Titanic survival prediction in R

Basic Titanic survival prediction using logistic regression

Load and clean dataset

```
library(titanic)
```

```
## Warning: package 'titanic' was built under R version 4.2.2
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
glimpse(titanic_train)
```

```
## 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"~
```

Age contains missing values

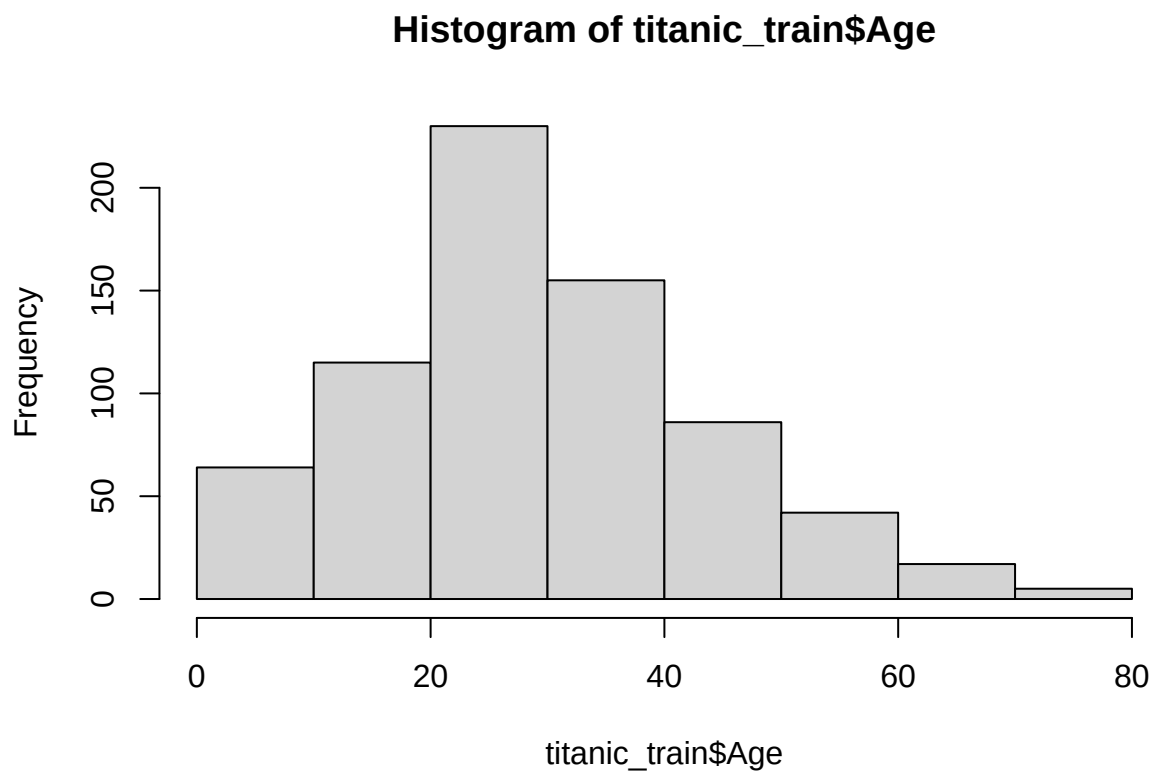
```
sapply(titanic_train, function(x) sum(is.na(x)))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

```
mean(titanic_train$Age, na.rm = T)
```

```
## [1] 29.69912
```

```
hist(titanic_train$Age)
```



Mean imputation

```
titanic_train <- titanic_train %>%
                  mutate(Age = replace_na(Age, mean(Age, na.rm=TRUE)))
sum(is.na(titanic_train$Age))
```

```
## [1] 0
```

Cabin and Embarked also contain null values

```
sapply(titanic_train, function(x) sum(x == ''))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
```

```
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

Drop rows where Embarked is null

```
titanic_train <- titanic_train[titanic_train$Embarked != '',]
sapply(titanic_train, function(x) sum(x == ''))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           0
```

Split data

```
set.seed(7)
n <- nrow(titanic_train)
id <- sample(1:n, n*0.8)
train_df <- titanic_train[id,]
test_df <- titanic_train[-id,]
nrow(train_df)
```

```
## [1] 711
```

```
nrow(test_df)
```

```
## [1] 178
```

Train model

Exclude Cabin as it contains many null values

```
train_df <- train_df[, !names(train_df) %in% c(
  'PassengerId', 'Name', 'Ticket', 'Cabin'
)]
model <- glm(Survived ~ ., data = train_df, family = 'binomial')
model$coefficients
```

```
##  (Intercept)       Pclass      Sexmale          Age        SibSp        Parch
##  5.029219521 -0.944220326 -2.767849503 -0.034667963 -0.341908207 -0.167848190
##         Fare    EmbarkedQ    EmbarkedS
##  0.002443679 -0.416601411 -0.616101504
```

Evaluate model

```r
confus_matrix <- function(model, df, pred_col) {
  p <- predict(model, newdata = df, type = 'response')
  pred <- if_else(p >= 0.5, 1, 0)
  conM <- table(pred, df[,pred_col],
                  dnn = c('Predicted', 'Actual'))

  accuracy <- (conM[1,1] + conM[2,2]) / sum(conM)
  precision <- conM[2,2] / (conM[2,1] + conM[2,2])
  recall <- conM[2,2] / (conM[1,2] + conM[2,2])
  f1 <- 2 * (precision * recall) / (precision + recall)

  return(list(
    conM = conM,
    accuracy = accuracy,
    precision = precision,
    recall = recall,
    f1 = f1
  ))
}
```

```r
confus_matrix(model, train_df, 'Survived')
```

```
## $conM
##          Actual
## Predicted   0   1
##         0 384  80
##         1  55 192
##
## $accuracy
## [1] 0.8101266
##
## $precision
## [1] 0.7773279
##
## $recall
## [1] 0.7058824
##
## $f1
## [1] 0.7398844
```

```r
confus_matrix(model, test_df, 'Survived')
```

```
## $conM
##          Actual
## Predicted  0  1
##         0 94 21
##         1 16 47
##
## $accuracy
## [1] 0.7921348
##
## $precision
```

```
## [1] 0.7460317
##
## $recall
## [1] 0.6911765
##
## $f1
## [1] 0.7175573
```