Homework 3 - PSTAT131

Kabir Snell 6342786

Contents


```
library(tidyverse)
library(tidymodels)
library(ggcorrplot)
library(parsnip)
library(discrim)
```

Preproccessing

```
# Reading in the data
titanic <- read.csv("data/titanic.csv")

# Changing survived, pclass and embarked to factors
titanic$survived <- as.factor(titanic$survived)
titanic$pclass <- as.factor(titanic$pclass)
titanic$embarked <- as.factor(titanic$embarked)
titanic$sex <- as.factor(titanic$sex)

set.seed(727)
```

Question 1

```
# Splitting the data
titanic_split <- initial_split(titanic, prop = .80, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)

titanic_split
```

```
## <Training/Testing/Total>
## <712/179/891>
```

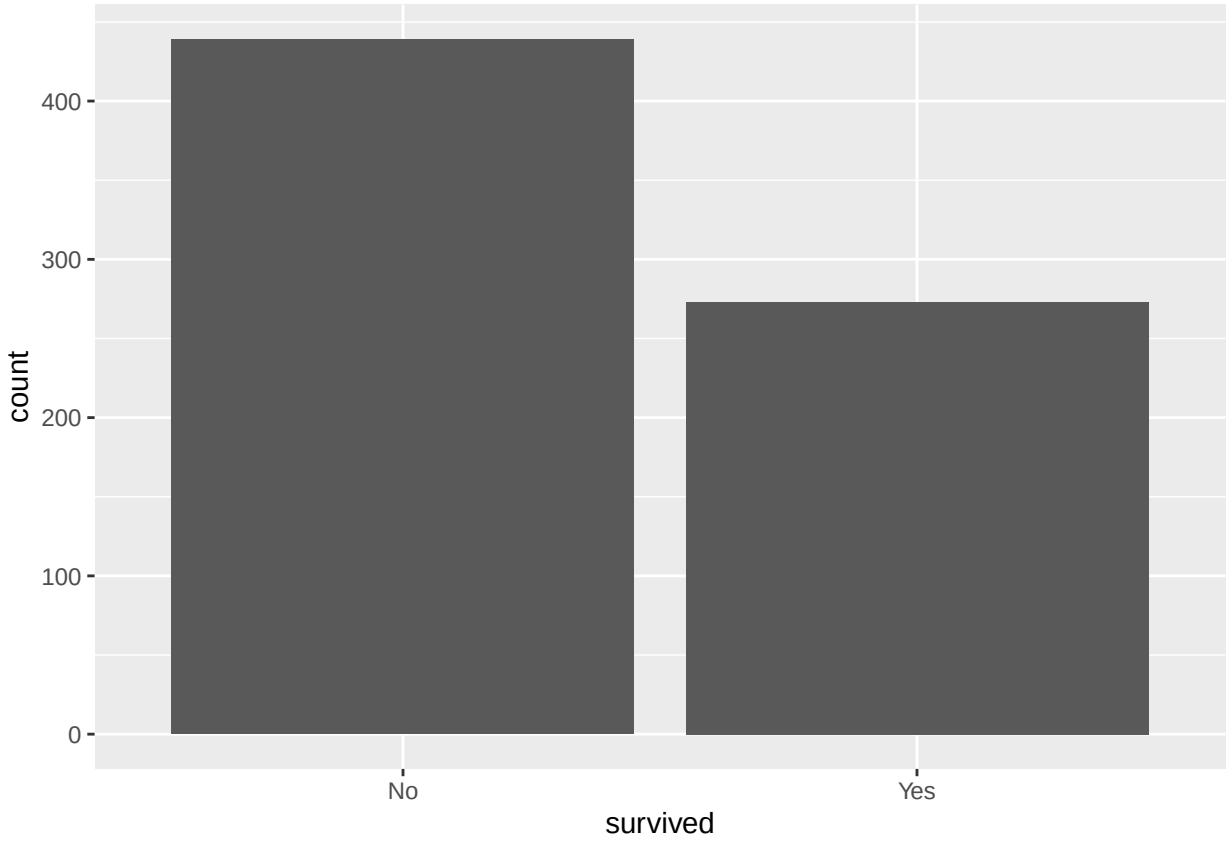
The training data set has 712 observations and the testing data set has 179 observations.

As for missing values, many cabin numbers are missing. Additionally, some of the ages of the passengers are also missing.

Stratified sampling is a good idea for this sample because it captures key population characteristics in the sample. This is similar to a weighted average, as this method of sampling produces characteristics in the sample that are proportional to the overall population

Question 2

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

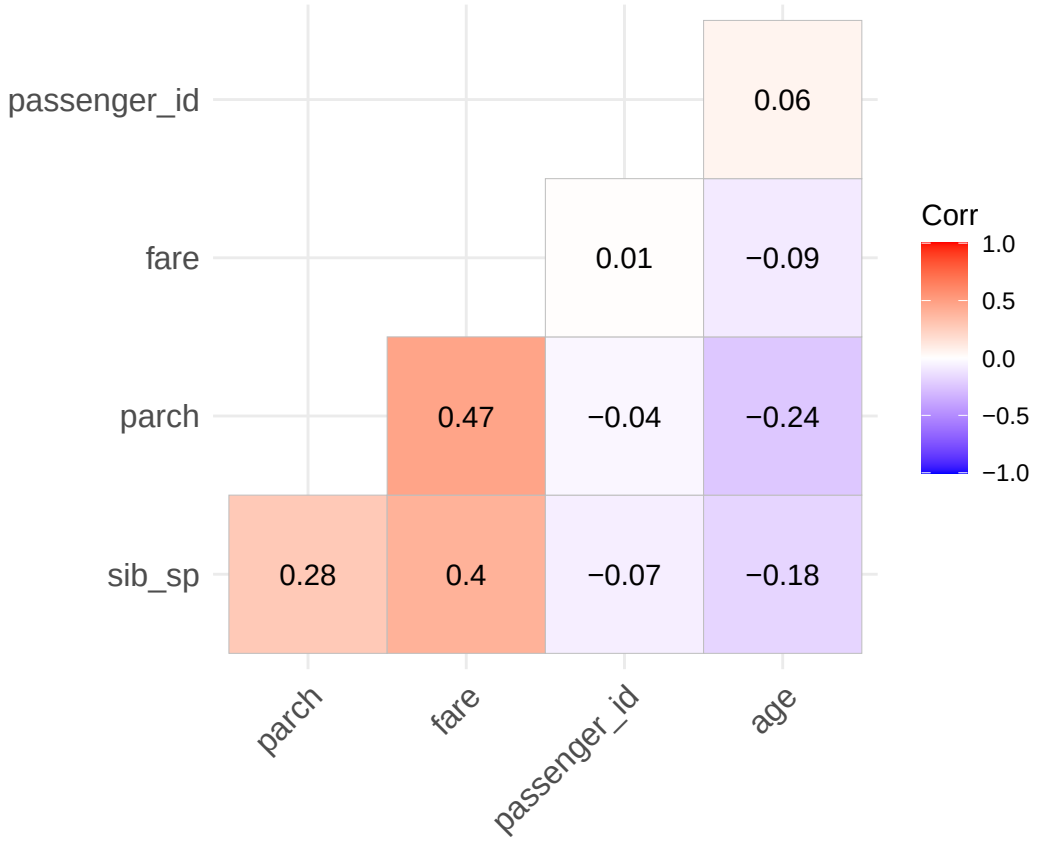


The majority of passengers did not survive. The ratio of passengers who did not survive compared to those who did survive is a little bit less than 2:1

Question 3

```
corr <- titanic_train %>%
        drop_na() %>%
        select(where(is.numeric)) %>%
        correlations()

ggcorrplot(corr, hc.order = TRUE, type="lower", lab=TRUE)
```



The two most correlated variables are {fare, parch}, and {fare, sib_sp}. This would follow intuitive thinking as you would expect fare to increase with the number of passengers on board. Age also was negatively correlated with parch, which makes sense as you would expect younger children to have more parents with them on board. This could be useful for rows with missing age values as it could give us some sort of indication of general age value.

Question 4

```
# Creating a recipe
titanic_recipe <- recipe(survived ~ pclass + sex + age + sib_sp + parch + fare, data = titanic_train) %
  step_impute_linear(age) %>%
  step_dummy(all_nominal_predictors()) %>%
  prep() %>%
  step_interact(~ age:fare) %>%
  step_interact(~ sex_male:fare)

titanic_recipe$term_info
```

```
## # A tibble: 8 x 4
##   variable  type    role      source
##   <chr>     <chr>   <chr>     <chr>
## 1 age       numeric predictor original
## 2 sib_sp    numeric predictor original
## 3 parch     numeric predictor original
```

```
## 4 fare      numeric predictor original
## 5 survived  nominal outcome   original
## 6 pclass_X2 numeric predictor derived
## 7 pclass_X3 numeric predictor derived
## 8 sex_male  numeric predictor derived
```

Question 5

```
# Linear Regression Model
log_mod <- logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

log_wflow <- workflow() %>%
  add_model(log_mod) %>%
  add_recipe(titanic_recipe)

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

Question 6

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

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

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

Question 7

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

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

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

Question 8

```
# Naive Bayes model
nba_mod <- naive_Bayes() %>%
  set_mode("classification") %>%
  set_engine("klaR") %>%
  set_args(usekernel = FALSE)

nba_wflow <- workflow() %>%
  add_model(nba_mod) %>%
  add_recipe(titanic_recipe)

nba_fit <- fit(nba_wflow, titanic_train)
```

Question 9

```
# Fitting the models
# Logistic Regression
results <- bind_cols(titanic_train$survived ,c(predict(log_fit, titanic_train)))
log_acc <- accuracy(results, truth = ...1, estimate = .pred_class)
log_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.816
```

```
# LDA
results <- bind_cols(titanic_train$survived ,c(predict(lda_fit, titanic_train)))
lda_acc <- accuracy(results, truth = ...1, estimate = .pred_class)
lda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.792
```

```
# QDA
results <- bind_cols(titanic_train$survived ,c(predict(qda_fit, titanic_train)))
qda_acc <- accuracy(results, truth = ...1, estimate = .pred_class)
qda_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.775
```

```r
# Naive Bayes
results <- bind_cols(titanic_train$survived ,c(predict(nba_fit, titanic_train)))
nba_acc <- accuracy(results, truth = ...1, estimate = .pred_class)
nba_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.768
```

```r
accuracyResults <- data.frame("Logistic" = log_acc$.estimate, "LDA" = lda_acc$.estimate,
                              "QDA" = qda_acc$.estimate, "Bayes" = nba_acc$.estimate)
accuracyResults
```

```
##    Logistic       LDA       QDA     Bayes
## 1 0.8160112 0.7921348 0.7752809 0.7682584
```

The logistic regression model achieved the highest accuracy on the training data

Question 10

```r
# Applying the logistic regression model to the test set
results <- bind_cols(titanic_test$survived, c(predict(log_fit, titanic_test)))
```

```
## New names:
## * `` -> `...1`
```

```r
log_acc <- accuracy(results, truth = ...1, estimate = .pred_class)
log_acc
```

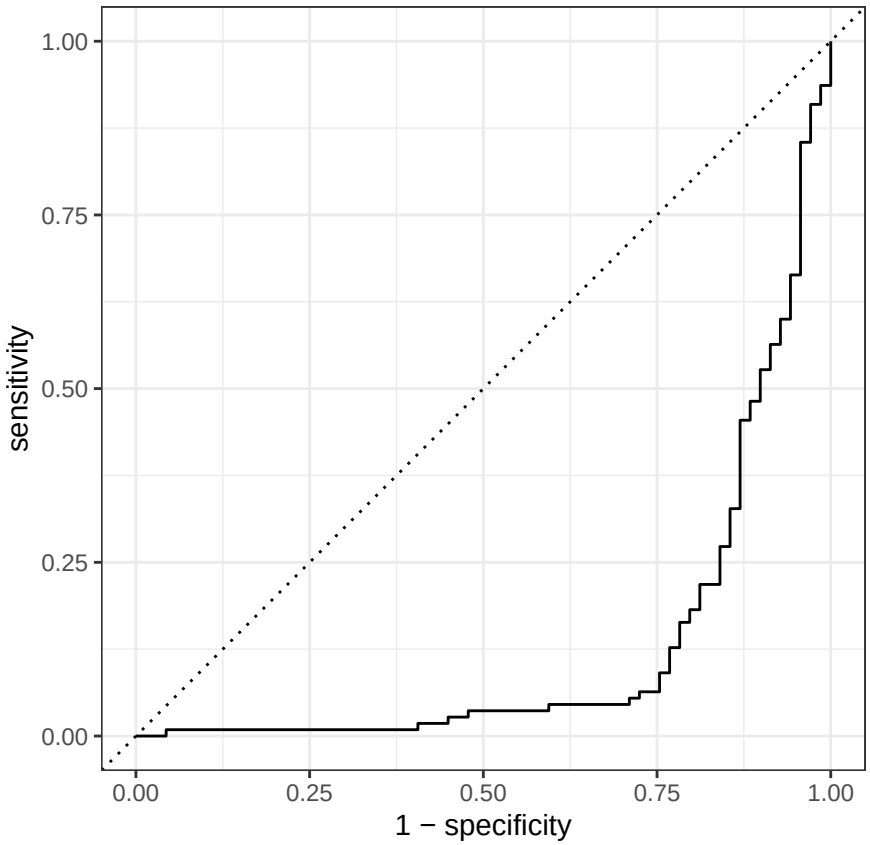
```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.838
```

The results show that the logistic regression model was ~83.799% accurate (surprisingly higher accuracy than it had on the training data)

```r
# Confusion Matrix
augment(log_fit, new_data = titanic_test) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction No Yes
##        No  96  15
##        Yes 14  54
```

```
# ROC Curve
augment(log_fit, new_data = titanic_test) %>%
  roc_curve(survived, .pred_Yes) %>%
  autoplot()
```




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

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.127
```

The testing and training accuracy metric did differ slightly. Surprisingly, the accuracy measurement on the testing data set was about 2% better than that of the training data set. One reason for this could just be luckiness with outliers in the testing and training data sets as I don't expect this result to repeat itself.