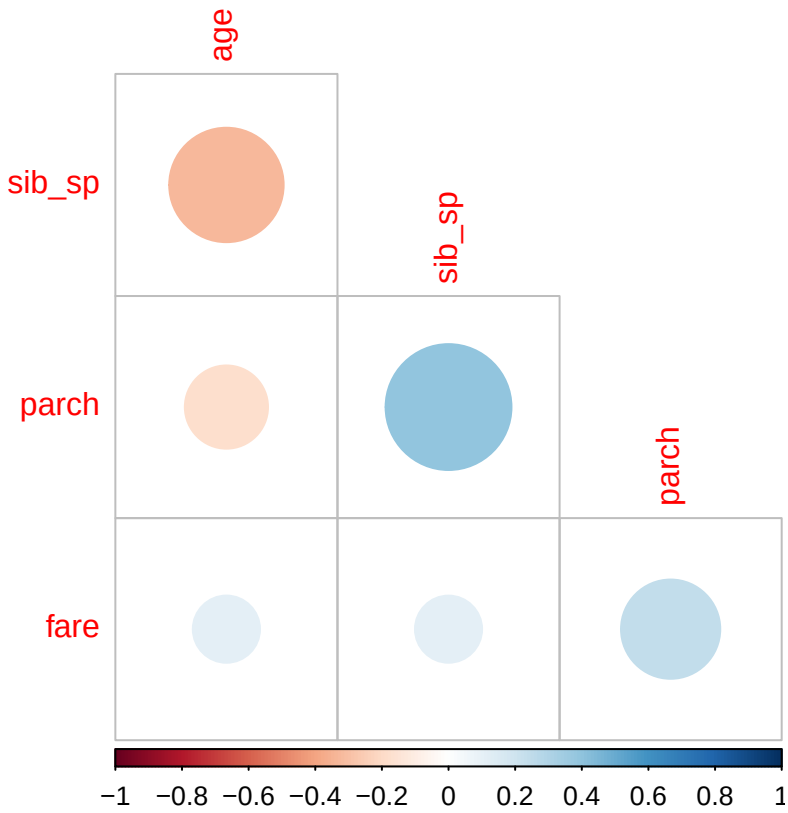


There is an imbalance between two outcomes,but not a lot.

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?

```
titanic_train %>%
  select(is.numeric, -passenger_id) %>%
  cor(use = "complete.obs") %>%
  corrplot(type = "lower", diag = FALSE)
```



`age` and `sib_sp` are negatively correlated. It shows that older passengers are less likely to have siblings on board. `sib_sp` and `parch` are positively correlated, which means that having one parent or children on board are more likely to have another parent or children on board.

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.

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

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

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

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

Question 6

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

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

lda_workflow <- workflow() %>%
  add_model(lda_mod) %>%
  add_recipe(titanic_recipe)

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

Question 7

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

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

qda_workflow <- workflow() %>%
  add_model(qda_mod) %>%
  add_recipe(titanic_recipe)

qda_fit <- fit(qda_workflow, 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`.

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

nb_workflow <- workflow() %>%
  add_model(nb_mod) %>%
  add_recipe(titanic_recipe)

nb_fit <- fit(nb_workflow, 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?

```
log_acc <-predict(log_fit, new_data = titanic_train, type = "class") %>%
  bind_cols(titanic_train %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)
lda_acc <- predict(lda_fit, new_data = titanic_train, type = "class") %>%
  bind_cols(titanic_train %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)
qda_acc <- predict(qda_fit, new_data = titanic_train, type = "class") %>%
  bind_cols(titanic_train %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)
nb_acc <- predict(nb_fit, new_data = titanic_train, type = "class") %>%
  bind_cols(titanic_train %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)

results <- bind_rows(log_acc, lda_acc, qda_acc, nb_acc) %>%
  tibble() %>% mutate(model = c("Logistic", "LDA", "QDA", "NB")) %>%
  select(model, .estimate) %>%
  arrange(.estimate)
results
```

```
## # A tibble: 4 x 2
##   model    .estimate
##   <chr>        <dbl>
## 1 NB           0.788
## 2 QDA          0.795
## 3 Logistic     0.798
## 4 LDA          0.809
```

The logistic model performs the best.

Question 10

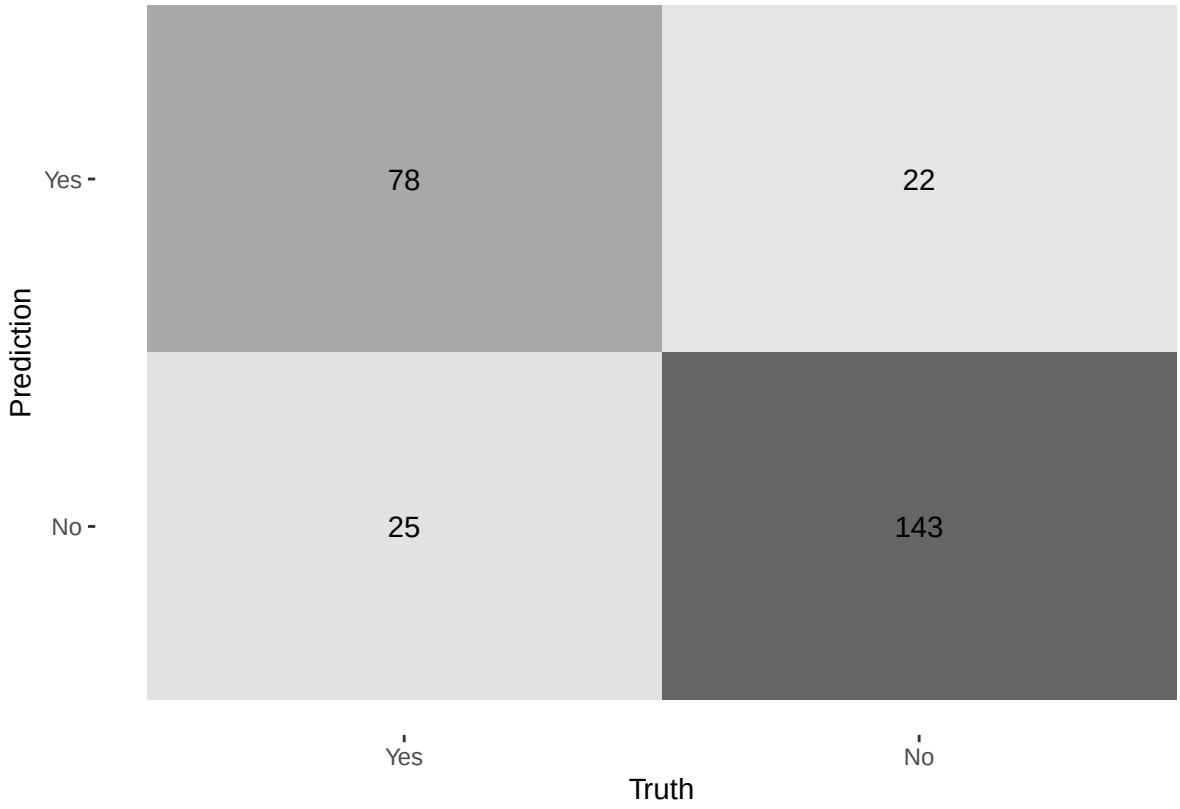
Fit the model with the highest training accuracy to the **testing** data. Report the accuracy of the model on the **testing** data.

```
lda_test <- fit(lda_workflow, titanic_test)
predict(lda_test, new_data = titanic_test, type = "class") %>%
  bind_cols(titanic_test %>% select(survived)) %>%
  accuracy(truth = survived, estimate = .pred_class)
```

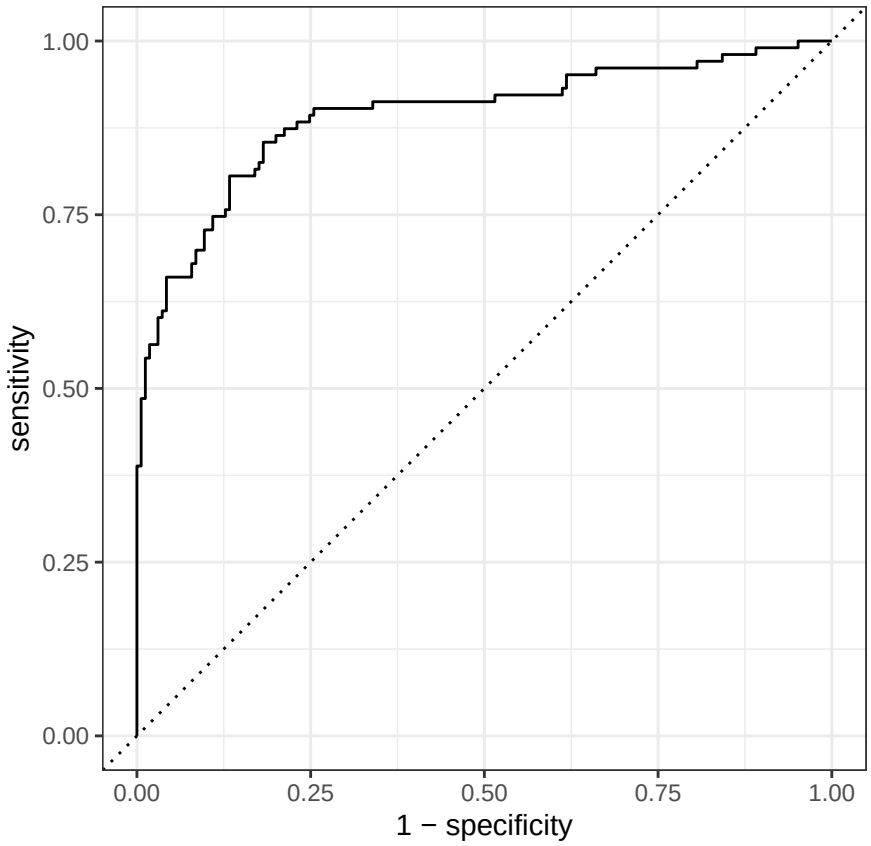
```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.825
```

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

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




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




```
augment(lda_test, new_data = titanic_test) %>%
  roc_auc(survived, .pred_Yes)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.893
```

The area under curve is approximately 0.869.

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

The model did well in predicting the survival of titanic population. The accuracy is increasing on the testing data. It is because model has low variance.