Assignment 1

Kevin Linares

2025-01-28

Setup

```
pacman::p_load(titanic, caret, pROC, jtools, tidymodels)
options(scipen=999)
```

Data

In this notebook we use the Titanic data that is used on Kaggle (https://www.kaggle.com) as an introductory competition for getting familiar with machine learning. It includes information on a set of Titanic passengers, such as age, sex, ticket class and whether he or she survived the Titanic tragedy. Source: https://www.kaggle.com/c/titanic/data

```
titanic <- titanic_train |> as_tibble()
glimpse(titanic)
```

```
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"~
```

We begin with some minor data preparations. The `lapply()` function is a handy tool if the task is to apply the same transformation (e.g. `as.factor()`) to multiple columns of a data frame.

```
titanic <- titanic |> mutate_at(vars(2, 3, 5, 12), factor)
```

- The age variable has some NAs, as a quick and dirty solution we can create a categorized age variable with NAs as an additional factor level.

```
titanic <-
  titanic |>
  mutate(Age_c = cut(Age, 5), Age_c = addNA(Age_c))

# print summary of new age category variable with NA level
titanic |> pull(Age_c) |> summary()
```

```
(0.34,16.3] (16.3,32.3] (32.3,48.2] (48.2,64.1] (64.1,80.1]        <NA>
        100         346         188          69          11         177
```

Train and test set

- Next we split the data into a training (80%) and a test (20%) part. This can be done by random sampling with `sample()`.

```
set.seed(9395)
# we use the rsample package to expand on this in later assignments.
titanic_split <- initial_split(titanic, prop = .80, breaks = 2)

# train split
titanic_train <- training(titanic_split)

# test split
titanic_test <- testing(titanic_split)
```

Logistic regression

- In this exercise we simply use logistic regression as our prediction method, since we want to focus on the evaluation part. Build a first logit model with `Survived` as the outcome and `Pclass`, `Sex`, `Age_c`, `Fare` and `Embarked` as features.

```
m1 <- glm(Survived ~ Pclass + Sex + Age_c + Fare + Embarked,
          data = titanic_train, family = "binomial")
```

- A quick look at the coefficients of the first logit model.

```
summ(m1)
```

```
MODEL INFO:
Observations: 712
Dependent Variable: Survived
Type: Generalized linear model
  Family: binomial
  Link function: logit

MODEL FIT:
χ²(12) = 326.56, p = 0.00
Pseudo-R² (Cragg-Uhler) = 0.50
Pseudo-R² (McFadden) = 0.34
AIC = 650.22, BIC = 709.61

Standard errors:MLE
------------------------------------------------------------
                          Est.      S.E.   z val.      p
---------------------- -------- --------- -------- ------
(Intercept)              17.99   1691.69     0.01   0.99
Pclass2                  -0.87      0.33    -2.64   0.01
Pclass3                  -2.27      0.32    -7.01   0.00
Sexmale                  -2.54      0.21   -11.95   0.00
Age_c(16.3,32.3]         -0.85      0.33    -2.56   0.01
Age_c(32.3,48.2]         -1.23      0.38    -3.28   0.00
Age_c(48.2,64.1]         -1.58      0.50    -3.15   0.00
Age_c(64.1,80.1]        -16.96    697.69    -0.02   0.98
Age_cNA                  -1.18      0.38    -3.10   0.00
Fare                     -0.00      0.00    -0.05   0.96
EmbarkedC               -14.17   1691.69    -0.01   0.99
EmbarkedQ               -13.99   1691.69    -0.01   0.99
```

```
EmbarkedS                -14.75    1691.69     -0.01    0.99
-----------------------------------------------------------
```

- Now, build an additional logit model that uses the same features, but includes at least one interaction or non-linear term.

```
m2 <- glm(Survived ~ Pclass*Sex + Age_c + Fare + Embarked,
          data = titanic_train, family = "binomial")
```

- Again, summarize the resulting object.

```
summ(m2)
```

```
MODEL INFO:
Observations: 712
Dependent Variable: Survived
Type: Generalized linear model
  Family: binomial
  Link function: logit

MODEL FIT:
χ²(14) = 346.70, p = 0.00
Pseudo-R² (Cragg-Uhler) = 0.52
Pseudo-R² (McFadden) = 0.36
AIC = 634.08, BIC = 702.60

Standard errors:MLE
------------------------------------------------------------
                          Est.      S.E.   z val.      p
---------------------- -------- --------- -------- ------
(Intercept)              18.17   1691.17     0.01   0.99
Pclass2                  -0.68      0.79    -0.87   0.39
Pclass3                  -3.53      0.68    -5.18   0.00
Sexmale                  -3.51      0.64    -5.48   0.00
Age_c(16.3,32.3]         -0.91      0.32    -2.83   0.00
Age_c(32.3,48.2]         -1.34      0.38    -3.54   0.00
Age_c(48.2,64.1]         -1.74      0.54    -3.24   0.00
Age_c(64.1,80.1]        -16.95    703.90    -0.02   0.98
Age_cNA                  -1.20      0.36    -3.29   0.00
Fare                     -0.00      0.00    -0.29   0.77
EmbarkedC               -13.38   1691.17    -0.01   0.99
EmbarkedQ               -13.20   1691.17    -0.01   0.99
```

```
EmbarkedS                -14.01   1691.17    -0.01   0.99
Pclass2:Sexmale           -0.60      0.85    -0.70   0.48
Pclass3:Sexmale            1.72      0.69     2.48   0.01
---------------------------------------------------------
```

Prediction in test set

- Given both logit objects, we can generate predicted risk scores/ predicted probabilities of `Survived` in the test set.

```
pred1 <- predict(m1, newdata = titanic_test, type="response")
pred2 <- predict(m2, newdata = titanic_test,  type="response")
```

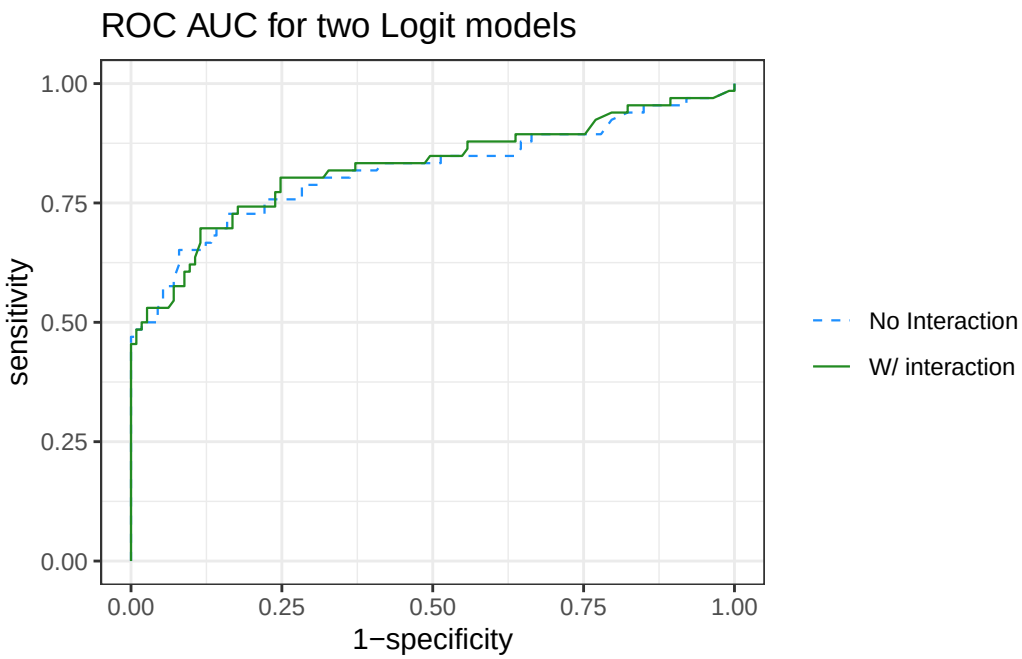
- It is often useful to first get an idea of prediction performance independent of specific classification thresholds. Use the `pROC` (or `PRROC`) package to create roc objects for both risk score vectors.

```
# create roc object for m1
roc_m1 <- roc(titanic_test$Survived, pred1)

# create roc object for m2
roc_m2 <- roc(titanic_test$Survived, pred2)
```

- Now, you can print and plot the resulting `roc` objects.

```
# plot ROC curves
ggroc(list(roc_m1, roc_m2), aes = c("colour", "linetype"),
            legacy.axes = TRUE,
            linewidth = 0.4) +
  scale_linetype_manual("", values=c(2, 1),
                        labels = c("No Interaction",
                                   "W/ interaction")) +
  scale_colour_manual("",
                      labels = c("No Interaction",
                                 "W/ interaction"),
                      values = c("dodgerblue",
                                 "forestgreen")) +
  theme_bw() +
  ggtitle("ROC AUC for two Logit models")
```




```
roc_m1
```

```
Call:
roc.default(response = titanic_test$Survived, predictor = pred1)

Data: pred1 in 113 controls (titanic_test$Survived 0) < 66 cases (titanic_test$Survived 1).
Area under the curve: 0.8172
```

```
roc_m2
```

```
Call:
roc.default(response = titanic_test$Survived, predictor = pred2)

Data: pred2 in 113 controls (titanic_test$Survived 0) < 66 cases (titanic_test$Survived 1).
Area under the curve: 0.825
```

In your own words, how would you interpret these ROC curves? What do you think about the ROC-AUCs we observe here?

- The Area Under the Receiver Operating Characteristic curve (AUC) helps us compare different predictive models by illustrating the trade-off between sensitivity (correctly identifying actual survivors) and specificity (correctly identifying those who did not survive). Our two models, one with and one without an interaction term between gender and passenger class, have similar AUCs of 0.817 and 0.825, respectively. A higher AUC indicates better performance in distinguishing survivors from non-survivors. Both models demonstrate the typical inverse relationship between sensitivity and specificity: high specificity corresponds to sensitivity around 0.5, and vice-versa, high sensitivity corresponds to specificity near 0.
- Since our goal is to effectively allocate limited resources to passengers unlikely to survive, we should prioritize specificity. Accurately identifying non-survivors (true negatives) is crucial. While a high threshold may risk misclassifying some survivors (false negatives), this is less detrimental than misclassifying those unlikely to survive (false positives). Therefore, we aim for high specificity. A specificity threshold of 0.85 appears reasonable, and both models suggest this would yield a sensitivity of approximately 0.70. This means we'd correctly identify 85% of those who did not survive, while still correctly identifying 70% of actual survivors.

As a next step, we want to predict class membership given the risk scores of our two models. Here we use the default classification threshold, 0.5.

```
# build function to predict based on threshold, & produce confusionmatrix
pred_class_member_fun <- function(roc_object, threshold_level,
                                  predictions, test_set) {
  # set threshold
  coords = coords(roc_object, threshold_level)
  threshold_used = coords$threshold

  message(cat("Optimal threshold based on ROC curve:", threshold_used, "\n"))

  # Predict classes based on threshold
  predicted_class = ifelse(predictions >= threshold_used, 1, 0)

  # Evaluate model performance
  confusion_matrix = table(test_set, predicted_class)
  return(confusionMatrix(confusion_matrix))
}
```

- On this basis, we can use `confusionMatrix()` to get some performance measures for the predicted classes.

```
# first model
m1_performance <- pred_class_member_fun(roc_object = roc_m1,
                      threshold_level = .5,
                      predictions = pred1,
                      test_set = titanic_test |> pull(Survived))
```

```
Optimal threshold based on ROC curve: 0.5
```

```
print(m1_performance)
```

```
Confusion Matrix and Statistics

        predicted_class
test_set  0  1
       0 98 15
       1 21 45

               Accuracy : 0.7989
                 95% CI : (0.7326, 0.855)
    No Information Rate : 0.6648
    P-Value [Acc > NIR] : 0.00005395

                  Kappa : 0.5597

 Mcnemar's Test P-Value : 0.4047

            Sensitivity : 0.8235
            Specificity : 0.7500
         Pos Pred Value : 0.8673
         Neg Pred Value : 0.6818
             Prevalence : 0.6648
         Detection Rate : 0.5475
   Detection Prevalence : 0.6313
      Balanced Accuracy : 0.7868

       'Positive' Class : 0
```

```r
m2_performance <- pred_class_member_fun(roc_object = roc_m2,
                      threshold_level = .50,
                      predictions = pred2,
                      test_set = titanic_test |> pull(Survived))
```

```
Optimal threshold based on ROC curve: 0.5
```

```r
print(m2_performance)
```

```
Confusion Matrix and Statistics

        predicted_class
test_set   0   1
       0 103  10
       1  27  39

               Accuracy : 0.7933
                 95% CI : (0.7265, 0.8501)
    No Information Rate : 0.7263
    P-Value [Acc > NIR] : 0.024545

                  Kappa : 0.5308

 Mcnemar's Test P-Value : 0.008529

            Sensitivity : 0.7923
            Specificity : 0.7959
         Pos Pred Value : 0.9115
         Neg Pred Value : 0.5909
             Prevalence : 0.7263
         Detection Rate : 0.5754
   Detection Prevalence : 0.6313
      Balanced Accuracy : 0.7941

       'Positive' Class : 0
```

Briefly explain potential limitations when measuring prediction performance as carried out in the last two code chunks.

- Perhaps the greatest risk or limitation to measuring prediction performance in this way is that a practitioner can manipulate the threshold value until a performance metric of interest is satisfied. This would be comparable to p-hacking in statistics. Other considerations are that if there is imbalance in the data, we have to closely consider which metric to use and report. For instance, we there is imbalance and our model has high accuracy, we may be able to predict correctly 100% of the time, but less so predicting the lower class which may be more important to the intervention. A third limitation is that confusionmatrices with threshold dependency treats errors equally, but in some instances a false negativemay be more problematic than a false positive such as in cancer diagnostics.