

# Correlation Test

- A statistical measure that indicates how strongly two variables are related
- Involves the relationship between multiple variables as well
- For example, when one variable increase and the other increases as well, then these two variables are said to be **positively correlated**.
- The other way around when a variable increase and the other decrease then these two variables are **negatively correlated**.
- In the case of no correlation no pattern will be seen between the two variables.
- For instance, if one is interested to know whether there is a relationship between the heights of fathers and sons, a correlation coefficient can be calculated to answer this question. Generally, it lies between -1 and +1. It is a scaled version of covariance and provides the direction and strength of a relationship.

# Types of correlation

There are mainly two types of correlation:

1. Parametric Correlation – **Pearson** correlation( $r$ ): It measures a **linear** dependence between two variables ( $x$  and  $y$ ) is known as a parametric correlation test because it depends on the distribution of the data.
2. Non-Parametric Correlation – **Kendall** and **Spearman**: They are **rank-based** correlation coefficients, are known as non-parametric correlation.

# Parametric Correlation

$$r = \frac{\sum (x - m_x)(y - m_y)}{\sqrt{\sum (x - m_x)^2 \sum (y - m_y)^2}}$$

Where,

**r:** pearson correlation coefficient

**x and y:** two vectors of length n

**mx and my:** corresponds to the means of x and y, respectively.

## Implementation in R

Syntax: `cor(x, y, method = "pearson")`

`cor.test(x, y, method = "pearson")`

Where:

**x, y:** numeric vectors with the same length

**method:** correlation method

## **Example:1**

```
x = c(1, 2, 3, 4, 5, 6, 7)
```

```
y = c(1, 3, 6, 2, 7, 4, 5)
```

```
result = cor(x, y, method = "pearson")
```

```
cat("Pearson correlation coefficient is:", result)
```

**OR**

```
result = cor.test(x, y, method = "pearson")
```

```
print(result)
```

## **Example:2**

```
Height of Father as x = c(65,66,67,67,68,69,71,73)
```

```
Height of Son as y= c(64,65,66,66,67,68,70,71)
```

```
result = cor(x, y, method = "pearson")
```

```
cat("Pearson correlation coefficient is:", result)
```

**OR**

```
result = cor.test(x, y, method = "pearson")
```

```
print(result)
```

**To analysis:**

**Scatter Plot:**



























```
plot(x,y,xlab='Father H',ylab = "Son  
H",col='red',xlim=c(60,80),ylim=c(60,80))
```

```
abline(lm(y~x),col="blue")
```

## Plotting symbols

```
pch = 0,square  
pch = 1,circle  
pch = 2,triangle point up  
pch = 3,plus  
pch = 4,cross  
pch = 5,diamond  
pch = 6,triangle point down  
pch = 7,square cross  
pch = 8,star  
pch = 9,diamond plus  
pch = 10,circle plus  
pch = 11,triangles up and down  
pch = 12,square plus  
pch = 13,circle cross  
pch = 14,square and triangle down  
pch = 15, filled square  
pch = 16, filled circle  
pch = 17, filled triangle point-up  
pch = 18, filled diamond  
pch = 19, solid circle  
pch = 20,bullet (smaller circle)
```

pch = 21, filled circle blue  
pch = 22, filled square blue  
pch = 23, filled diamond blue  
pch = 24, filled triangle point-up blue  
pch = 25, filled triangle point down blue

<b>0</b> 	<b>1</b> 	<b>2</b> 	<b>3</b> 	<b>4</b> 	
<b>5</b> 	<b>6</b> 	<b>7</b> 	<b>8</b> 	<b>9</b> 	
<b>10</b> 	<b>11</b> 	<b>12</b> 	<b>13</b> 	<b>14</b> 	
<b>15</b> 	<b>16</b> 	<b>17</b> 	<b>18</b> 	<b>19</b> 	
<b>20</b> 	<b>21</b> 	<b>22</b> 	<b>23</b> 	<b>24</b> 	<b>25</b> 

# Working with mean, median, mode, 5-point summary

**Mean**

$$\text{Mean } (\bar{x}) = \frac{\sum x}{n}$$



- It is also defined as average which is the sum divided by count.

- The function `mean()` is used to calculate this in R.

- Example:

```
> x=c(10,20,30,40,50)
> mean(x)
[1] 30
```

## Syntax

**`mean(x, trim = 0, na.rm = FALSE, ...)`**

**x** is the input vector.

**trim** is used to drop some observations from both end of the sorted vector.

**na.rm** is used to remove the missing values from the input vector.

## Applying Trim Option

```
> x=c(12,7,3,4.2,18,2,54,-21,8,-5)
```

```
> mean(x,trim = 0.3)
```

```
[1] 5.55
```

In this case the sorted vector is (-21, -5, 2, 3, 4.2, 7, 8, 12, 18, 54) and the values removed from the vector for calculating mean are (-21,-5,2) from left and (12,18,54) from right.

## Applying NA Option

If there are missing values, then the mean function **returns NA**.

To drop the missing values from the calculation use **na.rm = TRUE**. which means **remove the NA values**.

```
> x=c(10,20,30,40,NA)
```

```
> mean(x,na.rm = TRUE)
```

```
[1] 25
```

## Median

Odd

$$\frac{n+1}{2}$$

Even

$$\frac{n}{2}, \frac{n}{2} + 1$$

- The middle most value in a data series is called the median
- The median() function is used in R to calculate this value
- If the number of elements in the data set is odd then the center element is median and if it is even then the median would be the average of two central elements.

Example :

```
> x=c(10,30,20,40,60,33)
```

```
> median(x)
```

```
[1] 31.5
```

```
> x=c(10,30,20,40,60)
> median(x)
[1] 30
```

### **Syntax: median(x, na.rm = False)**

Where, X is a vector and na.rm is used to remove missing value

Example:

```
> x=c(10,30,20,40,60,NA)
> median(x,na.rm=TRUE)
[1] 30
```

## **Mode**

The mode is the value that has highest number of occurrences in a set of data.

- Unlike mean and median, mode can have both numeric and character data.
- R does not have a standard in-built function to calculate mode.
- `table()` method for this as it creates a categorical representation of data with the variable names
- `names()` - to set or get name of objects

```
marks <- c(97, 78, 57, 78, 97, 66, 87, 64, 87, 78)
```

```
names(sort(table(marks), decreasing = TRUE))[1]
```

## **Five point summary**

A five number summary is a way to summarize a dataset using the following five values:

The minimum

The first quartile  
The median  
The third quartile  
The maximum

## 1. Five Number Summary of Vector

### Example:

```
> data <- c(4, 6, 6, 7, 8, 9, 12, 13, 14, 15, 15, 18, 22)
> fivenum(data)
[1] 4 7 12 15 22
```

From output

The minimum: 4  
The first quartile: 7  
The median: 12  
The third quartile: 15  
The maximum: 22

**2. We can quickly visualize the five number summary by creating a boxplot:**

```
boxplot(data)
```

### **3. Five Number Summary of Column in Data Frame**

```
> df <- data.frame(team=c('A', 'B', 'C', 'D', 'E', 'F', 'G', 'H'),  
                    points=c(99, 90, 86, 88, 95, 87, 85, 89),  
                    assists=c(33, 28, 31, 39, 34, 30, 29, 25),  
                    rebounds=c(30, 28, 24, 24, 28, 30, 31, 35))
```

```
> df
```

	team	points	assists	rebounds
1	A	99	33	30
2	B	90	28	28
3	C	86	31	24
4	D	88	39	24
5	E	95	34	28
6	F	87	30	30
7	G	85	29	31
8	H	89	25	35

```
> fivenum(df$points)  
[1] 85.0 86.5 88.5 92.5 99.0
```

## 4. Five Number Summary of Multiple Columns

### Example:

```
> df <- data.frame(team=c('A', 'B', 'C', 'D', 'E', 'F', 'G', 'H'),  
  points=c(99, 90, 86, 88, 95, 87, 85, 89),  
  assists=c(33, 28, 31, 39, 34, 30, 29, 25),  
  rebounds=c(30, 28, 24, 24, 28, 30, 31, 35))
```

```
> sapply(df[c('points', 'assists', 'rebounds')], fivenum)
```

	points	assists	rebounds
[1,]	85.0	25.0	24.0
[2,]	86.5	28.5	26.0
[3,]	88.5	30.5	29.0
[4,]	92.5	33.5	30.5
[5,]	99.0	39.0	35.0