ĐẠI HỌC QUỐC GIA

ĐẠI HỌC BÁCH KHOA TP HỒ CHÍ MINH

BÀI TẬP LỚN MÔN XÁC SUẤT THỐNG KÊ

LỚP L14 --- NHÓM 6 --- HK 201

Giảng viên hướng dẫn: Nguyễn Kiều Dung

STT	Sinh viên thực hiện	Mã số sinh viên	Lớp	Ngành học	Ký tên tham dự
1	Vũ Đình Phú	1914674	L06	Khoa học Máy Tính	
2	Lâm Thiện Toàn	1915540	L06	Khoa học Máy Tính	
3	Ngô Đức Trí	1915656	L06	Khoa học Máy Tính	
4	Lý Thanh Bách	1910038	L14	Khoa học Máy Tính	
5	Ngô Thị Hà Bắc	1912700	L14	Khoa học Máy Tính	
6	Hoàng Đình Thành	1915130	L14	Khoa học Máy Tính	
7	**Huỳnh Đức Thịnh**	1910563	L14	Khoa học Máy Tính	
8	Nguyễn Văn Xuân Vũ	1915982	L14	Khoa học Máy Tính	

Thành phố Hồ Chí Minh – 2020

Table of Contents

I. PHẦN CHUNG:

1. Lý thuyết về hồi quy tuyến tính bội:

Phương trình hồi quy bội:

Giả sử mối quan hệ giữa biến phụ thuộc (biến phản hồi) Y và k biến độc lập (biến hồi quy)

$$Y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \varepsilon$$

Trong đó $\beta_0, \beta_1, \ldots, \beta_k$ là các tham số chưa biết, gọi là các hệ số hồi quy, β_0 gọi là hệ số chặn, $\beta_1, \ldots, \beta_k$ là các hệ số góc; ε là sai số ngẫu nhiên có kỳ vọng 0 và phương sai σ^2

Để tìm hiểu mô hình, tiến hành n quan sát và ghi lại kết quả dưới dạng bảng:

y	x_1	x_2	.	x_k
y_1	x_{11}	x_{12}	.	x_{1k}
.	.	.	.	
y_n	x_{n1}	x_{n2}	.	x_{nk}

Như vậy, dưới dạng quan sát, mô hình được viết lại dưới dạng:

$$\begin{cases} y_1 = \beta_0 + \beta_1 x_{11} + \ldots + \beta_k x_{1k} + \varepsilon_1 \\ \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \\ y_n = \beta_0 + \beta_1 x_{n1} + \ldots + \beta_k x_{nk} + \varepsilon_n \end{cases}$$

Để thuận lợi cho ký hiệu và các phân tích tiếp theo, chúng ta sử dụng các ký hiệu ma trận sau đây.

$$\mathbf{y} = \begin{bmatrix} y_1 \\ . \\ y_n \end{bmatrix}; \qquad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & . & . & . & x_{1k} \\ . & . & . & . & . & . & . \\ 1 & x_{n1} & x_{n2} & . & . & . & x_{nk} \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ . \\ \beta_k \end{bmatrix}; \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ . \\ \varepsilon_n \end{bmatrix}$$

Khi đó, phương trình tuyến tính bội được viết lại dưới dạng ma trận:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

Trong đó y là n-véc tơ quan sát, X là ma trận cấp n × p của các biến độc lập (p = k + 1) - còn gọi là ma trận kế hoạch, β là p-véc tơ các hệ số hồi quy, ε là n-véc tơ sai số ngẫu nhiên.

Tuyến tính hóa một số mô hình:

Mô hình hồi quy được gọi là tuyến tính vì nó tuyến tính với các tham số βi . Trong ứng dụng chúng ta thường gặp mô hình dạng

$$E[Y] = \beta_1 g_1(x_1, ..., x_\ell) + ... + \beta_p g_p(x_1, ..., x_\ell)$$

Trong đó g_1, …,g_p là các hàm nào đó của các biến hồi quy của biến x_1,…,x_l

Đây là mô hình tuyến tính với các tham số β_i , phi tuyến với các biến x_1 ,..., x_l . Xét phép đổi biến:

$$z_1 = g_1(x_1, ..., x_\ell); ...; \ z_p = g_p(x_1, ..., x_\ell).$$

Đưa hệ phương trình gốc về dạng thông thường:

$$E[Y] = \beta_1 z_1 + ... + \beta_p z_p$$

là mô hình tuyến tính với cả tham số lẫn các biến hồi quy. Như vậy từ nay ta vẫn gọi mô hình trên là tuyến tính.

Ước lượng hệ số hồi quy và tính chất của ƯL

Giả thiết đầu tiên cần có là ma trận X có số hàng ít nhất bằng số cột, $p = k + 1 \leq n$, và hạng của nó bằng số cột:

$$\text{Rank}(X) = p$$

Khi đó, ƯL làm cực tiểu tổng bình phương các sai số:

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{n} \varepsilon_i^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})$$

gọi là ƯL bình phương cực tiểu, ký biệu là $\hat{\beta}$, cho bởi:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Theo dạng quan sát được, ƯL cho sai số chung của mô hình là

$$\hat{\sigma}^2 = \frac{1}{n-p}\sum_{i=1}^{n} e_i^2 = \frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \ldots + \hat{\beta}_k x_{ik}$$

$$e_i = y_i - \hat{y}_i$$

y_i mũ là dự báo của quan sát thứ i, e_i là sai số cho dự báo thứ i

Kết luận:

$\hat{\beta}$ là ƯL không chệch của véc tơ tham số β: $E[\hat{\beta}] = \beta$

Ma trận covarian của $\hat{\beta}$ cho bởi:

$$Cov(\hat{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2$$

$\hat{\sigma}^2$ là ƯL không chệch của σ^2:

$$E[\hat{\sigma}^2] = \sigma^2$$

Nhận thấy vế phải của phương trình tính phương sai phần chung có chứa mẫu số (n – p) Vậy, khi số biến hồi quy p tăng lên, (chẳng hạn với hồi quy đa thức, khi số bậc của đa thức tăng) có thể sai số mô hình tăng lên. Ta sẽ có mô hình cực tồi nếu p gần bằng n.

Để nghiên cứu các tính chất của ƯL tham số, giống với trường hợp có 1 biến hồi quy, cần có giả thiết:

$\varepsilon_1, \ldots, \varepsilon_n$ độc lập, cùng phân bố chuẩn $N(0, \sigma^2)$

Kiểm định giả thuyết

- Kiểm định ý nghĩa của hồi quy. Đó là kiểm tra xem có một quan hệ tuyến tính nào đó giữa biến phản hồi Y với một tập con nào đó của các biến hồi quy $x_1, \ldots, x_k$ hay không. Cụ thể là xét bài toán kiểm định:

$H_0: \beta_1 = \beta_2 = \ldots = \beta_n = 0$ / $H_1: \beta_j \neq 0$ với ít nhất một $j \in \{1, \ldots, k\}$.

Nếu H0 bị bác bỏ thì có nghĩa là ít ra một trong các biến hồi quy x_1, ..., x_k có ý nghĩa đối với mô hình.

Dưới giả thuyết H_0 có thể chứng minh tổng bình phương hồi quy và tổng bình phương các sai số là những BNN độc lập và có bậc tự do tương ứng là k và n - p. Thế thì

$$F_0 = \frac{SS_R / k}{SS_E / (n-p)} = \frac{MS_R}{MS_E} \sim F(k; n-p).$$

Từ đó giả thuyết bị bác bỏ ở mức α nếu $F_0 \geq f_\alpha (k; n - p)$.

Các phần mềm thường dùng P–giá trị và đưa ra bảng phân tích phương sai cho thủ tục vừa nêu.

Người ta cũng xét kiểm định cho một tập con của các hệ số β_0, β_1, ..., β_k bằng 0.

- Hệ số xác định bội R^2 và hệ số xác định hiệu chỉnh R^2_{adj}

Với mô hình hồi quy nhiều biến định nghĩa hệ số xác định bội R^2 và các tính chất của nó như với trường hợp hồi quy đơn:

$$R^2 = \frac{SS_R}{SS_T} = 1 - \frac{SS_E}{SS_T}.$$

Tính chất đặc biệt của hệ số xác định là nó không giảm khi tăng số biến hồi quy. Từ đó, hệ số xác định khó nói cho ta biết việc tăng biến có lợi gì hay không, nhất là khi sự gia tăng hệ số xác định là nhỏ. Vì thế nhiều nhà phân tích lại thích dùng hệ số xác định hiệu chỉnh (adjusted R^2):

$$R^2_{adj} = 1 - \frac{SS_E / (n-p)}{SS_T / (n-1)}.$$

Mẫu ở vế phải là hằng số, còn tử là ước lượng của sai số; nó bé nhất khi và chỉ khi hệ số xác định hiệu chỉnh R^2_{adj} lớn nhất. Từ đó, một quy tắc lựa chọn biến hồi quy là: chọn một số trong các biến hồi quy x_1, …, x_k để R^2_{adj} lớn nhất.

- Kiểm định một tham số triệt tiêu (kiểm định T).

Xét bài toán kiểm định một tham số đơn lẻ nào đó triệt tiêu:

$$H_0: \beta_j = 0 / \ H_1: \beta_j \neq 0 \quad (j = 0,1,...,k).$$

Nếu giả thuyết không bị bác bỏ thì có nghĩa rằng biến hồi quy tương ứng không bị loại khỏi mô hình. Thống kê kiểm định là

$$T_j = \frac{\hat{\beta}_j}{se(\hat{\beta}_j)} = \frac{\hat{\beta}_j}{\sqrt{\hat{\sigma}^2 C_{jj}}}$$

trong đó C_{jj} là phần tử thứ j của đường chéo chính của ma trận $\mathbf{C} = (\mathbf{X}'\mathbf{X})^{-1}$ ứng với $\hat{\beta}_j$.

Vì $T_j \sim T(n-p)$ nên giả thuyết bị bác bỏ nếu $|T_j| > t_{\alpha/2}(n-p)$.

Ước lượng và dự đoán

- Khoảng tin cậy cho tham số đơn lẻ.

Khoảng tin cậy 100(1 - α)% cho tham số β_j cho bởi

$$\hat{\beta}_j \pm t_{\alpha/2}(n-p)\, se(\hat{\beta}_j), \quad (se(\hat{\beta}_j) = \sqrt{\hat{\sigma}^2 C_{jj}}).$$

- Khoảng tin cậy đáp ứng trung bình.

Giả sử quan sát tương lai thực hiện tại mức x_{01}, ..., x_{0k} của các biến hồi quy x_1, ..., x_k. Đặt $x_0 = (1, x_{01}, ..., x_{0k})^T$. Đáp ứng trung bình tại điểm này là $E[Y|x_0] = x_0^T\beta = \beta_0 + \beta_1 x_{01} + \beta_k x_{0k}$, ƯL điểm của nó là

$$\hat{y}_0 = \mathbf{x}_0^{\prime}\hat{\boldsymbol{\beta}} = \hat{\beta}_0 + \hat{\beta}_1 x_{01} + ... + \hat{\beta}_k x_{0k}.$$

Đối với MHHQ tuyến tính bội, khoảng tin cậy 100(1 - α)% cho đáp ứng trung bình tại điểm x_{01}, ..., x_{0k} là

$$\hat{y}_0 \pm t_{\alpha/2}(n-p)\sqrt{\hat{\sigma}^2 \mathbf{x}_0^T (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_0}$$

- Dự đoán cho quan sát mới. ƯL điểm của dự đoán cho quan sát tương lai tại mức x_{01}, ..., x_{0k} của các biến độc lập là

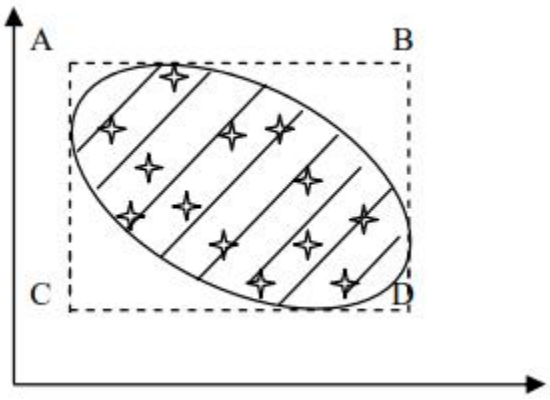
$$\hat{y}_0 = \mathbf{x}_0^T\boldsymbol{\beta} = \beta_0 + \beta_1 x_{01} + \ldots + \beta_k x_{0k}.$$

Khoảng dự đoán 100(1 - α)% cho quan sát tương lai này là

$$\hat{y}_0 \pm t_{\alpha/2}(n-p)\sqrt{\hat{\sigma}^2(1+\mathbf{x}_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_0)}.$$

- Vấn đề ngoại suy với mô hình hồi quy bội.

Vẫn có những chú ý tương tự như với hồi quy đơn, song vấn đề cần thận trọng hơn. Chẳng hạn, với mô hình có hai biến hồi quy x, y miền biến thiên của các biến hồi quy ở hình sau phải hiểu là elip chứ không phải hình chữ nhật ABCD. Tình hình sẽ khó khăn hơn khi số biến hồi quy tăng lên



Miền biến thiên của các biến hồi quy

2. Phần thực hành:

Bài tập 1. Tập tin "gia_nha.csv" chứa thông tin về giá bán ra thị trường (đơn vị đô la) của 21613 ngôi nhà ở quân King nước Mỹ trong khoảng thời gian từ tháng 5/2014 đến 5/2015. Bên cạnh giá nhà, dữ liệu còn bao gồm các thuộc tính mô tả chất lượng ngôi nhà. Dữ liệu gốc được cung cấp tại: https://www.kaggle.com/harlfoxem/housesalesprediction.

Các biến chính trong bộ dữ liệu:

- **price**: Giá nhà được bán ra.
- **sqft_living15**: Diện tích trung bình của 15 ngôi nhà gần nhất trong khu dân cư.
- **floors**: Số tầng của ngôi nhà được phân loại từ 1-3.5.

- **condition**: Điều kiện kiến trúc của ngôi nhà từ 1 − 5, 1: rất tệ và 5: rất tốt.
- **sqft_above**: Diện tích ngôi nhà.
- **sqft_living**: Diện tích khuôn viên nhà.

Bài làm

Thiết lập thư viện dùng cho cả bài:

Ta import các thư viện như sau:

```
library(stats)
library(dplyr)
library(magrittr)
```

Công dụng của các thư viện:

- stats: chứa các hàm tính toán cho thống kê và hàm sinh random.
- dplyr: chứa những hàm quan trọng nhất trong việc biến đổi dữ liệu cần thiết cho phân tích dữ liệu.
- magrittr: có một số cú pháp để rút gọn code cho dễ đọc, dễ sửa đổi, và giảm thời gian lập trình. (ví dụ như x%>%f thay vì f(x))

Đọc dữ liệu (Import Data):

Ta dùng lệnh **read.csv()** như sau:

```
#1
df = read.csv('gia_nha.csv');
```

Kết quả ta thu được một data frame gồm 21613 dòngvới rất nhiều fields như sau:

	X.2	X.1	X	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat	long	sqft_living15	sqft_lot15
1	1	1	1	7129300520	20141013T000000	221900	3	1.00	1180	5650	1.0	0	0	3	7	1180	0	1955	0	98178	47.5112	-122.257	1340	5650
2	2	2	2	6414100192	20141209T000000	538000	3	2.25	2570	7242	2.0	0	0	3	7	2170	400	1951	1991	98125	47.7210	-122.319	1690	7639
3	3	3	3	5631500400	20150225T000000	180000	2	1.00	770	10000	1.0	0	0	3	6	770	0	1933	0	98028	47.7379	-122.233	2720	8062
4	4	4	4	2487200875	20141209T000000	604000	4	3.00	1960	5000	1.0	0	0	5	7	1050	910	1965	0	98136	47.5208	-122.393	1360	5000
5	5	5	5	1954400510	20150218T000000	510000	3	2.00	1680	8080	1.0	0	0	3	8	1680	0	1987	0	98074	47.6168	-122.045	1800	7503
6	6	6	6	7237550310	20140512T000000	1225000	4	4.50	5420	101930	1.0	0	0	3	11	3890	1530	2001	0	98053	47.6561	-122.005	4760	101930
7	7	7	7	1321400060	20140627T000000	257500	3	2.25	1715	6819	2.0	0	0	3	7	1715	0	1995	0	98003	47.3097	-122.327	2238	6819
8	8	8	8	2008000270	20150115T000000	291850	3	1.50	1060	9711	1.0	0	0	3	7	1060	0	1963	0	98198	47.4095	-122.315	1650	9711
9	9	9	9	2414600126	20150415T000000	229500	3	1.00	1780	7470	1.0	0	0	3	7	1050	730	1960	0	98146	47.5123	-122.337	1780	8113
10	10	10	10	3793500160	20150312T000000	323000	3	2.50	1890	6560	2.0	0	0	3	7	1890	0	2003	0	98038	47.3684	-122.031	2390	7570
11	11	11	11	1736800520	20150403T000000	662500	3	2.50	3560	9796	1.0	0	0	3	8	1860	1700	1965	0	98007	47.6007	-122.145	2210	8925
12	12	12	12	9212900260	20140527T000000	468000	2	1.00	1160	6000	1.0	0	0	4	7	860	300	1942	0	98115	47.6900	-122.292	1330	6000
13	13	13	13	114101516	20140528T000000	310000	3	1.00	1430	19901	1.5	0	0	4	7	1430	0	1927	0	98028	47.7558	-122.229	1780	12697

Làm sạch dữ liệu (Data cleaning):

Ta sẽ thực hiện trích ra một dữ liệu con đặt tên là **new_DF** chỉ bao gồm các biến mà ta quan tâm. Sau đó, kiểm tra các dữ liệu bị khuyết trong tập tin. Ở đây ta có một số cách xử lí dữ liệu bị khuyết như là thay thế bằng một giá trị trung bình. Tuy nhiên, số lượng dữ liệu bị khuyết rất nhỏ so với tổng dữ liệu nên nhóm quyết định xóa bỏ những hàng chứa các dữ liệu khuyết này.

```
#2
new_DF = df %>% select('price', 'sqft_living15', 'floors','condition',
                       'sqft_above','sqft_living');
new_DF = new_DF %>% filter(!(is.na(price) | is.na(sqft_living15) | is.na(sqft_living)
                           | is.na(floors) | is.na(condition) | is.na(sqft_above)));
```

Kết quả ta thu được **new_DF** gồm 21593 dòng:

	price	sqft_living15	floors	condition	sqft_above	sqft_living
1	221900	1340	1.0	3	1180	1180
2	538000	1690	2.0	3	2170	2570
3	180000	2720	1.0	3	770	770
4	604000	1360	1.0	5	1050	1960
5	510000	1800	1.0	3	1680	1680
6	1225000	4760	1.0	3	3890	5420
7	257500	2238	2.0	3	1715	1715
8	291850	1650	1.0	3	1060	1060
9	229500	1780	1.0	3	1050	1780
10	323000	2390	2.0	3	1890	1890
11	662500	2210	1.0	3	1860	3560
12	468000	1330	1.0	4	860	1160
13	310000	1780	1.5	4	1430	1430

Làm rõ dữ liệu (Data visualization):

a. Chuyển đổi các biến **price**, **sqft_living15**, **sqft_above**, **sqft_living** lần lượt thành log(*price*), log(*sqft_living15*), log(*sqft_above*), và log(*sqft_living*). Ta thực hiện:

```
#3
#a)
new_DF = new_DF %>% rename('log(price)'=price, 'log(sqft_living15)'=sqft_living15,
                           'log(sqft_living)'=sqft_living, 'log(sqft_above)'=sqft_above);
Log = function(x) {log(x)};
new_DF = data.frame(new_DF %>% select('floors','condition'),
                    apply(new_DF %>% select('log(price)', 'log(sqft_living15)','log(sqft_living)','log(sqft_above)'),2,Log));
```

Nhận được kết quả **new_DF** như sau:

	floors	condition	log.price.	log.sqft_living15.	log.sqft_living.	log.sqft_above.
1	1.0	3	12.30998	7.200425	7.073270	7.073270
2	2.0	3	13.19561	7.432484	7.851661	7.682482
3	1.0	3	12.10071	7.908387	6.646391	6.646391
4	1.0	5	13.31133	7.215240	7.580700	6.956545
5	1.0	3	13.14217	7.495542	7.426549	7.426549
6	1.0	3	14.01845	8.468003	8.597851	8.266164
7	2.0	3	12.45877	7.713338	7.447168	7.447168
8	1.0	3	12.58400	7.408531	6.966024	6.966024
9	1.0	3	12.34366	7.484369	7.484369	6.956545
10	2.0	3	12.68541	7.779049	7.544332	7.544332
11	1.0	3	13.40378	7.700748	8.177516	7.528332
12	1.0	4	13.05622	7.192934	7.056175	6.756932
13	1.5	4	12.64433	7.484369	7.265430	7.265430
14	1.0	4	12.89922	7.222566	7.222566	7.222566

b. Đối với các biến liên tục, tính các giá trị thống kê mô tả bao gồm: trung bình, trung vị độ lệch chuẩn, giá trị lớn nhất và giá trị nhỏ nhất. Xuất kết quả dưới dạng bảng.

```
#b)colMeans(new_DF[,c(3:6)]);
table = data.frame(Name = c('mean','median','sd','min','max'),
        log.price.= c(mean(new_DF['log.price.']), median(new_DF['log.price.']),sd(new_DF['log.price.']),
                      min(new_DF['log.price.']),max(new_DF['log.price.'])),
        log.sqft_living15.= c(mean(new_DF['log.sqft_living15.']), median(new_DF['log.sqft_living15.']),sd(new_DF['log.sqft_living15.']),
                              min(new_DF['log.sqft_living15.']),max(new_DF['log.sqft_living15.'])),
        log.sqft_living.= c(mean(new_DF['log.sqft_living.']), median(new_DF['log.sqft_living.']),sd(new_DF['log.sqft_living.']),
                            min(new_DF['log.sqft_living.']),max(new_DF['log.sqft_living.'])),
        log.sqft_above.= c(mean(new_DF['log.sqft_above.']), median(new_DF['log.sqft_above.']),sd(new_DF['log.sqft_above.']),
                           min(new_DF['log.sqft_above.']),max(new_DF['log.sqft_above.']))
)
```

Ta nhận được **table** kết quả như sau:

	Name	log.price.	log.sqft_living15.	log.sqft_living.	log.sqft_above.
1	mean	13.047841	7.5394471	7.5503286	7.3948826
2	median	13.017003	7.5175209	7.5548585	7.3524411
3	sd	0.526574	0.3274562	0.4247722	0.4276433
4	min	11.225243	5.9889614	5.6698809	5.6698809
5	max	15.856731	8.7339162	9.5134035	9.1495282

c. Lập bảng tần số đối với các biến phân loại:

```
#c)
ftable(new_DF['floors'])
ftable(new_DF['condition'])
```

Ta nhận được kết quả như hình sau:

```
> ftable(new_DF[,'floors'])
     1   1.5     2   2.5     3   3.5

 10672  1909  8230   161   613     8
> ftable(new_DF[,'condition'])
     1     2     3     4     5

    30   172 14016  5677  1698
```

d. Dùng hàm **hist()** để vẽ đồ thị phân tán của biến **price**.

Code:

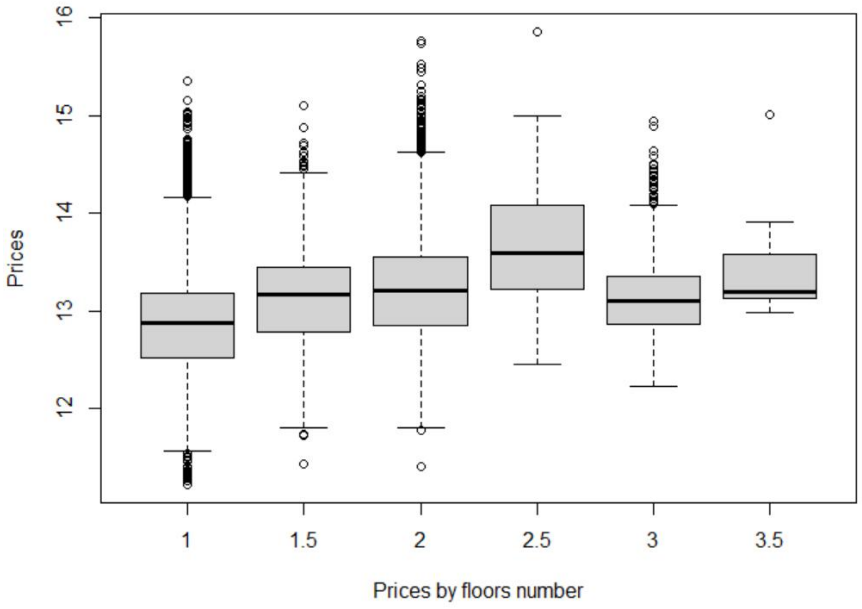
```
hist(new_DF$log.price, main="Distribution of Prices",
         xlab = "Prices", ylab = "Frequency", ylim = c(0,10000));
```

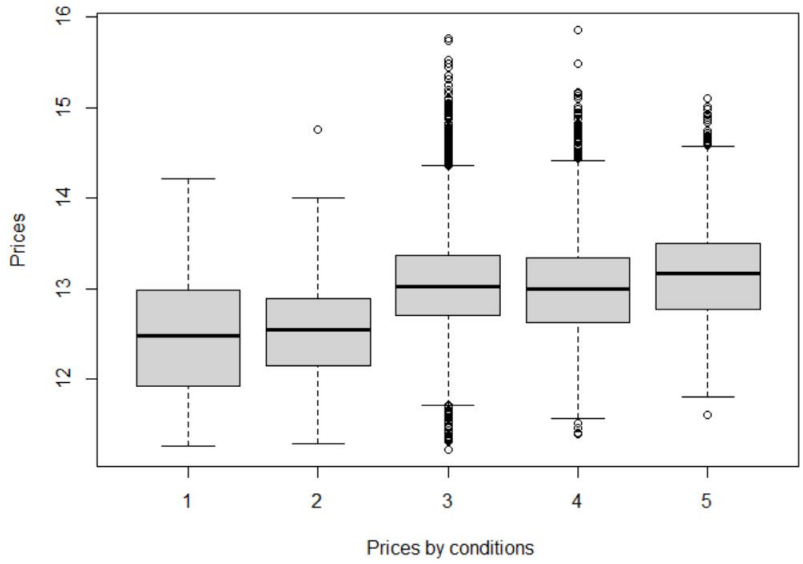
Output:

e. Dùng hàm **boxplot()** vẽ phân phối của biến **price** cho từng nhóm phân loại của biến **floors** và biến **condition**.

```
#e)
boxplot(new_DF$log.price. ~ new_DF$floors, ylab = "Total times", xlab = "Prices by floors numbrt")
boxplot(new_DF$log.price. ~ new_DF$condition, ylab = "Total times", xlab = " Prices by conditions")
```

Kết quả nhận được như sau:





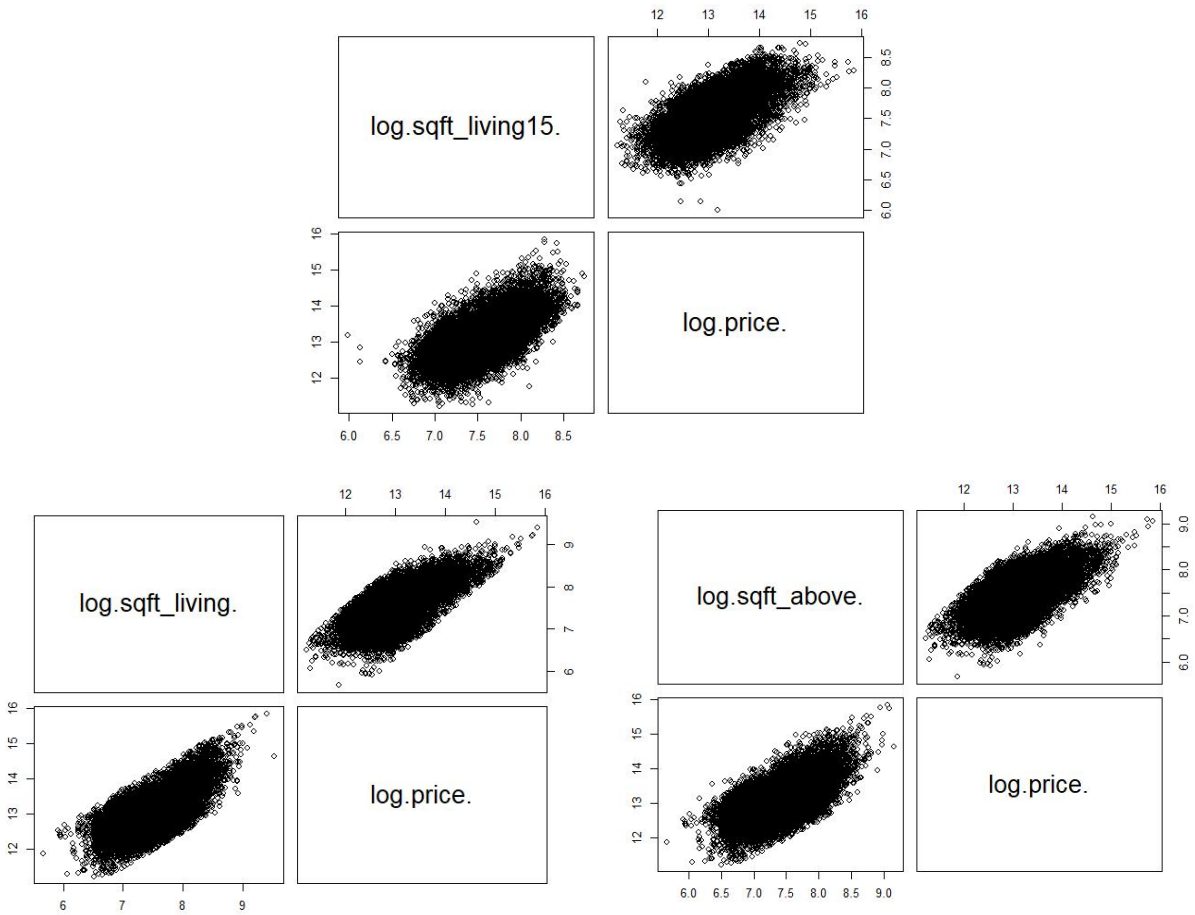
f. Dùng lệnh **pairs()** vẽ các phân phối của biến **price** lần lượt theo các biến **sqft_living15**,

sqft_above, và **sqft_living**.

Code:

```
#f)
pairs(new_DF[,c('log.sqft_living15.','log.price.')])
pairs(new_DF[,c('log.sqft_living.','log.price.')])
pairs(new_DF[,c('log.sqft_above.','log.price.')])
```

Kết quả nhận về là các hình sau:



Xây dựng các mô hình hồi quy tuyến tính (Fitting linear regression models):

Chúng ta muốn khám phá rằng có những nhân tố nào và tác động như thế nào đến giá nhà ở quận King.

a. Xét mô hình hồi quy tuyến tính bao gồm biến **price** là một biến phụ thuộc, và tất cả các biến còn lại đều là biến độc lập. Hãy dùng lệnh **lm()** để thực thi mô hình hồi quy tuyến tính bội.

Code:

```
lmPrice = lm(log.price.~log.sqft_living15.+log.sqft_living.+log.sqft_above.+
             floors+condition, new_DF)
summary(lmPrice)
```

Thu được kết quả **lmPrice** chứa thông tin, các tham số cho chương trình hồi quy, gọi hàm **summary():**

```
Residuals:
     Min       1Q   Median       3Q      Max
-1.25277 -0.27502  0.00764  0.24359  1.50543

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        5.442270   0.062423   87.18   <2e-16 ***
log.sqft_living15. 0.430556   0.011977   35.95   <2e-16 ***
log.sqft_living.   0.686935   0.013186   52.09   <2e-16 ***
log.sqft_above.   -0.178957   0.014021  -12.76   <2e-16 ***
floors             0.137069   0.005952   23.03   <2e-16 ***
condition          0.085465   0.004076   20.97   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3727 on 21587 degrees of freedom
Multiple R-squared:  0.499,     Adjusted R-squared:  0.4989
F-statistic:  4301 on 5 and 21587 DF,  p-value: < 2.2e-16
```

Các thông tin:

- **Residual**: thống kê thông tin về độ lệch so sánh giữa hàm giả thiết và dữ liệu cho trước

- **Coffecients**:

+ **Estimate**: các hệ số của phương trình hồi quy

Price = 5.4423 + 0.4306 * sqft_living15 + 0.6869 * sqft_living – 0.1789 * sqft_above + 0.1371 * floors + 0.085465 * condition

+ **Residual Standard Error**: Sai số chuẩn ước lượng

$$\text{SSE} = \sum_{i=1}^{n} \left(y_i - y_i \right)^2$$

n = 21587; k = 5

$$\delta = \sqrt{\frac{SSE}{n - (1 + k)}} = 0.3727$$

+ **Multiple R-squared**: Hệ số tương quan tuyến tính thể hiện mức độ phù hợp của mô hình

$$\text{SST} = \sum_{i=1}^{n} \left(y_i - \overline{y} \right)^2$$

$$SSE = \sum_{i=1}^{n} \left(y_i - y_i \right)^2$$

$$SSR = \sum_{i=1}^{n} \left(y_i - \overline{y} \right)^2$$

$$R^2 = \frac{SSR}{SST} \times 100\% \qquad hay \quad R^2 = \left(1 - \frac{SSE}{SST} \right) \times 100\%$$

Multiple R-sq = 0.499 Adjusted Multiple R-sq = 0.4989

+ **F-Statistic**: Tiêu chuẩn kiểm định

$$\text{t-stat} = \frac{\frac{SSR}{SSE}}{\frac{}{n-(k+1)}} = 4301$$

Giả thiết:

H_0: $R^2 = 0$

H_1: $R^2 \neq 0$

Miền bác bỏ:

$$W_\alpha = (F_\alpha(1, n-2);\ +\infty) = (3.8419;\ +\infty)$$

Tiêu chuẩn thuộc miền bác bỏ, bác bỏ H_0 thừa nhận H_1, biến **price** có mối quan hệ tuyến tính với tập giá trị input.

b. Dựa vào kết quả P-value (Pr > |t|) của 5 feature đều rất nhỏ, vậy rõ ràng khả năng bác bỏ H_0 của 5 biến trên đều rất cao ($2.2 * 10^{-16} << 0.05$), vậy không bác bỏ bất kỳ feature nào ở mức tin cậy 0.05.

c. Xét 2 mô hình tuyến tính cùng bao gồm biến **price** là biến phụ thuộc nhưng:

- Mô hình M1 chứa tất cả các biến còn lại là biến độc lập
- Mô hình M2 là loại bỏ biến **condition** từ mô hình M1.

Hãy dùng lệnh **anova()** để đề xuất mô hình hồi quy hợp lý hơn.

Mô hình M1 đã được thực hiện ở câu a.

Mô hình M2: loại bỏ **'condition'**

```
Residuals:
     Min       1Q   Median       3Q      Max
-1.29158 -0.27842  0.00637  0.25277  1.47354

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        5.801284   0.060635   95.67   <2e-16 ***
log.sqft_living15. 0.415294   0.012075   34.39   <2e-16 ***
log.sqft_living.   0.722335   0.013210   54.68   <2e-16 ***
log.sqft_above.   -0.203816   0.014112  -14.44   <2e-16 ***
floors             0.112966   0.005899   19.15   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3765 on 21588 degrees of freedom
Multiple R-squared:  0.4888,    Adjusted R-squared:  0.4887
F-statistic:  5161 on 4 and 21588 DF,  p-value: < 2.2e-16
```

Giả thiết:

$$H_0: R^2 = 0$$

$$H_1: R^2 \neq 0$$

Sau khi loại bỏ **‘condition’**, thông số R-sq giảm xuống, tuy vẫn bác bỏ mạnh H_0 thừa nhận tương quan tuyến tính của mô hình, mức bác bỏ vẫn rất cao (P-value = $2.2 * 10^{-16}$ << 0.05) nhưng không tốt bằng mô hình ban đầu.

Sử dụng lệnh **anova()** khi xét mô hình **price** theo **condition:**

```
#d)
lmPricy = lm(log.price.~condition,new_DF[,c('log.price.','condition')])
#summary(lmPricy)
anovaD = anova(lmPricy)
anovaD
```

```
Analysis of Variance Table

Response: log.price.
             Df Sum Sq Mean Sq F value    Pr(>F)
condition     1    9.5  9.4666  34.194 5.061e-09 ***
Residuals 21591 5977.6  0.2769
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Giả thiết:

H_0: Price không phụ thuộc vài sự biến thiên của condition

H_1: Price phụ thuộc vài sự biến thiên của condition

P-value $= 5.061 * 10^{-9} << \alpha = 0.05$. Vậy bác bỏ H_0 tốt, thừa nhận sự phụ thuộc vào sự biến thiên của condition, vậy rõ rang mô hình M1 là mô hình chuẩn xác hơn.

d. Mô hình hợp lý hơn ở câu c là mô hình M1.

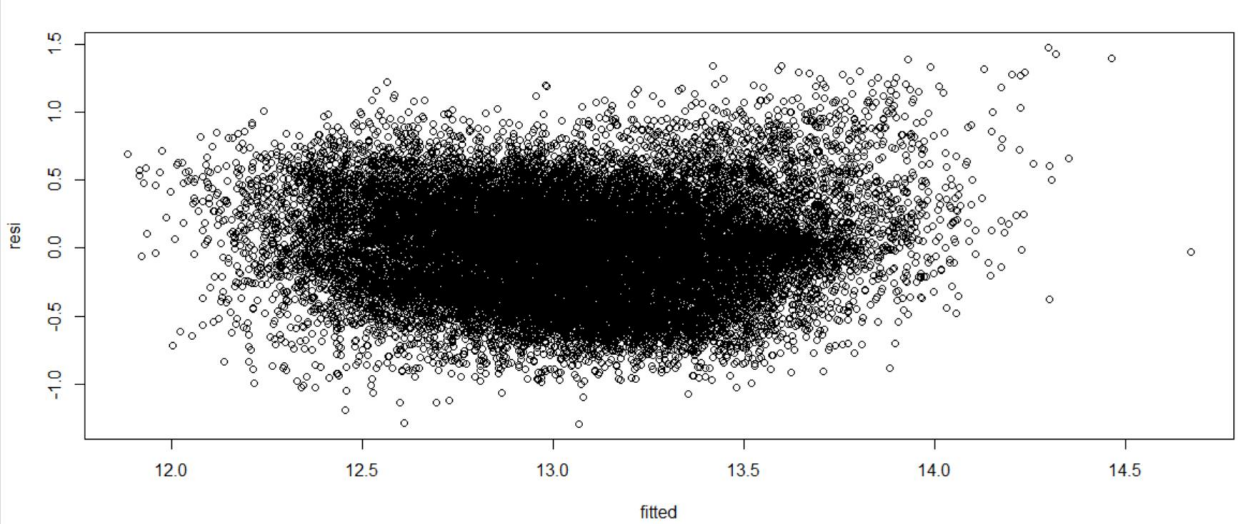
Các biến của M1 tác động vào giá trị **price** theo quan hệ tuyến tính xác định bằng hệ số **estimated coefficient**.

3 giá trị **sqft_living**, **sqft_living15**, **floors** tác động đáng kể vào giá nhà theo tỷ lệ thuận, **conditions** cũng tác động theo xu hướng tăng dần nhưng mức tác động thấp hơn.

Giá trị **sqft_above** có tác động theo chiều tỉ lệ nghịch với giá nhà do hệ số âm

e) Đồ thị biểu diễn sai số hồi quy so với giá trị dự báo:

```
resi = lmPriceC$residuals
fitted = lmPriceC$fitted.values
plot(fitted, resi)
```



Giá nhà dao động phần lớn xung quanh giá trị $10^{12} – 10^{14}$, log (sai số dự báo dao động) từ -1 đến 1 tức dự báo chênh lệch lên đến 10 lần, sự chênh lệch do dự báo độc lập với giá trị dự báo được bởi mô hình

Dự báo (Predictions):

Để tiến hành dự đoán giá nhà tại 2 thuộc tính x1 và x2, trước tiên ta cần phải tạo 2 data frame x1 chứa các biến sqft_living15 = mean(sqft_living15), sqft_living = mean(sqft_living), sqft_above = mean(sqft_above), floors = 2, condition = 3 và x2 chứa các biến sqft_living15 = max(sqft_living15), sqft_living = max(sqft_living), sqft_above = max(sqft_above), floors = 2, condition = 3. Sau đó sử dụng lệnh predict với mô hình hồi quy tuyến tính thu được ở câu 4c và độ tin cậy 95%.

Code:

```
x1 = data.frame(log.sqft_living15. = mean(new_DF[,'log.sqft_living15.']),
    log.sqft_living. = mean(new_DF[,'log.sqft_living.']), log.sqft_above. =
    mean(new_DF[,'log.sqft_above.']), floors = 2, condition = 3)
x2 = data.frame(log.sqft_living15. = max(new_DF[,'log.sqft_living15.']),
    log.sqft_living. = max(new_DF[,'log.sqft_living.']), log.sqft_above. =
    max(new_DF[,'log.sqft_above.']), floors = 2, condition = 3)
predict(lmPrice, x1, interval = "confidence")
predict(lmPrice, x2, interval = "confidence")
```

Ta thu được kết quả như sau:

```
>
> predict(lmPrice, x1, interval = "confidence")
       fit      lwr     upr
1 13.08217 13.07425 13.0901
> predict(lmPrice, x2, interval = "confidence")
       fit      lwr      upr
1 14.63096 14.60666 14.65525
>
```

Khoảng tin cậy cho giá trị price dự đoán, tính bằng upr – lwr, được ở x2 lớn hơn x1.

II. PHẦN RIÊNG:

Tập tin "**SkillCraft1_Dataset.csv**" chứa thông tin nghiên cứu Phép đo từ xa trò chơi điện tử như một công cụ quan trọng trong nghiên cứu học kỹ năng phức tạp của Thompson, Blair, Chen, & Henrey (2013). Các thuộc tính dữ liệu bao gồm số ID, tuổi người chơi, số giờ chơi mỗi tuần, tổng số giờ chơi được báo cáo và các số lượng đơn vị được thực hiện bằng phím trên mỗi dấu thời gian. Dữ liệu được đo từ xa của nhiều người chơi ở một số cấp độ kỹ năng. Dữ liệu gốc được cung cấp tại:
https://archive.ics.uci.edu/ml/datasets/SkillCraft1%2BMaster%2BTable%2BDataset

Các biến chính trong bộ dữ liệu :

- **Age**: Tuổi của người chơi.
- **HoursPerWeek**: Số giờ chơi được báo cáo mỗi tuần.
- **TotalHours**: Tổng số giờ chơi được báo cáo.
- **APM**: Hành động mỗi phút .
- **SelectByHotkeys**: Số lượng đơn vị hoặc lựa chọn tòa nhà được thực hiện bằng phím nóng trên mỗi dấu thời gian.
- **UniqueHotkeys**: Số lượng phím nóng duy nhất được sử dụng trên mỗi dấu thời gian.
- **MinimapAttacks**: Số hành động tấn công trên bản đồ nhỏ trên mỗi dấu thời gian .
- **MinimapRightClicks**: Số lần nhấp chuột phải vào bản đồ nhỏ trên mỗi dấu thời gian .

  (1 giây tương ứng 88.5 dấu thời gian)

Đọc dữ liệu: (Import Data)

Ta dùng lệnh **read.csv()** như sau:

```
# 1)
# Read data from csv file
data_fr = read.csv('SkillCraft1_Dataset.csv')
```

Kết quả ta thu được một data frame với rất nhiều fields như sau :

	GameID	LeagueIndex	Age	HoursPerWeek	TotalHours	APM	SelectByHotkeys	AssignToHotkeys	UniqueHotkeys	MinimapAttacks	MinimapRightClicks	NumberOfPACs	GapBetweenPACs	ActionLatency	ActionsInPAC	TotalMapExplored
1	52	5	27	10	3000	143.7180	3.515159e-03	2.196974e-04	7	1.098487e-04	3.923169e-04	0.004849036	32.6677	40.8673	4.7508	28
2	55	5	23	10	5000	129.2322	3.303812e-03	2.594617e-04	4	2.940566e-04	4.324362e-04	0.004307064	32.9194	42.3454	4.8434	22
3	56	4	30	10	200	69.9612	1.101091e-03	3.355705e-04	4	2.936242e-04	4.614094e-04	0.002925755	44.6475	75.3548	4.0430	22
4	57	3	19	20	400	107.6016	1.033542e-03	2.131015e-04	1	5.327537e-05	5.434088e-04	0.003782551	29.2203	53.7352	4.9155	19
5	58	3	32	10	500	122.8908	1.136014e-03	3.273259e-04	2	0.000000e+00	1.328558e-03	0.002368299	22.6885	62.0813	9.3740	15
6	60	2	27	6	70	44.4570	9.783903e-04	2.552323e-04	2	0.000000e+00	0.000000e+00	0.002424707	76.4405	98.7719	3.0965	16
7	61	1	21	8	240	46.9962	8.201141e-04	1.685166e-04	6	0.000000e+00	4.493776e-05	0.001988496	94.0227	90.5311	4.1017	15
8	72	7	17	42	10000	212.6022	9.039739e-03	6.762401e-04	6	1.163531e-03	1.253033e-03	0.004952464	24.6117	41.7671	6.6104	45
9	77	4	20	14	2708	117.4884	2.944275e-03	5.267713e-04	2	1.881326e-05	4.138917e-04	0.005399406	52.0140	46.4321	3.3746	29
10	81	4	18	24	800	155.9856	5.053906e-03	5.241090e-04	8	2.495757e-05	3.993212e-04	0.003568933	24.4632	52.1538	6.5664	27
11	83	3	16	16	6000	153.8010	1.676615e-03	3.185568e-04	4	0.000000e+00	8.215411e-04	0.003772383	23.4107	48.0711	7.0044	24
12	93	4	26	4	190	79.2948	3.785385e-04	2.551020e-04	3	1.645820e-05	1.645820e-04	0.003554970	39.6381	65.5000	4.2269	19
13	97	3	18	12	350	67.4754	4.225216e-04	1.690086e-04	1	2.414409e-05	1.448646e-04	0.002885219	42.4370	68.0502	4.3222	16
14	98	3	38	6	1000	119.4366	4.952043e-03	5.212677e-05	2	8.687795e-05	3.475118e-05	0.002727968	54.8718	79.2102	6.2293	21
15	100	5	16	30	5000	160.4754	4.253860e-03	4.317351e-04	2	7.745835e-04	4.063389e-04	0.004571312	36.2897	46.8889	5.4361	28
16	102	5	17	16	1500	81.7722	2.333484e-03	4.304486e-04	4	0.000000e+00	2.492071e-04	0.002899664	45.1654	64.7500	4.5312	15
17	105	4	28	8	2000	50.8374	6.641086e-04	2.213695e-04	1	1.844746e-05	1.475797e-04	0.002656434	45.7902	76.8889	3.5000	13
18	106	5	20	10	120	160.6464	3.430344e-03	6.336305e-04	7	0.000000e+00	9.468042e-05	0.005535163	28.3636	37.7947	4.7671	29
19	118	5	16	14	350	107.9118	6.701306e-03	7.061018e-04	5	1.332268e-05	4.662936e-04	0.003237410	67.0744	71.3251	4.3786	27
20	127	4	26	28	1100	114.7806	2.629613e-03	2.673894e-04	5	2.442810e-04	3.448672e-04	0.003649845	40.4427	59.9370	4.9961	16
21	132	5	21	10	800	115.1274	2.651149e-03	6.600370e-04	8	9.900554e-05	6.930388e-04	0.004147232	30.7660	49.4854	4.6790	30
22	138	6	21	6	500	133.7016	4.499795e-03	4.201053e-04	3	1.867135e-05	7.468539e-05	0.003874305	21.4686	50.5253	5.4892	29
23	139	5	18	20	800	99.5088	7.340200e-04	1.082980e-04	1	2.406623e-05	5.775895e-04	0.002695418	22.7265	68.7321	6.7054	23
24	140	5	26	10	500	83.9172	2.853513e-03	4.929658e-04	3	0.000000e+00	0.000000e+00	0.004607334	51.2082	52.0494	2.8663	31
25	141	4	17	14	500	216.6936	1.249465e-02	4.815925e-04	4	5.351027e-05	1.070205e-04	0.004334332	27.9255	45.1605	6.3395	18
26	142	4	23	20	800	129.8598	2.315078e-03	3.337689e-04	5	7.811612e-05	1.008408e-03	0.003472617	36.3934	57.5215	6.7587	36
27	144	6	18	70	2520	267.5586	2.781493e-02	7.081596e-04	10	0.000000e+00	1.278621e-04	0.005616099	34.6035	40.6025	4.1629	36
28	149	5	25	6	800	74.1174	8.750000e-04	2.053571e-04	6	1.785714e-05	4.464286e-05	0.003517857	46.4122	71.6173	3.7944	34
29	154	5	25	20	700	101.6796	1.406194e-03	4.271983e-04	5	0.000000e+00	1.245995e-04	0.003453186	26.7979	54.5979	5.3144	22

Làm sạch dữ liệu (Data cleaning):

Ta sẽ thực hiện trích ra một dữ liệu con đặt tên là **new_DF** chỉ bao gồm các biến mà ta quan tâm .

```
# 2.1)
# Create a new dataframe named "new_DF" which has particular expected fields
new_DF <- data.frame(data_fr$Age,data_fr$HoursPerWeek,data_fr$TotalHours,
    data_fr$APM,data_fr$SelectByHotkeys, data_fr$UniqueHotkeys,
    data_fr$MinimapAttacks,data_fr$MinimapRightClicks)
```

Sau đó ta chạy thêm lệnh sau để đặt lại tên các fields của **new_DF**:

```
# We rename for using easily
colnames(new_DF) <- c("Age","HoursPerWeek","TotalHours","APM",
                      "SelectByHotkeys","UniqueHotkeys","MinimapAttacks",
                      "MinimapRightClicks")
```

Kết quả ta thu được **new_DF** như sau:

	Age	HoursPerWeek	TotalHours	APM	SelectByHotkeys	UniqueHotkeys	MinimapAttacks	MinimapRightClicks
1	27	10	3000	143.7180	0.0035151591	7	1.098487e-04	3.923169e-04
2	23	10	5000	129.2322	0.0033038124	4	2.940566e-04	4.324362e-04
3	30	10	200	69.9612	0.0011010906	4	2.936242e-04	4.614094e-04
4	19	20	400	107.6016	0.0010335422	1	5.327537e-05	5.434088e-04
5	32	10	500	122.8908	0.0011360136	2	0.000000e+00	1.328558e-03
6	27	6	70	44.4570	0.0009783903	2	0.000000e+00	0.000000e+00
7	21	8	240	46.9962	0.0008201141	6	0.000000e+00	4.493776e-05
8	17	42	10000	212.6022	0.0090397391	6	1.163531e-03	1.253033e-03
9	20	14	2708	117.4884	0.0029442751	2	1.881326e-05	4.138917e-04
10	18	24	800	155.9856	0.0050539084	8	2.495757e-05	3.993212e-04
11	16	16	6000	153.8010	0.0016766146	4	0.000000e+00	8.215411e-04
12	26	4	190	79.2948	0.0003785385	3	1.645820e-05	1.645820e-04

Ta thấy có 3 cột có kiểu char

```
new_DF          3395 obs. of 8 variables
  data_fr.Age : chr "27" "23" "30" "19" ...
  data_fr.HoursPerWeek : chr "10" "10" "10" "20" ...
  data_fr.TotalHours : chr "3000" "5000" "200" "4...
  data_fr.APM : num 144 129 70 108 123 ...
  data_fr.SelectByHotkeys : num 0.00352 0.0033 0....
  data_fr.UniqueHotkeys : int 7 4 4 1 2 2 6 6 2 8...
  data_fr.MinimapAttacks : num 1.10e-04 2.94e-04 ...
  data_fr.MinimapRightClicks: num 0.000392 0.0004...
```

Ta chạy lệnh ép kiểu về num (các kí tự "?" sẽ được chuyển về N.A)

```
#2.1
#Check invalid value
new_DF$Age = as.numeric(new_DF$Age)
new_DF$HoursPerWeek = as.numeric(new_DF$HoursPerWeek)
new_DF$TotalHours = as.numeric(new_DF$TotalHours)
```

Ta chạy lệnh sau để xem thử có những điểm dữ liệu khuyết nào:

```
> # 2.2)
> # We inspect to check if it has NA value
> which(is.na(new_DF))
   [1]  3341  3342  3343  3344  3345  3346  3347  3348  3349  3350  3351  3352  3353  3354  3355  3356
  [17]  3357  3358  3359  3360  3361  3362  3363  3364  3365  3366  3367  3368  3369  3370  3371  3372
  [33]  3373  3374  3375  3376  3377  3378  3379  3380  3381  3382  3383  3384  3385  3386  3387  3388
  [49]  3389  3390  3391  3392  3393  3394  3395  5237  6736  6737  6738  6739  6740  6741  6742  6743
  [65]  6744  6745  6746  6747  6748  6749  6750  6751  6752  6753  6754  6755  6756  6757  6758  6759
  [81]  6760  6761  6762  6763  6764  6765  6766  6767  6768  6769  6770  6771  6772  6773  6774  6775
  [97]  6776  6777  6778  6779  6780  6781  6782  6783  6784  6785  6786  6787  6788  6789  6790  7149
 [113]  8632 10131 10132 10133 10134 10135 10136 10137 10138 10139 10140 10141 10142 10143 10144 10145
 [129] 10146 10147 10148 10149 10150 10151 10152 10153 10154 10155 10156 10157 10158 10159 10160 10161
 [145] 10162 10163 10164 10165 10166 10167 10168 10169 10170 10171 10172 10173 10174 10175 10176 10177
 [161] 10178 10179 10180 10181 10182 10183 10184 10185
> colSums(is.na(new_DF))
               Age       HoursPerWeek         TotalHours                APM    SelectByHotkeys
                55                 56                 57                  0                  0
     UniqueHotkeys     MinimapAttacks MinimapRightClicks
                 0                  0                  0
```

Như vậy, ta thấy một số lượng nhỏ dữ liệu bị khuyết.

Ở đây ta có một số cách xử lí dữ liệu bị khuyết như là thay thế bằng một giá trị trung bình. Tuy nhiên, số lượng dữ liệu bị khuyết rất nhỏ so với tổng dữ liệu nên nhóm quyết định xóa bỏ những hàng chứa các dữ liệu khuyết này.

Ta dùng lệnh sau:

```
new_DF = new_DF[complete.cases(new_DF),]
```

Làm rõ dữ liệu (Data visualization) :

Chuyển đổi các biến **SelectByHotkeys**, **MinimapAttacks**, **MinimapRightClicks** lần lượt thành **SelectByHotkeys * 10^5**, **MinimapAttacks* 10^5** , **MinimapRightClicks*10^5**. Từ đây mọi sự tính toán với các biến trên được hiểu là đã qua đổi biến dạng mũ.

```
# Change type for some fields
new_DF$SelectByHotkeys = new_DF$SelectByHotkeys*(10^5)
new_DF$MinimapAttacks = new_DF$MinimapAttacks*(10^5)
new_DF$MinimapRightClicks = new_DF$MinimapRightClicks * (10^5)
```

Dữ liệu lúc này trở thành:

	Age	HoursPerWeek	TotalHours	APM	SelectByHotkeys	UniqueHotkeys	MinimapAttacks	MinimapRightClicks
1	27	10	3000	143.7180	351.515910	7	10.9848700	39.231690
2	23	10	5000	129.2322	330.381240	4	29.4056600	43.243620
3	30	10	200	69.9612	110.109060	4	29.3624200	46.140940
4	19	20	400	107.6016	103.354220	1	5.3275370	54.340880
5	32	10	500	122.8908	113.601360	2	0.0000000	132.855820
6	27	6	70	44.4570	97.839030	2	0.0000000	0.000000
7	21	8	240	46.9962	82.011410	6	0.0000000	4.493776
8	17	42	10000	212.6022	903.973910	6	116.3530800	125.303310
9	20	14	2708	117.4684	294.427510	2	1.8813260	41.389170
10	18	24	800	155.9856	505.390840	8	2.4957572	39.932120
11	16	16	6000	153.8010	167.661460	4	0.0000000	82.154110
12	26	4	190	79.2948	37.853850	3	1.6458196	16.458200
13	18	12	350	67.4754	42.252160	1	2.4144092	14.486460
14	38	6	1000	119.4366	495.204340	2	8.6877954	3.475118

Ta chạy lệnh **summary()** cho **new_DF**, kết quả thu về lưu vào **summary_DF**:

```
# Creat summary_DF - from using function on new DF
summary_DF = as.data.frame(apply(subset(new_DF,
        select=-c(Age,HoursPerWeek,TotalHours,UniqueHotkeys)),2,summary))
```

Giá trị của **summary_DF** lúc này cho ta:

	APM	SelectByHotkeys	MinimapAttacks	MinimapRightClicks
Min.	22.0596	0.0000	0.000000	0.00000
1st Qu.	79.2315	124.4804	0.000000	13.88204
Median	107.0703	244.5127	3.864138	27.84002
Mean	114.5758	402.3309	9.378006	38.02441
3rd Qu.	140.1561	494.4798	11.343920	50.75841
Max.	389.8314	4308.8364	301.934650	368.76680

Thông tin từ lệnh **summary()** đã cho ta đủ thông tin về min, max, median, mean, 1st & 3rd quartile. Ở đây ta cần tính thêm độ lệch chuẩn và không cần sử dụng thông tin 1st & 3rd quartile nên ta thay giá trị 1st quartile bằng thông tin độ lệnh chuẩn và xóa đi hàng 3rd quartile. Ta dùng lệnh như sau:

```
# Delete 3rd Qu.row
summary_DF = summary_DF[-c(5),]
# Rename rows in summary_DF
rownames(summary_DF) = c("Min","Sd","Median","Mean","Max")
#Replace value from 1st Qu.row to Standard Deviation
summary_DF$APM[2] = sd(new_DF$APM)
summary_DF$SelectByHotkeys[2] = sd(new_DF$SelectByHotkeys)
summary_DF$MinimapAttacks[2] = sd(new_DF$MinimapAttacks)
summary_DF$MinimapRightClicks[2] = sd(new_DF$MinimapRightClicks)
```

Ta chỉ summary các biến liên tục

Kết quả cuối cùng ta thu được **summary_DF** chứa các thông tin cần như sau:

	APM	SelectByHotkeys	MinimapAttacks	MinimapRightClicks
Min	22.05960	0.0000	0.000000	0.00000
Sd	48.11191	472.6417	15.898127	35.94914
Median	107.07030	244.5127	3.864138	27.84002
Mean	114.57576	402.3309	9.378006	38.02441
Max	389.83140	4308.8364	301.934650	368.76680

Bây giờ ta sẽ tạo bảng cho các phân loại: ta dùng lệnh

```
#3b
#Distribute some disjointed variables
freq_Age = table(new_DF$Age)
View(freq_Age)
freq_HoursPerWeek = table(new_DF$HoursPerWeek)
View(freq_HoursPerWeek)
freq_TotalHours = table(new_DF$TotalHours)
View(freq_TotalHours)
freq_UniqueHotkeys = table(new_DF$UniqueHotkeys)
```

- Phân loại theo độ tuổi:

	Var1	Freq
1	16	256
2	17	247
3	18	324
4	19	313
5	20	357
6	21	344
7	22	314
8	23	259
9	24	225
10	25	168
11	26	136
12	27	111
13	28	73
14	29	52
15	30	32
16	31	29
17	32	21
18	33	15
19	34	15
20	35	17
21	36	8
22	37	5
23	38	5
24	39	3
25	40	4
26	41	3
27	43	1
28	44	1

- Phân loại theo HoursPerWeek:

	Var1	Freq
1	0	1
2	2	108
3	4	219
4	6	323
5	8	390
6	10	411
7	12	331
8	14	181
9	16	223
10	18	24
11	20	334
12	24	233
13	28	280
14	30	54

Phân loại theo TotalHours:

	Var1	Freq
1	3	1
2	7	1
3	10	5
4	12	4
5	16	1
6	20	10
7	21	1
8	24	2
9	25	4
10	26	1
11	30	17
12	35	1
13	36	1
14	40	10

Phân loại theo UniqueHotkeys:

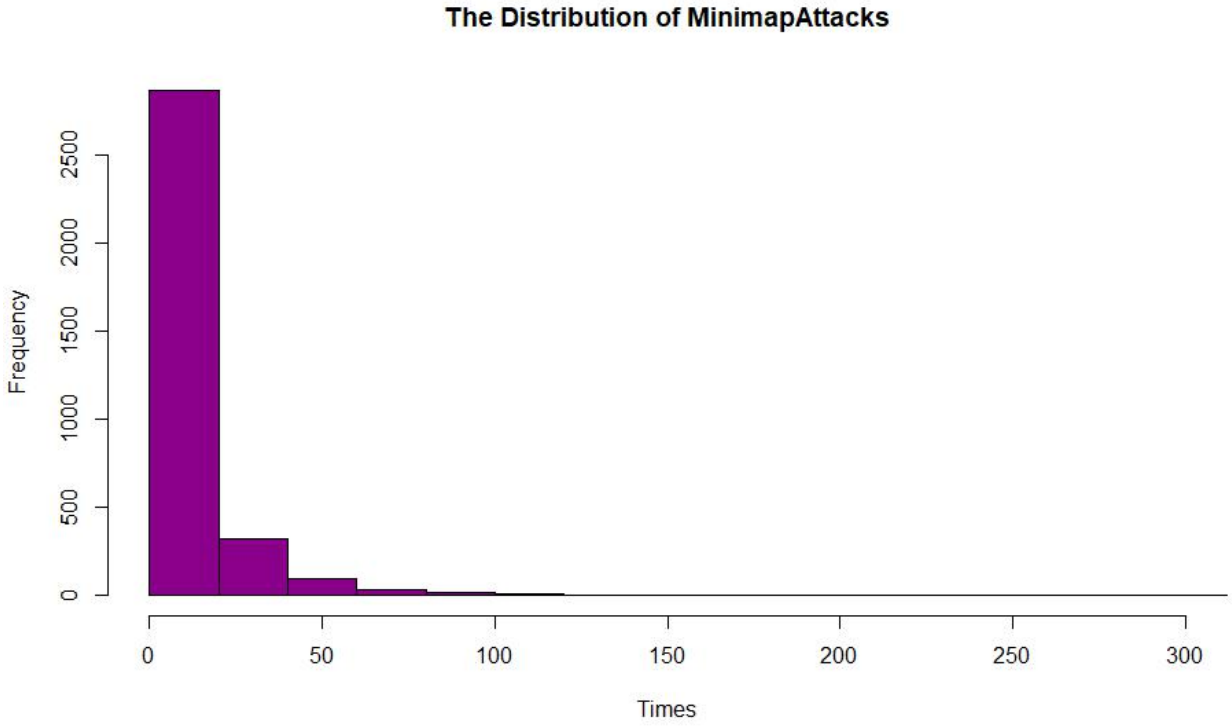
	Var1	Freq
1	0	189
2	1	226
3	2	345
4	3	438
5	4	572
6	5	612
7	6	391
8	7	272
9	8	152
10	9	46
11	10	95

Với Var1 là các phần tử dữ liệu xuất hiện trong các biến rời rạc. Freq là số lần xuất hiện tương ứng với từng Var1.

Ta dùng lệnh hist() để vẽ đồ thị phân phối của biến **MinimapAttacks** như sau:

```r
# Hist function
hist(new_DF$MinimapAttacks,main="The Distribution of MinimapAttacks",
    xlab="Times",ylab="Frequency",
    xlim=c(0,300),breaks=15,col="darkmagenta",freq=TRUE)
```

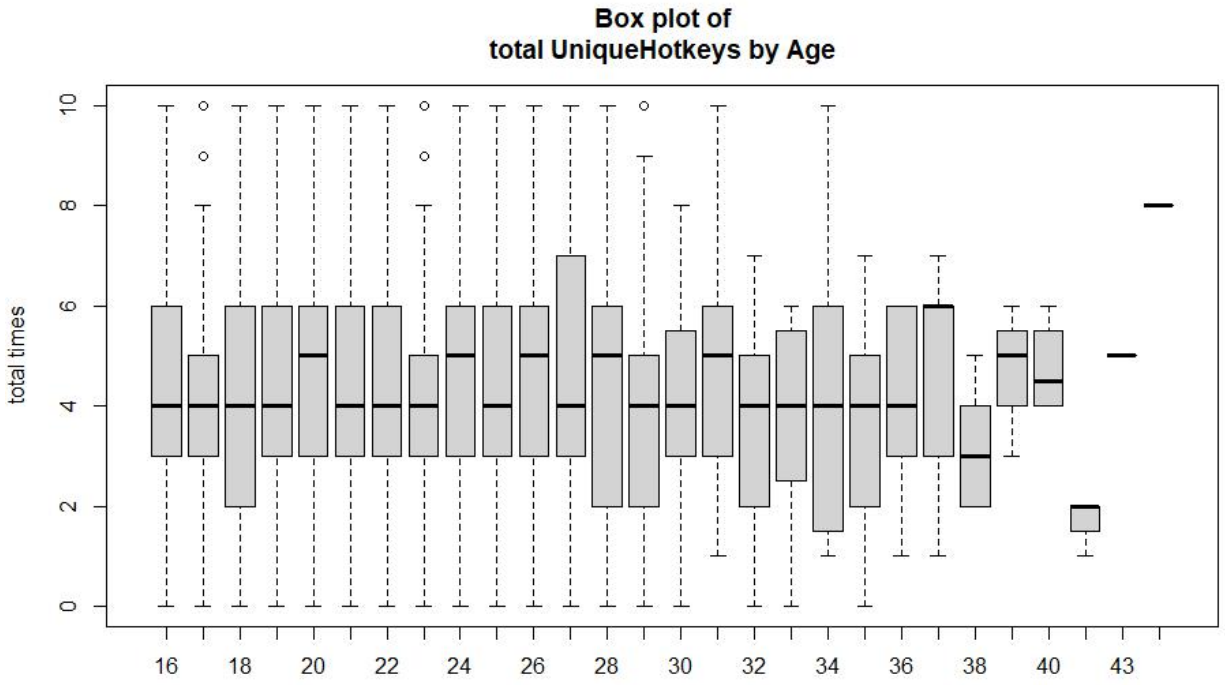
Kết quả ta thu được đồ thị như sau:

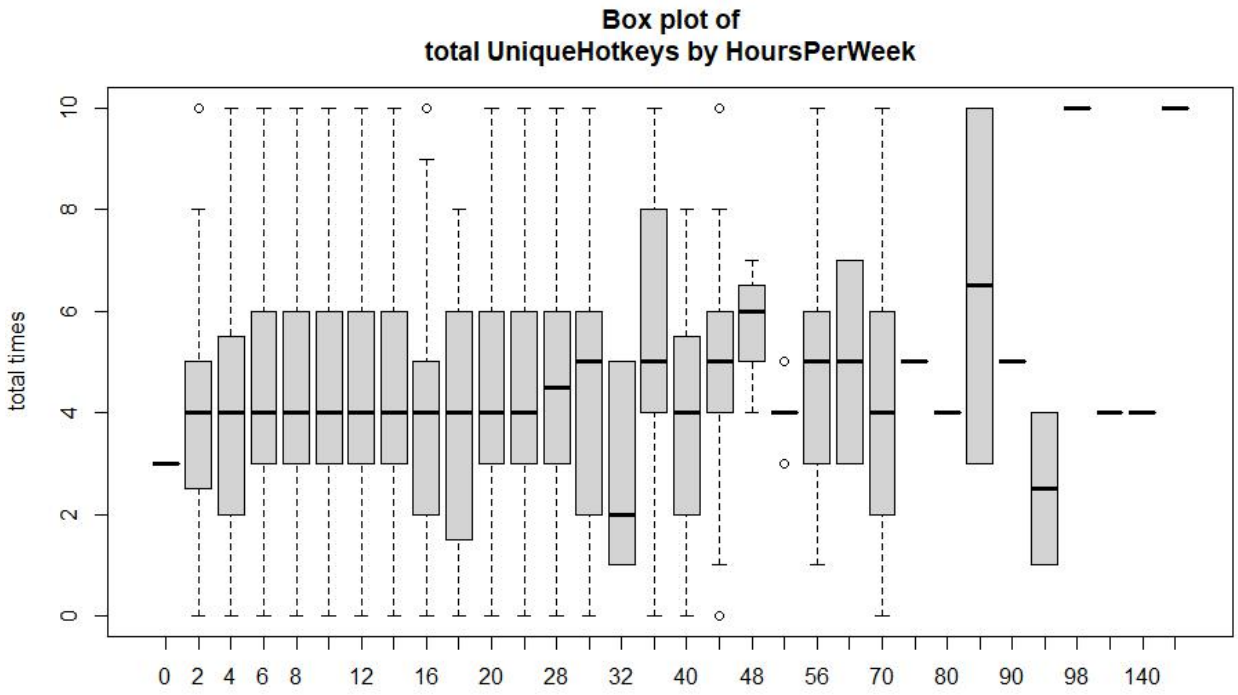


Dùng hàm **boxplot()** ta vẽ phân phối biến **UniqueHotkeys** cho từng nhóm phân loại của biến **Age**, **HoursPerWeek** như sau:

```r
# According to Age
boxplot(new_DF$UniqueHotkeys ~ new_DF$Age, main = "Box plot of
        total UniqueHotkeys by Age",ylab="total times",xlab="")
# According to HoursPerWeek
boxplot(new_DF$UniqueHotkeys ~ new_DF$HoursPerWeek, main = "Box plot of
        total UniqueHotkeys by HoursPerWeek",ylab="total times",xlab="")
```

Kết quả lần lượt ta thu được 2 đồ thị sau:

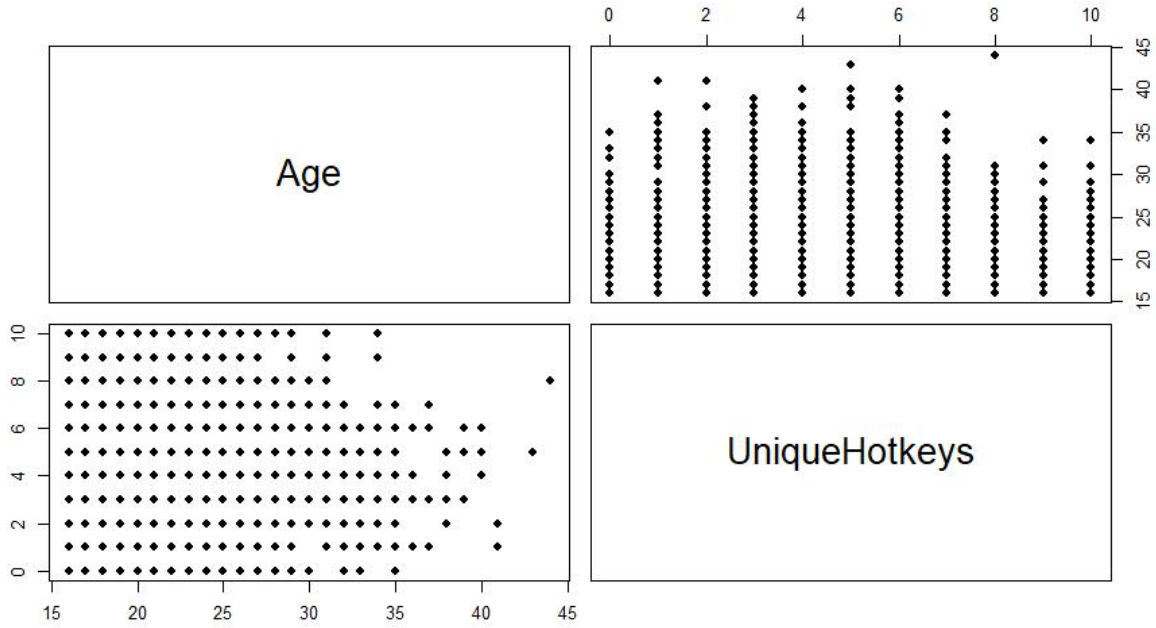


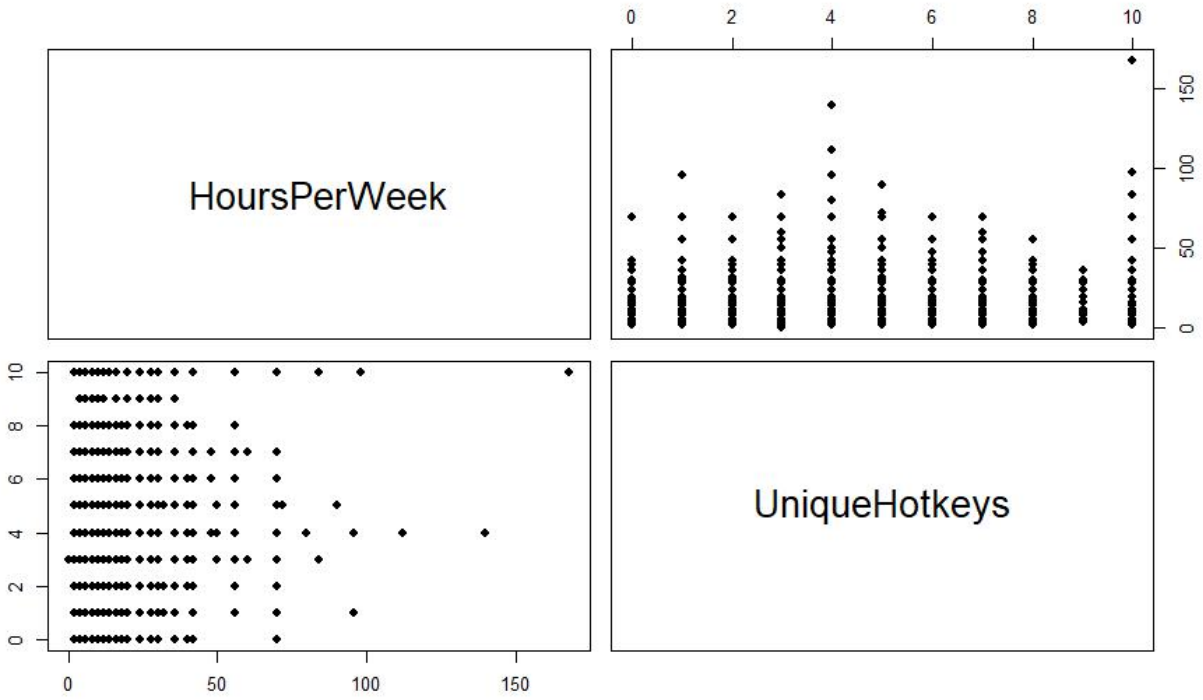


Dùng lệnh **pairs()** ta vẽ các phân phối của biến **UniqueHotkeys** lần lượt theo các biến **Age**, **HoursPerWeek**, **MinimapRightClicks** như sau:

```r
# UniqueHotkeys and Age
pairs(subset(new_DF,select=c(1,6)),pch=16)
# UniqueHotkeys and HoursPerWeek
pairs(subset(new_DF,select=c(2,6)),pch=16)
# UniqueHotkeys and MinimapRightClicks
pairs(subset(new_DF,select=c(8,6)),pch=16)
```

Kết quả thu được 3 đồ thị dưới :




HoursPerWeek
UniqueHotkeys
0
2
4
6
8
10
0
50
100
150


MinimapRightClicks
UniqueHotkeys
0
2
4
6
8
10
0
100
200
300

Xây dựng các mô hình hồi quy tuyến tính (Fitting linear regression models):

Chúng ta muốn khám phá rằng có những nhân tố nào và tác động như thế nào đến Số hành động tấn công trên bản đồ nhỏ trên mỗi dấu thời gian (**MinimapAttacks**).

a. Xét mô hình hồi quy tuyến tính với **MinimapAttacks** là biến phụ thuộc, các biến còn lại là biến độc lập. Ta dùng lệnh **lm()** như sau:

```
# lm function
modelMiniAtt = lm(MinimapAttacks ~ Age + HoursPerWeek +
TotalHours + APM + SelectByHotkeys + UniqueHotkeys + MinimapRightClicks,new_DF)
```

Chạy lệnh **summary()** để thu được các thông số từ **modelMiniAtt** như sau:

```
> summary(modelMiniAtt)

Call:
lm(formula = MinimapAttacks ~ Age + HoursPerWeek + TotalHours +
    APM + SelectByHotkeys + UniqueHotkeys + MinimapRightClicks,
    data = new_DF)

Residuals:
    Min      1Q  Median      3Q     Max
-29.137  -7.476  -3.659   2.340 288.182

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -1.101e+01  1.831e+00  -6.014 2.01e-09 ***
Age              3.385e-01  6.508e-02   5.201 2.10e-07 ***
HoursPerWeek     6.720e-02  2.288e-02   2.937  0.00334 **
TotalHours      -9.256e-06  1.522e-05  -0.608  0.54305
APM              7.172e-02  1.063e-02   6.748 1.76e-11 ***
SelectByHotkeys -2.674e-03  9.957e-04  -2.686  0.00727 **
UniqueHotkeys    5.148e-01  1.200e-01   4.289 1.85e-05 ***
MinimapRightClicks  6.929e-02  7.962e-03   8.703  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15.17 on 3330 degrees of freedom
Multiple R-squared:  0.09161,   Adjusted R-squared:  0.0897
F-statistic: 47.97 on 7 and 3330 DF,  p-value: < 2.2e-16
```

Phương trình hồi quy của mô hình **modelMiniAtt**:

MinimapAttacks = -1.101*10 + 3.385 $*10^{-1}$***Age** – 6.72 $*10^{-2}$***HoursPerWeek** – 9.256 $*10^{-6}$ ***TotalHours** + 7.172 * 10^{-2} ***APM** - 2.674 $*10^{-3}$ * **SelectByHotKeys** + 5.148 $*10^{-1}$ ***UniqueHotkeys** + 6.929 * 10^{-2} * **MinimapRightClicks**

b. Quan sát giá trị P-value, chính là giá trị Pr (>| t |) trong phần Coefficients.

Với mức tin cậy 5%: Ta sẽ loại bỏ biến **TotalHours** vì P-value > 0.05.

c. Xét mô hình ban đầu (M1) và mô hình tuyến tính M2 là loại bỏ biến **TotalHours**

từ M1. Dùng lệnh **anova()** để đề xuất mô hình hồi quy tuyến tính hợp lý hơn.

Mô hình M1 đã được thực hiện ở trên.

Mô hình M2:

```
# M2: remove TotalHours
modelMiniAtt2 = lm(MinimapAttacks ~ Age + HoursPerWeek + APM +
  SelectByHotkeys + UniqueHotkeys + MinimapRightClicks,new_DF)
```

```
> summary(modelMiniAtt2)

Call:
lm(formula = MinimapAttacks ~ Age + HoursPerWeek + APM + SelectByHotkeys +
    UniqueHotkeys + MinimapRightClicks, data = new_DF)

Residuals:
    Min      1Q  Median      3Q     Max
-29.149  -7.480  -3.665   2.340 288.182

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)       -11.004864   1.830445  -6.012 2.03e-09 ***
Age                 0.338520   0.065070   5.202 2.09e-07 ***
HoursPerWeek        0.067119   0.022877   2.934  0.00337 **
APM                 0.071638   0.010627   6.741 1.84e-11 ***
SelectByHotkeys    -0.002696   0.000995  -2.710  0.00677 **
UniqueHotkeys       0.515963   0.120016   4.299 1.76e-05 ***
MinimapRightClicks  0.069314   0.007961   8.707  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15.17 on 3331 degrees of freedom
Multiple R-squared:  0.09151,   Adjusted R-squared:  0.08987
F-statistic: 55.92 on 6 and 3331 DF,  p-value: < 2.2e-16
```

Phương trình hồi quy của mô hình modelUnique2:
MinimapAttacks = -11.004864 + 0.338520***Age** +0.067119 ***HoursPerWeek** + 0.071638* **APM** – 0.002696***SelectByHotKeys** + 0.515963***UniqueHotkeys** + 0.069314* **MinimapRightClicks**

Sử dụng lệnh **anova()** khi xét mô hình của biến **MinimapAttacks** theo biến **TotalHours**:

Giả thiết:

H_0: MinimapAttacks không phụ thuộc vào sự biến thiên của TotalHours

H_1: MinimapAttacks phụ thuộc vào sự biến thiên của TotalHours

```
> lmM2 <- lm(MinimapAttacks ~ TotalHours,new_DF)
> anovaD <- anova(lmM2)
> print(anovaD)
Analysis of Variance Table

Response: MinimapAttacks
              Df Sum Sq Mean Sq F value Pr(>F)
TotalHours     1      1   0.637  0.0025   0.96
Residuals   3336 843428 252.826
```

P-value = 0.96 > 0.05. Vậy thừa nhận H_0 , bác bỏ sự phụ thuộc vào sự biến thiên của TotalHours , vậy rõ ràng mô hình M2 là mô hình chuẩn xác hơn.

d. Chọn mô hình hợp lí hơn từ câu (c), hãy suy luận sự tác động của các biến lên **MinimapAttacks**.

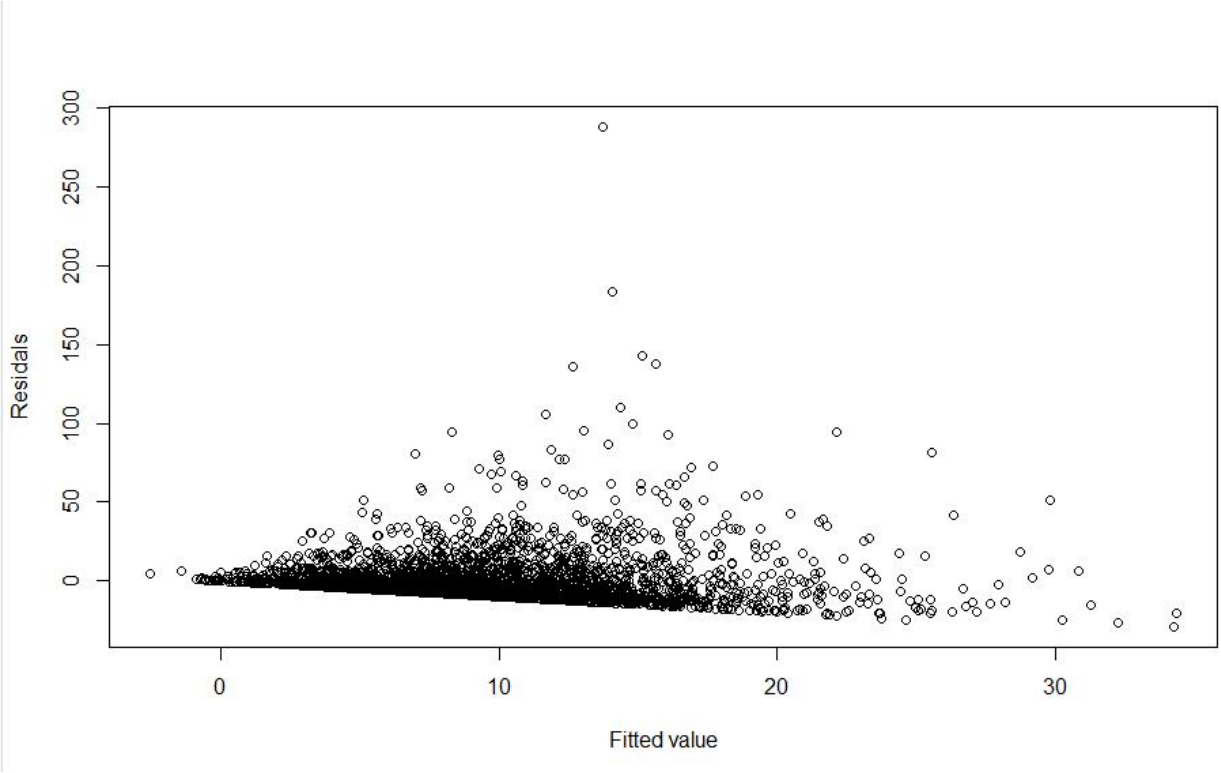
Các biến của M2 tác động vào giá trị **MinimapAttacks** theo quan hệ tuyến tính xác định bằng hệ số **estimated coefficient**.

Các giá trị **Age**,**HoursPerWeek** ,**APM**,**UniqueHotkeys**,**MinimapRightClicks** tác động vào giá trị **MinimapAttacks** theo tỷ lệ thuận. Trong đó giá trị **UniqueHotkeys** có tác động lớn nhất và giá trị **HoursPerWeek** có tác động nhỏ nhất.

Giá trị **SelectByHotkeys** tác động vào giá trị **MinimapAttacks** theo tỷ lệ nghịch. Vì **estimated coefficient** có giá trị âm.

e. Từ mô hình hồi quy chọn ở câu (c) hãy dùng lệnh **plot()** để vẽ‡đồ thị biểu thị sai số hồi quy **(residual)** và giá trị dự báo **(fitted values)**. Nêu ý nghĩa và nhận xét đồ thị.

```
resi = resid(modelMiniAtt2)
plot(fitted(modelMiniAtt2),resi,xlab="Fitted value",ylab="Residals")
```



Nhận xét: **MinimapAttacks** phần lớn dao động trong khoảng từ 0 đến 20 / (10^5), sai số dự báo dao động gần như nhỏ hơn 0, sự chênh lệch do dự báo độc lập với giá trị dự báo được bởi mô hình.

Dự báo (Prediction):

a. Giả sử đây là game dành cho người lớn trên 18 tuổi, chúng ta cần thống kê số người dùng có độ tuổi dưới 18 tuổi đã tham gia để đưa ra các biện pháp xử lý. Thống kê tỷ lệ đạt (Age >= 18) hoặc không đạt (Age < 18) của người dùng. Tính tỷ lệ đạt/không đạt.

Trong **new_DF** ta tạo thêm cột mới **"evaluate"** và thêm các giá trị 0 hoặc 1 vào cột này tùy vào giá trị tuổi **Age** (lớn hơn hoặc bằng 18 tuổi tương ứng với giá trị 1, nhỏ hơn 18 tuổi tương ứng với giá trị 0) .

```
#5a:
new_DF = transform(new_DF, evaluate = ifelse(Age>=18,1,0))
#
num_pass= sum(new_DF$Age[]>=18)
#so nguoi dung tren 18 tuoi :
num_pass
num_fail=sum(new_DF$Age[]<18)
#so nguoi dung duoi 18 tuoi:
num_fail
pass_fail_rate = num_pass/num_fail
#ty le:
pass_fail_rate
```

```
> #5a:
> new_DF = transform(new_DF, evaluate = ifelse(Age>=18,1,0))
> #
> num_pass= sum(new_DF$Age[]>=18)
> #so nguoi dung tren 18 tuoi :
> num_pass
[1] 2835
> num_fail=sum(new_DF$Age[]<18)
> #so nguoi dung duoi 18 tuoi:
> num_fail
[1] 503
> pass_fail_rate = num_pass/num_fail
> #ty le:
> pass_fail_rate
[1] 5.636183
```

b. Xét mô hình tuyến tính cùng bao gồm biến **Age** là biến phụ thuộc, còn lại là biến độc lập. Sử dụng lệnh **predict()** để dự báo tuổi của người dùng tại 2 thuộc tính như sau:

x1: APM = mean(APM), SelectByHotkeys = mean(SelectByHotkeys), UniqueHotkeys = mean(UniqueHotkeys), HoursPerWeek= 10, TotalHours = 730.
x2: APM = max(APM), SelectByHotkeys = max(SelectByHotkeys), UniqueHotkeys= max(UniqueHotkeys), HoursPerWeek= 10, TotalHours = 730.
So sánh khoảng tin cậy cho 2 giá trị dự báo này.

Tạo 2 data frame x1 và x2 :

```
#5b:
#
x1= data.frame(APM = mean(new_DF[,'APM']), SelectByHotkeys = mean(new_DF[,'SelectByHotkeys']), UniqueHotkeys
= mean(new_DF[,'UniqueHotkeys']), HoursPerWeek= 10, TotalHours = 730)
x2= data.frame(APM = max(new_DF[,'APM']), SelectByHotkeys = max(new_DF[,'SelectByHotkeys']), UniqueHotkeys
= max(new_DF[,'UniqueHotkeys']), HoursPerWeek= 10, TotalHours = 730)
```

Sử dụng lệnh **predict()** với mô hình hồi quy tuyến tính trên với độ tin cậy 95%.

```
# xay dung mo hinh tuyen tinh boi
lmAge = lm(Age ~ APM + SelectByHotkeys + UniqueHotkeys + HoursPerWeek + TotalHours,new_DF)

predict(lmAge, x1, interval = "confidence")
predict(lmAge, x2, interval = "confidence")
```

Kết quả thu được:

```
> #5b:
> # tao 2 data frame x1 va x2
> x1= data.frame(APM = mean(new_DF[,'APM']), SelectByHotkeys = mean(new_DF[,'SelectByHotkeys']), UniqueHotkeys
+ = mean(new_DF[,'UniqueHotkeys']), HoursPerWeek= 10, TotalHours = 730)
> x2= data.frame(APM = max(new_DF[,'APM']), SelectByHotkeys = max(new_DF[,'SelectByHotkeys']), UniqueHotkeys
+ = max(new_DF[,'UniqueHotkeys']), HoursPerWeek= 10, TotalHours = 730)
>
>
> # xay dung mo hinh tuyen tinh boi
> lmAge = lm(Age ~ APM + SelectByHotkeys + UniqueHotkeys + HoursPerWeek + TotalHours,new_DF)
>
> predict(lmAge, x1, interval = "confidence")
       fit      lwr      upr
1 21.94201 21.78787 22.09614
> predict(lmAge, x2, interval = "confidence")
       fit      lwr      upr
1 19.77023 18.56417 20.97629
```

Như vậy, khoảng tin cậy cho giá trị **Age** (tuổi), tính bằng upr – lwr, dự đoán ở x2 lớn hơn x1.

Tài liệu tham khảo

1. Slide bài giảng môn Xác suất thống kê – Cô Nguyễn Kiều Dung.
2. Giáo trình Xác suất và thống kê; Bài tập Xác suất và thống kê – Tác giả Nguyễn Đình Huy, Đậu Thế Cấp – NXB Đại học quốc gia TPHCM – 2013.
3. Tài liệu BTL ĐHBK Chương 2 Áp dụng MS-Excel trong thống kê suy lí – Tác giả Nguyễn Đình Huy.
4. Applied Statistics and Probability for Engineers – Douglas C. Montgomery, George C. Runger – Hoboken, NJ: Wiley – 2017.