Homework 3

PSTAT 131/231

Contents

Classification

For this assignment, we will be working with part of a Kaggle data set that was the subject of a machine learning competition and is often used for practicing ML models. The goal is classification; specifically, to predict which passengers would survive the Titanic shipwreck.

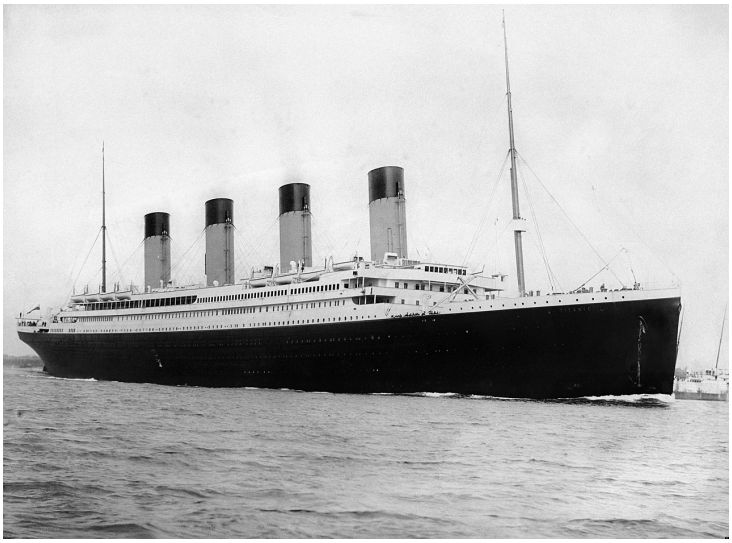


Figure 1: Fig. 1: RMS Titanic departing Southampton on April 10, 1912.

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.

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

# preprocessing
titanic$survived <- factor(titanic$survived, levels = c("Yes","No"))
titanic$pclass <- as.factor(titanic$pclass)
```

```r
set.seed(42069) # for reproducibility
```

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?

Answer:

```r
# count total occurences of values in survived column
table(titanic$survived)
```

```
##
## Yes  No
## 342 549
```

There are an unequal proportion of "Yes" and "No"'s, so it's a good idea to stratify the data so that the training data has a representative number of Yes's and the model does not artifically predict No more often because of the poor training data.

```r
titanic_split <- initial_split(titanic,prop=0.8,
                               strata=survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)

nrow(titanic_train)/nrow(titanic)
```

```
## [1] 0.7991021
```

```r
nrow(titanic_test)/nrow(titanic)
```

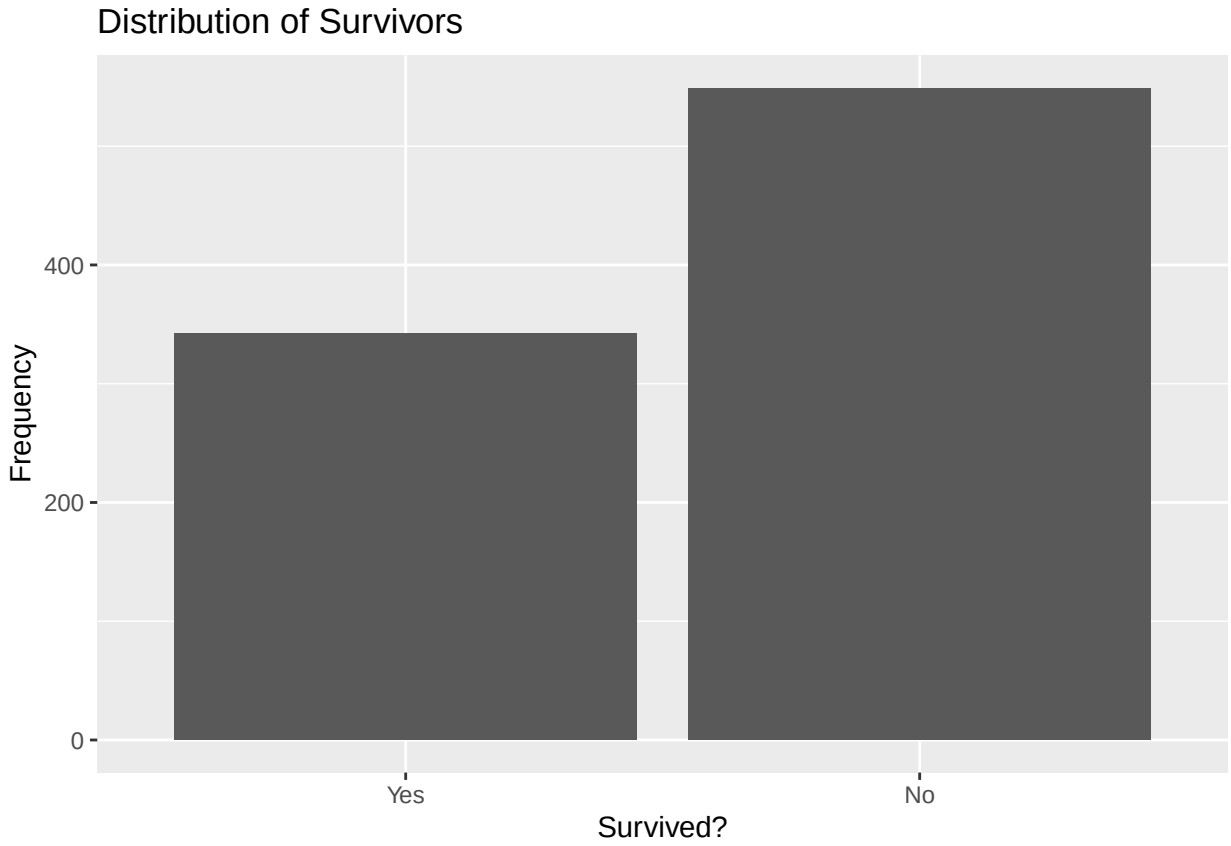
```
## [1] 0.2008979
```

We can notice that almost all entries in the `cabin` column are `NA`, and there are a significant number of `NA`'s in the `age` column as well.

Question 2

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

```r
titanic %>%
  ggplot(aes(x = survived))+
  geom_bar()+
  ggtitle("Distribution of Survivors")+
  labs(x="Survived?",y="Frequency")
```



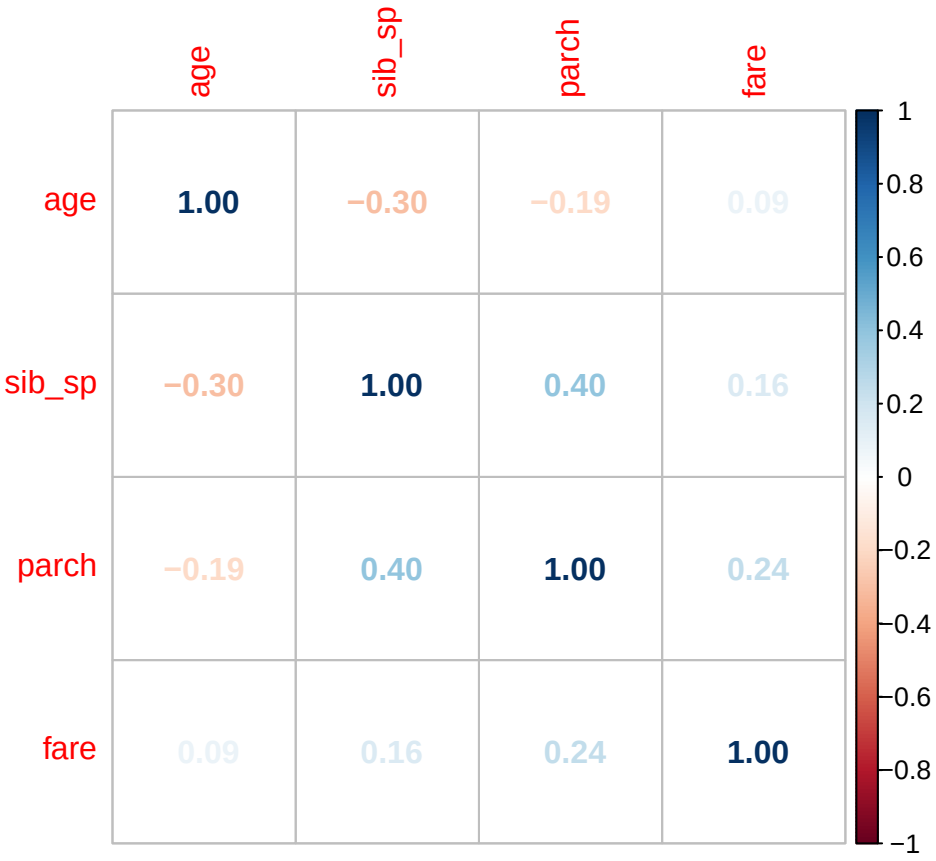
We have that approximately 38% of the passenger survived, so we have that `survived` is binomially distributed with $p \approx 0.38$.

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?

```
corr_mat <- titanic_train%>%
  select(c(age,sib_sp,parch,fare))%>%
  na.omit()%>%
  cor()

corr_mat%>%
  corrplot(method="number")
```



Overall, the predictors are not strongly linearly correlated. The strongest correlation is between `sib_sp` and `parch`. So the number of siblings or spouses a passenger had is positively correlated with the number of parents or children the passenger had. This could make sense as many people probably attended the cruise as entire families.

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.

```
recipe_data<-titanic_train%>%select(c(survived,pclass,sex,age,sib_sp,parch,fare))
titanic_recipe <- recipe(survived ~ ., data = recipe_data)%>%
  step_impute_linear(age)%>%
  step_dummy(all_nominal_predictors())%>%
  step_interact(terms = ~starts_with("sex"):fare + age:fare)%>%
  step_center(all_predictors())%>%
  step_scale(all_predictors())
```

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

log_fit <- fit(log_wkflow, 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_mode("classification") %>%
  set_engine("MASS")

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

lda_fit <- fit(lda_wkflow, 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_mode("classification") %>%
  set_engine("MASS")

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

qda_fit <- fit(qda_wkflow, 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_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, 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_train_res <- predict(log_fit, new_data = titanic_train)%>%
  bind_cols(titanic_train%>%select(survived))
lda_train_res <- predict(lda_fit, new_data = titanic_train)%>%
  bind_cols(titanic_train%>%select(survived))
qda_train_res <- predict(qda_fit, new_data = titanic_train)%>%
  bind_cols(titanic_train%>%select(survived))
nb_train_res <- predict(nb_fit, new_data = titanic_train)%>%
  bind_cols(titanic_train%>%select(survived))

log_acc <- log_train_res %>%
  accuracy(truth = survived, estimate = .pred_class)

lda_acc <- lda_train_res %>%
  accuracy(truth = survived, estimate = .pred_class)

qda_acc <- qda_train_res %>%
  accuracy(truth = survived, estimate = .pred_class)

nb_acc <- nb_train_res %>%
  accuracy(truth = survived, estimate = .pred_class)

accuracies <- c(log_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      0.817 Logistic Regression
## 2      0.798 LDA
## 3      0.782 QDA
## 4      0.768 Naive Bayes
```

Question 10

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

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

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

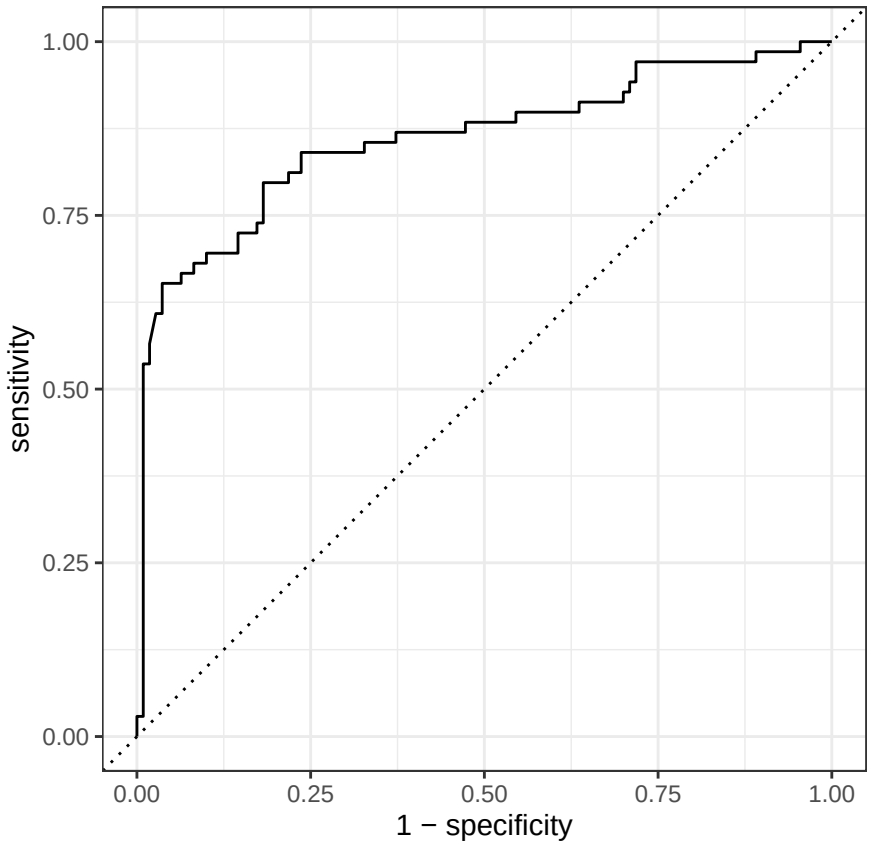
```
log_test_acc <- predict(log_fit, new_data = titanic_test)%>%
  bind_cols(titanic_test%>%select(survived))%>%
  accuracy(truth = survived, estimate = .pred_class)
log_test_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.821
```

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

```
##           Truth
## Prediction Yes  No
##        Yes  46   9
##        No   23 101
```

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




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

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.859
```

The accuracy of the training data was actually slightly higher than that of the testing data, but overall, they were basically the same.

Required for 231 Students

In a binary classification problem, let p represent the probability of class label 1, which implies that $1 - p$ represents the probability of class label 0. The *logistic function* (also called the "inverse logit") is the cumulative distribution function of the logistic distribution, which maps a real number z to the open interval $(0, 1)$.

Question 11

Given that:

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

Prove that the inverse of a logistic function is indeed the *logit* function:

$$z(p) = ln\left(\frac{p}{1-p}\right)$$

Answer: Note that $e^z \neq 0$ for all $z \in \mathbb{R}$, so we can divide the numerator and denominator by e^z and see

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

Now to show that p and z are inverses, we need to show that $p(z(x)) = z(p(x)) = x$. We also note that the domain of z is $(0, 1)$ and the domain of p is $\mathbb{R}$, while the codomain of z is $\mathbb{R}$ and the codomain of p is $(0, 1)$, so we are off to a good start. Now we have that

$$p(z(x)) = \frac{1}{1 + \exp\left(-\ln\left(\frac{x}{1-x}\right)\right)} = \frac{1}{1 + \frac{1-x}{x}} = \frac{1}{\frac{1}{x}} = x.$$

Furthermore, we see

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

so z and p are indeed inverses.

Question 12

Assume that $z = \beta_0 + \beta_1 x_1$ and $p = logistic(z)$. How do the odds of the outcome change if you increase x_1 by two? Demonstrate this.

Assume now that β_1 is negative. What value does p approach as x_1 approaches ∞? What value does p approach as x_1 approaches $-\infty$?

Answer: We have that

$$p(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-\beta_0 - \beta_1 x_1}} = \frac{e^{\beta_0}}{e^{\beta_0} + e^{-\beta_1 x_1}}$$

which means that

$$\frac{p(z)}{1-p(z)} = \frac{\frac{1}{1+e^{-\beta_0-\beta_1 x_1}}}{1-\frac{1}{1+e^{-\beta_0-\beta_1 x_1}}} = e^{\beta_0+\beta_1 x_1} = e^{\beta_0} e^{\beta_1 x_1}.$$

So if we have that $x_1 \to x_1 + 2$, then $e^{\beta_0} e^{\beta_1 x_1} \to e^{\beta_0} e^{\beta_1 (x_1+2)} = e^{\beta_0} e^{2\beta_1} e^{\beta_1 x_1}$. Therefore, if x_1 increases by 2, then the odds of the outcome get scaled by $e^{2\beta_1}$.

For the second question, if β_1 is negative, then

$$\lim_{x_1 \to \infty} e^{-\beta_0-\beta_1 x_1} = \infty,$$

so it follows that

$$\lim_{x_1 \to \infty} p(z) = \lim_{x_1 \to \infty} \frac{1}{1+e^{-\beta_0-\beta_1 x_1}} = 0.$$

Similarly,

$$\lim_{x_1 \to -\infty} e^{-\beta_0-\beta_1 x_1} = 0 \implies \lim_{x_1 \to -\infty} \frac{1}{1+e^{-\beta_0-\beta_1 x_1}} = \frac{1}{1+0} = 1.$$