**TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT**

**KHOA HỆ THỐNG THÔNG TIN QUẢN LÝ**


# PHÂN TÍCH DỮ LIỆU VỚI R/PYTHON

## BÁO CÁO ĐỒ ÁN SEMINAR 28/07/2022

**Giảng viên bộ môn:** ThS. Nguyễn Quang Phúc

**Thực hiện bởi: Nhóm 6**

| | |
|---|---|
| Vũ Nhật Hoàng | K194060784 |
| Nguyễn Thị Thu | K194060820 |
| Lưu Thị Thúy | K194060822 |
| Nguyễn Thị Hoàng Thương | K194060826 |
| Trương Bảo Trân | K194060833 |

TP. Hồ Chí Minh, tháng 7 năm 2022

# MỤC LỤC

# DANH MỤC HÌNH ẢNH

# DANH MỤC BẢNG BIỂU

# CHƯƠNG 1: PHÂN TÍCH DỰ BÁO VỚI MÔ HÌNH HỒI QUY TUYẾN TÍNH (ĐƠN BIẾN, ĐA BIẾN)

## 1.1. Lý thuyết về mô hình hồi quy tuyến tính:

## 1.2. Bài toán đặt ra:

Thông qua dataset này, đầu ra của bài toán là dự đoán giá bán xe ô tô

- Trước tiên, chúng ta phải xem yếu tố nào có tác động nhiều hơn đến Giá bán xe và tiến hành chọn ra mô hình thể hiện tốt nhất dự tác động của yếu tố đối với Giá bán xe thông qua mô hình hồi quy tuyến tính đơn biến
- Thứ hai, chúng ta phải xây dựng được mô hình hồi quy đa biến tốt nhất của Giá bán xe. Từ đó, dự đoán Giá bán xe và kiểm tra độ chính xác của dự đoán.

Dataset: Car sales

Tác giả: GaganBhatia

Nguồn lấy dataset: https://www.kaggle.com/

Link của dataset: https://www.kaggle.com/datasets/gagandeep16/car-sales

Đây là Dataset Car Sales (Giá bán ô tô) bao gồm thông tin về các loại ô tô khác nhau. Dataset này đang được lấy từ Analytixlabs cho mục đích dự đoán.

Gồm 157 dòng và 16 cột:

- Manufacturer:
- Model:
- Sales_in_thousands
- __year_resale_value
- Vehicle_type
- Price_in_thousands
- Engine_size
- Horsepower
- Wheelbase
- Width
- Length

- Curb_weight
- Fuel_capacity
- Fuel_efficiency
- Latest_Launch
- Power_perf_factor

## 1.3. Phân tích dự báo với mô hình hồi quy tuyến tính đơn biến bằng R

### *1.3.1. Nạp thư viện*

```
library(plotly)

library(skimr)

library(zoo)

library(xts)

library(corrplot)

library(tidyr)

library(naniar)

library("Hmisc")
```

### *1.3.2. Tải dữ liệu*

```
df <- read.csv('./data/Car_sales_cleaning_final.csv')
```

| | X | Manufacturer | Model | Sales_in_thousands | X_year_resale_value | Price_in_thousands | Engine_size | Horsepower | Wheelbase | Width | Length | Curb_weight | Fuel_capacity | Fuel_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | Acura | Integra | 16.9190 | 16.3600 | 21.5000 | 1.80000 | 140 | 101.200 | 67.3000 | 172.400 | 2.63900 | 13.2000 | 28 |
| 2 | 1 | Acura | TL | 39.3840 | 19.8750 | 28.4000 | 3.20000 | 225 | 108.100 | 70.3000 | 192.900 | 3.51700 | 17.2000 | 25 |
| 3 | 2 | Acura | CL | 14.1140 | 18.2250 | 35.2000 | 3.20000 | 225 | 106.900 | 70.6000 | 192.000 | 3.47000 | 17.2000 | 26 |
| 4 | 3 | Acura | RL | 8.58800 | 29.7250 | 42.0000 | 3.50000 | 210 | 114.600 | 71.4000 | 196.600 | 3.85000 | 18.0000 | 22 |
| 5 | 4 | Audi | A4 | 20.3970 | 22.2550 | 23.9900 | 1.80000 | 150 | 102.600 | 68.2000 | 178.000 | 2.99800 | 16.4000 | 27 |
| 6 | 5 | Audi | A6 | 18.7800 | 23.5550 | 33.9500 | 2.80000 | 200 | 108.700 | 76.1000 | 192.000 | 3.56100 | 18.5000 | 22 |
| 7 | 6 | Audi | A8 | 1.38000 | 39.0000 | 62.0000 | 4.20000 | 310 | 113.000 | 74.0000 | 198.200 | 3.90200 | 23.7000 | 21 |
| 8 | 7 | BMW | 323i | 19.7470 | 33.8375 | 26.9900 | 2.50000 | 170 | 107.300 | 68.4000 | 176.000 | 3.17900 | 16.6000 | 26 |
| 9 | 8 | BMW | 328i | 9.23100 | 28.6750 | 33.4000 | 2.80000 | 193 | 107.300 | 68.5000 | 176.000 | 3.19700 | 16.6000 | 24 |
| 10 | 9 | BMW | 528i | 17.5270 | 36.1250 | 38.9000 | 2.80000 | 193 | 111.400 | 70.9000 | 188.000 | 3.47200 | 18.5000 | 25 |
| 11 | 10 | Buick | Century | 91.5610 | 12.4750 | 21.9750 | 3.10000 | 175 | 109.000 | 72.7000 | 194.600 | 3.36800 | 17.5000 | 25 |
| 12 | 11 | Buick | Regal | 39.3500 | 13.7400 | 25.3000 | 3.80000 | 240 | 109.000 | 72.7000 | 196.200 | 3.54300 | 17.5000 | 23 |
| 13 | 12 | Buick | Park Avenue | 27.8510 | 20.1900 | 31.9650 | 3.80000 | 205 | 113.800 | 74.7000 | 206.800 | 3.77800 | 18.5000 | 24 |
| 14 | 13 | Buick | LeSabre | 83.2570 | 13.3600 | 27.8850 | 3.80000 | 205 | 112.200 | 73.5000 | 200.000 | 3.59100 | 17.5000 | 25 |
| 15 | 14 | Cadillac | DeVille | 63.7290 | 22.5250 | 39.8950 | 4.60000 | 275 | 115.300 | 74.5000 | 207.200 | 3.97800 | 18.5000 | 22 |
| 16 | 15 | Cadillac | Seville | 15.9430 | 27.1000 | 44.4750 | 4.60000 | 275 | 112.200 | 75.0000 | 201.000 | 3.91050 | 18.5000 | 22 |
| 17 | 16 | Cadillac | Eldorado | 6.53600 | 25.7250 | 39.6650 | 4.60000 | 275 | 108.000 | 75.5000 | 200.600 | 3.84300 | 19.0000 | 22 |
| 18 | 17 | Cadillac | Catera | 11.1850 | 18.2250 | 31.0100 | 3.00000 | 200 | 107.400 | 70.3000 | 194.800 | 3.77000 | 18.0000 | 22 |
| 19 | 18 | Cadillac | Escalade | 14.7850 | 13.7375 | 46.2250 | 5.70000 | 255 | 117.500 | 77.0000 | 201.200 | 5.57200 | 30.0000 | 15 |
| 20 | 19 | Chevrolet | Cavalier | 145.519 | 9.25000 | 13.2600 | 2.20000 | 115 | 104.100 | 67.9000 | 180.900 | 2.67600 | 14.3000 | 27 |
| 21 | 20 | Chevrolet | Malibu | 135.126 | 11.2250 | 16.5350 | 3.10000 | 170 | 107.000 | 69.4000 | 190.400 | 3.05100 | 15.0000 | 25 |
| 22 | 21 | Chevrolet | Lumina | 24.6290 | 10.3100 | 18.8900 | 3.10000 | 175 | 107.500 | 72.5000 | 200.900 | 3.33000 | 16.6000 | 25 |

*Bảng 1: Bảng dữ liệu Car sale sau khi đã nạp vào PyCharm bằng ngôn ngữ R*

### *1.3.3. Kiểm tra thông tin dữ liệu:*

```
miss_var_summary(df)
```

skimr::skim(df)

| | variable | n_miss | pct_miss |
|---|---|---|---|
| | <chr> | <int> | <dbl> |
| 1 | X | 0 | 0 |
| 2 | Manufacturer | 0 | 0 |
| 3 | Model | 0 | 0 |
| 4 | Sales_in_thousands | 0 | 0 |
| 5 | X__year_resale_value | 0 | 0 |
| 6 | Price_in_thousands | 0 | 0 |
| 7 | Engine_size | 0 | 0 |
| 8 | Horsepower | 0 | 0 |
| 9 | Wheelbase | 0 | 0 |
| 10 | Width | 0 | 0 |
| 11 | Length | 0 | 0 |
| 12 | Curb_weight | 0 | 0 |
| 13 | Fuel_capacity | 0 | 0 |
| 14 | Fuel_efficiency | 0 | 0 |
| 15 | Latest_Launch | 0 | 0 |
| 16 | Power_perf_factor | 0 | 0 |

*Bảng 2: Dữ liệu Null có trong bảng*

```
-- Data Summary ------------------------
                           Values
Name                       df
Number of rows             157
Number of columns          16
-----------------------
Column type frequency:
  character                3
  numeric                  13
-----------------------
Group variables            None

-- Variable type: character -----------------------------------------------
  skim_variable n_missing complete_rate min max empty n_unique whitespace
1 Manufacturer          0             1   3  10     0       30          0
2 Model                 0             1   2  14     0      156          0
3 Latest_Launch         0             1   8  10     0      130          0
```

```
-- Variable type: numeric ------------------------------------------------------------------------
   skim_variable        n_missing complete_rate   mean     sd     p0    p25    p50    p75   p100 hist
 1 X                            0             1  78    45.5     0     39     78    117    156
  <U+2587><U+2587><U+2587><U+2587><U+2587>
 2 Sales_in_thousands           0             1  53.0  68.0     0.11  14.1   29.4   68.0  541.
  <U+2587><U+2581><U+2581><U+2581><U+2581>
 3 X__year_resale_value         0             1  18.5  11.4     5.16  11.5   14.0   19.9   67.6
  <U+2587><U+2582><U+2581><U+2581><U+2581>
 4 Price_in_thousands           0             1  27.4  14.3     9.24  18.1   23.4   32.0   85.5
  <U+2587><U+2585><U+2582><U+2581><U+2581>
 5 Engine_size                  0             1   3.06  1.04    1      2.3    3      3.5    8
  <U+2586><U+2587><U+2583><U+2581><U+2581>
 6 Horsepower                   0             1 186.   56.8    55    150    180    215    450
  <U+2583><U+2587><U+2583><U+2581><U+2581>
 7 Wheelbase                    0             1 108.    7.63   92.6  103    107    112.   139.
  <U+2583><U+2587><U+2585><U+2581><U+2581>
 8 Width                        0             1  71.2   3.45   62.6   68.4   70.6   73.5   79.9
  <U+2581><U+2587><U+2587><U+2585><U+2582>
 9 Length                       0             1 187.   13.4   149.   178.   188    196.   224.
  <U+2581><U+2585><U+2587><U+2585><U+2581>
10 Curb_weight                  0             1   3.38  0.628   1.90   2.98   3.37   3.82   5.57
```

```
11 Fuel_capacity                0             1  17.9   3.88   10.3   15.8   17.2   19.5   32
 <U+2583><U+2587><U+2583><U+2581><U+2581>
12 Fuel_efficiency              0             1  23.8   4.26   15     21     24     26     45
 <U+2583><U+2587><U+2582><U+2581><U+2581>
13 Power_perf_factor            0             1  77.3  25.1    23.3   60.7   72.3   90.2  188.
 <U+2583><U+2587><U+2583><U+2581><U+2581>
```

*Bảng 3: Data Summary*

### *1.3.4. Khai báo biến*

price <- as.numeric(unlist(c(df["Price_in_thousands"])))

sale <-  as.numeric(unlist(c(df['Sales_in_thousands'])))

resale <- as.numeric(unlist(c(df['X__year_resale_value'])))

enginesize <- as.numeric(unlist(c(df['Engine_size'])))

horsepower <- as.numeric(unlist(c(df['Horsepower'])))

wheelbase <- as.numeric(unlist(c(df['Wheelbase'])))

width <- as.numeric(unlist(c(df['Width'])))

length <- as.numeric(unlist(c(df['Length'])))

curbweight <- as.numeric(unlist(c(df['Curb_weight'])))

fuelcapacity <- as.numeric(unlist(c(df['Fuel_capacity'])))

fuelefficiency <- as.numeric(unlist(c(df['Fuel_efficiency'])))

power <- as.numeric(unlist(c(df['Power_perf_factor'])))

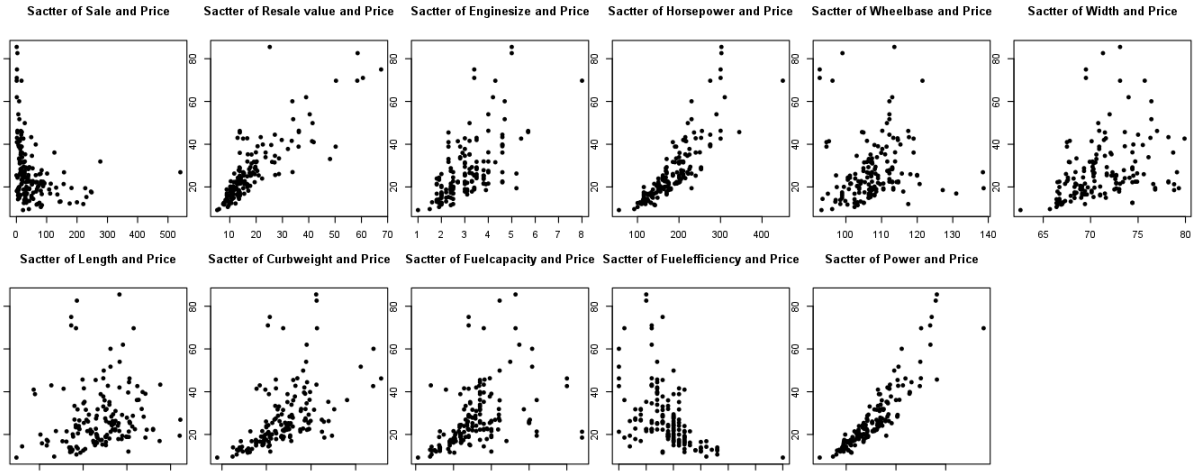
### *1.3.5. Trực quan hóa dữ liệu để xác định mô hình*

*1.3.5.1. Mối tương quan giữa các biến thông quan biểu đồ Scatter*

```
par(mar=c(1,1,5,1))

par(mfrow=c(2,6))

plot(sale, price,pch=16, xlab = "Sales", ylab = "Price", main = "Sactter of Sale and Price" )

plot(resale, price,pch=16, xlab = "Resale value", ylab = "Price", main = "Sactter of Resale value and Price" )

plot(enginesize, price,pch=16, xlab = "Enginesize", ylab = "Price", main = "Sactter of Enginesize and Price" )

plot(horsepower, price,pch=16, xlab = "Horsepower", ylab = "Price", main = "Sactter of Horsepower and Price" )

plot(wheelbase, price,pch=16, xlab = "Wheelbase", ylab = "Price", main = "Sactter of Wheelbase and Price" )

plot(width, price,pch=16, xlab = "Width", ylab = "Price", main = "Sactter of Width and Price" )

plot(length, price,pch=16, xlab = "Length", ylab = "Price", main = "Sactter of Length and Price" )

plot(curbweight,pch=16, price, xlab = "Curbweight", ylab = "Price", main = "Sactter of Curbweight and Price" )

plot(fuelcapacity,pch=16, price,xlab = "Fuelcapacity", ylab = "Price", main = "Sactter of Fuelcapacity and Price" )

plot(fuelefficiency,pch=16, price,xlab = "Fuelefficiency", ylab = "Price", main = "Sactter of Fuelefficiency and Price")

plot(power, price,pch=16,xlab = "Power", ylab = "Price", main = "Sactter of Power and Price" )

par(mfrow=c(1,1))
```

Nhận xét mối tương quan giữa các biến thông qua biểu đồ Scatter: Biểu đồ phân tán sử dụng các dấu chấm để thể hiện các giá trị (điểm giao nhau) của hai biến số khác nhau. Mục đích chủ yếu của biểu đồ Scatter trong tập dữ liệu này là để quan sát và thể hiện mối tương quan giữa 2 biến số là giá xe (price) và 11 thuộc tính khác () ứng với 11 biểu đồ Scatter. Trong đó biến phụ thuộc (price) chạy cố định trên trục tung và biến độc

lập chạy cố định dựa vào trục hoành. Các dấu chấm trong biểu đồ phân tán không chỉ thể hiện giá trị của một điểm dữ liệu mà còn thể hiện xu hướng khi chúng ta nhìn tổng thể toàn bộ tập dữ liệu.




*Hình 1: Scatter mối tương quan giữa các biến với Price*

*2.3.5.2. Mối tương quan giữa các biến thông quan biểu đồ heatmap*

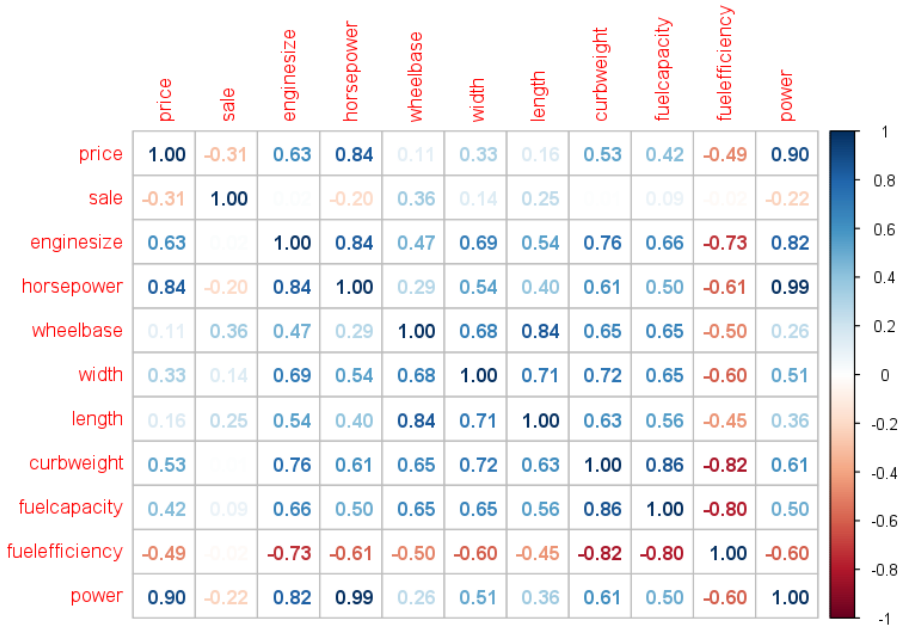
x <- data.frame(price, sale, enginesize, horsepower, wheelbase, width, length, curbweight, fuelcapacity, fuelefficiency, power)

print(x)

correlation <- cor(x, y = NULL, use = "everything", method = c("pearson"))

corrplot(correlation, method = "number")

Kiểm định lại bằng biểu đồ heatmap thể hiện tương quan bằng r-pearson ta có:




*Bảng 4: Heatmap tương quan các biến*

Hệ số tương quan pearson (r) chỉ có ý nghĩa khi và chỉ khi mức ý nghĩa quan sát (sig.) nhỏ hơn mức ý nghĩa α = 5%.

Nếu r nằm trong khoảng từ 0,50 đến ± 1, thì nó được cho là tương quan mạnh.

Nếu r nằm trong khoảng từ 0,30 đến ± 0,49, thì nó được gọi là tương quan trung bình.

Nếu r nằm dưới ± 0,29, thì nó được gọi là một mối tương quan yếu.

Trên đồ thị phân tán Scatter, nếu r = -1 dữ liệu sẽ phân bổ trên một đường thẳng với độ dốc âm, r = 1 dữ liệu sẽ phân bổ trên một đường thẳng với độ dốc dương.

*2.3.5.3. Đưa ra nhận xét*

Nhìn vào biểu đồ Scatter và biểu đồ heatmap ta có nhận xét sự tương quan giữa các biến với Price như sau:

- Sale: -0.31 tương quan âm và tương quan trung bình
- Enginesize: 0.63 tương quan dương và tương quan mạnh
- Horsepowwer 0.84 tương quan dương và tương quan mạnh
- Wheelbase: 0.11 tương quan dương và tương quan yếu
- Width 0.33 tương quan dương và tương quan trung bình
- Length 0.16 tương quan dương, tương quan yếu
- Curbweight: 0.53 tương quan dương, tương quan mạnh

- Fuelcapacity; 0.42 tương quan dương, tương quan trung bình
- Fuelefficency: -0,49 tương quan âm, tương quan trung bình
- Power: 0.9 tương quan dương, tương quan mạnh

Trong tất cả các biến vừa khảo sát trên thì biến Power (0.9) có tương quan mạnh nhất đối với biến Price. Chính vì vậy ta sẽ tiến hành xây dựng mô hình hồi quy đơn biến với biến độc lập là Power và biến phụ thuộc là  Price. Và sau đó tiến hành dự đoán thông qua mô hình này.

### *1.3.6. Xây dựng mô hình hồi quy tuyến tính đơn biến*

Sau khi đã xác định mối tương quan giữa các biến: ta có thể xây dựng mô hình dự báo giá xe (Price) dựa vào biến Power bằng mô hình hồi quy đơn biến như sau:

```
Call:
lm(formula = price ~ power, data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-14.5773  -4.5539   0.0283   2.6507  25.5158

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -11.9589     1.6488  -7.253 1.82e-11 ***
power         0.5099     0.0203  25.120  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.359 on 155 degrees of freedom
Multiple R-squared:  0.8028,    Adjusted R-squared:  0.8015
F-statistic:   631 on 1 and 155 DF,  p-value: < 2.2e-16
```

*Bảng 5: Mô hình hồi quy đơn biến*

Nhìn vào biểu đồ có thể thấy giá trị p (ở đây là < 2,2e-16, hoặc gần như bằng không), sẽ cho biết liệu mô hình có ý nghĩa thống kê, phù hợp với dữ liệu hay không.

Một hồi quy tuyến tính đơn giản dự đoán giá xe (price) (biến phụ thuộc) từ từ hệ số công suất (power) của xe (biến độc lập) có $R^2$ là 0,8028. Từ giá trị $R^2$ này, chúng ta biết rằng:

- 80,28% phương sai trong giá xe được dự đoán theo hệ số công suất của xe
- 19,72% phương sai trong giá xe là không giải thích được bằng mô hình

Hệ số công suất của xe có ảnh hưởng lớn đến giá xe

Mô hình hồi quy tuyến tính đơn biến có dạng như sau:

**Prices = -11.9589 + 0,5099 Power**

Ý nghĩa của mô hình: Điều này có nghĩa là cứ tăng 1 đơn vị Hệ số công suất (Power), thì Giá xe (Price) tăng 0,5099 đơn vị.

### *1.3.7. Biểu đồ minh họa mô hình hồi quy tuyến tính*

plot(power, price, pch=20, xlab = "Power", ylab = "Price", main = "Linear Regression of Power and Price")

abline(model, col="red", lwd=3)


*Bảng 6: Biểu đồ minh họa mô hình hồi quy tuyến tính*

### *1.3.8. Dự báo*

```
redict_values <- data.frame(power = c(200,150,300))

pred <- predict.lm(model, predict_values)
```

| 1 | 2 | 3 |
|---|---|---|
| 90.01498 | 64.52151 | 141.00192 |

*Bảng 7: Dự báo*

Ta có:

Với Hệ số công suất = 200 thì giá xe là 90.01498 ngàn đô

Với Hệ số công suất = 100 thì giá xe là 64.01498 ngàn đô

Với Hệ số công suất = 300 thì giá xe là 141.01498 ngàn đô

## 1.4. Phân tích dự báo với mô hình hồi quy tuyến tính đa biến với R

### *1.4.1. Nạp thư viện*

```
install.packages("car")

install.packages("ggplot")

#Import library

library(plotly)

library(skimr)

library(zoo)

library(xts)

library(corrplot)

library(tidyr)

library(naniar)

library("Hmisc")

library("car") #package của function VIF

library(ggplot2)
```

### *1.4.2. Tải  dữ liệu*

```
df <- read.csv('./data/Car_sales_cleaning_final.csv')
```

### *1.4.3. Kiểm tra thông tin dữ liệu*

```
miss_var_summary(df)

skimr::skim(df)
```

| | variable | n_miss | pct_miss |
|---|---|---|---|
| | <chr> | <int> | <dbl> |
| 1 | X | 0 | 0 |
| 2 | Manufacturer | 0 | 0 |
| 3 | Model | 0 | 0 |
| 4 | Sales_in_thousands | 0 | 0 |
| 5 | X__year_resale_value | 0 | 0 |
| 6 | Price_in_thousands | 0 | 0 |
| 7 | Engine_size | 0 | 0 |
| 8 | Horsepower | 0 | 0 |
| 9 | Wheelbase | 0 | 0 |
| 10 | Width | 0 | 0 |
| 11 | Length | 0 | 0 |
| 12 | Curb_weight | 0 | 0 |
| 13 | Fuel_capacity | 0 | 0 |
| 14 | Fuel_efficiency | 0 | 0 |
| 15 | Latest_Launch | 0 | 0 |
| 16 | Power_perf_factor | 0 | 0 |

Dữ liệu Null có trong bảng

```
-- Data Summary ------------------------
                           Values
Name                       df
Number of rows             157
Number of columns          16
-----------------------
Column type frequency:
  character                3
  numeric                  13
-----------------------
Group variables            None

-- Variable type: character -----------------------------------------------
  skim_variable n_missing complete_rate min max empty n_unique whitespace
1 Manufacturer          0             1   3  10     0       30          0
2 Model                 0             1   2  14     0      156          0
3 Latest_Launch         0             1   8  10     0      130          0
```

```
-- Variable type: numeric ------------------------------------------------------------------------
   skim_variable        n_missing complete_rate   mean     sd     p0    p25    p50    p75   p100 hist
 1 X                            0             1  78    45.5     0     39     78    117    156
  <U+2587><U+2587><U+2587><U+2587><U+2587>
 2 Sales_in_thousands           0             1  53.0  68.0     0.11  14.1   29.4   68.0  541.
  <U+2587><U+2581><U+2581><U+2581><U+2581>
 3 X__year_resale_value         0             1  18.5  11.4     5.16  11.5   14.0   19.9   67.6
  <U+2587><U+2582><U+2581><U+2581><U+2581>
 4 Price_in_thousands           0             1  27.4  14.3     9.24  18.1   23.4   32.0   85.5
  <U+2587><U+2585><U+2582><U+2581><U+2581>
 5 Engine_size                  0             1   3.06  1.04    1      2.3    3      3.5    8
  <U+2586><U+2587><U+2583><U+2581><U+2581>
 6 Horsepower                   0             1 186.   56.8    55    150    180    215    450
  <U+2583><U+2587><U+2583><U+2581><U+2581>
 7 Wheelbase                    0             1 108.    7.63   92.6  103    107    112.   139.
  <U+2583><U+2587><U+2585><U+2581><U+2581>
 8 Width                        0             1  71.2   3.45   62.6   68.4   70.6   73.5   79.9
  <U+2581><U+2587><U+2587><U+2585><U+2582>
 9 Length                       0             1 187.   13.4   149.   178.   188    196.   224.
  <U+2581><U+2585><U+2587><U+2585><U+2581>
10 Curb_weight                  0             1   3.38  0.628   1.90   2.98   3.37   3.82   5.57
```

```
11 Fuel_capacity                0             1  17.9   3.88   10.3   15.8   17.2   19.5   32
 <U+2583><U+2587><U+2583><U+2581><U+2581>
12 Fuel_efficiency              0             1  23.8   4.26   15     21     24     26     45
 <U+2583><U+2587><U+2582><U+2581><U+2581>
13 Power_perf_factor            0             1  77.3  25.1    23.3   60.7   72.3   90.2  188.
 <U+2583><U+2587><U+2583><U+2581><U+2581>
```

### *1.4.4. Khai báo biến*

price <- as.numeric(unlist(c(df["Price_in_thousands"])))

sale <-  as.numeric(unlist(c(df['Sales_in_thousands'])))

resale <- as.numeric(unlist(c(df['X__year_resale_value'])))

enginesize <- as.numeric(unlist(c(df['Engine_size'])))

horsepower <- as.numeric(unlist(c(df['Horsepower'])))

wheelbase <- as.numeric(unlist(c(df['Wheelbase'])))

width <- as.numeric(unlist(c(df['Width'])))

length <- as.numeric(unlist(c(df['Length'])))

curbweight <- as.numeric(unlist(c(df['Curb_weight'])))

fuelcapacity <- as.numeric(unlist(c(df['Fuel_capacity'])))

fuelefficiency <- as.numeric(unlist(c(df['Fuel_efficiency'])))

power <- as.numeric(unlist(c(df['Power_perf_factor'])))

### *1.4.5. Mối tương quan các biến thông qua mô hình heatmap*

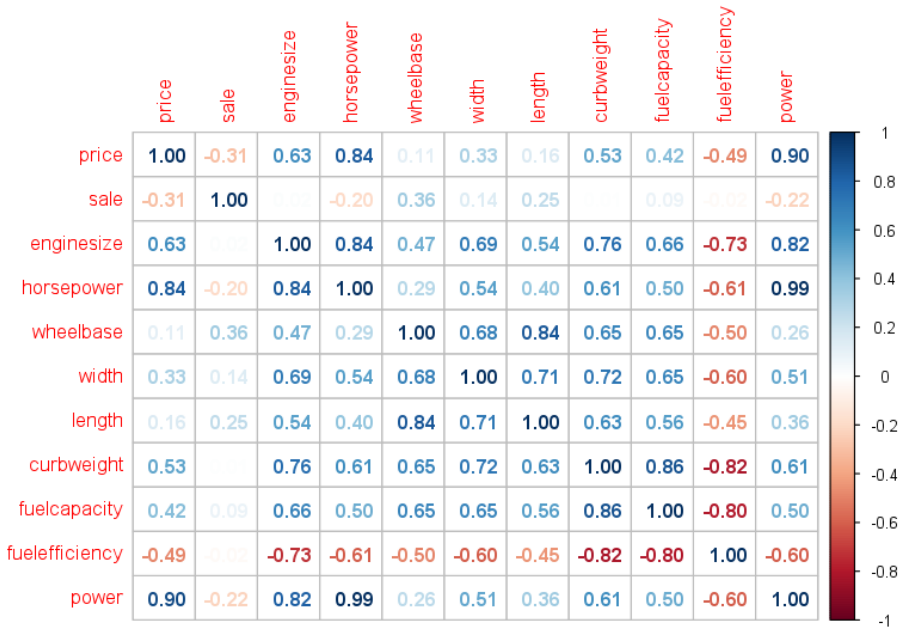
x <- data.frame(price, sale, enginesize, horsepower, wheelbase, width, length, curbweight, fuelcapacity, fuelefficiency, power)

print(x)

correlation <- cor(x, y = NULL, use = "everything", method = c("pearson"))

corrplot(correlation, method = "number")

Kiểm định lại bằng biểu đồ heatmap thể hiện tương quan bằng r-pearson ta có:




| | price | sale | enginesize | horsepower | wheelbase | width | length | curbweight | fuelcapacity | fuelefficiency | power |
|---|---|---|---|---|---|---|---|---|---|---|---|
| price | 1.00 | -0.31 | 0.63 | 0.84 | 0.11 | 0.33 | 0.16 | 0.53 | 0.42 | -0.49 | 0.90 |
| sale | -0.31 | 1.00 | | -0.20 | 0.36 | 0.14 | 0.25 | | 0.09 | | -0.22 |
| enginesize | 0.63 | | 1.00 | 0.84 | 0.47 | 0.69 | 0.54 | 0.76 | 0.66 | -0.73 | 0.82 |
| horsepower | 0.84 | -0.20 | 0.84 | 1.00 | 0.29 | 0.54 | 0.40 | 0.61 | 0.50 | -0.61 | 0.99 |
| wheelbase | 0.11 | 0.36 | 0.47 | 0.29 | 1.00 | 0.68 | 0.84 | 0.65 | 0.65 | -0.50 | 0.26 |
| width | 0.33 | 0.14 | 0.69 | 0.54 | 0.68 | 1.00 | 0.71 | 0.72 | 0.65 | -0.60 | 0.51 |
| length | 0.16 | 0.25 | 0.54 | 0.40 | 0.84 | 0.71 | 1.00 | 0.63 | 0.56 | -0.45 | 0.36 |
| curbweight | 0.53 | | 0.76 | 0.61 | 0.65 | 0.72 | 0.63 | 1.00 | 0.86 | -0.82 | 0.61 |
| fuelcapacity | 0.42 | 0.09 | 0.66 | 0.50 | 0.65 | 0.65 | 0.56 | 0.86 | 1.00 | -0.80 | 0.50 |
| fuelefficiency | -0.49 | | -0.73 | -0.61 | -0.50 | -0.60 | -0.45 | -0.82 | -0.80 | 1.00 | -0.60 |
| power | 0.90 | -0.22 | 0.82 | 0.99 | 0.26 | 0.51 | 0.36 | 0.61 | 0.50 | -0.60 | 1.00 |

Nhìn vào biểu đồ heatmap ta có nhận xét sự tương quan giữa các biến với Price như sau:

- Sale: -0.31 tương quan âm và tương quan trung bình
- Enginesize: 0.63 tương quan dương và tương quan mạnh
- Horsepowwer 0.84 tương quan dương và tương quan mạnh
- Wheelbase: 0.11 tương quan dương và tương quan yếu
- Width 0.33 tương quan dương và tương quan trung bình
- Length 0.16 tương quan dương, tương quan yếu
- Curbweight: 0.53 tương quan dương, tương quan mạnh
- Fuelcapacity; 0.42 tương quan dương, tương quan trung bình
- Fuelefficency: -0,49 tương quan âm, tương quan trung bình
- Power: 0.9 tương quan dương, tương quan mạnh

Vì tất cả các biến đều có mức độ tương quan với giá xe nhất định nên khi xây dựng mô hình hồi quy đa biến ta chọn hết tất cả các biến vào mô hình.

### *1.4.6. Chia tập dữ liệu Train với Test*

sample <- sample(c(TRUE, FALSE), nrow(df), replace = T, prob = c(0.9,0.1))

train <- df[sample, ]

test <- df[!sample, ]

### *1.4.7. Xây dựng mô hình hồi quy tuyến tính đa biến:*

*1.4.7.1. Mô hình 1:*

model_1 <- lm(price ~ sale + resale + enginesize + horsepower + wheelbase + width + length + curbweight + fuelcapacity + fuelefficiency + power , train)

summary(model_1)

```
lm(formula = price ~ sale + resale + enginesize + horsepower +
    wheelbase + width + length + curbweight + fuelcapacity +
    fuelefficiency + power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2407 -0.1001 -0.0357  0.0257  6.5169

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    -0.1900753  1.4594615  -0.130    0.897
sale           -0.0001325  0.0007859  -0.169    0.866
resale         -0.0007752  0.0076390  -0.101    0.919
enginesize     -0.7400776  0.1062471  -6.966 1.06e-10 ***
horsepower     -0.9051248  0.0092459 -97.894  < 2e-16 ***
wheelbase      -0.0028325  0.0129353  -0.219    0.827
width          -0.0162775  0.0226259  -0.719    0.473
length          0.0009517  0.0069765   0.136    0.892
curbweight      0.1899974  0.1935285   0.982    0.328
fuelcapacity    0.0035297  0.0255759   0.138    0.890
fuelefficiency  0.0245525  0.0207317   1.184    0.238
power           2.5697159  0.0221297 116.121  < 2e-16 ***
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5518 on 145 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic:  9475 on 11 and 145 DF,  p-value: < 2.2e-16
```

*1.4.7.2. Mô hình 2:*

Vì p-value của Resale > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Resale

model_2 <- lm(price ~ sale +  enginesize + horsepower + wheelbase + width + length + curbweight + fuelcapacity + fuelefficiency + power , train)

summary(model_2)

```
Call:
lm(formula = price ~ sale + enginesize + horsepower + wheelbase +
    width + length + curbweight + fuelcapacity + fuelefficiency +
    power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2417 -0.1017 -0.0372  0.0250  6.5174

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    -0.2217612  1.4208305   -0.156    0.876
sale           -0.0001371  0.0007820   -0.175    0.861
enginesize     -0.7386766  0.1049886   -7.036 7.12e-11 ***
horsepower     -0.9046574  0.0079902 -113.220  < 2e-16 ***
wheelbase      -0.0026979  0.0128234   -0.210    0.834
width          -0.0161447  0.0225114   -0.717    0.474
length          0.0009622  0.0069520    0.138    0.890
curbweight      0.1937453  0.1893269    1.023    0.308
fuelcapacity    0.0030710  0.0250879    0.122    0.903
fuelefficiency  0.0246675  0.0206304    1.196    0.234
power           2.5683332  0.0173787  147.787  < 2e-16 ***
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.55 on 146 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic: 1.049e+04 on 10 and 146 DF,  p-value: < 2.2e-16
```

*1.4.7.3. Mô hình 3:*

Vì p-value của Fuel capacity và Fuel efficiency > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Fuel capacity, Fuel efficiency

model_3 <- lm(price ~ sale +  enginesize + horsepower + wheelbase + width + length + curbweight +  power , train)

summary(model_3)

```
Call:
lm(formula = price ~ sale + enginesize + horsepower + wheelbase +
    width + length + curbweight + power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2284 -0.0988 -0.0404  0.0128  6.5852

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.6106293  1.2471202    0.490    0.625
sale        -0.0001983  0.0007790   -0.255    0.799
enginesize  -0.7563129  0.1036553   -7.296 1.67e-11 ***
horsepower  -0.9062204  0.0078399 -115.591  < 2e-16 ***
wheelbase   -0.0037852  0.0123737   -0.306    0.760
width       -0.0156815  0.0224552   -0.698    0.486
length       0.0029083  0.0067017    0.434    0.665
curbweight   0.0863785  0.1429762    0.604    0.547
power        2.5714565  0.0171072  150.314  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 0.5491 on 148 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic: 1.316e+04 on 8 and 148 DF,  p-value: < 2.2e-16
```

*1.4.7.4. Mô hình 4*

Vì p-value của Sale > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Sale

model <- lm(price ~ enginesize + horsepower + wheelbase + width + length + curbweight + power , train)

summary(model)

```
Call:
lm(formula = price ~ enginesize + horsepower + wheelbase + width +
    length + curbweight + power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2221 -0.0949 -0.0383  0.0082  6.5880

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.663404   1.225906    0.541    0.589
enginesize  -0.763242   0.099704   -7.655 2.25e-12 ***
horsepower  -0.906207   0.007815 -115.956  < 2e-16 ***
wheelbase   -0.004754   0.011737   -0.405    0.686
width       -0.015656   0.022384   -0.699    0.485
length       0.002957   0.006678    0.443    0.658
curbweight   0.094716   0.138738    0.683    0.496
power        2.571719   0.017022  151.079  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.5473 on 149 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic: 1.513e+04 on 7 and 149 DF,  p-value: < 2.2e-16
```

*1.4.7.5. Mô hình 5*

Vì p-value của Wheelbase > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Wheelbase

model <- lm(price ~ enginesize + horsepower +  width + length + curbweight +  power , train)

summary(model)

```
lm(formula = price ~ enginesize + horsepower + width + length +
    curbweight + power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2376 -0.0991 -0.0361  0.0086  6.5953

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.615684   1.216826    0.506    0.614
enginesize  -0.761891   0.099371   -7.667 2.05e-12 ***
horsepower  -0.906159   0.007792 -116.288  < 2e-16 ***
width       -0.017100   0.022037   -0.776    0.439
length       0.001196   0.005053    0.237    0.813
curbweight   0.078068   0.132139    0.591    0.556
power        2.571886   0.016970  151.556  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5458 on 150 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic: 1.776e+04 on 6 and 150 DF,  p-value: < 2.2e-16
```

*1.4.7.6. Mô hình 6*

Vì p-value của Fuel capacity và Length > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Length

model <- lm(price ~ enginesize + horsepower +  width + curbweight +  power , train)

summary(model)

```
Call:
lm(formula = price ~ enginesize + horsepower + width + curbweight +
    power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2391 -0.1003 -0.0357  0.0119  6.5978

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.659707   1.198761    0.550    0.583
enginesize  -0.762756   0.098992   -7.705 1.61e-12 ***
horsepower  -0.905597   0.007398 -122.411  < 2e-16 ***
width       -0.015062   0.020221   -0.745    0.458
curbweight   0.087946   0.124981    0.704    0.483
power        2.570586   0.016006  160.599  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5441 on 151 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9985
F-statistic: 2.144e+04 on 5 and 151 DF,  p-value: < 2.2e-16
```

*1.4.7.7. Mô hình 7*

Vì p-value của Curbweight > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Curbweight

model <- lm(price ~ enginesize + horsepower + width + power , train)

summary(model)

```
Call:
lm(formula = price ~ enginesize + horsepower + width + power,
    data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2502 -0.0810 -0.0426  0.0025  6.6194

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)  0.418161   1.146654    0.365    0.716
enginesize  -0.732582   0.089074   -8.224 8.13e-14 ***
horsepower  -0.907335   0.006962 -130.331  < 2e-16 ***
width       -0.008315   0.017774   -0.468    0.641
power        2.574343   0.015065  170.888  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5432 on 152 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9986
F-statistic: 2.689e+04 on 4 and 152 DF,  p-value: < 2.2e-16
```

*1.4.7.8. Mô hình 8*

Vì p-value của Width > 0.05 => Không có ý nghãi về mặt thống kê => Bỏ giá trị Width

model <- lm(price ~ enginesize + horsepower +  power , train)

summary(model)

```
Call:
lm(formula = price ~ enginesize + horsepower + power, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-0.2101 -0.0753 -0.0392 -0.0034  6.6289

Coefficients:
            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -0.11283    0.16237   -0.695    0.488
enginesize  -0.75305    0.07739   -9.730   <2e-16 ***
horsepower  -0.90784    0.00686 -132.335   <2e-16 ***
power        2.57558    0.01479  174.131   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5418 on 153 degrees of freedom
Multiple R-squared:  0.9986,    Adjusted R-squared:  0.9986
F-statistic: 3.604e+04 on 3 and 153 DF,  p-value: < 2.2e-16
```

Nhìn vào mô hình có thể thấy 3 biến này đều có giá trị P value đều nhỏ hơn 0.05 (mức ý nghĩa 5%)

Suy ra ba biến này có ý nghĩa về mặt thống kế đối với mô hình này

**Price = -0.75305 Enginesize - 0.90784 Horsepower + 2.57558 Power - 0.11283**

### *1.4.8. Kiểm  định mô hình hồi quy đa biến*

Giữa các biến độc lập không có mối quan hệ đa cộng tuyến hoàn hảo

vif(model)

Theo Gujarati và Porter (2009) chỉ ra một số dấu hiệu của hiện tượng đa cộng tuyển trong mô hình khi:

(1) VIF >= 10

(2) Hệ số tương quan r của bất kì cắp biến nào trong mô hình lớn hơn 0.8

Theo ta thấy thì giữa biến Horsepowwer và biến Power có sự đa cộng tuyến vô cùng lớn

```
enginesize horsepower      power
  3.455221  80.604595  73.144607
```

Xây dựng thêm hai mô hình giữa biến

- Enginesive và Horsepower với Price

- Enginesive và Power với Price

### *1.4.8.1. Mô hình Enginesive và Horsepower với Price*

```
Call:
lm(formula = price ~ enginesize + horsepower, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-15.092  -4.212  -0.432   2.251  34.260

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -11.09205    2.10485  -5.270 4.54e-07 ***
enginesize   -3.34694    1.06834  -3.133  0.00207 **
horsepower    0.26181    0.01961  13.353  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.622 on 154 degrees of freedom
Multiple R-squared:  0.7185,    Adjusted R-squared:  0.7149
F-statistic: 196.6 on 2 and 154 DF,  p-value: < 2.2e-16
```

```
enginesize horsepower
  3.327215   3.327215
```

*1.4.8.2. Mô hình Enginesive và Power với Price*

```
Call:
lm(formula = price ~ enginesize + power, data = train)

Residuals:
     Min       1Q   Median       3Q      Max
-13.0356  -2.8685  -0.3174   1.8345  24.5001

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -10.03509    1.54252  -6.506 1.03e-09 ***
enginesize   -4.39093    0.77485  -5.667 6.96e-08 ***
power         0.65903    0.03219  20.476  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.803 on 154 degrees of freedom
Multiple R-squared:  0.8368,    Adjusted R-squared:  0.8347
F-statistic: 394.9 on 2 and 154 DF,  p-value: < 2.2e-16
```

```
enginesize      power
   3.01928    3.01928
```

*1.4.8.3. Kết luận*

Cả hai mô hình trên đều có hệ số VIF < 10 nên không có hiện tượng đa cộng tuyến giữa các biến này

R square hiệu chỉnh là 83.68 % VIF đều bằng 3.01928 đều nhỏ hơn 10.

Một hồi quy tuyến tính đơn giản dự đoán giá xe (price) (biến phụ thuộc) từ từ hệ số công suất (power) của xe (biến độc lập) và dung tích xi lanh (Enginesize) có $R^2$ là 0,8368. Từ giá trị $R^2$ này, chúng ta biết rằng:

- 83,68% phương sai trong giá xe được dự đoán theo hệ số công suất và dung tích xi lanh của xe
- 16,32% phương sai trong giá xe là không giải thích được bằng mô hình

Hệ số công suất và Dung tích xi lanh của xe có ảnh hưởng lớn đến giá xe

Do đó chọn mô hình Enginesive và Power với Price

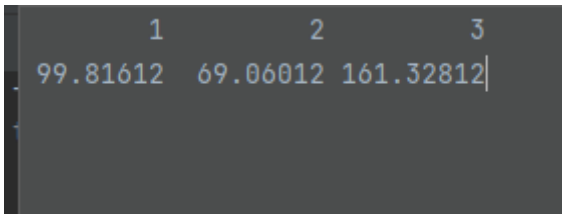
Kết luận: Ta có mô hình hồi quy đa biến như sau:

**Price = -4.39093 Enginesize + 0.65903 Power - 10.03509**

Ý nghĩa mô hình: Điều này có nghĩa là cứ tăng 1 đơn vị Dung tích xi lanh, thì Giá xe (price) giảm 4.3903 đơn vị. Trong khi đó, cứ tăng 1 đơn vị Hệ số xông suất thì Giá xe (Price) tăng 0.65903 đơn vị.

### *1.4.9. Dự báo*

```
predict_values <- data.frame(power = c(200,150,300), enginesize = c(5,4.5,6))

pred_da <- predict(model, predict_values)
```




Ta có:

Với Hệ số công suất = 200 và Dung tích xi lanh = 5 thì Giá xe = 99.81612 ngàn đô

Với Hệ số công suất = 150 và Dung tích xi lanh = 5 thì Giá xe = 69.06012 ngàn đô

Với Hệ số công suất = 300 và Dung tích xi lanh = 5 thì Giá xe = 161.32812 ngàn đô

# CHƯƠNG 2: PHÂN TÍCH DỰ BÁO VỚI HỒI QUY LOGISTIC

## 2.1. Lý thuyết về mô hình hồi quy logistic:

Hồi quy logistic: là một kỹ thuật thống kê xem xét mối liên hệ giữa biến độc lập (biến liên tục hoặc nhị phân) và biến phụ thuộc (biến nhị phân).

$y = \alpha + \beta x + \varepsilon$

y: biến phụ thuộc với 2 trạng thái (0/1; true/false; yes/no)

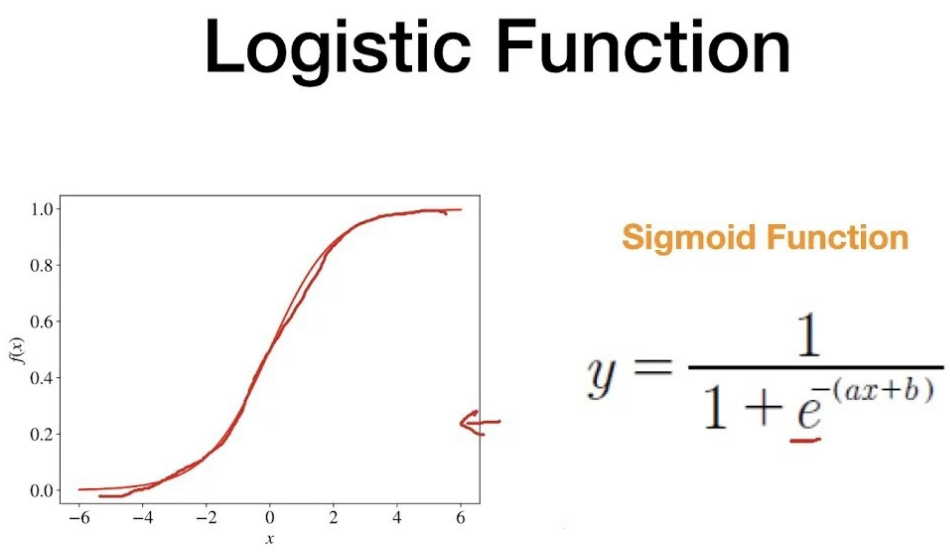
Mô hình hồi quy logistic được phát biểu như sau:

$\log(p/(1-p)) = \alpha + \beta x + \varepsilon$

p là xác suất biến cố xảy ra và 1-p là xác suất biến cố không xảy ra

Xác suất tiên lượng theo trị số của x:

$p = e^{(\alpha + \beta x)}/(1 + e^{(\alpha + \beta x)})$




*Hình 2: Đồ thị mô hình hồi quy logistic*

## 2.2. Bài toán đặt ra:

Đầu ra của bài toán là dự đoán hành khách có thường xuyên đi du lịch hay không dựa trên các biến độc lập thu được như Age, Frequent Flyer, Annual Income Class, Service Optd, Account Synced to Social Media, Book Hotel or Not và biến phụ thuộc nhị phân Target.

## 2.3. Mô hình hồi quy logistic đơn biến bằng R:

### *2.3.1. Nhập thư viện*

```
library(tidyverse)

library(caret)

library(DataExplorer)

library(caTools)
```

### *2.3.2. Tải dữ liệu*

```
getwd()

df <- read.csv("./Data/Customertravel_Clean.csv")

str(df)

summary(df)
```

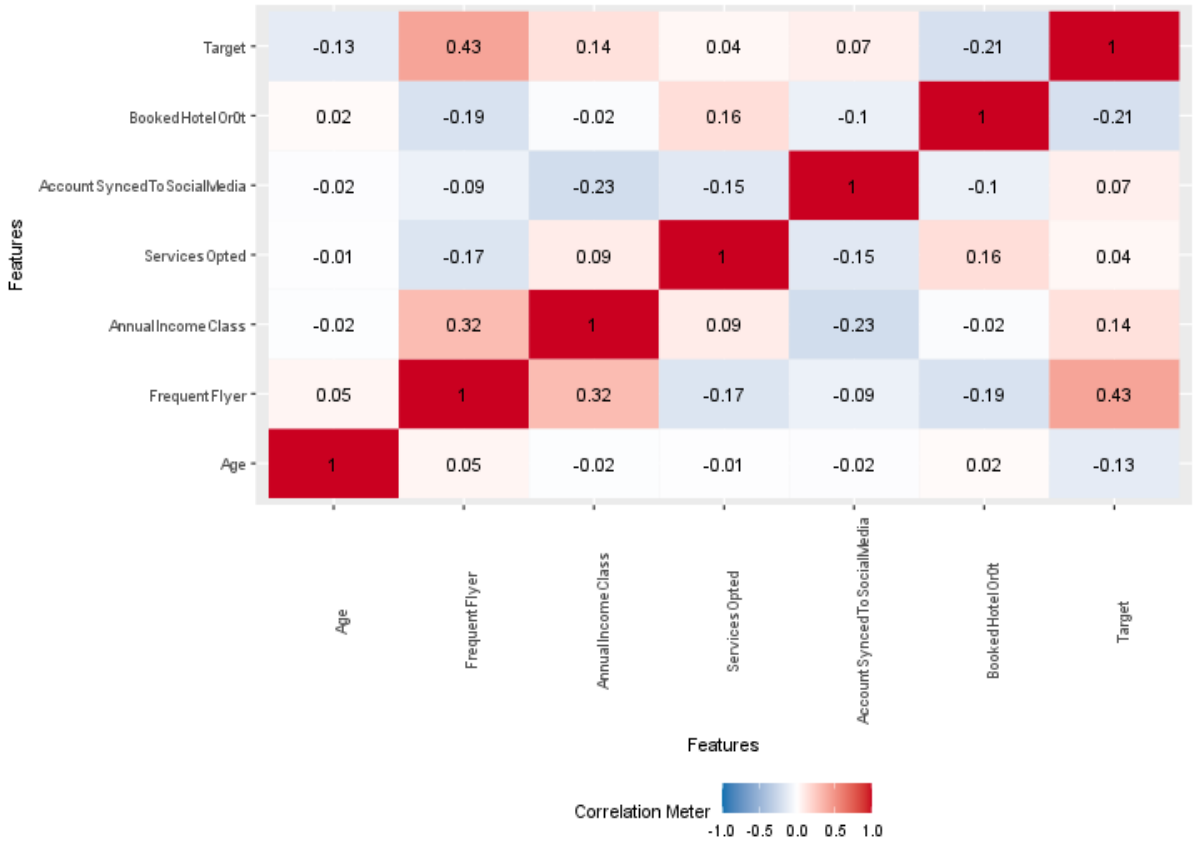
Dữ liệu được dùng ở đây là bộ dữ liệu Customertravel.csv, tuy nhiên đã được mã hóa thành các số như sau:

- FrequentFlyer: Yes = 1, No = 0
- AnnualIncomeClass: Low Income = 1, Middle Income = 2, High Income = 3
- AccountSyncedToSocialMedia: Yes = 1, No = 0
- BookedHotelOrNot: Yes = 1, No = 0

```
> df <- read.csv("./Data/Customertravel_Clean.csv")
> str(df)
'data.frame':   954 obs. of  7 variables:
 $ Age                      : int  34 34 37 30 30 27 34 34 30 36 ...
 $ FrequentFlyer            : int  0 1 0 0 0 1 0 0 0 1 ...
 $ AnnualIncomeClass        : int  2 1 2 2 1 3 2 1 1 3 ...
 $ ServicesOpted            : int  6 5 3 2 1 1 4 2 3 1 ...
 $ AccountSyncedToSocialMedia: int  0 1 1 0 0 0 1 1 0 0 ...
 $ BookedHotelOr0t          : int  1 0 0 0 0 1 1 0 1 0 ...
 $ Target                   : int  0 1 0 0 0 1 0 1 0 1 ...
```

### *2.3.3. Tương quan dữ liệu*

```
plot_correlation(na.omit(df), maxcat = 5L)
```




*Hình 3: Biểu đồ thể hiện sự tương quan giữa các biến*

Dựa vào biểu đồ trên, ta có thể thấy được FrequentFlyer có sự tương quan lớn nhất đến biến Target là 0.43

Nên ta lựa chọn biến độc lập là FrequentFlyer, tương quan với biến phụ thuộc Target

### *2.3.4. Chia tập train, test với tỉ lệ 90-10*

```
sample <- sample(c(TRUE, FALSE), nrow(df), replace = T, prob = c(0.9,0.1))
train <- df[sample, ]
test <- df[!sample, ]
```

### *2.3.5. Xây dựng model thể hiện mối quan hệ giữa Target và FrequentFlyer*

```
model <- glm(Target ~FrequentFlyer, data=train, family = binomial)
summary(model)
```

```
Call:
glm(formula = Target ~ FrequentFlyer, family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1968  -0.4729  -0.4729  -0.4729   2.1196

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    -2.1345     0.1322  -16.15   <2e-16 ***
FrequentFlyer   2.1800     0.1806   12.07   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 935.15  on 868  degrees of freedom
Residual deviance: 774.36  on 867  degrees of freedom
AIC: 778.36

Number of Fisher Scoring iterations: 4
```

*Hình 4: Hệ số của mô hình hồi quy Logistic đơn biến*

Ta ra được các hệ số của mô hình là intercept : -2.1345, FrequentFlyer: 2.1800

Từ đó suy ra được hàm số có dạng p = e^(-2.1345 + 2.1800 * FrequentFlyer)/(1+ e^(-2.1345 + 2.1800 * FrequentFlyer))

Có nghĩa là với mỗi giá trị FrequentFlyer ta có thể dự đoán xác suất hành khách có thường xuyên đi du lịch hay không. Hệ số của FrequentFlyer là giá trị dương, điều này có nghĩa là sự gia tăng FrequentFlyer có liên quan đến việc tăng xác suất hành khách đi du lịch

### *2.3.6. Phân tích độ lệch*

anova(model, test="Chisq")

```
> anova(model, test="Chisq")
Analysis of Deviance Table

Model: binomial, link: logit

Response: Target

Terms added sequentially (first to last)


             Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                           868     935.15
FrequentFlyer  1    160.8       867     774.36 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

*Hình 5: Kết quả phân tích ANOVA*

### *2.3.7. Dự đoán*

fitted.results <- predict(model,newdata=subset(test,select=c(2,3)),type='response')

fitted.results <- ifelse(fitted.results > 0.5,1,0)

```
> probabilities=predict(model,type="response")
> probabilities[1:5]
        1         3         4         5         6
0.1057851 0.1057851 0.1057851 0.1057851 0.5113636
```

*Hình 6: Dự đoán xác suất đơn biến hành khách đi du lịch*

### *2.3.8. Kiểm tra độ chính xác của mô hình*

misClasificError <- mean(fitted.results != test$Target)

print(paste('Accuracy',1-misClasificError))

```
> probabilities=predict(model,type="response")
> probabilities[1:5]
        1         3         4         5         6
0.1057851 0.1057851 0.1057851 0.1057851 0.5113636
> fitted.results <- predict(model,newdata=subset(test,select=c(2,3)),type='response')
> fitted.results <- ifelse(fitted.results > 0.5,1,0)
> misClasificError <- mean(fitted.results != test$Target)
> print(paste('Accuracy',1-misClasificError))
[1] "Accuracy 0.729411764705882"
```

*Hình 7: Độ chính xác của model đơn biến*

### *2.3.9. Ma trận hỗn loạn*

install.packages("e1071")

confusionMatrix(data=as.factor(fitted.results), reference=as.factor(test$Target))

```
          Reference
Prediction  0  1
         0 50 13
         1 10 12

               Accuracy : 0.7294
                 95% CI : (0.6221, 0.8201)
    No Information Rate : 0.7059
    P-Value [Acc > NIR] : 0.3661

                  Kappa : 0.3247

 Mcnemar's Test P-Value : 0.6767

            Sensitivity : 0.8333
            Specificity : 0.4800
         Pos Pred Value : 0.7937
         Neg Pred Value : 0.5455
             Prevalence : 0.7059
         Detection Rate : 0.5882
   Detection Prevalence : 0.7412
      Balanced Accuracy : 0.6567

       'Positive' Class : 0
```

*Hình 8: Ma trận hỗn loạn*

## 2.4. Mô hình hồi quy logistic đa biến bằng R:

### *2.4.1. Nhập thư viện*

```
library(tidyverse)

library(caret)

library(DataExplorer)

library(caTools)

library(GGally)

library(ggplot2)
```

### *2.4.2. Tải dữ liệu*

```
getwd()

df <- read.csv("./Data/Customertravel_Clean.csv")

str(df)

summary(df)
```

```
> df <- read.csv("./Data/Customertravel_Clean.csv")
> str(df)
'data.frame':   954 obs. of  7 variables:
 $ Age                       : int  34 34 37 30 30 27 34 34 30 36 ...
 $ FrequentFlyer             : int  0 1 0 0 0 1 0 0 0 1 ...
 $ AnnualIncomeClass         : int  2 1 2 2 1 3 2 1 1 3 ...
 $ ServicesOpted             : int  6 5 3 2 1 1 4 2 3 1 ...
 $ AccountSyncedToSocialMedia: int  0 1 1 0 0 0 1 1 0 0 ...
 $ BookedHotelOr0t           : int  1 0 0 0 0 1 1 0 1 0 ...
 $ Target                    : int  0 1 0 0 0 1 0 1 0 1 ...
```

### *2.4.3. Chia dữ liệu thánh train và test*

```
set.seed(150)

split <- sample.split(df$Target, SplitRatio <- 0.75)

training <- subset(df, split == TRUE)

test <- subset(df, split == FALSE)
```

Ta tách tập dữ liệu thành 2 tập gồm tập huấn luyện (training) 75%  và tập kiểm tra (test) 25%

### *2.4.4. Tạo model*

```
model <- glm( Target ~ FrequentFlyer + Age + AnnualIncomeClass +ServicesOpted +

AccountSyncedToSocialMedia + BookedHotelOr0t,

                    data = training, family = binomial)
```

summary(model)$coef

```
Debug> summary(model)$coef
                              Estimate Std. Error    z value     Pr(>|z|)
(Intercept)                2.72795536 1.11048280  2.4565489 1.402787e-02
FrequentFlyer              2.52353175 0.25059982 10.0699663 7.500379e-24
Age                       -0.18287261 0.03456560 -5.2905953 1.219189e-07
AnnualIncomeClass          0.01748809 0.14207502  0.1230905 9.020354e-01
ServicesOpted              0.32112946 0.06912849  4.6453998 3.394184e-06
AccountSyncedToSocialMedia 0.93038536 0.23317918  3.9900019 6.607277e-05
BookedHotelOr0t           -1.13830111 0.25374739 -4.4859618 7.258573e-06
```

*Hình 9: Hệ số của mô hình hồi quy Logistic đa biến*

### *2.4.5. Dự đoán xác suất*

probabilities <- model %>% predict(test, type = "response")

head(probabilities)

# dự đoán các lớp

predicted.classes <- ifelse(probabilities > 0.5, "1", "0")

head(predicted.classes)

```
Debug> probabilities <- model %>% predict(test, type = "response")
Debug> head(probabilities)
         2          5         10         15         17         18
0.83024143 0.08168986 0.27721209 0.13597981 0.06738808 0.74446149
Debug> predicted.classes <- ifelse(probabilities > 0.5, "1", "0")
Debug> head(predicted.classes)
  2   5  10  15  17  18
"1" "0" "0" "0" "0" "1"
```

*Hình 10: Dự đoán xác suất đa biến hành khách đi du lịchlịch*

Phân loại các cá nhân thành hai nhóm dựa trên xác suất dự đoán (p) là thường xuyên đi du lịch. Các cá nhân, với p trên 0,5 (phỏng đoán ngẫu nhiên), được coi là thường xuyên đi du lịch.

### *2.4.6. Độ chính xác của mô hình*

mean(predicted.classes == test$Target)

```
Debug> mean(predicted.classes == test$Target)
[1] 0.7563025
```

*Hình 11: Độ chính xác của model đa biến*

Độ chính xác của dự đoán phân loại là khoảng 75,6%, là tốt. Tỷ lệ lỗi phân loại sai là 24%.

# CHƯƠNG 3: PHÂN TÍCH DỮ LIỆU CHUỖI THỜI GIAN VỚI MÔ HÌNH AR, ARMA, ARIMA

## 3.1. Phân tích dữ liệu chuỗi thời gian với mô hình AR

### *3.1.1.Bài toán đặt ra*

Bài toán nhằm mục đích dự đoán  tỉ lệ thất nghiệp trong những năm sắp tới

Nguồn dữ liệu : Kaggle

Gồm:

- M: Thời gian
- U: Tỉ lệ thất nghiệp

Dataset: Unemployment

Tác giả: GAURAV DUTTA

Nguồn dữ liệu:

https://www.kaggle.com/code/gauravduttakiit/forecasting-unemployment-with-ar-method/data

Cách giải bài toán: Sử dụng mô hình AR để dự đoán tỉ lệ thất nghiệp

### *3.1.2. Thực nghiệm*

- Import các thư viện cần thiết:

  *library*(tidyverse)

  *library*(forecast)

  *library*(lubridate)

  *library*(MLmetrics)

  *library*(zoo)

  *library*(plotly)

  *library*(xts)

  *library*(TSstudio)

  *library*(tseries)

  *library*(lubridate)

  *library*(ggplot2)

  *library*(urca)

- Đọc dữ liệu

  df=*read.csv*('./Data/UNRATE.csv', header = TRUE)

*table*(*is.na*(df))

*summary*(df)

*table*(*is.na*(df))

```
> summary(df)
     M                   U          
Length:866         Min.   : 2.50  
Class :character   1st Qu.: 4.50  
Mode  :character   Median : 5.50  
                   Mean   : 5.73  
                   3rd Qu.: 6.80  
                   Max.   :10.80  
```

*Hình 12: Thông tin mô tả dữ liệu*

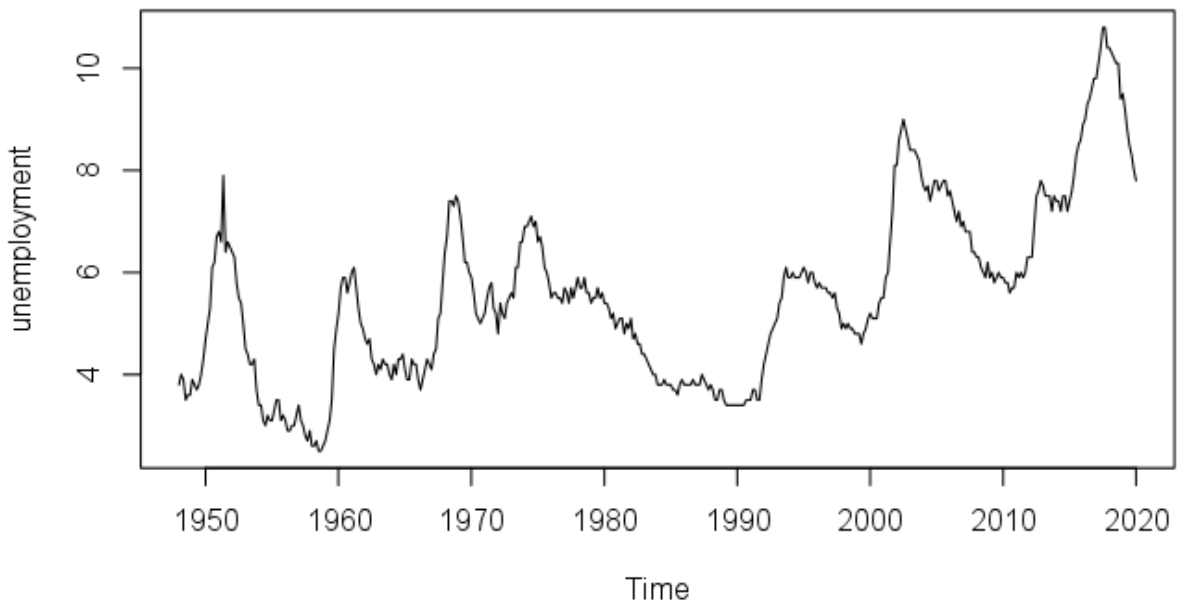
```
FALSE 
 1732 
```

*Hình 13: Kiểm tra giá trị null của  dữ liệu*

- Trực quan hoá dữ liệu:

  unemployment <- *ts*(wine_venda$Unemployment, start=*c*(1948, 1), end=*c*(2020, 1), frequency=6)

  *plot*(unemployment)

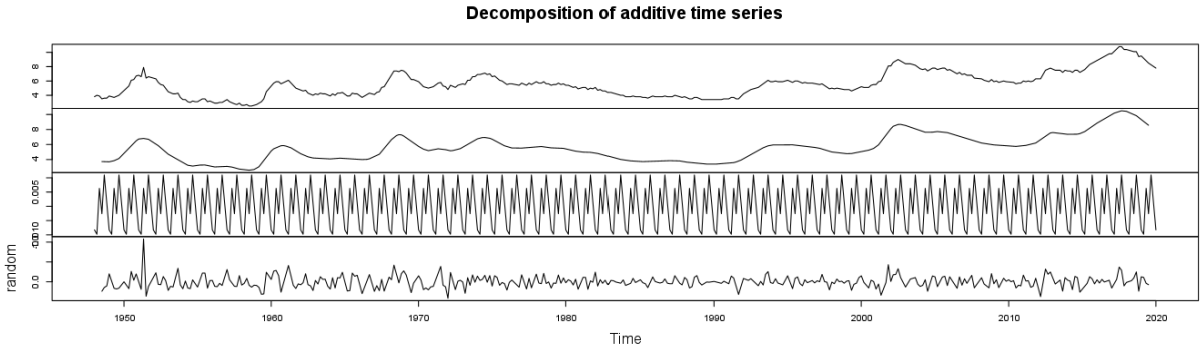



*Hình 14: Biểu đồ tổng quan thể hiện dữ liệu*

Từ biểu đồ trên, ta nhận thấy biểu đồ có khuynh hướng tăng

- Phân rã dữ liệu thời gian:

  *plot*(*decompose*(unemployment))




*Hình 15: Biểu đồ phân rã dữ liệu*

Từ biểu đồ có thể thấy dữ liệu có xu hướng tăng và có tính chu kỳ và mùa vụ

- Kiểm tra chuỗi dừng bằng ADF

y_none=*ur.df*(unemployment,type = "none", selectlags = "AIC" )

*summary*(y_none)

y_drift=*ur.df*(unemployment,type="drift", selectlags = "AIC")

*summary*(y_drift)

y_trend=*ur.df*(unemployment,type="trend", selectlags = "AIC")

*summary*(y_trend)

*adf.test*(unemployment)

```
data:  unemployment
Dickey-Fuller = -3.8398, Lag order = 7, p-value = 0.01714
alternative hypothesis: stationary
```

*Hình 16: Kết quả kiểm định ADF*

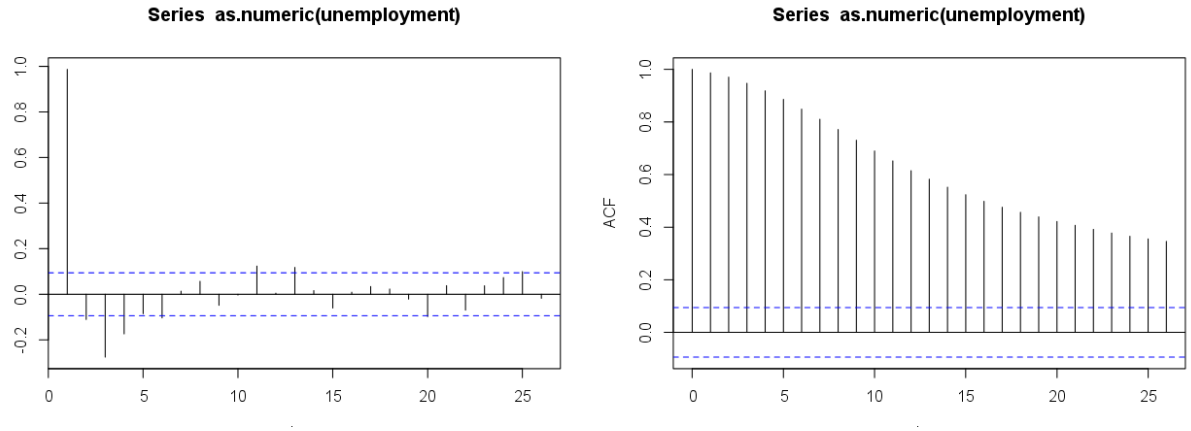
Vì p-value<0.05, từ quan sát biểu đồ trên => dữ liệu có tính dừng

- Sử dụng giản đồ PACF và ACF tìm p,q

  *par*(mfrow = *c*(1,2))

  *pacf*(*as.numeric*(unemployment))

*acf*(*as.numeric*(unemployment))




*Hình 17: Biểu đồ PACF và ACF của dữ liệu*

Tuy nhiên, nếu không thực hiện bất kì tích hợp nào (tức d= 0) thì độ trễ trong đồ thị tự tương quan ở trên là cao. Vì vậy, dữ liệu chuỗi thời gian này có tính dừng ít, vì vậy cần sử dụng sai phân để chuyển đổi chuỗi dừng.

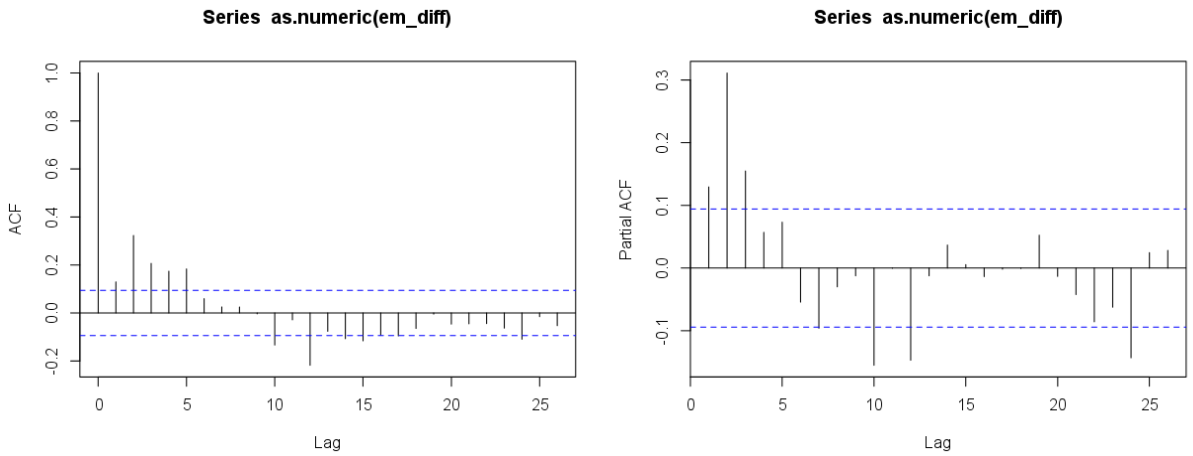
em_diff <- *diff*(unemployment, differences = 1)

#Kiểm tra sau khi tính sai phân và chọn tham số d

*pacf*(*as.numeric*(em_diff))

*acf*(*as.numeric*(em_diff))

*adf.test*(em_diff)




*Hình 18: Biểu đồ ACF và PACF của dữ liệu sau khi tính sai phân bậc 1*

Sau khi lấy sai phân bậc 1 , từ biểu đồ trên chọn p=2

- Chia dữ liệu train/test:

train_dat <- *window*(unemployment, start=*c*(1948,1), end=*c*(2020,1))

test_dat <- *window*(unemployment, start=*c*(1948,7), end=*c*(2020,1))

- Tạo model:

  model <- *arima*(train_dat , order = *c*(2, 0, 0))

  *summary*(model)

- Dự đoán tỉ lệ thất nghiệp trong 24 tháng tới

  fcst <- *forecast*(model, h = 24)

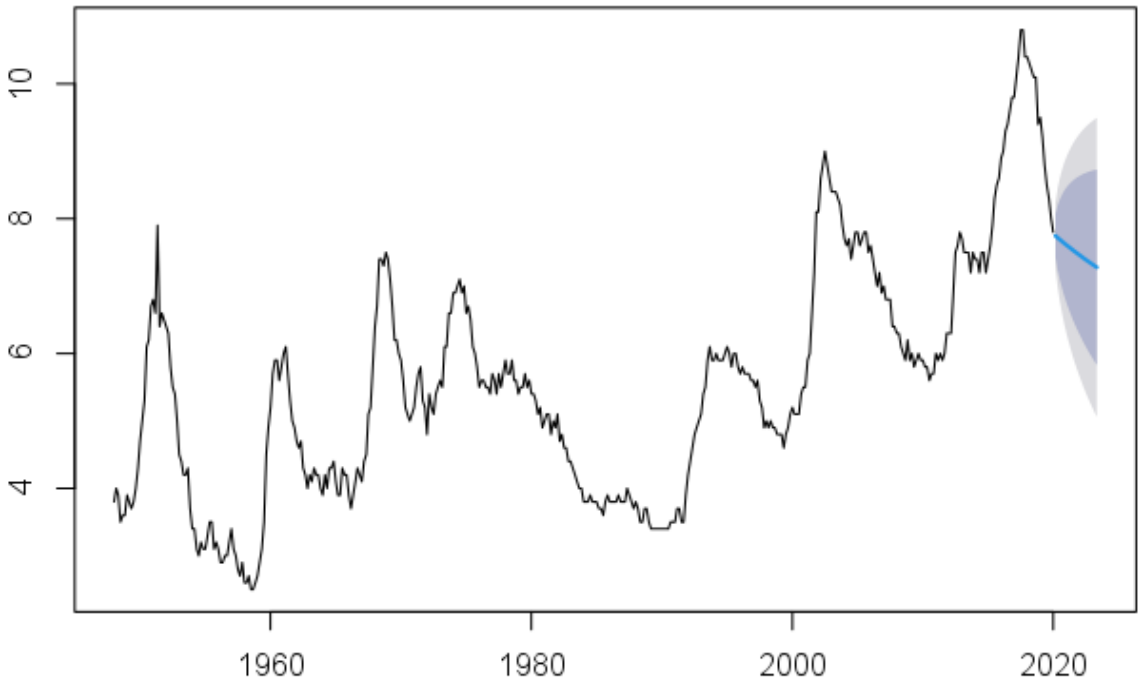
  fcst

  #Call the point predictions

  fcst$mean

  #Plot the forecast

  *plot*(fcst)




*Hình 19: Kết quả dự báo tỉ lệ thất nghiệp trong 24 tháng tới*

Từ kết quả dự báo có thể thấy tỉ lệ thất nghiệp trong 24 tháng tới có chiều hướng giảm

## 3.2. Phân tích dữ liệu chuỗi thời gian với mô hình ARMA

### *3.2.1.Bài toán đặt ra*

Bài toán nhằm mục đích dự đoán  số hành khách máy bay trong những năm sắp tới

Nguồn dữ liệu : Kaggle

Gồm:

- M: Thời gian
- P: Số lượng hành khách

Dataset: Airline passenger traffic

Tác giả: GAURAV DUTTA

Nguồn dữ liệu: https://www.kaggle.com/datasets/gauravduttakiit/airline-passenger-traffic

Cách giải bài toán: Sử dụng mô hình ARMA để dự đoán số lượng hành khách

### *3.2.2. Thực nghiệm*

- Import các thư viện cần thiết:

```
library(tidyverse)
library(forecast)
library(lubridate)
library(MLmetrics)
library(zoo)
library(plotly)
library(xts)
library(TSstudio)
library(tseries)
library(lubridate)
library(ggplot2)
library(urca)
```

- Đọc dữ liệu:

```
df=read.csv('./Data/airline-passenger-traffic(1).csv', header = TRUE)
table(is.na(df))
summary(df)
```

```
FALSE  TRUE
  282     4
> summary(df)
      M                    P
 Length:143         Min.   :104.0
 Class :character   1st Qu.:180.5
 Mode  :character   Median :269.0
                    Mean   :281.9
                    3rd Qu.:361.0
                    Max.   :622.0
                    NA's   :4
```

*Hình 20: Thông tin mô tả của dữ liệu*

Có thể thấy dữ liệu có 4 giá trị null, mình tiến hành loại bỏ 4 dòng này:

traffic2 <- traffic %>%

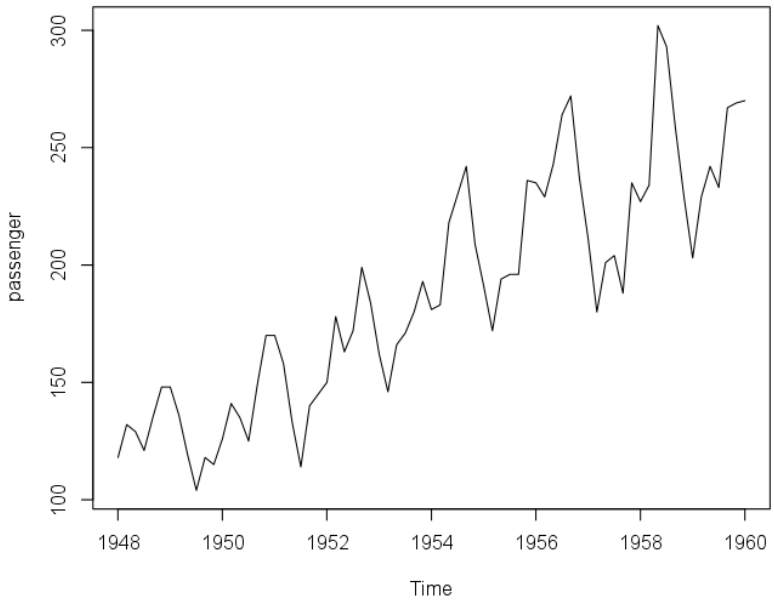
*drop_na*(Month,Passenger)

*summary*(traffic2)

- Trực quan dữ liệu:

  passenger <- *ts*(traffic2$Passenger, start=*c*(1948, 1), end=*c*(1960, 1), frequency=6)

  *plot*(passenger)

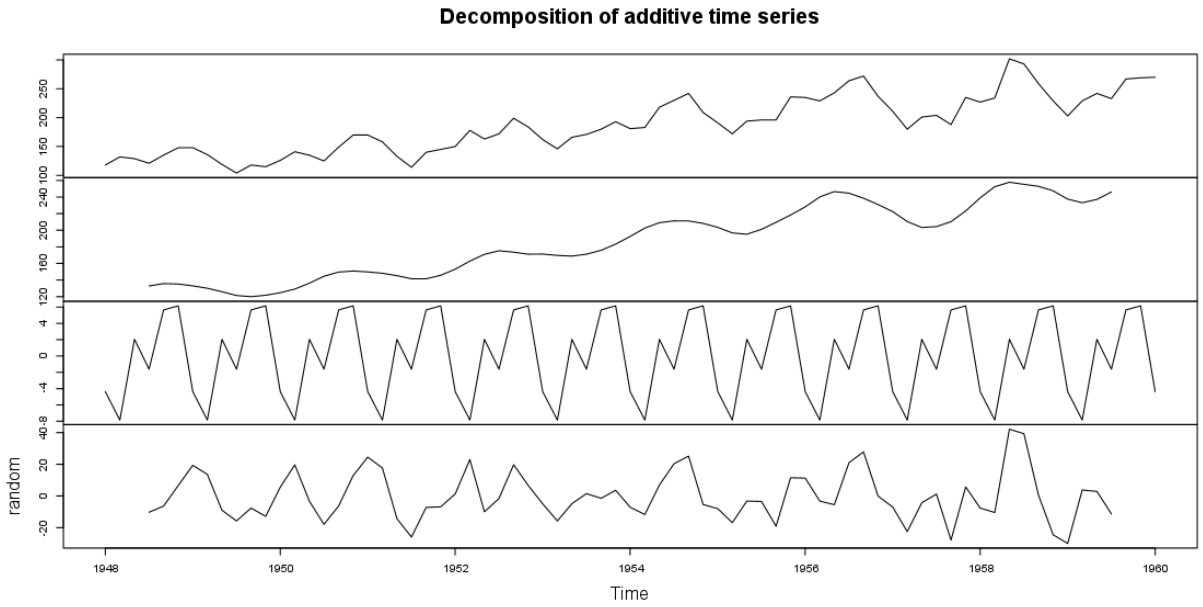



*Hình 21: Biểu đồ mô tả tổng quan dữ liệu*

Từ biểu đồ có thể thấy dữ liệu có chiều hướng đi lên

- Phân rã dữ liệu:

*plot*(*decompose*(passenger))



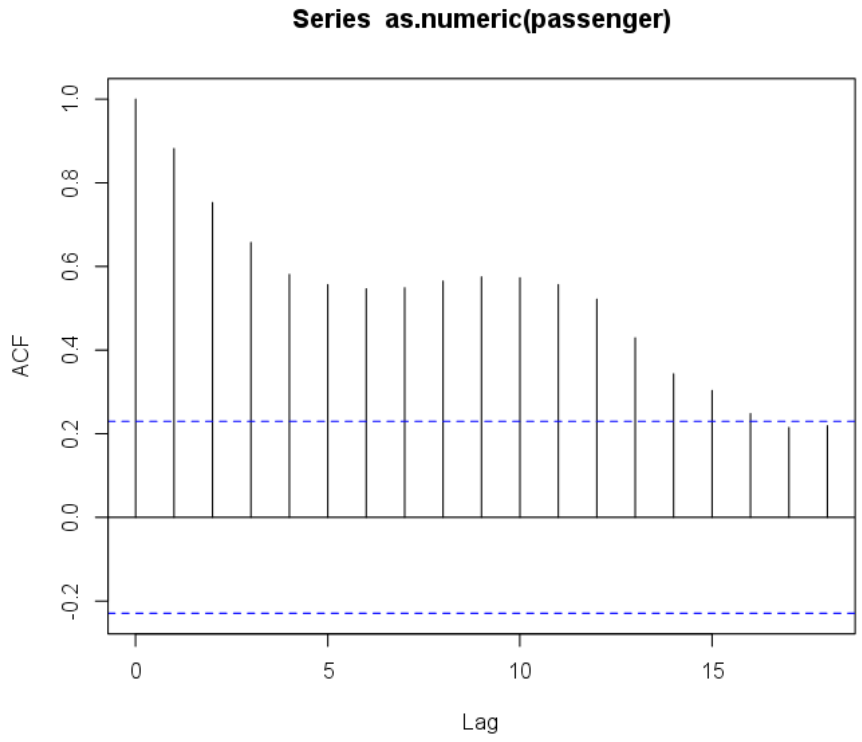

*Hình 22: Biểu đồ phân rã dữ liệu*

Từ các biểu đồ có thể thấy dữ liệu có xu hướng tăng, và có tính chu kì, mùa vụ

- Tính acf:

  *par*(mfrow = *c*(1,2))

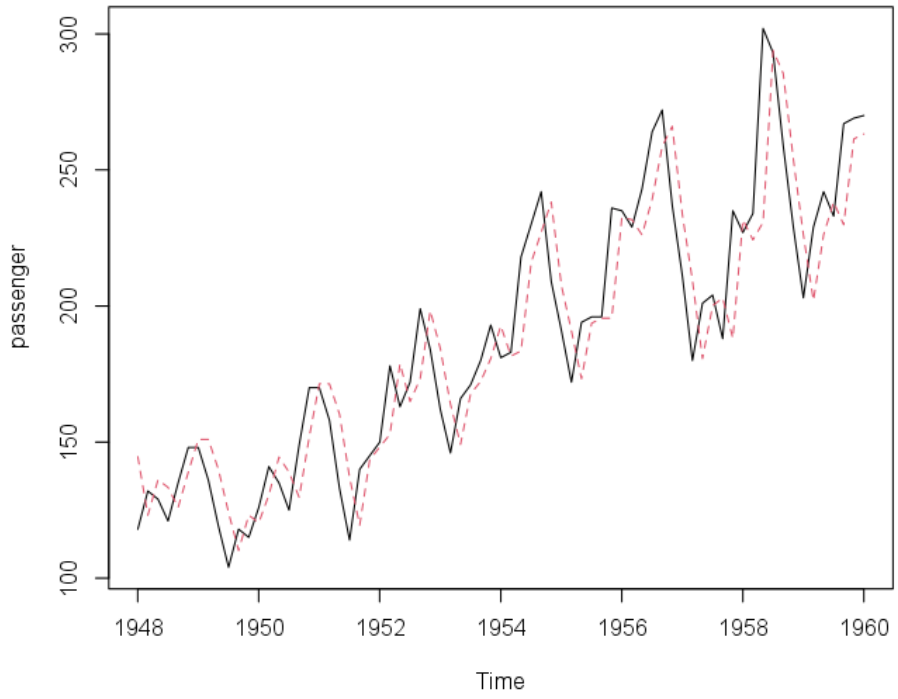
  *acf*(*as.numeric*(passenger))




Hình: Biểu đồ ACF của dữ liệu

- Xây dựng mô hình AR

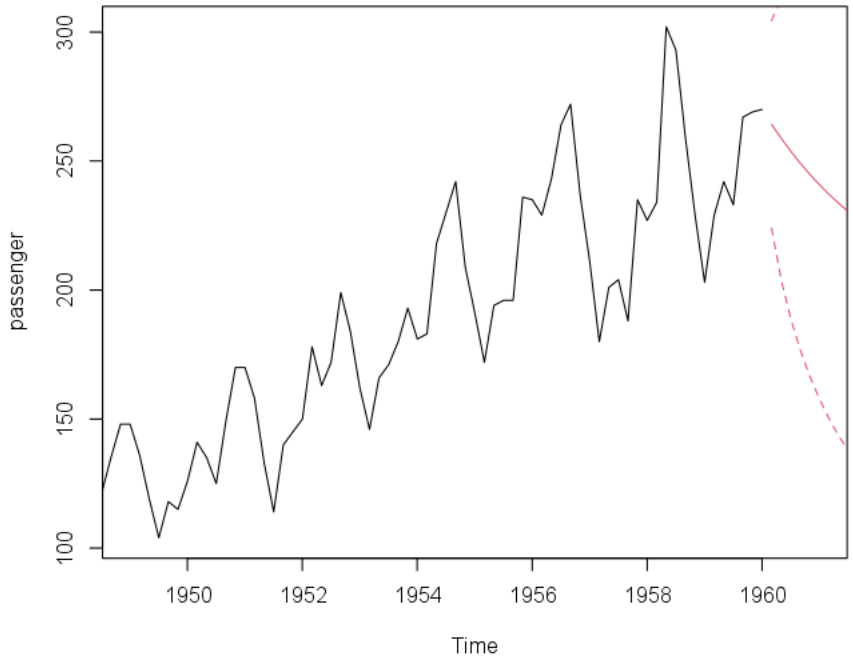
```
AR <- arima(passenger, order = c(1,0,0))
print(AR)
ts.plot(passenger)
AR_fit <- passenger - residuals(AR)
points(AR_fit, type = "l", col = 2, lty = 2)
```




Hình: Kết quả từ mô hình AR

- Dự đoán bằng mô hình AR:

```
predict_AR <- predict(AR)
predict_AR$pred[1]
predict(AR, n.ahead = 10)
#Vẽ biểu đồ chuỗi AirPassenger cộng với khoảng thời gian dự đoán và dự đoán 95%.
ts.plot(passenger, xlim = c(1949, 1961))
AR_forecast <- predict(AR, n.ahead = 10)$pred
AR_forecast_se <- predict(AR, n.ahead = 10)$se
points(AR_forecast, type = "l", col = 2)
points(AR_forecast - 2*AR_forecast_se, type = "l", col = 2, lty = 2)
points(AR_forecast + 2*AR_forecast_se, type = "l", col = 2, lty = 2)
```




Hình: Kết quả dự báo số lượng khác hàng từ mô hình AR

- Xây dựng mô hình MA để dự đoán:

  MA <- *arima*(passenger, order = *c*(0,0,1))

  *print*(MA)

  *ts.plot*(passenger)

  MA_fit <- passenger - *resid*(MA)

  *points*(MA_fit, type = "l", col = 2, lty = 2)

- Dự đoán bằng mô hình MA:

  predict_MA <- *predict*(MA)

  predict_MA$pred[1]

  *predict*(MA,n.ahead=10)

  *ts.plot*(passenger, xlim = *c*(1949, 1961))

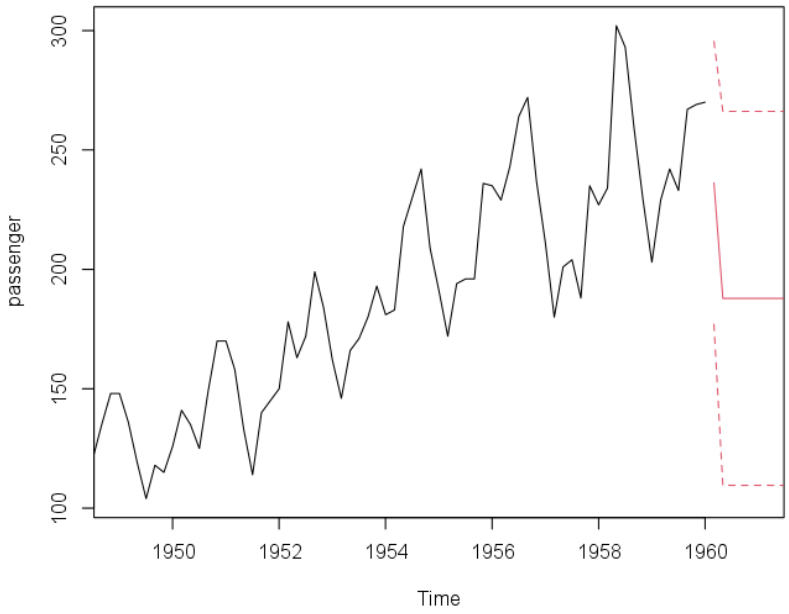
  MA_forecasts <- *predict*(MA, n.ahead = 10)$pred

  MA_forecast_se <- *predict*(MA, n.ahead = 10)$se

  *points*(MA_forecasts, type = "l", col = 2)

  *points*(MA_forecasts - 2*MA_forecast_se, type = "l", col = 2, lty = 2)

  *points*(MA_forecasts + 2*MA_forecast_se, type = "l", col = 2, lty = 2)




*Hình 23: Kết quả dự báo từ mô hình MA*

- Tìm mối tương quan giữa AR và MA:

*cor*(AR_fit, MA_fit)

# Find AIC of MA

*AIC*(MA)

# Find AIC of MA

*AIC*(MA)

# Find BIC of AR

*BIC*(AR)

# Find BIC of MA

*BIC*(MA)

```
> cor(AR_fit, MA_fit)
[1] 0.9165788
> AIC(AR)
[1] 653.3021
> AIC(MA)
[1] 709.1985
> BIC(AR)
[1] 660.1735
> BIC(MA)
[1] 716.0699
```

*Hình 24: Kết quả thể hiện giá trị AIC và BIC cho mô hình AR và MA*

Với mô hình AR cho giá trị AIC và BIC thấp hơn nên ta chọn mô hình AR và lấy kết quả dự đoán từ mô hình AR

## 3.3. Phân tích dữ liệu chuỗi thời gian với mô hình ARIMA

### *3.3.1. Bài toán đặt ra*

Bài toán nhằm mục đích dự đoán số lượng hành khách tham gia giao thông ở Portland (Oregon) trung bình các tháng tiếp theo dựa vào dữ liệu chuỗi thời gian các tháng trước đó.

Nguồn dữ liệu: Gồm 114 dòng và 2 cột:

- Month: Thời gian
- Traffic: số lượng hành khách trung bình ở Portland (Oregon) mỗi tháng.

Dataset: Time Series Analysis and forecasting using ARIMA

Tác giả: Hsankesara

Link: https://www.kaggle.com/code/hsankesara/time-series-analysis-and-forecasting-using-arima/data

Nguồn dữ liệu: https://www.kaggle.com

Cách giải bài toán: Sử dụng mô hình ARIMA để dự đoán số lượng hành khách ở Portland (Oregon) trung bình các tháng tiếp theo.

Ngôn ngữ : R

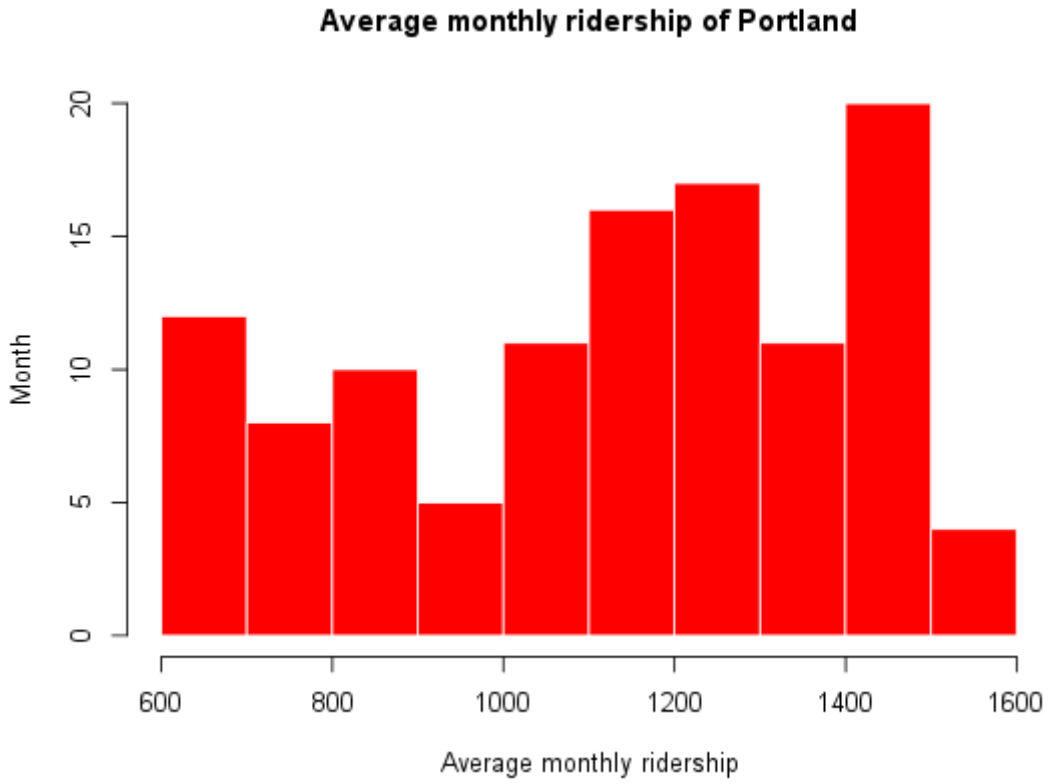
### *3.3.2. Phân tích thực nghiệm*

```
#%% - Import Library
library(urca)
library(ggplot2)
library(readr)
library(fpp2)
library(tidyverse)
library(forecast)
library(lubridate)
library(MLmetrics)
library(zoo)
library(plotly)
library(xts)
```

```
library(TSstudio)
library(tseries)

#%% - Read Data
df = read.csv('./data/portland-oregon-average-monthly-1.csv', header = TRUE)

#%% - Change format to date and rename
Rname <- df %>% select(c(Month,traffic))
colnames(Rname) <- c('Month','Average monthly ridership')
head(Rname)

#%% - Convert to times series data
Average_monthly_ridership <- ts(Rname$`Average monthly ridership`, start = c(1960,1), end = c(1969,6), frequency = 12)
```

Vẽ biểu đồ trực quan về số lượng hành khách ở Portland trung bình hàng tháng:

```
#%% - Visualization Hist Plot
hist(Average_monthly_ridership, col="red", border =
"white",xlab="Average monthly ridership",ylab="Month", main = "Average monthly ridership of Portland")
```

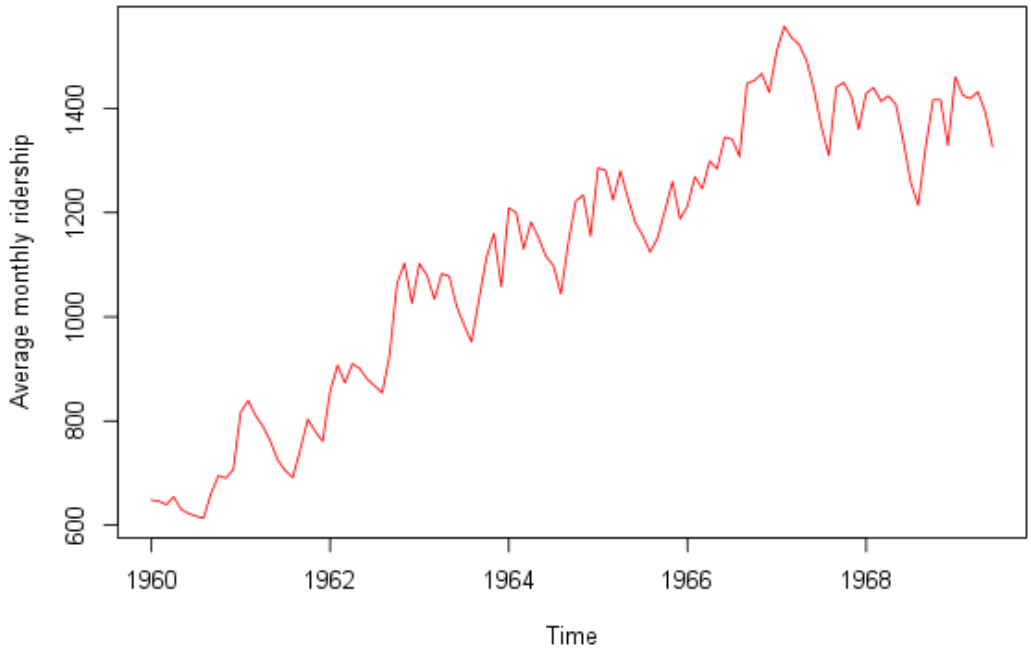



*Hình 25: Biểu đồ trực quan về số lượng hành khách ở Portland trung bình hàng tháng*

Sau đó thực hiện vẽ biểu đồ thể hiện toàn bộ dữ liệu chuỗi thời gian.

*#%% - visualization data*

*plot*(Average_monthly_ridership)



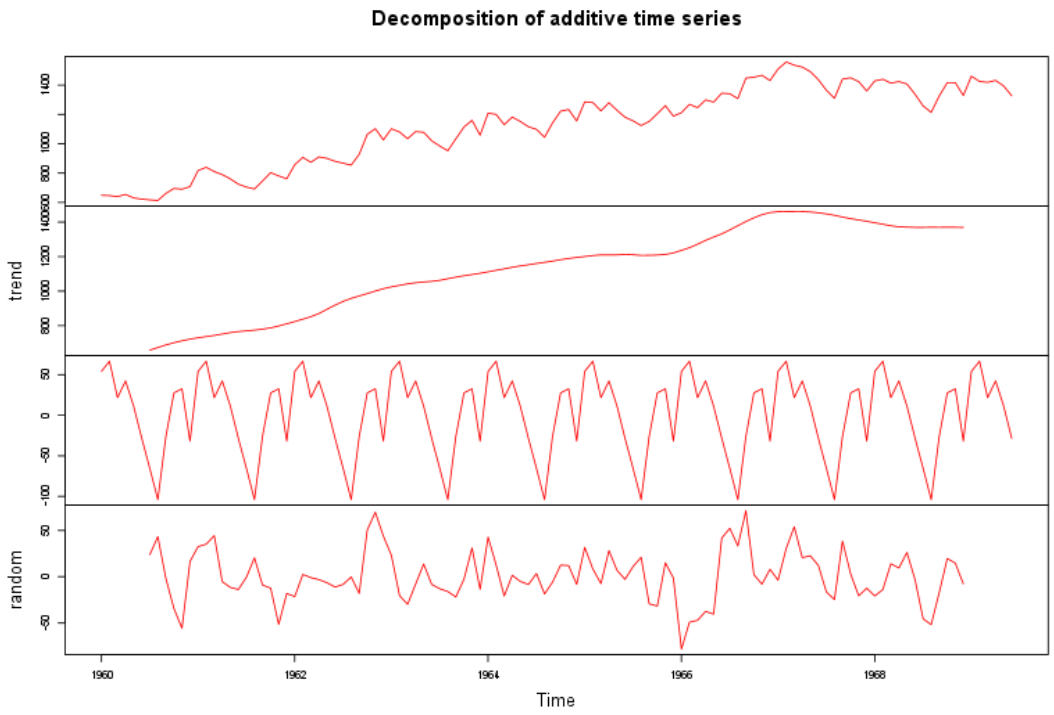

*Hình 26: Biểu đồ trực quan toàn bộ dữ liệu chuỗi thời gian*

Tiếp theo, thực hiện phân rã dữ liệu chuỗi thời gian thành 4 thành phần (Random, Seasonal, Trend và Observed)

*#%% - Decomposition of time series data*

decomp <- *decompose*(Average_monthly_ridership)

*plot*(decomp)




*Hình 27: Phân rã dữ liệu chuỗi thời gian*

Nhìn vào biểu đồ phân rã dữ liệu chuỗi thời gian trên, ta có thể thấy dữ liệu có tính thời vụ (Seasonal) và có xu hướng (Trend) tăng dần theo thời gian.

Sau khi phân rã xong, tiếp tục sử dụng thử nghiệm ADF để kiểm tra tính dừng của chuỗi thời gian. Kiểm tra bằng cách đối chiếu với tham số p-value = 0.05, nếu giá trị p-value của phép thử nhỏ hơn 0.05 thì ta bác bỏ giả thuyết Null Hypothesis, tức là chuỗi dừng. Ngược lại, nếu lớn hơn 0.05 thì chuỗi không dừng.

*#%% - Check stationary by using ADF test*

*adf.test*(Average_monthly_ridership)

```
    Augmented Dickey-Fuller Test

data:  Average_monthly_ridership
Dickey-Fuller = -2.2173, Lag order = 4, p-value = 0.4864
alternative hypothesis: stationary
```

*Hình 28: Kết quả thử nghiệm ADF*

p-value = 0.4864 > 0.05 cho nên chuỗi chưa có tính dừng

Vì chưa dừng, chúng ta sử dụng tính sai phân bậc 1 để tiếp tục thử nghiệm.

*#%%*

Average_monthly_ridership.DIFF <- *diff*(Average_monthly_ridership)

*adf.test*(Average_monthly_ridership.DIFF)

```
    Augmented Dickey-Fuller Test

data:  Average_monthly_ridership.DIFF
Dickey-Fuller = -5.4987, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

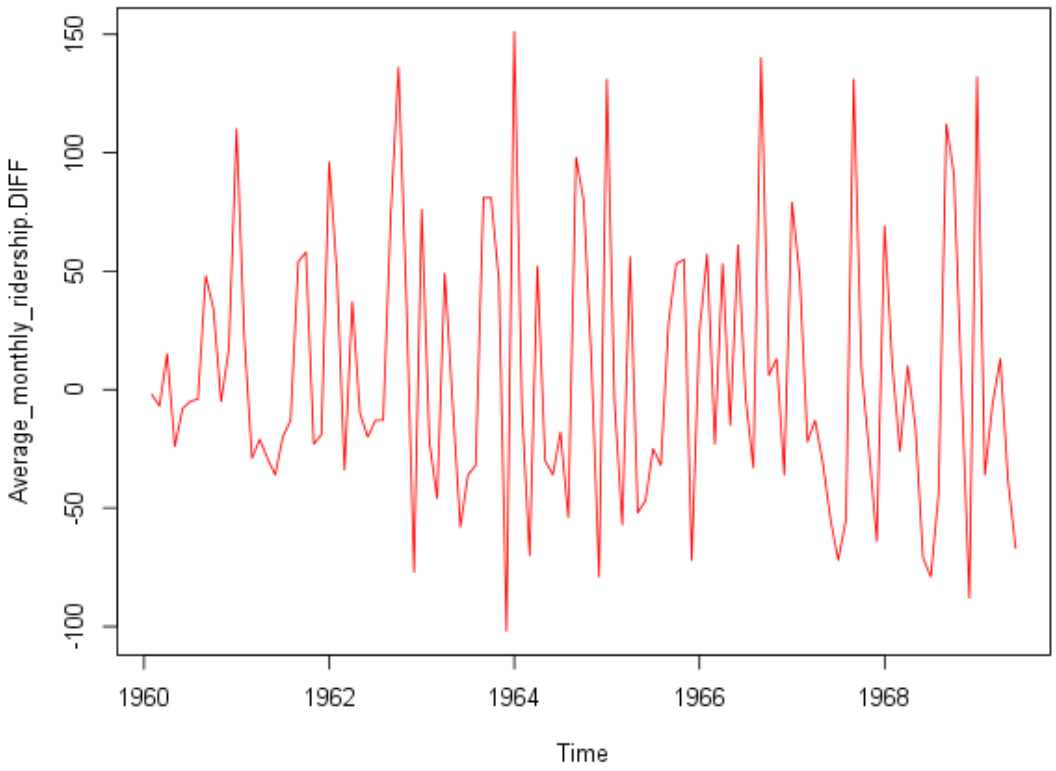
*Hình 29: Kết quả thử nghiệm ADF với sai phân bậc 1*

p-value = 0.01 < 0.05 nên suy ra chuỗi có tính dừng

Sau khi tính sai phân bậc 1 của dữ liệu ta có các biểu đồ tổng quan về dữ liệu và tự tương quan như sau:

*#%%*

*plot.ts*(Average_monthly_ridership.DIFF, col="red")

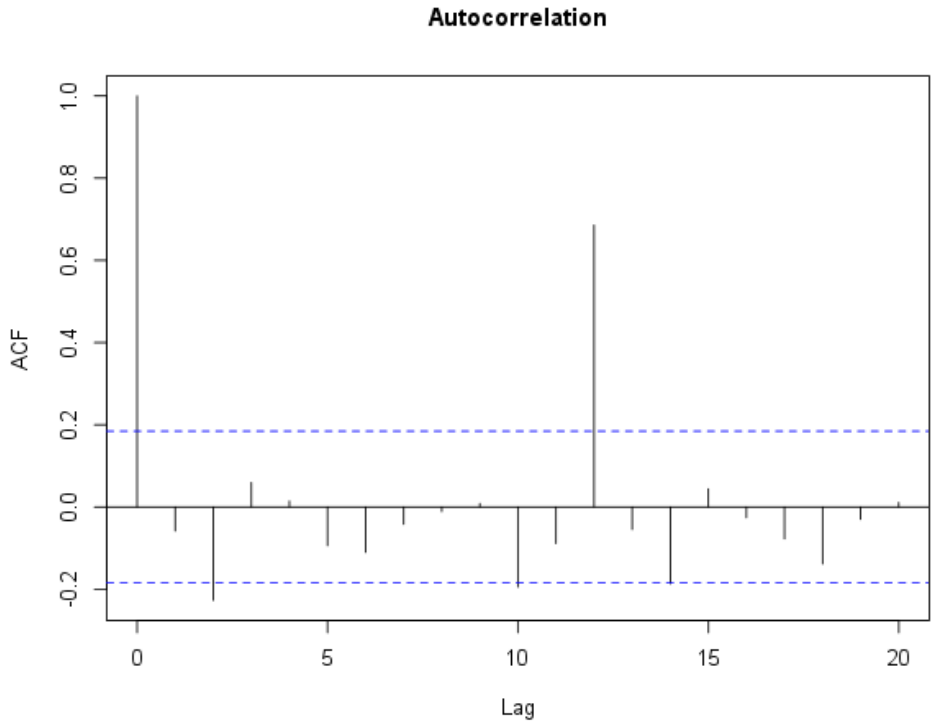



*Hình 30: Biểu đồ tổng quan về dữ liệu (sai phân bậc 1)*

Qua biểu đồ tổng quan về dữ liệu sau khi lấy sai phân bậc 1, ta có thể thấy đồ thị dường như có tính ổn định và có các khoảng biến thiên lên xuống đều đặn hơn so với trước khi lấy sai phân bậc 1.

Thực hiện vẽ đồ thị tương quan ACF và PACF với sai phân bậc 1.

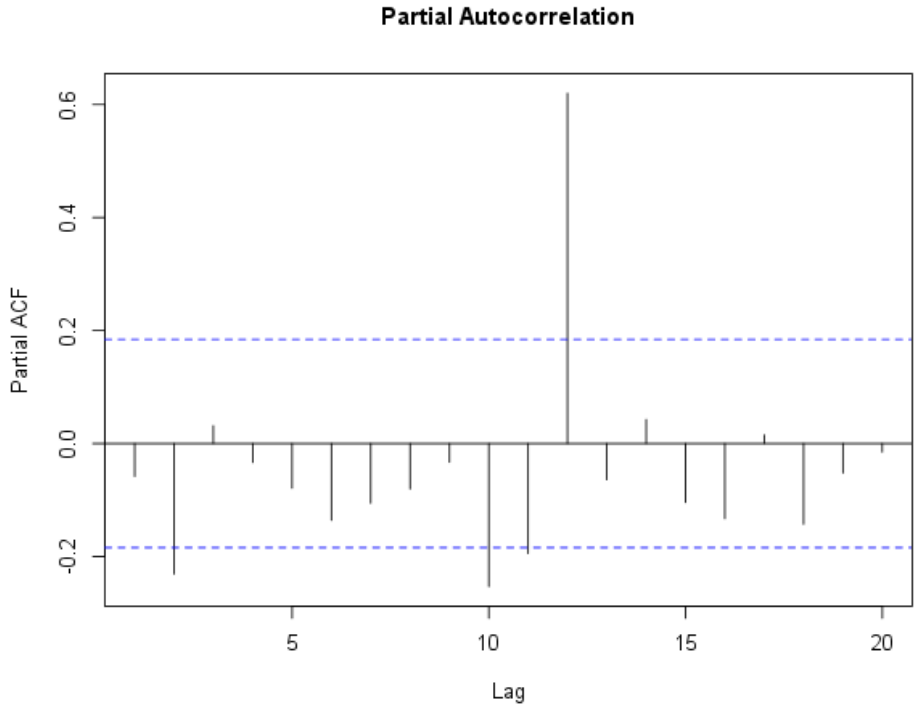
```
#%% - ACF Plot with d =1
acf(as.numeric(Average_monthly_ridership.DIFF),
main="Autocorrelation)
```




*Hình 31: Đồ thị tự tương quan ACF với sai phân bậc 1*

*#%% - PACF Plot with d =1*

*pacf*(*as.numeric*(Average_monthly_ridership.DIFF),
main="Partial Autocorrelation")




*Hình 32: Đồ thị tự tương quan từng phần PACF với sai phân bậc 1*

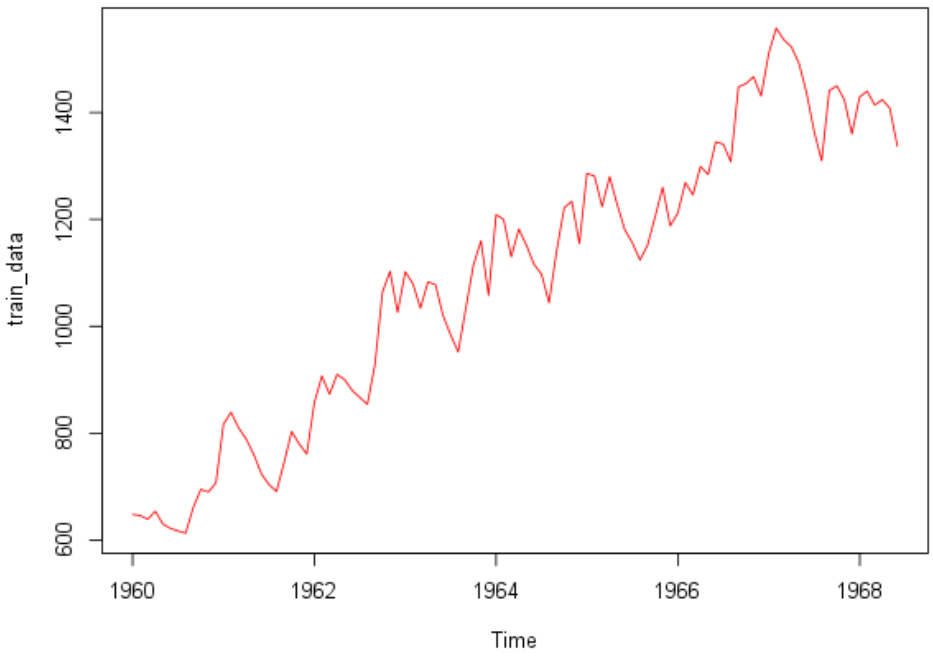
Tiếp theo, để hiểu rõ hơn độ chính xác của các dự báo của mình, chúng ta thực hiện chia dữ liệu thành 2 tập Train data và Test data. Đặt khoảng thời gian để test là 12 tháng cuối của chuỗi dữ liệu (Từ tháng 7 năm 1968 đến tháng 6 năm 1969). Sau đó thực hiện tiến hành vẽ đồ thị trực quan dữ liệu Train data.

*#%% - Train data and Test data*

train_data <- *window*(Average_monthly_ridership, start = *c*(1960,1), end = *c*(1968,6))

test_data <- *window*(Average_monthly_ridership, start = *c*(1968,7), end = *c*(1969,6))

*plot*(train_data, col = 'red')




*Hình 33: Đồ thị trực quan dữ liệu Train data*

Sử dụng auto.arima() để chọn ra model phù hợp nhất.

*#%% - Build ARIMA model*

model <- *auto.arima*(train_data, approximation = FALSE,

trace = TRUE)

*summary*(model)

```
ARIMA(2,0,2)(1,1,1)[12] with drift         : Inf
ARIMA(0,0,0)(0,1,0)[12] with drift         : 1086.526
ARIMA(1,0,0)(1,1,0)[12] with drift         : 913.6368
ARIMA(0,0,1)(0,1,1)[12] with drift         : Inf
ARIMA(0,0,0)(0,1,0)[12]                    : 1147.596
ARIMA(1,0,0)(0,1,0)[12] with drift         : 920.2057
ARIMA(1,0,0)(2,1,0)[12] with drift         : 911.4088
ARIMA(1,0,0)(2,1,1)[12] with drift         : Inf
ARIMA(1,0,0)(1,1,1)[12] with drift         : Inf
ARIMA(0,0,0)(2,1,0)[12] with drift         : 1047.454
ARIMA(2,0,0)(2,1,0)[12] with drift         : Inf
ARIMA(1,0,1)(2,1,0)[12] with drift         : 913.2701
ARIMA(0,0,1)(2,1,0)[12] with drift         : 984.42
ARIMA(2,0,1)(2,1,0)[12] with drift         : Inf
ARIMA(1,0,0)(2,1,0)[12]                    : 911.7107

Best model: ARIMA(1,0,0)(2,1,0)[12] with drift
```

*Hình 34: Kết quả sau khi chạy auto.arima()*

Từ đó tìm ra model phù hợp nhất là ARIMA(1,0,0)(2,1,0)

```
> summary(model)
Series: train_data
ARIMA(1,0,0)(2,1,0)[12] with drift

Coefficients:
         ar1     sar1     sar2   drift
      0.9341  -0.4052  -0.2716  7.8904
s.e.  0.0418   0.1087   0.1233  2.6185

sigma^2 = 1290:  log likelihood = -450.35
AIC=910.69   AICc=911.41   BIC=923.19

Training set error measures:
                    ME     RMSE      MAE         MPE    MAPE      MASE
Training set -0.7987827 32.97908 23.61119 -0.05231924 2.05706 0.1996437
                   ACF1
> |
```

*Hình 35: Kết quả của mô hình ARIMA(1, 0, 0)*

Sau khi áp dụng mô hình ARIMA phù hợp vào dữ liệu đào tạo và tiến hành dự đoán thử nghiệm, chúng ta sử dụng hàm forecast() để dự báo lượng khách trung bình ở PortLand trong 12 tháng tiếp theo, từ tháng 6 năm 1968 tới tháng 5 năm 1969.

*#%% - Forecast*

forecast_model <- *forecast*(model, h=12)

| | Point Forecast | Lo 80 | Hi 80 | Lo 95 | Hi 95 |
|---|---|---|---|---|---|
| Jul 1968 | 1298.512 | 1252.484 | 1344.541 | 1228.118 | 1368.907 |
| Aug 1968 | 1263.594 | 1200.609 | 1326.579 | 1167.267 | 1359.921 |
| Sep 1968 | 1377.994 | 1303.296 | 1452.693 | 1263.753 | 1492.236 |
| Oct 1968 | 1408.053 | 1324.465 | 1491.641 | 1280.216 | 1535.890 |
| Nov 1968 | 1418.143 | 1327.508 | 1508.779 | 1279.529 | 1556.758 |
| Dec 1968 | 1364.004 | 1267.641 | 1460.367 | 1216.629 | 1511.379 |
| Jan 1969 | 1429.863 | 1328.767 | 1530.959 | 1275.249 | 1584.476 |
| Feb 1969 | 1465.536 | 1360.484 | 1570.588 | 1304.873 | 1626.199 |
| Mar 1969 | 1447.645 | 1339.260 | 1556.030 | 1281.884 | 1613.406 |
| Apr 1969 | 1472.569 | 1361.357 | 1583.781 | 1302.485 | 1642.653 |
| May 1969 | 1460.736 | 1347.115 | 1574.356 | 1286.968 | 1634.503 |
| Jun 1969 | 1433.238 | 1317.557 | 1548.919 | 1256.319 | 1610.157 |

*Hình 36: Kết quả dự báo lượng hành khách ở Portland trong 12 tháng tiếp theo*

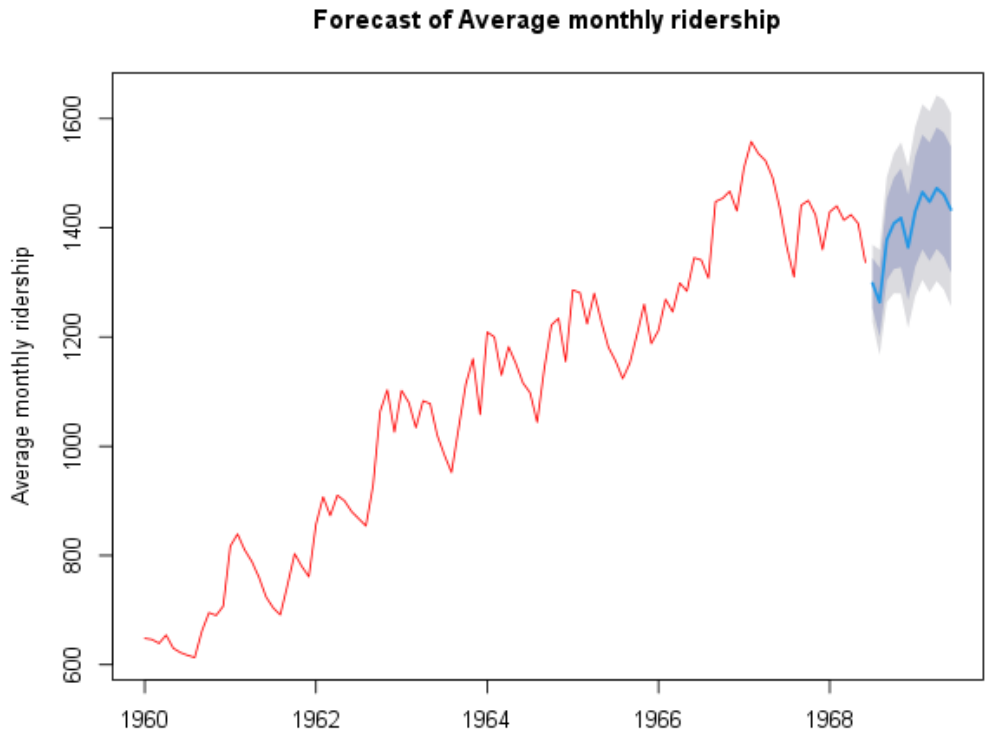
*#%% - Prediction Plot*

*plot*(forecast_model,col="red",ylab="Average monthly ridership,

main = "Forecast of Average monthly ridership")

forecast_model

Để có cái nhìn rõ nét hơn kết quả dự báo, ta vẽ đồ thị lượng khách trung bình ở PortLand trong 12 tháng tiếp theo, từ tháng 6 năm 1968 tới tháng 5 năm 1969.

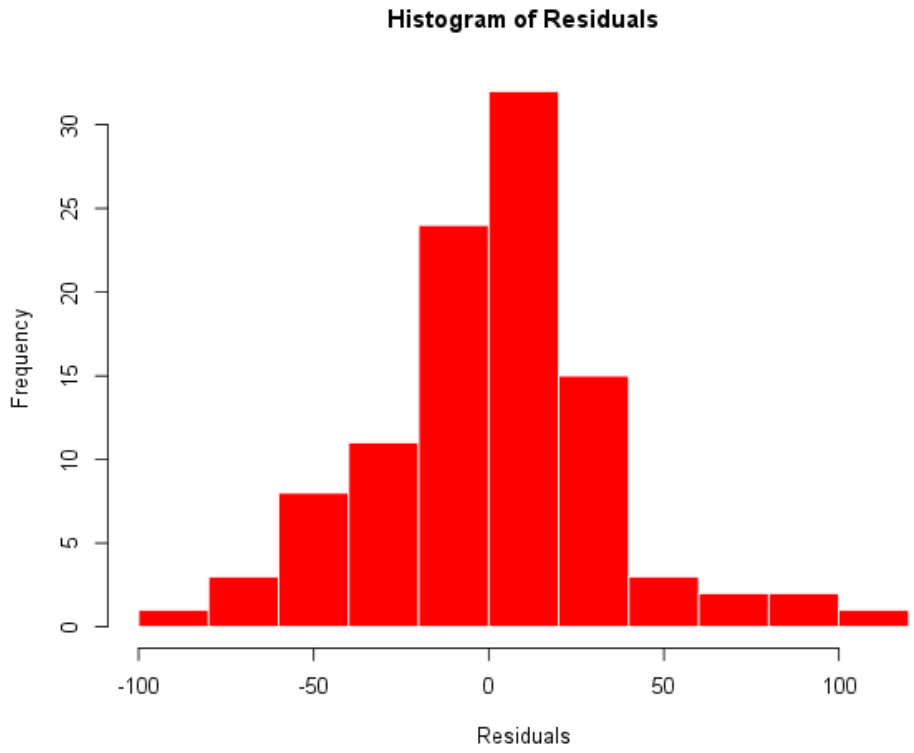



*Hình 37: : Biểu đồ dự đoán lượng khách trung bình ở PortLand trong 12 tháng tiếp theo*

Nhìn vào biểu đồ dự báo trên, đường màu xanh là phần dự đoán lượng khách trung bình ở Portland trong 12 tháng tiếp theo. Có thể thấy phần dự đoán có xu hướng đi lên và không biến động nhiều và có khoảng tin cậy tăng dần lên. Nhìn chung có thể thấy tình hình khách tham gia phương tiện giao thông ở Portland sẽ có xu hướng tăng.

Tiếp theo chúng ta thực hiện vẽ đồ thị phần dư

```
#%% - Residuals Plot
hist(model$residuals, col="blue", border = "white",
xlab="Residuals", main= "Histogram of Residuals")
```




*Hình 38: Đồ thị phần dư*

*#%% - Accuracy*

*accuracy*(forecast_model,test_data)

Sử dụng accuracy để hiển thị các chỉ số về độ chính xác thường được sử dụng để đánh giá dự báo chuỗi thời gian là:

• Lỗi tỷ lệ phần trăm tuyệt đối trung bình (MAPE)

• Lỗi trung bình (ME)

• Lỗi tuyệt đối trung bình (MAE)

• Lỗi tỷ lệ phần trăm trung bình (MPE)

• Lỗi bình phương trung bình gốc (RMSE)

• Lỗi tự động tương quan trễ 1 (ACF1)

```
> accuracy(forecast_model,test_data)
                     ME     RMSE      MAE         MPE     MAPE      MASE
Training set  -0.7987827 32.97908 23.61119 -0.05231924 2.057060 0.1996437
Test set     -35.0738659 49.10022 41.75457 -2.63900770 3.099447 0.3530545
                   ACF1 Theil's U
Training set 0.06299949        NA
Test set     0.21707751 0.6758222
```

*Hình 39: Các chỉ số về độ chính xác*

Nhìn vào các chỉ số trong hình, với 3.1% MAPE có nghĩa là mô hình có độ chính xác khoảng 100% - 3.1% = 96,9% trong việc dự đoán.