Ho Chi Minh City University of Technology

Report

Probability and Statistics

Lecture: Nguyễn Tiến Dũng

Class : CC01

Group 5

Members :

Lê Đức Cầm	1952588
Lý Kim Phong	1952916
Huỳnh Phước Thiện	1952463
Trần Nguyễn Anh Khoa	1952790

Table of Contents:

Project 1 - Topic 6

<u>Exercise 1:</u>

Base theory

Independent variable (x)

Dependent variable (y)

Relationship between these two variables (curve, straight line):

If they move the same direction -> positive relationship

If they move the conflict direction -> negative relationship

->Least-squares regression method: The line minimizes the sum of the squares of the vertical deviations from each data point to the line. Because the deviations are first squared, then summed, there are no cancellations between positive and negative values.

Linear regression equation formula: y=b+a*x

In which:

y: dependent variable

x: explanatory variable

b: y-intercept

a: estimated effect of x on y

Implementation: To Perform the linear regression analysis:

```
> data1.lm <- lm(Y ~ X, data = data1.xlsx)
> summary(data1.lm)
```

```
Call:
lm(formula = Y ~ X, data = data1)

Residuals:
     Min       1Q   Median       3Q      Max 
-226.831  -30.495    0.132   44.307  188.746 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  40.0500    46.5321   0.861  0.40212   
X             0.9241     0.2605   3.547  0.00268 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 83.95 on 16 degrees of freedom
Multiple R-squared:  0.4402,	Adjusted R-squared:  0.4052 
F-statistic: 12.58 on 1 and 16 DF,  p-value: 0.002682
```

To install the packages you need for the analysis, run this code (you only need to do this once):

```
> install.packages("ggplot2")

> install.packages("dplyr")

> install.packages("broom")

> install.packages("ggpubr")
```

Next, load the packages into your R environment by running this code (you need to do this every time you restart R):

```
> library(ggplot2)

> library(dplyr)

> library(broom)

> library(ggpubr)
```

Load the data into R (in this exercise, it’s called “data1.xlsx”)

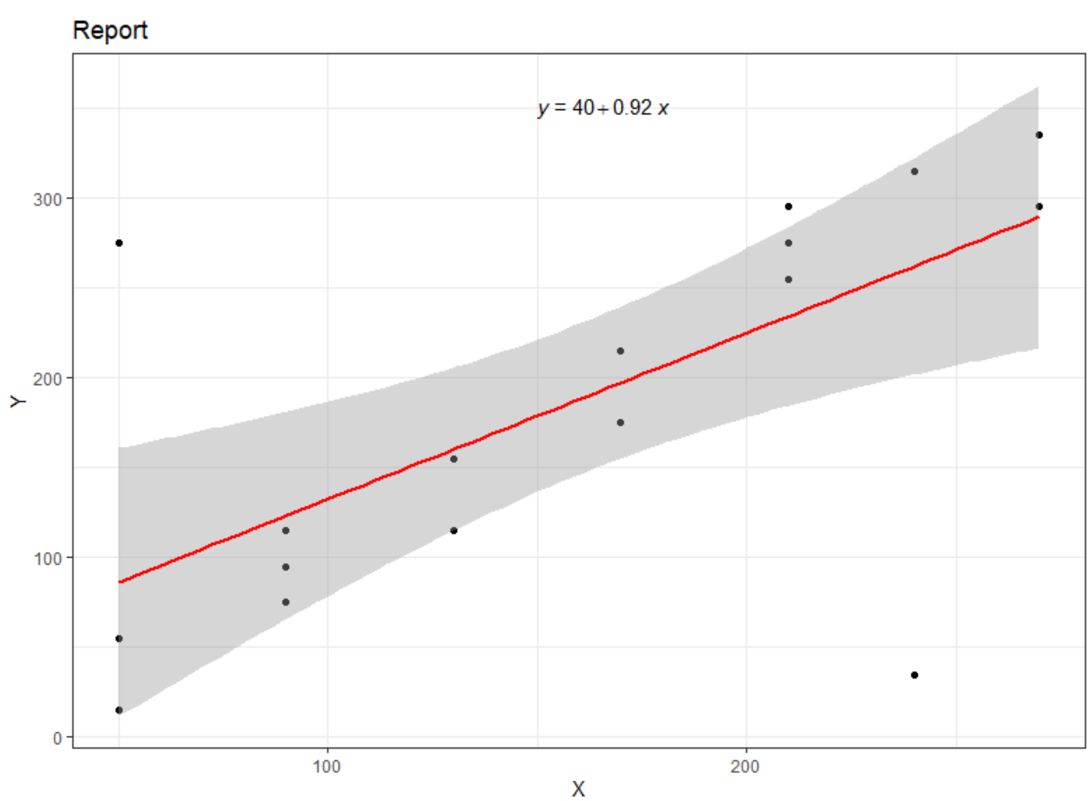
Plot the data points on a graph, add the linear regression line to the plotted data, add the equation for the regression line, make the graph ready for publication:

```
> data1.graph <- ggplot(data1, aes(x=X, y=Y)) + geom_point()

> data1.graph <- data1.graph + geom_smooth(formula = y ~ x, method="lm", col="red")

> data1.graph <- data1.graph + stat_regline_equation(label.x = 150, label.y = 350)

> data1.graph + theme_bw() + labs(title = "Report", x = "X", y = "Y")
```

Add the regression line using geom_smooth() and typing in lm as your method for creating the line. This will add the line of the linear regression as well as the standard error of the estimate as a light grey stripe surrounding the line.

We can add some style parameters using theme_bw() and making custom labels using labs()

Output:



Exercise 2:

Step 1: Set up hypothesis

- H_0 : μ_M - μ_F = 0 (There is no change in the duration between Male and Female)
- H_1 : μ_M - μ_F ≠ 0 (There is a change in the duration between Male and Female)

Step 2: Calculate P-value

Install this package: install.packages("readxl")

Then load the package in R Environment: library(readxl)

Load data into RStudio, which is an Excel file named "data.xlsx" using the command: read_excel("The link to the file")

This is the data:

Gender	Sound	Light	Pulse
Male	10.0	6.0	9.1
	7.2	3.7	5.8
	6.8	5.1	6.0
	6.0	4.0	4.0
	5.0	3.2	5.1
Female	10.5	6.6	7.3
	8.8	4.9	6.1
	9.2	2.5	5.2
	8.1	4.2	2.5
	13.4	1.8	3.9

- We will calculate the T-value of each signal column of both genders.
- This is the formula of T-value:

$$t = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}}$$

$$t = \frac{(\bar{x}_1 - \bar{x}_2)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

- First, we input data from each signal column from each gender and store the Male's signal variables as m1 and Female's as m2. Use the "←" syntax to store the variables in m1 and m2.

  Example: *m1 ← data$...* (choose the signals: *Sound, Light, Pulse*)

  m2 ← data$... (choose the signals: *Sound, Light, Pulse*)

- Second, calculate the standard deviation of each signal column from both genders. Use the syntax *sd(data$...)* to calculate then use the "←" syntax to store the result in sd1 and sd2.

Example: *sd1 ← sd(data$...)*(choose the signals: *Sound, Light, Pulse*) - Male

sd2 ← sd(data$...)(choose the signals: *Sound, Light, Pulse*) - Female

- There are 15 men and 15 women participating in this experiment. Each 5-men and each 5-women will be tested in 3 different signals. Thus, the number of people on each signal test is 5 for men and 5 for women. Each sample size is 5. Store each in num1 and num2.

  Example: *num1 = 5*

  num2 = 5

- Third, calculate the square root of the sum of square of sd1 over num1 and square of sd2 over num2 then use the "←" syntax to store the result in se.

  Example: *se ← sqrt(sd1*sd1/num1 + sd2*sd2/num2)*

- Finally, calculate the T-value by taking (m1 - m2) divided by se.

  t = (m1 - m2)/se

- After calculating the T-value, we use the function *pt(t, df = pmin(num1, num2) - 1)* to calculate the P-values of each man and each woman's test. Then, use function *mean()* to calculate the average P-value of each signal.

  Example: *averageP ← pt(t, df = pmin(num1, num2) - 1)*

  mean(averageP) = ...

Signals:

- ***Sound:***
  + Follow those steps above then we get the average P-value = 0.1284261 which is bigger than the significance level => H_a: $\mu_M - \mu_F \neq 0$
  + CODE:

```
Console   Terminal ×   Jobs ×
~/
> library(readxl)
> data <- read_excel("C:/Users/ASUS/Desktop/data.xlsx")
New names:
* Gender -> Gender...1
* Sound -> Sound...2
* Light -> Light...3
* Pulse -> Pulse...4
* Gender -> Gender...5
* ...
> View(data)
> m1 <- data$Sound...2
> m2 <- data$Sound...6
> sd1 <- sd(data$Sound...2)
> sd2 <- sd(data$Sound...6)
> num1 = 5
> num2 = 5
> se = sqrt(sd1*sd1/num1 + sd2*sd2/num2)
> t = (m1-m2)/se
> pt(t, df = pmin(num1,num2) - 1)
[1] 0.355511381 0.135939087 0.064377262 0.085000842 0.001301977
> averageP <- pt(t, df = pmin(num1,num2) - 1)
> mean(averageP)
[1] 0.1284261
> |
```

- ***Light:***
  + Follow those steps above then we get the average P-value = 0.543277 which is bigger than the significance level => H_a: $\mu_M - \mu_F \neq 0$
  + CODE:

```
> library(readxl)
> data <- read_excel("C:/Users/ASUS/Desktop/data.xlsx")
New names:
* Gender -> Gender...1
* Sound -> Sound...2
* Light -> Light...3
* Pulse -> Pulse...4
* Gender -> Gender...5
* ...
> View(data)
> m1 <- data$Light...3
> m2 <- data$Light...7
> sd1 <- sd(data$Light...3)
> sd2 <- sd(data$Light...7)
> num1 = 5
> num2 = 5
> se <- sqrt(sd1*sd1/num1 + sd2*sd2/num2)
> t <- (m1-m2)/se
> pt(t,df=pmin(num1,num2)-1)
[1] 0.2896912 0.1473384 0.9703045 0.4253241 0.8837267
> averageP <- pt(t,df=pmin(num1,num2)-1)
> mean(averageP)
[1] 0.543277
> 
```

- ***PULSE:***
  + Follow those steps above then we get the average P-value = 0.7420474 which is bigger than the significance level => H_a: $\mu_M - \mu_F \neq 0$
  + CODE:

```
Console   Terminal ×   Jobs ×
~/
> library(readxl)
> data <- read_excel("C:/Users/ASUS/Desktop/data.xlsx")
New names:
* Gender -> Gender...1
* Sound -> Sound...2
* Light -> Light...3
* Pulse -> Pulse...4
* Gender -> Gender...5
* ...
> View(data)
> m1 <- data$Pulse...4
> m2 <- data$Pulse...8
> sd1 <- sd(data$Pulse...4)
> sd2 <- sd(data$Pulse...8)
> num1 = 5
> num2 = 5
> se <- sqrt(sd1*sd1/num1 + sd2*sd2/num2)
> t <- (m1-m2)/se
> pt(t,df=pmin(num1,num2)-1)
[1] 0.8970919 0.4069140 0.7304075 0.8614992 0.8143241
> averageP <- pt(t,df=pmin(num1,num2)-1)
> mean(averageP)
[1] 0.7420474
>
```

Step 3: Compare the p-value with the significance level α = 5%

- P-Value of Sound > α (0.1284261 > 0.05)
- P-Value of Light > α (0.543277 > 0.05)
- P-Value of Pulse > α (0.7420474 > 0.05)
- The p-value of each signal test is bigger than the significance level

Step 4: Conclusion:

- Cannot reject the null hypothesis
- We do not have strong evidence to conclude that Genders and Signal factors does interact with each other.

Exercise 3:

Step 1: Set up hypothesis

- H_0 : The size of the company does not affect the advertising effectiveness
- H_1 : The size of the company affects the advertising effectiveness

Step 2: Calculate p-value

The data is shown below:

Company size category	Advertising effectiveness		
	High	Moderate	Low
Small	20	52	32
Medium	53	47	28
Large	67	32	25

```
~/ex3/
> no.company = matrix(c(20,52,32,53,47,28,67,32,25), ncol = 3 , byrow = TRUE)
> colnames(no.company) = c("High","Moderate","Low")
> rownames(no.company) = c("Small","Medium","Large")
> no.company = as.table(no.company)
> Chi_test = chisq.test(no.company, simulate.p.value = TRUE)
> Chi_test

        Pearson's Chi-squared test with simulated p-value (based on 2000 replicates)

data:  no.company
X-squared = 29.638, df = NA, p-value = 0.0004998
```

Step 3: Compare p-value with significant level $\alpha = 0.1$

$p < \alpha$ $(0.004998 < 0.1)$

Therefore, we can reject the null hypothesis

Step 4: Conclusion:

With significance $\alpha = 0.1$, we can conclude that the company size affects the advertising effectiveness

Exercise 4:

*Using anova:

Step 1: Set up hypothesis

- Null-hypothesis H_0 : There is no significant difference in the number of late arrivals among different days of the week.
- Alternative-hypothesis H_1 : There exist some differences in the number of late arrivals among different days of the week.

Step 2: Calculate F-value and p_value in ANOVA using function *aov()*

```
> monday = c(5,4,5,7)
> tuesday = c(4,5,3,2)
> wednesday = c(4,3,4,5)
> thursday = c(4,4,3,2)
> x = c(monday,tuesday,wednesday,thursday)
> x
 [1] 5 4 5 7 4 5 3 2 4 3 4 5 4 4 3 2
> group = c(rep("monday",4),rep("tuesday",4),rep("wednesday",4),rep("thursday",4))
> dat=data.frame(x,group)
> dat
   x     group
1  5    monday
2  4    monday
3  5    monday
4  7    monday
5  4   tuesday
6  5   tuesday
7  3   tuesday
8  2   tuesday
9  4 wednesday
10 3 wednesday
11 4 wednesday
12 5 wednesday
13 4  thursday
14 4  thursday
15 3  thursday
16 2  thursday
> av = aov(x ~ as.factor(group))
> summary(av)
                 Df Sum Sq Mean Sq F value Pr(>F)
as.factor(group)  3    9.5   3.167   2.621 0.0988 .
Residuals        12   14.5   1.208
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> |
```

Step 3: Compare p-value with significant level $\alpha = 5\%$

$p > \alpha$ (0.0988 > 0.05)

⇒We can’t reject H0

Step 4: Conclusion:

With a significant level α = 5%, there is no significant difference in the number of late arrivals among different days of the week.

Project 2 - Topic 1

1 Base Theory

What Is Multiple Linear Regression (MLR)?

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variables.

Formula and Calculation of Multiple Linear Regression

$$y_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + \ldots + w_n x_{in} + error$$

where: for i = n observations:

y_i: dependent variable

x_i: explanatory variables

w_i: slope coefficients for each variable

w_0: y-intercept

error: model’s error term (aka residuals)

2 Introduction

We want to explore what factors may affect home prices in a particular region and luckily there is a good model which can help us to do so, that is linear regression. Our group uses the data set about house sale prices in King County-U.S, which includes Seattle. Those data were collected between May 2014 and May 2015 for analysis purposes. The data set is classified into many groups but we will mainly focus on 6 features in order to have an insight into the demands and forecast price for a specific house with input features(ex: 3 floors, 1000 square feet,...). In addition, the source of our data set is taken from the BKEL, which is provided by our lecturer Nguyễn Tiến Dũng.

3 Data Interpretation

- Data Description

• price - Price of each home sold

• sqft_living - Square footage of the apartments interior living space

• floors - Number of floors

• condition - An index from 1 to 5 on the condition of the apartment,

• sqft_above - The square footage of the interior housing space that is above ground level

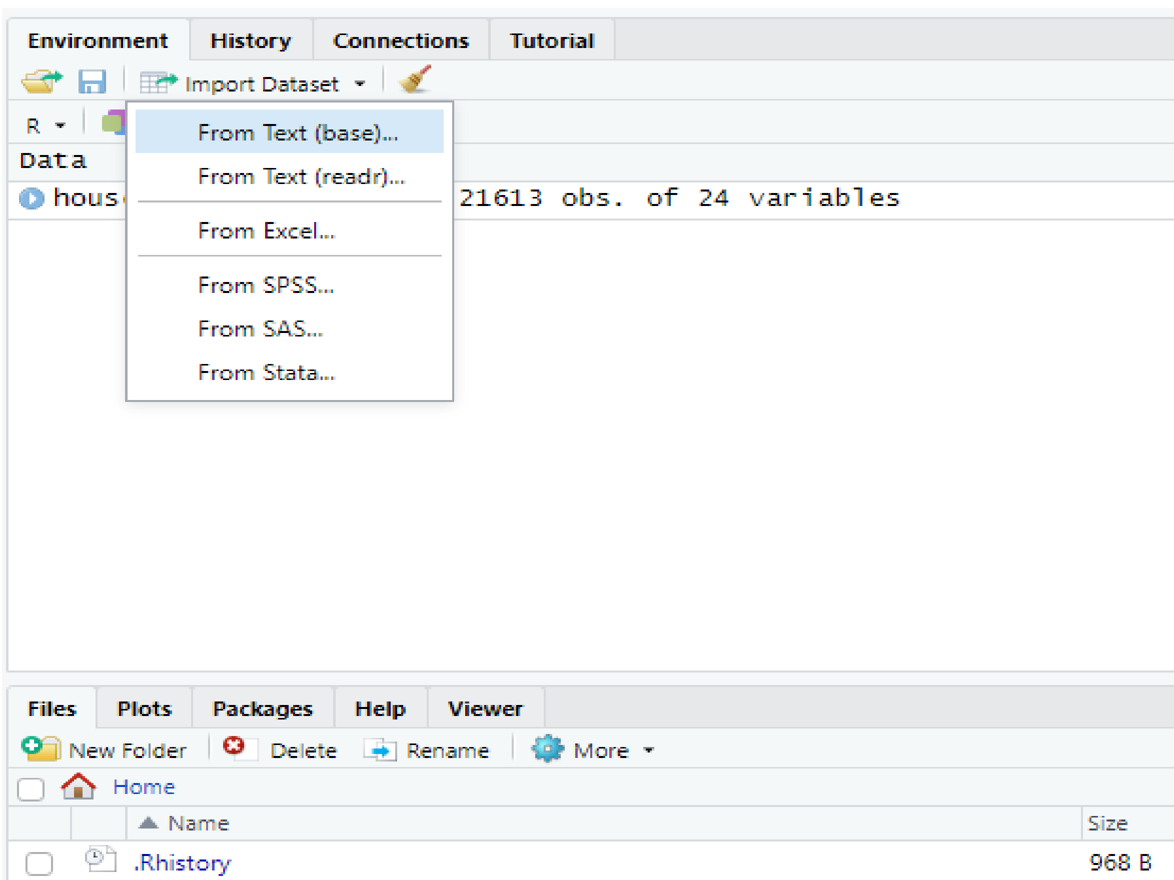
• sqft_living15 - The square footage of interior housing living space for the nearest 15 neighbors

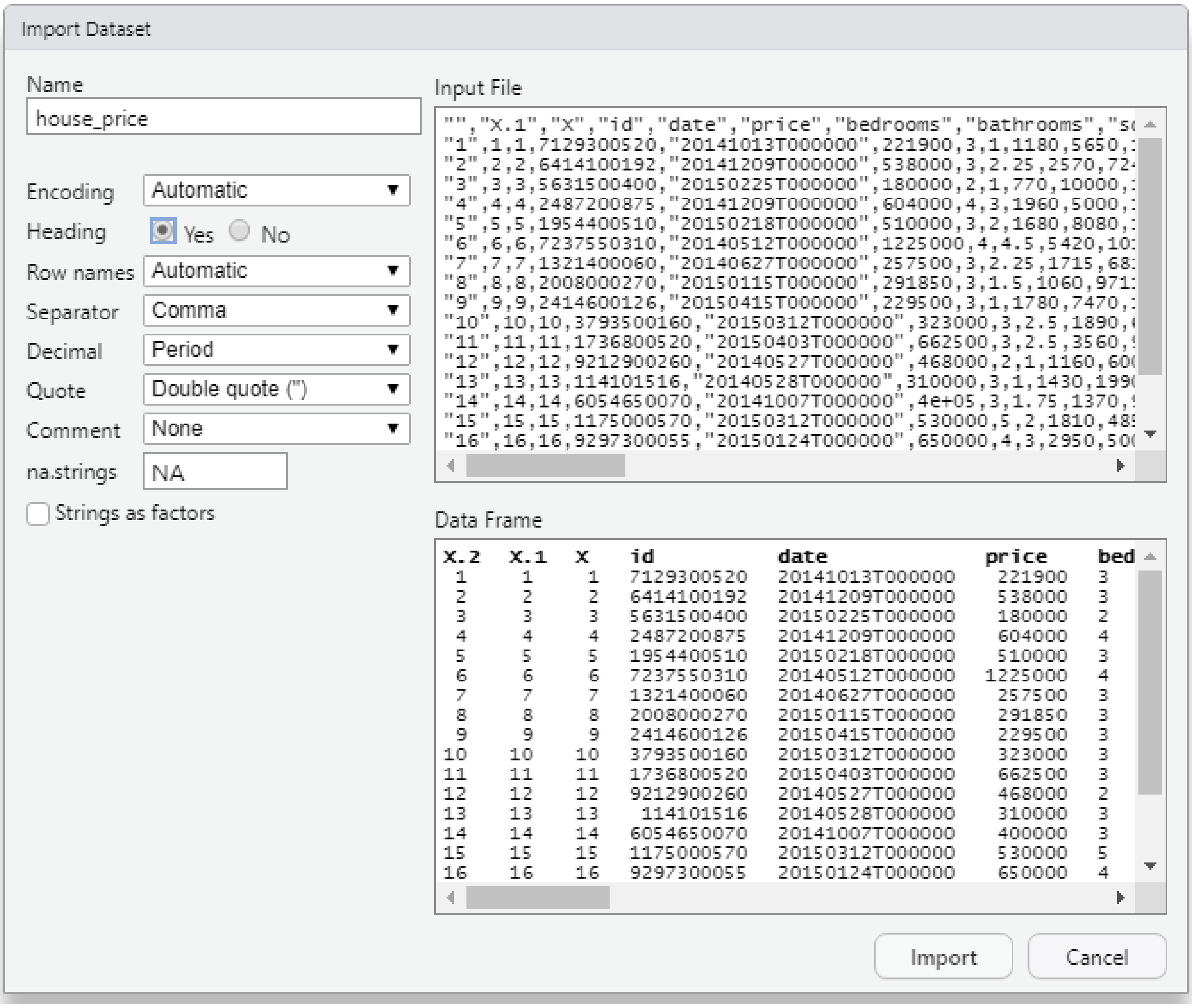
Step 1: import data: **house_price.csv**

+First way: type below command

>house_price <- read.csv("C:/Users/HP/Desktop/house_price.csv")

+Second way: Open R studio then click the "**import dataset**" on the Environment tab, choose from Text (base)… and then import the downloaded file.




Import Dataset
Name
house_price
Encoding
Automatic
Heading
Yes
No
Row names
Automatic
Separator
Comma
Decimal
Period
Quote
Double quote (")
Comment
None
na.strings
NA
Strings as factors
Input File
"","X.1","X","id","date","price","bedrooms","bathrooms","s
"1",1,1,7129300520,"20141013T000000",221900,3,1,1180,5650,
"2",2,2,6414100192,"20141209T000000",538000,3,2.25,2570,72
"3",3,3,5631500400,"20150225T000000",180000,2,1,770,10000,
"4",4,4,2487200875,"20141209T000000",604000,4,3,1960,5000,
"5",5,5,1954400510,"20150218T000000",510000,3,2,1680,8080,
"6",6,6,7237550310,"20140512T000000",1225000,4,4.5,5420,10
"7",7,7,1321400060,"20140627T000000",257500,3,2.25,1715,68
"8",8,8,2008000270,"20150115T000000",291850,3,1.5,1060,971
"9",9,9,2414600126,"20150415T000000",229500,3,1,1780,7470,
"10",10,10,3793500160,"20150312T000000",323000,3,2.5,1890,
"11",11,11,1736800520,"20150403T000000",662500,3,2.5,3560,
"12",12,12,9212900260,"20140527T000000",468000,2,1,1160,60
"13",13,13,114101516,"20140528T000000",310000,3,1,1430,199
"14",14,14,6054650070,"20141007T000000",4e+05,3,1.75,1370,
"15",15,15,1175000570,"20150312T000000",530000,5,2,1810,48
"16",16,16,9297300055,"20150124T000000",650000,4,3,2950,50
Data Frame
X.2 X.1 X id date price bed
1 1 1 7129300520 20141013T000000 221900 3
2 2 2 6414100192 20141209T000000 538000 3
3 3 3 5631500400 20150225T000000 180000 2
4 4 4 2487200875 20141209T000000 604000 4
5 5 5 1954400510 20150218T000000 510000 3
6 6 6 7237550310 20140512T000000 1225000 4
7 7 7 1321400060 20140627T000000 257500 3
8 8 8 2008000270 20150115T000000 291850 3
9 9 9 2414600126 20150415T000000 229500 3
10 10 10 3793500160 20150312T000000 323000 3
11 11 11 1736800520 20150403T000000 662500 3
12 12 12 9212900260 20140527T000000 468000 2
13 13 13 114101516 20140528T000000 310000 3
14 14 14 6054650070 20141007T000000 400000 3
15 15 15 1175000570 20150312T000000 530000 5
16 16 16 9297300055 20150124T000000 650000 4
Import
Cancel

***View the data we have just imported:**

>View(house_price)

house_price ×

Filter

	X.2	X.1	X	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	cond
1	1	1	1	7129300520	20141013T000000	221900	3	1.00	1180	5650	1.0	0	0	
2	2	2	2	6414100192	20141209T000000	538000	3	2.25	2570	7242	2.0	0	0	
3	3	3	3	5631500400	20150225T000000	180000	2	1.00	770	10000	1.0	0	0	
4	4	4	4	2487200875	20141209T000000	604000	4	3.00	1960	5000	1.0	0	0	
5	5	5	5	1954400510	20150218T000000	510000	3	2.00	1680	8080	1.0	0	0	
6	6	6	6	7237550310	20140512T000000	1225000	4	4.50	5420	101930	1.0	0	0	
7	7	7	7	1321400060	20140627T000000	257500	3	2.25	1715	6819	2.0	0	0	
8	8	8	8	2008000270	20150115T000000	291850	3	1.50	1060	9711	1.0	0	0	
9	9	9	9	2414600126	20150415T000000	229500	3	1.00	1780	7470	1.0	0	0	
10	10	10	10	3793500160	20150312T000000	323000	3	2.50	1890	6560	2.0	0	0	
11	11	11	11	1736800520	20150403T000000	662500	3	2.50	3560	9796	1.0	0	0	
12	12	12	12	9212900260	20140527T000000	468000	2	1.00	1160	6000	1.0	0	0	
13	13	13	13	114101516	20140528T000000	310000	3	1.00	1430	19901	1.5	0	0	
14	14	14	14	6054650070	20141007T000000	400000	3	1.75	1370	9680	1.0	0	0	
15	15	15	15	1175000570	20150312T000000	530000	5	2.00	1810	4850	1.5	0	0	
16	16	16	16	9297300055	20150124T000000	650000	4	3.00	2950	5000	2.0	0	3	
17	17	17	17	1875500060	20140731T000000	395000	3	2.00	1890	14040	2.0	0	0	
18	18	18	18	6865200140	20140529T000000	485000	4	1.00	1600	4300	1.5	0	0	
19	19	19	19	16000397	20141205T000000	189000	2	1.00	1200	9850	1.0	0	0	
20	20	20	20	7983200060	20150424T000000	230000	3	1.00	1250	9774	1.0	0	0	

Showing 1 to 20 of 21,613 entries, 24 total columns

Step 2: Data cleaning: : NA (Not available)

- As we only focus on 6 main features which were mentioned above in data description, so that we extract only those variables and put it in our data frame(df):

> attach(house_price)

>df=data.frame(price,sqft_living,floors,condition,sqft_above,sqft_living15,stringsAsFactors = FALSE)

> View(df)

house_price | df

Filter

	price	sqft_living	floors	condition	sqft_above	sqft_living15
1	221900	1180	1.0	3	1180	1340
2	538000	2570	2.0	3	2170	1690
3	180000	770	1.0	3	770	2720
4	604000	1960	1.0	5	1050	1360
5	510000	1680	1.0	3	1680	1800
6	1225000	5420	1.0	3	3890	4760
7	257500	1715	2.0	3	1715	2238
8	291850	1060	1.0	3	1060	1650
9	229500	1780	1.0	3	1050	1780
10	323000	1890	2.0	3	1890	2390
11	662500	3560	1.0	3	1860	2210
12	468000	1160	1.0	4	860	1330
13	310000	1430	1.5	4	1430	1780
14	400000	1370	1.0	4	1370	1370
15	530000	1810	1.5	3	1810	1360
16	650000	2950	2.0	3	1980	2140
17	395000	1890	2.0	3	1890	1890
18	485000	1600	1.5	4	1600	1610
19	189000	1200	1.0	4	1200	1060
20	230000	1250	1.0	4	1250	1280

Showing 1 to 21 of 21,613 entries, 6 total columns

- After that, we check whether Nan-value occurs or not:

  > summary(is.na(df))

```
> summary(is.na(df))
   price          sqft_living       floors          condition        sqft_above       sqft_living15
 Mode :logical    Mode :logical    Mode :logical    Mode :logical    Mode :logical    Mode :logical
 FALSE:21593      FALSE:21613      FALSE:21613      FALSE:21613      FALSE:21613      FALSE:21613
 TRUE :20
>
```

- Because of existing not-available value (20 nan-value in price column), we have to exclude those value and also get rid of some data from other features accompanied with those nan- value:

  > df <- na.omit(df)

  > summary(is.na(df))

```
> summary(is.na(df))
   price          sqft_living       floors          condition        sqft_above       sqft_living15
 Mode :logical    Mode :logical    Mode :logical    Mode :logical    Mode :logical    Mode :logical
 FALSE:21593      FALSE:21593      FALSE:21593      FALSE:21593      FALSE:21593      FALSE:21593
> |
```

Step 3: Data visualization

(a) Transformation:

- We summarize our data frame at first to have a look on basic attributes such as min-value, max-value, mean, median:

```
> summary(df)
     price          sqft_living        floors          condition        sqft_above      sqft_living15
 Min.   :  75000   Min.   :  290   Min.   :1.000   Min.   :1.000   Min.   : 290   Min.   : 399
 1st Qu.: 322000   1st Qu.: 1427   1st Qu.:1.000   1st Qu.:3.000   1st Qu.:1190   1st Qu.:1490
 Median : 450000   Median : 1910   Median :1.500   Median :3.000   Median :1560   Median :1840
 Mean   : 540068   Mean   : 2080   Mean   :1.494   Mean   :3.409   Mean   :1788   Mean   :1987
 3rd Qu.: 645000   3rd Qu.: 2550   3rd Qu.:2.000   3rd Qu.:4.000   3rd Qu.:2210   3rd Qu.:2360
 Max.   :7700000   Max.   :13540   Max.   :3.500   Max.   :5.000   Max.   :9410   Max.   :6210
> |
```

- Because the value of some variables are quite large (ex: price) compared with the others and most of those features have right-skewed distribution so we transform the data by *using a log function* to make it easier for computation and look like Normal Distribution.

```
> logDF <- log(df)
> View(logDF)
```

house_price × | df × | logDF ×

Filter

	price	sqft_living	floors	condition	sqft_above	sqft_living15
1	12.30998	7.073270	0.0000000	1.0986123	7.073270	7.200425
2	13.19561	7.851661	0.6931472	1.0986123	7.682482	7.432484
3	12.10071	6.646391	0.0000000	1.0986123	6.646391	7.908387
4	13.31133	7.580700	0.0000000	1.6094379	6.956545	7.215240
5	13.14217	7.426549	0.0000000	1.0986123	7.426549	7.495542
6	14.01845	8.597851	0.0000000	1.0986123	8.266164	8.468003
7	12.45877	7.447168	0.6931472	1.0986123	7.447168	7.713338
8	12.58400	6.966024	0.0000000	1.0986123	6.966024	7.408531
9	12.34366	7.484369	0.0000000	1.0986123	6.956545	7.484369
10	12.68541	7.544332	0.6931472	1.0986123	7.544332	7.779049
11	13.40378	8.177516	0.0000000	1.0986123	7.528332	7.700748
12	13.05622	7.056175	0.0000000	1.3862944	6.756932	7.192934
13	12.64433	7.265430	0.4054651	1.3862944	7.265430	7.484369
14	12.89922	7.222566	0.0000000	1.3862944	7.222566	7.222566
15	13.18063	7.501082	0.4054651	1.0986123	7.501082	7.215240
16	13.38473	7.989560	0.6931472	1.0986123	7.590852	7.668561
17	12.88664	7.544332	0.6931472	1.0986123	7.544332	7.544332
18	13.09190	7.377759	0.4054651	1.3862944	7.377759	7.383989
19	12.14950	7.090077	0.0000000	1.3862944	7.090077	6.966024
20	12.34583	7.130899	0.0000000	1.3862944	7.130899	7.154615

Showing 1 to 21 of 21,593 entries, 6 total columns

- Later on, we will use logDF as our data frame

(b) Descriptive statistics for each of the variables:

- There is a library used for doing this stuff : “psych”

  > install.packages("psych")

  > library(psych)

- First,we will take a general look inside out data frame:

  > describe(df)   #describe original data frame

  > describe(logDF)   #describe data frame after transformation

```
> library(psych)
> describe(df)
              vars     n      mean        sd   median  trimmed       mad   min       max     range skew kurtosis
price            1 21593 540067.61 367074.23 450000.0 481706.46 222390.00 75000 7700000.0 7625000.0 4.03    34.62
sqft_living      2 21593   2079.86    918.37   1910.0   1984.39    800.60   290   13540.0   13250.0 1.47     5.25
floors           3 21593      1.49      0.54      1.5      1.45      0.74     1       3.5       2.5 0.62    -0.48
condition        4 21593      3.41      0.65      3.0      3.30      0.00     1       5.0       4.0 1.03     0.52
sqft_above       5 21593   1788.41    828.16   1560.0   1682.94    667.17   290    9410.0    9120.0 1.45     3.40
sqft_living15    6 21593   1986.53    685.29   1840.0   1914.06    607.87   399    6210.0    5811.0 1.11     1.60
                   se
price         2498.03
sqft_living      6.25
floors           0.00
condition        0.00
sqft_above       5.64
sqft_living15    4.66
> describe(logDF)
              vars     n  mean   sd median trimmed  mad   min   max range  skew kurtosis se
price            1 21593 13.05 0.53  13.02   13.03 0.51 11.23 15.86  4.63  0.43     0.69  0
sqft_living      2 21593  7.55 0.42   7.55    7.55 0.43  5.67  9.51  3.84 -0.04    -0.06  0
floors           3 21593  0.34 0.35   0.41    0.32 0.60  0.00  1.25  1.25  0.28    -1.50  0
condition        4 21593  1.21 0.18   1.10    1.19 0.00  0.00  1.61  1.61  0.34     2.44  0
sqft_above       5 21593  7.39 0.43   7.35    7.38 0.45  5.67  9.15  3.48  0.25    -0.32  0
sqft_living15    6 21593  7.54 0.33   7.52    7.53 0.34  5.99  8.73  2.74  0.21    -0.21  0
```

-> We can see all the statistics for each of the variables like: number of values in each feature, mean, standard deviation, min, max, range, …

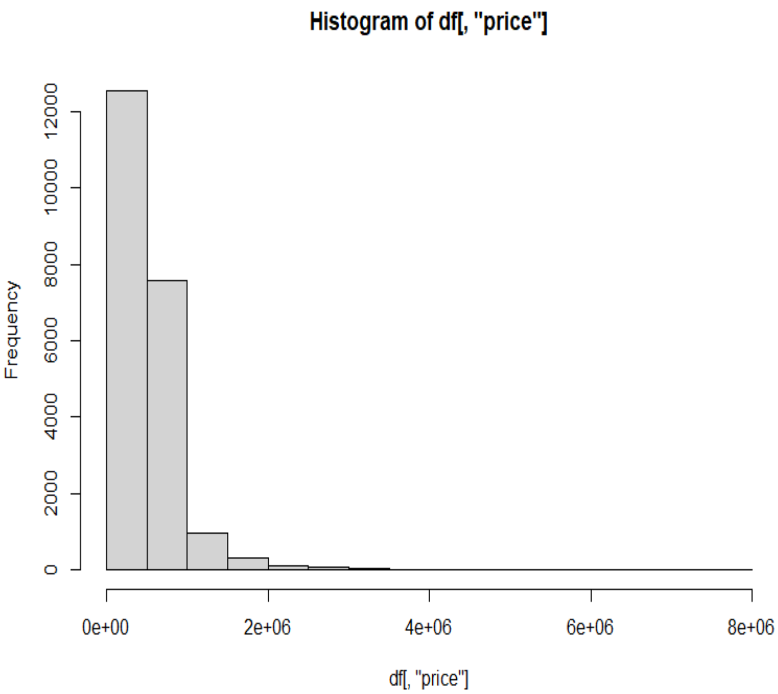
(c) Graphs: visualize using histogram, boxplot, pairs

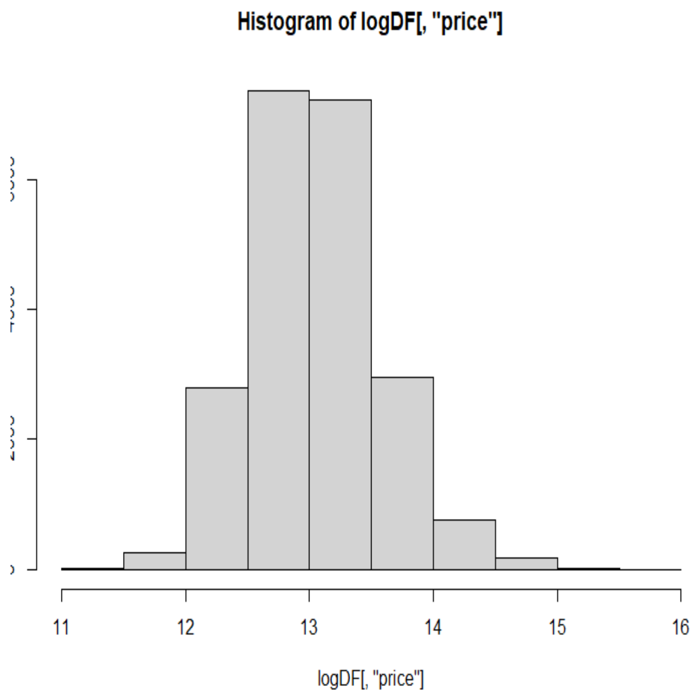
- hist: help us to see the frequency at each level

  *price:

  > hist (df[,"price"])

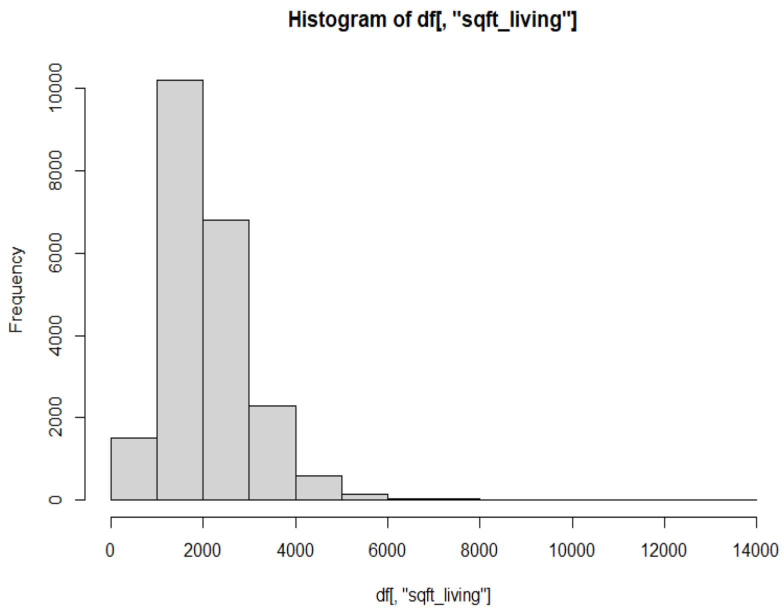
  > hist (logDF[,"price"])

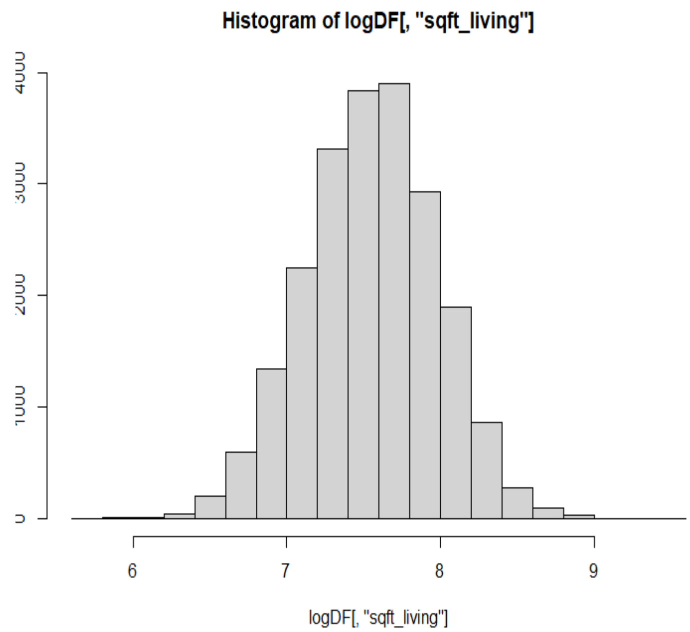




*sqft_living:

```
> hist (df[,"sqft_living"])
> hist (logDF[,"sqft_living"])
```

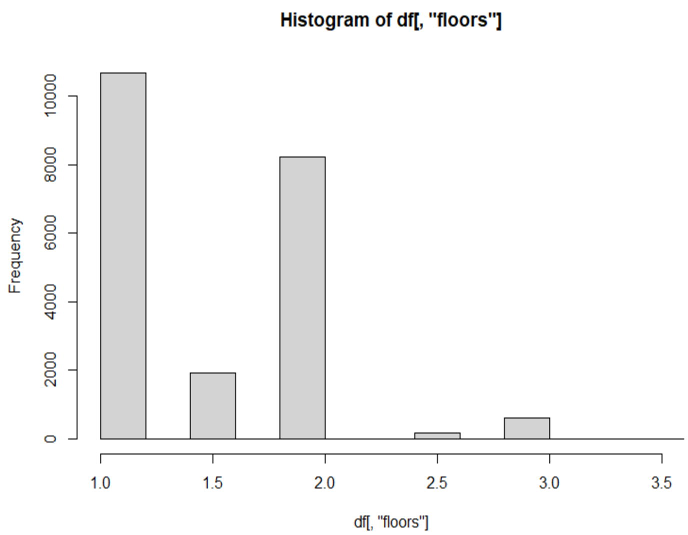


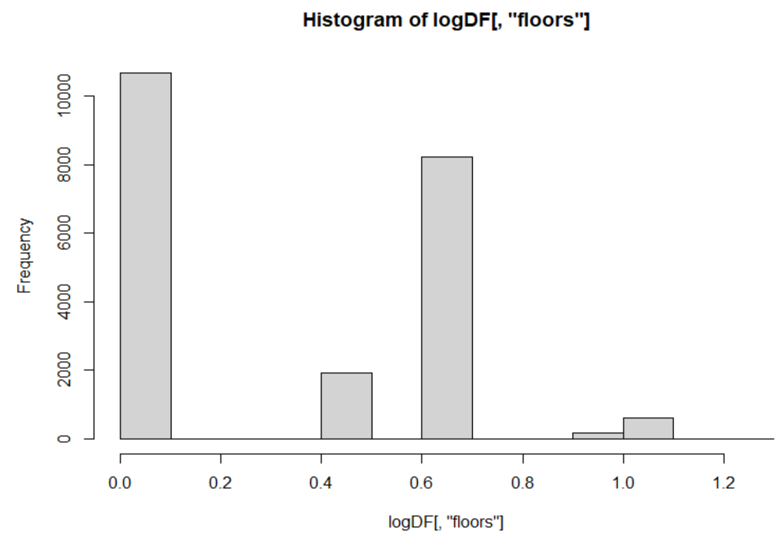


*floors:

> hist (df[,"floors"])

> hist (logDF[,"floors"])

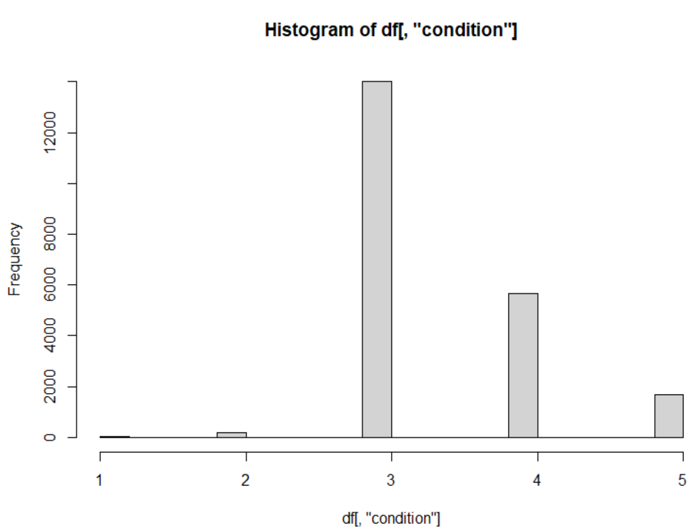


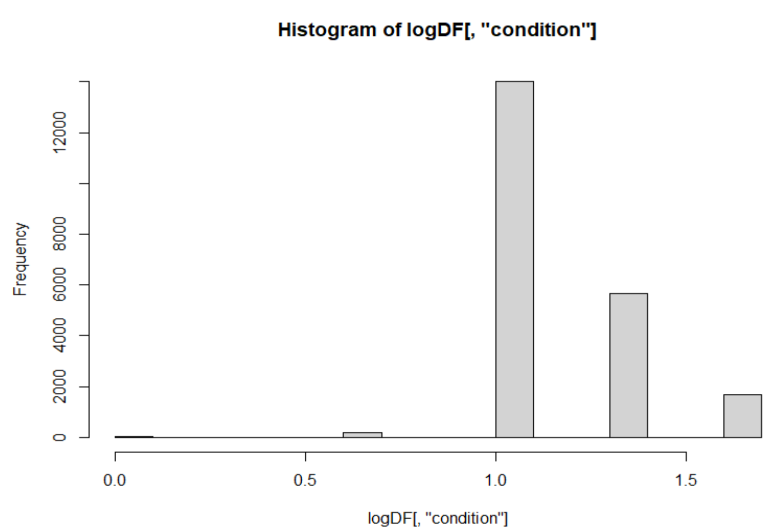


*condition:

> hist (df[,"condition"])

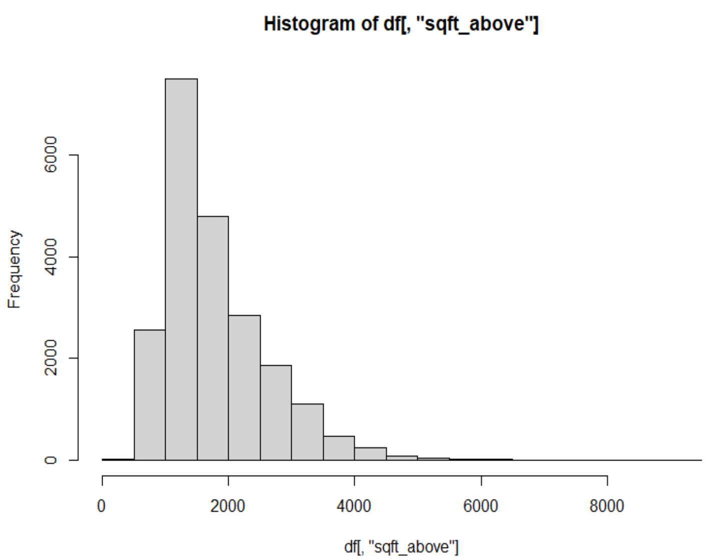
> hist (logDF[,"condition"])

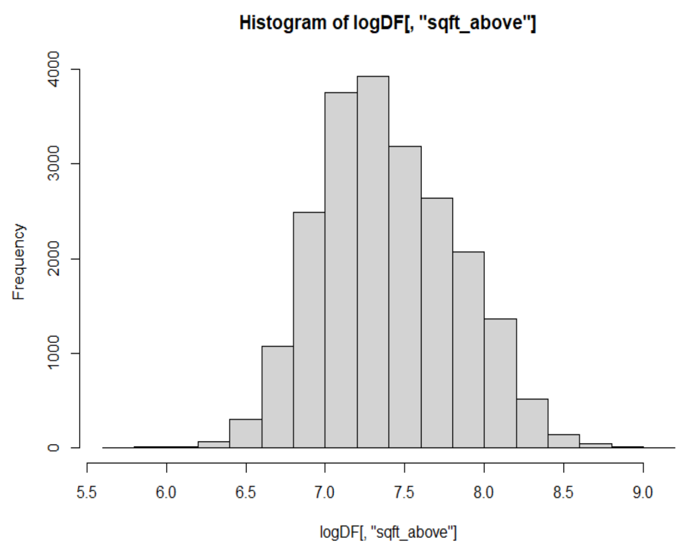




*sqft_above:

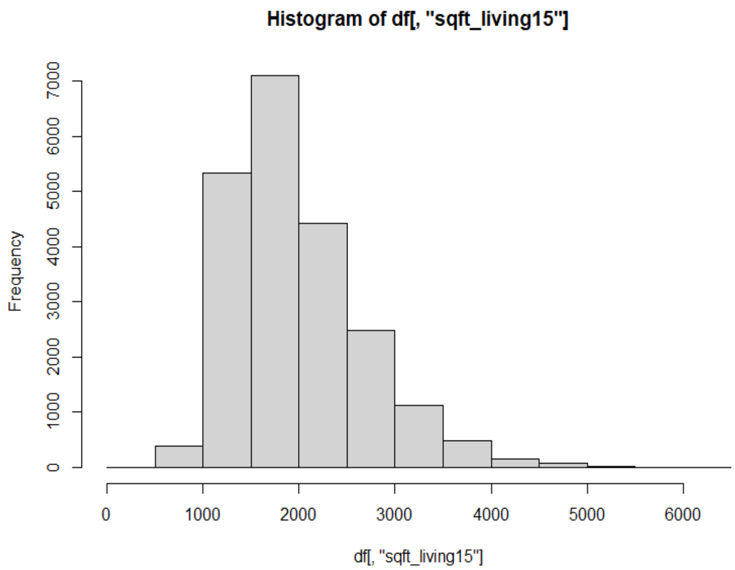
```
> hist (df[,"sqft_above"])
> hist (logDF[,"sqft_above"])
```

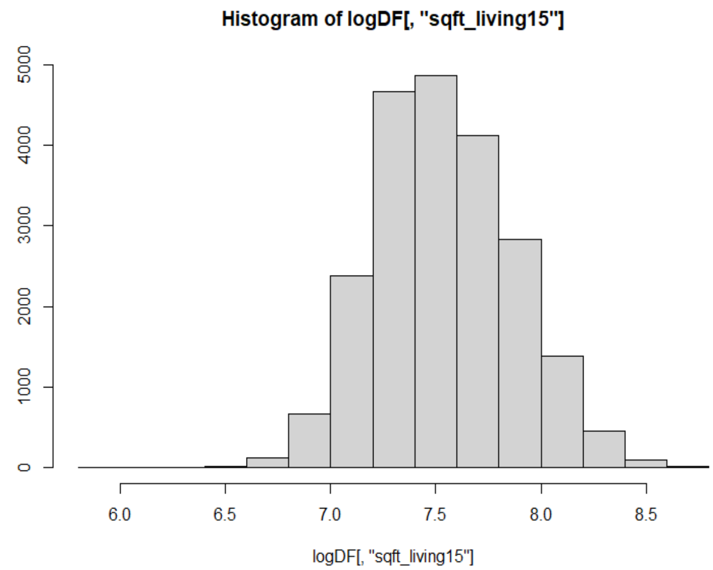




*sqft_living15:

```
> hist (df[,"sqft_living15"])
> hist (logDF[,"sqft_living15"])
```



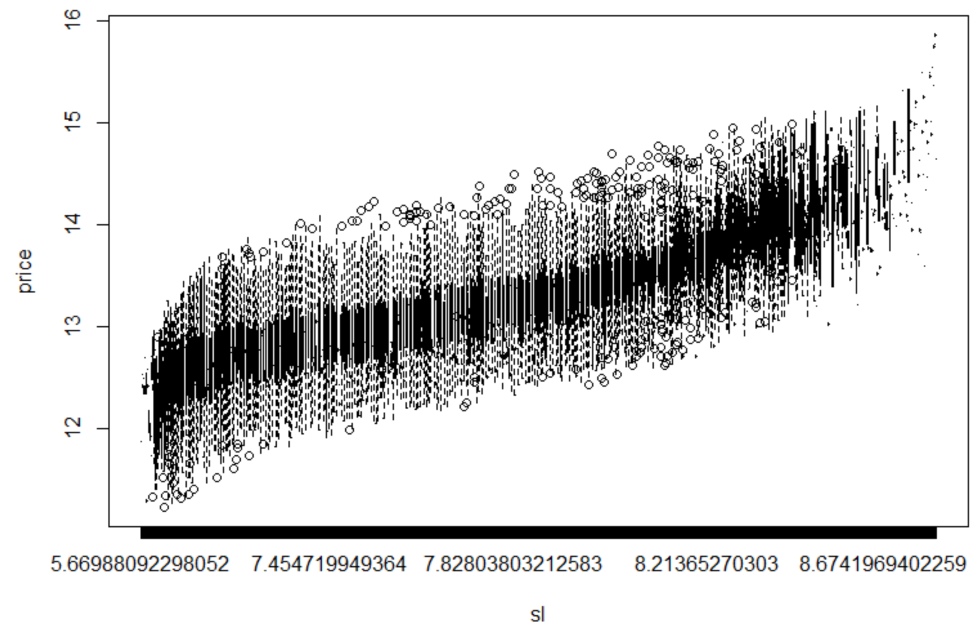


- boxplot: draw the relation between price and other features in Log data frame (logDF)
  - Prepare:

    ```
    > price = logDF[,"price"]
    > sl = logDF[,"sqft_living"]
    > floors = logDF[,"floors"]
    > condition = logDF[,"condition"]
    > sa = logDF[,"sqft_above"]
    > sl15 = logDF[,"sqft_living15"]
    ```

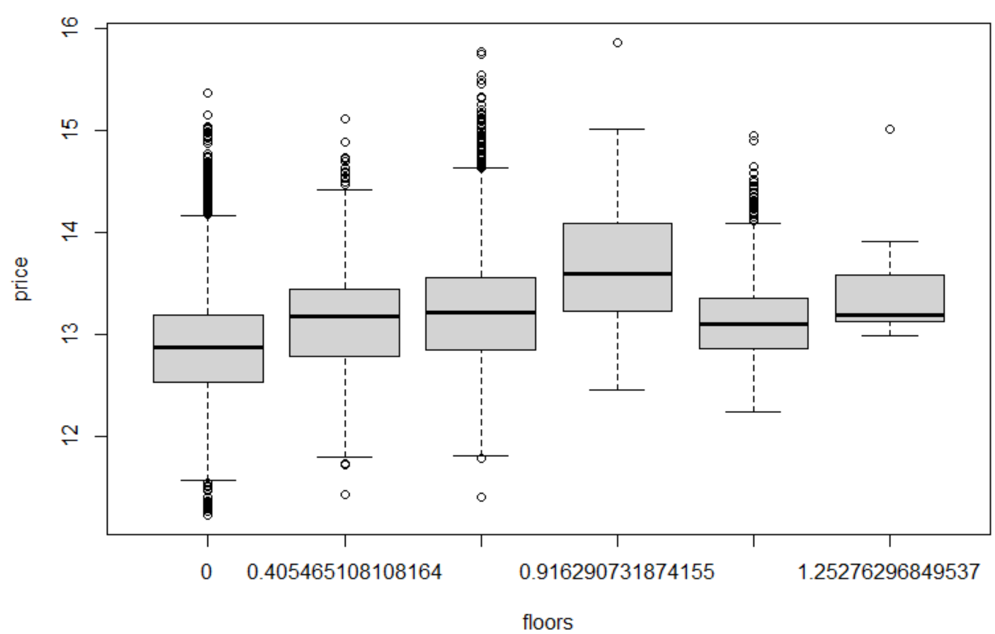
  - price ~ sqft_living:

    ```
    > boxplot(price~sl)
    ```



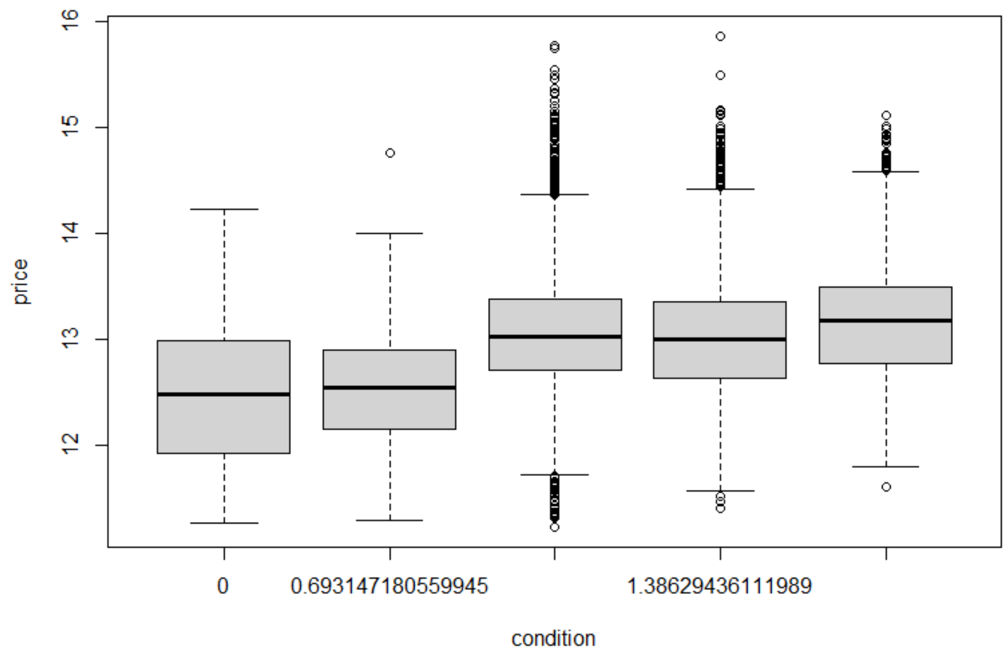
+ price ~ floors:

> boxplot(price~floors)



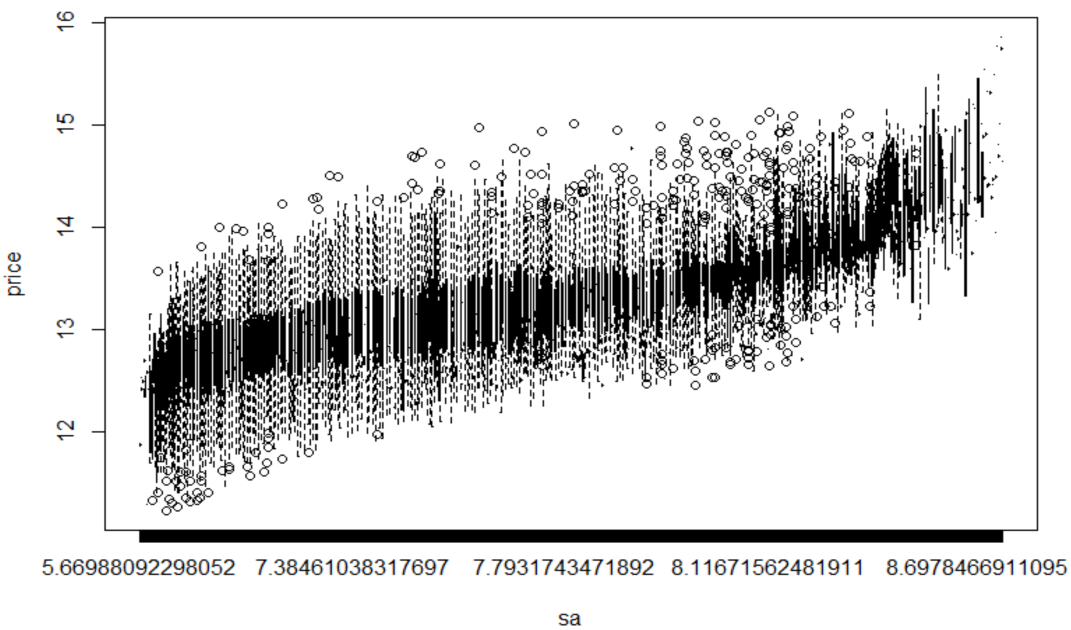
+ price ~ condition:

> boxplot(price~condition)



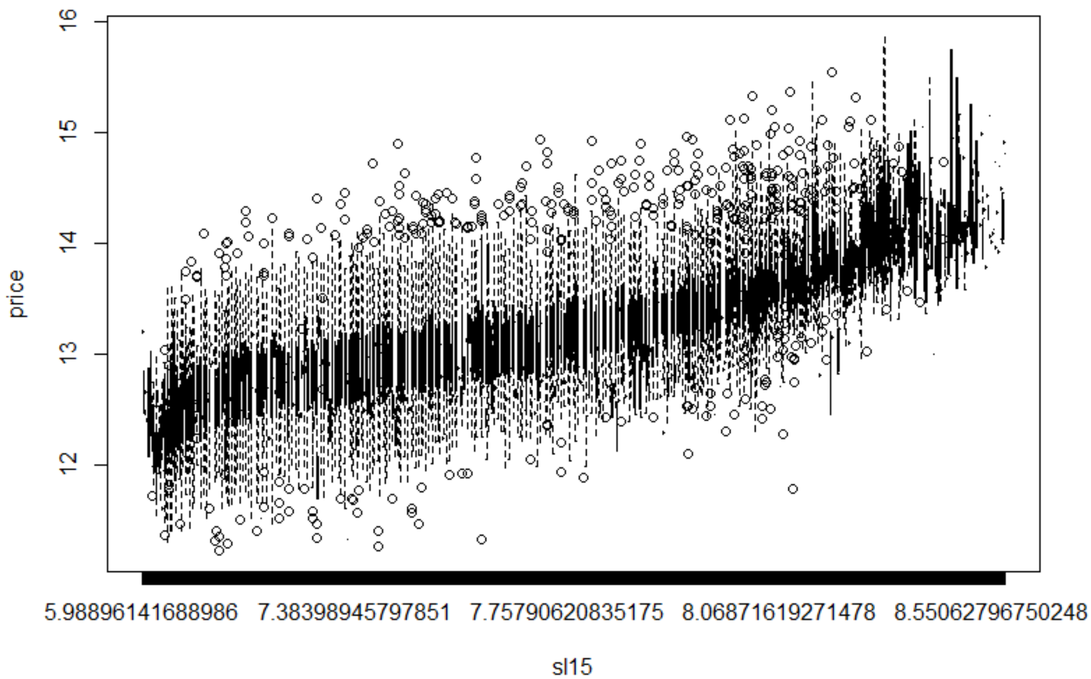
+ price ~ sqft_above:

> boxplot(price~sa)



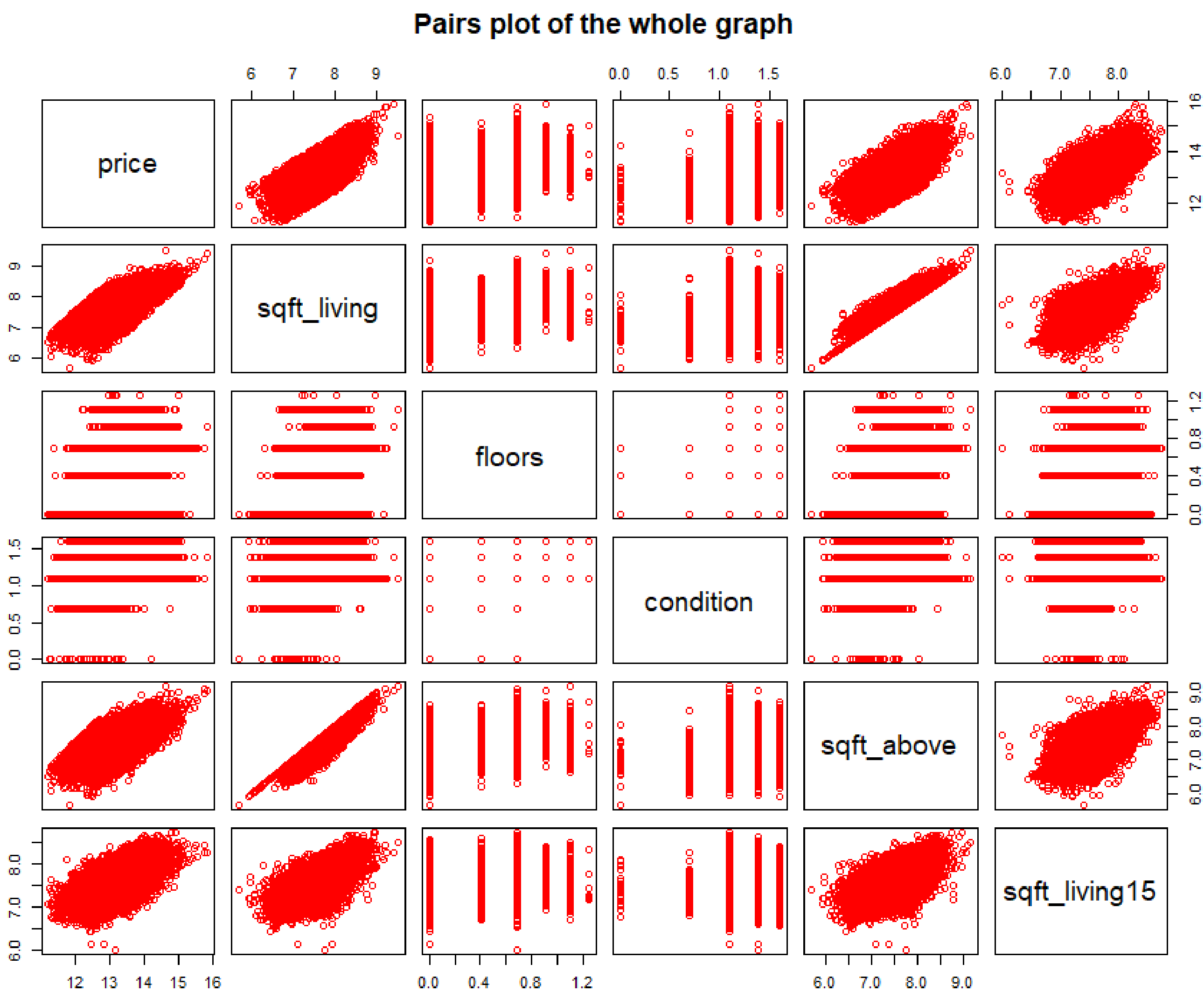
+ price ~ sqft_living15:

> boxplot(price~sl15)


price
16
15
14
13
12
5.98896141688986
7.38398945797851
7.75790620835175
8.06871619271478
8.55062796750248
sl15

- pairs: illustrate the whole dataframe(logDF) in terms of matrix using *pairs function*

> pairs(logDF,col="red",main ="Pairs plot of the whole graph")



Step 4. Fitting linear regression models to explore how these factors affect house prices in King County.

Now we perform a linear regression analysis to evaluate the relationship between the independent and dependent variables. Let's see if there's a linear relationship between price to condition, floors, sqft_living, sqft_living15, sqft_above.

- Build the linear regression model:

  > linear_model <- lm(price ~ sqft_living + floors + condition + sqft_above +sqft_living15, data = logDF)

  > summary(linear_model)

```
> summary(linear_model)

Call:
lm(formula = price ~ sqft_living + floors + condition + sqft_above +
    sqft_living15, data = logDF)

Residuals:
     Min       1Q   Median       3Q      Max
-1.25252 -0.27608  0.00872  0.24537  1.50616

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    5.555655   0.065254   85.14   <2e-16 ***
sqft_living    0.687838   0.013231   51.99   <2e-16 ***
floors         0.202622   0.009371   21.62   <2e-16 ***
condition      0.292837   0.014591   20.07   <2e-16 ***
sqft_above    -0.182165   0.014248  -12.79   <2e-16 ***
sqft_living15  0.427492   0.011993   35.65   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3734 on 21587 degrees of freedom
Multiple R-squared:  0.4973,	Adjusted R-squared:  0.4971
F-statistic:  4270 on 5 and 21587 DF,  p-value: < 2.2e-16
```

- According to the figure above, we have a weight-table:

	sqft_living	floors	condition	sqft_above	sqft_living15
estimated weight	0.6878	0.2026	0.2928	-0.1822	0.4275

=> It means that the price of house in King county can be represented by following equation:

price = 5.5556 + 0.6878*sqft_living + 0.2026*floors + 0.2928*condition - 0.1822*sqft_above + 0.4275*sqft_living15

- The standard errors for these regression coefficients are not really significant, and the t-statistics are quite large. The *p*-values reflect these small errors and large t-statistics. For both parameters, there is almost zero probability that the house price is a random variable => it is dependent variable

Step 5: Prediction

- Case 1: sqft_living15 = mean(sqft_living15), sqft_above = mean(sqft_above), sqft_living = mean(sqft_living), floor = 2, condition = 3
    + prepare testing data frame:

```
> sqft_living <- mean(logDF[,"sqft_living"])

> floors <- 2

> condition <- 3

> sqft_above <- mean(logDF[,"sqft_above"])

> sqft_living15<-mean(logDF[,"sqft_living15"])

> dat = data.frame(sqft_living,floors,condition,sqft_above,sqft_living15)

> dat
```

```
> dat
  sqft_living floors condition sqft_above sqft_living15
1    7.550329      2         3   7.394883      7.539447
```

+ automatically predict:

> predict(linear_model,dat)

```
> predict(linear_model,dat)
       1
13.90877
>
```

+ manually predict:

price = 5.5556 + 0.6878*7.5503 + 0.2026*2 + 0.2928*3 - 0.1822*7.3949 + 0.4275*7.5394 ≅ 13.908

Because we use log transformation -> the value: 13.90877 means that the $predicted\ price \cong e^{13.90877} = 1097746.5$ (dolar)

- Case 2: sqft_living15 = max(sqft_living15), sqft_above = max(sqft_above), sqft_living = max(sqft_living), floor = 2, condition = 3.
  + prepare testing data frame:

```
> sqft_living <- max(logDF[,"sqft_living"])
> floors <- 2
> condition <- 3
> sqft_above <- max(logDF[,"sqft_above"])
> sqft_living15<-max(logDF[,"sqft_living15"])
> dat = data.frame(sqft_living,floors,condition,sqft_above,sqft_living15)
> View(dat)
```

	sqft_living	floors	condition	sqft_above	sqft_living15
1	9.513404	2	3	9.149528	8.733916

+ automatically predict:

> predict(linear_model,dat)

```
> predict(linear_model,dat)
       1
15.45004
```

+ manually predict:

  price = 5.5556 + 0.6878*9.5134 + 0.2026*2 + 0.2928*3 - 0.1822*9.1495 + 0.4275*8.7339 ≅ 15.45

Because of using log transformation -> the value: 15.45004 means that the *predicted price* ≅ $e^{15.45004}$ = 5127044.858 (dolar)