# Essentials of Data Science with R Software - 2
**Sampling Theory and Linear Regression Analysis**

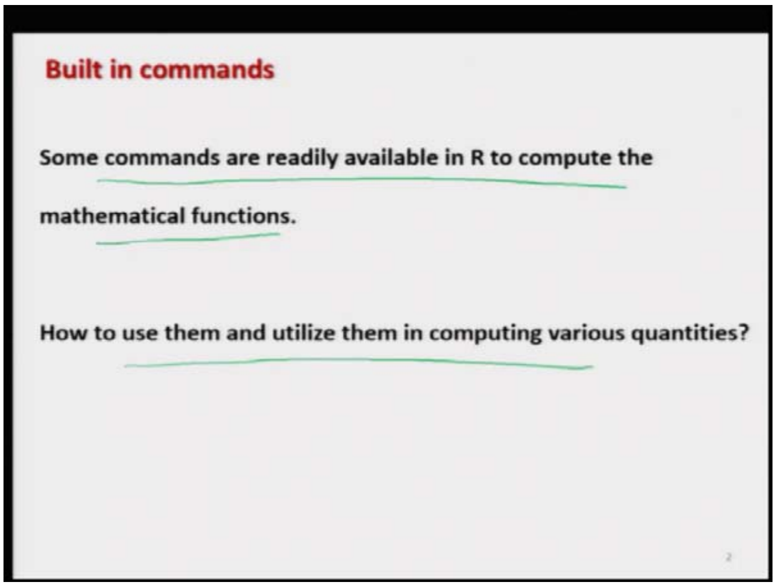
**Prof. Shalabh**
**Department of Mathematics and Statistics**
**Indian Institute of Technology, Kanpur**

## Introduction to R Software
**Lecture - 05**
**Built-in Commands and Missing Data Handling**

Hello, welcome to the course Essentials of Data Science with R Software 2, where we are handling the topics of Sampling Theory and Linear Regression Analysis and we are going to continue the module on Introduction to R Software in this lecture. In the last few lecture we have understood different types of operation under the R software. We have used R as a calculator and then we have also learnt how R will act and perform when we are trying to deal with data vectors.

Continuing on the same lines, today I am going to talk about different built in functions inside the R software. There are some special functions which have already been programmed and they can be called by certain names. For example, if you want to find out the sum, multiplication, or maxima, minima, absolute value of data or data vectors, then you need not to write a special program, but you can simply call the program call the function and execute it. So, we are going to discuss those thing today and after that I will try to briefly explain you that how we can handle the missing values in R software. So, let us try to understand all these things and we begin our lecture from here.

(Refer Slide Time: 01:56)



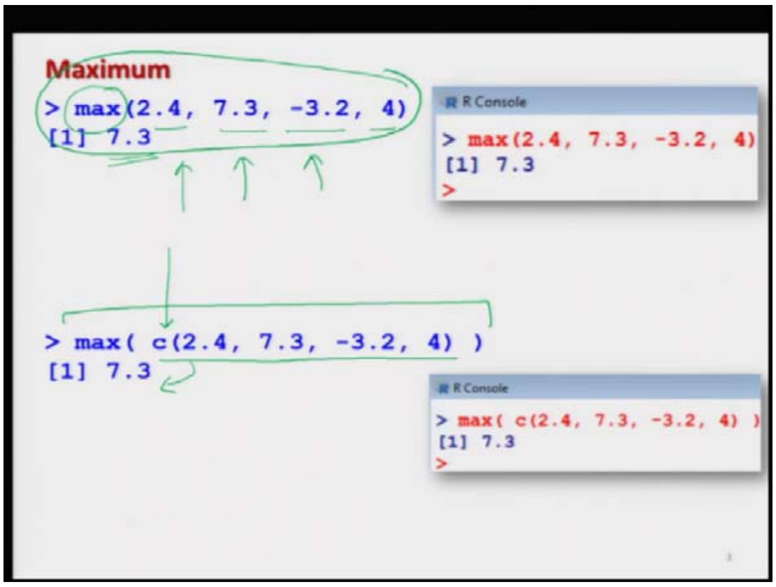
## Built in commands

Some commands are readily available in R to compute the mathematical functions.

How to use them and utilize them in computing various quantities?

So, there are some commands which are available inside the R software and they can compute different types of mathematical functions. And now we are going to understand how we are going to use those built in commands and how we can utilize them directly in computing various types of mathematical and statistical functions, right.

(Refer Slide Time: 02:20)



## Maximum

```
> max(2.4, 7.3, -3.2, 4)
[1] 7.3
```

```
> max( c(2.4, 7.3, -3.2, 4) )
[1] 7.3
```

```
> max(2.4, 7.3, -3.2, 4)
[1] 7.3
>
```

```
> max( c(2.4, 7.3, -3.2, 4) )
[1] 7.3
>
```

So, first I will try to take some examples and then I will try to show you that what is really happening. Suppose you want to find out the maximum of some values, for example let me take here an example that I have here four numbers 2.4, 7.3, minus 3.2 and 4 I and want to find out the maximum value among these 4. So, for that I can use here a function m a x max and inside the parenthesis, I can write down these values and as soon as I enter I will get here the values whichever is the maximum. For example, in this case, this will be 7.3. Here I would like to have your attention. Now, if you try to see here I am trying to use the same function, but the difference here is this here I am trying to use the command c. Whereas, if you try to see here I am not using the command c.

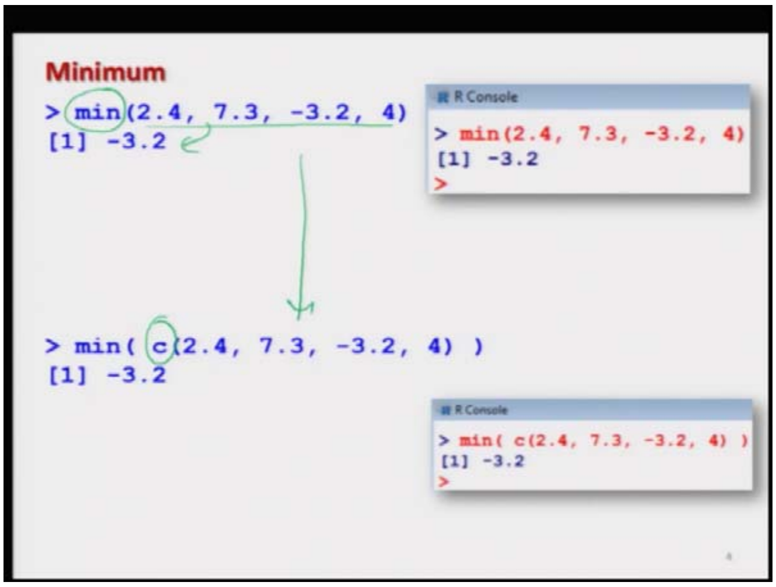
But in case if I try to give the values inside a data vector using here command c still I am getting the maximum value with the same command 7.3. And in the first case also here also I am not using c, but still I am getting this the same value, correct value, 7.3.

So, that is the point where I would like to have your attention, you see what is really happening? In R, we have understood that whenever we are trying to give an input in the form of a data vector, we have to always use c. But somehow this is happening in R with some functions that they do not require the c command. I do not know the exact reason but one possible reason is that because R has been developed by different people and these commands have been written by different people. So, possibly they used some different types of structures. But anyway, that is not my concern here, my concern is only and my objective is here only that we would like to have the correct value. So now there can be confusion that sometime you have to give the data with c commands, sometime you do not have to use the c command to give your input data.

So, my suggestion to you is that you always use c, because if you do not use c there will be a confusion; but if you use the c command there would not be any confusion and at least you will be getting always the correct outcome. So. and means. I have not encountered any situation where I can say that if somebody has used the c command, but still the function has given the wrong value.

Whereas it is possible that in case if we are not using the c command, then possibly it may give us a wrong value, right. So, let us try to continue with our remaining commands, right.

(Refer Slide Time: 06:01)


Minimum

```
> min(2.4, 7.3, -3.2, 4)
[1] -3.2
```

R Console

```
> min(2.4, 7.3, -3.2, 4)
[1] -3.2
>
```

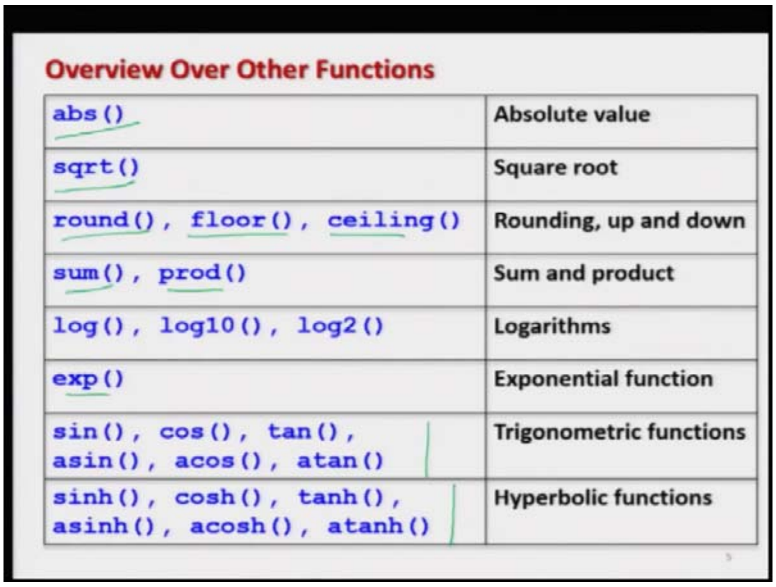
```
> min( c(2.4, 7.3, -3.2, 4) )
[1] -3.2
```

R Console

```
> min( c(2.4, 7.3, -3.2, 4) )
[1] -3.2
>
```

So, similarly if you want to find out here the minimum value, then we have a command here min. And similarly I can give here 4 values, 2.4, 7.3, minus 3.2 and 4 and if I try to execute the minimum command this will come out to be -3.2 and the same thing is happening here when I am trying to give the this data using the c command. So, m a x is used for finding out the maximum and m i n is used for finding out the minimum.

(Refer Slide Time: 06:38)

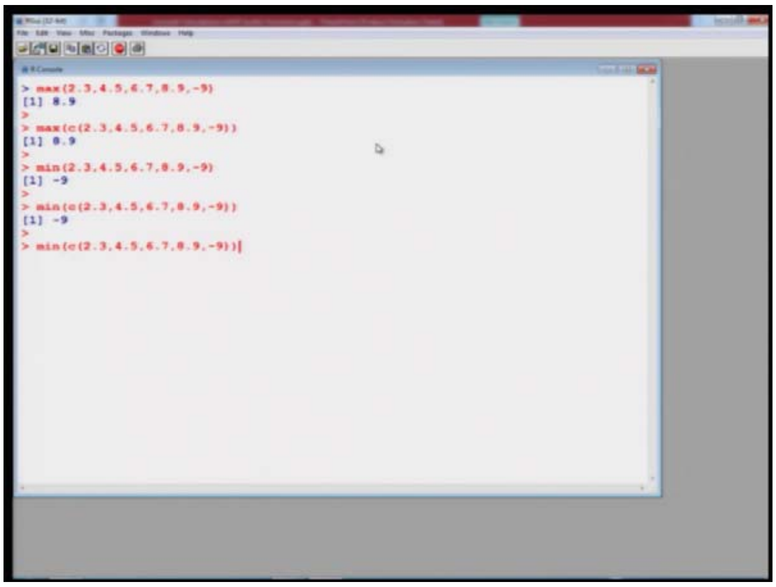

Overview Over Other Functions

| | |
|---|---|
| abs() | Absolute value |
| sqrt() | Square root |
| round(), floor(), ceiling() | Rounding, up and down |
| sum(), prod() | Sum and product |
| log(), log10(), log2() | Logarithms |
| exp() | Exponential function |
| sin(), cos(), tan(), asin(), acos(), atan() | Trigonometric functions |
| sinh(), cosh(), tanh(), asinh(), acosh(), atanh() | Hyperbolic functions |

And similarly, there is a long list of the functions which can be used for different jobs. For example, there is a function here a b s which can be used for finding out the absolute values there is a function s q r t we which can be used to find out the square root of the values and there is a function here round, floor ceiling, sum, product, log, exponentials and trigonometric function, hyperbolic trigonometric functions and so on.

So, there is a long list and it is not really possible for me to cover all the command. But I have given you some representative commands and I believe that whenever you will require them, you can just take the help from the help function and then you can use it. But before going further let me try to show you how you are going to execute it on the R console.

(Refer Slide Time: 07:37)



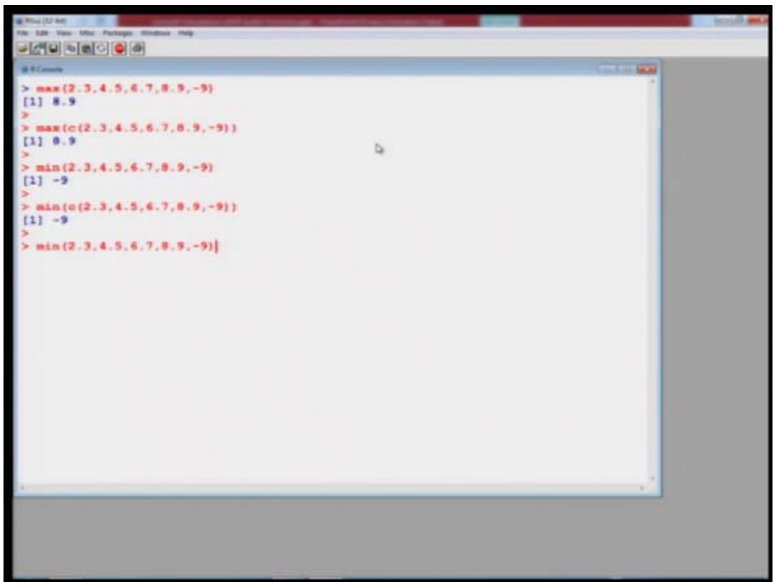
```
> max(2.3,4.5,6.7,8.9,-9)
[1] 8.9
>
> max(c(2.3,4.5,6.7,8.9,-9))
[1] 8.9
>
> min(2.3,4.5,6.7,8.9,-9)
[1] -9
>
> min(c(2.3,4.5,6.7,8.9,-9))
[1] -9
>
> min(c(2.3,4.5,6.7,8.9,-9))
```

So for example, if I say here I can show you here, suppose I want to find out the maximum of say four values, 2.3, 4.5, 6.7, 8.9 and say minus 9, right. So, here you see I am not using here the command here c and you are getting here a value here 8.9 and now the same thing I try to do here by using the command here c. So, I try to put all these data inside this vector using c command and you can see here that it is giving us the same value 8.9, ok.

Now, similarly if I try to operate the minimum function on the same data set. So, you can see here that I am not using here the command c to combine the data, but still I am getting the minimum value which is -9 in this case.

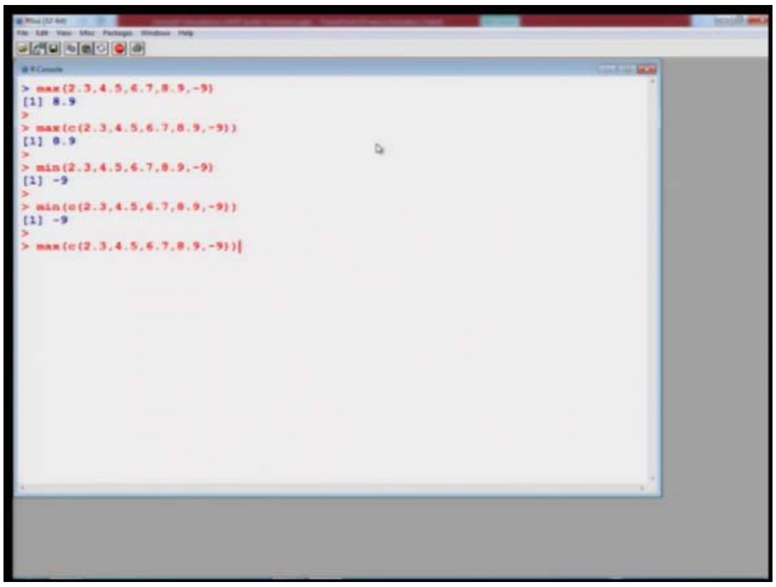
And also if I try to use here the c command then I am getting here the minimum value of all this data vector, right, this is -9. Well one thing I would like to clarify here that when you are using R, using the arrow keys you can repeat the command, you can go back into the previous command.

(Refer Slide Time: 09:14)



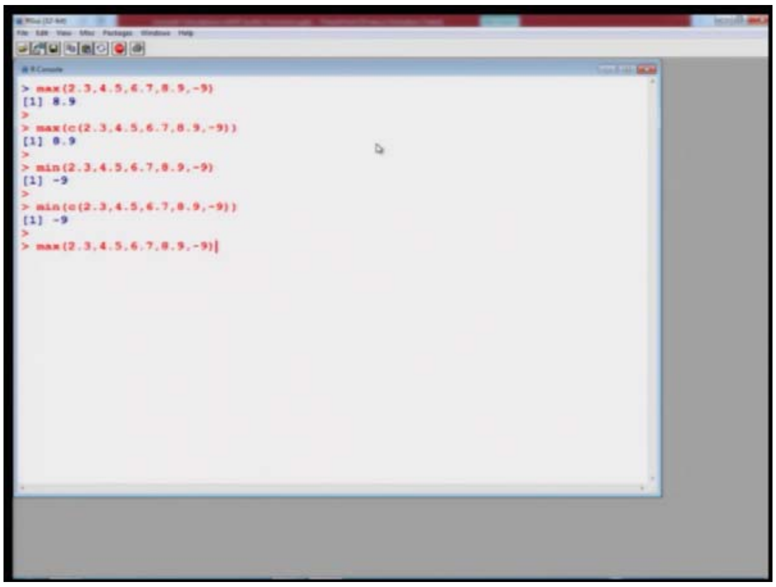
```
> max(2.3,4.5,6.7,8.9,-9)
[1] 8.9
>
> max(c(2.3,4.5,6.7,8.9,-9))
[1] 8.9
>
> min(2.3,4.5,6.7,8.9,-9)
[1] -9
>
> min(c(2.3,4.5,6.7,8.9,-9))
[1] -9
>
> min(2.3,4.5,6.7,8.9,-9)
```

(Refer Slide Time: 09:15)



```
> max(2.3,4.5,6.7,8.9,-9)
[1] 8.9
>
> max(c(2.3,4.5,6.7,8.9,-9))
[1] 8.9
>
> min(2.3,4.5,6.7,8.9,-9)
[1] -9
>
> min(c(2.3,4.5,6.7,8.9,-9))
[1] -9
>
> max(c(2.3,4.5,6.7,8.9,-9))
```

(Refer Slide Time: 09:15)

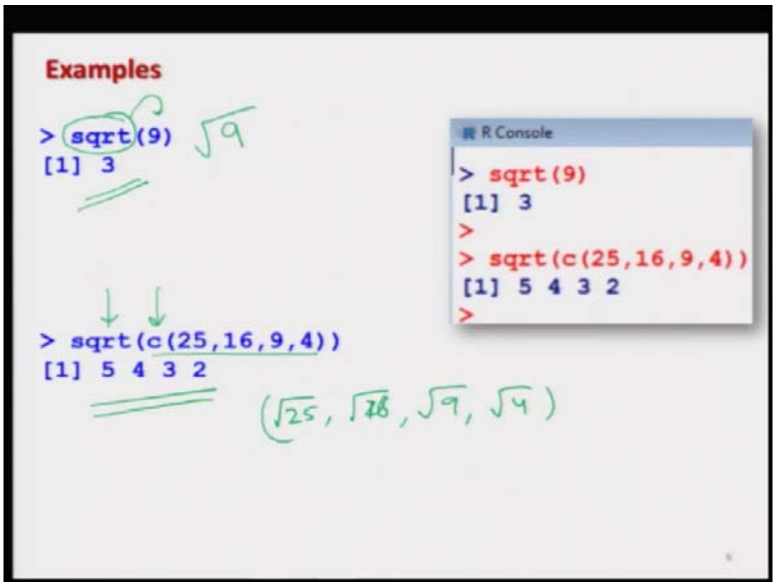


```
> max(2.3,4.5,6.7,8.9,-9)
[1] 8.9
>
> max(c(2.3,4.5,6.7,8.9,-9))
[1] 8.9
>
> min(2.3,4.5,6.7,8.9,-9)
[1] -9
>
> min(c(2.3,4.5,6.7,8.9,-9))
[1] -9
>
> max(2.3,4.5,6.7,8.9,-9)
```

For example, if I use the upward arrow key like this 1 2 3 4 and so on. You can see here whatever command I have used earlier they will reappear. So, that is what I am using here that instead of typing the entire command again and again, I am trying to use the arrow keys to repeat the command. So, do not get surprised, but you just practice this, you will also learn it.

And similarly if you try to obtain here means all other values also, that we have shown you those things can be done. And first let me go to the slides and then I will come back to R console to show you the applications, right.

(Refer Slide Time: 09:55)



## Examples

```
> sqrt(9)
[1] 3

> sqrt(c(25,16,9,4))
[1] 5 4 3 2
```

$\sqrt{9}$

$(\sqrt{25}, \sqrt{16}, \sqrt{9}, \sqrt{4})$

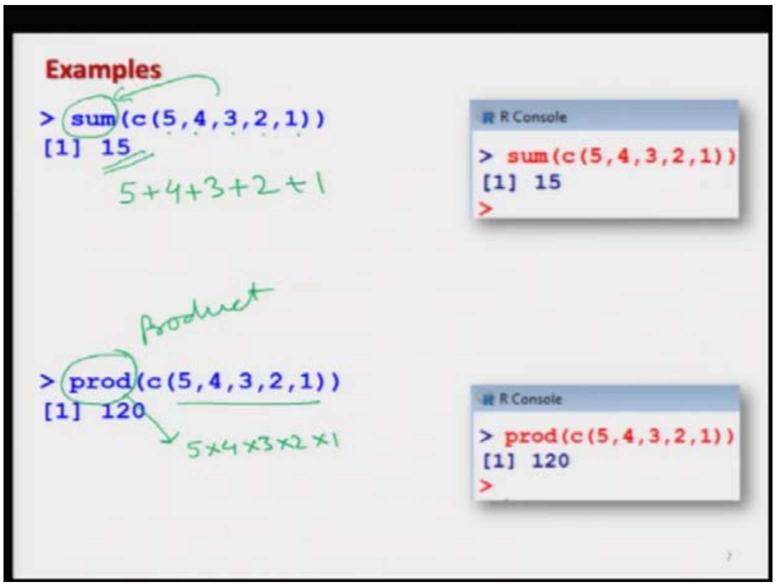
R Console

```
> sqrt(9)
[1] 3
>
> sqrt(c(25,16,9,4))
[1] 5 4 3 2
>
```

So, similarly if you want to find out the square root of certain number, then I can use here a command s q r t and if you want to find out the square root of given number just one number. For example, here I am trying to find out the square root of here 9 which is square root of here 9. The answer will come out to be here 3 and the same function is square root that is s q r t can also be used on a data vector for example, I am trying to take here a data vector of four values 25, 16, 9 and 4 and I try to operate here the command s q r t.

So, essentially I will be getting it is something like square root of 25, square root of say 16, square root of 9 and a square root of here 4 and the answer comes out to be here 5 4 3 2. So now, you can see here as a rule I will always use here the c command. I do not want to create any confusion that way that whatever is my answer by using c or without using c, right.

(Refer Slide Time: 11:07)



Examples

```
> sum(c(5,4,3,2,1))
[1] 15
```

```
> prod(c(5,4,3,2,1))
[1] 120
```

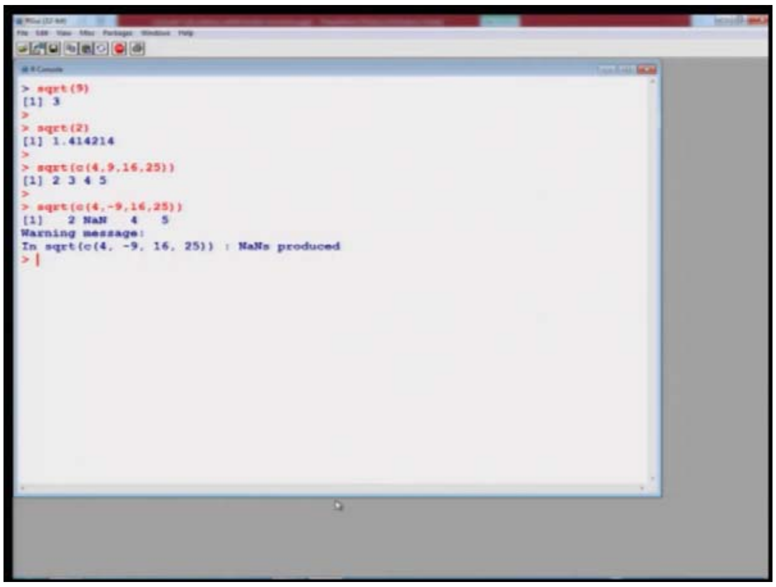
```
> sum(c(5,4,3,2,1))
[1] 15
>
```

```
> prod(c(5,4,3,2,1))
[1] 120
>
```

Similarly, there is another function here say sum, sum as it indicates by it is name, this is a function which will provide you the sum of all the values inside the data vector. So, if I have taken here five values 5 4 3 2 1, then if and then I am operating the function sum over it then this is going to give us the value 5 plus 4 plus 3 plus 2 plus 1, right and this is coming out to be here 15.

And similarly there is another function here p r o d which is for product; product means the multiplication of all the values inside the data vector. So, if I try to take here these 5 values 5 4 3 2 1, then this product function is trying to give us 5 into 4 into 3 into 2 into 1, right which is equal to here 120, right, ok. So, let me try to first show you here these two function square root, sum and product. So, now suppose if I take here, first let me clear the screen, as we discussed control L.

(Refer Slide Time: 12:32)

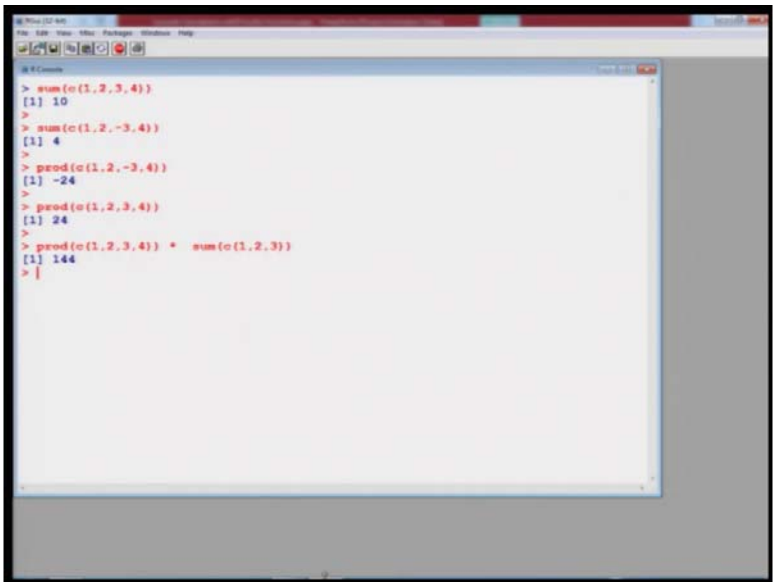


```
> sqrt(9)
[1] 3
>
> sqrt(2)
[1] 1.414214
>
> sqrt(c(4,9,16,25))
[1] 2 3 4 5
>
> sqrt(c(4,-9,16,25))
[1]   2 NaN   4   5
Warning message:
In sqrt(c(4, -9, 16, 25)) : NaNs produced
> |
```

Now, if I take here the function square root, suppose if I say square root of here 9, we can see here this is giving me here 3. And similarly if I try to take it here a square root of here 2, you know that is the popular value that we used to remember, that is 1.414. And similarly if I try to operate this square root function over a data vector, suppose I can say here 4 9 16 25. So, you know that these values are 2 3 4 5; that means, the square root function is really operating on individual data points inside the data vector, right. Similarly, well let me show you here one thing more that if I try to take here one value to be here negative, you know that what will happen. So, this is trying to give you here NaN, right. What is Na, what is NaN? That is what we are trying to that we will try to discuss in the forthcoming slides.

But, here you can see here this is indicating you that there is something wrong and there is a warning message and the warning message is clearly indicating that you cannot find out the real square root of the value -9, right. That is going to be an imaginary value.

(Refer Slide Time: 14:01)



```
> sum(c(1,2,3,4))
[1] 10
>
> sum(c(1,2,-3,4))
[1] 4
>
> prod(c(1,2,-3,4))
[1] -24
>
> prod(c(1,2,3,4))
[1] 24
>
> prod(c(1,2,3,4)) *  sum(c(1,2,3))
[1] 144
> |
```

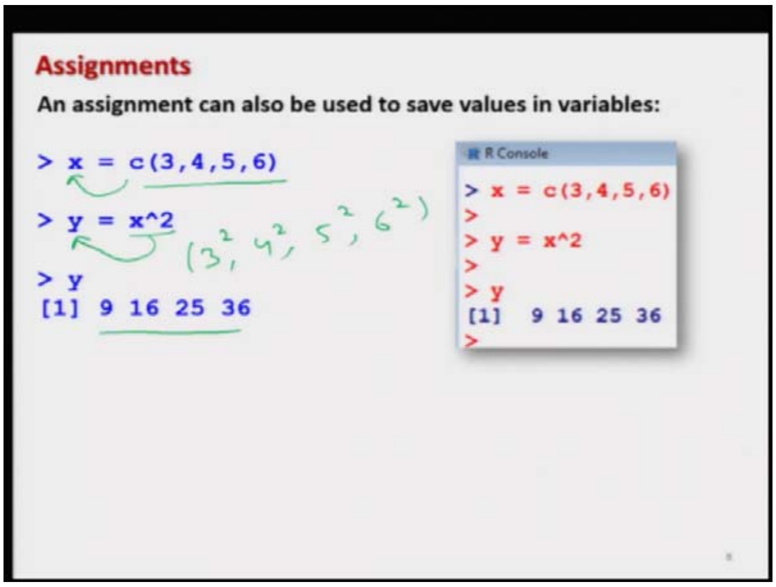
So, now we consider the function sum. So, let me try to give some data using the command c, say here sum of 1 2 3 4 which is 1 plus 2 plus 3 plus 4 that we know 1 plus 2 is 3, 3 plus 3 is 6, and 6 plus 4 is 10. So, you can see here the answer is coming out to be here 10, right.

And yeah, the similar thing can be done if you try to take here some minus value here, suppose if I take 1 2 -3 4, then this is going to be 1 plus 2 minus 3 plus 4. So, the answer is going to be here 1 plus 2 is 3, 3 minus 3 is 0 and then 0 plus 4 is 4, right.

Similarly, if I try to take here the product, so instead of here sum I try to write down here p r o d, so that will give me a value of 1 into 2 into minus 3 into 4. So, 1 2 is a 2, 2 3 is a 6, 6 4 is a 24 with the minus sign which is -24 you can see here.

And similarly if you do not use here the minus sign here, so that will be 1 into 2 into 3 into 4, right. And yeah means if you want to use the two functions together, for example I can use here the product and some function together. For example, I can find out the product of 1, 2, 3, 4 and then I can find out the sum of say 1, 2, 3. And suppose if I want to multiply the sum of 1, 2, 3 with the product of 1, 2, 3, 4. So, you can see here the answer is here how it is coming product of this 1, 2, 3, 4 is 4 3 is a 12, 12 2 is a 24 and this sum is here 3 plus 2 plus 1 which is here 6. So, 6 into this multiplication will give you the answer 144. So, you can see here that this functions can also be used together, right, ok.

(Refer Slide Time: 16:12)



## Assignments

**An assignment can also be used to save values in variables:**

```
> x = c(3,4,5,6)

> y = x^2

> y
[1]  9 16 25 36
```

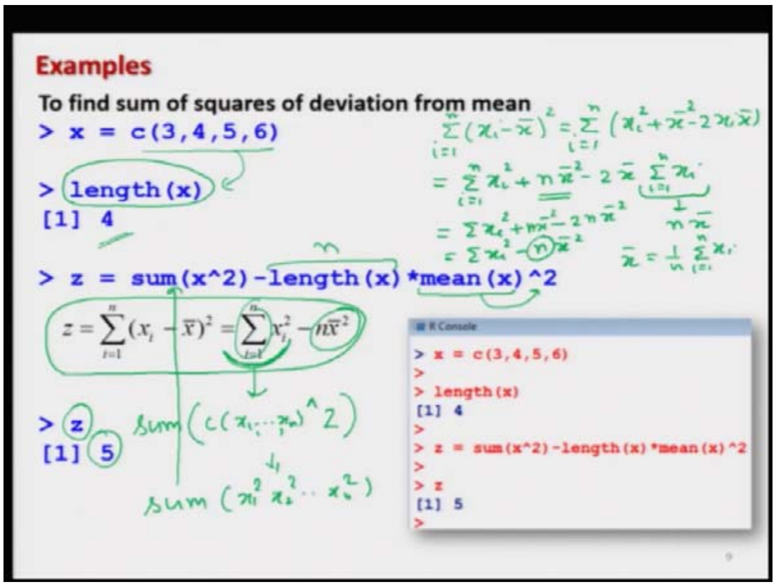
$(3^2, 4^2, 5^2, 6^2)$

```
> x = c(3,4,5,6)
>
> y = x^2
>
> y
[1]  9 16 25 36
>
```

So, let me try to come back on our slides. Now, you can see here whatever functions I have used here they can also be operated on the data vectors and their outcome can also be assigned to a variable.

For example, if I take here a data vector here say of four value 3, 4, 5, 6 and if I try to find out and if I assign it to a variable here x and if I try to find out the square of x which is the data vector. So, that we had discussed in the earlier lecture that this will become here say 3 square 4 square 5 square and 6 square. And then whatever is the outcome that will be assigned to a data vector here y new data vector and then y will come out to be here like this, ok.

(Refer Slide Time: 17:06)



## Examples

To find sum of squares of deviation from mean

```
> x = c(3,4,5,6)

> length(x)
[1] 4

> z = sum(x^2)-length(x)*mean(x)^2
```

$$z = \sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2$$

```
> z
[1] 5
```

$$\sum_{i=1}^{n}(x_i-\bar{x})^2 = \sum_{i=1}^{n}(x_i^2+\bar{x}^2-2x_i\bar{x})$$
$$= \sum_{i=1}^{n} x_i^2 + n\bar{x}^2 - 2\bar{x}\underbrace{\sum_{i=1}^{n} x_i}_{n\bar{x}}$$
$$= \sum x_i^2 + n\bar{x}^2 - 2n\bar{x}^2$$
$$= \sum x_i^2 - n\bar{x}^2$$
$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

sum(c(x1, ..., xn)^2)

sum(x1^2 x2^2 .. xn^2)

R Console

```
> x = c(3,4,5,6)
>
> length(x)
[1] 4
>
> z = sum(x^2)-length(x)*mean(x)^2
>
> z
[1] 5
>
```

So, that is another thing and now I would simply try to show you 1 simple application of this function, means as I said that these small functions will help you in bigger calculation.

For example in statistics, there is a very popular function which is here summation x i minus x bar whole square right, it gives us couple of things. So, if you try to open it here suppose if I say i goes from here 1 to n. So, if you try to open it here this will be $\sum_{i=1}^{n} x_i^2 + \bar{x}^2 - 2\sum_{i=1}^{n} x_i\,\bar{x}$ .

And if you try to open it here this will be $\sum_{i=1}^{n} x_i^2 +$ this will become $n\bar{x}^2 - 2\bar{x}$ and then $\sum_{i=1}^{n} x_i$ . So, this $\sum_{i=1}^{n} x_i$ is same as your $n\bar{x}$, why? Because your $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ .

So, this quantity will become here $\sum_{i=1}^{n} x_i^2 + n\bar{x}^2 -$ this will become here $2n\bar{x}^2$. So, this is $\sum_{i=1}^{n} x_i^2 - n\bar{x}^2$ and what is here? Your here n is the number of observations involved. And so suppose if I want to compute this quantity which I have written here like this, then in order to compute this quantity in R I need here 2 component; one is here x and say another is here n.

So, instead of using n as a number of observation which have to be supplied from external function, I can use here another function which is here length. Length function will give us the number of data values present inside a data vector So for example, if I try to take a data vector say c 3, 4, 5, 6, so there are 4 values here.

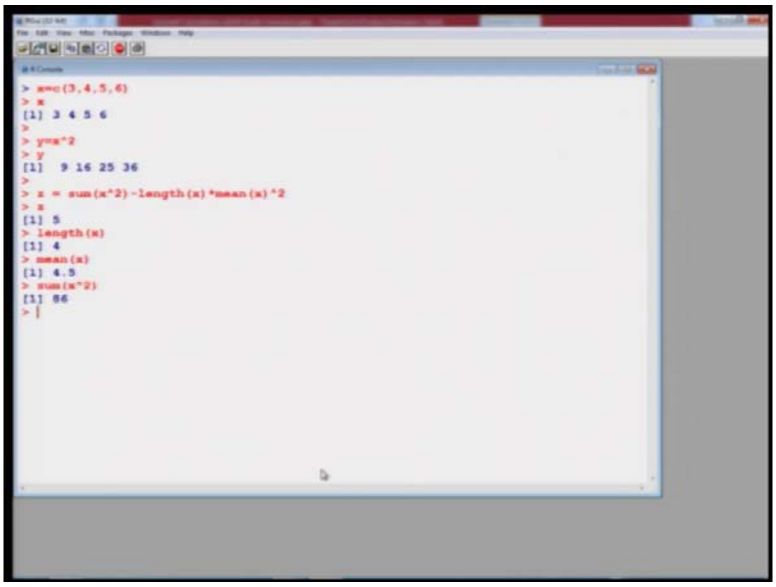
So, if I try to find out here length of x. So, this will come out here 4 there are 4 data values. Now, in case if I want to write this thing, then now if you try to see if I have here a data vector here $x_1$ $x_2$ $x_n$ something like this here c.

What I want? I want here first the square of individual values. So, using the rules of R I can simply write down here hat 2. So, this is going to give us the values $x_1^2, x_2^2, \ldots, x_n^2$ whatever are my data points and then after this I have to sum it. So, I can use here of means another command another function over here sum of this thing; sum of this thing.

So, you can see here this is what I have written here $\sum_{i=1}^{n} x_i^2$. So, you can see entire function can be written in a very simple form. And now this is here $n\bar{x}^2$, so n is given by here a length of x and $\bar{x}$ is obtained by mean of x and if you try to square it you will get here $\bar{x}^2$, right. So, you can see here such a function can be written in a one line very easily using the R commands.

So, if you try to take here these things this c equal to 3, 4, 5, 6 and if you will try to execute it you will get here the value of value of here z as 5. So, before going further let me try to show you these operations on the R console first.

(Refer Slide Time: 21:19)

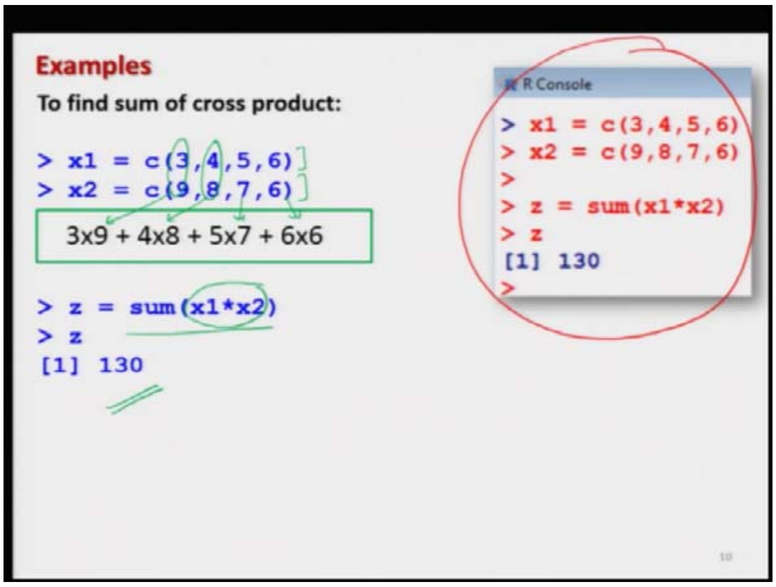

```
> x=c(3,4,5,6)
> x
[1] 3 4 5 6
>
> y=x^2
> y
[1]  9 16 25 36
>
> z = sum(x^2)-length(x)*mean(x)^2
> z
[1] 5
> length(x)
[1] 4
> mean(x)
[1] 4.5
> sum(x^2)
[1] 86
> |
```

So, for example, let me try to first take here a data vector here c say here say 3, 4, 5 and here 6, right. So, you can see here this is here x, so if you try to see the outcome of x this is x. Now, what I am trying to define? I am trying to define here y as x say square. So, this will be x hat 2 and if you try to see here the values of here y they will be simply the square of 3, 4, 5 and 6.

So, the outcome of the square is also assigned to a new variable here y and now suppose if I want to find out this function here $\sum_{i=1}^{n}(x_i - \bar{x})^2$ . So, I can copy here this function and I simply try to paste it to save the time, right.

So, you can see here this function comes out to be here 5, right. And here if you try to see the individual values length of here x will come out to be here 4, mean of here x will come out to be here 4.5 and so on, right. So, this is how you can see here and if you want to see what is here sum of x square so you can verify it here. So, sum of x square will be 9 plus 16 plus 25 plus 36 which is here 86, right.

(Refer Slide Time: 22:56)



**Examples**

To find sum of cross product:

```
> x1 = c(3,4,5,6)
> x2 = c(9,8,7,6)
```

3x9 + 4x8 + 5x7 + 6x6

```
> z = sum(x1*x2)
> z
[1] 130
```

R Console

```
> x1 = c(3,4,5,6)
> x2 = c(9,8,7,6)
>
> z = sum(x1*x2)
> z
[1] 130
>
```

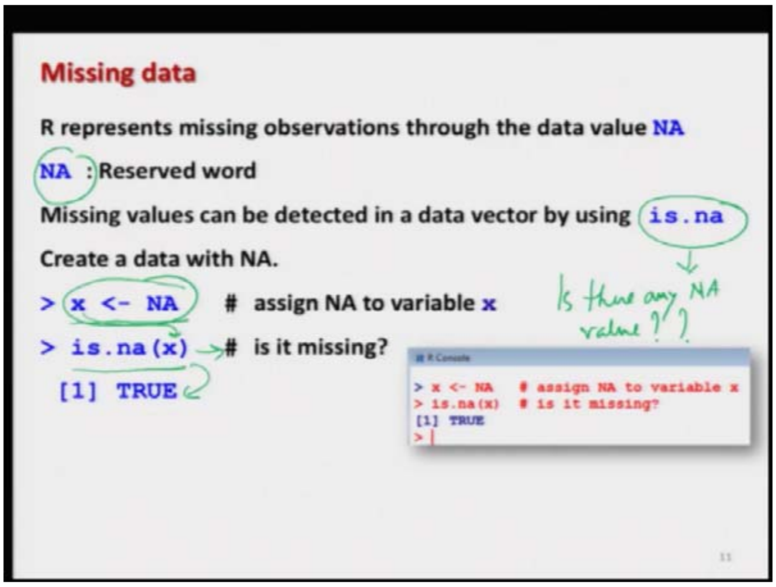
So now, you can see here that it is not that difficult to work in R. And similarly if I try to take here another command where I would like to show you that if you have two data vectors, then how different types of things can be obtained over here?

For example, if I take here 2 data vectors ,here x 1 equal to 3, 4, 5, 6 and x 2 is equal to say 9, 8, 7, 6 and suppose if you want to find out a function where you are trying to or where you want to multiply the corresponding numbers.

Corresponding numbers mean the number at the corresponding positions. For example, here 3 and 9 they are in the first position of the data vector, 4 and 8 they are in the second position of the data vector. So, what you want here you want to multiply 3 and 9, you want to multiply 4 and 8, and so on, you want to multiply 5 and 7 you want to multiply 6 and 6 and then you want to find out their sum.

So, now you can see here that first you need to find out the products. So, the so in order to find out the product I can simply use here a command x 1 star x 2 and then I have to find out the sum. So, just by writing sum of x 1 star x 2 I can get this outcome over here.

(Refer Slide Time: 24:17)



## Missing data

R represents missing observations through the data value NA

NA : Reserved word

Missing values can be detected in a data vector by using is.na

Create a data with NA.

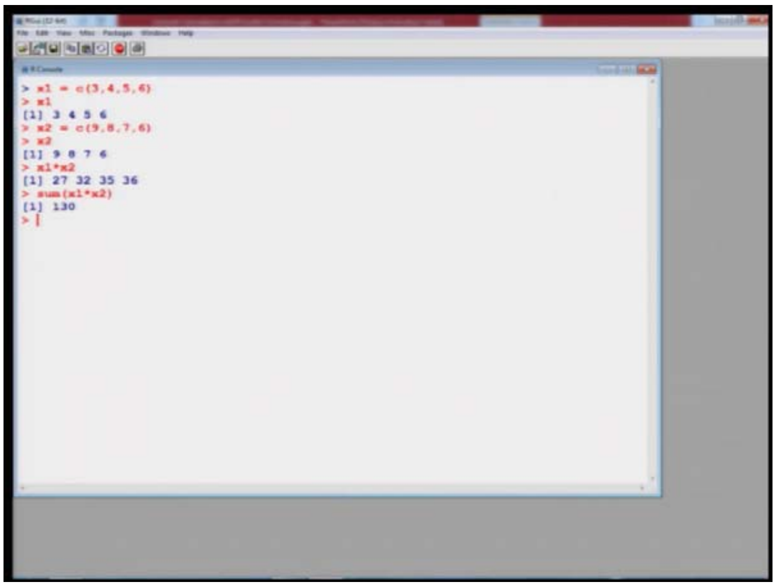
```
> x <- NA      # assign NA to variable x
> is.na(x)     # is it missing?
  [1] TRUE
```

Is there any NA value ??

```
> x <- NA      # assign NA to variable x
> is.na(x)     # is it missing?
[1] TRUE
> |
```

So, let us try to see it on the R console also, so that you get here more confident.

(Refer Slide Time: 24:33)



```
> x1 = c(3,4,5,6)
> x1
[1] 3 4 5 6
> x2 = c(9,8,7,6)
> x2
[1] 9 8 7 6
> x1*x2
[1] 27 32 35 36
> sum(x1*x2)
[1] 130
> |
```

So, first let me try to create this data vectors over here. So, you can see here this is my here x 1 and then I try to create here another data vector here. So, you can see here this is like this and if you try to find out their product x 1 into x 2 this is like this.

And if you write sum of x 1 into x 2 you can obtain 130 which is the same outcome that you have obtained here you can see here, right. And this is here the screenshot of the outcome, right, ok.

So, now I would try to address another issue. Whenever we are trying to deal with real data, there is always a possibility that some data values may be missing and in this statistic, this is very popular. For example, if I say suppose there is a medicine and I want to make a clinical experiment. So, I decide that I will take a group of 5 patients, I will call them every day and I will give them the medicine.

And every day I will try to record their say, for example blood pressure. Now there are five patients and they are supposed to come to me say for 7 days every day for 7 days. Now, suppose there is one patient which does not come on a particular day, suppose he does not turn up to my office on say third day.

Now, think of a situation, I have got all the data set, but there is one value which is missing. Now, I have two options; first option is this I would like to use my statistical tool on the data which is available to us and second option is this somehow using my statistical tool I will try to estimate or compute the value which is missing.

What does this mean? I would like to find out the value which would have been there if the patient came to my office on the third day also. So, using the available dat,a using the statistical technique, I can do it. But anyway that is not I am going to discuss here that is called as missing data modeling well, that is also a part of decision sciences. But surely this is out of the purview of this course, so I am not going to deal it here.

But what I am going to do it here that how I am going to use my different types of functions or different types of function which are available in R when there is a missing value. And what we want? We simply want that we will request R that Mr. R please compute the values by ignoring the missing values.

But when I am trying to make a request to R; R is a program R has to understand the language that how I am going to indicate that the value is missing. So, in order to indicate a missing value in R, the rule is the missing values are indicated by the symbol NA capital N capital A, right, there are some other notations which are used like as N A and NULL. So, those are different thing that I will try to address one by one.

So, at this moment I am going to address the role of NA that is capital NA. When I am going to address the issue of missing values you can imagine a situation  in statistics. Now you are going for decision sciences where the data size is going to be very huge there can be millions and billions of data values. So, you cannot see from your eyes and can check in the data set if the data value is missing or not.

Well, the first question comes that when you are trying to conduct a survey, when you are trying to collect the data, then you have to instruct the program or the programmer to write the program in such a way such that if any value is missing that will be indicated by capital N capital A.

If you want to use that data into R and if the programmer has used some other symbol, so you have to do the data cleaning, data preparation and you have to replace those values by capital N capital A otherwise R will not understand it and it will give you simply error.

So, my first come first job becomes that as soon as I get the data, first I have to determine; is there any missing value in my data? If the answer comes yes, then I have to use the tool appropriately and in case if the answer comes false; that means, there is no missing data values means I can use all the functions directly. For example, we up to now we have discussed some product maximum minimum etc.

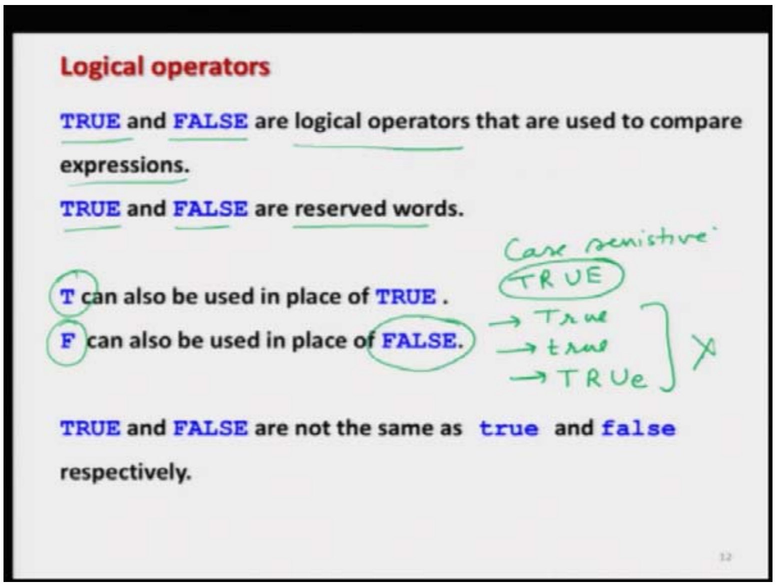
But we have assumed that all the data values are available, but these functions have to be modified if the data value is not available. So, these are the two simple question which I am going to now address. Firstly, how to determine whether a data set has a missing value or not and the second thing is this, how to handle the situation, how to modify your command what we have to do, right, ok.

So, one thing now we have understood that if a value is missing in R, the missing value is indicated by capital N capital A. And if you want to know whether a data set has a missing value or not, then we have a command here is dot na i s dot na which is say simply is there any N A value that is a simple question which I am going to ask my R.

And in order to illustrate it what do I mean? Suppose if I say suppose I try to define a variable and if I try to define the value here NA that x takes value NA, now in case if I use the command here is dot n a on what? On this here x which has to be given inside the parenthesis.

So, this command is trying to find out is there any missing value in the data vector x. Since you know that x is equal to NA this is what you have assigned yourself, so the answer will come out to be here true this is capital TRUE. There are two letters T R U E and F A L S E which are the reserved words which I will try to explain you, right. But TRUE means yes there is a missing value and if there is no missing value it will tell you FALSE, right.

(Refer Slide Time: 33:14)



## Logical operators

TRUE and FALSE are logical operators that are used to compare expressions.

TRUE and FALSE are reserved words.

T can also be used in place of TRUE .

F can also be used in place of FALSE.

TRUE and FALSE are not the same as true and false respectively.

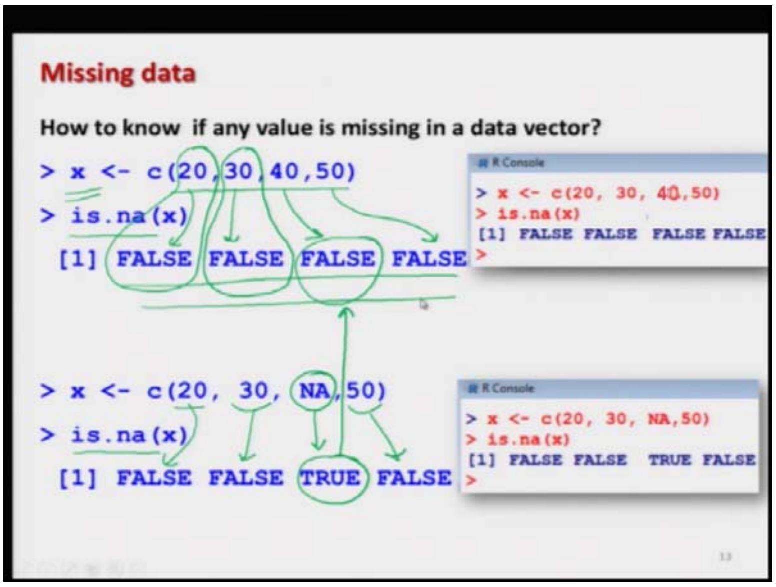
Case sensitive
TRUE
→ True
→ true
→ TRUe

So, now you can see here I have introduced here two more operators TRUE and FALSE, they are written in capital letters and actually TRUE and FALSE they are the logical operators and they are used to compare the expressions and values.

And TRUE and FALSE they are the reserved words, means you cannot use this TRUE and FALSE in capital letters as a variable name, right. And the and there is a possibility that instead of using TRUE and FALSE you can also use the first letter of TRUE and FALSE as T and F, but in capital letter right.

And remember one thing this is case sensitive, case sensitive means you have to use here only capital letter TRUE. If you try to write down here True if you write down here true or if you try to write down here t r u e, all this will not work only capital letter TRUE and capital letter F A L S E will work. So, that is what you have to keep in mind.

(Refer Slide Time: 34:47)



## Missing data

**How to know if any value is missing in a data vector?**

```
> x <- c(20,30,40,50)
> is.na(x)
 [1] FALSE FALSE FALSE FALSE
```

```
> x <- c(20, 30, 40,50)
> is.na(x)
[1] FALSE FALSE  FALSE FALSE
>
```

```
> x <- c(20, 30, NA,50)
> is.na(x)
 [1] FALSE FALSE TRUE FALSE
```

```
> x <- c(20, 30, NA,50)
> is.na(x)
[1]  FALSE FALSE  TRUE FALSE
>
```

And yeah, let me try to take here an example and then I will try to show you it on the R console also. Now suppose I take here four values 20, 30, 40 and 50 and I if I and I assign them to a data vector x.

Now, you can see here that there is no missing value, there is no value which is here indicated here as NA. But suppose I do not know the entire data vector, but I know that there are some values in x. So now, I can use here the command is dot n a on x; that means, I want to know is there any missing value in the data vector x the outcome will come out to be like this FALSE FALSE FALSE FALSE.

So, what is this indicating? This is indicating this 20 goes with the first FALSE, this 30 goes with the second FALSE, the 40 goes with the third FALSE and 50 goes with the fourth FALSE. So, it is saying that for the first value 20, the command is dot n a 20 is saying that it is FALSE; that means, there is no missing value.

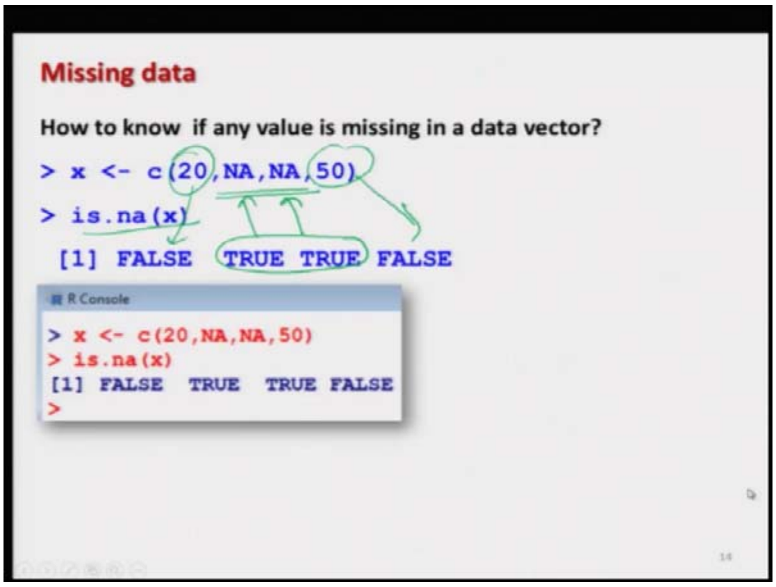
And you can see here this value is here 20 and there is no missing value. Now the operation comes on the second value 30 and it says that ok, this 30 value operator on is dot n a is FALSE; that means, there is no missing value and so on. So, the this is trying to tell you that there are no missing values in the data set.

Now, similarly what I do that in the same data vector I try to replace 40 by here NA, now I try to operate the same command here is dot na. So now, you can see here that in the third value it is here TRUE. Whereas, it means earlier in the third value it was FALSE.

So, what is this saying? The s dot n a command is going to 20 and it find that yeah, the value is present there is no missing value. So, it says FALSE, then it goes to second value it says FALSE because the value is available then it goes to third value and it says yes there is a value NA.

So that means this is TRUE; that means, the one value is missing and then it goes to fourth value which is saying that value 50 which is available. So, there is no missing value, right.

(Refer Slide Time: 37:28)



## Missing data

**How to know if any value is missing in a data vector?**

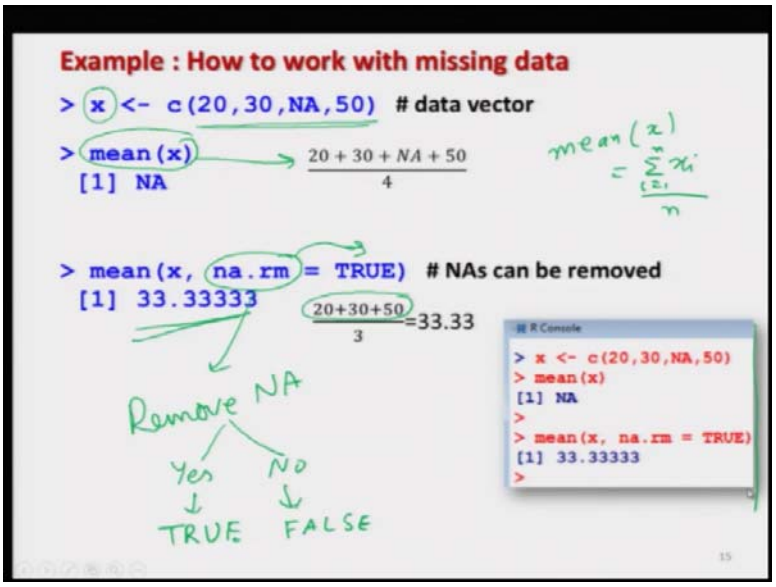
```
> x <- c(20,NA,NA,50)
> is.na(x)
 [1] FALSE  TRUE TRUE FALSE
```

```
> x <- c(20,NA,NA,50)
> is.na(x)
[1] FALSE  TRUE  TRUE FALSE
>
```

So, and in case if you try to take more than one missing values inside the data vector, even then this command works. For example, if I try to take here two missing values in the same data vector and if I try to operate the command here is dot na, you can see here these 2 values of TRUE they correspond to these two NAs.

That means yes there are two values NA which are or there are two values in the data vector which are missing and obviously those values which are present 20 and 50 corresponding to them it is giving us the outcome FALSE.

(Refer Slide Time: 38:11)



## Example : How to work with missing data

```
> x <- c(20,30,NA,50)  # data vector
> mean(x)
 [1] NA

> mean(x, na.rm = TRUE)  # NAs can be removed
 [1] 33.33333
```

$\frac{20 + 30 + NA + 50}{4}$

$mean(x) = \frac{\sum_{i=1}^{n} x_i}{n}$

$\frac{20+30+50}{3} = 33.33$

Remove NA

Yes → TRUE

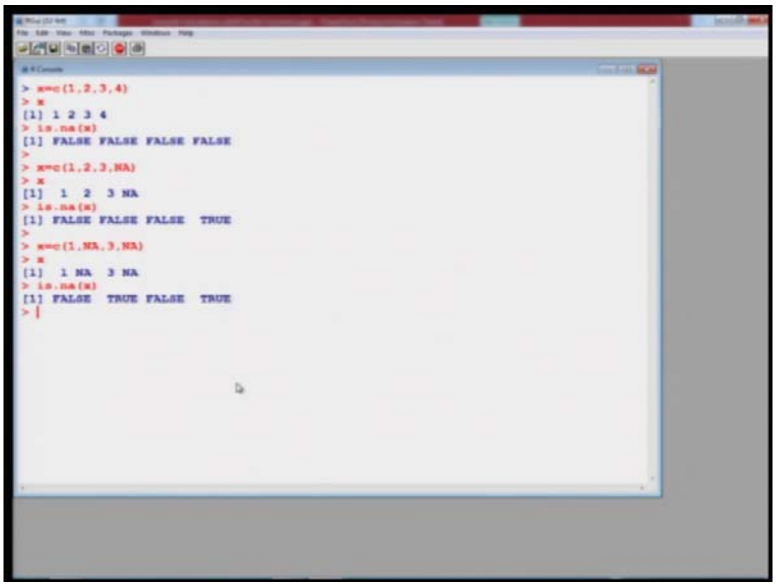
No → FALSE

```
> x <- c(20,30,NA,50)
> mean(x)
[1] NA
>
> mean(x, na.rm = TRUE)
[1] 33.33333
>
```

15

So, this is how we are going to do it, but before going further let me try to show you it on the R console.

(Refer Slide Time: 38:19)



```
> x=c(1,2,3,4)
> x
[1] 1 2 3 4
> is.na(x)
[1] FALSE FALSE FALSE FALSE
>
> x=c(1,2,3,NA)
> x
[1]  1  2  3 NA
> is.na(x)
[1] FALSE FALSE FALSE  TRUE
>
> x=c(1,NA,3,NA)
> x
[1]  1 NA  3 NA
> is.na(x)
[1] FALSE  TRUE FALSE  TRUE
> |
```

So, now we take a vector here say here x c 1, 2, 3, 4 yeah as simple vector and you can see here that all the values are here 1, 2, 3, 4 there is no value which is missing. So, if I try to use the command here is dot na on x you will see here it is giving you here FALSE FALSE FALSE FALSE; that means, all the values are available.

Now, what I do? I try to replace here one value in the same data vector here as say NA. Now, you can see here that there is one value in x which is missing. Now, if I try to take here is dot na, you can see here that everything remain the same as FALSE, but the last value here is changed to TRUE; that means, there is one missing value in the data vector.

And now if I try to change here say more values, suppose if I make 2 and 4 to be not available. So, you can see here there are two values which are missing and if I try to use the same command is dot na. So, this is giving me FALSE TRUE FALSE TRUE; that means, two values are missing two values are FALSE.

So, now this is how you can actually find that whether a value is missing in a given data set or not. Now, the next question arises suppose there is a missing value in the data set and suppose you want to operate some mathematical or statistical function. So, what do you want? You really want that the missing values should be removed from the data set and the function should be calculated on the basis of available values.

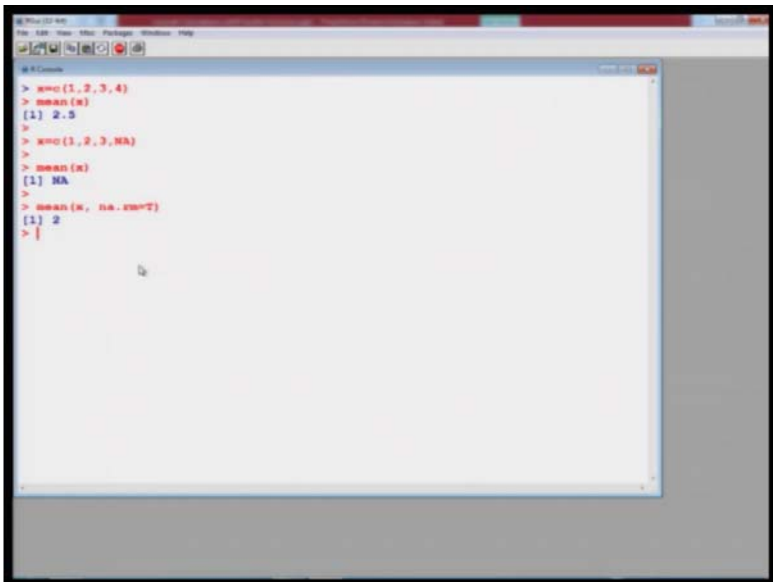
So, now I am trying to take here an example. Suppose I try to take here 4 values 20, 30, NA and 50 and I assign them to a variable here x and suppose I want to find out the arithmetic mean of the values in x. So, although we have not discussed here, but you know that mean is the command m e a n and inside the parenthesis, you try to give the data vector and this will try to give you the value here. So, all the sum of values inside the data vector divided by the total number of observation.

So, now once you try to take here the data vector 20, 30, NA and 50. So, and if you try to operate the command here mean on this x it will give you 20 plus 30 plus NA plus 50 divided by 4 which is here NA, because it cannot operate mathematically. So, now what do you want? You want to ask the software that Mr. R please look into the data set and if there is a value which is missing, please remove it and then execute the function on the remaining values.

So, in order to ask R, to remove the values, we have a command na dot rm, it simply means remove NA; that means remove the missing value- this is an instruction. Now, this instruction has to be followed or not- if you say yes remove it or you say no do not remove it.

So, if you say please remove the NA values, this will be indicated by TRUE and if you are saying n a dot r m is equal to FALSE. That means, please do not remove the missing value which is the default actually. So now, I am using here the command here n a dot r m is equal to TRUE. Now this mean function will remove the NA value and now it will compute the arithmetic mean on the basis of 20 plus 30 plus 50 divided by total number of observation which are now 3 after ignoring NA. And then it will give you the correct value here 33.33 which is the screenshot over here, I will try to show you it on the R console also, right, ok. So, you can see here that for example, if I try to take here x is equal to c say 1 2 3 4.

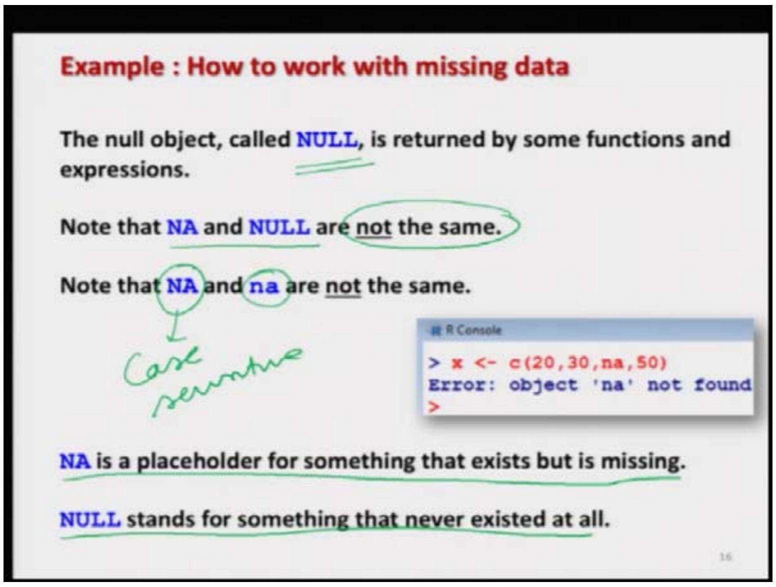
(Refer Slide Time: 42:37)



```
> x=c(1,2,3,4)
> mean(x)
[1] 2.5
>
> x=c(1,2,3,NA)
>
> mean(x)
[1] NA
>
> mean(x, na.rm=T)
[1] 2
> |
```

So, you can see here the mean of x is coming out to be here 2.5 which is obvious 1 plus 2 plus 3 plus 4 divided by 4 and if I try to make it here say 1 value to be here NA, then mean of x will come out to be here NA. But now if I try to use here the command mean of x na dot rm is equal to here true, then you can see here this is coming out to be 2. Why? This is now trying to find out the mean of 1, 2 and 3 here 1 plus 2 plus 3 divided by 3.

(Refer Slide Time: 43:16)


## Example : How to work with missing data

The null object, called NULL, is returned by some functions and expressions.

Note that NA and NULL are not the same.

Note that NA and na are not the same.

Case sensitive

```
> x <- c(20,30,na,50)
Error: object 'na' not found
>
```

NA is a placeholder for something that exists but is missing.

NULL stands for something that never existed at all.

16

So, this is how we can do it here ok. Now, let us try to come back on the on our slide and let me try to address the last topic of this lecture. There are two things in R which one is here NA and another here is NULL. And this is actually an object which is written by some functions or some expressions. Remember one thing that NA and NULL, they are not the same, these are two different things. What is the difference between the two?

The difference is this NA is a placeholder for something that exists but it is missing at the moment and NULL is the command or that indicates that something that never existed at all. What does this mean?

Let me take a simple example. Suppose there are 5 candidates which appear in a test and based on the test, only 3 students have to be admitted in a school. Suppose the students are named as 1, 2, 3, 4 and 5. Suppose after the exam the student number 1, 2 and 3, they got selected and student number 4 and 5, they are not selected.

Now, student number 1, 2, 3 starts coming to the classes and during the  attendance, suppose one day the student number 2 is absent. Now, my question is whether it is NA or NULL? Remember one thing student number 2 has become an authorized student of the school. But and his name is there in the attendance list, but today because of any reason he is missing. So, this is going to be indicated by NA.

But now suppose if you ask for candidate numbers 4 and 5, they could not qualify the examination, so they are not the student of the school. So, now in case if you take an attendance and if you roll call student number 4 or 5, what they will be NA or NULL? That will be NULL, because that they never existed, they were not admitted. So, they were not the part of the school. So, these values are going to be denoted by NULL which never existed and NA is missing, right.

And then remember one thing, that this capital NA this is also case sensitive; case sensitive means this has to be used only with the capital letters capital NA and small n a they are not the same. So, this is what you have to keep in mind.

So, now let me stop in this lecture. So, I have tried to give you some more aspect how to use the built in function, how to handle the missing values and so on.

And all these things, they are an integral part of a decision science, whenever you have a big data  you cannot do all the things with your hand manually, you cannot do all the calculations, you have to depend on the software command, your data size might be millions and billions and trillions of bytes.

So, in those situation, you always need a statistical tool or a programming tool which can give you the correct outcome. So, all this thing whatever I have used here they may look very simple thing very trivial things, but believe me they are going to be extremely useful when you are going to deal with the real data science.

So, you try to practice it try to settle your commands inside your mind, take some example from the book from the assignment, practice them and I will see you in the next lecture till then good bye.