Progetto

Andrea Corvaglia

17 agosto 2019

TO DO:

> La Gpu con 112 livelli è inutilizzabile, possiamo dividere in integrata e dedicata, oppure dividere per le marche.
>
> > @Dario: Per la gpu farei innanzitutto una divisione tra integrata e dedicata, e tra quelle dedicate le suddividerei per VRAM (inizialmente pensavo di cercare dei prezzi per avere una stima quantitativa di quanto valga la gpu dedicata, però bisognerebbe cercare prezzi medi e visto che i modelli son tanti mi pare troppo lavoro e non chissà quanto significativo)
>
> Anche la memoria è problematica, ho già estratto la presenza o meno dell'SSD, ma bisogna pensare ad un modo di valutare la memoria.
>
> > @Dario: ottimo la divisione SSD/HD, per avere un indicazione quantitativa potremmo cercare il prezzo di un gb di SSD e di un gb di HD da una fonte affidabile, e pesare l'importanza delle dimensioni su questo indicatore, che ne dite?

```r
data <- read.csv("../data/Laptop2.csv")
str(data)
```

```
## 'data.frame':    1303 obs. of  17 variables:
##  $ X               : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Company         : Factor w/ 19 levels "Acer","Apple",..: 2 2 8 2 2 1 2 2 3 1 ...
##  $ Product         : Factor w/ 618 levels "110-15ACL (A6-7310/4GB/500GB/W10)",..: 302 300 51 302 302
##  $ TypeName        : Factor w/ 6 levels "2 in 1 Convertible",..: 5 5 4 5 5 4 5 5 5 5 ...
##  $ Inches          : num  13.3 13.3 15.6 15.4 13.3 15.6 15.4 13.3 14 14 ...
##  $ ScreenResolution: Factor w/ 40 levels "1366x768","1440x900",..: 24 2 9 26 24 1 26 2 9 16 ...
##  $ Cpu             : Factor w/ 118 levels "AMD A10-Series 9600P 2.4GHz",..: 55 53 64 75 57 15 74 53 9
##  $ Ram             : int  8 8 8 16 8 4 16 8 16 8 ...
##  $ Memory          : Factor w/ 39 levels "1.0TB HDD","1.0TB Hybrid",..: 5 3 17 30 17 27 16 16 30 17
##  $ Gpu             : Factor w/ 110 levels "AMD FirePro W4190M",..: 59 52 54 10 60 18 61 52 98 62 ...
##  $ OpSys           : Factor w/ 9 levels "Android","Chrome OS",..: 5 5 6 5 5 7 4 5 7 7 ...
##  $ Weight          : num  1.37 1.34 1.86 1.83 1.37 2.1 2.04 1.34 1.3 1.6 ...
##  $ Price           : num  1340 899 575 2537 1804 ...
##  $ Frequenza       : num  2.3 1.8 2.5 2.7 3.1 3 2.2 1.8 1.8 1.6 ...
##  $ Risoluzione     : Factor w/ 15 levels "1366x768","1440x900",..: 11 2 4 13 11 1 13 2 4 4 ...
##  $ Pixel           : int  4096000 1296000 2073600 5184000 4096000 1049088 5184000 1296000 2073600 207
##  $ SolidStateDisk  : Factor w/ 2 levels "False","True": 2 1 2 2 2 1 1 1 2 2 ...
```

```r
head(data)
```

```
##   X Company     Product  TypeName Inches
## 1 1   Apple MacBook Pro Ultrabook   13.3
## 2 2   Apple Macbook Air Ultrabook   13.3
## 3 3      HP      250 G6  Notebook   15.6
## 4 4   Apple MacBook Pro Ultrabook   15.4
## 5 5   Apple MacBook Pro Ultrabook   13.3
## 6 6    Acer    Aspire 3  Notebook   15.6
##                        ScreenResolution                            Cpu Ram
## 1 IPS Panel Retina Display 2560x1600       Intel Core i5 2.3GHz   8
## 2                           1440x900       Intel Core i5 1.8GHz   8
## 3                  Full HD 1920x1080 Intel Core i5 7200U 2.5GHz   8
## 4 IPS Panel Retina Display 2880x1800       Intel Core i7 2.7GHz  16
## 5 IPS Panel Retina Display 2560x1600       Intel Core i5 3.1GHz   8
## 6                           1366x768    AMD A9-Series 9420 3GHz   4
##                Memory                          Gpu      OpSys Weight
## 1           128GB SSD Intel Iris Plus Graphics 640      macOS   1.37
## 2 128GB Flash Storage       Intel HD Graphics 6000      macOS   1.34
## 3           256GB SSD        Intel HD Graphics 620      No OS   1.86
## 4           512GB SSD           AMD Radeon Pro 455      macOS   1.83
## 5           256GB SSD Intel Iris Plus Graphics 650      macOS   1.37
## 6           500GB HDD                AMD Radeon R5 Windows 10   2.10
##     Price Frequenza Risoluzione   Pixel SolidStateDisk
## 1 1339.69       2.3   2560x1600 4096000           True
## 2  898.94       1.8    1440x900 1296000          False
## 3  575.00       2.5   1920x1080 2073600           True
## 4 2537.45       2.7   2880x1800 5184000           True
## 5 1803.60       3.1   2560x1600 4096000           True
## 6  400.00       3.0    1366x768 1049088          False
```

```
summary(data)
```

```
##        X             Company                  Product
##  Min.   :   1.0   Dell   :297   XPS 13            :  30
##  1st Qu.: 331.5   Lenovo :297   Inspiron 3567     :  29
##  Median : 659.0   HP     :274   250 G6            :  21
##  Mean   : 660.2   Asus   :158   Legion Y520-15IKBN:  19
##  3rd Qu.: 990.5   Acer   :103   Vostro 3568       :  19
##  Max.   :1320.0   MSI    : 54   Inspiron 5570     :  18
##                   (Other):120   (Other)           :1167
##                 TypeName       Inches
##  2 in 1 Convertible:121   Min.   :10.10
##  Gaming            :205   1st Qu.:14.00
##  Netbook           : 25   Median :15.60
##  Notebook          :727   Mean   :15.02
##  Ultrabook         :196   3rd Qu.:15.60
##  Workstation       : 29   Max.   :18.40
##
##                                   ScreenResolution
##  Full HD 1920x1080                        :507
##  1366x768                                 :281
##  IPS Panel Full HD 1920x1080              :230
##  IPS Panel Full HD / Touchscreen 1920x1080: 53
##  Full HD / Touchscreen 1920x1080          : 47
##  1600x900                                 : 23
##  (Other)                                  :162
##                          Cpu           Ram
##  Intel Core i5 7200U 2.5GHz :190   Min.   : 2.000
##  Intel Core i7 7700HQ 2.8GHz:146   1st Qu.: 4.000
##  Intel Core i7 7500U 2.7GHz :134   Median : 8.000
##  Intel Core i7 8550U 1.8GHz : 73   Mean   : 8.382
##  Intel Core i5 8250U 1.6GHz : 72   3rd Qu.: 8.000
##  Intel Core i5 6200U 2.3GHz : 68   Max.   :64.000
##  (Other)                    :620
##                   Memory                          Gpu
##  256GB SSD           :412   Intel HD Graphics 620  :281
##  1TB HDD             :223   Intel HD Graphics 520  :185
##  500GB HDD           :132   Intel UHD Graphics 620 : 68
##  512GB SSD           :118   Nvidia GeForce GTX 1050: 66
##  128GB SSD +  1TB HDD: 94   Nvidia GeForce GTX 1060: 48
##  128GB SSD           : 76   Nvidia GeForce 940MX   : 43
##  (Other)             :248   (Other)                :612
##         OpSys          Weight          Price        Frequenza
##  Windows 10:1072   Min.   :0.690   Min.   : 174   Min.   :0.900
##  No OS     :  66   1st Qu.:1.500   1st Qu.: 599   1st Qu.:2.000
##  Linux     :  62   Median :2.040   Median : 977   Median :2.500
##  Windows 7 :  45   Mean   :2.039   Mean   :1124   Mean   :2.299
##  Chrome OS :  27   3rd Qu.:2.300   3rd Qu.:1488   3rd Qu.:2.700
##  macOS     :  13   Max.   :4.700   Max.   :6099   Max.   :3.600
##  (Other)   :  18
##     Risoluzione      Pixel         SolidStateDisk
##  1920x1080:841   Min.   :1049088   False:460
##  1366x768 :308   1st Qu.:1440000   True :843
##  3840x2160: 43   Median :2073600
##  3200x1800: 27   Mean   :2168807
##  1600x900 : 23   3rd Qu.:2073600
##  2560x1440: 23   Max.   :8294400
##  (Other)  : 38
```

```
nums <- sapply(data, is.numeric)
var_numeric <- data[,nums]
head(var_numeric)
```

```
##   X Inches Ram Weight   Price Frequenza   Pixel
## 1 1   13.3   8   1.37 1339.69       2.3 4096000
## 2 2   13.3   8   1.34  898.94       1.8 1296000
## 3 3   15.6   8   1.86  575.00       2.5 2073600
## 4 4   15.4  16   1.83 2537.45       2.7 5184000
## 5 5   13.3   8   1.37 1803.60       3.1 4096000
## 6 6   15.6   4   2.10  400.00       3.0 1049088
```

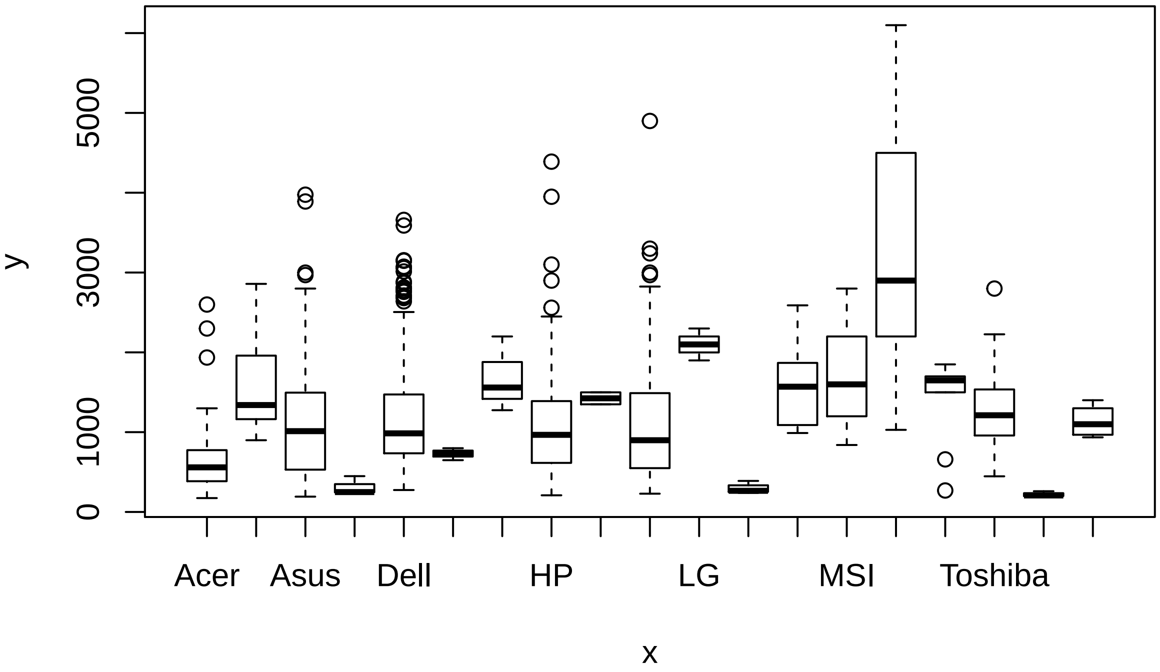
```
data$Weight<-as.numeric(data$Weight)
data$Ram<-as.numeric(data$Ram)
```

```
sapply(data, function(x)(sum(is.na(x))))
```

```
##                X          Company          Product         TypeName
##                0                0                0                0
##           Inches ScreenResolution              Cpu              Ram
##                0                0                0                0
##           Memory              Gpu            OpSys           Weight
##                0                0                0                0
##            Price        Frequenza      Risoluzione            Pixel
##                0                0                0                0
##   SolidStateDisk
##                0
```

```
# Non ci sono missing data!
```

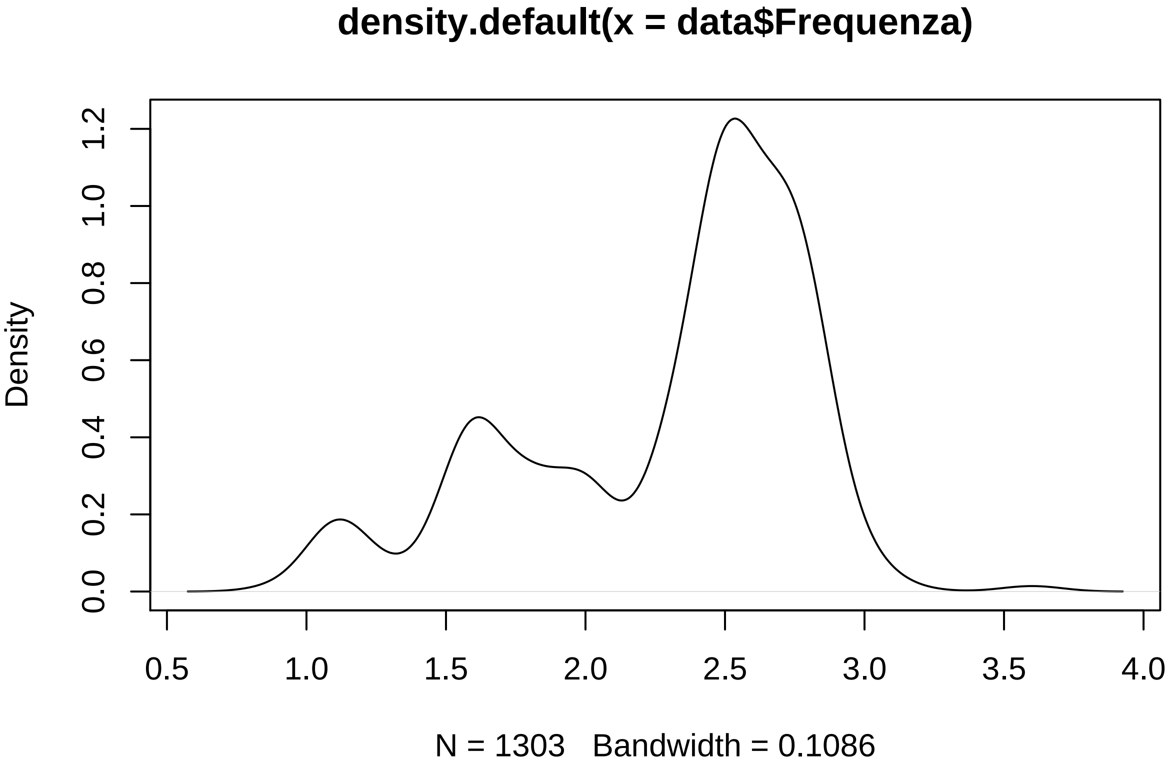
```
plot(data$Company,data$Price)
```



```r
class(data$Ram)
```

```
## [1] "numeric"
```

```r
plot(density(data$Frequenza))
```

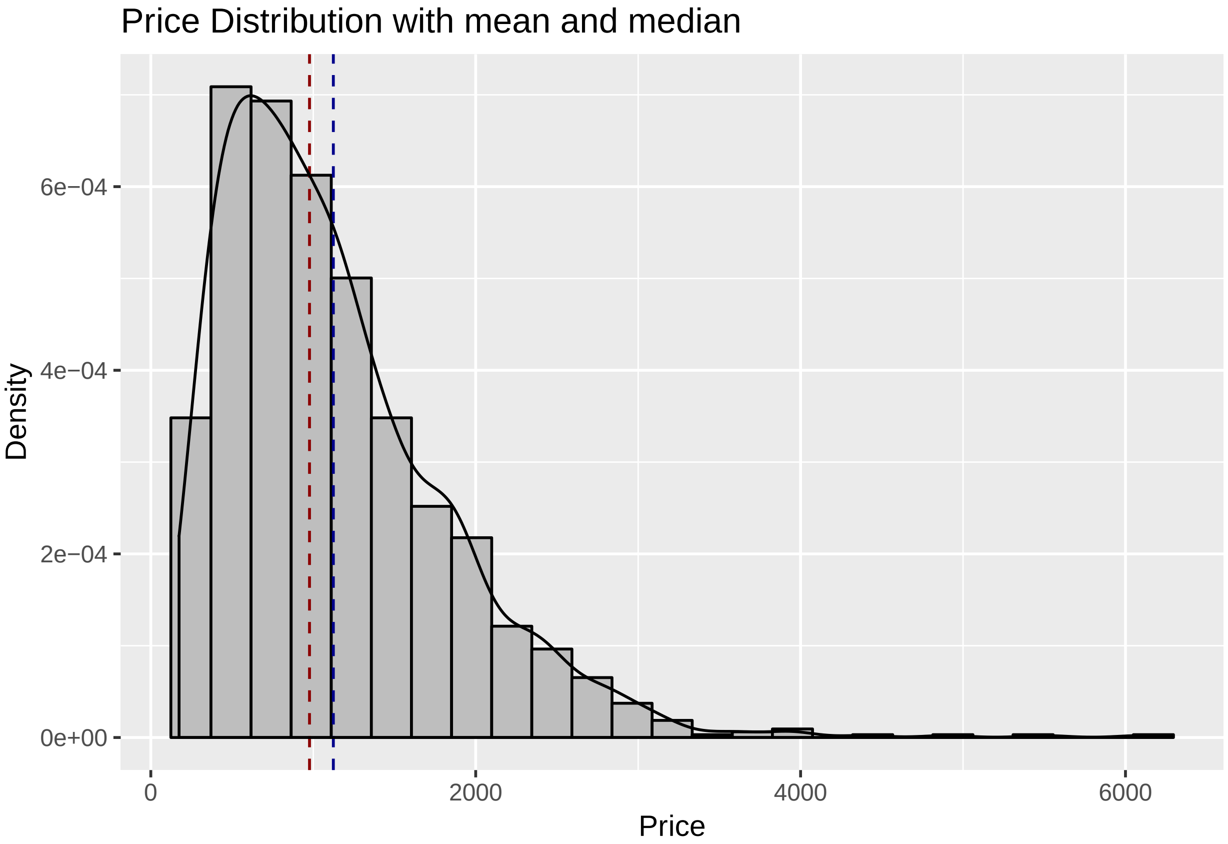


```r
#hist(data$Price, breaks=25, probability=TRUE)
#lines(density(data$Price))
```

```r
library(ggplot2)
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

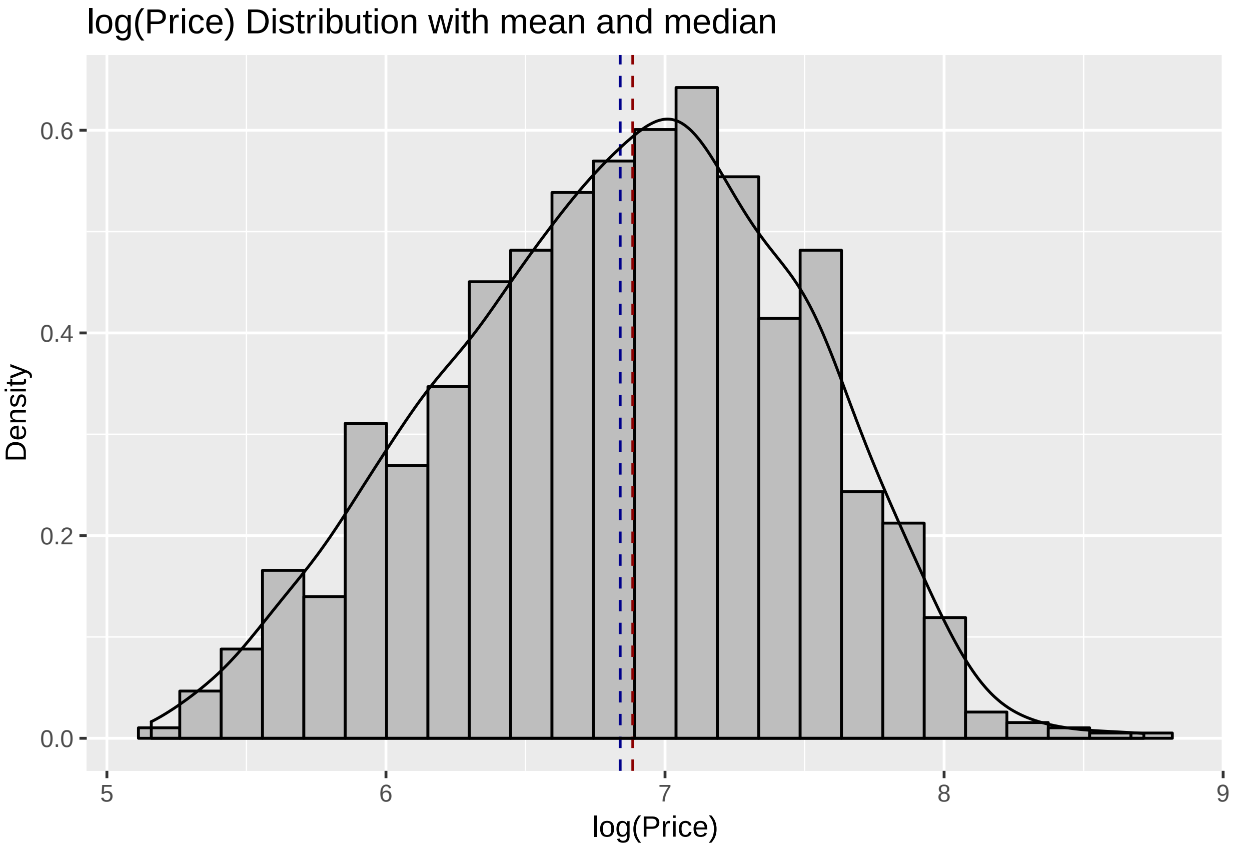
```r
ggplot(data,aes(x = Price)) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = quantile(data$Price, 0.50), color = "dark red", lty = 2) +
        geom_vline(xintercept = mean(data$Price), color = "dark blue", lty = 2) +
        labs(x = "Price", y ="Density") +
        ggtitle("Price Distribution with mean and median") +
          geom_density()
```



Quite skewed to the right, mean > media

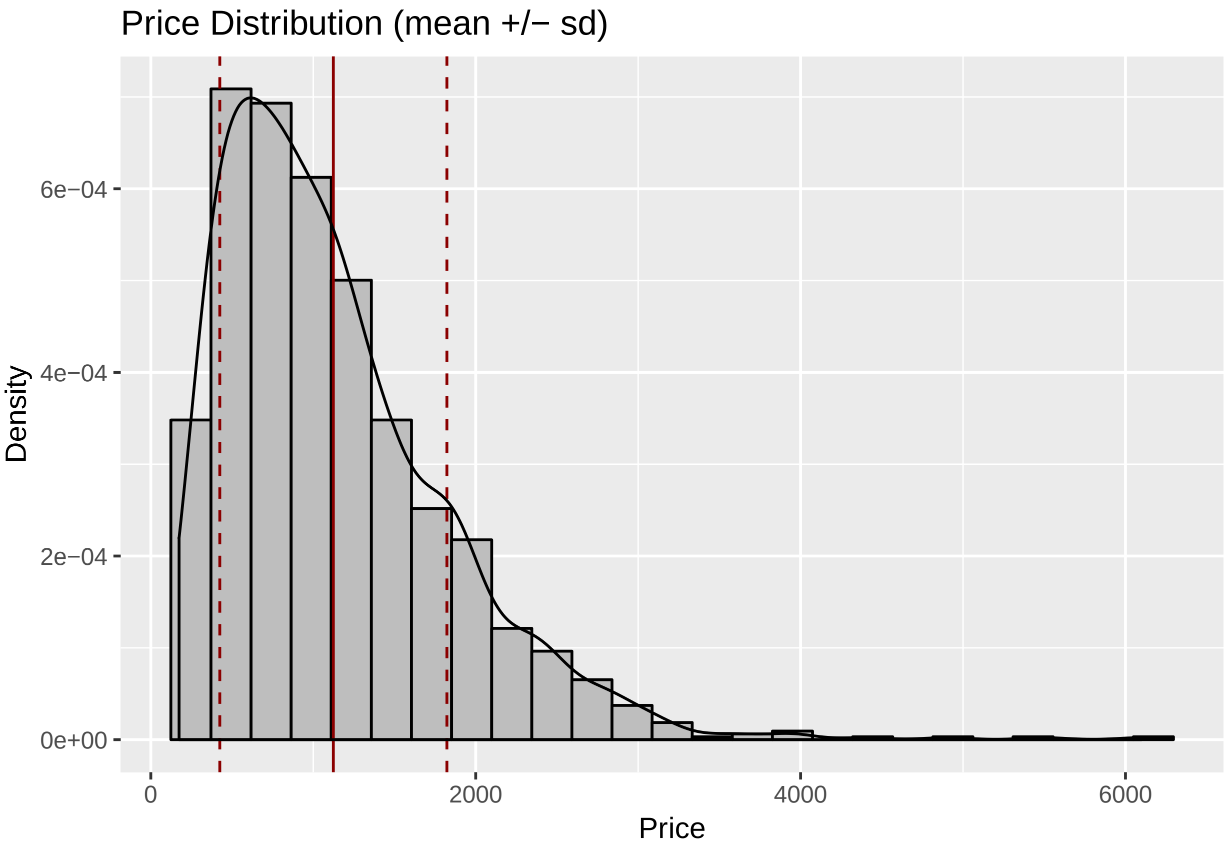
We could try to apply a correction like Log(Y)

```r
data$LogPrice=log(data$Price)
ggplot(data,aes(x = log(Price))) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = quantile(data$LogPrice, 0.50), color = "dark red", lty = 2) +
        geom_vline(xintercept = mean(data$LogPrice), color = "dark blue", lty = 2) +
        labs(x = "log(Price)", y ="Density") +
        ggtitle("log(Price) Distribution with mean and median")+  geom_density()
```

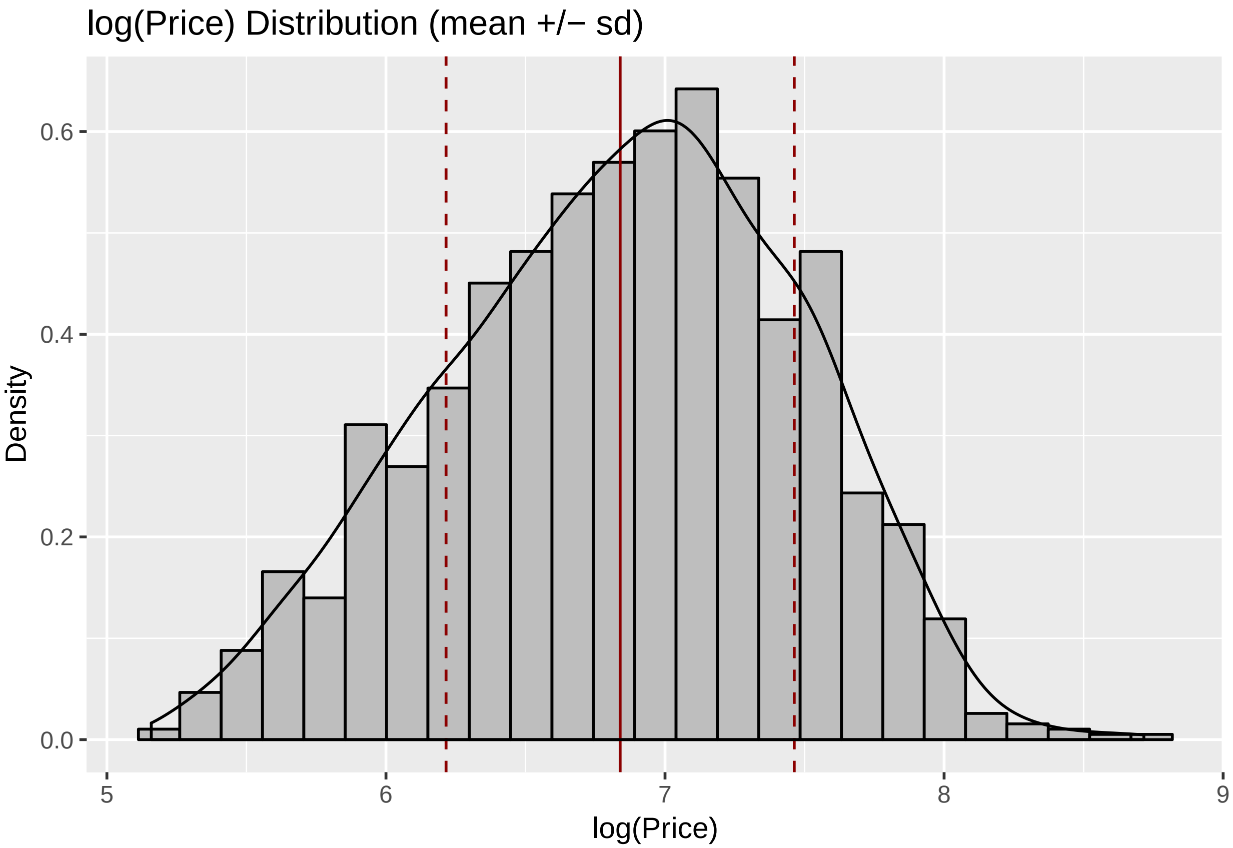


Now the distribution is looking a bit better (as regards normality)

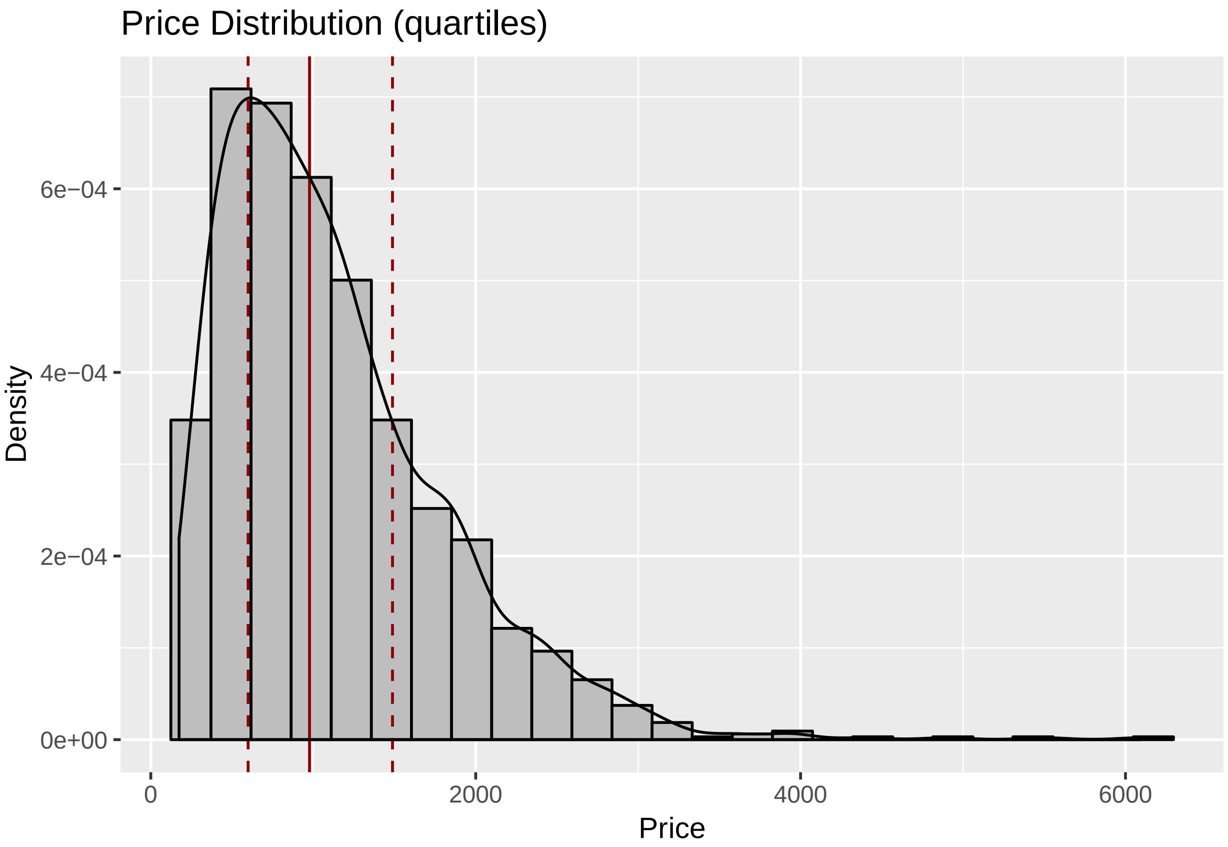
```
ggplot(data,aes(x = Price)) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = mean(data$Price), color = "dark red") +
        geom_vline(xintercept = mean(data$Price) + sd(data$Price), color = "dark red", lty = 2) +
        geom_vline(xintercept = mean(data$Price) - sd(data$Price), color = "dark red", lty = 2) +
        labs(x = "Price", y ="Density") +
        ggtitle("Price Distribution (mean +/- sd)") +
          geom_density()
```



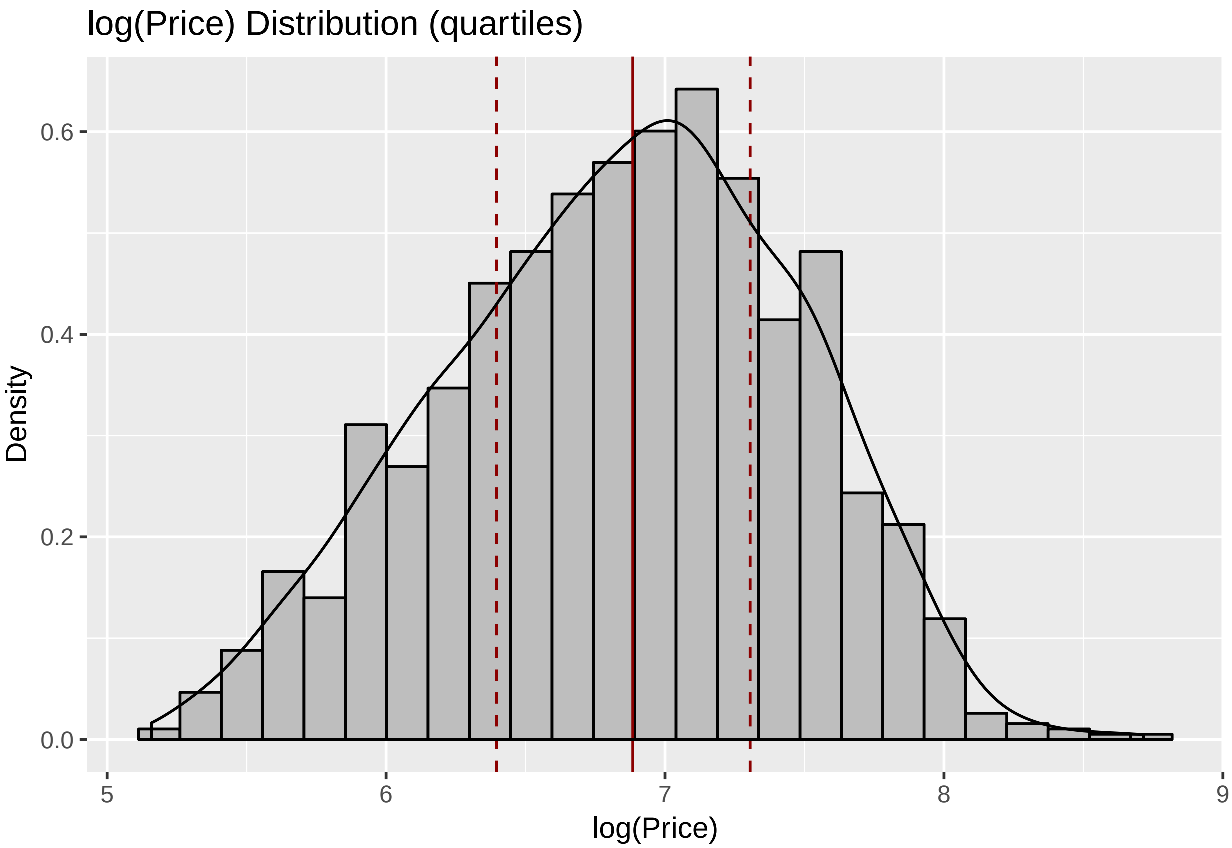
```
ggplot(data,aes(x = log(Price))) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = mean(data$LogPrice), color = "dark red") +
        geom_vline(xintercept = mean(data$LogPrice) + sd(data$LogPrice), color = "dark red", lty = 2) +
        geom_vline(xintercept = mean(data$LogPrice) - sd(data$LogPrice), color = "dark red", lty = 2) +
        labs(x = "log(Price)", y ="Density") +
        ggtitle("log(Price) Distribution (mean +/- sd)") +
          geom_density()
```



```
ggplot(data,aes(x = Price)) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = quantile(data$Price, 0.25), color = "dark red",lty = 2) +
        geom_vline(xintercept = quantile(data$Price, 0.5), color = "dark red", ) +
        geom_vline(xintercept = quantile(data$Price, 0.75), color = "dark red", lty = 2) +
        labs(x = "Price", y ="Density") +
        ggtitle("Price Distribution (quartiles)") +
          geom_density()
```

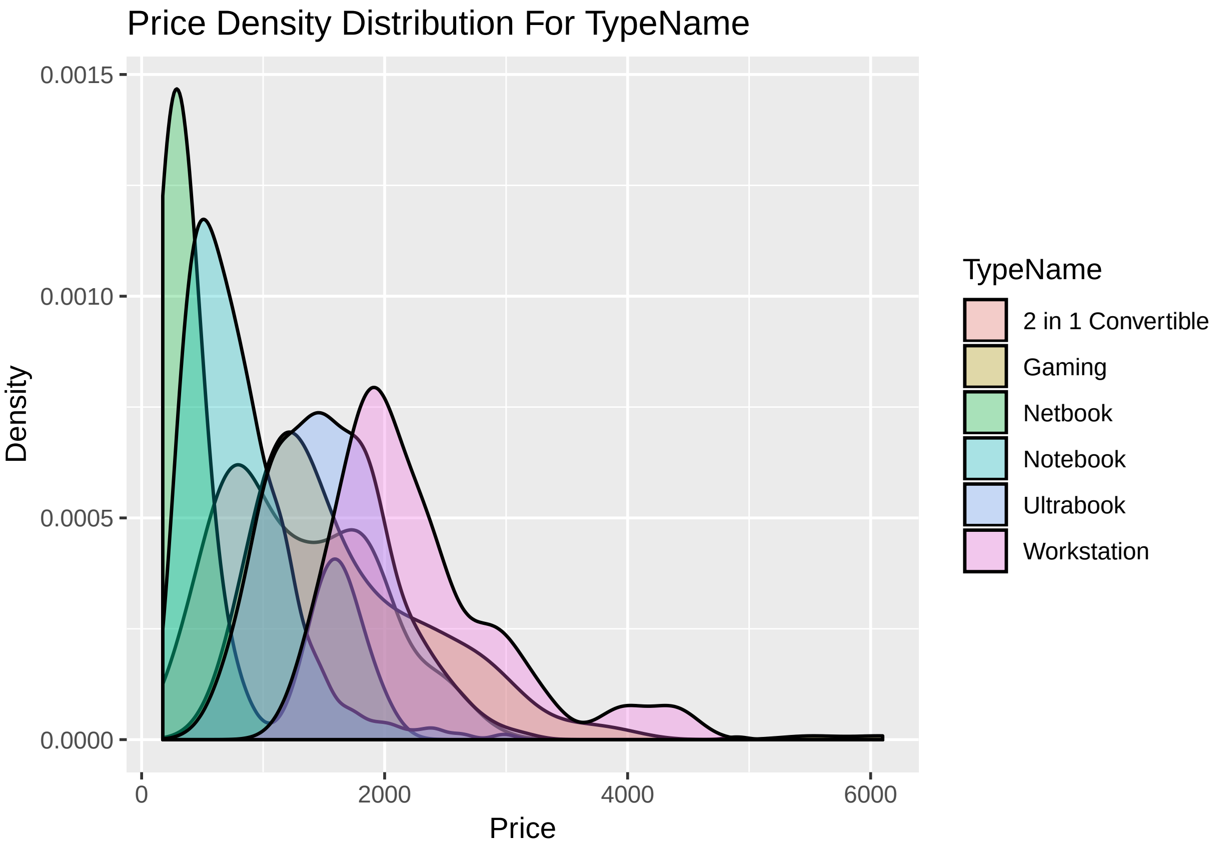


```
ggplot(data,aes(x = log(Price))) +
        geom_histogram(aes(y =..density..),
                       bins= 25,
                       fill = "grey",
                       color ="black") +
        geom_vline(xintercept = quantile(data$LogPrice, 0.25), color = "dark red",lty = 2) +
        geom_vline(xintercept = quantile(data$LogPrice, 0.5), color = "dark red", ) +
        geom_vline(xintercept = quantile(data$LogPrice, 0.75), color = "dark red", lty = 2) +
           labs(x = "log(Price)", y ="Density") +
        ggtitle("log(Price) Distribution (quartiles)") +
          geom_density()
```

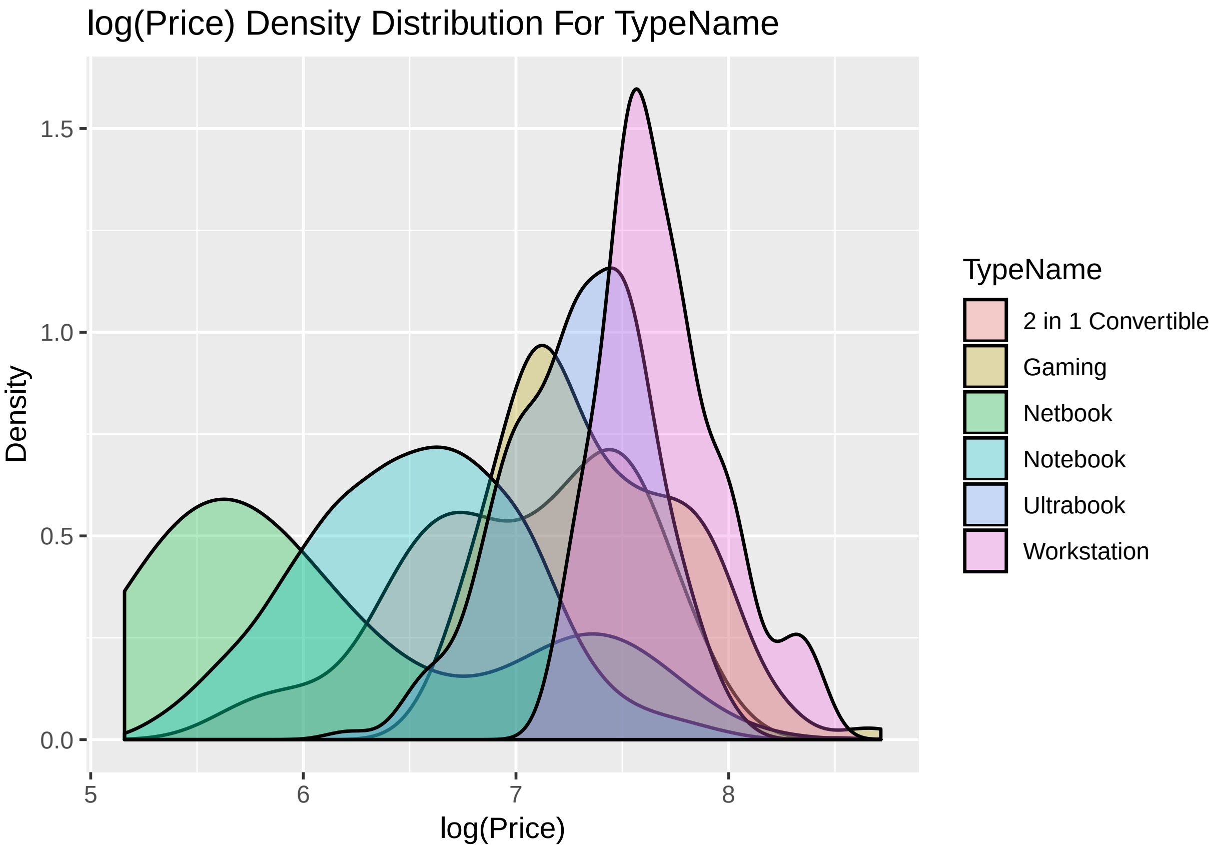


Descrittive variabile dipendente price

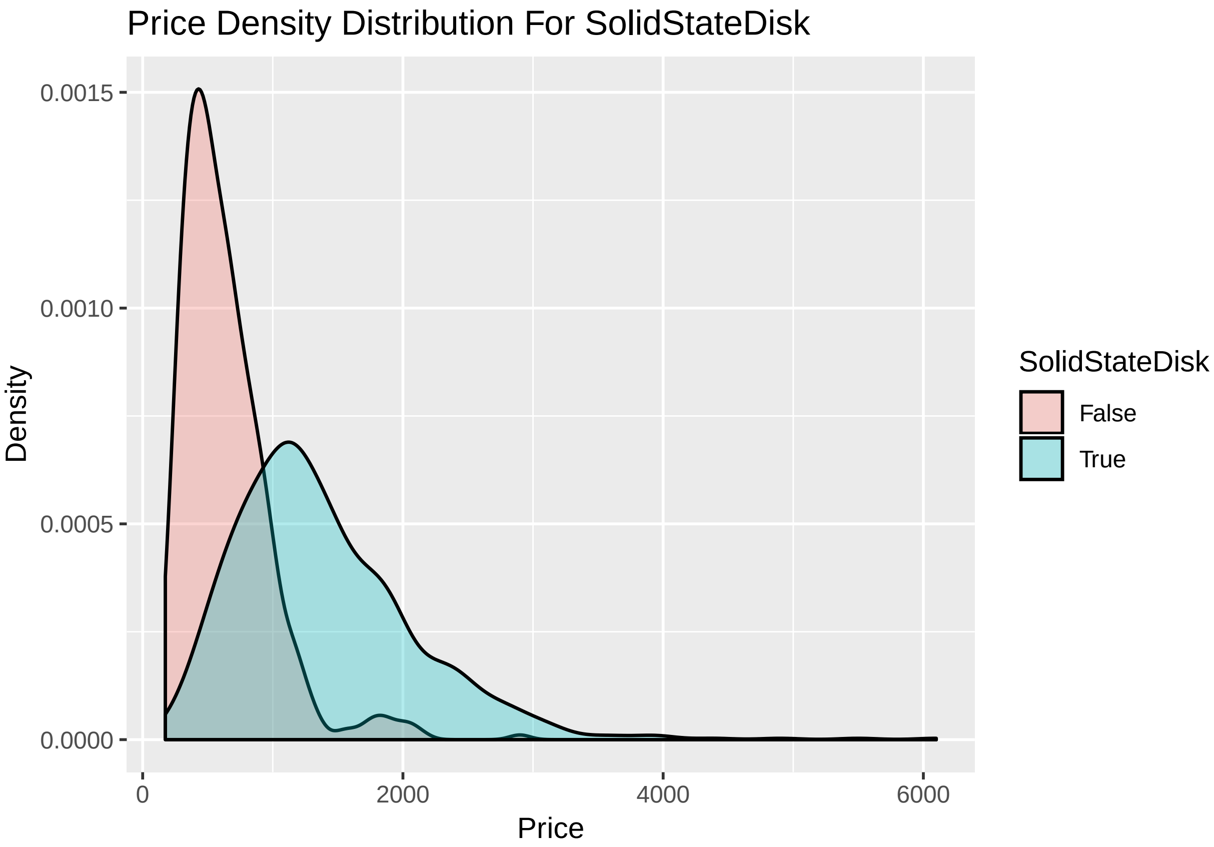
```
ggplot(data, aes(x = Price, fill = TypeName)) +
        geom_density(size = 0.6, alpha = .3) +
        labs(x = "Price", y ="Density", fill = "TypeName") +
        ggtitle("Price Density Distribution For TypeName")
```



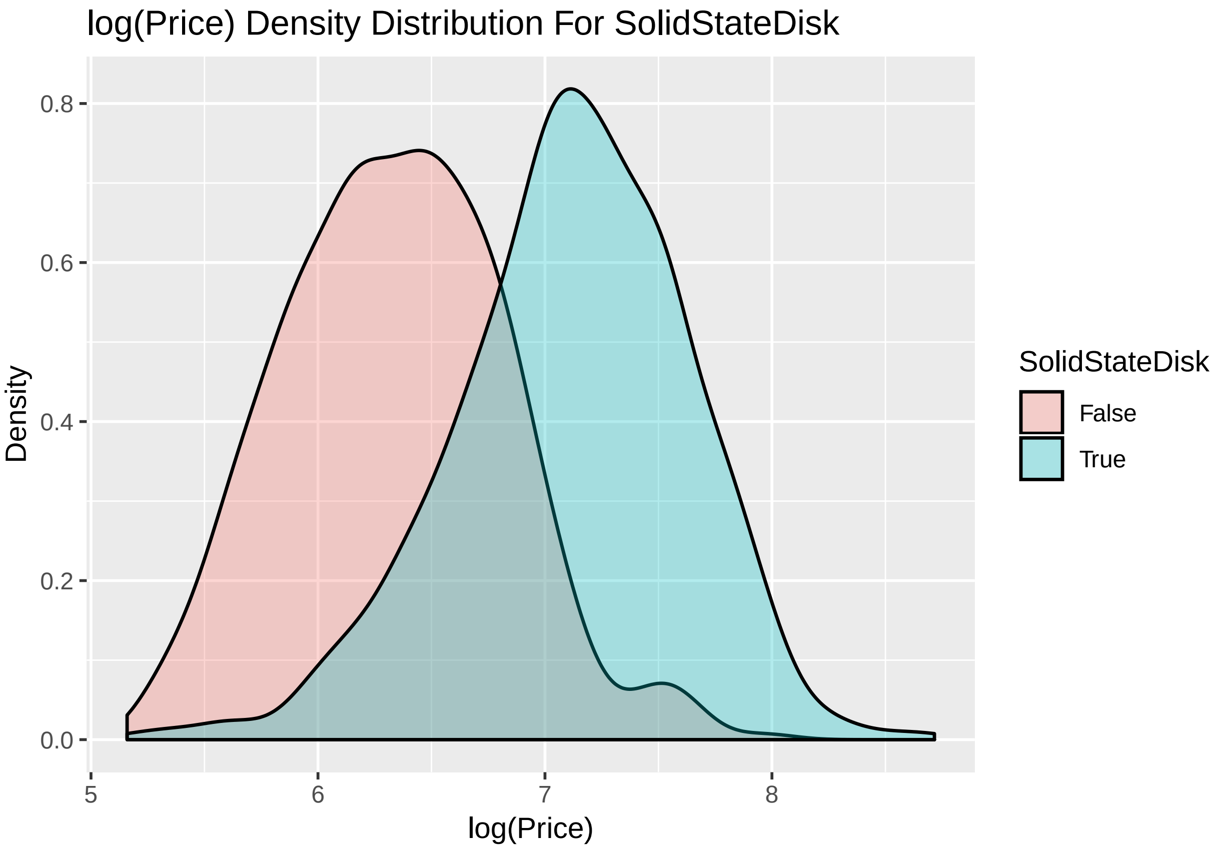
```
ggplot(data, aes(x = log(Price), fill = TypeName)) +
        geom_density(size = 0.6, alpha = .3) +
        labs(x = "log(Price)", y ="Density", fill = "TypeName") +
        ggtitle("log(Price) Density Distribution For TypeName")
```



```
ggplot(data, aes(x = Price, fill = SolidStateDisk)) +
        geom_density(size = 0.6, alpha = .3) +
        labs(x = "Price", y ="Density", fill = "SolidStateDisk") +
        ggtitle("Price Density Distribution For SolidStateDisk")
```



```
ggplot(data, aes(x = log(Price), fill = SolidStateDisk)) +
        geom_density(size = 0.6, alpha = .3) +
        labs(x = "log(Price)", y ="Density", fill = "SolidStateDisk") +
        ggtitle("log(Price) Density Distribution For SolidStateDisk")
```



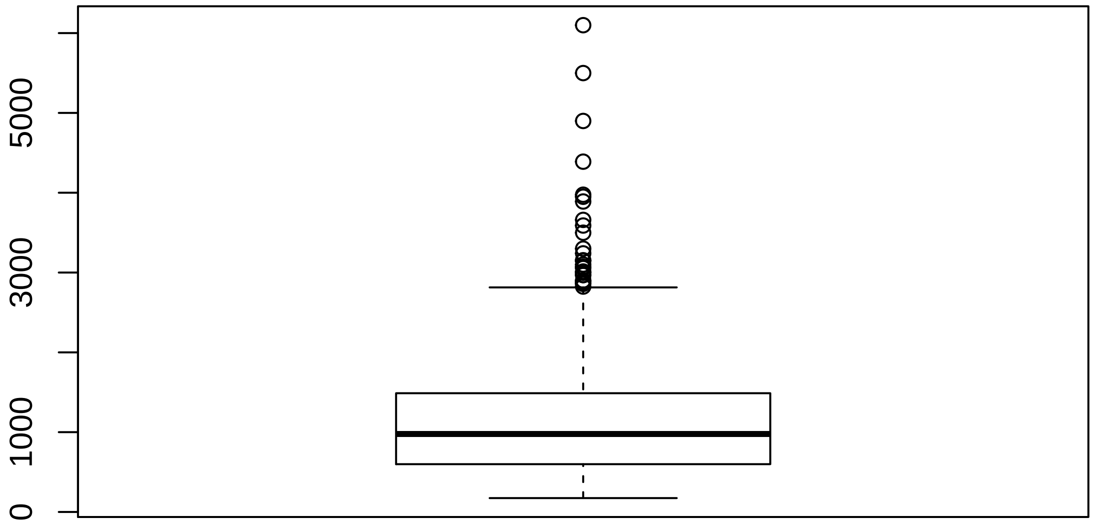
```
library(psych)
```

```
##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

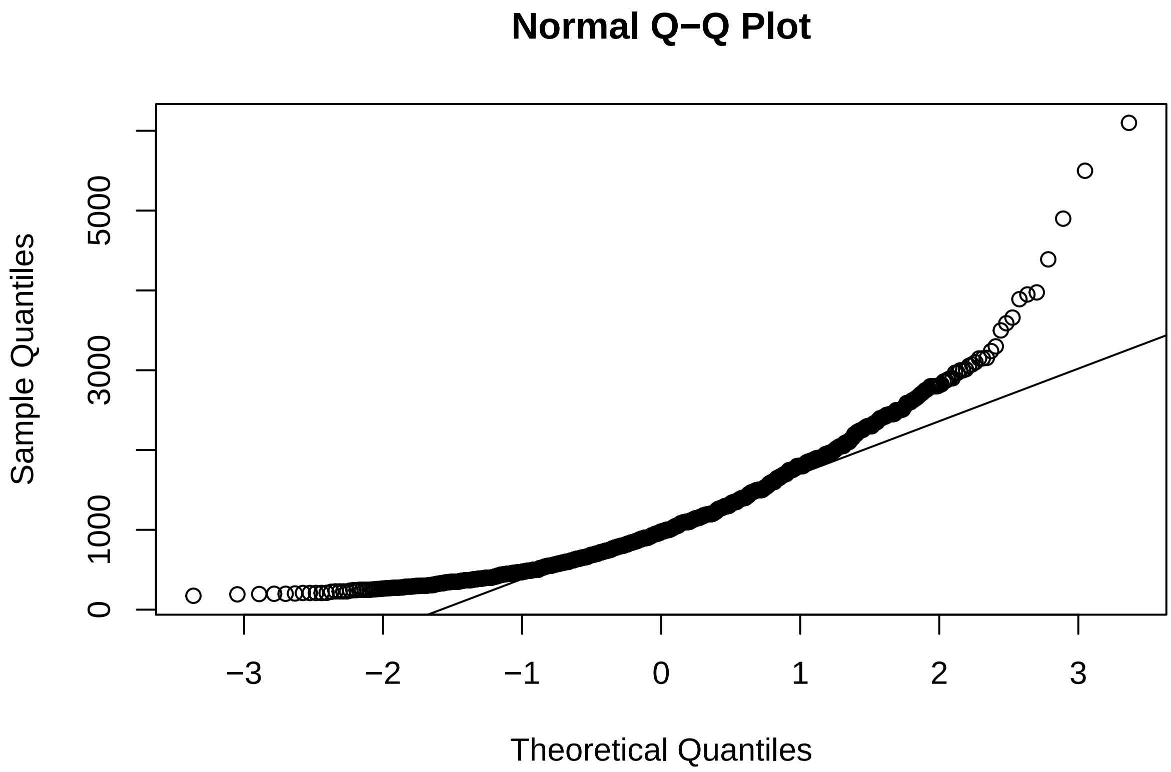
```
describe(data$Price)
```

```
##    vars    n    mean     sd median trimmed    mad min  max range skew
## X1    1 1303 1123.69 699.01    977 1038.47 619.73 174 6099  5925 1.52
##    kurtosis    se
## X1     4.34 19.36
```

```
library(nortest)
# NORMALITA'

boxplot(data$Price)
```



```
qqnorm(data$Price);qqline(data$Price)
```



```
shapiro.test(data$Price)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$Price
## W = 0.89382, p-value < 2.2e-16
```

```
ad.test(data$Price)
```

```
##
##  Anderson-Darling normality test
```

```
##
## data:  data$Price
## A = 28.319, p-value < 2.2e-16
```

```
#wilcox.test(data$Price, conf.int = TRUE, mu = ) #worth it?
#if(!require(Envstats)) install.packages("EnvStats")
library(EnvStats)
```

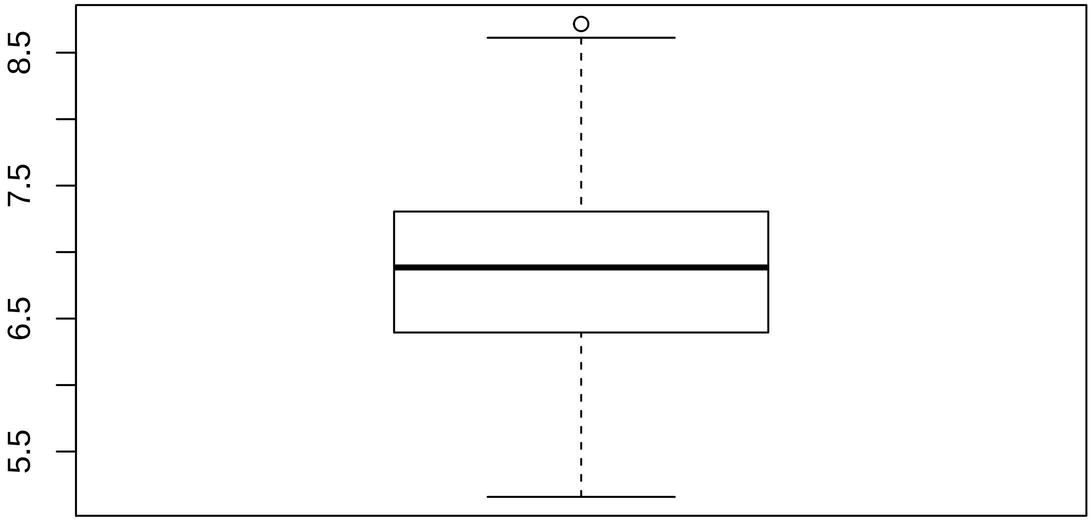
```
##
## Attaching package: 'EnvStats'

## The following objects are masked from 'package:stats':
##
##     predict, predict.lm

## The following object is masked from 'package:base':
##
##     print.default
```

```
varTest(sample(data$Price), sigma.squared = (sd(data$Price)*sd(data$Price)))
```

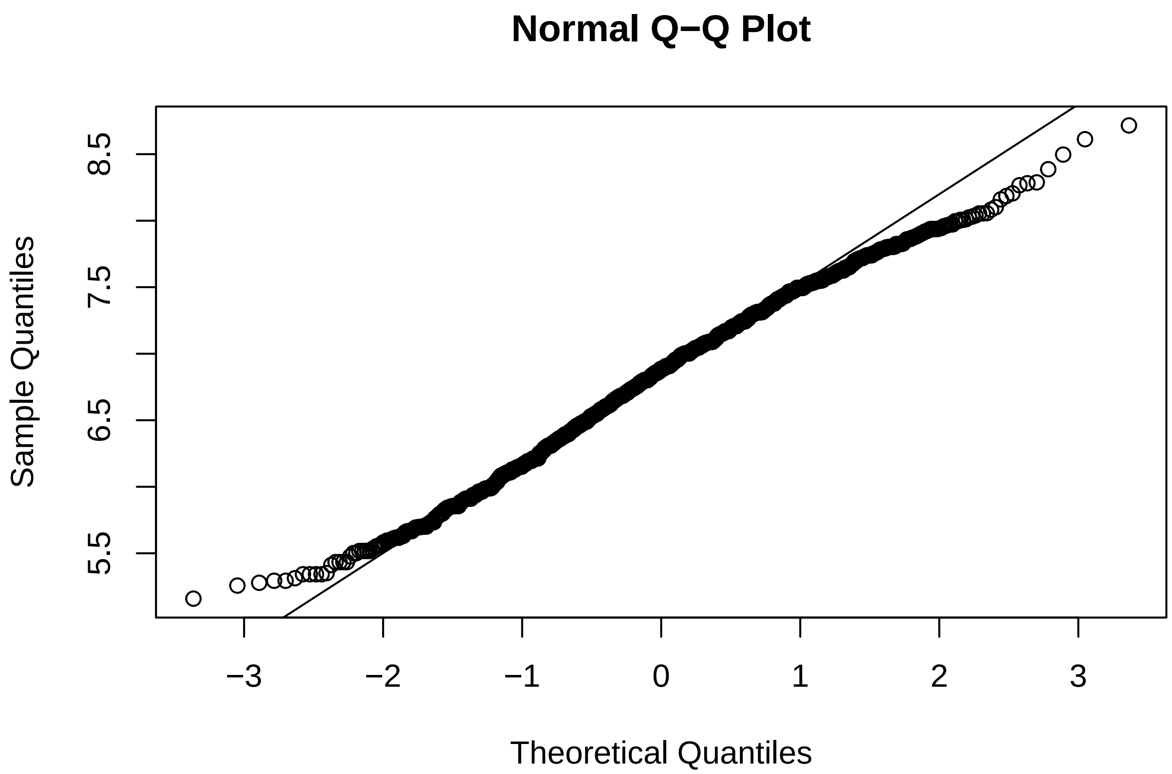
```
##
##  Chi-Squared Test on Variance
##
## data:  sample(data$Price)
## Chi-Squared = 1302, df = 1302, p-value = 0.9896
## alternative hypothesis: true variance is not equal to 488613.6
## 95 percent confidence interval:
##  453149.5 528432.0
## sample estimates:
## variance
## 488613.6
```

Trying with the log correction:

```
# Correzione NORMALITA'
library(nortest)
boxplot(data$LogPrice)
```



```
qqnorm(data$LogPrice);qqline(data$LogPrice)
```



```
shapiro.test(data$LogPrice) #better than before, but still not normal according to shapiro
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$LogPrice
## W = 0.99252, p-value = 3.628e-06
```

```
ad.test(data$LogPrice)
```

```
##
##  Anderson-Darling normality test
##
## data:  data$LogPrice
## A = 2.5942, p-value = 1.515e-06
```

T-test

```
# One sample
ref <- mean(data$Price)
Apple<-data$Price[data$Company=="Apple"]
t.test(Apple,mu=ref,alternative = "greater")
```

```
##
##  One Sample t-test
##
## data:  Apple
## t = 3.5944, df = 20, p-value = 0.000906
## alternative hypothesis: true mean is greater than 1123.687
## 95 percent confidence interval:
##  1352.823      Inf
## sample estimates:
```

```
## mean of x
##  1564.199
```

```r
# Wilcoxon Signed Rank Test
wilcox.test(Apple, mu=ref, conf.int = TRUE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  Apple
## V = 206, p-value = 0.0008516
## alternative hypothesis: true location is not equal to 1123.687
## 95 percent confidence interval:
##  1234.50 1829.26
## sample estimates:
## (pseudo)median
##       1514.275
```

```r
#Two sample
Other <-data$Price[data$Company!="Apple"]
wilcox.test(Apple, Other, alternative = "g")
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Apple and Other
## W = 19689, p-value = 0.0001358
## alternative hypothesis: true location shift is greater than 0
```

```r
# F test sulla varianza
var.test(Apple, Other, alternative = "two.sided")
```

```
##
##  F test to compare two variances
##
## data:  Apple and Other
## F = 0.64574, num df = 20, denom df = 1281, p-value = 0.2401
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.3755878 1.3509884
## sample estimates:
## ratio of variances
##          0.6457382
```

Variabili qualitative: tabella di contingenza e chi quadro

```r
b<-data
b.table<-table(b$SolidStateDisk,b$TypeName)
b.table
```

```
##
##         2 in 1 Convertible Gaming Netbook Notebook Ultrabook Workstation
##   False                 26     27      13      372        16           6
##   True                  95    178      12      355       180          23
```

```r
prop.table(b.table,2)
```

```
##
##         2 in 1 Convertible     Gaming    Netbook   Notebook  Ultrabook
##   False         0.21487603 0.13170732 0.52000000 0.51169188 0.08163265
##   True          0.78512397 0.86829268 0.48000000 0.48830812 0.91836735
##
##         Workstation
##   False  0.20689655
##   True   0.79310345
```

```
# chi square test

chisq.test(b.table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  b.table
## X-squared = 203.18, df = 5, p-value < 2.2e-16
```

```
chi=chisq.test(b.table)
chi_norm=chi$statistic/(nrow(b)*min(nrow(b.table)-1,ncol(b.table)-1))
chi_norm
```

```
## X-squared
## 0.1559288
```

```
summary(b.table)
```

```
## Number of cases in table: 1303
## Number of factors: 2
## Test for independence of all factors:
##  Chisq = 203.18, df = 5, p-value = 5.944e-42
```

Correlazione per variabili quantitative

```
# seleziona solo variabili quantitative
nums <- sapply(data, is.numeric)
var_numeric <- data[,nums]
head(var_numeric)
```

```
##   X Inches Ram Weight   Price Frequenza   Pixel LogPrice
## 1 1   13.3   8   1.37 1339.69       2.3 4096000 7.200194
## 2 2   13.3   8   1.34  898.94       1.8 1296000 6.801216
## 3 3   15.6   8   1.86  575.00       2.5 2073600 6.354370
## 4 4   15.4  16   1.83 2537.45       2.7 5184000 7.838915
## 5 5   13.3   8   1.37 1803.60       3.1 4096000 7.497540
## 6 6   15.6   4   2.10  400.00       3.0 1049088 5.991465
```

```
var_numeric$X=NULL
```

```
# Matrice di correlazione
R<-cor(var_numeric)
R
```

```
##                Inches       Ram      Weight      Price Frequenza
## Inches     1.00000000 0.2379928  0.82763110 0.06819667 0.3078698
## Ram        0.23799280 1.0000000  0.38387409 0.74300714 0.3680005
## Weight     0.82763110 0.3838741  1.00000000 0.21036980 0.3204336
## Price      0.06819667 0.7430071  0.21036980 1.00000000 0.4302931
## Frequenza  0.30786980 0.3680005  0.32043359 0.43029310 1.0000000
## Pixel     -0.08639917 0.3963585 -0.04403379 0.51548639 0.1352935
## LogPrice   0.04432871 0.6848033  0.15167383 0.92758068 0.5041461
##                   Pixel   LogPrice
## Inches    -0.08639917 0.04432871
## Ram        0.39635848 0.68480333
## Weight    -0.04403379 0.15167383
## Price      0.51548639 0.92758068
## Frequenza  0.13529350 0.50414608
## Pixel      1.00000000 0.48490475
## LogPrice   0.48490475 1.00000000
```

```r
# Test di correlazione. (Spearsman's o Kendall tau)
cor.test(var_numeric$Inches, var_numeric$Weight)
```

```
##
##  Pearson's product-moment correlation
##
## data:  var_numeric$Inches and var_numeric$Weight
## t = 53.187, df = 1301, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8097181 0.8440031
## sample estimates:
##       cor
## 0.8276311
```

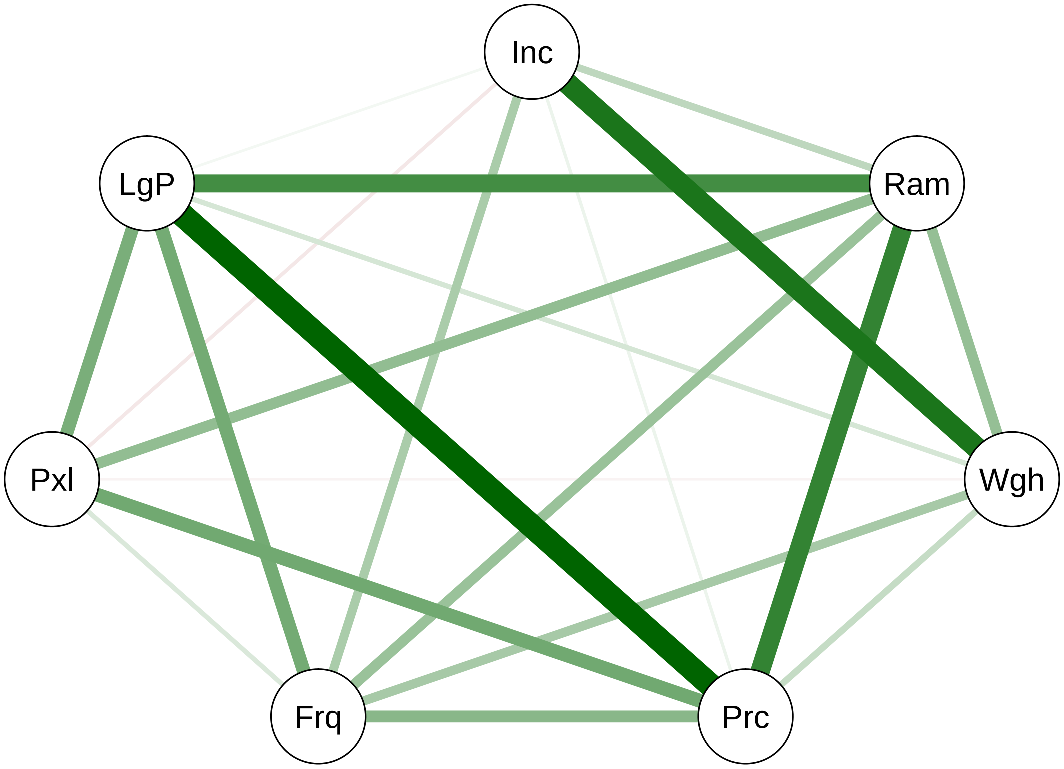
```r
#corrgram(var_numeric)

# Correlazione come grafo
library(qgraph)
```

```
## Registered S3 methods overwritten by 'huge':
##   method    from
##   plot.sim  BDgraph
##   print.sim BDgraph
```

```r
detcor=cor(as.matrix(var_numeric), method="pearson")
round(detcor, 2)
```

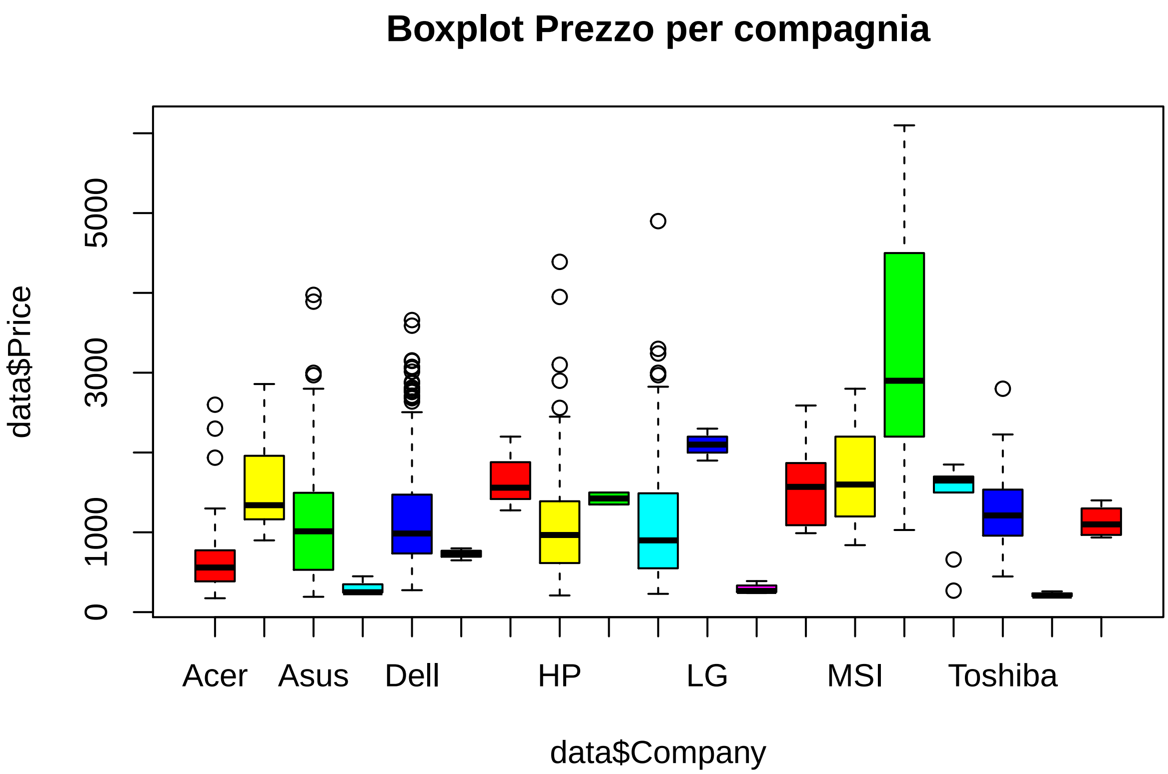
```
##           Inches  Ram Weight Price Frequenza Pixel LogPrice
## Inches      1.00 0.24   0.83  0.07      0.31 -0.09     0.04
## Ram         0.24 1.00   0.38  0.74      0.37  0.40     0.68
## Weight      0.83 0.38   1.00  0.21      0.32 -0.04     0.15
## Price       0.07 0.74   0.21  1.00      0.43  0.52     0.93
## Frequenza   0.31 0.37   0.32  0.43      1.00  0.14     0.50
## Pixel      -0.09 0.40  -0.04  0.52      0.14  1.00     0.48
## LogPrice    0.04 0.68   0.15  0.93      0.50  0.48     1.00
```

```r
# plot corr matrix: green positive red negative
qgraph(detcor, shape="circle", posCol="darkgreen", negCol="darkred")
```

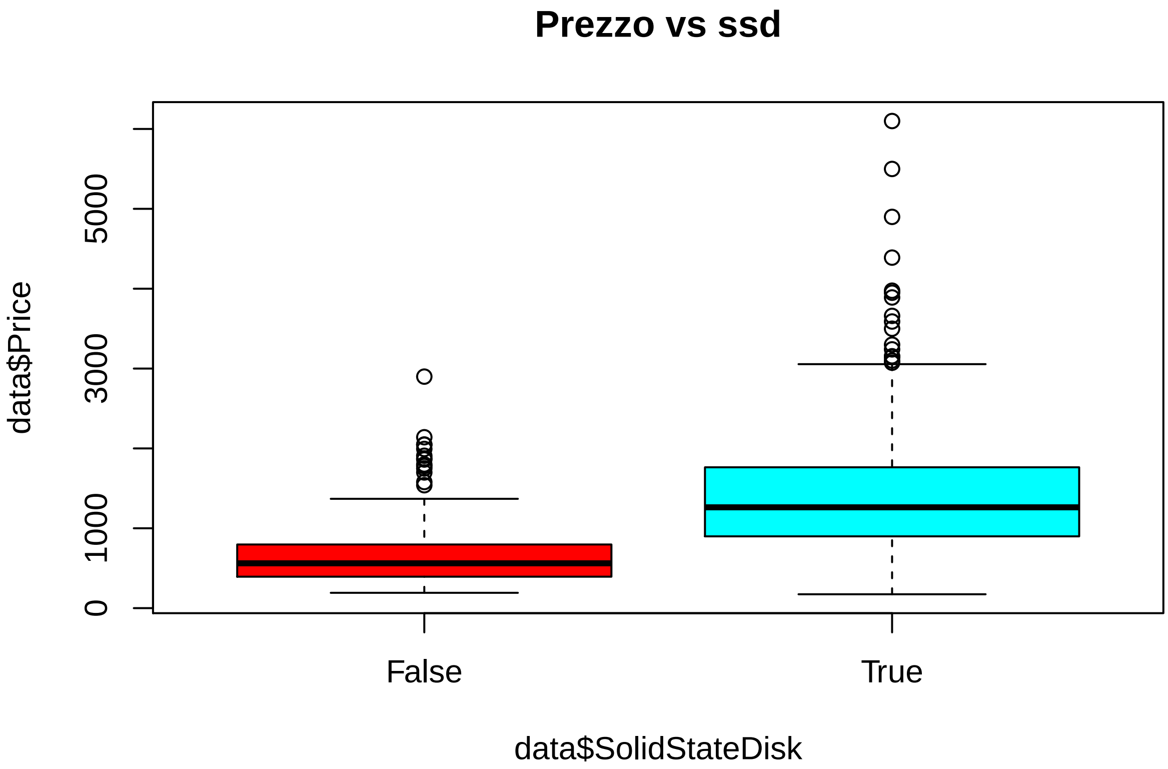


Boxplot di confronto (pre-anova)

```
boxplot(data$Price~data$Company,
        main="Boxplot Prezzo per compagnia",
        col= rainbow(6),
        horizontal = F)
```



```
boxplot(data$Price~data$SolidStateDisk,
        main="Prezzo vs ssd",
        col= rainbow(2),
        horizontal = F)
```



ANOVA

A una via

```
lmA = lm(Price ~ SolidStateDisk, data=data)
summary(lmA)
```

```
##
## Call:
## lm(formula = Price ~ SolidStateDisk, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1214.8  -343.3   -96.8   261.1  4710.2
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)          637.86      27.98   22.80   <2e-16 ***
## SolidStateDiskTrue   750.93      34.78   21.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 600 on 1301 degrees of freedom
## Multiple R-squared:  0.2638, Adjusted R-squared:  0.2632
## F-statistic: 466.2 on 1 and 1301 DF,  p-value: < 2.2e-16
```

```
drop1(lmA, test = 'F')
```

```
## Single term deletions
##
## Model:
## Price ~ SolidStateDisk
##                Df Sum of Sq       RSS   AIC F value    Pr(>F)
## <none>                      468355897 16672
## SolidStateDisk  1 167819064 636174961 17069  466.17 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lmA)
```

```
## Analysis of Variance Table
##
## Response: Price
##                  Df    Sum Sq   Mean Sq F value    Pr(>F)
## SolidStateDisk    1 167819064 167819064  466.17 < 2.2e-16 ***
## Residuals      1301 468355897    359997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
library(lsmeans)
```

```
## Loading required package: emmeans
```

```
## The 'lsmeans' package is now basically a front end for 'emmeans'.
## Users are encouraged to switch the rest of the way.
## See help('transition') for more information, including how to
## convert old 'lsmeans' objects and scripts to work with 'emmeans'.
```

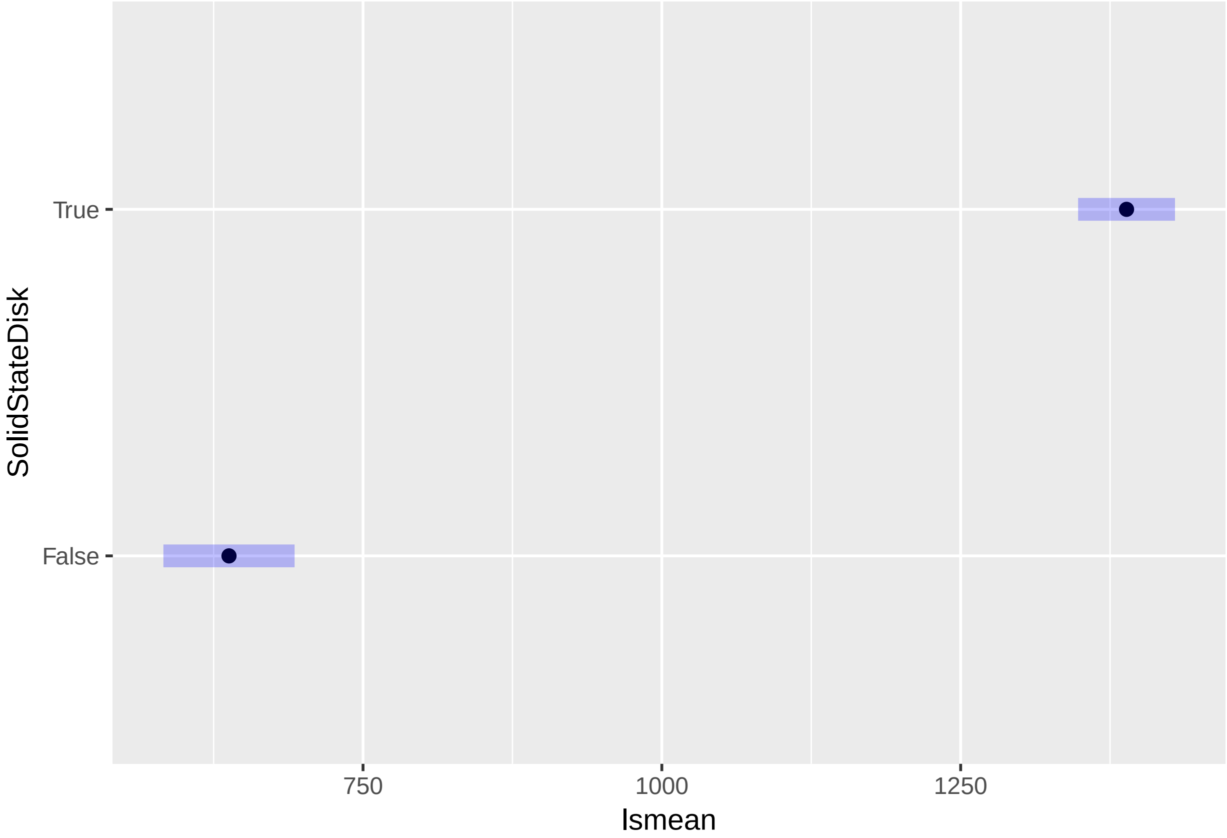
```
ls_SolidStateDisk = lsmeans(lmA,pairwise ~ SolidStateDisk,adjust = 'tukey')
ls_SolidStateDisk$contrasts
```

```
##  contrast     estimate   SE   df t.ratio p.value
##  False - True     -751 34.8 1301 -21.591 <.0001
```

```
ls_SolidStateDisk$lsmeans
```

```
##  SolidStateDisk lsmean   SE   df lower.CL upper.CL
##  False             638 28.0 1301      583      693
##  True             1389 20.7 1301     1348     1429
##
## Confidence level used: 0.95
```

```
plot(ls_SolidStateDisk$lsmeans, alpha = .05)
```



```
lmB = lm(Price ~ Company, data=data)
summary(lmB)
```

```
##
## Call:
## lm(formula = Price ~ Company, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2317.1  -452.8  -127.4   288.5  3812.6
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         626.78      63.43   9.881  < 2e-16 ***
## CompanyApple        937.42     154.14   6.082 1.57e-09 ***
## CompanyAsus         477.39      81.53   5.856 6.03e-09 ***
## CompanyChuwi      -312.48     377.06  -0.829 0.407416
## CompanyDell        559.29      73.62   7.597 5.80e-14 ***
## CompanyFujitsu     102.22     377.06   0.271 0.786352
## CompanyGoogle     1050.89     377.06   2.787 0.005397 **
## CompanyHP          441.00      74.41   5.927 3.96e-09 ***
## CompanyHuawei      797.22     459.62   1.735 0.083065 .
## CompanyLenovo      459.61      73.62   6.243 5.81e-10 ***
## CompanyLG         1472.22     377.06   3.904 9.93e-05 ***
## CompanyMediacom   -331.78     251.46  -1.319 0.187270
## CompanyMicrosoft   985.53     270.37   3.645 0.000278 ***
## CompanyMSI        1102.13     108.16  10.190  < 2e-16 ***
## CompanyRazer      2719.37     251.46  10.814  < 2e-16 ***
## CompanySamsung     786.67     223.77   3.515 0.000454 ***
## CompanyToshiba     641.04     112.51   5.698 1.50e-08 ***
## CompanyVero       -409.35     328.08  -1.248 0.212365
## CompanyXiaomi      506.69     328.08   1.544 0.122740
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 643.8 on 1284 degrees of freedom
## Multiple R-squared:  0.1635, Adjusted R-squared:  0.1518
## F-statistic: 13.94 on 18 and 1284 DF,  p-value: < 2.2e-16
```

```
drop1(lmB, test = 'F')
```

```
## Single term deletions
##
## Model:
## Price ~ Company
##         Df Sum of Sq       RSS   AIC F value    Pr(>F)
## <none>                532160971 16873
## Company 18 104013991 636174961 17069  13.943 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

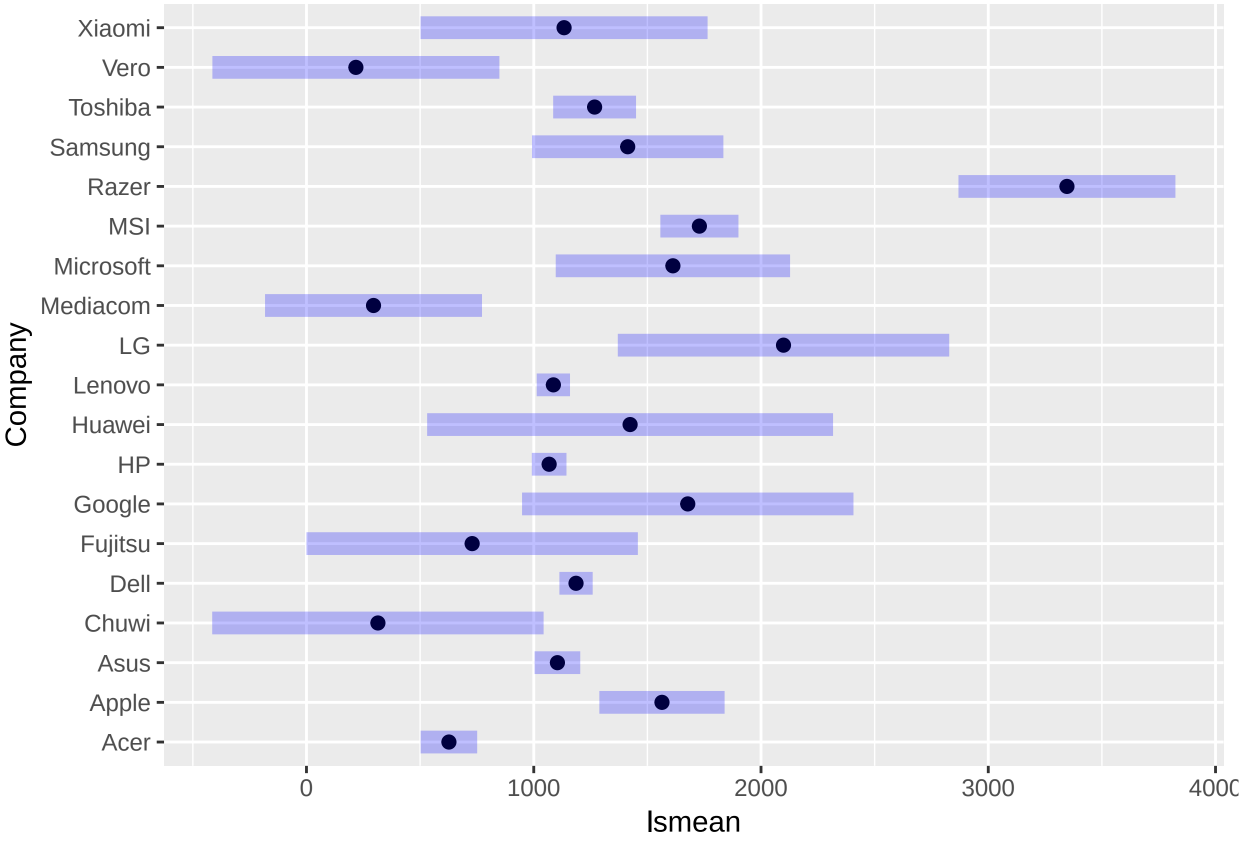
```
anova(lmB)
```

```
## Analysis of Variance Table
##
## Response: Price
##             Df    Sum Sq Mean Sq F value    Pr(>F)
## Company     18 104013991 5778555  13.943 < 2.2e-16 ***
## Residuals 1284 532160971  414456
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
ls_Company = lsmeans(lmB,pairwise ~ Company,adjust = 'tukey')
#ls_Company$contrasts #too long to be printed
ls_Company$lsmeans
```

```
##  Company   lsmean    SE   df lower.CL upper.CL
##  Acer         627  63.4 1284  502.331      751
##  Apple       1564 140.5 1284 1288.594     1840
##  Asus        1104  51.2 1284 1003.692     1205
##  Chuwi        314 371.7 1284 -414.885     1043
##  Dell        1186  37.4 1284 1112.783     1259
##  Fujitsu      729 371.7 1284   -0.182     1458
##  Google      1678 371.7 1284  948.485     2407
##  HP          1068  38.9 1284  991.475     1144
##  Huawei      1424 455.2 1284  530.938     2317
##  Lenovo      1086  37.4 1284 1013.099     1160
##  LG          2099 371.7 1284 1369.818     2828
##  Mediacom     295 243.3 1284 -182.362      772
##  Microsoft   1612 262.8 1284 1096.699     2128
##  MSI         1729  87.6 1284 1557.038     1901
##  Razer       3346 243.3 1284 2868.781     3824
##  Samsung     1413 214.6 1284  992.451     1834
##  Toshiba     1268  92.9 1284 1085.517     1450
##  Vero         217 321.9 1284 -414.065      849
##  Xiaomi      1133 321.9 1284  501.972     1765
##
## Confidence level used: 0.95
```

```
plot(ls_Company$lsmeans, alpha = .05)
```



```
lmC = lm(Price ~ TypeName, data=data)
summary(lmC)
```

```
##
## Call:
## lm(formula = Price ~ TypeName, data = data)
##
## Residuals:
##      Min      1Q  Median      3Q     Max
## -1049.2  -381.7   -98.1   267.6  4367.6
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1282.40      50.01  25.642  < 2e-16 ***
## TypeNameGaming        448.98      63.07   7.119 1.79e-12 ***
## TypeNameNetbook      -646.17     120.86  -5.347 1.06e-07 ***
## TypeNameNotebook     -500.32      54.01  -9.263  < 2e-16 ***
## TypeNameUltrabook     265.83      63.60   4.180 3.12e-05 ***
## TypeNameWorkstation   997.96     113.74   8.774  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 550.1 on 1297 degrees of freedom
## Multiple R-squared:  0.383,  Adjusted R-squared:  0.3806
## F-statistic:   161 on 5 and 1297 DF,  p-value: < 2.2e-16
```

```
drop1(lmC, test = 'F')
```

```
## Single term deletions
##
## Model:
## Price ~ TypeName
##          Df Sum of Sq       RSS   AIC F value    Pr(>F)
## <none>                392518380 16450
## TypeName  5 243656581 636174961 17069  161.02 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lmC)
```

```
## Analysis of Variance Table
##
## Response: Price
##             Df    Sum Sq  Mean Sq F value    Pr(>F)
## TypeName     5 243656581 48731316  161.02 < 2.2e-16 ***
## Residuals 1297 392518380   302636
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

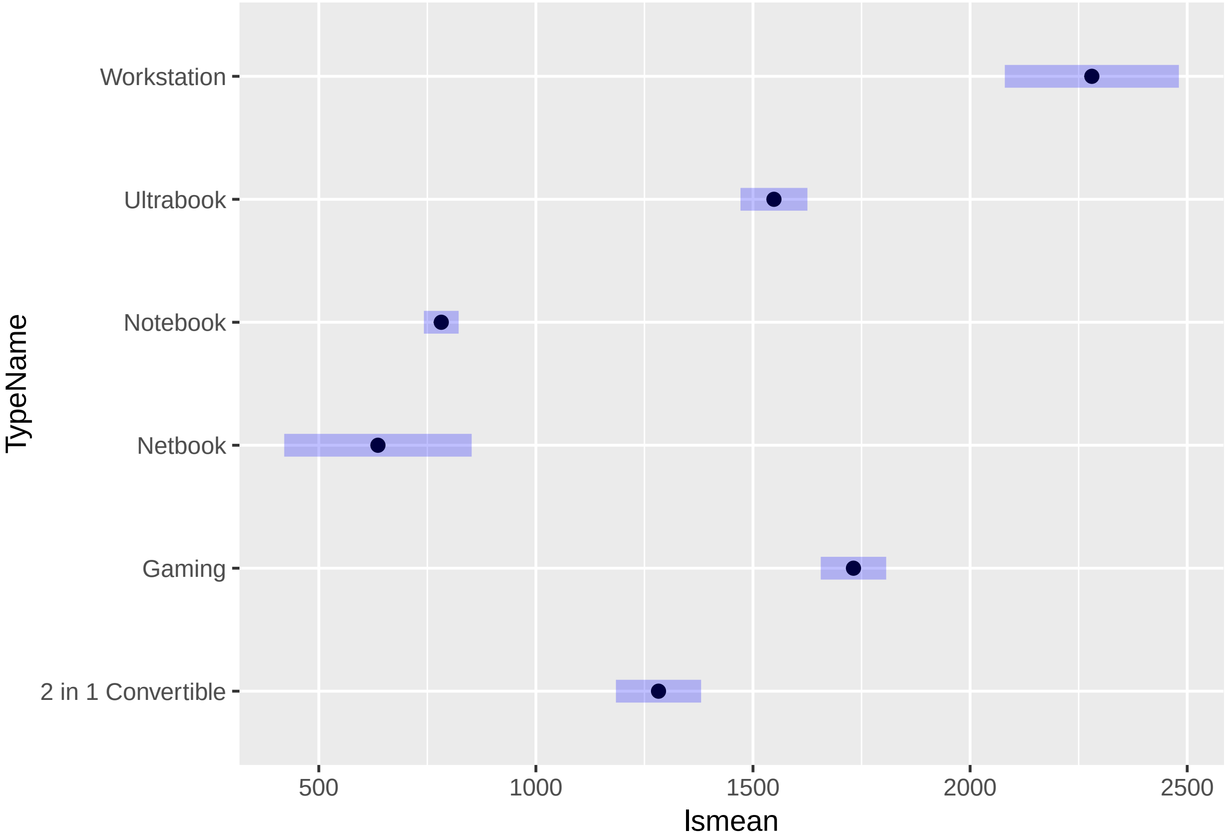
```
ls_TypeName = lsmeans(lmC,pairwise ~ TypeName,adjust = 'tukey')
ls_TypeName$contrasts
```

```
##  contrast                         estimate    SE   df t.ratio p.value
##  2 in 1 Convertible - Gaming          -449  63.1 1297  -7.119 <.0001
##  2 in 1 Convertible - Netbook          646 120.9 1297   5.347 <.0001
##  2 in 1 Convertible - Notebook         500  54.0 1297   9.263 <.0001
##  2 in 1 Convertible - Ultrabook       -266  63.6 1297  -4.180 0.0004
##  2 in 1 Convertible - Workstation     -998 113.7 1297  -8.774 <.0001
##  Gaming - Netbook                     1095 116.5 1297   9.397 <.0001
##  Gaming - Notebook                     949  43.5 1297  21.821 <.0001
##  Gaming - Ultrabook                    183  55.0 1297   3.333 0.0114
##  Gaming - Workstation                 -549 109.1 1297  -5.030 <.0001
##  Netbook - Notebook                   -146 111.9 1297  -1.303 0.7833
##  Netbook - Ultrabook                  -912 116.8 1297  -7.806 <.0001
##  Netbook - Workstation               -1644 150.1 1297 -10.951 <.0001
##  Notebook - Ultrabook                  -766  44.3 1297 -17.304 <.0001
##  Notebook - Workstation               -1498 104.2 1297 -14.383 <.0001
##  Ultrabook - Workstation               -732 109.5 1297  -6.689 <.0001
##
## P value adjustment: tukey method for comparing a family of 6 estimates
```

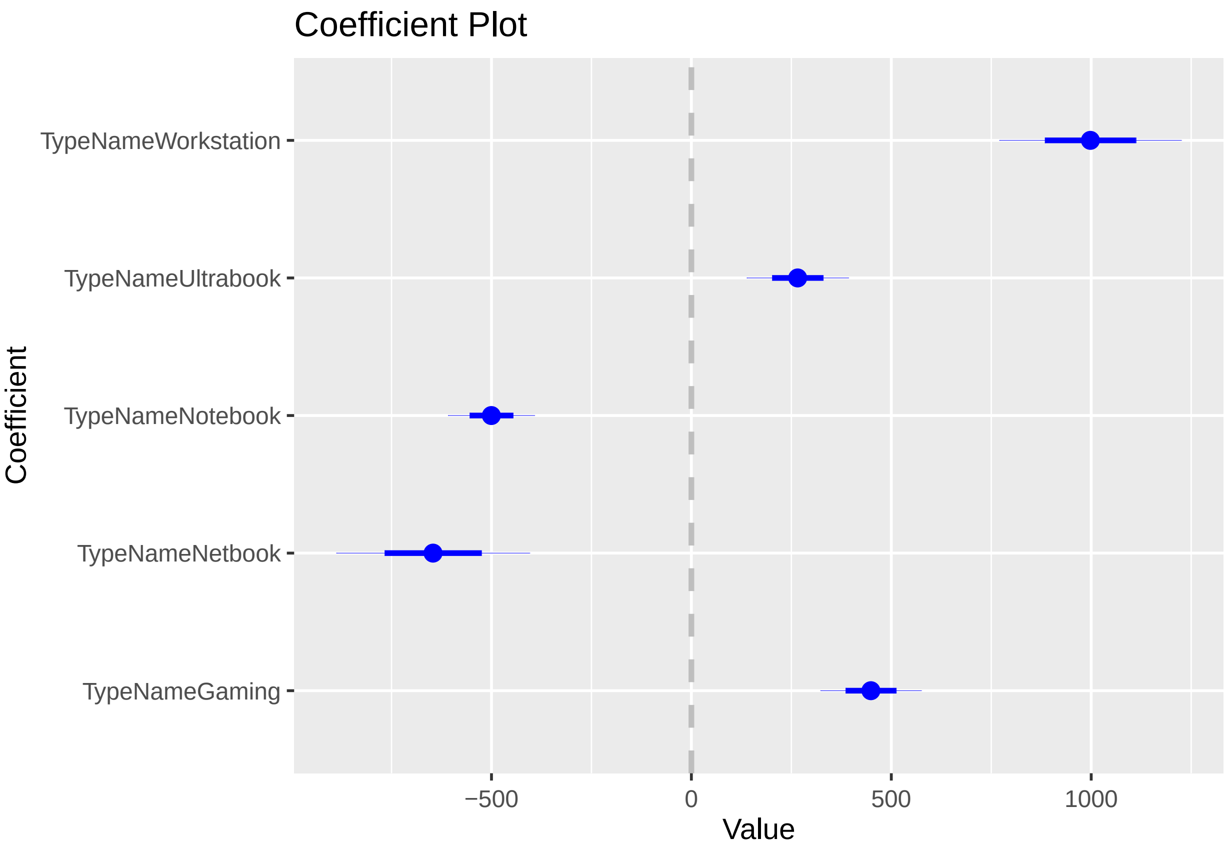
```
ls_TypeName$lsmeans
```

```
##  TypeName           lsmean    SE   df lower.CL upper.CL
##  2 in 1 Convertible   1282  50.0 1297     1184     1381
##  Gaming               1731  38.4 1297     1656     1807
##  Netbook               636 110.0 1297      420      852
##  Notebook              782  20.4 1297      742      822
##  Ultrabook            1548  39.3 1297     1471     1625
##  Workstation          2280 102.2 1297     2080     2481
##
## Confidence level used: 0.95
```

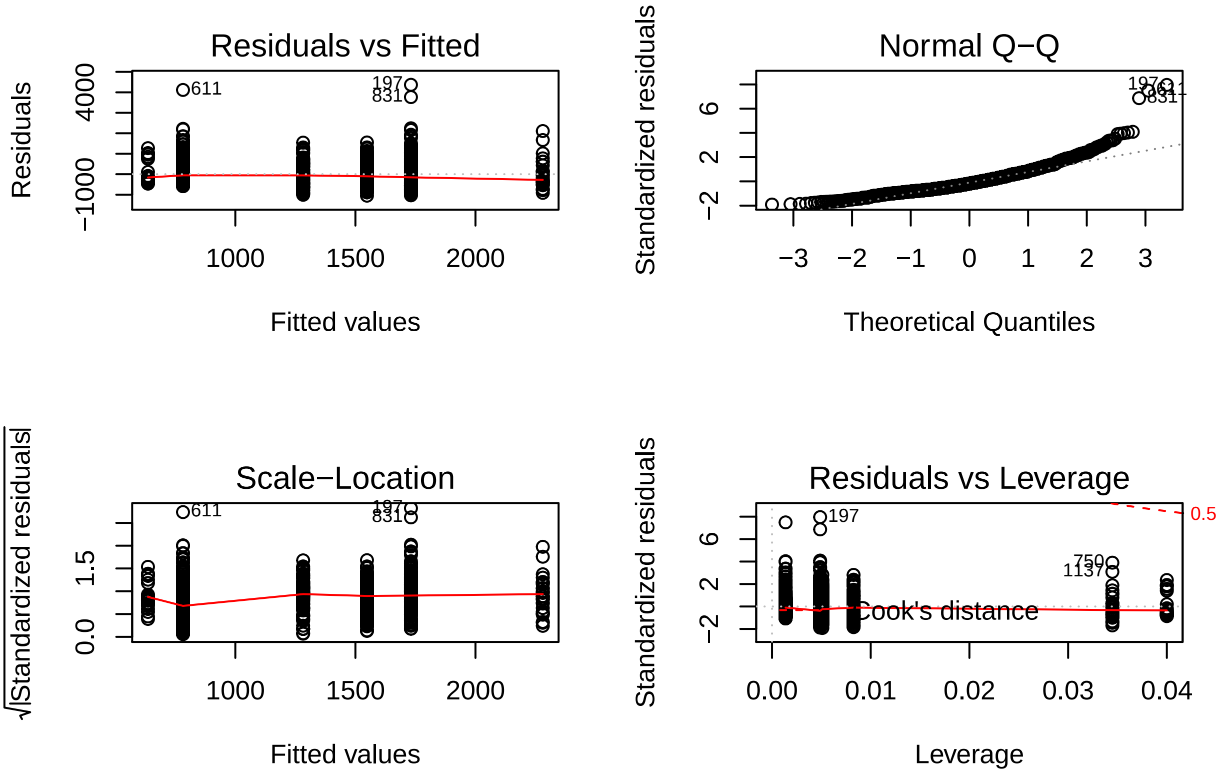
```
plot(ls_TypeName$lsmeans, alpha = .05)
```



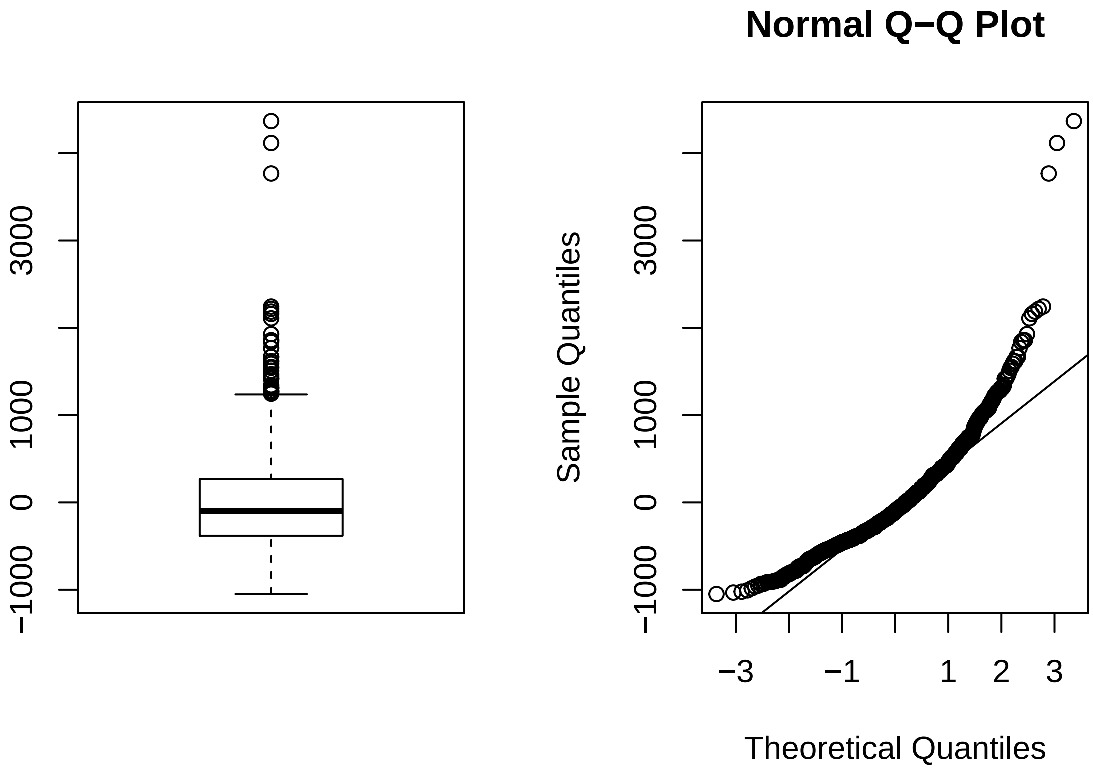
```
library(coefplot)
#library(forestmodel)
coefplot(lmC, intercept = FALSE)
```



```r
par(mfrow = c(2,2))
plot(lmC)
```



```r
#(not) normal distribution of residuals
par(mfrow=c(1,2))
boxplot(lmC$residuals)
qqnorm(lmC$residuals);qqline(lmC$residuals)
```



```r
ad.test(lmC$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  lmC$residuals
## A = 22.667, p-value < 2.2e-16
```

```r
shapiro.test(lmC$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lmC$residuals
## W = 0.89641, p-value < 2.2e-16
```

```r
#let's try again with the log correction
lmC_log = lm(log(Price) ~ TypeName, data=data)
summary(lmC_log)#R^2 increases
```

```
##
## Call:
## lm(formula = log(Price) ~ TypeName, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.40971 -0.33589  0.00698  0.33215  1.96853
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.02648    0.04379 160.456  < 2e-16 ***
## TypeNameGaming       0.33865    0.05522   6.133 1.15e-09 ***
## TypeNameNetbook     -0.91149    0.10583  -8.613  < 2e-16 ***
## TypeNameNotebook    -0.49823    0.04729 -10.534  < 2e-16 ***
## TypeNameUltrabook    0.26648    0.05569   4.785 1.91e-06 ***
## TypeNameWorkstation  0.66479    0.09959   6.675 3.65e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4817 on 1297 degrees of freedom
## Multiple R-squared:  0.4061, Adjusted R-squared:  0.4038
## F-statistic: 177.4 on 5 and 1297 DF,  p-value: < 2.2e-16
```

```
drop1(lmC_log, test = 'F')
```

```
## Single term deletions
##
## Model:
## log(Price) ~ TypeName
##          Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>                300.95 -1897.5
## TypeName  5    205.76 506.71 -1228.7  177.36 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lmC_log)
```

```
## Analysis of Variance Table
##
## Response: log(Price)
##             Df Sum Sq Mean Sq F value    Pr(>F)
## TypeName     5 205.76  41.152  177.36 < 2.2e-16 ***
## Residuals 1297 300.95   0.232
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

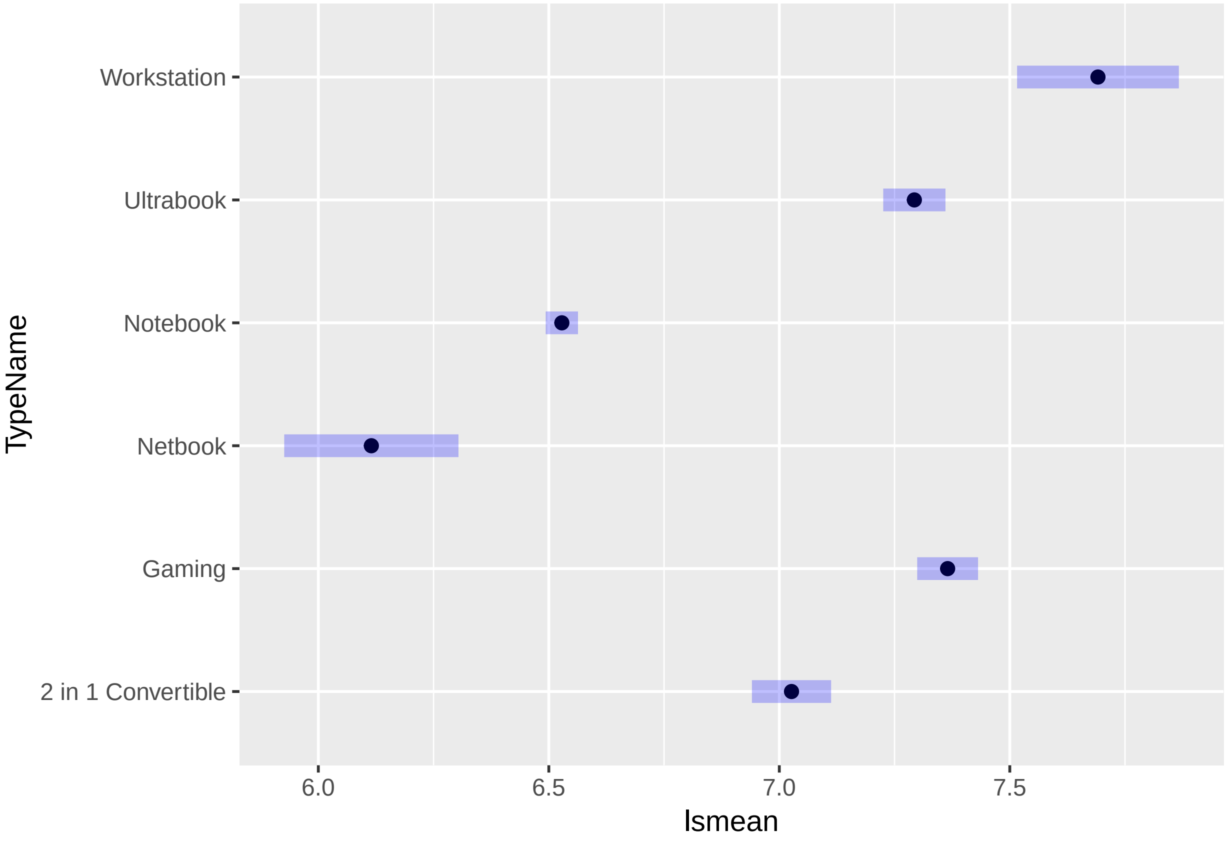
```
ls_TypeName_log = lsmeans(lmC_log,pairwise ~ TypeName,adjust = 'tukey')
ls_TypeName_log$contrasts
```

```
##  contrast                         estimate     SE   df t.ratio p.value
##  2 in 1 Convertible - Gaming       -0.3387 0.0552 1297  -6.133 <.0001
##  2 in 1 Convertible - Netbook       0.9115 0.1058 1297   8.613 <.0001
##  2 in 1 Convertible - Notebook      0.4982 0.0473 1297  10.534 <.0001
##  2 in 1 Convertible - Ultrabook    -0.2665 0.0557 1297  -4.785 <.0001
##  2 in 1 Convertible - Workstation  -0.6648 0.0996 1297  -6.675 <.0001
##  Gaming - Netbook                   1.2501 0.1020 1297  12.251 <.0001
##  Gaming - Notebook                  0.8369 0.0381 1297  21.970 <.0001
##  Gaming - Ultrabook                 0.0722 0.0481 1297   1.500 0.6644
##  Gaming - Workstation              -0.3261 0.0956 1297  -3.413 0.0087
##  Netbook - Notebook                -0.4133 0.0980 1297  -4.218 0.0004
##  Netbook - Ultrabook               -1.1780 0.1023 1297 -11.515 <.0001
##  Netbook - Workstation             -1.5763 0.1315 1297 -11.990 <.0001
##  Notebook - Ultrabook               -0.7647 0.0388 1297 -19.725 <.0001
##  Notebook - Workstation             -1.1630 0.0912 1297 -12.750 <.0001
##  Ultrabook - Workstation            -0.3983 0.0958 1297  -4.156 0.0005
##
## Results are given on the log (not the response) scale.
## P value adjustment: tukey method for comparing a family of 6 estimates
```

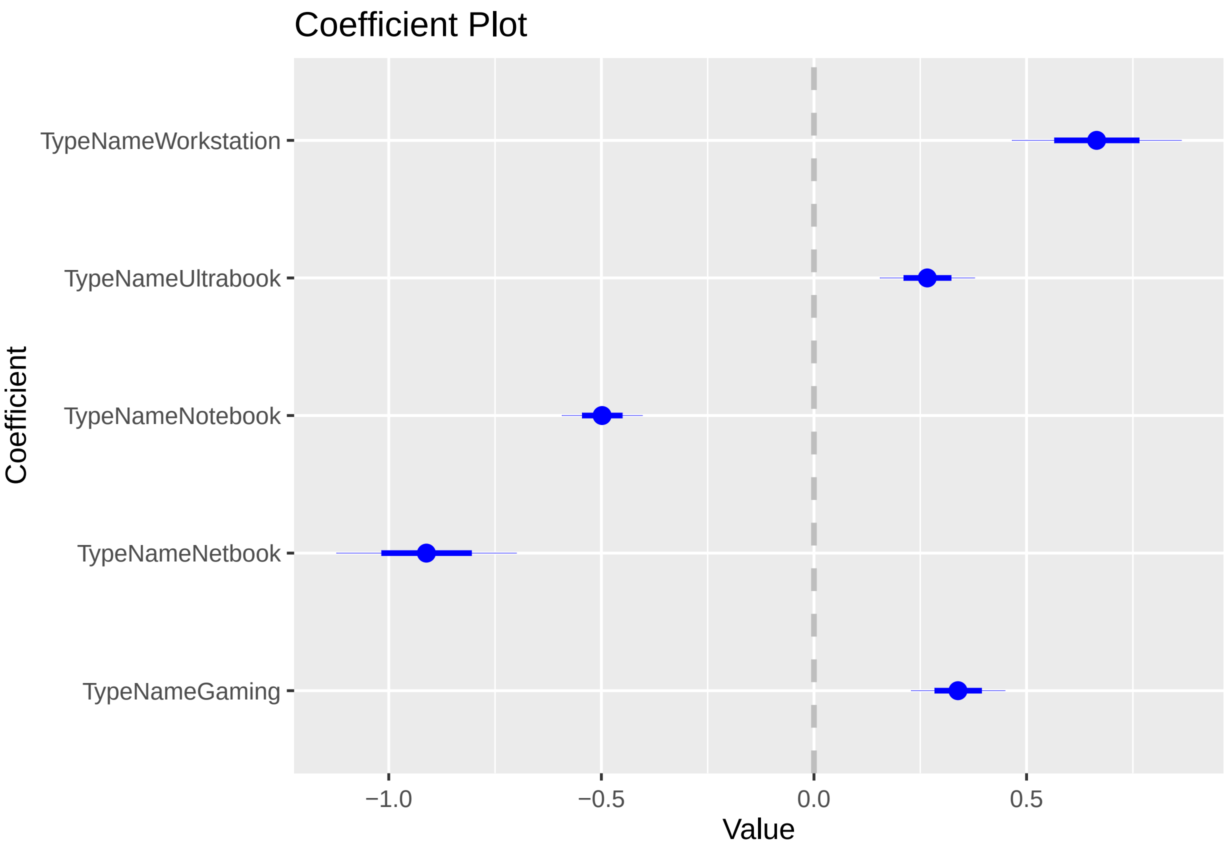
```
ls_TypeName_log$lsmeans
```

```
##  TypeName           lsmean     SE   df lower.CL upper.CL
##  2 in 1 Convertible   7.03 0.0438 1297     6.94     7.11
##  Gaming               7.37 0.0336 1297     7.30     7.43
##  Netbook              6.11 0.0963 1297     5.93     6.30
##  Notebook             6.53 0.0179 1297     6.49     6.56
##  Ultrabook            7.29 0.0344 1297     7.23     7.36
##  Workstation          7.69 0.0894 1297     7.52     7.87
##
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
```

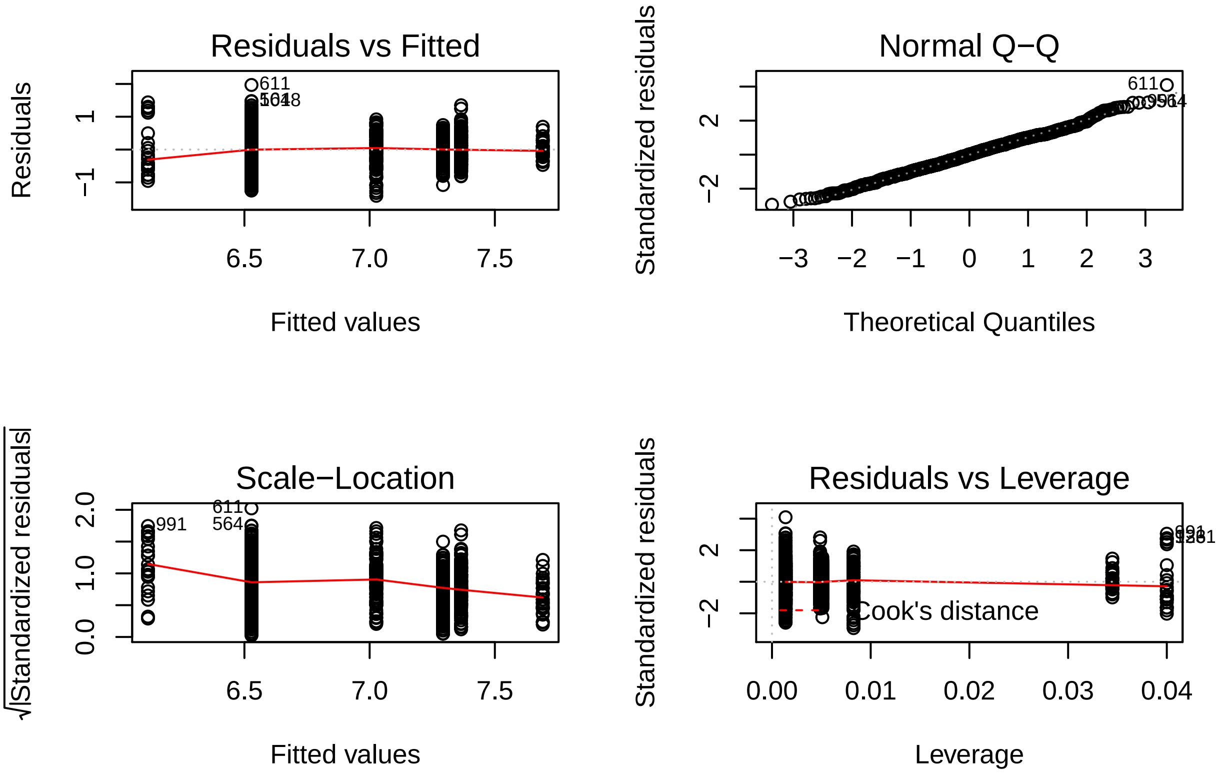
```
plot(ls_TypeName_log$lsmeans, alpha = .05)
```



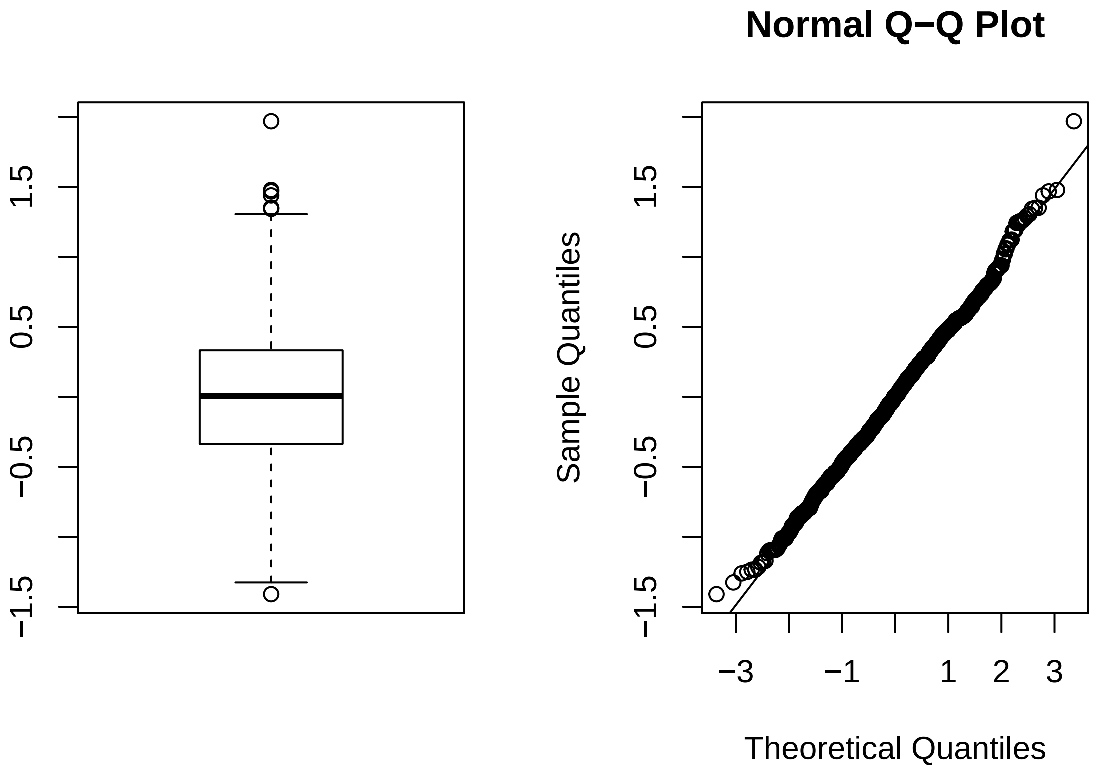
```
coefplot(lmC_log, intercept = FALSE)
```



```r
par(mfrow = c(2,2))
plot(lmC_log)
```



```
#(not) normal distribution of residuals
par(mfrow=c(1,2))
boxplot(lmC_log$residuals)
qqnorm(lmC_log$residuals);qqline(lmC_log$residuals)
```



```
ad.test(lmC_log$residuals) #normal now!
```

```
##
##  Anderson-Darling normality test
##
## data:  lmC_log$residuals
## A = 0.51757, p-value = 0.1886
```

```
shapiro.test(lmC_log$residuals) #borderline now!
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lmC_log$residuals
## W = 0.99764, p-value = 0.05462
```

A due vie

```
# Con interazione
lmC = lm(Price ~ Company*TypeName  , data=data)
drop1(lmC, test="F")
```

```
## Single term deletions
##
## Model:
## Price ~ Company * TypeName
##                    Df Sum of Sq       RSS   AIC F value    Pr(>F)
## <none>                          320739568 16273
## Company:TypeName 25  29159364 349898932 16336  4.5602 1.181e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#summary(lmC) #FIXME: too long to be printed

lmC = lm(Price ~ Company+TypeName  , data=data)
# type I effects A, B/A   C/A,B
anova(lmC)
```

```
## Analysis of Variance Table
##
## Response: Price
##             Df    Sum Sq  Mean Sq F value    Pr(>F)
## Company     18 104013991  5778555  21.123 < 2.2e-16 ***
## TypeName     5 182262038 36452408 133.246 < 2.2e-16 ***
## Residuals 1279 349898932   273572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# type III effects A/B,C , B/A,C   C/A,B
drop1(lmC, test="F")
```

```
## Single term deletions
##
## Model:
## Price ~ Company + TypeName
##          Df Sum of Sq       RSS   AIC  F value    Pr(>F)
## <none>                349898932 16336
## Company  18  42619448 392518380 16450   8.6549 < 2.2e-16 ***
## TypeName  5 182262038 532160971 16873 133.2460 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

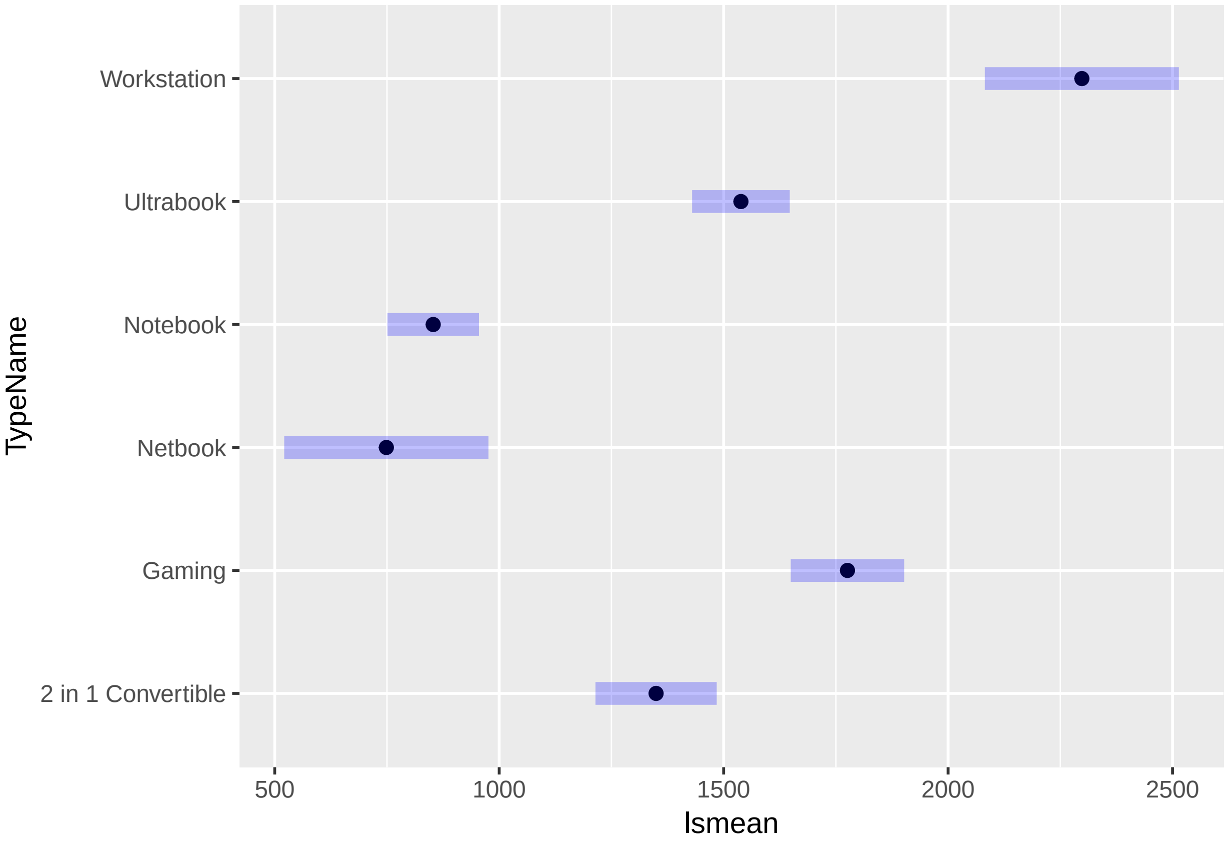
```
summary(lmC)
```

```
##
## Call:
## lm(formula = Price ~ Company + TypeName, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2147.6  -343.2   -81.9   243.1  4081.9
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)            991.52      69.88  14.189  < 2e-16 ***
## CompanyApple           383.70     132.62   2.893  0.00388 **
## CompanyAsus            168.81      67.79   2.490  0.01290 *
## CompanyChuwi          -180.68     306.47  -0.590  0.55559
## CompanyDell            350.73      60.52   5.796 8.56e-09 ***
## CompanyFujitsu         234.02     306.47   0.764  0.44525
## CompanyGoogle          497.17     309.44   1.607  0.10837
## CompanyHP              337.48      60.85   5.546 3.55e-08 ***
## CompanyHuawei         243.50     375.96   0.648  0.51731
## CompanyLenovo         322.12      60.24   5.348 1.05e-07 ***
## CompanyLG             918.50     309.44   2.968  0.00305 **
## CompanyMediacom      -270.91     204.43  -1.325  0.18534
## CompanyMicrosoft      431.81     223.95   1.928  0.05406 .
## CompanyMSI            311.10      98.62   3.155  0.00165 **
## CompanyRazer         1996.14     207.24   9.632  < 2e-16 ***
## CompanySamsung        438.88     183.82   2.388  0.01710 *
## CompanyToshiba        601.45      92.12   6.529 9.52e-11 ***
## CompanyVero          -277.55     266.70  -1.041  0.29821
## CompanyXiaomi         295.72     267.40   1.106  0.26896
## TypeNameGaming        426.29      65.51   6.507 1.10e-10 ***
## TypeNameNetbook      -600.94     115.75  -5.192 2.42e-07 ***
## TypeNameNotebook     -496.54      51.98  -9.552  < 2e-16 ***
## TypeNameUltrabook     188.98      63.81   2.962  0.00312 **
## TypeNameWorkstation   948.46     109.22   8.684  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 523 on 1279 degrees of freedom
## Multiple R-squared:   0.45,  Adjusted R-squared:  0.4401
## F-statistic:  45.5 on 23 and 1279 DF,  p-value: < 2.2e-16
```

```
# contrasti
library(lsmeans)
ls=lsmeans(lmC, #FIXME: @Andrea, c'era lmB ma credo tu volessi scrivere lmC, in case check it
           pairwise ~ TypeName ,
           adjust="tukey")
ls$lsmeans
```

```
##  TypeName           lsmean    SE   df lower.CL upper.CL
##  2 in 1 Convertible   1350  68.9 1279     1214     1485
##  Gaming               1776  64.4 1279     1649     1902
##  Netbook               749 115.9 1279      521      976
##  Notebook              853  52.0 1279      751      955
##  Ultrabook            1538  55.5 1279     1430     1647
##  Workstation          2298 110.1 1279     2082     2514
##
## Results are averaged over the levels of: Company
## Confidence level used: 0.95
```

```
# plot lsmeans and 95% confid int
plot(ls$lsmeans, alpha = .05)
```



```
# contrasts between predicted lsmeans
ls$contrasts
```

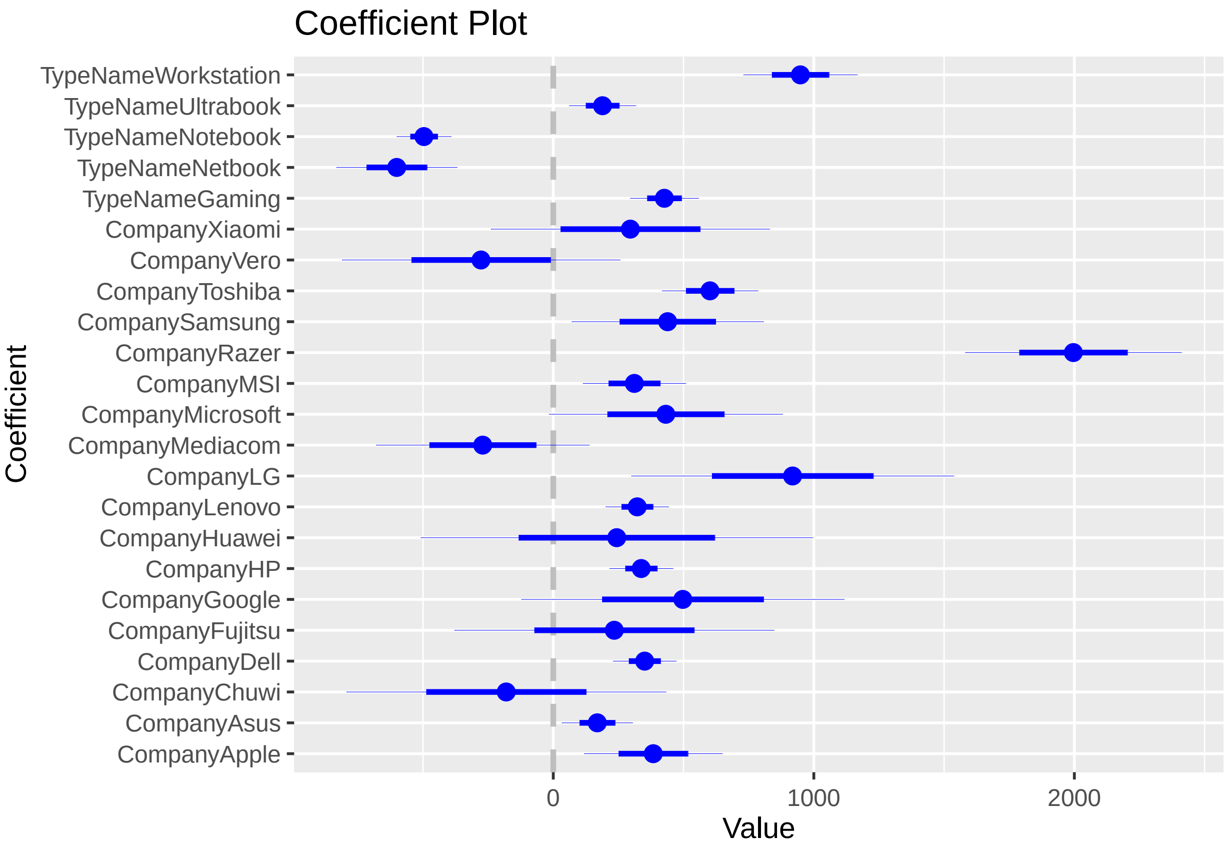
```
##  contrast                               estimate    SE   df t.ratio p.value
##  2 in 1 Convertible - Gaming                -426  65.5 1279  -6.507 <.0001
##  2 in 1 Convertible - Netbook                601 115.7 1279   5.192 <.0001
##  2 in 1 Convertible - Notebook               497  52.0 1279   9.552 <.0001
##  2 in 1 Convertible - Ultrabook             -189  63.8 1279  -2.962 0.0367
##  2 in 1 Convertible - Workstation           -948 109.2 1279  -8.684 <.0001
##  Gaming - Netbook                           1027 114.5 1279   8.972 <.0001
##  Gaming - Notebook                           923  49.4 1279  18.671 <.0001
##  Gaming - Ultrabook                          237  61.1 1279   3.882 0.0015
##  Gaming - Workstation                       -522 108.3 1279  -4.820 <.0001
##  Netbook - Notebook                         -104 107.0 1279  -0.975 0.9258
##  Netbook - Ultrabook                        -790 113.3 1279  -6.969 <.0001
##  Netbook - Workstation                     -1549 143.8 1279 -10.774 <.0001
##  Notebook - Ultrabook                       -686  46.5 1279 -14.754 <.0001
##  Notebook - Workstation                    -1445  99.8 1279 -14.475 <.0001
##  Ultrabook - Workstation                    -759 106.4 1279  -7.138 <.0001
##
## Results are averaged over the levels of: Company
## P value adjustment: tukey method for comparing a family of 6 estimates
```

```
# if at least one contrast is significant, the variable
# is significant in the anova table # drop1 effects

# contrast among predicted lsmeans and overall lsmean
c= contrast(ls, method = "eff")
```

```
c
```

```
## $lsmeans
##  contrast                  estimate   SE   df t.ratio p.value
##  2 in 1 Convertible effect    -77.7 47.9 1279  -1.623 0.1048
##  Gaming effect                348.6 46.0 1279   7.583 <.0001
##  Netbook effect              -678.6 90.2 1279  -7.521 <.0001
##  Notebook effect             -574.2 31.8 1279 -18.032 <.0001
##  Ultrabook effect             111.3 43.8 1279   2.542 0.0134
##  Workstation effect           870.7 84.6 1279  10.287 <.0001
##
## Results are averaged over the levels of: Company
## P value adjustment: fdr method for 6 tests
##
## $contrasts
##  contrast                                estimate    SE   df t.ratio
##  2 in 1 Convertible - Gaming effect        -150.6  71.6 1279  -2.103
##  2 in 1 Convertible - Netbook effect        876.6 121.9 1279   7.192
##  2 in 1 Convertible - Notebook effect       772.2  51.2 1279  15.077
##  2 in 1 Convertible - Ultrabook effect       86.7  57.9 1279   1.498
##  2 in 1 Convertible - Workstation effect   -672.8  74.0 1279  -9.093
##  Gaming - Netbook effect                   1302.9 123.6 1279  10.544
##  Gaming - Notebook effect                  1198.5  55.4 1279  21.649
##  Gaming - Ultrabook effect                  513.0  60.9 1279   8.416
##  Gaming - Workstation effect               -246.5  77.4 1279  -3.186
##  Netbook - Notebook effect                  171.2 107.0 1279   1.600
##  Netbook - Ultrabook effect                -514.3 110.5 1279  -4.655
##  Netbook - Workstation effect             -1273.7 119.6 1279 -10.649
##  Notebook - Ultrabook effect               -409.9  55.3 1279  -7.416
##  Notebook - Workstation effect            -1169.3  71.6 1279 -16.325
##  Ultrabook - Workstation effect            -483.8  84.4 1279  -5.730
##  p.value
##  0.0411
##  <.0001
##  <.0001
##  0.1345
##  <.0001
##  <.0001
##  <.0001
##  <.0001
##  0.0018
##  0.1177
##  <.0001
##  <.0001
##  <.0001
##  <.0001
##  <.0001
##
## Results are averaged over the levels of: Company
## P value adjustment: fdr method for 15 tests
```

```
library(coefplot)
coefplot(lmC, intercept=FALSE) #FIXME: @Andrea, same goes here
```



ANOVA k way

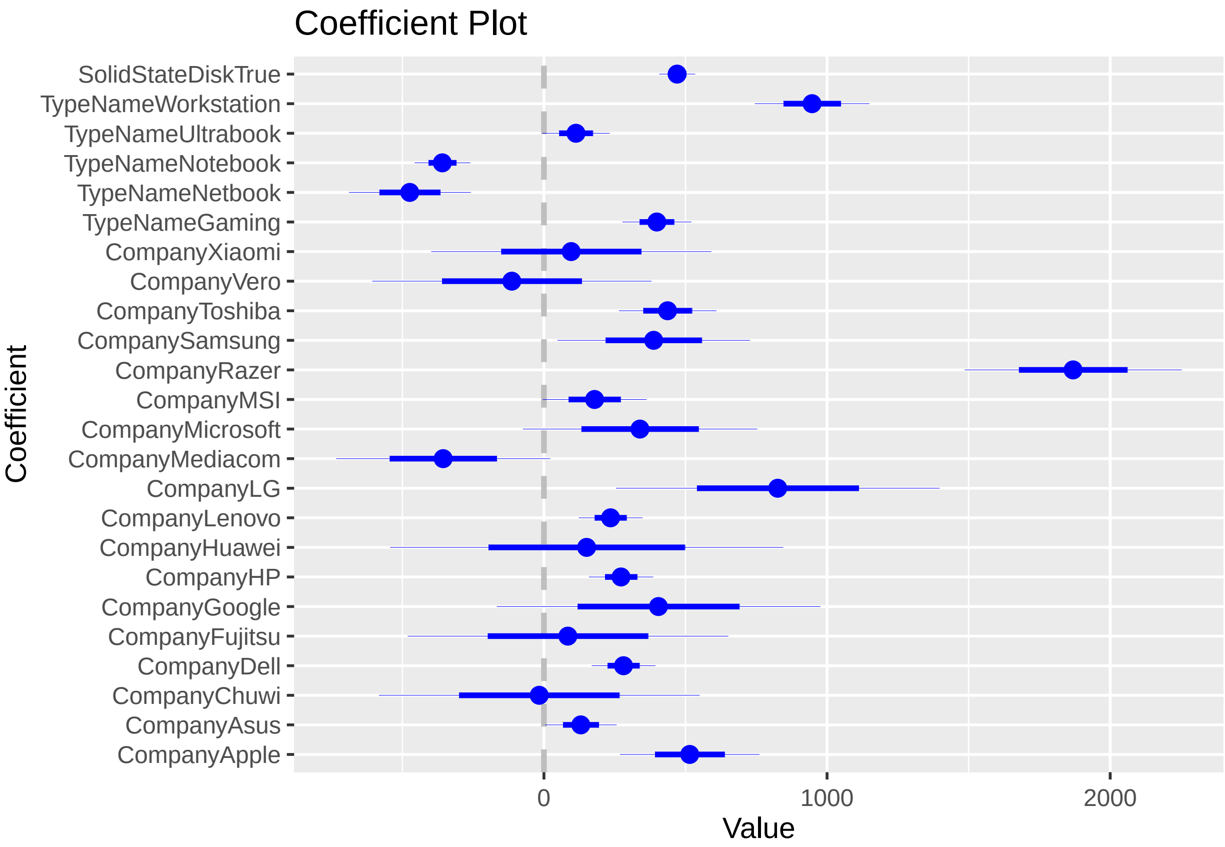
```
lmK = lm(Price ~ Company+TypeName+SolidStateDisk  , data=data)
summary(lmK)
```

```
##
## Call:
## lm(formula = Price ~ Company + TypeName + SolidStateDisk, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2113.2  -301.6   -49.8   210.4  3862.4
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             689.90      67.44  10.230  < 2e-16 ***
## CompanyApple            514.90     122.54   4.202 2.83e-05 ***
## CompanyAsus             130.26      62.53   2.083  0.03744 *
## CompanyChuwi            -16.66     282.67  -0.059  0.95300
## CompanyDell             280.98      55.97   5.020 5.88e-07 ***
## CompanyFujitsu           84.49     282.64   0.299  0.76505
## CompanyGoogle           404.40     285.26   1.418  0.15653
## CompanyHP               272.28      56.25   4.840 1.45e-06 ***
## CompanyHuawei           150.73     346.56   0.435  0.66368
## CompanyLenovo           235.35      55.81   4.217 2.65e-05 ***
## CompanyLG               825.73     285.26   2.895  0.00386 **
## CompanyMediacom        -356.00     188.50  -1.889  0.05918 .
## CompanyMicrosoft        339.04     206.50   1.642  0.10087
## CompanyMSI            178.78      91.32   1.958  0.05047 .  
## CompanyRazer         1868.90     191.19   9.775  < 2e-16 ***
## CompanySamsung        387.48     169.45   2.287  0.02238 *  
## CompanyToshiba        436.72      85.60   5.102 3.87e-07 ***
## CompanyVero          -113.54     246.04  -0.461  0.64456    
## CompanyXiaomi          96.18     246.80   0.390  0.69680    
## TypeNameGaming        398.61      60.40   6.599 6.05e-11 ***
## TypeNameNetbook      -473.92     107.01  -4.429 1.03e-05 ***
## TypeNameNotebook     -358.93      48.77  -7.360 3.28e-13 ***
## TypeNameUltrabook     113.04      59.02   1.915  0.05570 .  
## TypeNameWorkstation   946.96     100.66   9.407  < 2e-16 ***
## SolidStateDiskTrue    470.33      31.17  15.089  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 482.1 on 1278 degrees of freedom
## Multiple R-squared:  0.5332, Adjusted R-squared:  0.5244 
## F-statistic: 60.82 on 24 and 1278 DF,  p-value: < 2.2e-16
```

```r
drop1(lmK, test="F") # type III SS
```

```
## Single term deletions
## 
## Model:
## Price ~ Company + TypeName + SolidStateDisk
##                 Df Sum of Sq       RSS   AIC  F value    Pr(>F)    
## <none>                       296988657 16125                       
## Company        18  33990309 330978966 16230   8.1259 < 2.2e-16 ***
## TypeName        5 109128253 406116910 16523  93.9200 < 2.2e-16 ***
## SolidStateDisk  1  52910275 349898932 16336 227.6832 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coefplot(lmK, intercept=FALSE)
```

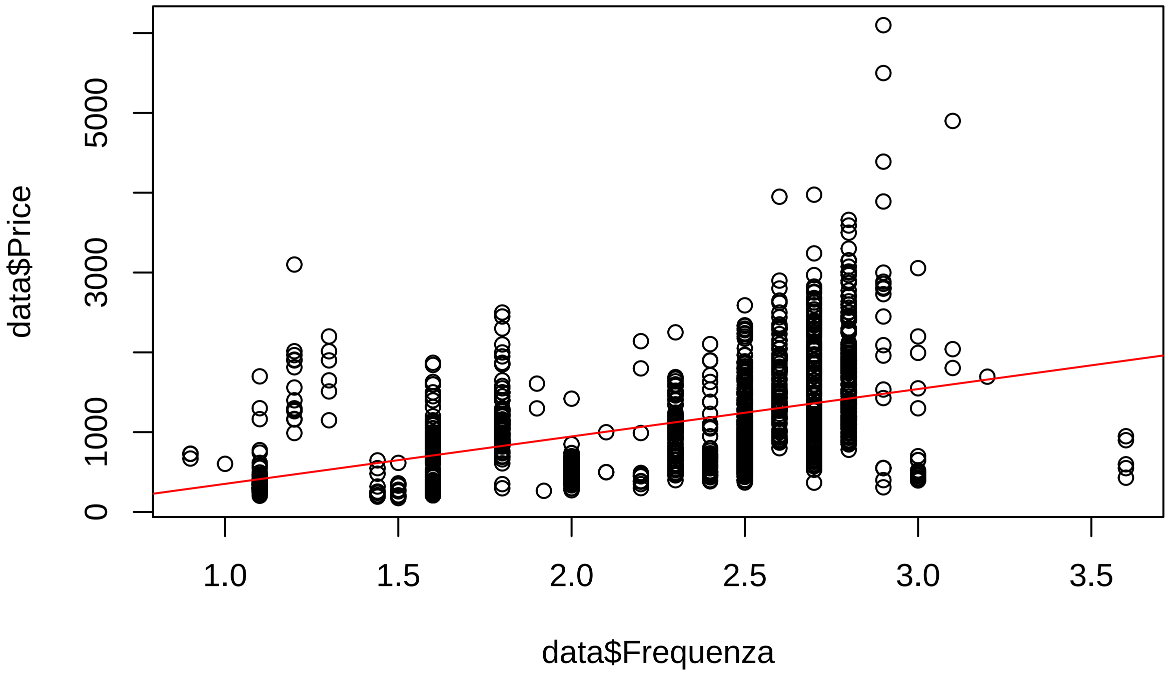


Regressione lineare

```
lmA<-lm(Price ~ Frequenza  , data=data)
summary(lmA)
```

```
##
## Call:
## lm(formula = Price ~ Frequenza, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1467.6  -453.8  -119.6   327.6  4618.2
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -241.84      81.32  -2.974    0.003 **
## Frequenza     594.02      34.55  17.194   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 631.2 on 1301 degrees of freedom
## Multiple R-squared:  0.1852, Adjusted R-squared:  0.1845
## F-statistic: 295.6 on 1 and 1301 DF,  p-value: < 2.2e-16
```

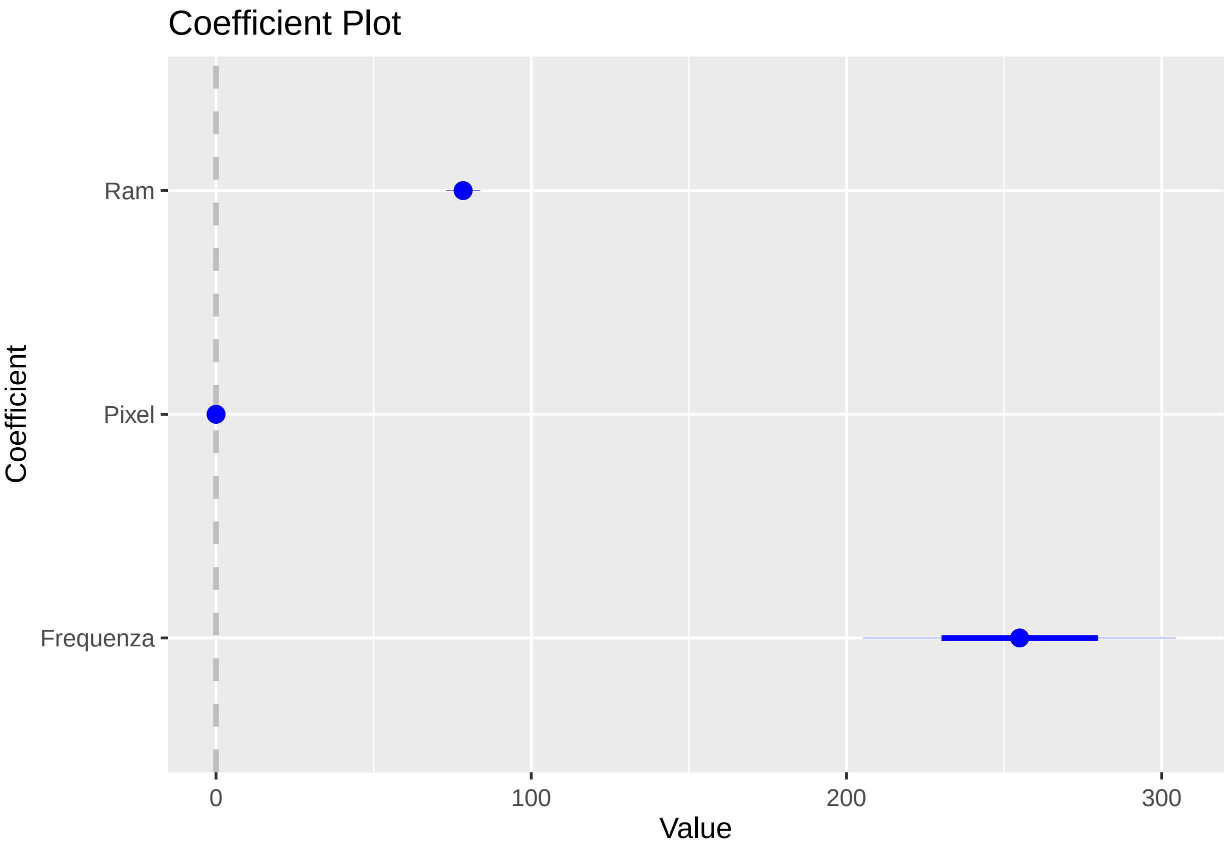
```
plot(data$Frequenza,data$Price)
abline(lmA,col="red")
```



```
lmA<-lm(Price ~ Frequenza+Pixel+Ram  , data=data)
summary(lmA)
```

```
##
## Call:
## lm(formula = Price ~ Frequenza + Pixel + Ram, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1785.72  -257.23   -66.06   191.11  2791.53
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.076e+02  5.547e+01  -7.349 3.52e-13 ***
## Frequenza    2.549e+02  2.474e+01  10.306  < 2e-16 ***
## Pixel        1.329e-04  9.117e-06  14.575  < 2e-16 ***
## Ram          7.839e+01  2.658e+00  29.488  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 420.2 on 1299 degrees of freedom
## Multiple R-squared:  0.6395, Adjusted R-squared:  0.6386
## F-statistic:   768 on 3 and 1299 DF,  p-value: < 2.2e-16
```

```
coefplot(lmA, intercept=FALSE)
```



ANCOVA

```r
lmK = lm(Price ~ Company+TypeName+SolidStateDisk+ Frequenza+Pixel+Ram  , data=data)
summary(lmK)
```

```
##
## Call:
## lm(formula = Price ~ Company + TypeName + SolidStateDisk + Frequenza +
##     Pixel + Ram, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1838.5  -211.8   -28.2   169.3  1894.4
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.491e+02  6.691e+01  -2.229  0.02602 *
## CompanyApple         2.826e+02  9.043e+01   3.125  0.00182 **
## CompanyAsus          5.438e+01  4.587e+01   1.185  0.23609
## CompanyChuwi        -7.683e+01  2.082e+02  -0.369  0.71213
## CompanyDell          1.124e+02  4.132e+01   2.720  0.00662 **
## CompanyFujitsu       5.168e+01  2.071e+02   0.250  0.80294
## CompanyGoogle        3.062e+02  2.105e+02   1.455  0.14602
## CompanyHP            2.045e+02  4.134e+01   4.947 8.54e-07 ***
## CompanyHuawei        5.510e+01  2.539e+02   0.217  0.82822
## CompanyLenovo        1.260e+02  4.108e+01   3.066  0.00221 **
## CompanyLG            6.759e+02  2.090e+02   3.235  0.00125 **
## CompanyMediacom     -1.108e+02  1.392e+02  -0.796  0.42603
## CompanyMicrosoft     2.369e+02  1.515e+02   1.564  0.11807    
## CompanyMSI           2.046e+02  6.686e+01   3.061  0.00225 ** 
## CompanyRazer         1.085e+03  1.428e+02   7.594 5.95e-14 ***
## CompanySamsung       9.436e+01  1.246e+02   0.757  0.44896    
## CompanyToshiba       2.871e+02  6.306e+01   4.553 5.79e-06 ***
## CompanyVero          1.440e+01  1.811e+02   0.080  0.93663    
## CompanyXiaomi       -1.743e+01  1.808e+02  -0.096  0.92322    
## TypeNameGaming      -2.977e+01  4.812e+01  -0.619  0.53621    
## TypeNameNetbook     -1.142e+02  7.947e+01  -1.437  0.15105    
## TypeNameNotebook    -2.440e+02  3.642e+01  -6.700 3.11e-11 ***
## TypeNameUltrabook    9.405e+01  4.338e+01   2.168  0.03034 *  
## TypeNameWorkstation  7.172e+02  7.500e+01   9.562  < 2e-16 ***
## SolidStateDiskTrue   1.997e+02  2.432e+01   8.212 5.28e-16 ***
## Frequenza            1.701e+02  2.320e+01   7.335 3.94e-13 ***
## Pixel                8.315e-05  8.292e-06  10.028  < 2e-16 ***
## Ram                  6.541e+01  2.578e+00  25.368  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 352.9 on 1275 degrees of freedom
## Multiple R-squared:  0.7504, Adjusted R-squared:  0.7452 
## F-statistic:   142 on 27 and 1275 DF,  p-value: < 2.2e-16
```

```
drop1(lmK, .~., test="F")
```

```
## Single term deletions
## 
## Model:
## Price ~ Company + TypeName + SolidStateDisk + Frequenza + Pixel + 
##     Ram
##                 Df Sum of Sq       RSS   AIC  F value    Pr(>F)    
## <none>                       158760389 15315                       
## Company         18  13404444 172164833 15384   5.9806 4.092e-14 ***
## TypeName         5  35077529 193837917 15565  56.3413 < 2.2e-16 ***
## SolidStateDisk   1   8397143 167157532 15380  67.4372 5.281e-16 ***
## Frequenza        1   6698755 165459144 15367  53.7975 3.940e-13 ***
## Pixel            1  12521049 171281438 15412 100.5562 < 2.2e-16 ***
## Ram              1  80130237 238890626 15845 643.5236 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
ls=lsmeans(lmK,
           pairwise ~ Company ,
           adjust="tukey")
c= contrast(ls, method = "eff")
#c #FIXME: too long to be printed
```

```
data$LogPrice=NULL
data$Product=NULL
data$X=NULL
str(data)
```

```
## 'data.frame':    1303 obs. of  15 variables:
##  $ Company         : Factor w/ 19 levels "Acer","Apple",..: 2 2 8 2 2 1 2 2 3 1 ...
##  $ TypeName        : Factor w/ 6 levels "2 in 1 Convertible",..: 5 5 4 5 5 4 5 5 5 5 ...
##  $ Inches          : num  13.3 13.3 15.6 15.4 13.3 15.6 15.4 13.3 14 14 ...
##  $ ScreenResolution: Factor w/ 40 levels "1366x768","1440x900",..: 24 2 9 26 24 1 26 2 9 16 ...
##  $ Cpu             : Factor w/ 118 levels "AMD A10-Series 9600P 2.4GHz",..: 55 53 64 75 57 15 74 53 9
##  $ Ram             : num  8 8 8 16 8 4 16 8 16 8 ...
##  $ Memory          : Factor w/ 39 levels "1.0TB HDD","1.0TB Hybrid",..: 5 3 17 30 17 27 16 16 30 17
##  $ Gpu             : Factor w/ 110 levels "AMD FirePro W4190M",..: 59 52 54 10 60 18 61 52 98 62 ...
##  $ OpSys           : Factor w/ 9 levels "Android","Chrome OS",..: 5 5 6 5 5 7 4 5 7 7 ...
##  $ Weight          : num  1.37 1.34 1.86 1.83 1.37 2.1 2.04 1.34 1.3 1.6 ...
##  $ Price           : num  1340 899 575 2537 1804 ...
##  $ Frequenza       : num  2.3 1.8 2.5 2.7 3.1 3 2.2 1.8 1.8 1.6 ...
##  $ Risoluzione     : Factor w/ 15 levels "1366x768","1440x900",..: 11 2 4 13 11 1 13 2 4 4 ...
##  $ Pixel           : int  4096000 1296000 2073600 5184000 4096000 1049088 5184000 1296000 2073600 207
##  $ SolidStateDisk  : Factor w/ 2 levels "False","True": 2 1 2 2 2 1 1 1 2 2 ...
```

```r
lm_full = lm(Price ~ ., data = data)
#summary(lm_full) #FIXME: wayyy too long to be printed, R^2 =0.9586
anova(lm_full, test="F")
```

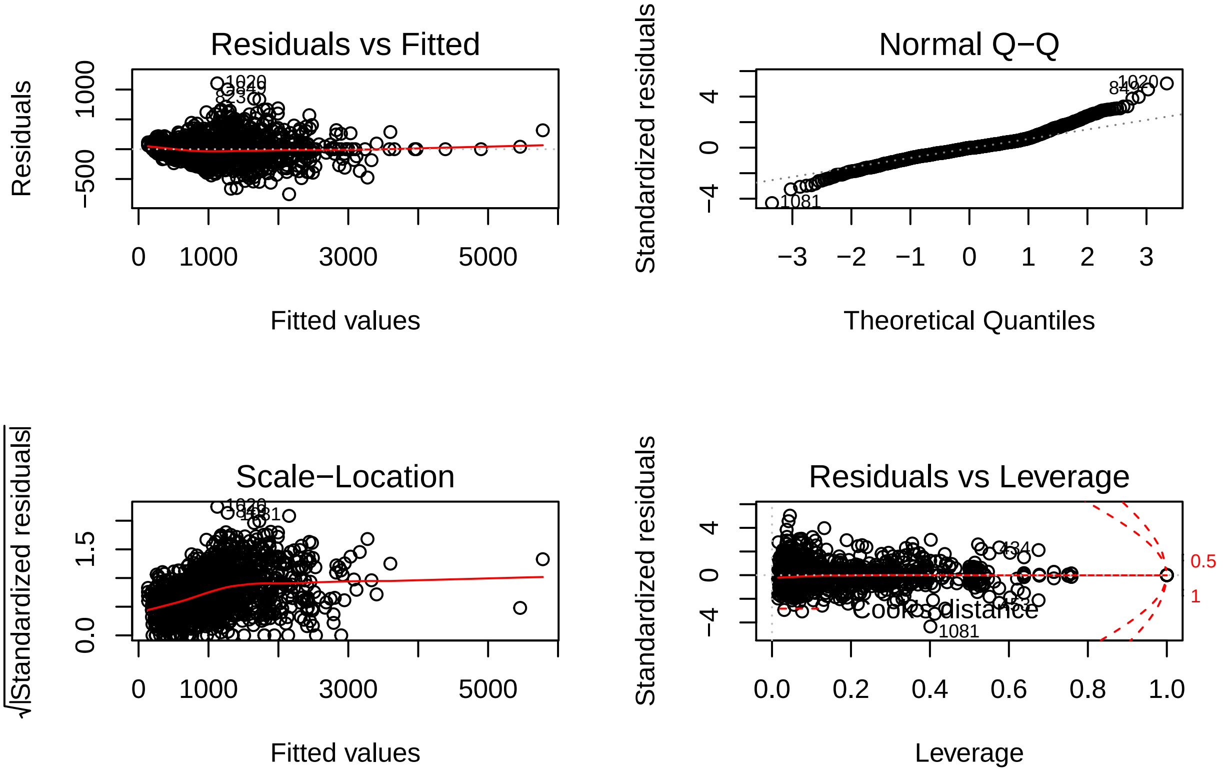
```
## Analysis of Variance Table
##
## Response: Price
##                     Df    Sum Sq  Mean Sq  F value    Pr(>F)
## Company             18 104013991  5778555 114.5882 < 2.2e-16 ***
## TypeName             5 182262038 36452408 722.8478 < 2.2e-16 ***
## Inches               1   6163570  6163570 122.2230 < 2.2e-16 ***
## ScreenResolution    36 108074619  3002073  59.5308 < 2.2e-16 ***
## Cpu                110  95329933   866636  17.1853 < 2.2e-16 ***
## Ram                  1  34947028 34947028 692.9963 < 2.2e-16 ***
## Memory              35  17134540   489558   9.7079 < 2.2e-16 ***
## Gpu                 88  34242874   389124   7.7163 < 2.2e-16 ***
## OpSys                6   3526085   587681  11.6537 1.198e-12 ***
## Weight               1       973      973   0.0193    0.8895
## Residuals         1001  50479311    50429
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
drop1(lm_full, test="F")
```

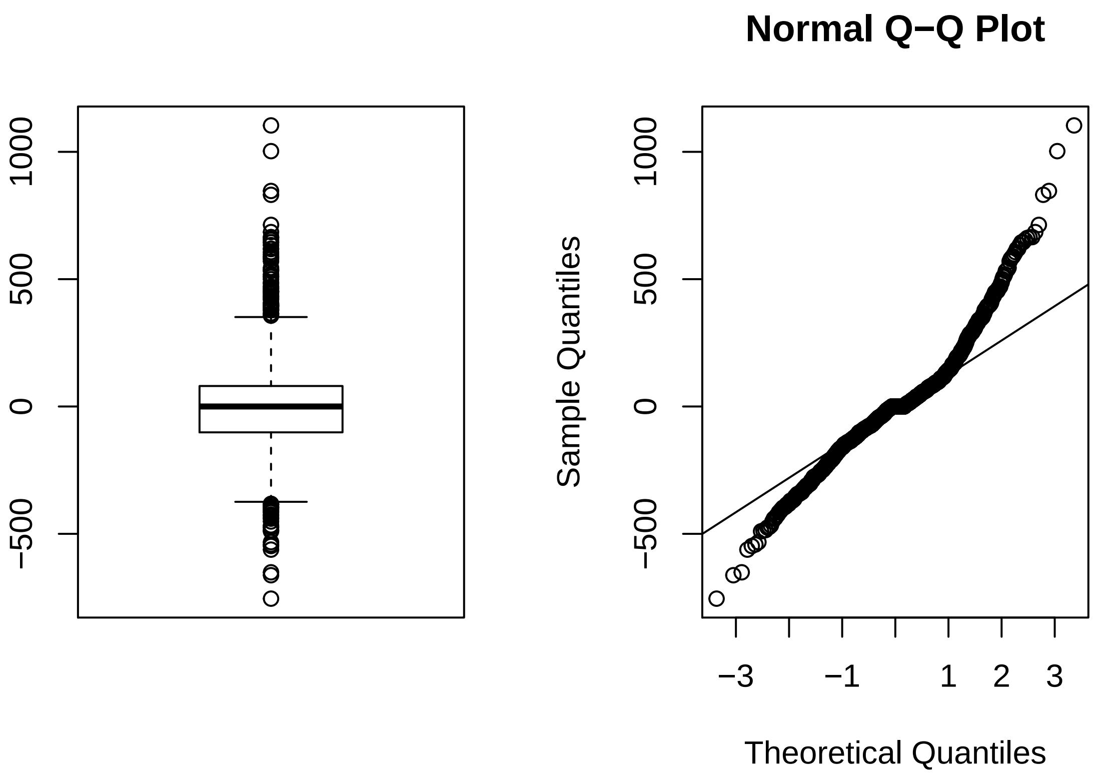
```
## Single term deletions
##
## Model:
## Price ~ Company + TypeName + Inches + ScreenResolution + Cpu +
##     Ram + Memory + Gpu + OpSys + Weight + Frequenza + Risoluzione +
##     Pixel + SolidStateDisk
##                  Df Sum of Sq      RSS   AIC F value    Pr(>F)
## <none>                        50479311 14370
## Company          14   6197922 56677232 14493  8.7789 < 2.2e-16 ***
## TypeName          5   3685931 54165241 14452 14.6183  7.50e-14 ***
## Inches            1    210134 50689445 14373  4.1669   0.04148 *
## ScreenResolution 23   5322877 55802188 14454  4.5892  8.09e-12 ***
## Cpu              88  16408116 66887427 14560  3.6974 < 2.2e-16 ***
## Ram               1   4481351 54960662 14479 88.8648 < 2.2e-16 ***
## Memory           34  10507055 60986365 14548  6.1281 < 2.2e-16 ***
## Gpu              88  30459868 80939179 14809  6.8638 < 2.2e-16 ***
## OpSys             6   3518495 53997806 14446 11.6286  1.28e-12 ***
## Weight            1       973 50480284 14368  0.0193   0.88953
## Frequenza          0          0 50479311 14370
## Risoluzione        0          0 50479311 14370
## Pixel              0          0 50479311 14370
## SolidStateDisk     0          0 50479311 14370
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#coefplot(lm_full, intercept=FALSE) #meglio di no ahah

par(mfrow=c(2,2))
plot(lm_full)
```

```
## Warning: not plotting observations with leverage one:
##   13, 15, 18, 29, 34, 46, 84, 128, 160, 173, 178, 179, 205, 232, 267, 271, 299, 303, 324, 348, 388,

## Warning: not plotting observations with leverage one:
##   13, 15, 18, 29, 34, 46, 84, 128, 160, 173, 178, 179, 205, 232, 267, 271, 299, 303, 324, 348, 388,

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



```
par(mfrow=c(1,1))
par(mfrow=c(1,2))
boxplot(lm_full$residuals)
qqnorm(lm_full$residuals);qqline(lm_full$residuals) # probably the correction would work pretty fine he
```



```
#tests
ad.test(lm_full$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  lm_full$residuals
## A = 19.821, p-value < 2.2e-16
```

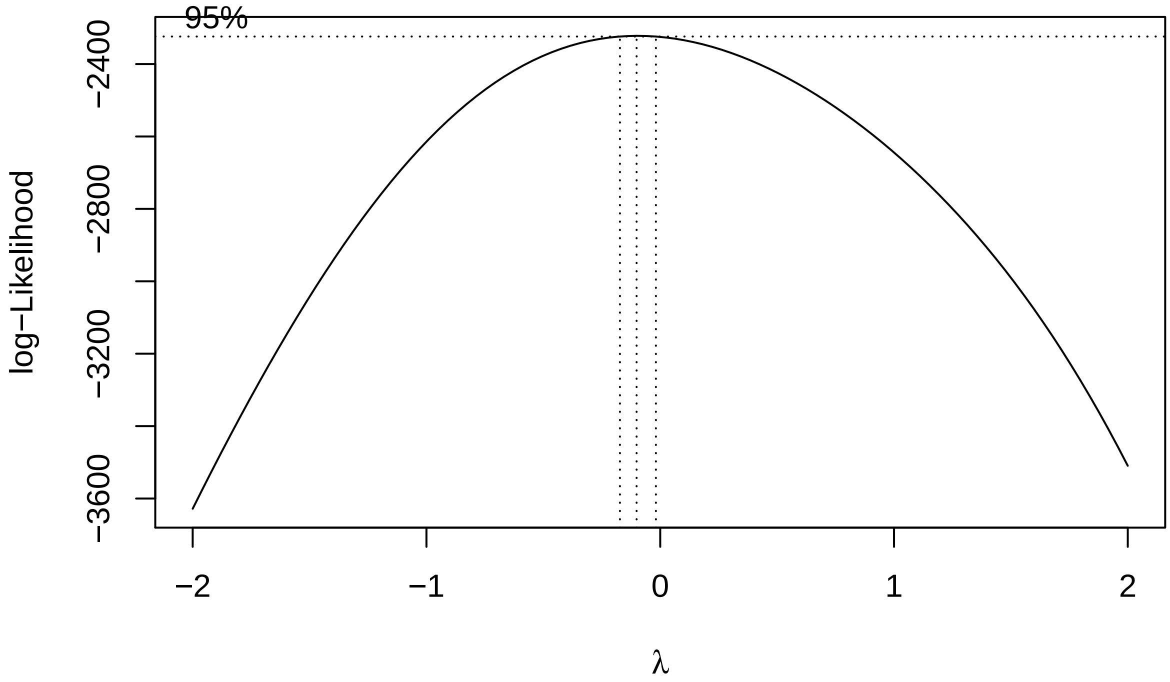
```
shapiro.test(lm_full$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lm_full$residuals
## W = 0.94917, p-value < 2.2e-16
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:EnvStats':
##
##     boxcox
```

```
boxcoxreg1<-boxcox(lm_full)
```



```
which.max(boxcoxreg1$y)
```

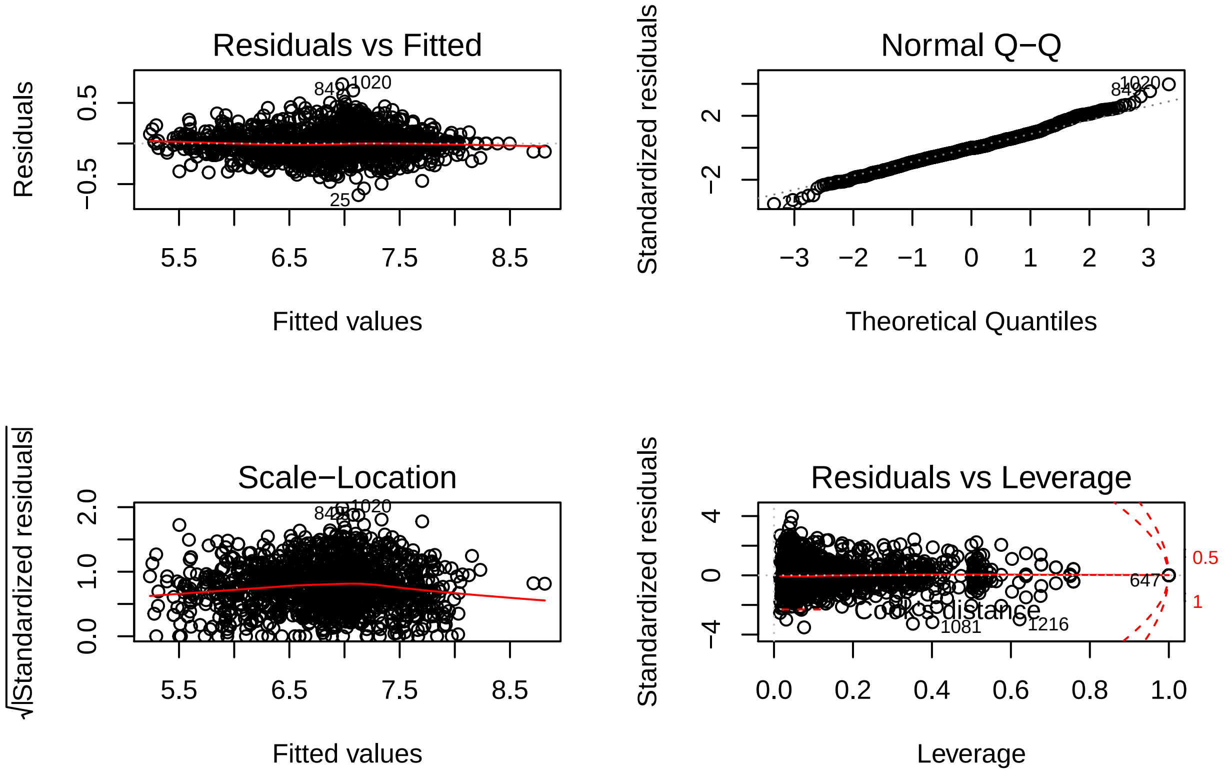
```
## [1] 48
```

```
lambda=boxcoxreg1$x[which.max(boxcoxreg1$y)]
lambda
```

```
## [1] -0.1010101
```

```
lm_full_t = lm(log(Price) ~ ., data = data)

par(mfrow=c(2,2))
plot(lm_full_t) #quite better
```

```
## Warning: not plotting observations with leverage one:
##   13, 15, 18, 29, 34, 46, 84, 128, 160, 173, 178, 179, 205, 232, 267, 271, 299, 303, 324, 348, 388,

## Warning: not plotting observations with leverage one:
##   13, 15, 18, 29, 34, 46, 84, 128, 160, 173, 178, 179, 205, 232, 267, 271, 299, 303, 324, 348, 388,

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



```
ad.test(lm_full_t$residuals) #not really
```

```
##
##  Anderson-Darling normality test
##
## data:  lm_full_t$residuals
## A = 7.5169, p-value < 2.2e-16
```

```
shapiro.test(lm_full_t$residuals)  #not really
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lm_full_t$residuals
## W = 0.98478, p-value = 1.874e-10
```

A look over outliers

```
library(car)
```

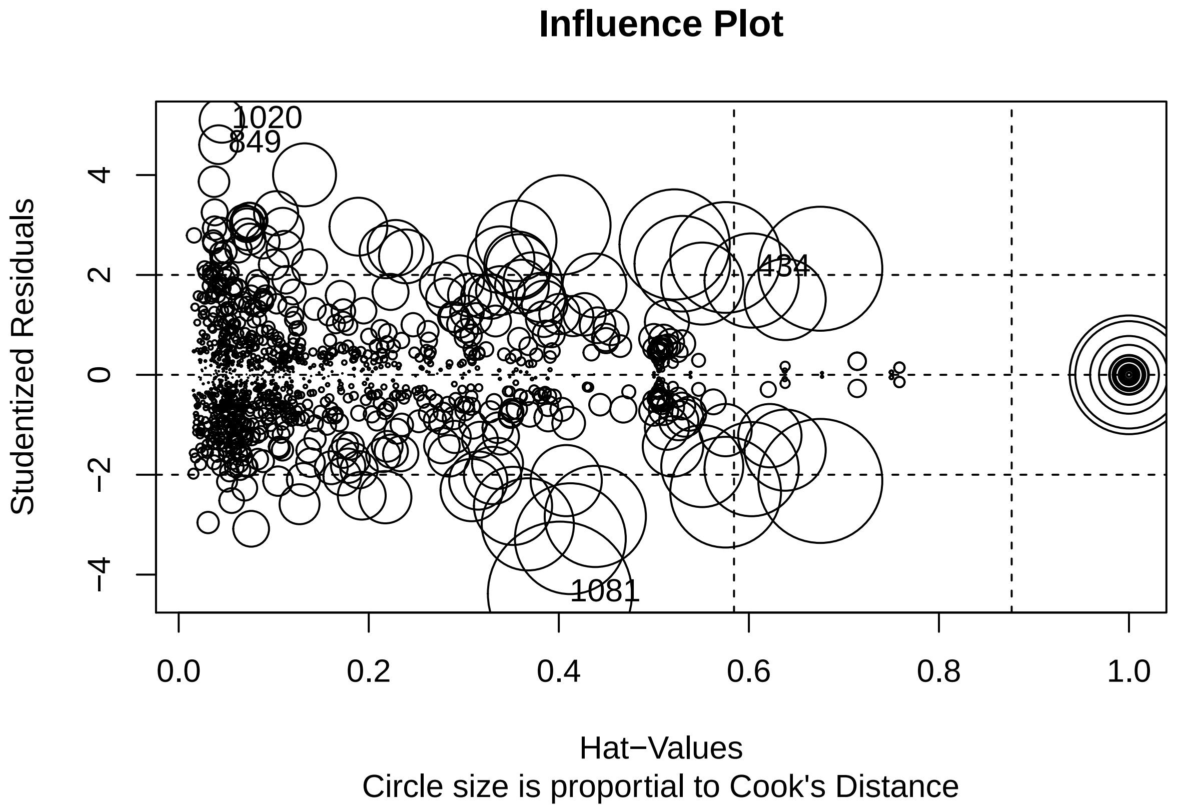
```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:EnvStats':
##
##     qqPlot
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

```r
influencePlot(lm_full,main="Influence Plot", sub="Circle size is proportial to Cook's Distance" )
```

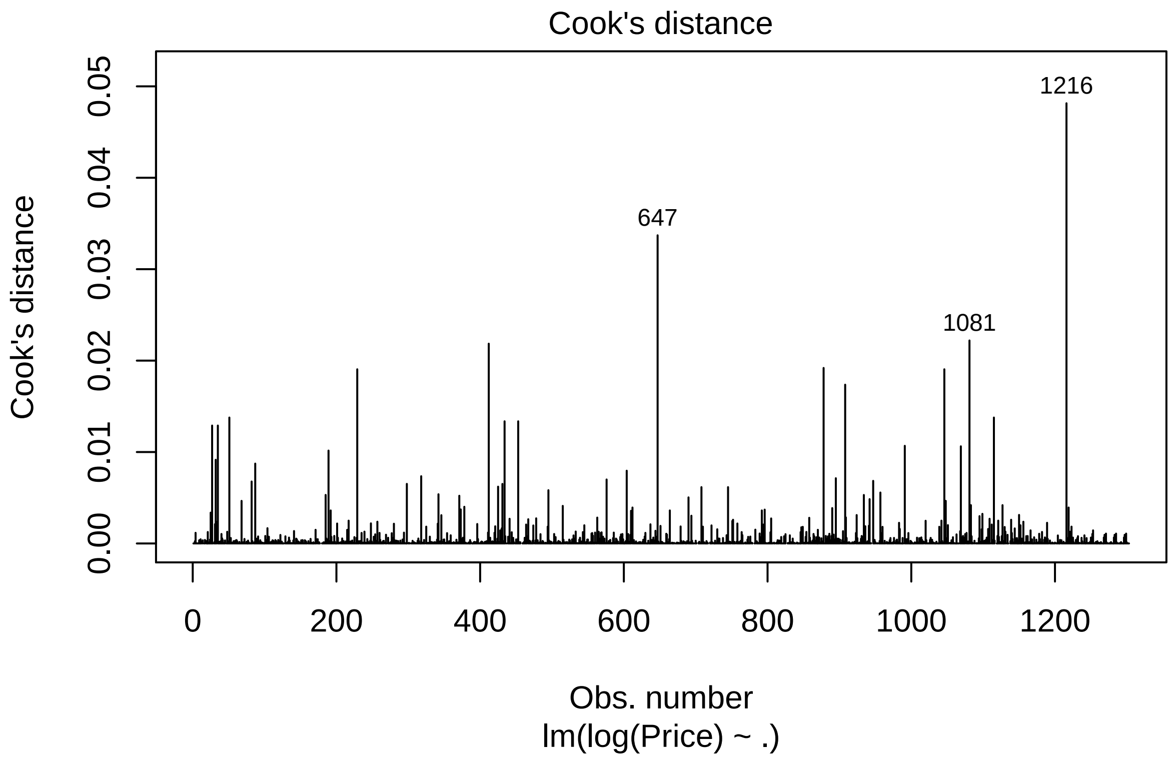


```
##         StudRes        Hat       CookD
## 13          NaN 1.00000000         NaN
## 15          NaN 1.00000000         NaN
## 434   2.124383 0.67519240 0.030955546
## 849   4.607995 0.04198279 0.003020118
## 1020  5.093056 0.04545388 0.003990584
## 1081 -4.381193 0.40114343 0.041814955
```

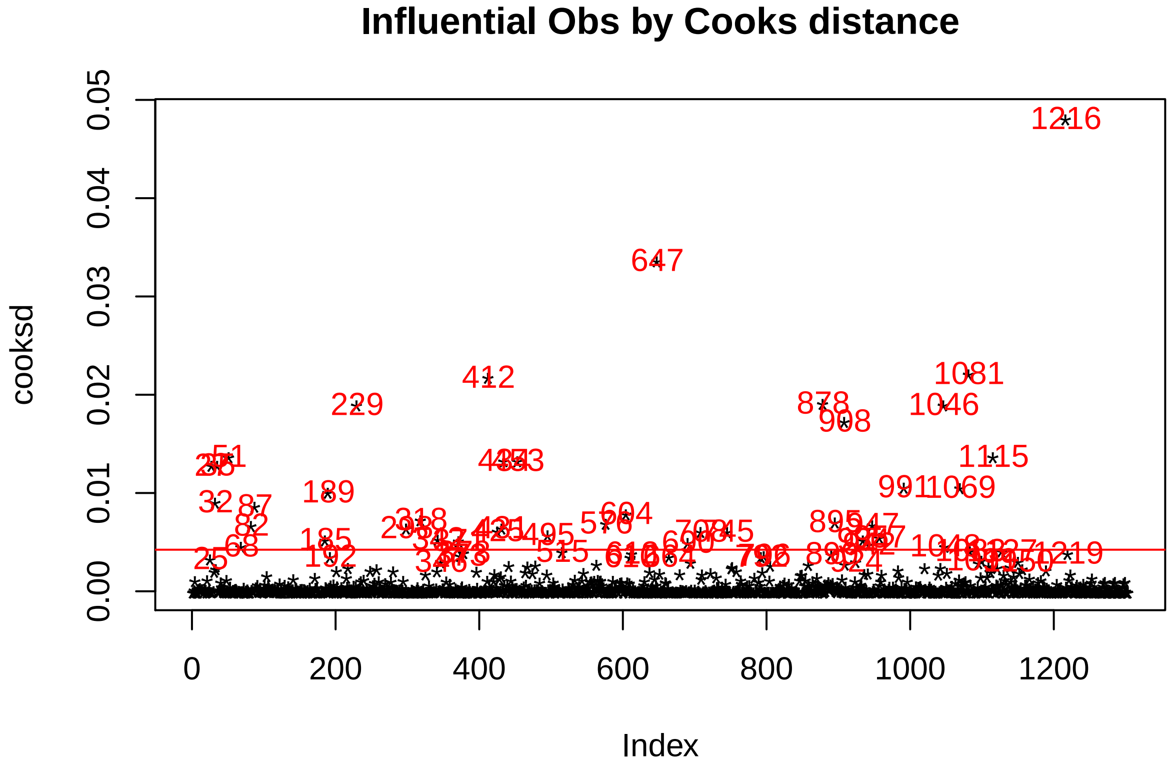
```r
#Cook's Distance
cooksd <- cooks.distance(lm_full_t)
cooksda=data.frame(cooksd)
summary(cooksd)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
## 0.00000 0.00003 0.00013 0.00076 0.00053 0.04815      88
```

```r
# identify D values > 4/(n-k-1)
# Cook's D plot
cutoff <- 4/((nrow(data)-length(lm_full_t$coefficients)-2))
plot(lm_full_t, which=4, cook.levels=cutoff)
```

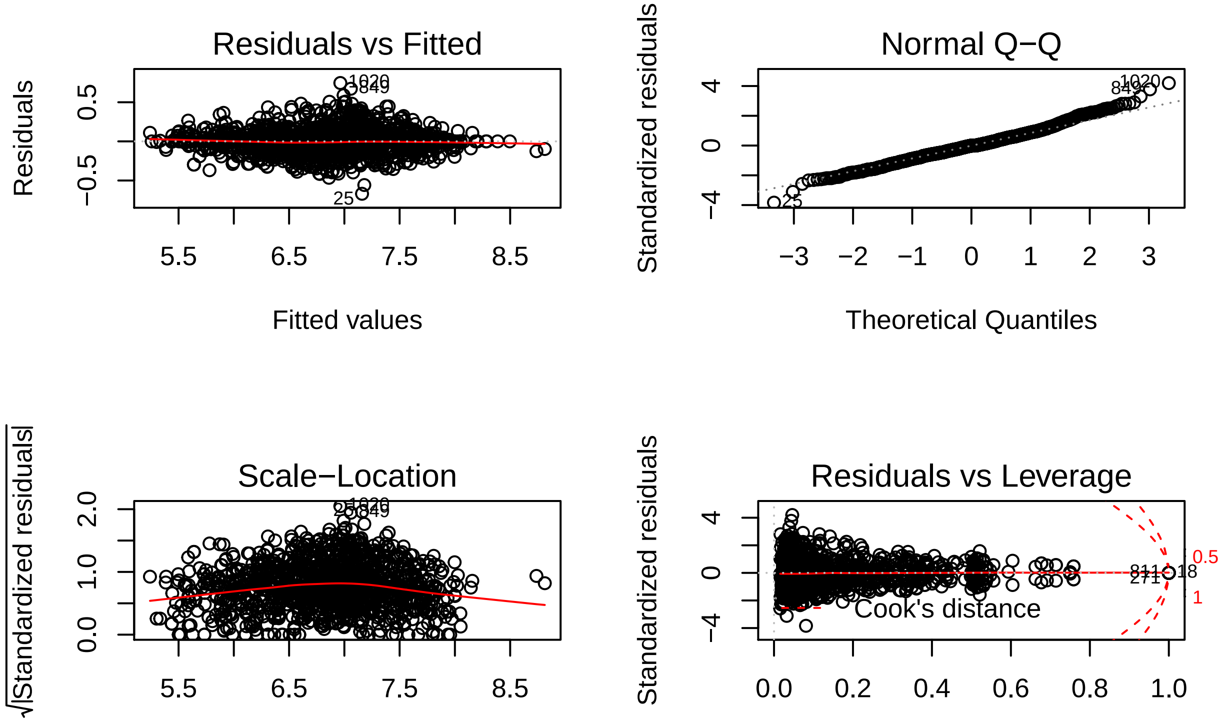


```
plot(cooksd, pch="*", cex=1, main="Influential Obs by Cooks distance") # plot cook's distance
abline(h = cutoff, col="red") # add cutoff line
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4*mean(cooksd, na.rm=T),names(cooksd),""),
col="red")#add labels
```



```r
#extract influencial obs
influential <- as.numeric(names(cooksd)[(cooksd > cutoff)]) # influential row numbers
influ=data.frame(data[cooksd > cutoff, ])
filtered_data <- data[ !(row.names(data) %in% c(influential)), ]
#Outlier rimossi
lm_full_t_no_OUTliers = lm(log(Price) ~ ., data = filtered_data)
par(mfrow=c(2,2))
plot(lm_full_t_no_OUTliers)
```

```
## Warning: not plotting observations with leverage one:
##   15, 21, 32, 43, 148, 172, 196, 222, 288, 292, 312, 335, 408, 420, 430, 438, 439, 447, 456, 481, 485

## Warning: not plotting observations with leverage one:
##   15, 21, 32, 43, 148, 172, 196, 222, 288, 292, 312, 335, 408, 420, 430, 438, 439, 447, 456, 481, 485

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



```r
#summary(lm_full_t_no_OUTliers) #FIXME: too long to be printed, R^2=0.9727
ncvTest(lm_full_t_no_OUTliers)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 1.740444, Df = 1, p = 0.18708
```

```r
null = lm(log(Price) ~ 1, data = filtered_data)
full = lm(log(Price) ~ ., data = filtered_data)
lm_fit = stepAIC(null, scope = list(upper = full), direction = "both", trace = FALSE)
drop1(lm_fit, test = 'F')
```

```
## Single term deletions
##
## Model:
## log(Price) ~ Cpu + Memory + OpSys + Gpu + TypeName + ScreenResolution +
##     Company + Ram + Inches
##                  Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>                        32.161 -4053.8
## Cpu              84   15.5056 47.667 -3724.9  5.5789 < 2.2e-16 ***
## Memory           34    9.9754 42.136 -3780.6  8.8673 < 2.2e-16 ***
## OpSys             5    5.9181 38.079 -3850.5 35.7726 < 2.2e-16 ***
## Gpu              85   10.7290 42.890 -3860.2  3.8149 < 2.2e-16 ***
## TypeName          4    2.1668 34.328 -3979.5 16.3717 5.483e-13 ***
## ScreenResolution 28    4.7325 36.893 -3936.4  5.1082 5.348e-16 ***
## Company          14    3.0807 35.242 -3966.3  6.6506 4.801e-13 ***
## Ram               1    1.5499 33.711 -3996.4 46.8425 1.360e-11 ***
## Inches            1    1.0393 33.200 -4015.7 31.4100 2.720e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```