Homework Assignment 3

Vasyl Ostapenko (774 970 8)

April 12, 2022

Load Data

```
DATA_FOLDER = "./data"
IMAGES_FOLDER = "./images"
TITANIC_FNAME = file.path(DATA_FOLDER, "titanic.csv")
data = read.csv(TITANIC_FNAME)
data$survived = as.factor(data$survived)
data$pclass = as.factor(data$pclass)
```

Question 1

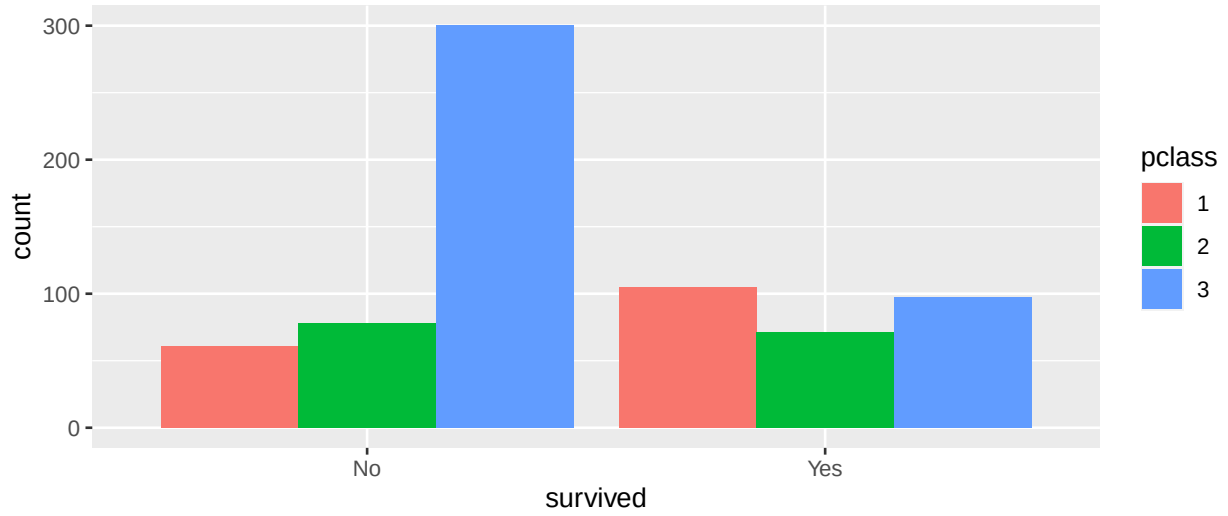
```
titanic_split = data %>%
  initial_split(prop=0.8, strata="survived")

titanic_train = training(titanic_split)
titanic_test = testing(titanic_split)
print(sapply(titanic_train, function(x) sum(is.na(x))))
```

```
## passenger_id     survived       pclass         name          sex          age
##            0            0            0            0            0          140
##       sib_sp        parch       ticket         fare        cabin     embarked
##            0            0            0            0          554            2
```

It is a good idea to use stratified sampling for the data to ensure that we have proper representation of both classes in the training and test splits.

Question 2

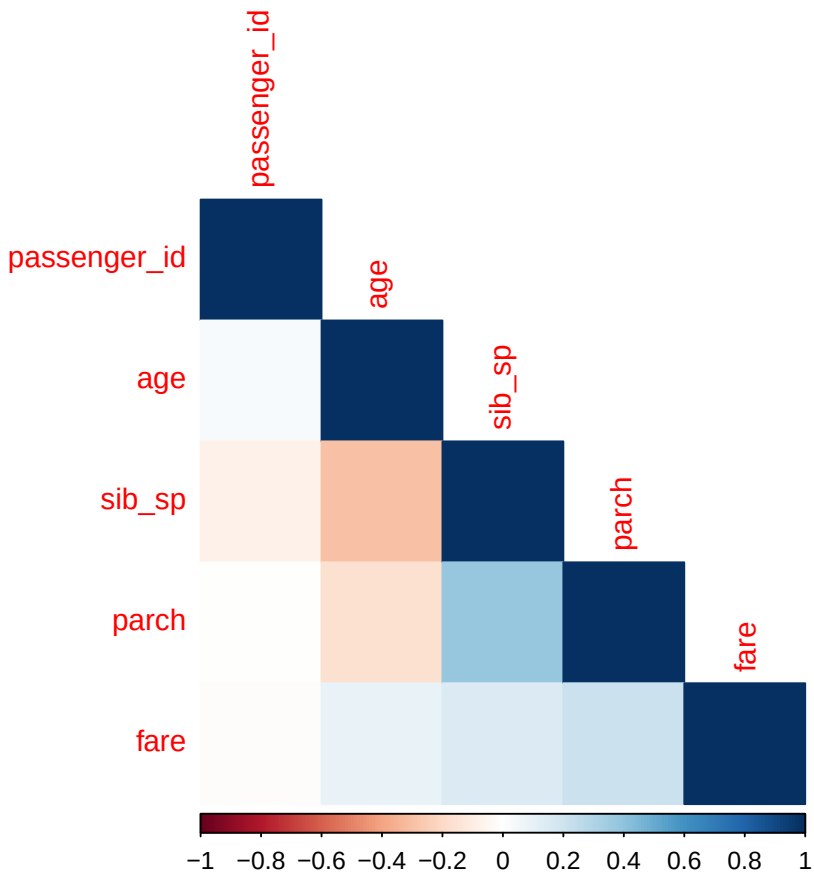
```
ggplot(titanic_train, aes(x=survived)) +
  geom_bar(aes(fill=pclass), position="dodge")
```



There were more people who did not survive in general. Out of those who did not survive, the majority were 3rd class passengers. Out of those who did survive, the majority were 1st class passengers.

Question 3

```
titanic_train2 = titanic_train[ , !(names(titanic_train) %in% c("cabin", "embarked"))] %>% copy()
titanic_train2 = titanic_train2 %>% drop_na()
corrplot(cor(titanic_train2[ , sapply(titanic_train2, is.numeric)]),
         method="color", type="lower")
```



There is some positive correlation between the parch and sib_sp variables. This makes sense, because if there are more children there are more siblings and if there are more parents there are more spouses. There is also some negative correlation between the sib_sp and age variables. This also makes sense, because if a passenger has parents and siblings with them, they will probably tend to be younger.

Question 4

```
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( ~ starts_with("sex"):fare) %>%
  step_interact( ~ age:fare)
```

Question 5

```
glm_model = logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

glm_workflow = workflow() %>%
  add_model(glm_model) %>%
  add_recipe(titanic_recipe)

glm_fit = glm_workflow %>%
```

```
  fit(titanic_train)
```

Question 6

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

lda_workflow = workflow() %>%
  add_model(lda_model) %>%
  add_recipe(titanic_recipe)

lda_fit = lda_workflow %>%
  fit(titanic_train)
```

Question 7

```
qda_model = discrim_quad() %>%
  set_engine("MASS") %>%
  set_mode("classification")

qda_workflow = workflow() %>%
  add_model(qda_model) %>%
  add_recipe(titanic_recipe)

qda_fit = qda_workflow %>%
  fit(titanic_train)
```

Question 8

```
nb_model = naive_Bayes() %>%
  set_engine("klaR", usekernel=FALSE) %>%
  set_mode("classification")

nb_workflow = workflow() %>%
  add_model(nb_model) %>%
  add_recipe(titanic_recipe)

nb_fit = nb_workflow %>%
  fit(titanic_train)
```

Question 9

```
bound_train_data = bind_cols(predict(glm_fit, titanic_train),
                             predict(lda_fit, titanic_train),
                             predict(qda_fit, titanic_train),
                             predict(nb_fit, titanic_train),
                             titanic_train$survived)
colnames(bound_train_data) = c("GLM Predict", "LDA Predict", "QDA Predict",
                               "NB Predict", "True")
```

```
print(accuracy(bound_train_data,
               truth="True", estimate="GLM Predict")$.estimate)
```

```
## [1] 0.8104
```

```
print(accuracy(bound_train_data,
               truth="True", estimate="LDA Predict")$.estimate)
```

```
## [1] 0.7978
```

```
print(accuracy(bound_train_data,
               truth="True", estimate="QDA Predict")$.estimate)
```

```
## [1] 0.7823
```

```
print(accuracy(bound_train_data,
               truth="True", estimate="NB Predict")$.estimate)
```

```
## [1] 0.7795
```

The logistic regression model achieved the highest accuracy on the training set, scoring about 81%.

Question 10

```
bound_test_data = bind_cols(predict(glm_fit, titanic_test),
                            titanic_test$survived)
colnames(bound_test_data) = c("GLM Predict", "True")
print(accuracy(bound_test_data,
               truth="True", estimate="GLM Predict")$.estimate)
```

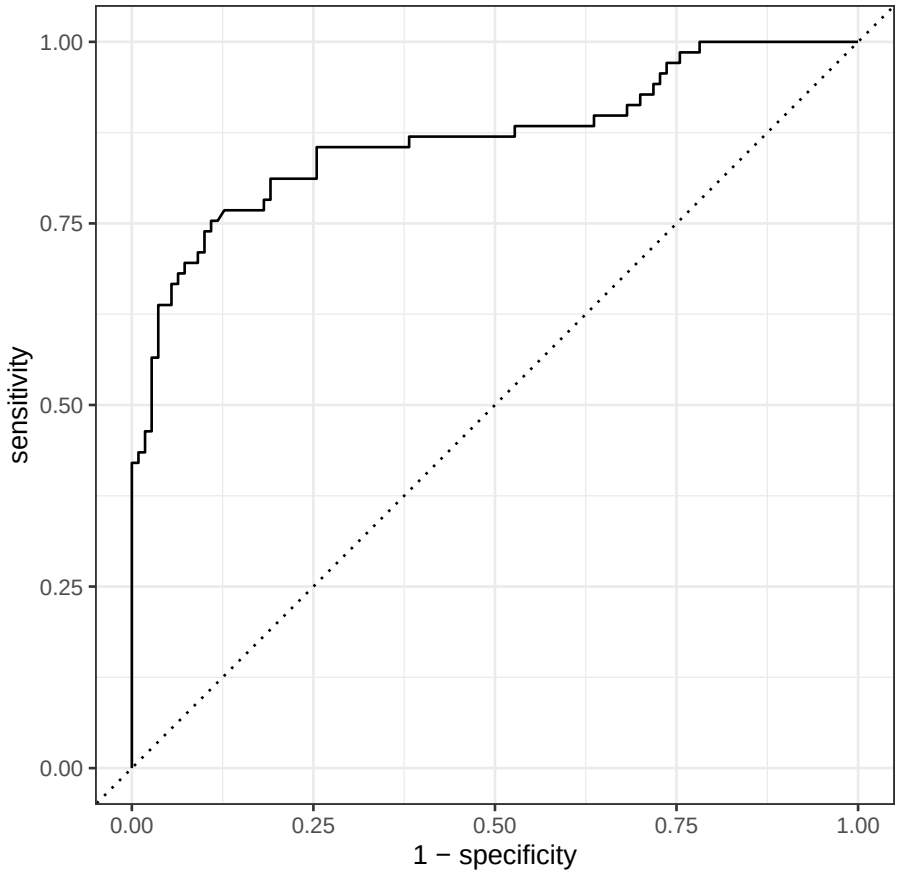
```
## [1] 0.838
```

The logistic regression model achieved about 84% accuracy on the test set.

```
conf_mat(bound_test_data, truth="True", estimate="GLM Predict")
```

```
##           Truth
## Prediction  No Yes
##        No  104  23
##        Yes   6  46
```

```
roc = glm_fit %>%
  predict(new_data=titanic_test, type="prob") %>%
  bind_cols(titanic_test) %>%
  roc_curve(survived, .pred_Yes, event_level="second")
autoplot(roc)
```




```
auc = glm_fit %>%
  predict(new_data=titanic_test, type="prob") %>%
  bind_cols(titanic_test) %>%
  roc_auc(survived, .pred_Yes, event_level="second")
print(auc$.estimate)
```

```
## [1] 0.8673
```

The model performed OK for very limited feature engineering and no hyperparameter optimization. We got around 81% train and 84% test accuracy respectively. The values differ due to randomness in the data as well as the train/test split that was done.