```
a <- 2
```

```
a
```

```
2
```

```
install.packages("stats")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("ggfortify")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Warning message:
"package 'stats' is a base package, and should not be updated"
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

also installing the dependency 'gridExtra'
```

```
library(stats)
library(dplyr)
library(ggplot2)
library(ggfortify)
View(iris)
```

```
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
```

A data.frame: 150 × 5

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
<dbl>	<dbl>	<dbl>	<dbl>	<fct>
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1.0	0.2	setosa

```
5.1    3.3    1.7    0.5    setosa
4.8    3.4    1.9    0.2    setosa
5.0    3.0    1.6    0.2    setosa
5.0    3.4    1.6    0.4    setosa
5.2    3.5    1.5    0.2    setosa
5.2    3.4    1.4    0.2    setosa
4.7    3.2    1.6    0.2    setosa
 ⋮      ⋮      ⋮      ⋮       ⋮
6.9    3.2    5.7    2.3    virginica
5.6    2.8    4.9    2.0    virginica
7.7    2.8    6.7    2.0    virginica
6.3    2.7    4.9    1.8    virginica
6.7    3.3    5.7    2.1    virginica
7.2    3.2    6.0    1.8    virginica
6.2    2.8    4.8    1.8    virginica
6.1    3.0    4.9    1.8    virginica
6.4    2.8    5.6    2.1    virginica
7.2    3.0    5.8    1.6    virginica
7.4    2.8    6.1    1.9    virginica
7.9    3.8    6.4    2.0    virginica
6.4    2.8    5.6    2.2    virginica
6.3    2.8    5.1    1.5    virginica
6.1    2.6    5.6    1.4    virginica
7.7    3.0    6.1    2.3    virginica
6.3    3.4    5.6    2.4    virginica
6.4    3.1    5.5    1.8    virginica
6.0    3.0    4.8    1.8    virginica
6.9    3.1    5.4    2.1    virginica
```

```
mydata = select(iris,c(1,2,3,4))
```

```
6.9    3.1    5.1    2.3    virginica
```

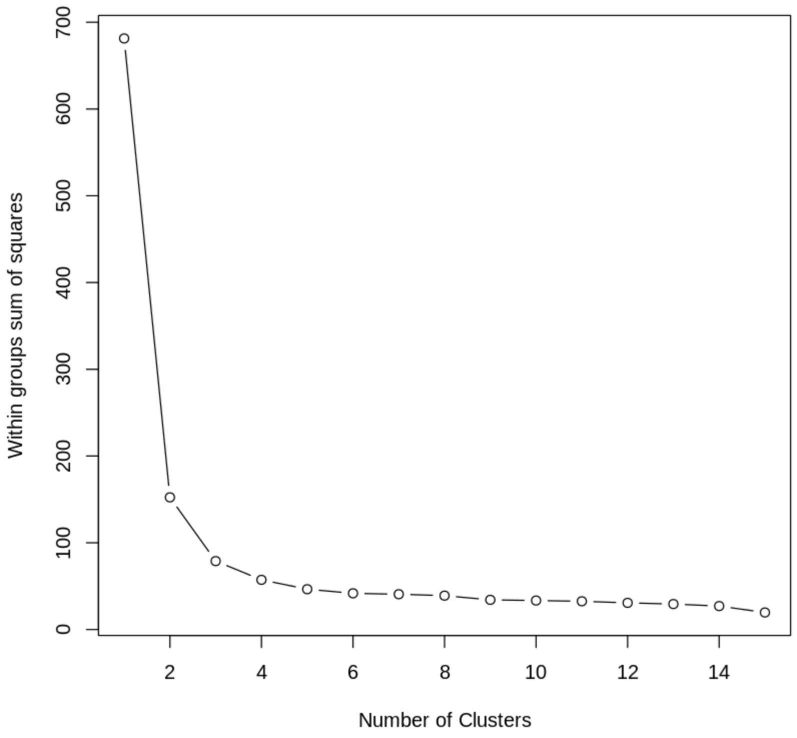
```
mydata
```

```
              A data.frame: 150 × 4
Sepal.Length Sepal.Width Petal.Length Petal.Width
       <dbl>       <dbl>        <dbl>       <dbl>
         5.1         3.5          1.4         0.2
         4.9         3.0          1.4         0.2
         4.7         3.2          1.3         0.2
         4.6         3.1          1.5         0.2
         5.0         3.6          1.4         0.2
         5.4         3.9          1.7         0.4
         4.6         3.4          1.4         0.3
         5.0         3.4          1.5         0.2
         4.4         2.9          1.4         0.2
         4.9         3.1          1.5         0.1
         5.4         3.7          1.5         0.2
         4.8         3.4          1.6         0.2
         4.8         3.0          1.4         0.1
         4.3         3.0          1.1         0.1
         5.8         4.0          1.2         0.2
         5.7         4.4          1.5         0.4
         5.4         3.9          1.3         0.4
         5.1         3.5          1.4         0.3
         5.7         3.8          1.7         0.3
         5.1         3.8          1.5         0.3
         5.4         3.4          1.7         0.2
         5.1         3.7          1.5         0.4
         4.6         3.6          1.0         0.2
         5.1         3.3          1.7         0.5
         4.8         3.4          1.9         0.2
         5.0         3.0          1.6         0.2
         5.0         3.4          1.6         0.4
         5.2         3.5          1.5         0.2
         5.2         3.4          1.4         0.2
         4.7         3.2          1.6         0.2
           ⋮           ⋮            ⋮           ⋮
```

6.9	3.2	5.7	2.3
5.6	2.8	4.9	2.0
7.7	2.8	6.7	2.0
6.3	2.7	4.9	1.8
6.7	3.3	5.7	2.1
7.2	3.2	6.0	1.8
6.2	2.8	4.8	1.8
6.1	3.0	4.9	1.8
6.4	2.8	5.6	2.1
7.2	3.0	5.8	1.6
7.4	2.8	6.1	1.9
7.9	3.8	6.4	2.0
6.4	2.8	5.6	2.2
6.3	2.8	5.1	1.5
6.1	2.6	5.6	1.4
7.7	3.0	6.1	2.3
6.3	3.4	5.6	2.4
6.4	3.1	5.5	1.8
6.0	3.0	4.8	1.8
6.9	3.1	5.4	2.1
6.7	3.1	5.6	2.4
6.9	3.1	5.1	2.3
5.8	2.7	5.1	1.9
6.8	3.2	5.9	2.3
6.7	3.3	5.7	2.5
6.7	3.0	5.2	2.3
6.3	2.5	5.0	1.9
6.5	3.0	5.2	2.0

```
wssplot <- function(data, nc=15, seed=1234){
 wss <- (nrow(data)-1)*sum(apply(data,2,var))
 for (i in 2:nc){
   set.seed(seed)
   wss[i] <- sum(kmeans(data, centers=i)$withinss)}
 plot(1:nc, wss, type="b", xlab="Number of Clusters",
      ylab="Within groups sum of squares")
}
```

```
wssplot(mydata)
```

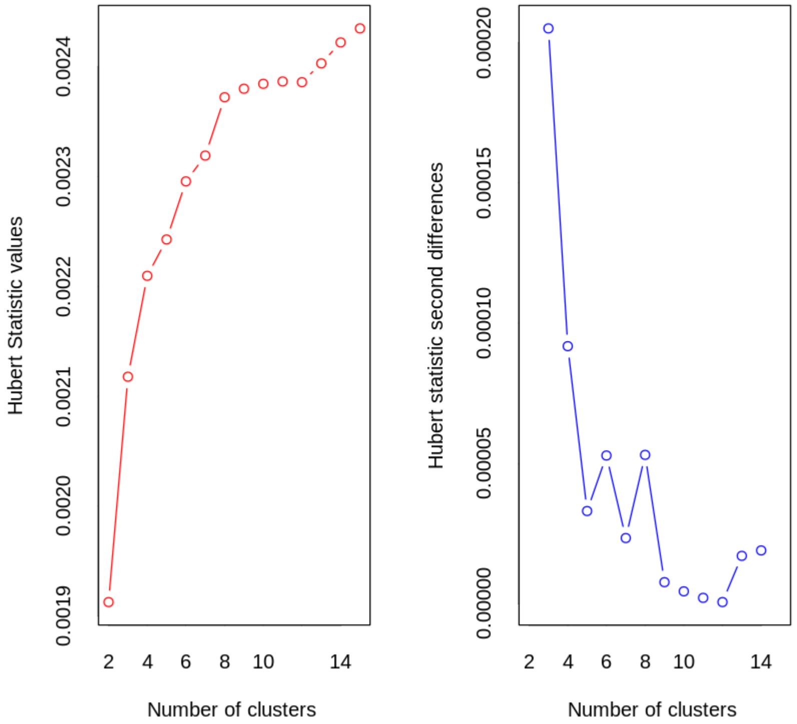


```
install.packages("NbClust")
library(NbClust)
```

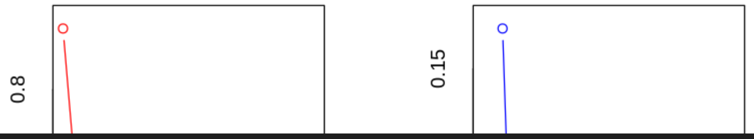
```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
set.seed(1234)
nc <- NbClust(mydata,min.nc=2,max.nc=15,method="kmeans")
```

```
*** : The Hubert index is a graphical method of determining the number of clus
                In the plot of Hubert index, we seek a significant knee that c
                significant increase of the value of the measure i.e the signi
                index second differences plot.
```

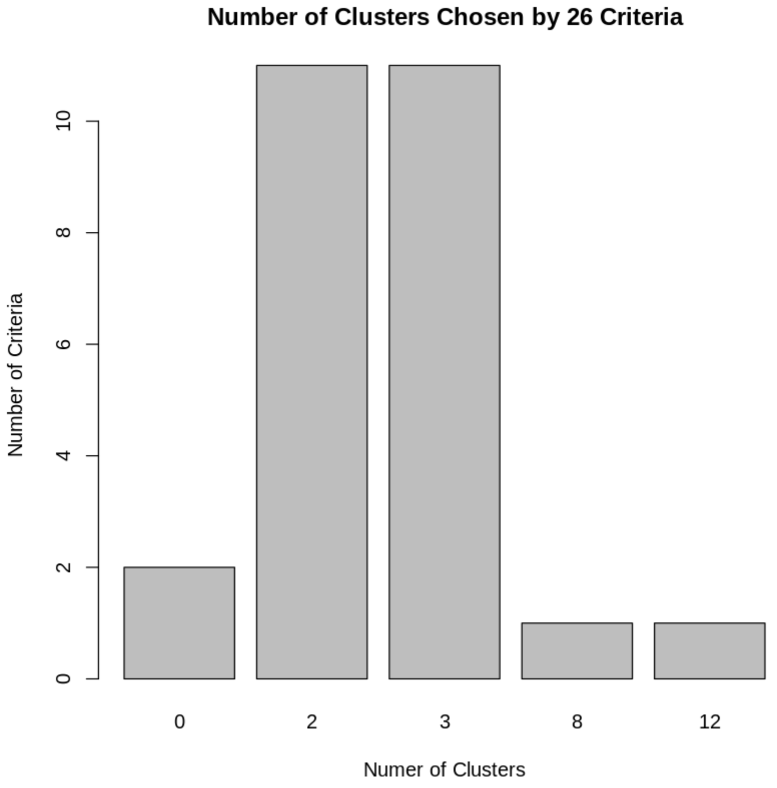


```
*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the signif
                second differences plot) that corresponds to a significant inc
                the measure.

*******************************************************************
* Among all indices:
* 11 proposed 2 as the best number of clusters
* 11 proposed 3 as the best number of clusters
* 1 proposed 8 as the best number of clusters
* 1 proposed 12 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2


*******************************************************************
```



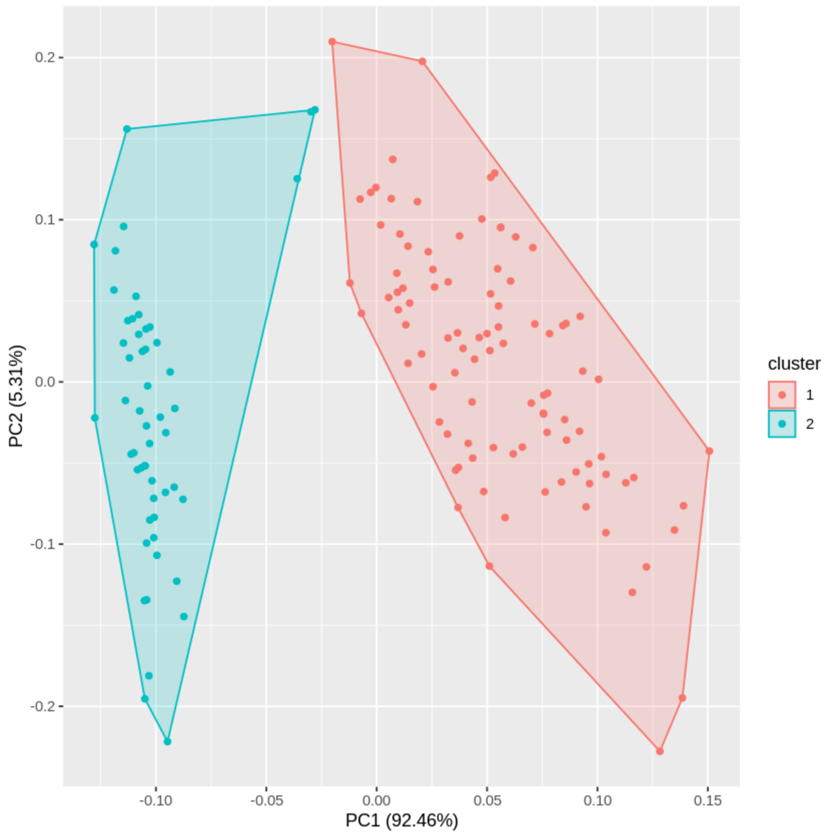
```
barplot(table(nc$Best.n[1,]),
              xlab="Numer of Clusters", ylab="Number of Criteria",
                    main="Number of Clusters Chosen by 26 Criteria")
table(nc$Best.n[1,])
```

```
 0  2  3  8 12
 2 11 11  1  1
```



```
KM = kmeans(mydata,2)
```

```
autoplot(KM,mydata,frame=TRUE)
```



```
KM$centers
```

A matrix: 2 × 4 of type dbl

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	6.301031	2.886598	4.958763	1.695876
2	5.005660	3.369811	1.560377	0.290566

```
install.packages("caTools")
install.packages("randomForest")
library(caTools)
library(randomForest)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependency ‘bitops’


Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

randomForest 4.7-1.1

Type rfNews() to see new features/changes/bug fixes.


Attaching package: ‘randomForest’


The following object is masked from ‘package:ggplot2’:

    margin


The following object is masked from ‘package:dplyr’:

    combine
```

```
split <- sample.split(iris,SplitRatio=0.7)
```

```
train <- subset(iris,split == "TRUE")
```

```
test <- subset(iris,split == "FALSE")
```

```
set.seed(120)
rfc = randomForest(x=train[-5],y=train$Species,ntree=500)
```

```
rfc
```

```

Call:
 randomForest(x = train[-5], y = train$Species, ntree = 500) 
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 3.33%
Confusion matrix:
           setosa versicolor virginica class.error
setosa         30          0         0  0.00000000
versicolor      0         28         2  0.06666667
virginica       0          1        29  0.03333333
```

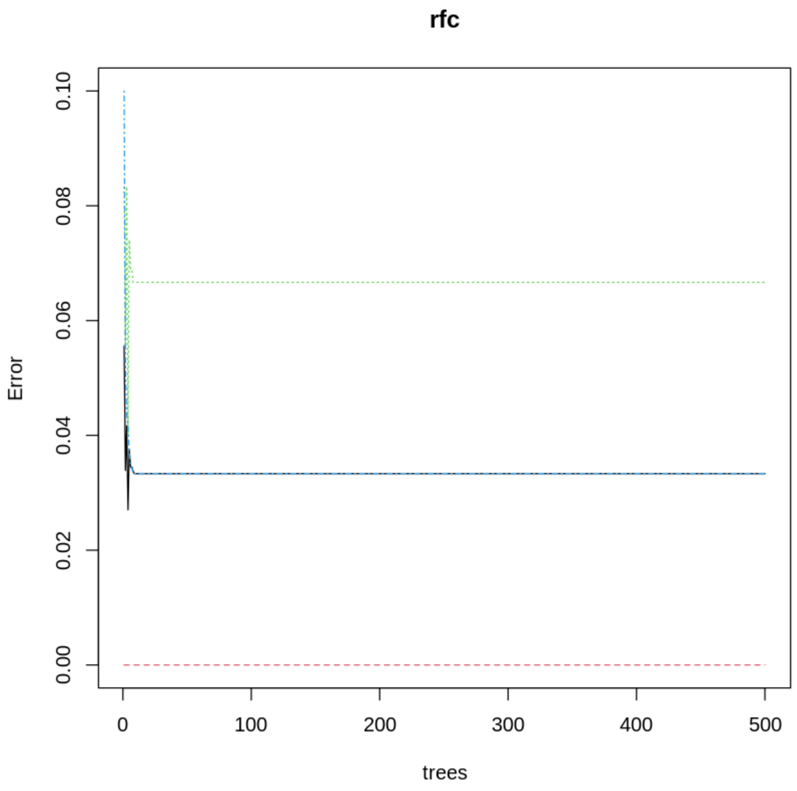
```
ypred=predict(rfc,newdata=test[-5])
```

```
cm = table(test[, 5],ypred)
```

```
cm
```

```
            ypred
             setosa versicolor virginica
  setosa         20          0         0
  versicolor      0         20         0
  virginica       0          4        16
```

```
plot(rfc)
```



```
importance(rfc)
```

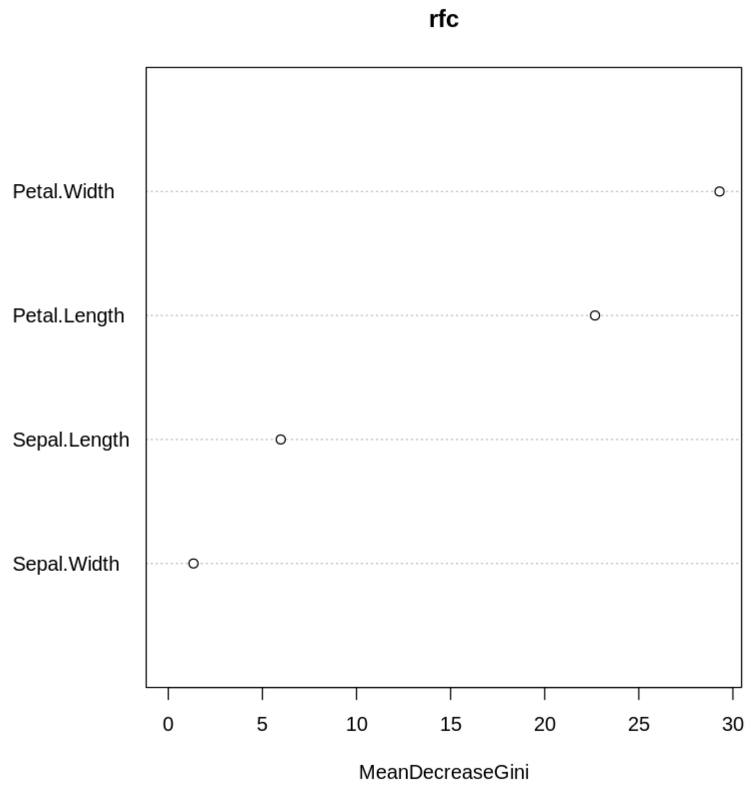
A matrix: 4 × 1 of type dbl

	MeanDecreaseGini
Sepal.Length	5.975146
Sepal.Width	1.340794
Petal.Length	22.673748
Petal.Width	29.289378

```
cm
```

```
            ypred
             setosa versicolor virginica
  setosa         20          0         0
  versicolor      0         20         0
  virginica       0          4        16
```

```
varImpPlot(rfc)
```



```
summary(rfc)
```

```
                Length Class  Mode
call               4   -none- call
type               1   -none- character
predicted         90   factor numeric
err.rate        2000   -none- numeric
confusion         12   -none- numeric
votes            270   matrix numeric
oob.times         90   -none- numeric
classes            3   -none- character
importance         4   -none- numeric
importanceSD       0   -none- NULL
localImportance    0   -none- NULL
```

```
install.packages("datarium")

data("marketing",package="datarium")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
marketing
```

```
 84.72   19.20   48.96   12.60
     ⋮       ⋮       ⋮       ⋮
 60.00   13.92   22.08   10.08
197.40   25.08   56.88   17.40
 23.52   24.12   20.40    9.12
202.08    8.52   15.36   14.04
266.88    4.08   15.72   13.80
332.28   58.68   50.16   32.40
298.08   36.24   24.36   24.24
204.24    9.36   42.24   14.04
332.04    2.76   28.44   14.16
198.72   12.00   21.12   15.12
187.92    3.12    9.96   12.60
262.20    6.48   32.88   14.64
 67.44    6.84   35.64   10.44
345.12   51.60   86.16   31.44
304.56   25.56   36.00   21.12
246.00   54.12   23.52   27.12
167.40    2.52   31.92   12.36
229.32   34.44   21.84   20.76
343.20   16.68    4.44   19.08
 22.44   14.52   28.08    8.04
 47.40   49.32    6.96   12.96
 90.60   12.96    7.20   11.88
 20.64    4.92   37.92    7.08
200.16   50.40    4.32   23.52
179.64   42.72    7.20   20.76
 45.84    4.44   16.56    9.12
113.04    5.88    9.72   11.64
212.40   11.16    7.68   15.36
340.32   50.40   79.44   30.60
278.52   10.32   10.44   16.08
```

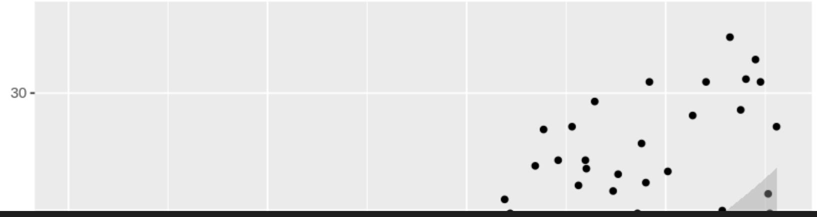
```
head(marketing,4)
```

A data.frame: 4 × 4

	youtube	facebook	newspaper	sales
	<dbl>	<dbl>	<dbl>	<dbl>
1	276.12	45.36	83.04	26.52
2	53.40	47.16	54.12	12.48
3	20.64	55.08	83.16	11.16
4	181.80	49.56	70.20	22.20

```
ggplot(marketing,aes(x=youtube,y=sales)) + geom_point() + stat_smooth()
# geom_point() does the scattering
# stat_smooth makes the blue line
# aes = aesthetic
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

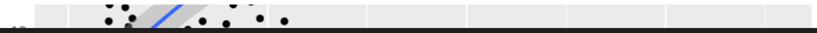


```
cor(marketing$sales,marketing$youtube)
```

```
0.782224424861606
```

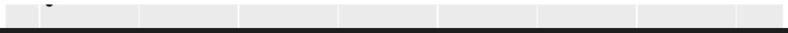
```
cor(marketing$sales,marketing$facebook)
```

```
0.576222574571055
```



```
cor(marketing$sales,marketing$newspaper)
```

```
0.228299026376165
```



```
cor(marketing$sales,marketing$sales)
```

```
1
```

```
sales <- marketing$sales
```

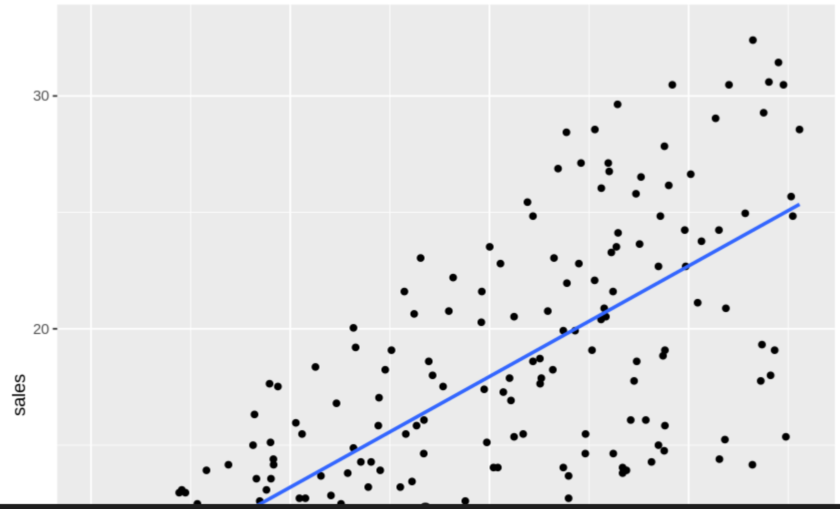
```
yt <- marketing$youtube
```

```
model <- lm(sales~yt,data=marketing)
```

```
model
```

```
Call:
lm(formula = sales ~ yt, data = marketing)

Coefficients:
(Intercept)           yt
    8.43911      0.04754
```

```
ggplot(marketing,aes(x=youtube,y=sales)) + geom_point() + stat_smooth(method=lm,se=
`geom_smooth()` using formula = 'y ~ x'
```



```
summary(model)
```

```
Call:
lm(formula = sales ~ yt, data = marketing)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.0632  -2.3454  -0.2295   2.4805   8.6548 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 8.439112   0.549412   15.36   <2e-16 ***
yt          0.047537   0.002691   17.67   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.91 on 198 degrees of freedom
Multiple R-squared:  0.6119,	Adjusted R-squared:  0.6099 
F-statistic: 312.1 on 1 and 198 DF,  p-value: < 2.2e-16
```

```
# Installing the package
install.packages("caTools")    # For Logistic regression
install.packages("ROCR")       # For ROC curve to evaluate model

# Loading package
library(caTools)
library(ROCR)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘gtools’, ‘gplots’
```

```
split <- sample.split(mtcars, SplitRatio = 0.8)
```

```
split

train_reg <- subset(mtcars, split == "TRUE")
test_reg <- subset(mtcars, split == "FALSE")
```

TRUE · TRUE · TRUE · TRUE · FALSE · TRUE · TRUE · FALSE · TRUE · FALSE · TRUE

```
# Training model
logistic_model <- glm(vs ~ wt + disp,
                      data = train_reg,
                      family = "binomial")
logistic_model
```

```
Call:  glm(formula = vs ~ wt + disp, family = "binomial", data = train_reg)

Coefficients:
(Intercept)           wt         disp
    3.87861      0.61985     -0.02875

Degrees of Freedom: 22 Total (i.e. Null);  20 Residual
Null Deviance:      30.79
Residual Deviance: 13.7         AIC: 19.7
```

```
# Summary
summary(logistic_model)
```

```
Call:
glm(formula = vs ~ wt + disp, family = "binomial", data = train_reg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6482  -0.3704  -0.1030   0.3991   1.8648

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.87861    3.69659   1.049   0.2941
wt           0.61985    1.91238   0.324   0.7458
disp        -0.02875    0.01683  -1.708   0.0877 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 30.789  on 22  degrees of freedom
Residual deviance: 13.698  on 20  degrees of freedom
AIC: 19.698

Number of Fisher Scoring iterations: 6
```

```
# Predict test data based on model
predict_reg <- predict(logistic_model,
                       test_reg, type = "response")
```