hw4_sela_amir

Amir Sela 2024-11-09

```
library(tidyverse)
library(tidymodels)
library(car)
library(ISLR)
library(titanic)
library(recipes)
```

```
glimpse(titanic_train) # take a glimpse at the dataset
```

```
## Rows: 891
## Columns: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…
## $ Survived    <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1…
## $ Pclass      <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3…
## $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl…
## $ Sex         <chr> "male", "female", "female", "female", "male", "male", "mal…
## $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 14, …
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0…
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0…
## $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", "37…
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,…
## $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6", "C…
## $ Embarked    <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", "S", "S"…
```

Main Variables

- Outcome variable :
  - Survived (1 = survived, 0 = did not survive)
- Predictor variables:
  - Pclass (passenger class), Sex, Age, SibSp (siblings/spouses aboard), Parch (parents/children aboard), and Fare.

```
titanic_train %>% # the dataframe
  select(Pclass, Sex, Survived) %>%  # select only needed columns
  group_by(Pclass, Sex) %>% # grouping by class and sex
  summarise(
    avg_survivalRate_class_sex = mean(Survived) # get the avarage
  )
```

```
## `summarise()` has regrouped the output.
## ℹ Summaries were computed grouped by Pclass and Sex.
## ℹ Output is grouped by Pclass.
## ℹ Use `summarise(.groups = "drop_last")` to silence this message.
## ℹ Use `summarise(.by = c(Pclass, Sex))` for per-operation grouping
##   (`?dplyr::dplyr_by`) instead.
```

```
## # A tibble: 6 × 3
## # Groups:   Pclass [3]
##   Pclass Sex    avg_survivalRate_class_sex
##    <int> <chr>                       <dbl>
## 1      1 female                      0.968
## 2      1 male                        0.369
## 3      2 female                      0.921
## 4      2 male                        0.157
## 5      3 female                      0.5
## 6      3 male                        0.135
```

- Pclass column has one of three variables:
  - 1 (Upper class socio status)
  - 2 (Middle Class socio status)
  - 3 (Lower Class socio status)

    By the analysis we can say that the higher the socio status (Pclass) and being a female strongly influences the survival rate

```
# these chunks are same as the code above just age is also added as an indicator
younger_titanic <- titanic_train %>%
  filter(Age <= 18) %>%
  select(Pclass, Sex, Survived, Age) %>%
  group_by(Pclass, Sex) %>%
  summarise(
    avg_survivalRate_class_sex_young = mean(Survived)
  )
```

```
## `summarise()` has regrouped the output.
## ℹ Summaries were computed grouped by Pclass and Sex.
## ℹ Output is grouped by Pclass.
## ℹ Use `summarise(.groups = "drop_last")` to silence this message.
## ℹ Use `summarise(.by = c(Pclass, Sex))` for per-operation grouping
##   (`?dplyr::dplyr_by`) instead.
```

```
younger_titanic
```

```
## # A tibble: 6 × 3
## # Groups:   Pclass [3]
##   Pclass Sex    avg_survivalRate_class_sex_young
##    <int> <chr>                             <dbl>
## 1      1 female                            0.909
## 2      1 male                              0.8
## 3      2 female                            1
## 4      2 male                              0.6
## 5      3 female                            0.512
## 6      3 male                              0.216
```

```
older_titanic <- titanic_train %>%
  filter(Age > 18) %>%
  select(Pclass, Sex, Survived, Age) %>%
  group_by(Pclass, Sex) %>%
  summarise(
    avg_survivalRate_class_sex_old = mean(Survived)
  )
```

```
## `summarise()` has regrouped the output.
## ℹ Summaries were computed grouped by Pclass and Sex.
## ℹ Output is grouped by Pclass.
## ℹ Use `summarise(.groups = "drop_last")` to silence this message.
## ℹ Use `summarise(.by = c(Pclass, Sex))` for per-operation grouping
##   (`?dplyr::dplyr_by`) instead.
```

```
older_titanic
```

```
## # A tibble: 6 × 3
## # Groups:   Pclass [3]
```

```
##   Pclass Sex    avg_survivalRate_class_sex_old
##    <int> <chr>                           <dbl>
## 1      1 female                          0.973
## 2      1 male                            0.375
## 3      2 female                          0.9
## 4      2 male                            0.0714
## 5      3 female                          0.424
## 6      3 male                            0.134
```

As we can see from the analyzed data, the survival rate of the young(less than 18) and the old(more than 18) doesnt change for passangers in class 1(upper class), but as the socio classes decrease there is a higher chance of survival if you are young, so age moderatly influences the survival rate as an indicator.

```
titanic_train %>%
  count(if_any(everything(), is.na)) # checks if any columns contain NA values and
```

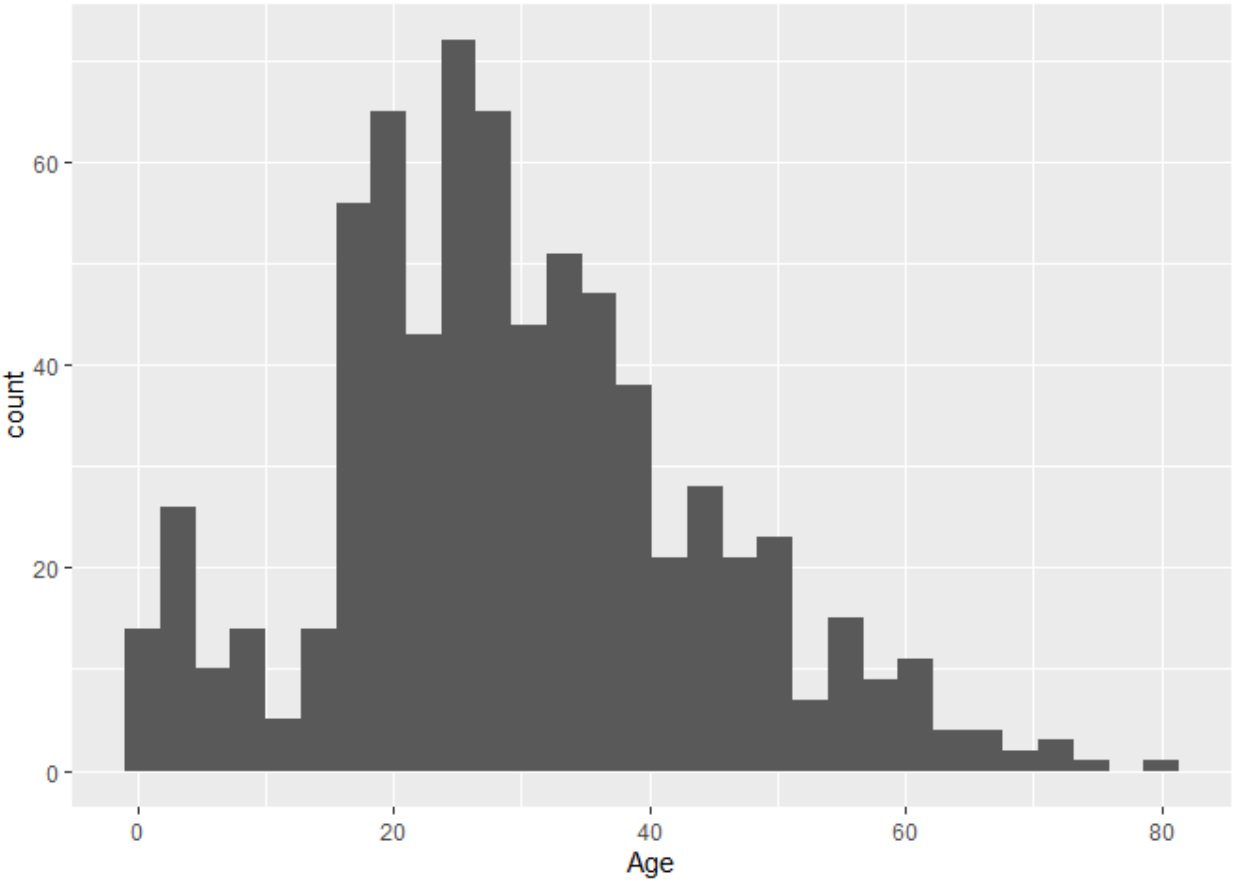
```
##   if_any(everything(), is.na)   n
## 1                       FALSE 714
## 2                        TRUE 177
```

As we can see of 0.1986532 percent the data is missing so filling out the missing values data is a good idea. For us to fill the missing values with mean age we need to check for data so it isnt strongly skewed.

```
ggplot(titanic_train,aes( x = Age)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.

## Warning: Removed 177 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



By visualizing the distribution of the age, we can see that the data is only lightly skewed to the right, so its safe for us to fill the missing value with the mean age. To take it a step further I will fill the missing Age by class. If missing age is in class 1, i will put the mean age of that class, and so on. This way not all 177 missing ages are filled with the same value.

```
titanic_train <- titanic_train %>%
  group_by(Pclass) %>%
  mutate(Age = ifelse(is.na(Age), mean(Age, na.rm = TRUE), Age))# if age is missin

#now check if the dataframe has any missing values

titanic_train %>%
  count(if_any(everything(), is.na))
```

```
## # A tibble: 3 × 3
## # Groups:   Pclass [3]
##   Pclass `if_any(everything(), is.na)`     n
##    <int> <lgl>                         <int>
## 1      1 FALSE                           216
## 2      2 FALSE                           184
```

```
## 3      3 FALSE                          491
```

There are not any missing values anymore, so now we can continue working with the data to create accurate models.

Splitting the data with 80/20 %

```
set.seed(1116)

titanic_split = initial_split(titanic_train, prop = 0.80)

titanic_train_data <- training(titanic_split)
titanic_test_data <- testing(titanic_split)
```

We will use the 80% data to train the model and the 20% data to test the model

Now we will create dummy variables using recipes

```
#| label: Recipe-Dummy
titanic_train_data <- titanic_train_data %>%
  mutate(Survived = as.factor(Survived))

train_rec <- recipe(Survived ~ Age + Pclass + Sex, data = titanic_train_data) %>%

  step_mutate(Pclass = as.factor(Pclass)) %>% # we need to treat the Pclass catego
  #we wanna remove passangerId, name,fare, ticket number and cabin because they do
  step_dummy(all_nominal(), -all_outcomes()) # creating dummy variables, only affe


train_rec_prepped <- prep(train_rec) # by preping the data we remove the columns w
titanic_rec_clean <- bake(train_rec_prepped, new_data = NULL) # by baking the data

head(titanic_rec_clean) # now we can see the we removed the unneccesary columns
```

```
## # A tibble: 6 × 5
##     Age Survived Pclass_X2 Pclass_X3 Sex_male
##   <dbl> <fct>        <dbl>     <dbl>    <dbl>
## 1  25.1 0                0         1        1
## 2  25.1 0                0         1        1
```

```
## 3  46   0               0         0        1
## 4  30   0               1         0        1
## 5  27   0               0         1        1
## 6  25.1 0               0         1        1
```

Now we have a clean recipe with dummy variables and removed unneccesary variables from the dataframe, we are ready to build the model

```
titanic_mod <- logistic_reg() %>%
  set_engine("glm", family = "binomial") # giving the variable a model

titanic_workflow <- workflow() %>%  # creating a workflow for simplicity
  add_model(titanic_mod) %>%
  add_recipe(train_rec)

titanic_fit <- titanic_workflow %>% # fitting the data to the model and the recipe
  fit(data = titanic_train_data)
  # view the model

titanic_tidy <- tidy(titanic_fit) # tidying the data so its easy to understant

titanic_tidy
```

```
## # A tibble: 5 × 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   3.44     0.420        8.19 2.70e-16
## 2 Age          -0.0322   0.00838     -3.84 1.21e- 4
## 3 Pclass_X2    -1.28     0.290       -4.41 1.03e- 5
## 4 Pclass_X3    -2.36     0.277       -8.51 1.81e-17
## 5 Sex_male     -2.45     0.204      -12.0  4.30e-33
```

Based on this we could ask questions about survival rate and estimate the propability E.x what is the survival rate for a third-class female passanger aged 20? The propability that a third-class female passenger aged 20 survives is 60.7324031 %. Here to calculate the propability i used the 1 / (1 + exp(-logit_p)) formula and just plugged in the number. Based on my finding i predict that the passenger will survive. I assume that if the propability is less than .5 the chances are that the passenger will not survive

```
titanic_pred <- predict(titanic_fit, titanic_test_data, type = "prob") %>%
```

```
  bind_cols(titanic_test_data %>% select(Survived))
```

```
## Adding missing grouping variables: `Pclass`
```

```
# i will make a function for the prediction code chunk what way when we want to ch

calculate_titanic <- function(cutoff,metric) { # creating the function
  titanic_pred <- titanic_pred %>%
    mutate(predicted = ifelse(.pred_1 > cutoff, "1", "0")) # dataframe titanic_pre

  predicted_matrix <- table(Predicted = titanic_pred$predicted,
                            Actual = titanic_pred$Survived) # fpr the actual and pre
  accuracy <- mean(titanic_pred$predicted == titanic_pred$Survived)

  sensitivity <- predicted_matrix["1","1"] / (predicted_matrix["1","1"] + predicte
  specificity <- predicted_matrix["0","0"] / (predicted_matrix["0","0"] + predicte

# if else loop to return based on what we want
if (metric == "sensitivity") {
  return(sensitivity)
}
  else if( metric == "specificity"){
    return(specificity)
  }
  else if( metric == "accuracy"){
    return(accuracy)
  }
  else if(metric == "predicted_matrix"){
    return(predicted_matrix)
  }
}

calculate_titanic(0.5, "predicted_matrix")
```

```
##          Actual
## Predicted  0  1
##         0 87 16
##         1 14 62
```

With a cutoff of .5, the accuracy of the model is 0.8324022, the sensitivity is 0.7948718 and the specificity is 0.8613861. With a cutoff of .5, the rate of sensitivity(true positive

rate) and the specificity(true negative rate) are close to each other which both are around .8(specificity is higher). This means that most of the time we will get correct predictions when using this model with a cutoff of .5. Now lets change the cutoff to .3(if prediction that the passenger will survive is greater than .3 it will mark it as the passenger will survive).

With a cutoff of .3, the accuracy of the model is 0.8100559, the sensitivity is 0.8974359 and the specificity is 0.7425743. As we can see the accuracy has slightly decreased(by 0.0223464). But the most notable differences are between specificity and sensitivity. When the cutoff is .3 then we can see that the sensitivity has increased meaning there is a lesser chance of a having a false negative, and the specificity has decreased meaning there is a higher chance of a false positive.

Now again lets change the cutoff to .7.

With a cutoff of .7, the accuracy of the model is 0.7988827, the sensitivity is 0.5512821 and the specificity is 0.990099. The accuracy of the model has not decreased much, but the specificity has has increases dramatically(close to 1) and the sensitivity has also decreased almost to one half.

If we prioritize sensitivity(true positive rate) than that would mean that the model can better predict the number of surivivors, and can reduce the risk of missing someone who would survive. If we prioritize specificity(tru negative rate) that means we can more accuretaly predict those who wont survive. In this case we wanna prioritize sensitivity, so that way we can capture as many as we can correctly, because we would rather save those who we know we can save and then we can also save those who the model predicted would die but lived.