Chapter 17: Linear Regression

Exercise 2: Petrol consumption

Yêu cầu 1: Áp dụng Line Regression để dự đoán Petrol_Consumption dựa trên Petrol_tax, Population_Driver_licence(%)

Cho dữ liệu petrol_consumption.csv. Hãy áp dụng Line Regression để dự đoán Petrol_Consumption dựa trên Petrol_tax, Population_Driver_licence(%)

- Đọc dữ liệu và gán cho biến data.
- Xem thông tin data: head(), số dòng, số cột, str, summary
- Vẽ biểu đồ quan sát mối liên hệ giữa Petrol_tax với Petrol_Consumption, Average_income với Petrol_Consumption, Paved_Highways với Petrol_Consumption, Population_Driver_licence(%) với Petrol_Consumption
- Kiểm tra outliers => loại outliers
- Tạo train:test từ dữ liệu data với tỉ lệ 80:20
- Thực hiện Linenear Regression với train.
- In summary của model
- Dự đoán y_pred từ test => so sánh với y_test
- Tính Mean Square Error (mse)
- Tính Coefficients, Intercept và Variance score
- Nhận xét dựa trên kết quả

Yêu cầu 2: Áp dụng BMA cho dữ liệu trên để lựa chọn model với các thuộc tính phù hợp cho việc dùng Linear Regression dự đoán Petrol_Consumption

Gợi ý:

Yêu cầu 1

In [1]:
```
# dataset understanding
data <- read.csv("petrol_consumption.csv")
print(is.data.frame(data))
print(paste("cols",ncol(data)))
print(paste("rows:",nrow(data)))
```

```
[1] TRUE
[1] "cols 5"
[1] "rows: 48"
```

In [2]:
```
print(head(data))
```

```
  Petrol_tax Average_income Paved_Highways Population_Driver_licence...
1        9.0           3571           1976                        0.525
2        9.0           4092           1250                        0.572
3        9.0           3865           1586                        0.580
4        7.5           4870           2351                        0.529
5        8.0           4399            431                        0.544
6       10.0           5342           1333                        0.571
  Petrol_Consumption
1               541
2               524
3               561
4               414
5               410
6               457
```

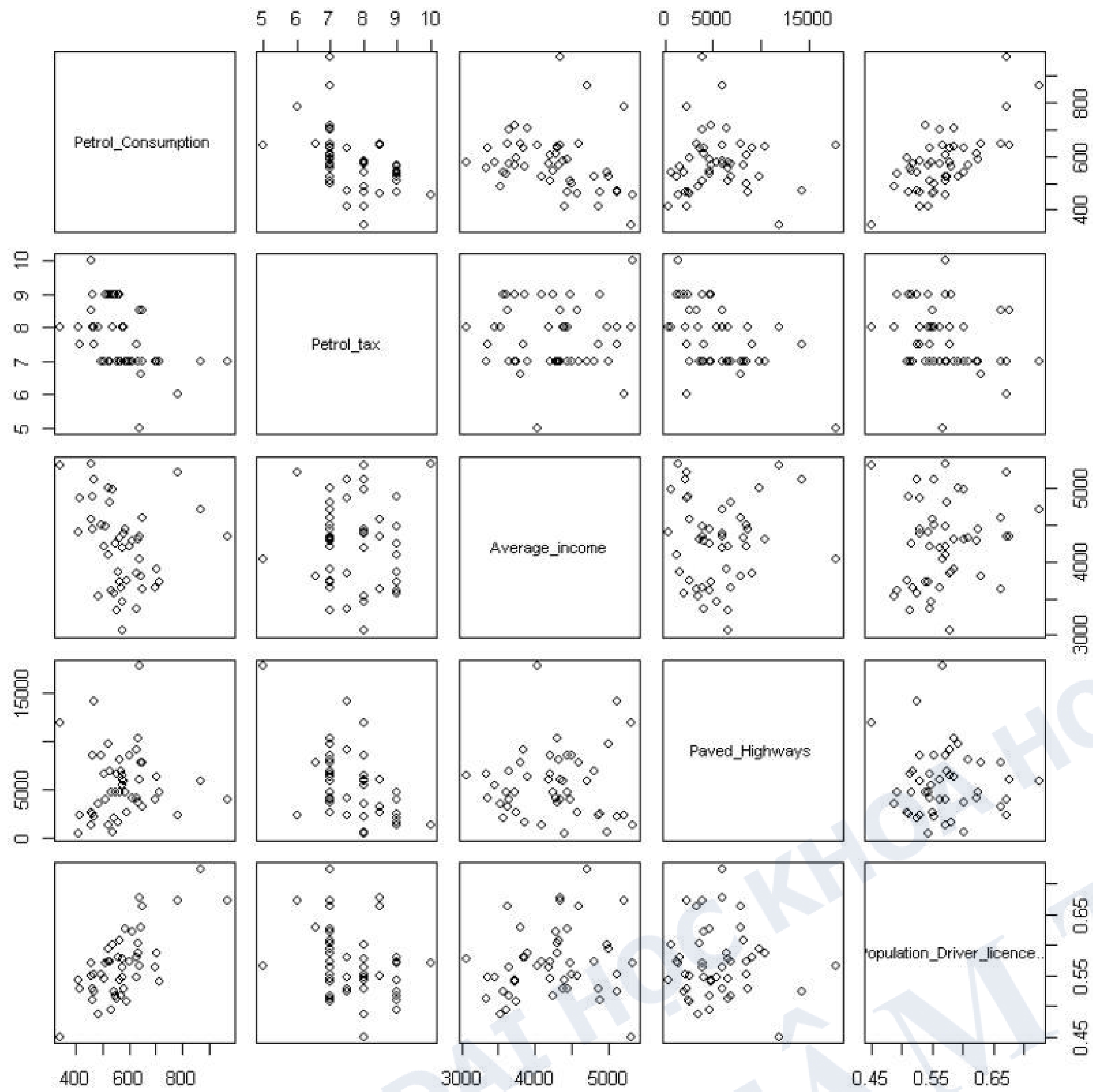
In [3]:
```
str(data)
```

```
'data.frame':   48 obs. of  5 variables:
 $ Petrol_tax                  : num  9 9 9 7.5 8 10 8 8 8 7 ...
 $ Average_income              : int  3571 4092 3865 4870 4399 5342 5319 5126 4
447 4512 ...
 $ Paved_Highways              : int  1976 1250 1586 2351 431 1333 11868 2138 8
577 8507 ...
 $ Population_Driver_licence...: num  0.525 0.572 0.58 0.529 0.544 0.571 0.451
0.553 0.529 0.552 ...
 $ Petrol_Consumption          : int  541 524 561 414 410 457 344 467 464 498
...
```

In [4]:
```
summary(data)
```

```
   Petrol_tax    Average_income Paved_Highways  Population_Driver_licence...
 Min.   : 5.000   Min.   :3063   Min.   :  431   Min.   :0.4510
 1st Qu.: 7.000   1st Qu.:3739   1st Qu.: 3110   1st Qu.:0.5298
 Median : 7.500   Median :4298   Median : 4736   Median :0.5645
 Mean   : 7.668   Mean   :4242   Mean   : 5565   Mean   :0.5703
 3rd Qu.: 8.125   3rd Qu.:4579   3rd Qu.: 7156   3rd Qu.:0.5952
 Max.   :10.000   Max.   :5342   Max.   :17782   Max.   :0.7240
 Petrol_Consumption
 Min.   :344.0
 1st Qu.:509.5
 Median :568.5
 Mean   :576.8
 3rd Qu.:632.8
 Max.   :968.0
```

In [5]:
```
# visualization
pairs(~Petrol_Consumption+Petrol_tax+Average_income+Paved_Highways+Population_Dr:
      data = data)
```

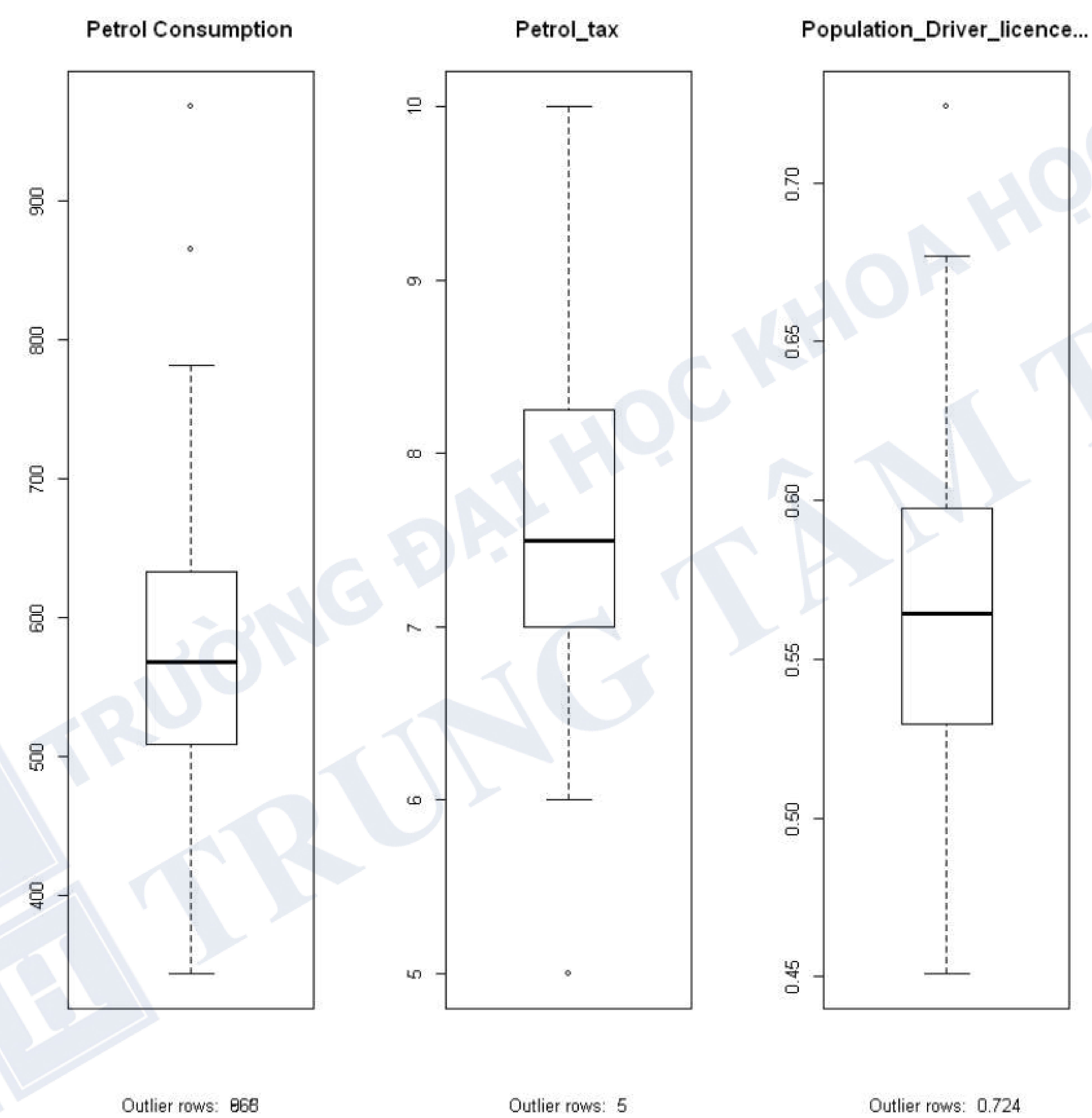


In [6]:
```
input <- data[,c("Petrol_tax", "Population_Driver_licence...",
                 "Petrol_Consumption")]
print(head(input))
```

```
  Petrol_tax Population_Driver_licence... Petrol_Consumption
1        9.0                        0.525                541
2        9.0                        0.572                524
3        9.0                        0.580                561
4        7.5                        0.529                414
5        8.0                        0.544                410
6       10.0                        0.571                457
```

In [7]:

```
# BoxPlot to Check for outliers
par(mfrow=c(1, 3))  # divide graph area in 3 columns
boxplot(input$Petrol_Consumption,
        main="Petrol Consumption",
        sub=paste("Outlier rows: ",
                  boxplot.stats(input$Petrol_Consumption)$out))
boxplot(input$Petrol_tax,
        main="Petrol_tax",
        sub=paste("Outlier rows: ",
                  boxplot.stats(input$Petrol_tax)$out))
boxplot(input$Population_Driver_licence...,
        main="Population_Driver_licence...",
        sub=paste("Outlier rows: ",
                  boxplot.stats(input$Population_Driver_licence...)$out))
```



In [8]:

```
pc_outliers <- boxplot.stats(input$Petrol_Consumption)$out
print("pc_outliers: ")
print(pc_outliers)

pt_outliers <- c(boxplot.stats(input$Petrol_tax)$out)
print("pt_outliers: ")
print(pt_outliers)

pd_outliers <- c(boxplot.stats(input$Population_Driver_licence...)$out)
print("pd_outliers: ")
print(pd_outliers)
```

```
[1] "pc_outliers: "
[1] 865 968
[1] "pt_outliers: "
[1] 5
[1] "pd_outliers: "
[1] 0.724
```

In [9]:

```
#drop rows have outliers
print(paste("Before drop:", nrow(input)))

for (record in pc_outliers){
  input <- input[input$Petrol_Consumption != record,]
}
for (record in pt_outliers)
{
  input <- input[input$Petrol_tax != record,]
}
for (record in pd_outliers)
{
  input <- input[input$Population_Driver_licence... != record,]
}

print(paste("After drop:", nrow(input)))
```

```
[1] "Before drop: 48"
[1] "After drop: 45"
```

In [10]:

```
# calculate correlation between
print("Correlations pc vs pt and pdl:")
print(cor(input$Petrol_Consumption,
          input$Petrol_tax))
print(cor(input$Petrol_Consumption,
          input$Population_Driver_licence...))
```

```
[1] "Correlations pc vs pt and pdl:"
[1] -0.4629515
[1] 0.6052256
```

In [11]:
```
# Create the training (development) and test (validation) data.
set.seed(42)  # setting seed to reproduce results of random sampling
trainingRowIndex <- sample(1:nrow(input), 0.8*nrow(input))
print("Selected training row indexes:")
print(trainingRowIndex)
trainingData <- input[trainingRowIndex, ] # training data
testData  <- input[-trainingRowIndex, ]   # test data
print("Rows of training data and test data:")
print(nrow(trainingData))
print(nrow(testData))
```

```
[1] "Selected training row indexes:"
 [1] 42 45 13 35 27 21 29  6 25 26 17 37 31  9 15 39 30  4 43 33 23 28 34 40  2
[26] 11  8 44 19 14 12 36 38 32  1 16
[1] "Rows of training data and test data:"
[1] 36
[1] 9
```

In [12]:
```
# Create the relationship model.
lmMod <- lm(Petrol_Consumption~Petrol_tax+Population_Driver_licence...,
            data = trainingData)
```

In [13]:
```
cPred <- predict(lmMod, testData)  # predict Petrol Consumption

# mean square error according to model
mse <- mean(lmMod$residuals^2)
print(paste("mse: ", mse))

# mean square error of testData
mse_test = mean((testData$Petrol_Consumption - cPred)^2)
print(paste("mse in test: ", mse_test))
```

```
[1] "mse:  3188.42645742408"
[1] "mse in test:  7706.63929472164"
```

In [14]:
```
# Show the model.
print(summary(lmMod))
```

```
Call:
lm(formula = Petrol_Consumption ~ Petrol_tax + Population_Driver_licence...,
    data = trainingData)

Residuals:
    Min      1Q  Median      3Q     Max 
-128.93  -50.19    9.01   43.28  114.56 

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)    
(Intercept)                    311.51     175.40   1.776 0.084953 .  
Petrol_tax                     -27.13      11.65  -2.328 0.026180 *  
Population_Driver_licence...   822.05     220.98   3.720 0.000739 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 58.98 on 33 degrees of freedom
Multiple R-squared:  0.4469,	Adjusted R-squared:  0.4134 
F-statistic: 13.33 on 2 and 33 DF,  p-value: 5.698e-05
```

In [15]:
```
# => r^2 has low value, this model fits ~ 45% data => not good!
```

In [16]:
```
# Get the Intercept and coefficients as vector elements.
cat("# # # # The Coefficient Values # # # ","\n")

b <- coef(lmMod)[1]
print(b)

mph <- coef(lmMod)[2]
mpd <- coef(lmMod)[3]

print(mph)
print(mpd)
```

```
# # # # The Coefficient Values # # # 
(Intercept) 
   311.5122 
Petrol_tax 
 -27.12654 
Population_Driver_licence... 
                     822.047 
```

In [17]:
```
# new predictions
#pt = 9, pd = 0.58
x1 <- 9
x2 <- 0.58


y <- (mph*x1 + mpd*x2 + b)
print("Solution 1 - results:")
print(y)

# solution 2
y1 <- predict(lmMod, data.frame(Petrol_tax = x1,
                                Population_Driver_licence... = x2))
print("Solution 2 - results:")
print(y1)
```

```
[1] "Solution 1 - results:"
Petrol_tax
  544.1606
[1] "Solution 2 - results:"
       1
544.1606
```

Yêu cầu 2: Áp dụng BMA cho dữ liệu trên để lựa chọn model với các thuộc tính phù hợp cho việc dùng Linear Regression dự đoán Petrol_Consumption

In [18]:
```
library(BMA)
```

```
Loading required package: survival

Loading required package: leaps

Loading required package: robustbase


Attaching package: 'robustbase'


The following object is masked from 'package:survival':

    heart


Loading required package: inline

Loading required package: rrcov

Scalable Robust Estimators with High Breakdown Point (version 1.4-3)
```

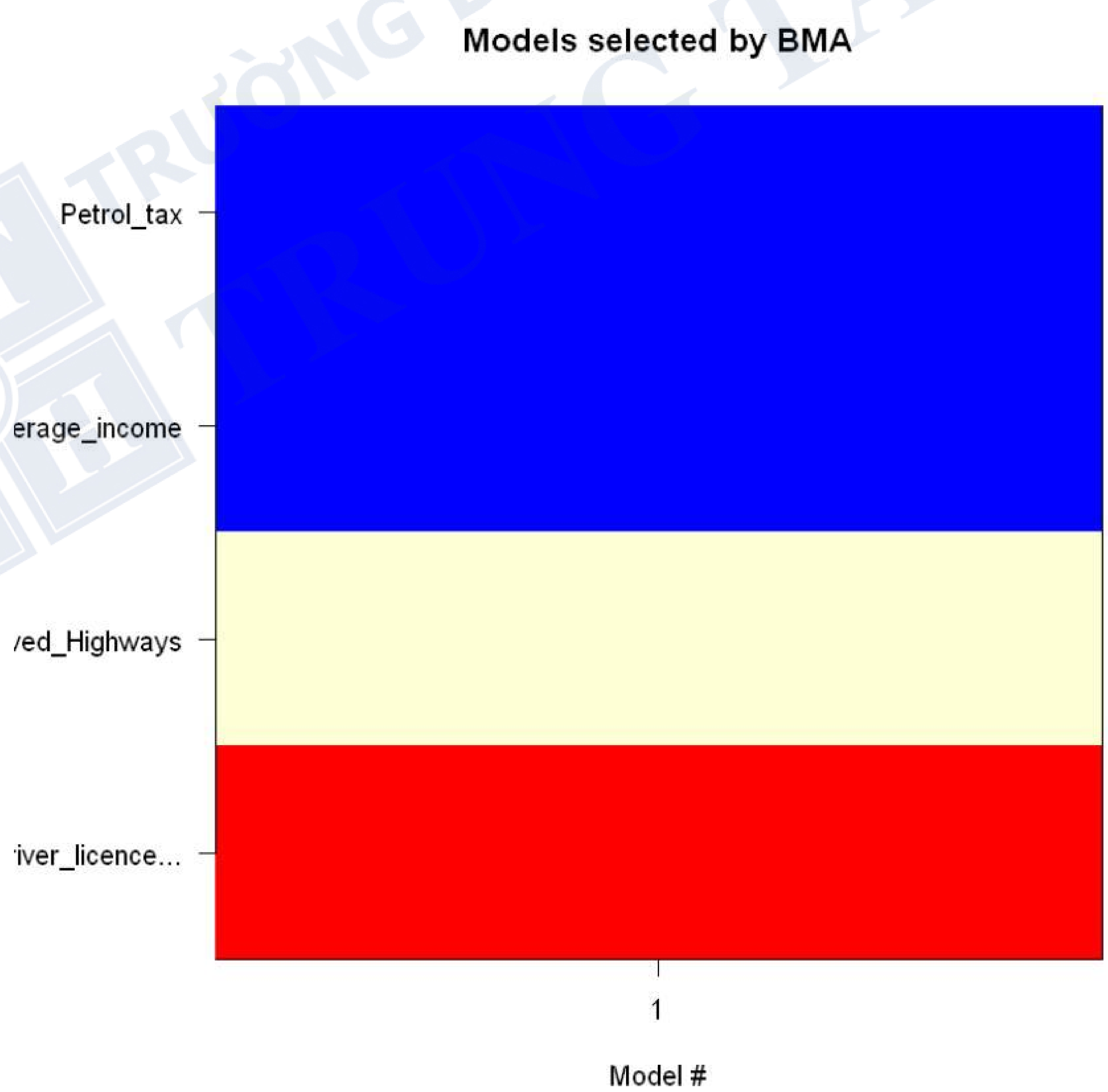
In [19]:
```
yvar = data[, ("Petrol_Consumption")]
xvars = data[, c(-5)]
bma = bicreg(xvars, yvar, strict = F, OR=2)
summary(bma)
imageplot.bma(bma)
```

```
Call:
bicreg(x = xvars, y = yvar, strict = F, OR = 2)


  1  models were selected
 Best  1  models (cumulative posterior probability =  1 ):

                              p!=0   EV          SD          model 1
Intercept                     100     307.32790  156.83067   307.32790
Petrol_tax                    100     -29.48381   10.58358   -29.48381
Average_income                100      -0.06802    0.01701    -0.06802
Paved_Highways                  0       0.00000    0.00000        .
Population_Driver_licence...  100    1374.76841  183.66954  1374.76841

nVar                                                                3
r2                                                              0.675
BIC                                                         -42.31437
post prob                                                         1
```



In [20]:
```
# Select model with: Petrol_tax, Average_income, Population_Driver_licence...
```