Tutorial 1

Q1) we have the role of height by predicting weight based on the following sample:

data 1

height	weight
160	58
167	65
170	73
175	74
174	78
169	65
168	67
163	61

Find

1- mean, harmonic mean, geometric mean, range, quantiles, variance, and standard deviation for height.
2- histogram plot for height.
3- simple linear regression equation, correlation coefficient, and Scatter plot.
4- AIC, BIC, and LOOCV.

Solution:

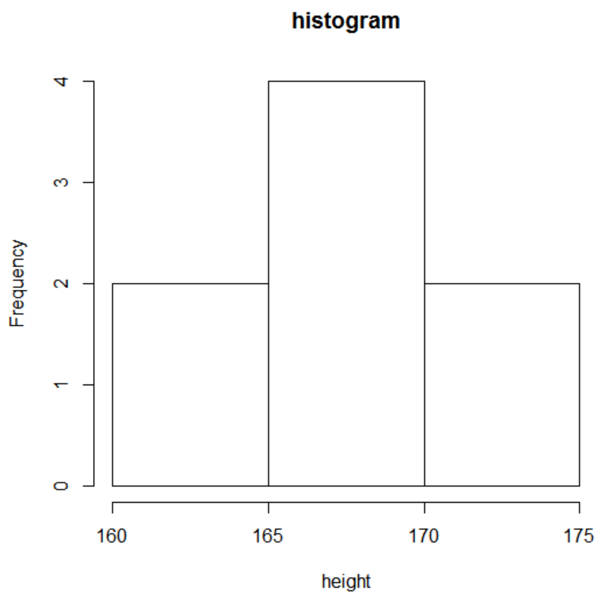
```
> setwd("c:/R")
> model0 <- read.csv(file="data1.csv", header=TRUE, sep=";")
> model0
  height weight
1    160     58
2    167     65
3    170     73
4    175     74
5    174     78
6    169     65
7    168     67
8    163     61
> y <- model0$weight
> y
[1] 58 65 73 74 78 65 67 61
> x <- model0$height
> x
[1] 160 167 170 175 174 169 168 163
```

1-

```
> summary(model0)
     height          weight
 Min.   :160.0   Min.   :58.00
 1st Qu.:166.0   1st Qu.:64.00
 Median :168.5   Median :66.00
 Mean   :168.2   Mean   :67.62
 3rd Qu.:171.0   3rd Qu.:73.25
 Max.   :175.0   Max.   :78.00
> harmean <- 1/mean(1/x)
> harmean
[1] 168.1156
> geomean <- exp( mean( log(x) ) )
> geomean
[1] 168.183
> range(x)
[1] 160 175
> max(x)-min(x)
[1] 15

> quantile(x)
   0%   25%   50%   75%  100%
160.0 166.0 168.5 171.0 175.0
> quantile( x, probs=seq(0,1,.25) )
   0%   25%   50%   75%  100%
160.0 166.0 168.5 171.0 175.0
> quantile( x, probs=c(.25,.5,.75) )
  25%   50%   75%
166.0 168.5 171.0
> quantile( x, probs=.5 )
  50%
168.5
> var(x); sd(x)
[1] 25.64286
[1] 5.063878
> var(y); sd(y)
[1] 46.83929
[1] 6.843923
> var(x, y)
[1] 32.53571
> cov(model0[ , 1:2])
         height   weight
height 25.64286 32.53571
weight 32.53571 46.83929
```

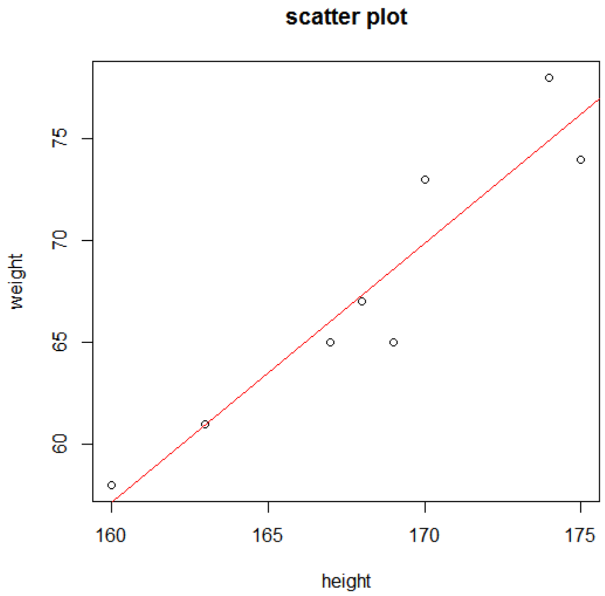
2-

```
> hist(x, main="histogram", xlab="height")
```



3-

```
> model1 <- lm(y~x,data=model0)
> model1

Call:
lm(formula = y ~ x, data = model0)

Coefficients:
(Intercept)            x
   -145.851        1.269

> summary(model1)

Call:
lm(formula = y ~ x, data = model0)

Residuals:
    Min      1Q  Median      3Q     Max
-3.5766 -1.3266 -0.1358  1.4018  3.1546

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -145.8510    31.9907  -4.559 0.003854 **
x              1.2688     0.1901   6.676 0.000547 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.546 on 6 degrees of freedom
Multiple R-squared:  0.8813,    Adjusted R-squared:  0.8616
F-statistic: 44.57 on 1 and 6 DF,  p-value: 0.0005471



> anova(model1)
Analysis of Variance Table

Response: y
          Df  Sum Sq Mean Sq F value    Pr(>F)
x          1 288.970 288.970  44.565 0.0005471 ***
Residuals  6  38.905   6.484
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> vcov(model1)
            (Intercept)           x
(Intercept) 1023.404261 -6.07782308
x             -6.077823  0.03612376


> cor(x, y)
[1] 0.9387977
> cor(model0[ , 1:2])
          height    weight
height 1.0000000 0.9387977
weight 0.9387977 1.0000000

> plot(x, y, main="scatter plot", xlab="height", ylab="weight")
> abline( model1, col="red" )
```



4-

```
> AIC <- AIC(model1)
> AIC
[1] 41.35653
> BIC <- BIC(model1)
> BIC
[1] 41.59485
> attributes(model1)
$names
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"

$class
[1] "lm"

> model1infl <- lm.influence(model1)
```

```
> attributes(model1infl)
$names
[1] "hat"          "coefficients" "sigma"        "wt.res"

> rootloocv <-sqrt( mean ( ( model1$residuals/(1-model1infl$hat) )^2 ))
> rootloocv
[1] 2.896014

> install.packages("caret")

> library(caret)
> attach(model0)
> model2 <- train(weight ~ height,
+                data = model0,
+                method = "lm",
+                trControl = trainControl(method = "LOOCV"))
> model2
Linear Regression

8 samples
1 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 7, 7, 7, 7, 7, 7, ...
Resampling results:

  RMSE      Rsquared   MAE
  2.896014  0.8021643  2.382801

Tuning parameter 'intercept' was held constant at a value of TRUE

> summary(model2)

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-3.5766 -1.3266 -0.1358  1.4018  3.1546

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -145.8510    31.9907  -4.559 0.003854 **
height         1.2688     0.1901   6.676 0.000547 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.546 on 6 degrees of freedom
Multiple R-squared:  0.8813,   Adjusted R-squared:  0.8616
F-statistic: 44.57 on 1 and 6 DF,  p-value: 0.0005471
```

Q2) we have the role of height and blood sugar by predicting weight based on the following sample:

data 2

height	weight	blood sugar
172	70	113
178	80	128
162	60	100
175	75	119
167	60	89
163	59	94
170	90	165

Find

1- multiple linear regression equation, correlation coefficient, and Scatter plot.
2- AIC, BIC, and LOOCV.

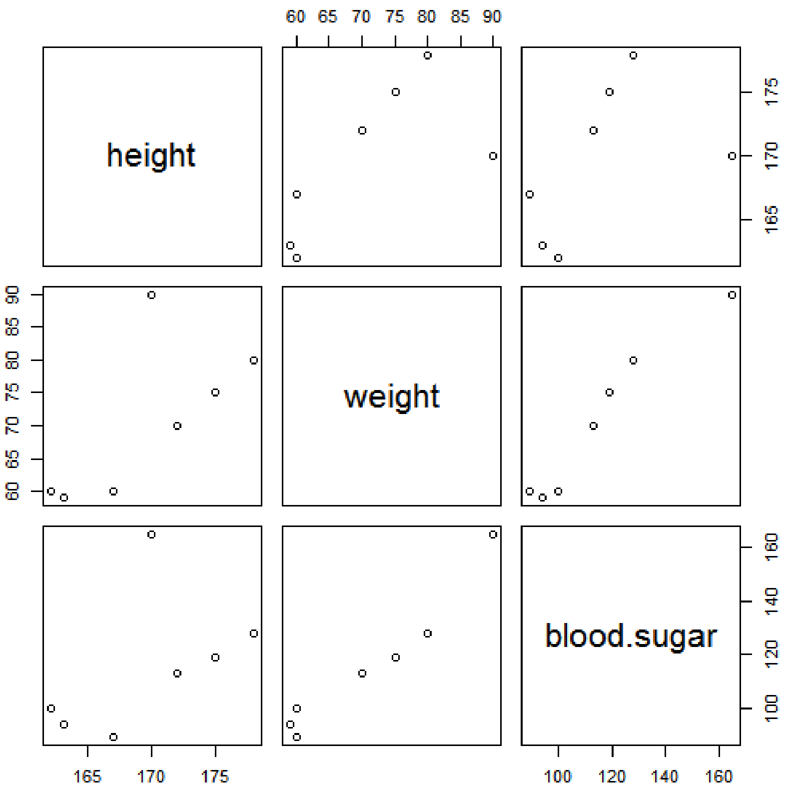
Solution:

```
> setwd("C:/R")
> model0 <- read.csv(file="data2.csv", header=TRUE, sep=";")
> model0
  height weight blood.sugar
1    172     70         113
2    178     80         128
3    162     60         100
4    175     75         119
5    167     60          89
6    163     59          94
7    170     90         165
> attach(model0)
```

1-

```
> model1 <- lm(weight~height+blood.sugar,data=model0)
> model1

Call:
lm(formula = weight ~ height + blood.sugar, data = model0)

Coefficients:
(Intercept)       height   blood.sugar
   -65.4312       0.5425        0.3812

> summary(model1)

Call:
lm(formula = weight ~ height + blood.sugar, data = model0)

Residuals:
       1        2        3        4        5        6        7
-0.96309  0.06308 -0.58168  0.12187  0.89944  0.16328  0.29711

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -65.43125    9.28691  -7.046 0.002139 **
height        0.54252    0.05879   9.228 0.000766 ***
blood.sugar   0.38125    0.01354  28.160 9.46e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7431 on 4 degrees of freedom
Multiple R-squared:  0.9974,    Adjusted R-squared:  0.9961
F-statistic:   762 on 2 and 4 DF,  p-value: 6.853e-06


> anova(model1)
Analysis of Variance Table

Response: weight
            Df Sum Sq Mean Sq F value    Pr(>F)
height       1 403.66  403.66  731.06 1.113e-05 ***
blood.sugar  1 437.84  437.84  792.97 9.462e-06 ***
Residuals    4   2.21    0.55
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> vcov(model1)
            (Intercept)        height   blood.sugar
(Intercept) 86.24671477 -0.5399109808  0.0466577877
height      -0.53991098  0.0034562050 -0.0003999239
blood.sugar  0.04665779 -0.0003999239  0.0001832985
```

```
> cor(height,weight)
[1] 0.6916886
> cor(blood.sugar,weight)
[1] 0.9703868
> cor(model0[,1:3])
               height    weight blood.sugar
height      1.0000000 0.6916886   0.5024562
weight      0.6916886 1.0000000   0.9703868
blood.sugar 0.5024562 0.9703868   1.0000000

plot(model0[,1:3])
```



2-

```
> AIC <- AIC(model1)
> AIC
[1] 19.79043
> BIC <- BIC(model1)
> BIC
[1] 19.57407
> attributes(model1)
$names
 [1] "coefficients" "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"       "qr"            "df.residual"
 [9] "xlevels"      "call"          "terms"         "model"

$class
[1] "lm"

> model1infl <- lm.influence(model1)
> attributes(model1infl)
$names
[1] "hat"          "coefficients" "sigma"        "wt.res"

> rootloocv <- sqrt ( mean ( ( model1$residuals/(1-model1infl$hat) )^2 ) )
> rootloocv
[1] 1.762332



> library(caret)
> model2 <- train(weight ~ height+blood.sugar,
+                 data = model0,
+                 method = "lm",
+                 trControl = trainControl(method = "LOOCV"))
> model2
Linear Regression

7 samples
2 predictors

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 6, 6, 6, 6, 6, 6, ...
Resampling results:

  RMSE      Rsquared   MAE
  1.762332  0.9823768  1.176131

Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
> summary(model2)

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
      X1       X2       X3       X4       X5       X6       X7 
-0.96309  0.06308 -0.58168  0.12187  0.89944  0.16328  0.29711 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -65.43125    9.28691  -7.046 0.002139 ** 
height        0.54252    0.05879   9.228 0.000766 ***
blood.sugar   0.38125    0.01354  28.160 9.46e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7431 on 4 degrees of freedom
Multiple R-squared:  0.9974,    Adjusted R-squared:  0.9961 
F-statistic:   762 on 2 and 4 DF,  p-value: 6.853e-06
```

Q3) we have the treadmill stress test and the incidence of Coronary Heart Disease (CHD). where we have data from 17 treadmill stress tests along with an associated diagnosis of CHD on the following: (يمثل مخطط التشتت لمرضى القلب التاجي)

data 3

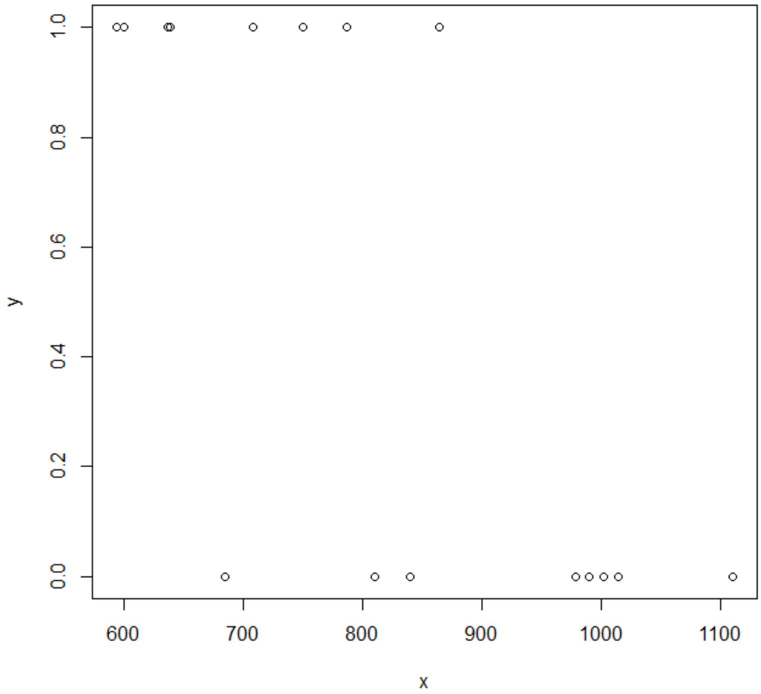
Seconds on treadmill	Presence of coronary Heart Disease (CHD) 0=health, 1= diseased
1014	.00
684	.00
810	.00
990	.00
840	.00
978	.00
1002	.00
1110	.00
864	1.00
636	1.00
638	1.00
708	1.00
786	1.00
600	1.00
750	1.00
594	1.00
750	1.00

Binary outcome

Find the logistic regression equation.

Solution:

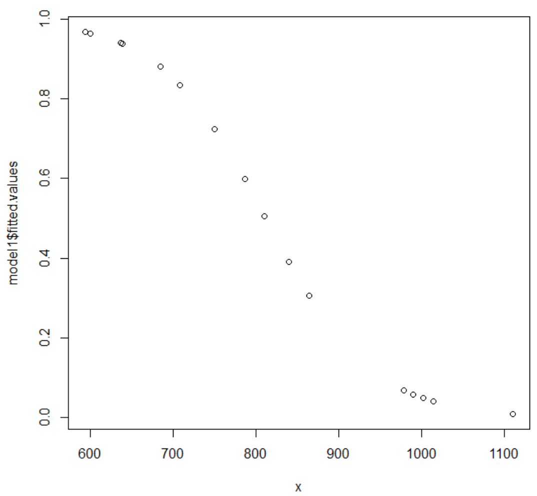
```
> setwd("C:/R")
> model0 <- read.csv(file="data3.csv", header=TRUE, sep=";"
> model0
      x y
1  1014 0
2   684 0
3   810 0
4   990 0
5   840 0
6   978 0
7  1002 0
8  1110 0
9   864 1
10  636 1
11  638 1
12  708 1
13  786 1
14  600 1
15  750 1
16  594 1
17  750 1
> attach(model0)

> plot(model0[,1:2])
```




```
> model1 <- glm(y ~ x,data=model0,family="binomial")
> model1

Call:  glm(formula = y ~ x, family = "binomial", data = model0)

Coefficients:
(Intercept)            x
   12.72736     -0.01568

Degrees of Freedom: 16 Total (i.e. Null);  15 Residual
Null Deviance:      23.51
Residual Deviance: 12.55        AIC: 16.55
> attributes(model1)
$names
 [1] "coefficients"      "residuals"         "fitted.values"
 [4] "effects"           "R"                 "rank"
 [7] "qr"                "family"            "linear.predictors"
[10] "deviance"          "aic"               "null.deviance"
[13] "iter"              "weights"           "prior.weights"
[16] "df.residual"       "df.null"           "y"
[19] "converged"         "boundary"          "model"
[22] "call"              "formula"           "terms"
[25] "data"              "offset"            "control"
[28] "method"            "contrasts"         "xlevels"

$class
[1] "glm" "lm"
```

```
> summary(model1) # display results

Call:
glm(formula = y ~ x, family = "binomial", data = model0)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0627  -0.3439   0.2547   0.5998   1.5405

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 12.727363   5.802709   2.193   0.0283 *
x           -0.015683   0.007255  -2.162   0.0307 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 23.508  on 16  degrees of freedom
Residual deviance: 12.550  on 15  degrees of freedom
AIC: 16.55

Number of Fisher Scoring iterations: 5

> exp(model1$coefficients) # exponentiated coefficients
 (Intercept)            x
3.368398e+05 9.844398e-01
> model1$fitted.values # predicted values
         1          2          3          4          5          6          7
0.04012715 0.88084908 0.50612307 0.05741252 0.39031623 0.06848639 0.04803689
         8          9         10         11         12         13         14
0.00919122 0.30526198 0.94009709 0.93830620 0.83536173 0.59889782 0.96503544
        15         16         17
0.72421101 0.96807516 0.72421101
> AIC <- AIC(model1)
> AIC
[1] 16.55022
> BIC <- BIC(model1)
> BIC
[1] 18.21664
>
> plot(x,model1$fitted.values)
```

Fitted value=p=$\frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$ (كأننا قمنا بالتحويل من رسم يحتوي احتمال حدوث أو عدمه الى رسم يحتوي نسبة كل دقة بالحدوث>>>>>from: p->TO: log(p/(1-p))



```
> confint(modell) # 95% CI for the coefficients using profiled log-likelihood
Waiting for profiling to be done...
                 2.5 %       97.5 %
(Intercept)  4.14459252 29.132348985
x           -0.03628542 -0.005069372
> exp(confint(modell)) # 95% CI for exponentiated coefficients
Waiting for profiling to be done...
               2.5 %       97.5 %
(Intercept) 63.091908 4.487644e+12
x            0.964365 9.949435e-01
> confint.default(modell)  # 95% CI for the coefficients using standard errors
                 2.5 %       97.5 %
(Intercept)  1.35426192 24.100463736
x           -0.02990286 -0.001462251
> exp(confint.default(modell)) # 95% CI for exponentiated coefficients
               2.5 %       97.5 %
(Intercept) 3.8739006 2.928859e+10
x           0.9705398 9.985388e-01
>
```

```
> predict(modell, type="response") # predicted values
         1          2          3          4          5          6          7
0.04012715 0.88084908 0.50612307 0.05741252 0.39031623 0.06848639 0.04803689
         8          9         10         11         12         13         14
0.00919122 0.30526198 0.94009709 0.93830620 0.83536173 0.59889782 0.96503544
        15         16         17
0.72421101 0.96807516 0.72421101
> residuals(modell, type="deviance") # residuals
         1          2          3          4          5          6          7
-0.2861973 -2.0626994 -1.1878290 -0.3438795 -0.9948014 -0.3766815 -0.3137802
         8          9         10         11         12         13         14
-0.1358950  1.5405096  0.3514886  0.3568724  0.5998174  1.0125851  0.2667975
        15         16         17
 0.8033336  0.2547373  0.8033336
```

```
> newdata <- with(modell, data.frame(x = 66))
> newdata
   x
1 66
> newdata$fit <- predict(modell, newdata = newdata, type = "response")
> newdata
   x       fit
1 66 0.9999916
>
```

```
> with(modell, null.deviance - deviance)
[1] 10.95793
> with(modell, df.null - df.residual)
[1] 1
> with(modell, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))
[1] 0.0009320384
> logLik(modell)
'log Lik.' -6.275108 (df=2)
> 
```

```
> predict <- predict(modell, type="response") # predicted values
> table(modell$y,predict>0.5)

    FALSE TRUE
  0     6    2
  1     1    8
> (6+8)/(6+2+1+8)
[1] 0.8235294
> 
```