Logistic Regression with Titanic Dataset

Supreeya Srasom

1. Load libraries

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(titanic)
library(knitr)
```

2. Display and clean the dataset

```
# Display the dataset
head(titanic_train)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                 Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                            Allen, Mr. William Henry   male  35     0     0
## 6                                    Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500              S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250              S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500              S
## 6           330877  8.4583              Q
```

```r
# Clean the dataset
titanic_train_df <- na.omit(titanic_train)
```

3. Building the Logistic Regression Model

```r
# Perform a 70/30 split of the data
train_test_split <- function(data, train_size = 0.7){
  set.seed(42)
  n <- nrow(data)
  id <- sample(1:n, size = n * train_size)
  train_data <- data[id, ]
  test_data <- data[-id, ]
  list(train = train_data, test = test_data)
}
```

```r
prep_data <- train_test_split(titanic_train_df)
train_data <- prep_data[[1]]
test_data <- prep_data[[2]]
```

3.1 Split data

```r
model_titanic <- glm(Survived ~ Pclass + Sex + Age, data = train_data, family = "binomial")
summary(model_titanic)
```

3.2 Train Model

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age, family = "binomial",
##     data = train_data)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.604600   0.637259   8.795  < 2e-16 ***
## Pclass      -1.443887   0.174955  -8.253  < 2e-16 ***
## Sexmale     -2.739281   0.262607 -10.431  < 2e-16 ***
## Age         -0.041450   0.009522  -4.353 1.34e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 673.56  on 498  degrees of freedom
## Residual deviance: 432.26  on 495  degrees of freedom
## AIC: 440.26
##
## Number of Fisher Scoring iterations: 5
```

4. Predicting on training and testing sets

```r
# Predict the train set
predict_train <- ifelse(predict(model_titanic,type = "response") >= 0.5,1,0)
```

```r
# Predict the test set
predict_test <- ifelse(predict(model_titanic,type="response",
                               newdata = test_data) >= 0.5,1,0)
```

5. Model Evaluation

```r
# Train model
conM_train <- table(predict_train,train_data$Survived,dnn = c("Predicted", "Actual"))
conM_train
```

5.1 Confusion Matrix

```
##           Actual
## Predicted   0   1
##         0 253  57
##         1  44 145
```

```r
# Test model
conM_test <- table (predict_test,test_data$Survived,dnn = c("Predicted","Actual"))
conM_test
```

```
##           Actual
## Predicted   0   1
##         0 106  27
##         1  21  61
```

```r
# Calculate accuracy, precision, recall, and F1 score for the train model
postrain11 <- conM_train[1,1]
postrain12 <- conM_train[1,2]
postrain21 <- conM_train[2,1]
postrain22 <- conM_train[2,2]
n_train <- sum(conM_train)
# Accuracy
accuracy_train <- (postrain11 + postrain22) / n_train
# Precision
precision_train <- postrain22 / (postrain21 + postrain22)
# Recall
recall_train <- postrain22 / (postrain12 + postrain22)
# F1 score
f1_train <-  2*(precision_train*recall_train)/(precision_train+recall_train)
```

```r
# Calculate accuracy, precision, recall, and F1 score for the test model
postest11 <- conM_test[1,1]
postest12 <- conM_test[1,2]
postest21 <- conM_test[2,1]
postest22 <- conM_test[2,2]
n_test <- sum(conM_test)
# Accuracy
accuracy_test <- (postest11 + postest22) / n_test
# Precision
precision_test <- postest22 / (postest21 + postest22)
# Recall
```

```r
recall_test <- postest22 / (postest12 + postest22)
# F1 score
f1_test <- 2*(precision_test*recall_test)/(precision_test+recall_test)
```

```r
# Combine the outputs in a table
metrics <- data.frame("Accuracy" = c(accuracy_train, accuracy_test),
                      "Precision" = c(precision_train, precision_test),
                      "Recall" = c(recall_train, recall_test),
                      "F1 Score" = c(f1_train, f1_test))

row.names(metrics) <- c("Train model", "Test model")

kable(metrics, caption ="Metrics Comparison")
```

5.2 Metrics

Table 1: Metrics Comparison

	Accuracy	Precision	Recall	F1.Score
Train model	0.7975952	0.7671958	0.7178218	0.7416880
Test model	0.7767442	0.7439024	0.6931818	0.7176471

The logistic regression model trained on Pclass, Sex, and Age achieved an accuracy of 77% on the test set, demonstrating good generalization ability.