PSTAT 131 Homework 3

Tammy Truong

2022-04-12

Classification

Question 1:

Splitting the data and stratifying the variable `survived`. We verify that the training and testing data sets have the appropriate number of observations.

```
set.seed(1004)
titanic_split <- initial_split(titanic, prop = 0.70, strata = survived)
titanic_train <- training(titanic_split)
titanic_test <- testing(titanic_split)
```

```
nrow(titanic_train) # number of rows/observations in training set
```

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

```
nrow(titanic_test) # number of obs on testing set
```

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

To check for any missing values in the training set, we use the function `is.na()`.

```
# obtaining the first couple obs since the training set has a large amount of rows
titanic_train_head <- head(titanic_train)
is.na(titanic_train_head)
```

```
##      passenger_id survived pclass  name   sex   age sib_sp parch ticket  fare
## [1,]        FALSE    FALSE  FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE FALSE
## [2,]        FALSE    FALSE  FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE FALSE
## [3,]        FALSE    FALSE  FALSE FALSE FALSE  TRUE  FALSE FALSE  FALSE FALSE
## [4,]        FALSE    FALSE  FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE FALSE
## [5,]        FALSE    FALSE  FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE FALSE
## [6,]        FALSE    FALSE  FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE FALSE
##      cabin embarked
## [1,]  TRUE    FALSE
## [2,]  TRUE    FALSE
## [3,]  TRUE    FALSE
## [4,] FALSE    FALSE
## [5,]  TRUE    FALSE
## [6,]  TRUE    FALSE
```

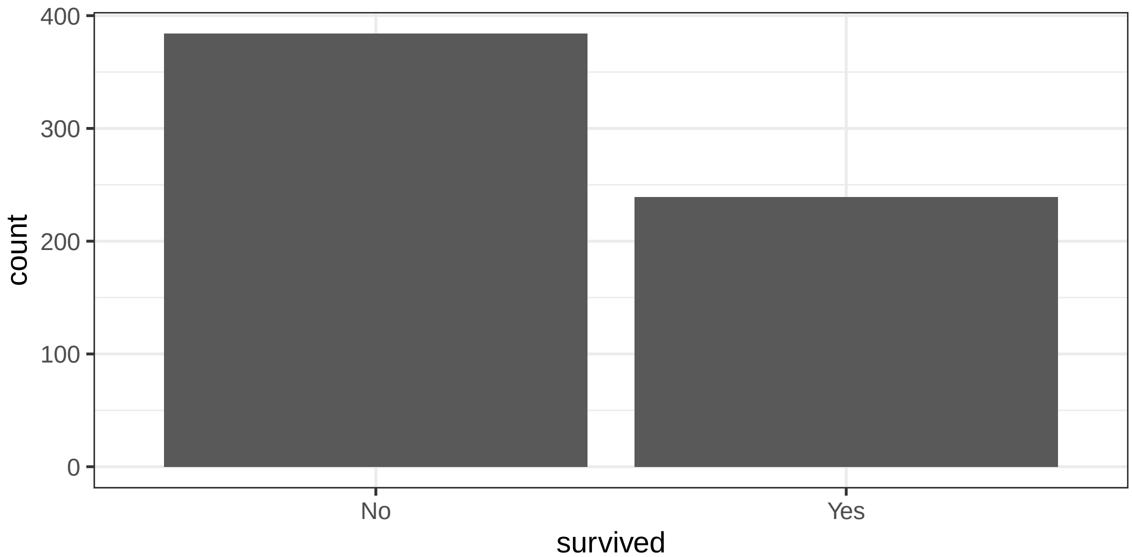
From the results above, there are clearly many missing values in the training set. This issue may cause further misrepresentation of the sample, thus providing an inaccurate prediction.

We use stratified sampling to produce a sample group that best represents the population, thus having every every subgroup of demographics represented within our testing.

Question 2:

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

```
titanic_train %>%
  ggplot(aes(x = survived)) +
  geom_histogram(bins = 60, stat = 'count') +
  theme_bw()
```



Since `survived` is a categorical variable, we see that the histogram above shows a higher count of the deceased than those who survived.

Question 3:

Using the **training** set, we create a correlation matrix of all continuous variables such as

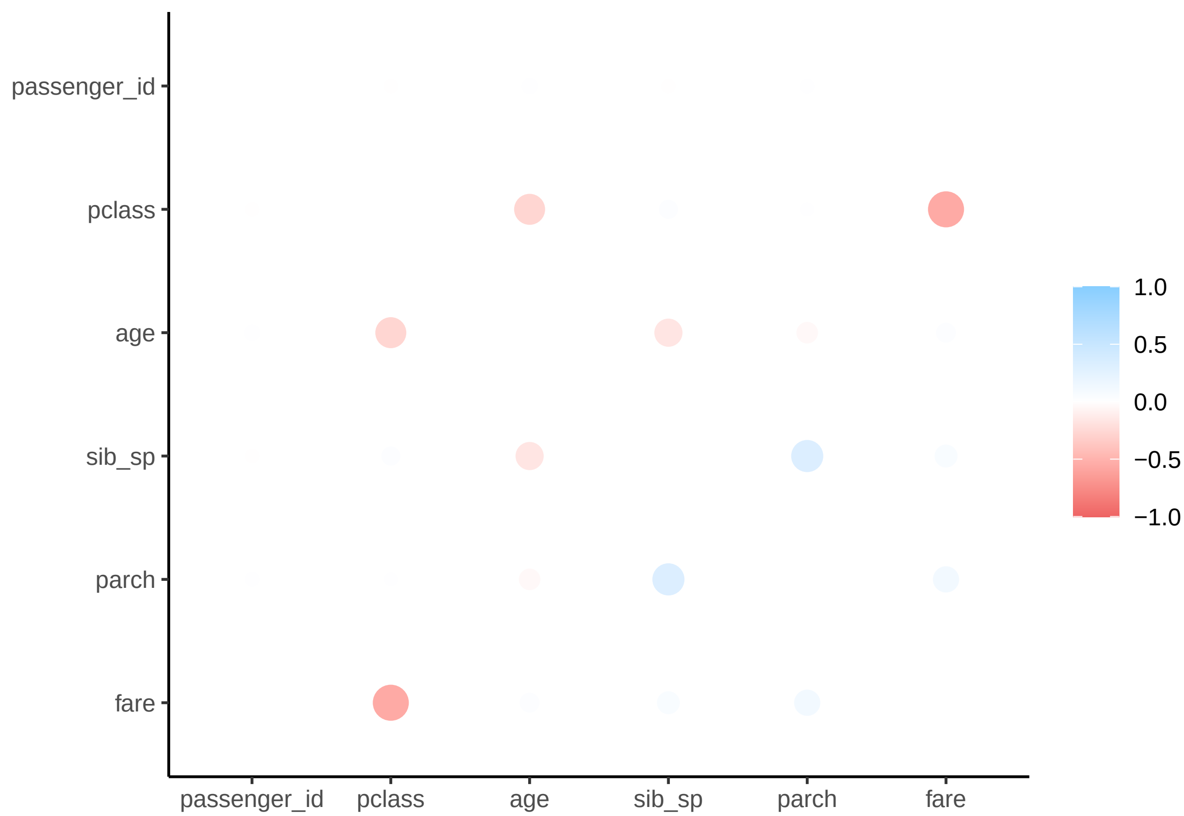
- `age`: age in years
- `sibsp`: number of siblings/spouses aboard
- `parch`: number of parents/children aboard

```
# correlation matrix
cor_titanic <- titanic_train %>%
  select(where(is.numeric)) %>%
  correlate(use = "pairwise.complete.obs", method = "pearson")
```

```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```

```
# visualization of the matrix
rplot(cor_titanic)
```

```
## Don't know how to automatically pick scale for object of type noquote. Defaulting to continuous.
```



The visualization of the correlation matrix shows a clear pattern of negatively correlated relationships between variables. The variables `pclass` and `fare`are very negatively correlated. `sib_sp` and `parch` are somewhat positively correlated while the rest are slightly negatively or positively correlated.

Question 4:

Using the **training** data set, we create a recipe predicting `survived`. We include the following predictors:

- ticket class
- sex
- age
- number of siblings or spouses aboard
- number of parents or children aboard
- passenger fare

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

Question 5:

Specifying a **logistic regression** model for classification using the `"glm"` engine and then creating a workflow. Adding it to my model and the appropriate recipe, we use the `fit` function to apply the workflow to the **training** data.

```
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:

LDA

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:

QDA

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:

Naive Bayes

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:

Assessing Performance: Predictions & Accuracy

After fitting 4 different models to my training data, I'll use `predict()` and `bind_colds()` to generate predictions using each of these 4 models and my **training** data. Then use the *accuracy* metric to assess the performance of each of the four models.

Logistic Regression

```
log_predict <- predict(log_fit, new_data = titanic_train, type = "prob")
log_predict <- bind_cols(log_predict, titanic_train %>% select(survived))

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

```
##            Truth
## Prediction  No Yes
##        No  343  72
##        Yes  41 167
```

```
augment(log_fit, new_data = titanic_train) %>%
  accuracy(truth = as.factor(survived), estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.819
```

LDA

```
lda_predict <- predict(lda_fit, new_data = titanic_train, type = "prob")
lda_predict <- bind_cols(lda_predict, titanic_train %>% select(survived))

augment(lda_fit, new_data = titanic_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##            Truth
## Prediction  No Yes
##        No  342  73
##        Yes  42 166
```

```
augment(lda_fit, new_data = titanic_train) %>%
  accuracy(truth = as.factor(survived), estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.815
```

QDA

```r
qda_predict <- predict(qda_fit, new_data = titanic_train, type = "prob")
qda_predict <- bind_cols(qda_predict, titanic_train %>% select(survived))

augment(qda_fit, new_data = titanic_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction  No Yes
##        No  352  88
##        Yes  32 151
```

```r
augment(qda_fit, new_data = titanic_train) %>%
  accuracy(truth = as.factor(survived), estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.807
```

Naive Bayes

```r
nb_predict <- predict(nb_fit, new_data = titanic_train, type = "prob")
nb_predict <- bind_cols(nb_predict, titanic_train %>% select(survived))

augment(nb_fit, new_data = titanic_train) %>%
  conf_mat(truth = survived, estimate = .pred_class)
```

```
##           Truth
## Prediction  No Yes
##        No  335  80
##        Yes  49 159
```

```r
augment(nb_fit, new_data = titanic_train) %>%
  accuracy(truth = as.factor(survived), estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.793
```

From the above results, the model with the best performance is Logistic Regression, which is followed by LDA, QDA, and Naive Bayes.

Question 10:

Now, we fit the model with the highest training accuracy to the *testing* data. From the results in Question 9, we know that the logistic regression model has the highest training accuracy.

```
log_test <- predict(log_fit, new_data = titanic_test, type = "prob")
log_test
```

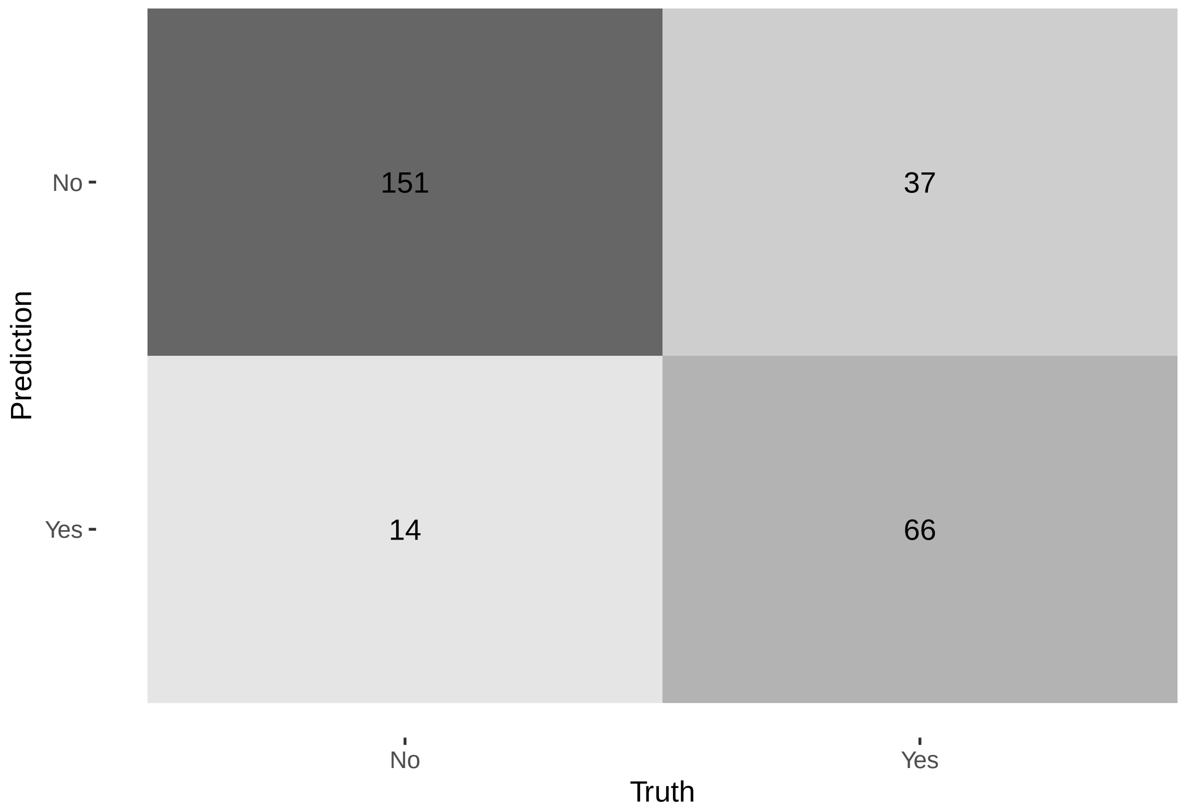
```
## # A tibble: 268 x 2
##    .pred_No .pred_Yes
##       <dbl>     <dbl>
##  1   0.0752    0.925
##  2   0.382     0.618
##  3   0.935     0.0649
##  4   0.149     0.851
##  5   0.261     0.739
##  6   0.779     0.221
##  7   0.771     0.229
##  8   0.270     0.730
##  9   0.739     0.261
## 10   0.371     0.629
## # ... with 258 more rows
```

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

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.810
```

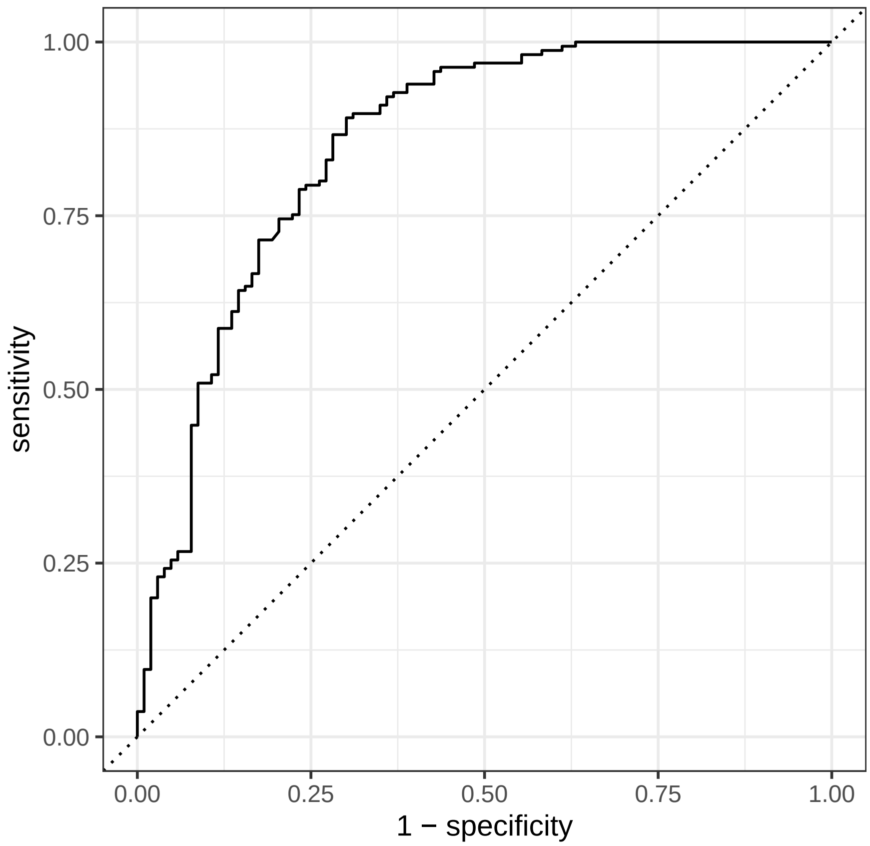
Confusion Matrix

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



ROC Curve

```
augment(log_fit, new_data = titanic_test) %>%
  roc_curve(truth = as.factor(survived), .pred_No) %>%
  autoplot()
```



Calculating the area under it with AUC.

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

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.853
```

The accuracy of the training and testing data set are both over 80%, which shows that it is accurate in its predictions. The testing data set has a higher accuracy than the training data set, and the cause of this may be from the small sample size compared to the training data.