* print (intersect(x, y))
(y will print common element from both vectors.

* aggregate functions

```
print(aggregate(chickwts$feed, by = list(chickwts$feed), FUN = length))
print(aggregate(chickwts$feed ~ feed, chickwts, length)) # Equivalent
```

Group.1	x
casein	12
horsebean	10
linseed	12
meatmeal	11

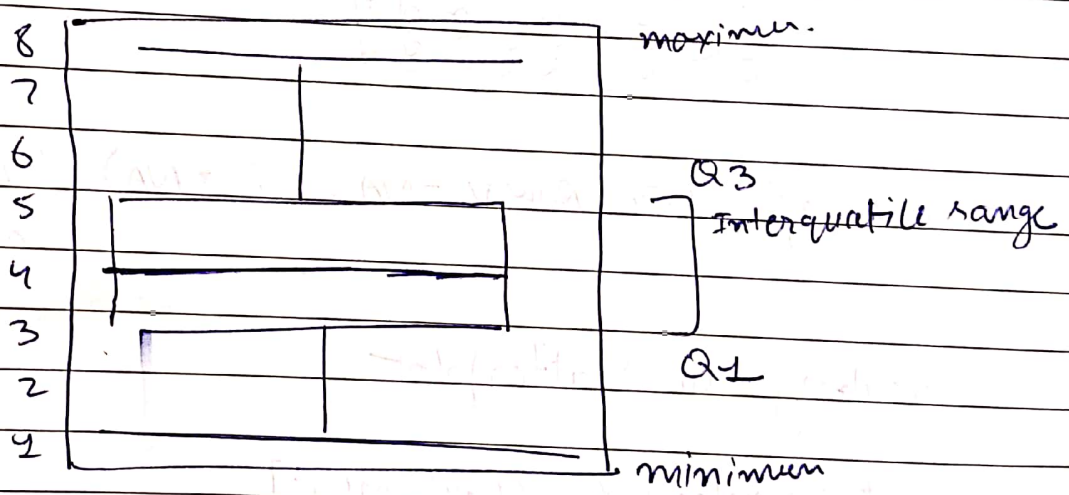
Box plots

Also known as whiskers diagrams, these display the distribution of data that is based on the five number summary

→ minimum → Third quartile
→ first quartile → Maximum
→ median

syntax

```
Vec ← c (3,2,5,6,4,8,1,2,3,2,4)
summary (Vec)
boxplot (Vec, var width = TRUE)
```



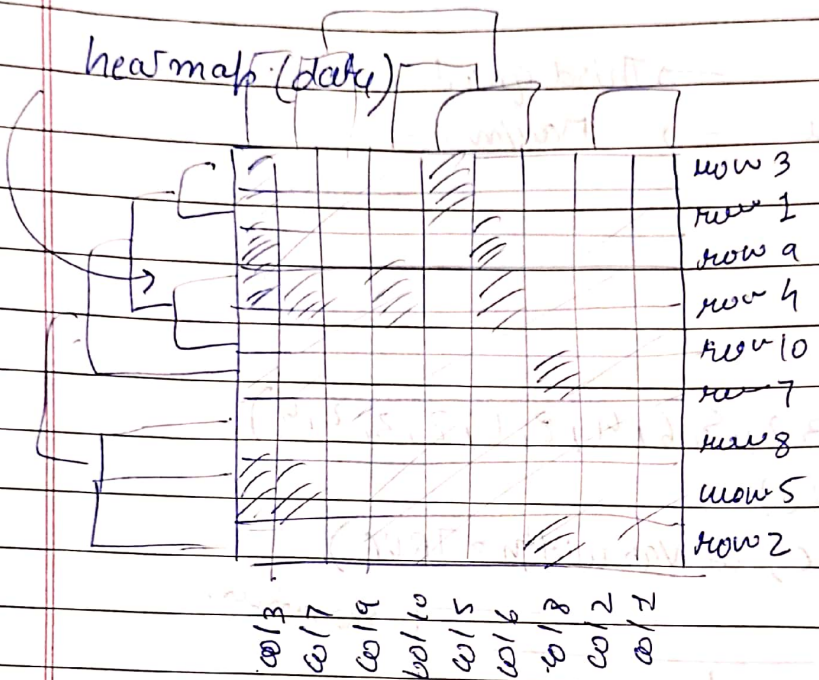
Heatmap → 2D visual representation of Data where values are encoded in colors.

→ dendogram - used visualize the result of a of a hierarchical clustering calculation.

The result of a clustering is presented either as the distance or similarity b/w clustered row & colum.

Heatmaps Example

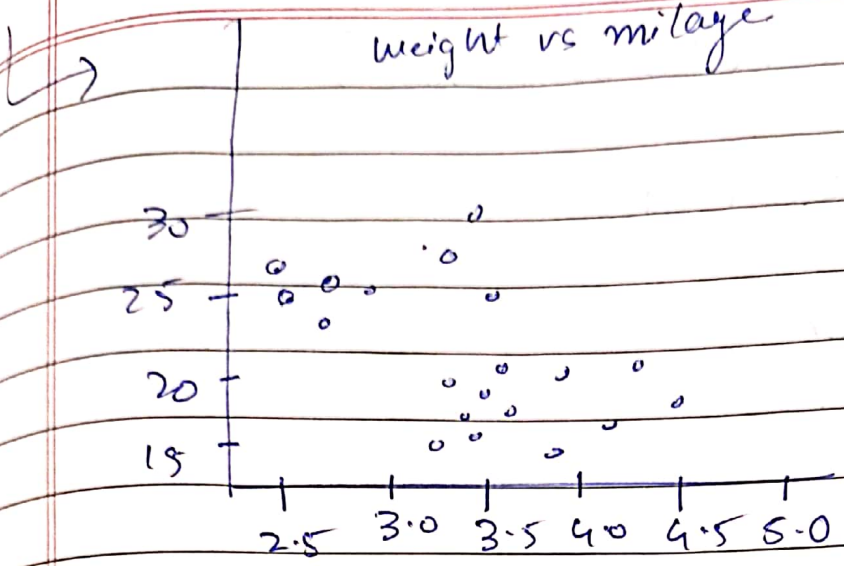
```
data ← matrix (norm (100, 0, 10), nrow = 16, ncol = 10)
colnames (data) ← paste0 ("col", 1:10)
rownames (data) ← paste0 ("row", 1:10)

heatmap (data)
```




```
heatmap (data, RowV = NA, ColV = NA) # will remove
                                       & dendogram
```

Creating the Scatterplot

```
input ← mtcars [, c ("wt", "mpg")]
# give the chart file a name.
png (file = "scatterplot.png")
plot (x = input $ wt, y = input $ mpg,
      xlab = "weight"
      ylab = "Milege"
      xlim = c (2.5, 5),
      ylim = c (15, 30),
      main = "weight vs milege"
)
dev.off() # save the file.
```



PART-2

Advanced R Notes (Short)

(1) Using which function in R

→ # creating a vector
vec ← c(5,4,3,2,1)
# Position of 5
which(vec == 5)

# position of values greater than 2
which(df > 2)

which() function with dataframes

df ← BOD
⇒ which(df$demand == '10-3'). or
which(df$demand == c(8-3, 16-0))
Find columns in a data frame with numeric values using which().

df ← datasets :: iris
df
test ← which(sapply(df, is.numeric))
colnames(df)[test]

→ "Sepal.length". "Sepal.width". "Petal.length". "petal.width"

which fundion with matrix
df ← matrix(rep(c(1,0,1),4), nrow = 4)

which(df == 0, arr.ind = T)

which:

```
df ← matrix (rep (c(1,0,1), 4), nrow = 4)
print(df)
print (which (df == 0))
print (which (df == 0, arr.ind = T))
```

→

```
     [,1] [,2] [,3]
[1,]    1    0    1
[2,]    0    1    1
[3,]    1    1    0
[4,]    1    0    1
```

→ [1] 2 5 8 11

```
     row col
[1,]   2   1
[2,]   1   2
[3,]   4   2
[4,]   3   3
```

② cut function in R

* rnorm() will generate random distribution of number
  syntax (: rnorm(n, mean, sd))
* The cut() is a built-in R function that divides the ranges of x into intervals and codes the values in x according to which interval they fall. To convert numeric to factor in R use the cut().

```
data ← stats:: rnorm(20)
c ← cut(data, breaks = -3:3)
```

→

```
[1] (0,1] (-1,0] (-2,-1] (0,1] (0,1] (1,2] (-1,0]
[10] (0,1] (-1,0] (0,1] (0,1] (-2,-1] (-1,0] (0,1] (-1,0]
[19] (-1,0] (-1,0]
Levels: (-3,-2] (-2,-1] (-1,0] (0,1] (1,2] (2,3]
```

```
data ← stats::rnorm(20)
c ← cut(data, breaks = -3:3)
summary(c)
(-3,-2] (-2,-1] (-1,0] (0,1] (1,2] (2,3]
      0       1       9       9       1       0

c ← cut(data, breaks = 2)
[1] (-1.39,0.534] (-1.36,0.534] (-1.39,0.534] ....
[6] (-1.39,0.534] (0.534,2.46] (-1.39,0.534] ....
[11] (-1.39,0.534] (0.534,2.46] (-1.39,0.534] ....
[16] (-1.39,0.534] (0.534,2.46] (-1.39,0.534] ....

levels : (-1.39,0.534] (0.534,2.46]
```