Linear Regression for Business

Contents

Extract, Transform, Load (ETL)

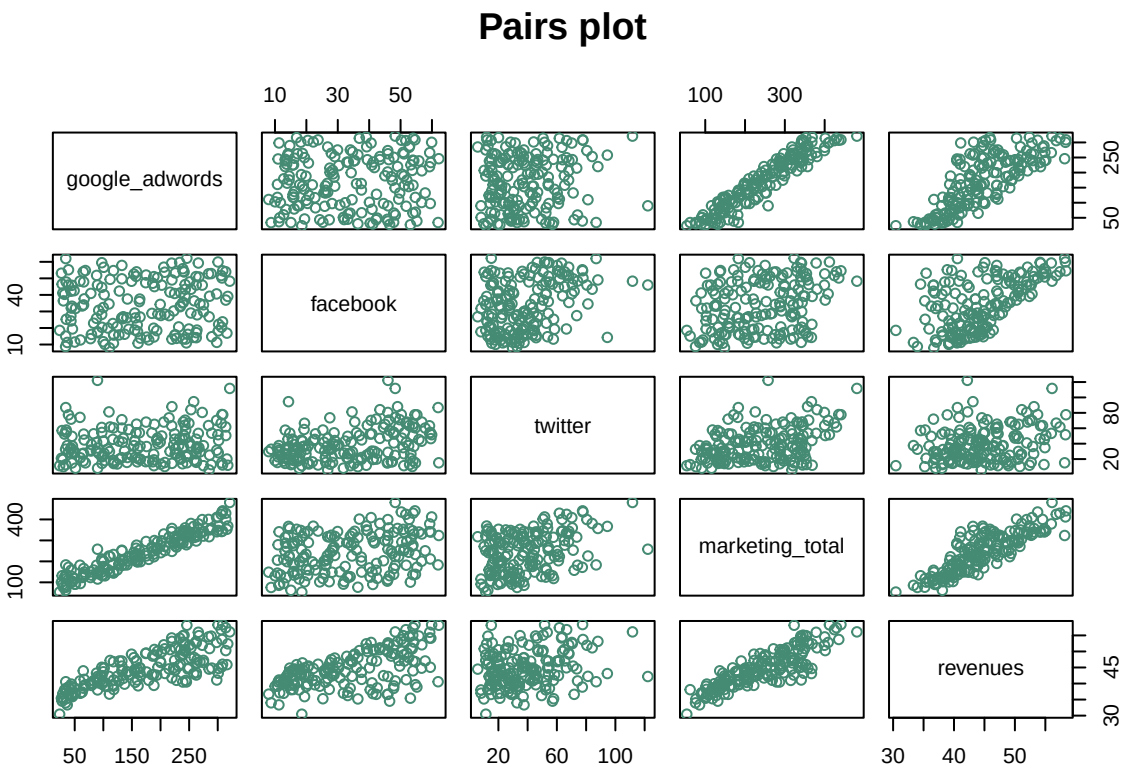
```
adverts = read.csv('marketing.csv')
```

Exploratory Data Analysis (EDA)

```
str(adverts)
```

```
## 'data.frame':    172 obs. of  5 variables:
##  $ google_adwords : num  65.7 39.1 174.8 34.4 78.2 ...
##  $ facebook       : num  47.9 55.2 52 62 40.9 ...
##  $ twitter        : num  52.5 77.4 68 86.9 30.4 ...
##  $ marketing_total: num  166 172 295 183 150 ...
##  $ revenues       : num  39.3 38.9 49.5 40.6 40.2 ...
```

```
pairs(adverts, main = 'Pairs plot', col = 'aquamarine4')
```



Model Training

Simple linear regression with only 1 response and 1 predictor variable.

```
model = lm(revenues~marketing_total, data = adverts)
model
```

```
##
## Call:
## lm(formula = revenues ~ marketing_total, data = adverts)
##
## Coefficients:
##     (Intercept)  marketing_total
##        32.00670          0.05193
```

Interpretation :

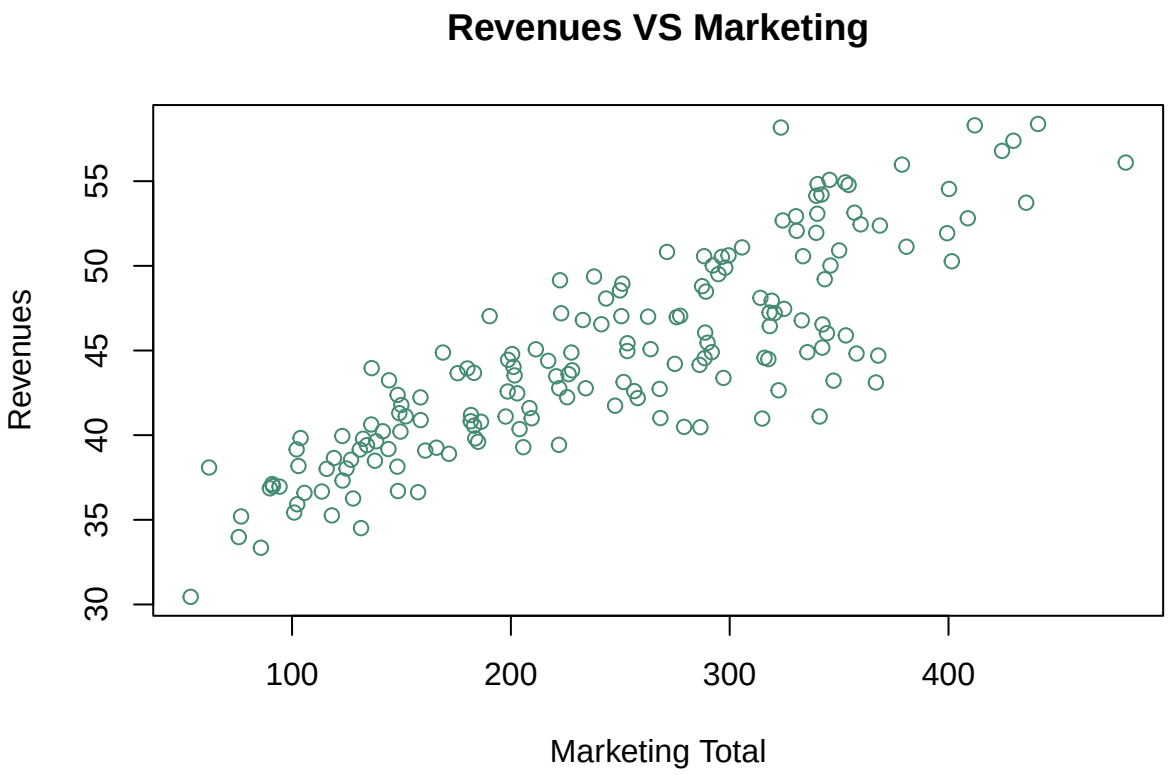
- revenues = 32.00670 + 0.05193 * marketing_total,
- revenues will increase by RM 51.93 for each RM 1,000 increase in the total marketing.
- revenue is RM 32, 007 when total marketing is RM 0

Model Assumption

Linearity

The relationship between response and predictor variable are linear. This can be validate through a scatter plot.

```r
plot(adverts$marketing_total, adverts$revenues, col = 'aquamarine4',
     main = 'Revenues VS Marketing', xlab = 'Marketing Total', ylab = 'Revenues')
```



Independence

The relationship between variables is independent of one another.

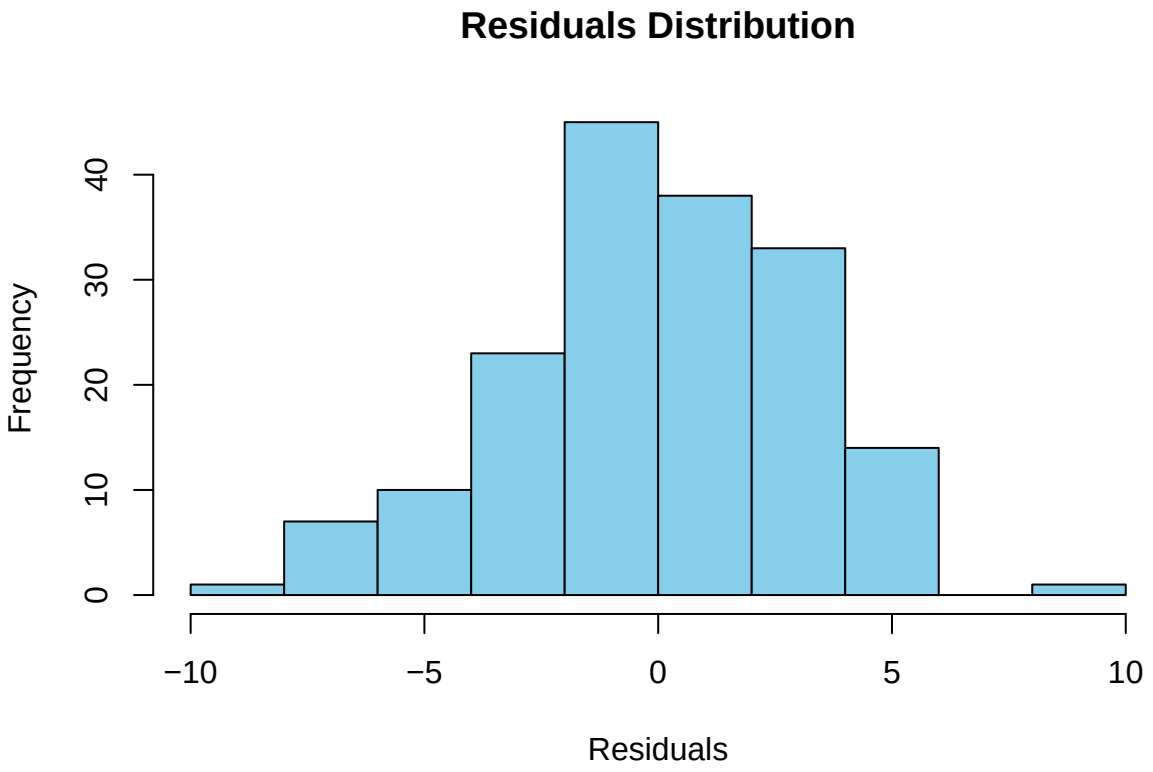
Normality

The residuals form a normal distribution around the regression line with a mean value of zero.

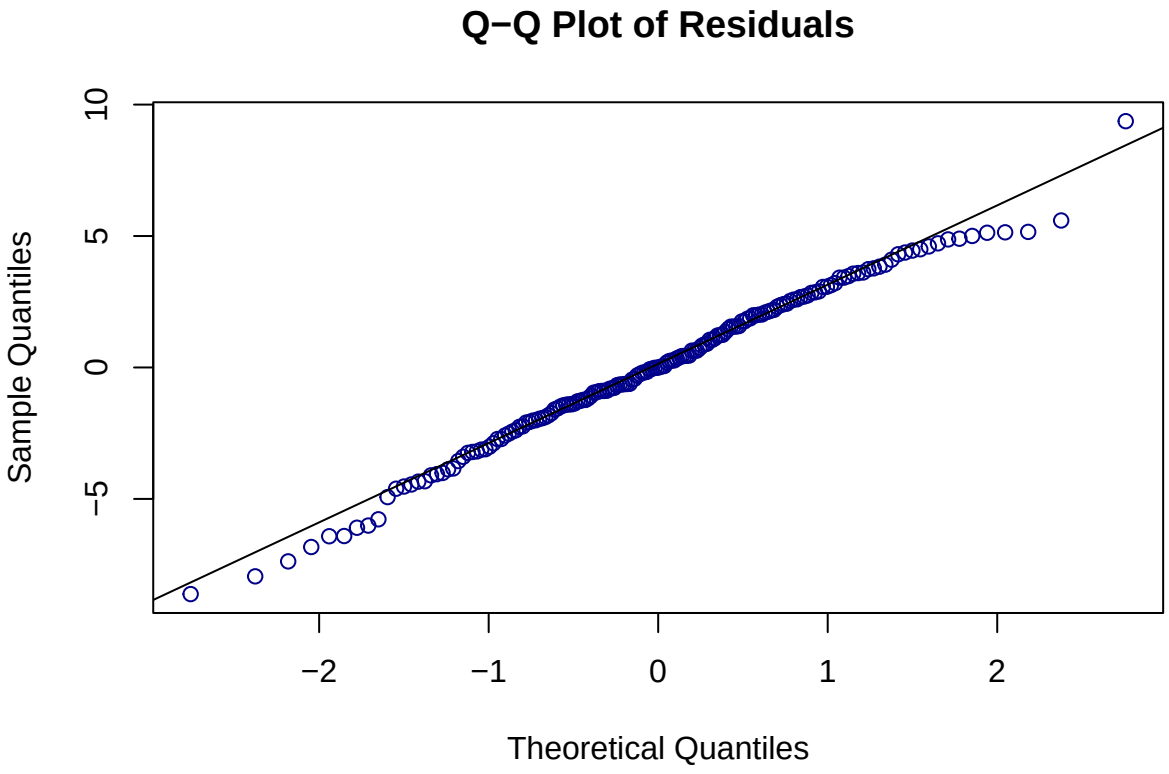
$$e \ N(0, \sigma 2)$$

This can be check by using histogram or qqplot,

```r
hist(model$residuals, xlab = 'Residuals', main = 'Residuals Distribution', col = 'skyblue')
```



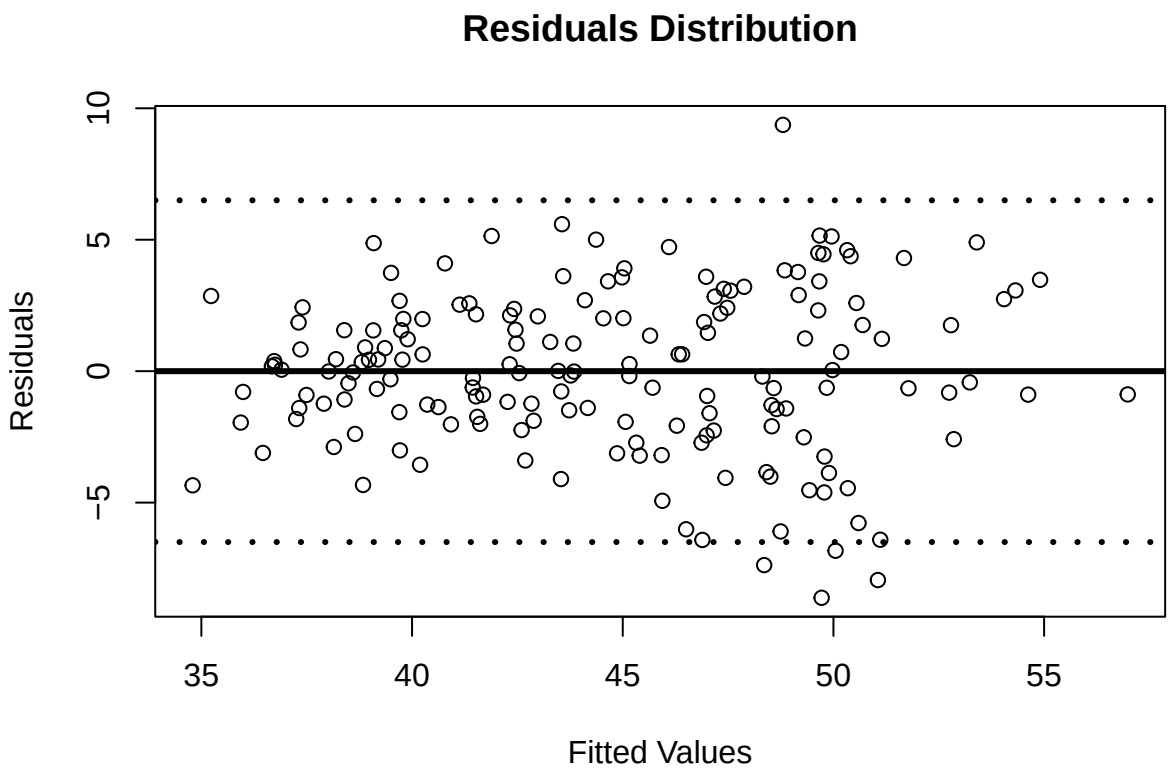

```r
qqnorm(model$residuals, main = 'Q-Q Plot of Residuals', col = 'darkblue')
qqline(model$residuals)
```



Equal Variance

Residuals form a random pattern distributed around a mean of 0

```r
plot(model$fitted.values, model$residuals, ylab = 'Residuals',
     xlab = 'Fitted Values', main = 'Residuals Distribution')
abline(0, 0, lwd = 3)
abline(h = c(-6.5,6.5), lwd = 3, lty = 3)
```



Model Evaluation

We can see the summary of the model

```r
summary(model)
```

```
##
## Call:
## lm(formula = revenues ~ marketing_total, data = adverts)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6197 -1.8963 -0.0006  2.1705  9.3689
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     32.006696   0.635590   50.36   <2e-16 ***
## marketing_total  0.051929   0.002437   21.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.054 on 170 degrees of freedom
## Multiple R-squared:  0.7277, Adjusted R-squared:  0.7261
## F-statistic: 454.2 on 1 and 170 DF,  p-value: < 2.2e-16
```

Interpretation:

- Residuals shows the median is around 0 shows that the distribution is relatively normal

- Both coefficients have p-value of <2e-16 which is way below alpha of 0.05. This indicates that the predictor variable is significant in predicting the response variable.
- Adjusted R-square of 0.7261 shows that the model is able to explain 72.61% of the error.
- The model p-value of <2e-16 which is way below alpha of 0.05 also shows that the overall model is significant.

Predicting

When predicting an output, it is the best practice to predict using value within the range

```
range(adverts$marketing_total)
```

```
## [1]  53.65 481.00
```

```
newdata = data.frame(marketing_total = 460)
predict.lm(model, newdata, level = 0.95, interval = 'predict')
```

```
##        fit      lwr      upr
## 1 55.89403 49.75781 62.03025
```

Sampling from Big Data

Take a random sample of 30% from the marketing data

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
set.seed(4510)
market_sample = sample_frac(adverts, 0.3, replace = FALSE)
samp_model = lm(revenues~marketing_total, data = market_sample)
samp_model
```

```
##
## Call:
## lm(formula = revenues ~ marketing_total, data = market_sample)
##
## Coefficients:
##     (Intercept)  marketing_total
##         33.0994           0.0458
```

```
confint(samp_model)
```

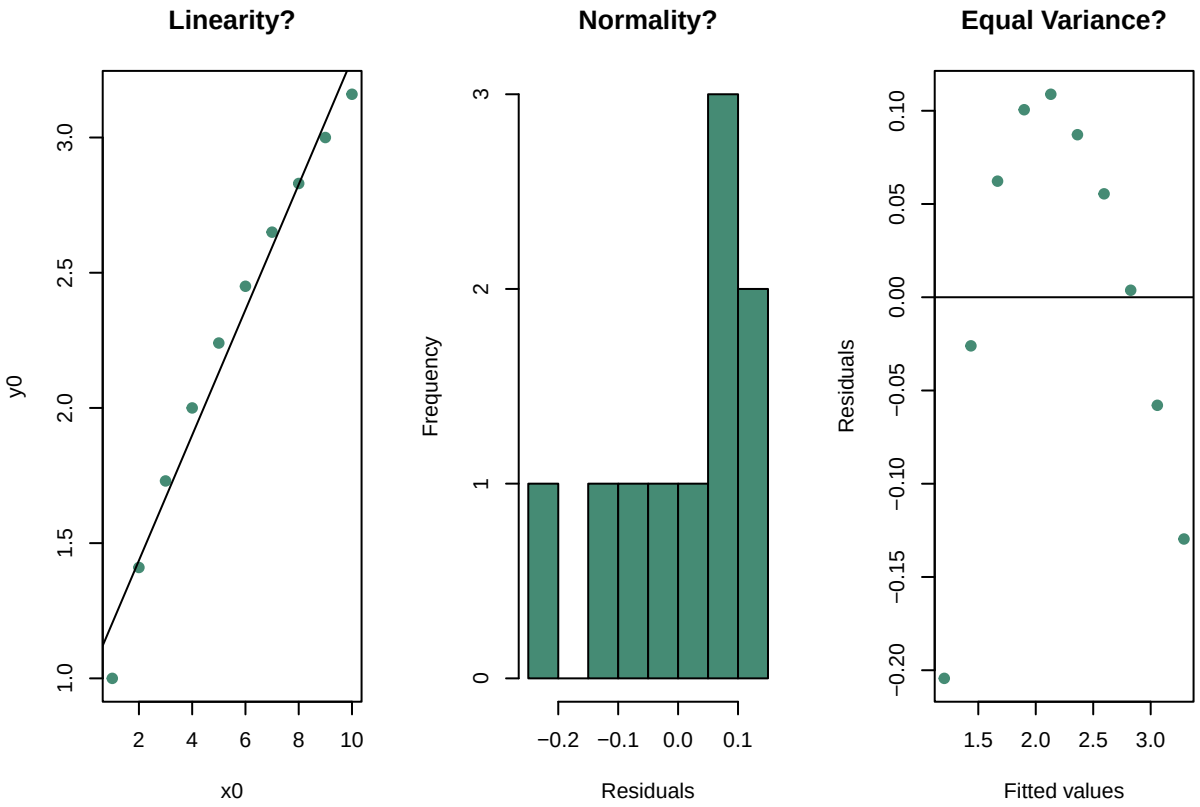
```
##                       2.5 %      97.5 %
## (Intercept)     30.82875906 35.37011129
## marketing_total  0.03644166  0.05515941
```

This shows that if you take 100 different samples from the population, then 95 out of 100 samples would estimate the slop of marketing_total between 0.03644166 and 0.05515941

Transforming data

We create a simulation of data that violate the LINE assumption.

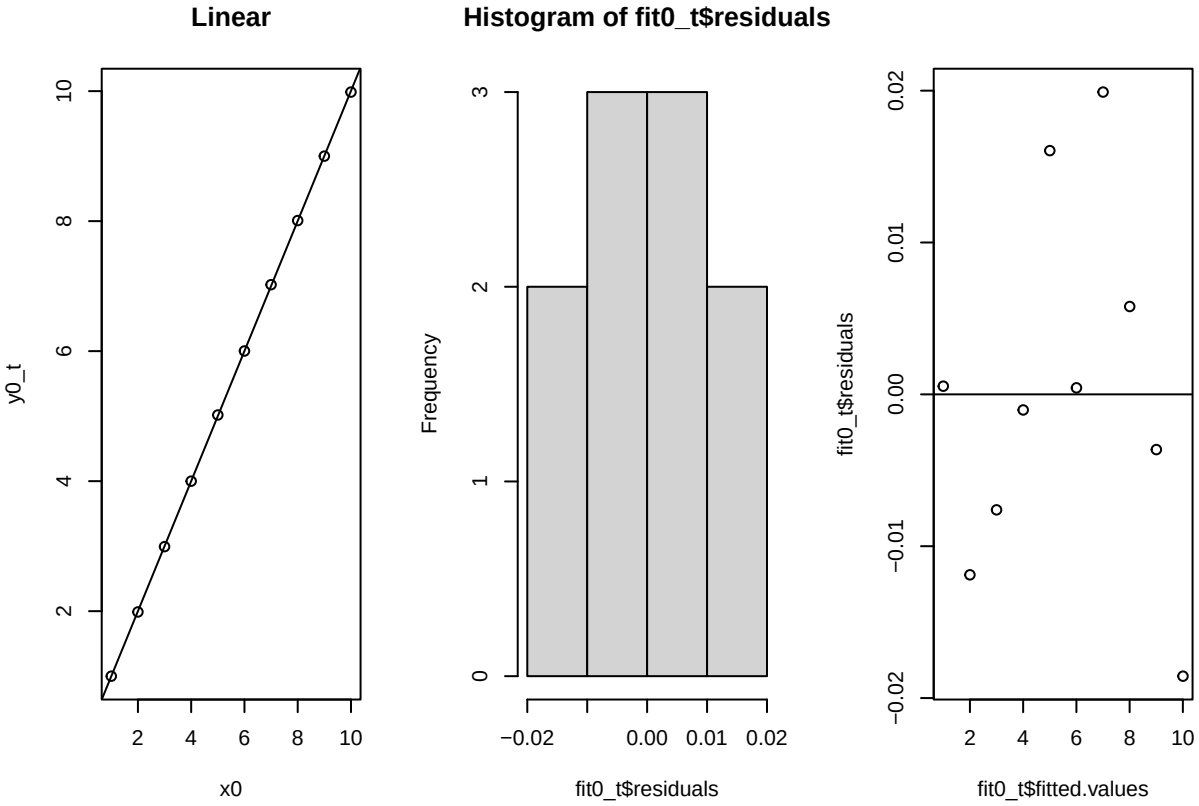
```
x0 <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
y0 <- c(1.00, 1.41, 1.73, 2.00, 2.24, 2.45, 2.65, 2.83, 3.00, 3.16)
fit0 <- lm(y0 ~ x0)
par(mfrow = c(1, 3))
plot(x0, y0, pch = 19, main = "Linearity?", col = 'aquamarine4'); abline(fit0)
hist(fit0$residuals, main = "Normality?", col = "aquamarine4", xlab = 'Residuals')
plot(fit0$fitted.values, fit0$residuals, main = "Equal Variance?", pch = 19,
     col = 'aquamarine4', xlab = 'Fitted values', ylab = 'Residuals')
abline(h = 0)
```



When assumptions are violated, here are the solution :

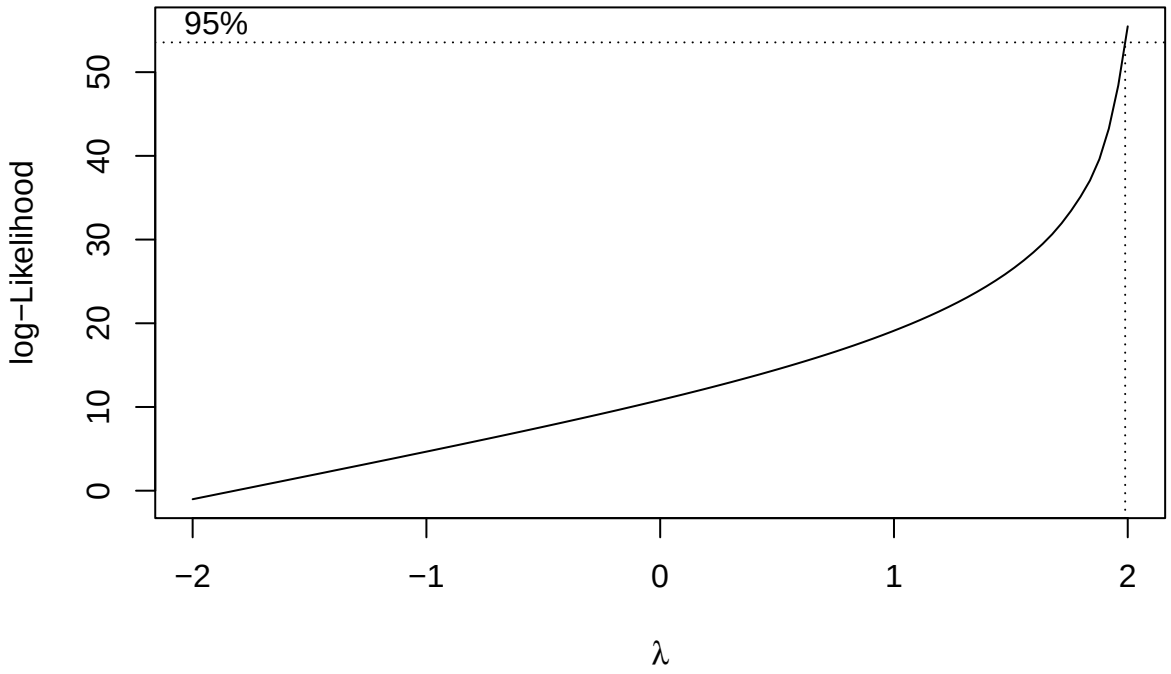
- **Not independent :** Use time series analysis
- **Not linear :** transform predictor variable
- **Not normal :** transform response variable
- **Not homoscedasticity :** transform response variable

```
y0_t = y0 ^ 2
fit0_t = lm(y0_t ~ x0)

par(mfrow=c(1,3))

plot(x0, y0_t, main = 'Linear')
abline(fit0_t)

hist(fit0_t$residuals)

plot(fit0_t$fitted.values, fit0_t$residuals)
abline(h = 0)
```



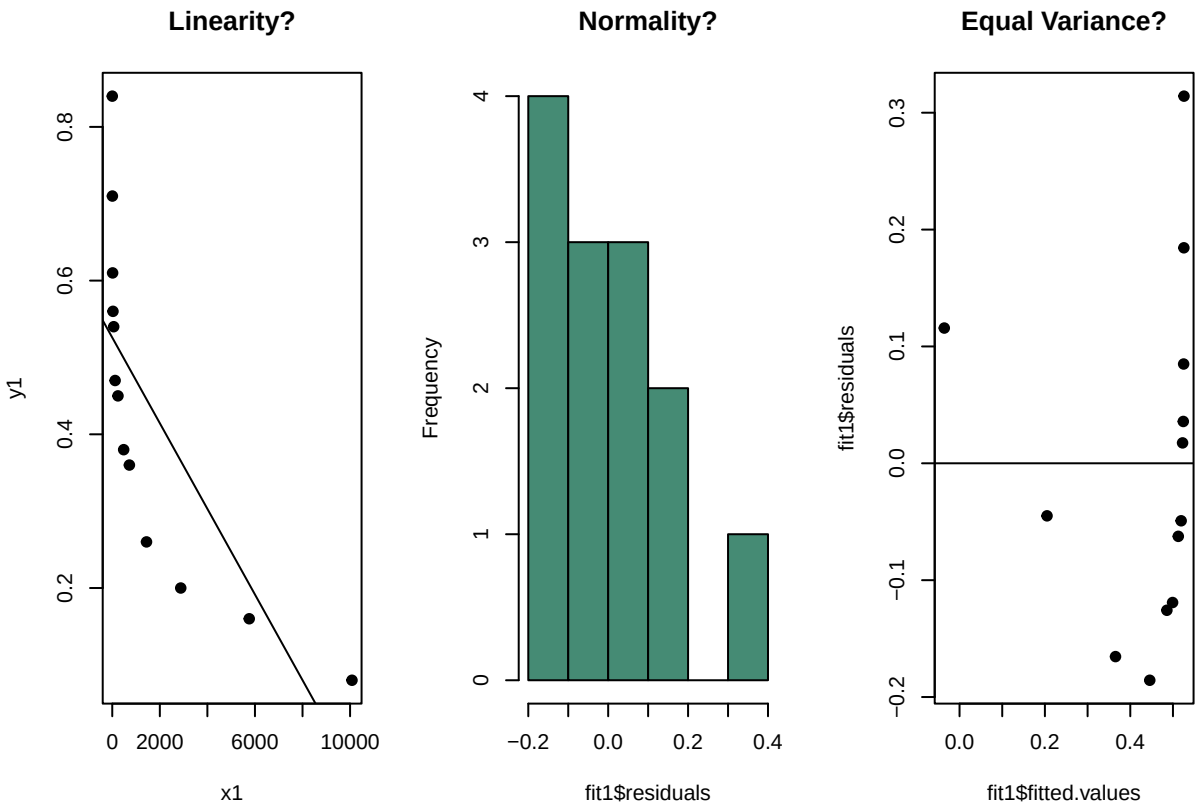
But how to know just how to know what is the appropriate transformation. So, we can use plot boxcox to find the appropriate transformation.

```
library(MASS)
boxcox(fit0)
```



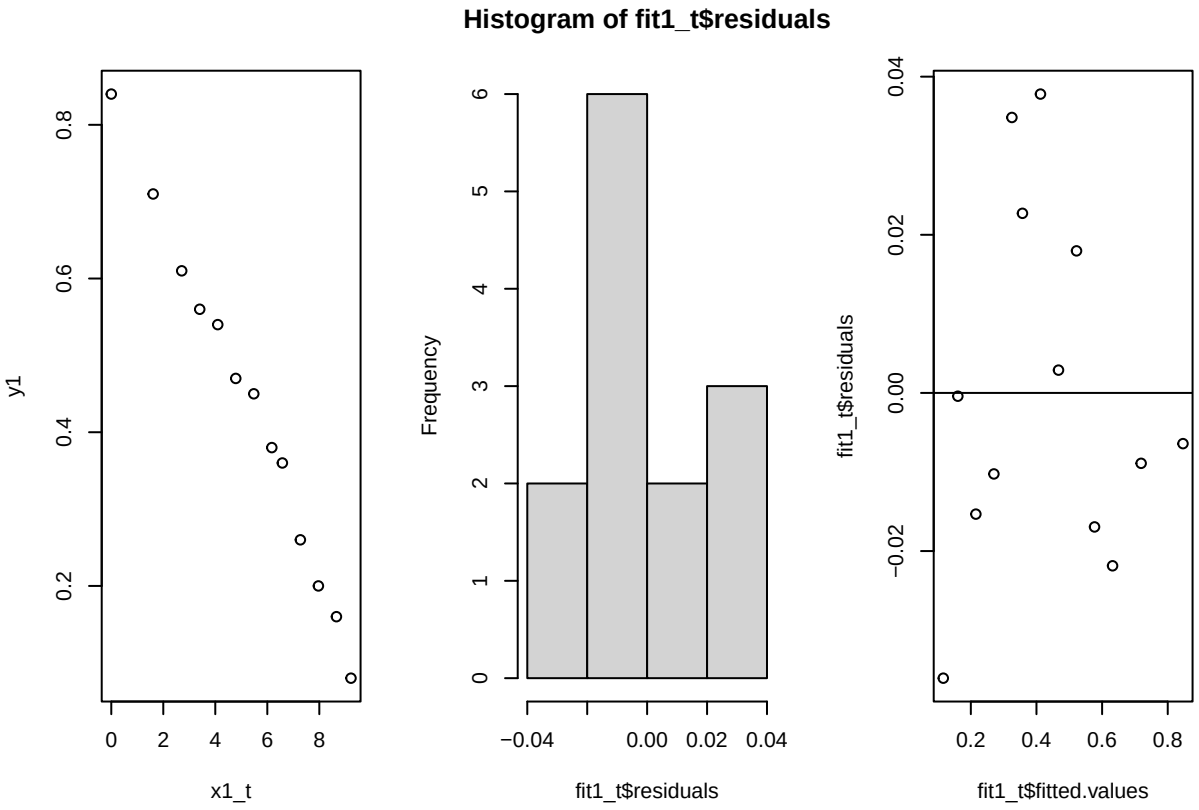
The graphical output shows that we can raise out response variable to 2. Now, we try to create another simulation which violate the linearity assumption.

```
x1 <- c(1, 5, 15, 30, 60, 120, 240, 480, 720, 1440, 2880, 5760, 10080)
y1 <- c(0.84, 0.71, 0.61, 0.56, 0.54, 0.47, 0.45, 0.38, 0.36, 0.26, 0.2, 0.16,
        0.08)
fit1 <- lm(y1 ~ x1)
par(mfrow=c(1,3))
plot(x1, y1, pch = 19, main = "Linearity?"); abline(fit1)
hist(fit1$residuals, main = "Normality?", col = "aquamarine4")
plot(fit1$fitted.values, fit1$residuals, main = "Equal Variance?", pch = 19)
abline(h = 0)
```



Unfortunately, boxcox() can't be used to transform predictor variable. Since, we can see that the plot of linearity has log pattern, we can start by transforming the independent variable using log()

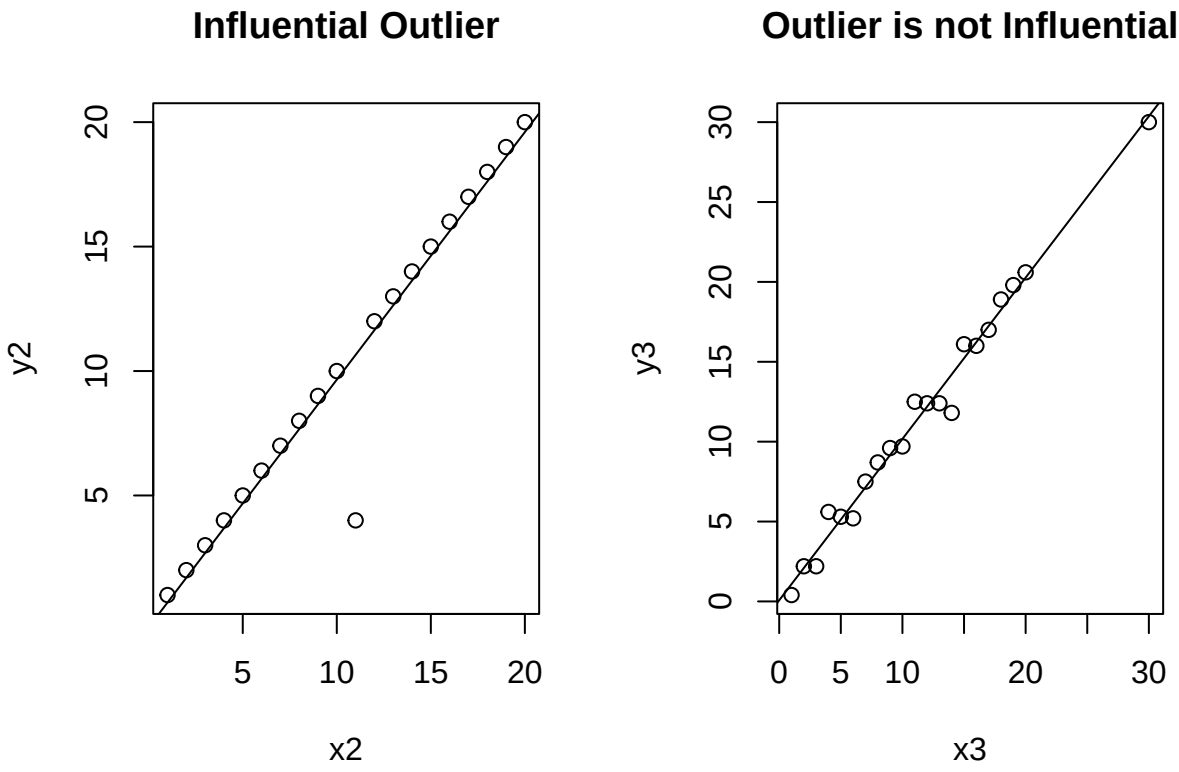
```
x1_t = log(x1)
fit1_t = lm(y1~x1_t)

par(mfrow=c(1,3))

plot(x1_t, y1)

hist(fit1_t$residuals)

plot(fit1_t$fitted.values, fit1_t$residuals);abline(h=0)
```



Now, we can see that it not only satisfy the linear assumption but the normality and homoscedasticity assumption on the same time.

Handling outlier

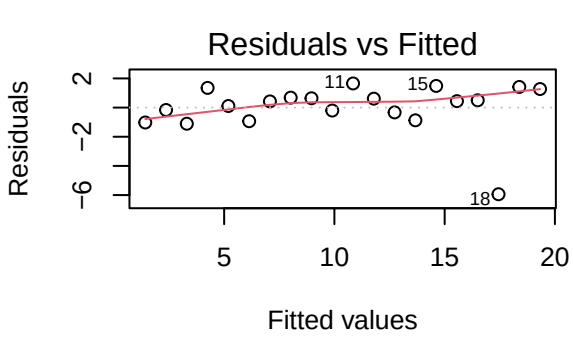
```
x2 <- 1:20
y2 <- c(1:10, 4, 12:20)
x3 <- c(1:20, 30)
y3 <- c(0.4, 2.2, 2.2, 5.6, 5.3, 5.2, 7.5, 8.7, 9.6, 9.7, 12.5, 12.4, 12.4, 11.8,
         16.1, 16, 17, 18.9, 19.8, 20.6, 30.0)

par(mfrow=c(1,2))

plot(x2, y2, main = 'Influential Outlier')
abline(lm(y2~x2))

plot(x3, y3, main = 'Outlier is not Influential')
abline(lm(y3~x3))
```

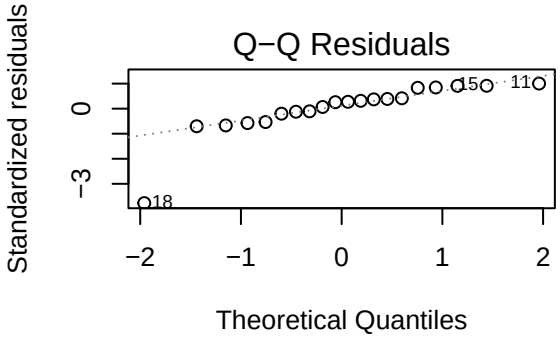


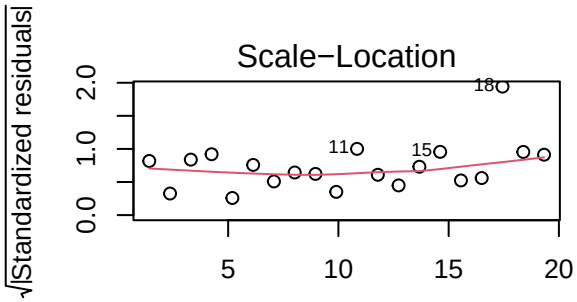
Why does the outlier in the first graphic is considered to be influential while the outlier in the second graphic is not considered not influential. Simple, the outlier in the first graphic can change the result of the regression if we remove it while the oulier in the second graphic will not affect the regression line much since it still fall near the regression line.

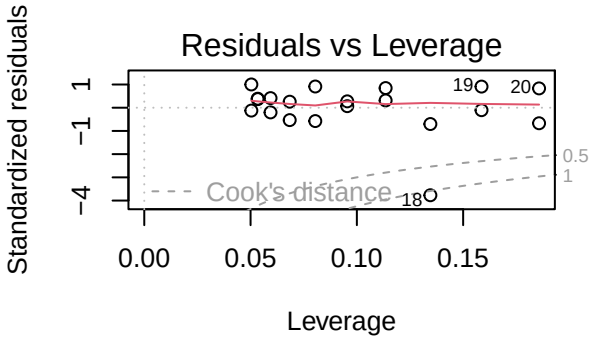
Another way we can determine if an outlier is influential or not is by looking at the cook's distance.

```
x4 <- c(1:20)
y4 <- c(0.4, 2.2, 2.2, 5.6, 5.3, 5.2, 7.5, 8.7, 9.6, 9.7, 12.5, 12.4, 12.4, 12.8,
        16.1, 16.0, 17.0, 11.5, 19.8, 20.6)
par(mfrow=c(2,2))
plot(lm(y4~x4))
```









Interpretation :

- Suspicious : cook's distance > 0.5
- Likely influential : cook's distance > 1
- Influential : cook's distance stands out from other point

Hypothesis testing

H_{0} : x1 = x2 = x3 = … = xn = 0 (There is no relationship between the predictors and the response)

if p-value < 0.05 which indicates that there is enough evidence to reject H null and there is enough evidence that there is relationship between predictors and response variable.

R-squared: model explain x% of variance in the data.

Case Study

ETL

```
student_mat = read.csv('./Data/student-mat.csv', sep =';')
student_por = read.csv('./Data/student-por.csv', sep =';')
```

EDA

```
str(student_por)
```

```
## 'data.frame':    649 obs. of  33 variables:
##  $ school    : chr  "GP" "GP" "GP" "GP" ...
```

```
##  $ sex       : chr  "F" "F" "F" "F" ...
##  $ age       : int  18 17 15 15 16 16 16 17 15 15 ...
##  $ address   : chr  "U" "U" "U" "U" ...
##  $ famsize   : chr  "GT3" "GT3" "LE3" "GT3" ...
##  $ Pstatus   : chr  "A" "T" "T" "T" ...
##  $ Medu      : int  4 1 1 4 3 4 2 4 3 3 ...
##  $ Fedu      : int  4 1 1 2 3 3 2 4 2 4 ...
##  $ Mjob      : chr  "at_home" "at_home" "at_home" "health" ...
##  $ Fjob      : chr  "teacher" "other" "other" "services" ...
##  $ reason    : chr  "course" "course" "other" "home" ...
##  $ guardian  : chr  "mother" "father" "mother" "mother" ...
##  $ traveltime: int  2 1 1 1 1 1 1 2 1 1 ...
##  $ studytime : int  2 2 2 3 2 2 2 2 2 2 ...
##  $ failures  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ schoolsup : chr  "yes" "no" "yes" "no" ...
##  $ famsup    : chr  "no" "yes" "no" "yes" ...
##  $ paid      : chr  "no" "no" "no" "no" ...
##  $ activities: chr  "no" "no" "no" "yes" ...
##  $ nursery   : chr  "yes" "no" "yes" "yes" ...
##  $ higher    : chr  "yes" "yes" "yes" "yes" ...
##  $ internet  : chr  "no" "yes" "yes" "yes" ...
##  $ romantic  : chr  "no" "no" "no" "yes" ...
##  $ famrel    : int  4 5 4 3 4 5 4 4 4 5 ...
##  $ freetime  : int  3 3 3 2 3 4 4 1 2 5 ...
##  $ goout     : int  4 3 2 2 2 2 4 4 2 1 ...
##  $ Dalc      : int  1 1 2 1 1 1 1 1 1 1 ...
##  $ Walc      : int  1 1 3 1 2 2 1 1 1 1 ...
##  $ health    : int  3 3 3 5 5 5 3 1 1 5 ...
##  $ absences  : int  4 2 6 0 0 6 0 2 0 0 ...
##  $ G1        : int  0 9 12 14 11 12 13 10 15 12 ...
##  $ G2        : int  11 11 13 14 13 12 12 13 16 12 ...
##  $ G3        : int  11 11 12 14 13 13 13 13 17 13 ...
```

```
str(student_mat)
```

```
## 'data.frame':    395 obs. of  33 variables:
##  $ school    : chr  "GP" "GP" "GP" "GP" ...
##  $ sex       : chr  "F" "F" "F" "F" ...
##  $ age       : int  18 17 15 15 16 16 16 17 15 15 ...
##  $ address   : chr  "U" "U" "U" "U" ...
##  $ famsize   : chr  "GT3" "GT3" "LE3" "GT3" ...
##  $ Pstatus   : chr  "A" "T" "T" "T" ...
##  $ Medu      : int  4 1 1 4 3 4 2 4 3 3 ...
##  $ Fedu      : int  4 1 1 2 3 3 2 4 2 4 ...
##  $ Mjob      : chr  "at_home" "at_home" "at_home" "health" ...
##  $ Fjob      : chr  "teacher" "other" "other" "services" ...
##  $ reason    : chr  "course" "course" "other" "home" ...
##  $ guardian  : chr  "mother" "father" "mother" "mother" ...
##  $ traveltime: int  2 1 1 1 1 1 1 2 1 1 ...
##  $ studytime : int  2 2 2 3 2 2 2 2 2 2 ...
##  $ failures  : int  0 0 3 0 0 0 0 0 0 0 ...
##  $ schoolsup : chr  "yes" "no" "yes" "no" ...
##  $ famsup    : chr  "no" "yes" "no" "yes" ...
##  $ paid      : chr  "no" "no" "yes" "yes" ...
##  $ activities: chr  "no" "no" "no" "yes" ...
##  $ nursery   : chr  "yes" "no" "yes" "yes" ...
##  $ higher    : chr  "yes" "yes" "yes" "yes" ...
##  $ internet  : chr  "no" "yes" "yes" "yes" ...
##  $ romantic  : chr  "no" "no" "no" "yes" ...
##  $ famrel    : int  4 5 4 3 4 5 4 4 4 5 ...
##  $ freetime  : int  3 3 3 2 3 4 4 1 2 5 ...
##  $ goout     : int  4 3 2 2 2 2 4 4 2 1 ...
##  $ Dalc      : int  1 1 2 1 1 1 1 1 1 1 ...
```

```
##  $ Walc      : int  1 1 3 1 2 2 1 1 1 1 ...
##  $ health    : int  3 3 3 5 5 5 3 1 1 5 ...
##  $ absences  : int  6 4 10 2 4 10 0 6 0 0 ...
##  $ G1        : int  5 5 7 15 6 15 12 6 16 14 ...
##  $ G2        : int  6 5 8 14 10 15 12 5 18 15 ...
##  $ G3        : int  6 6 10 15 10 15 11 6 19 15 ...
```