Document - Logistic Regression by R

Variable Relationship from titanic_train dataset

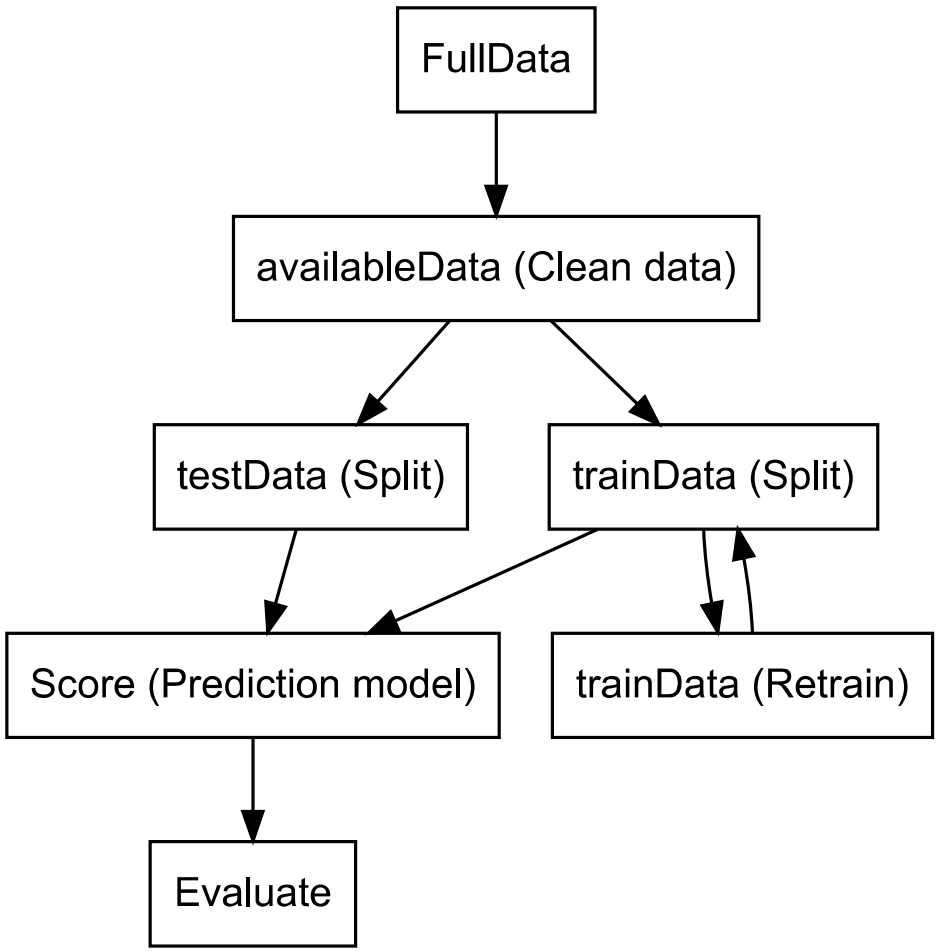
Phattharachai Maichin

2023-09-19

Before start

This document I created to cover how to use “R programming” to manipulate and analyst data to find a relationship (by using logistic regression) among variables to predict titanic passagers survived or not.

Introduction to Machine Learning: I have learnt about Basic Machine Learning Workflow and I summarized it in the picture below to demonstrate a basic machine learning workflow.



Install packages

Packages have been used in this project are “tidyverse” for manage data, “caret” for analyst data, and “titanic” where the example data is located.

Preview dataset

There are many ways to preview dataset such as View(), head(), glimpse(), and other.

```
# But first, let load dataset into a variable for future use.
data <- titanic_train

# Preview dataset - View(data), head(data), glimpse(data)
glimpse(data)
```

```
## 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"…
```

Variables and its description showed in this table.

Variable	Description
PassengerId	Unique id of each passenger
Survived	Outcome of survival (0 = No; 1 = Yes)
Pclass	Socio-economic class (1 = Upper class; 2 = Middle class; 3 = Lower class)
Name	Name of passenger
Sex	Sex of the passenger
Age	Age of the passenger (Some entries contain NaN)
SibSp	Number of siblings and spouses of the passenger aboard
Parch	Number of parents and children of the passenger aboard
Ticket	Ticket number of the passenger
Fare	Fare paid by the passenger
Cabin	Cabin number of the passenger (Some entries contain NaN)
Embarked	Port of embarkation of the passenger (C = Cherbourg; Q = Queenstown; S = Southampton)

Sorce: Titanic Dataset (https://indrayantom.github.io/titanic_guided/)

Check data integrity

by searching missing values is an important step for data integrity.

```
# For data integrity, I have created a function to show if there are missing value in dataset.
data_inegrity <- function(dataset){
  percent_existData <- dataset %>%
                       complete.cases() %>%
                       mean()*100
  nrowoffulldata <<- nrow(dataset)
  ## write the result
  cat(paste("Original data row:", nrowoffulldata, "\n"))
  cat(paste("Percent exist data:", as.character(percent_existData), "% \n"))
  if (percent_existData == 100){
  print("Percent exist data is 100%, the data is ready")
  } else {
  print("Percent exist data is not 100%, need to clean data")
  }
}
# And (if necessary) a function to clean data by omit NULL in dataset.
## If it need to clean data first, there are many option to clean like
## drop, omit, replace, and other, So in this document 'omit' method
## will be applied to the data set.
cleanbyomit <- function(dataset){
  cleanData <<- dataset %>%
                na.omit()
  nrowoffulldata <<- dataset %>%  ## used to update nrow of full data after omit
                na.omit() %>%
                nrow()
  cat(paste("Total data rows before clean:", nrow(dataset),
            "\nSo, Remaining exist data rows:", as.character(nrowoffulldata)))
}

# Call function data_inegrity
data_inegrity(data)
```

```
## Original data row: 891
## Percent exist data: 80.1346801346801 %
## [1] "Percent exist data is not 100%, need to clean data"
```

```
# Call function cleanbyomit
cleanbyomit(data)
```

```
## Total data rows before clean: 891
## So, Remaining exist data rows: 714
```

```
# Preview cleandata again
glimpse(cleanData)
```

```
## Rows: 714
## Columns: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19…
## $ Survived    <int> 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1…
## $ Pclass      <int> 3, 1, 3, 1, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 3, 2, 2, 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, 54, 2, 27, 14, 4, 58, 20, 39, 14, 55, …
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 1, 0, 0, 0…
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 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, 51.8625, 21.0750…
## $ Cabin       <chr> "", "C85", "", "C123", "", "E46", "", "", "", "G6", "C103"…
## $ Embarked    <chr> "S", "C", "S", "S", "S", "S", "S", "S", "C", "S", "S", "S"…
```

Split data into train_data and test_data

```
# Then, function below is used to split data into train_data and test_data
## Seed is set to keep the random sampling
## function need two input 1) Dataset 2) sample size by percent (from 0 - 1)
split_data <- function(availableData, percent_sample){
  set.seed(38)
  id <- sample(nrowoffulldata, size = percent_sample*nrowoffulldata)
  train_data <<- availableData[id, ]
  test_data <<- availableData[-id, ]
}

# call split_data(cleanData, 0.7) function to use cleanData that split 70% to train and 30% to t
est set
split_data(cleanData, 0.7)

# Just to see split data
(train_data); (test_data)
```

	PassengerId	Survived	Pcla…	Name
	<int>	<int>	<int>	<chr>
763	763	1	3	Barah, Mr. Hanna Assi
381	381	1	1	Bidois, Miss. Rosalie
403	403	0	3	Jussila, Miss. Mari Aina
207	207	0	3	Backstrom, Mr. Karl Alfred
313	313	0	2	Lahtinen, Mrs. William (Anna Sylfven)
145	145	0	2	Andrew, Mr. Edgardo Samuel

	PassengerId <int>	Survived <int>	Pcla… <int>	Name <chr>
822	822	1	3	Lulic, Mr. Nikola
333	333	0	1	Graham, Mr. George Edward
35	35	0	1	Meyer, Mr. Edgar Joseph
662	662	0	3	Badt, Mr. Mohamed

	PassengerId <int>	Survived <int>	Pclass <int>
2	2	1	1
4	4	1	1
7	7	0	1
19	19	0	3
24	24	1	1
40	40	1	3
41	41	0	3
42	42	0	2
50	50	0	3
51	51	0	3

1-10 of 215 rows | 1-4 of 13 columns Previous **1** 2 3 4 5 6 … 22 Next

Train Model

With in “caret”, the “glm” is used to fit generalized linear models and “logistic regression” is a part of it.

Logistic regression (AKA binary classification) have been used in this study because the model will predict binary outcome that is “Survived” and its values (0 for Not survived and 1 for survived).

There are many factor that can lead to survive such as Pclass, Sex, Age, SibSp, Parch and other. This model have taken variable above into account.

```
logistic_regression <- glm(Survived ~ Pclass + Sex + Age + SibSp + Parch, data = train_data, fam
ily = "binomial")

(logistic_regression)
```

```
##
## Call:  glm(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch,
##     family = "binomial", data = train_data)
##
## Coefficients:
## (Intercept)       Pclass      Sexmale          Age        SibSp        Parch
##     5.26968     -1.23370     -2.55112     -0.03845     -0.37241      0.02666
##
## Degrees of Freedom: 498 Total (i.e. Null);  493 Residual
## Null Deviance:       672.8
## Residual Deviance: 456.7     AIC: 468.7
```

```
#summary(model) is used to inspect model
summary(logistic_regression)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch,
##     family = "binomial", data = train_data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.26968    0.62708   8.404  < 2e-16 ***
## Pclass      -1.23370    0.16210  -7.611 2.72e-14 ***
## Sexmale     -2.55112    0.26252  -9.718  < 2e-16 ***
## Age         -0.03845    0.00940  -4.090 4.31e-05 ***
## SibSp       -0.37241    0.14556  -2.558   0.0105 *
## Parch        0.02666    0.14389   0.185   0.8530
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 672.78  on 498  degrees of freedom
## Residual deviance: 456.73  on 493  degrees of freedom
## AIC: 468.73
##
## Number of Fisher Scoring iterations: 5
```

As see from the summary, Number of parents and children of the passenger aboard (Parch) did not have statistics significance within the model due to P-value 0.85.

Test Model (AKA prediction)

Prediction is done by using probability from type = “response”. Then, “ifelse” is used to check conditions to categorize into two group: survived (Prob >= 0.5 showed as 1) and not survived (Prob < 0.5 showed as 0).

```
testmodel={} #create empty
testmodel$prob_survived <- predict(logistic_regression, newdata = test_data, type = "response")

testmodel$pred_survived <- ifelse(testmodel$prob_survived >= 0.5, 1, 0)
```

Model Evaluation

```
#Create confusion matrix
confusion <- table(testmodel$pred_survived, test_data$Survived, dnn=c("Predicted", "Actual"))
(confusion)
```

```
##          Actual
## Predicted   0   1
##         0 105  16
##         1  21  73
```

```
cat("Accuracy: ", (confusion[1, 1] + confusion[2, 2]) / sum(confusion), "\n")
```

```
## Accuracy:  0.827907
```

```
cat("Precision: ", confusion[2, 2] / (confusion[2, 1] + confusion[2, 2]), "\n")
```

```
## Precision:  0.7765957
```

```
Precision <- confusion[2, 2] / (confusion[2, 1] + confusion[2, 2])

cat("Recall: ", confusion[2, 2] / (confusion[1, 2] + confusion[2, 2]), "\n")
```

```
## Recall:  0.8202247
```

```
Recall <- confusion[2, 2] / (confusion[1, 2] + confusion[2, 2])

cat("F1 score: ", 2*(Precision*Recall)/(Precision+Recall), "\n")
```

```
## F1 score:  0.7978142
```

Summary

Based on model evaluation: the trained model have a capability to predict practically survived model, which are trained by Pclass, Sex, Age, SibSp, and Parch variables within titanic dataset, upto around 80 % of model accuracy and precision.