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

Sampling Theory with R Software
Lecture - 38
Bootstrap Methodology
Example of Bootstrap Analysis Using boot Package

Hello friends, welcome to the course Essentials of Data Science with R Software 2 where we are trying to learn the topics of Sampling Theory and Linear Regression Analysis. In this module on the Sampling Theory with R Software, we are going to learn the topic of Bootstrap Methodology with R Software.

So, you can recall that in the earlier lecture I have explained the theory and fundamentals behind the bootstrap methodology and then, I also explained how to compute them on the R software, how to compute the value of the estimator, bias, standard error, and confidence interval. So, now, in this lecture, I am simply going to take one more example and whatever I had done, this I will try to do in a single lecture, but on a different function.

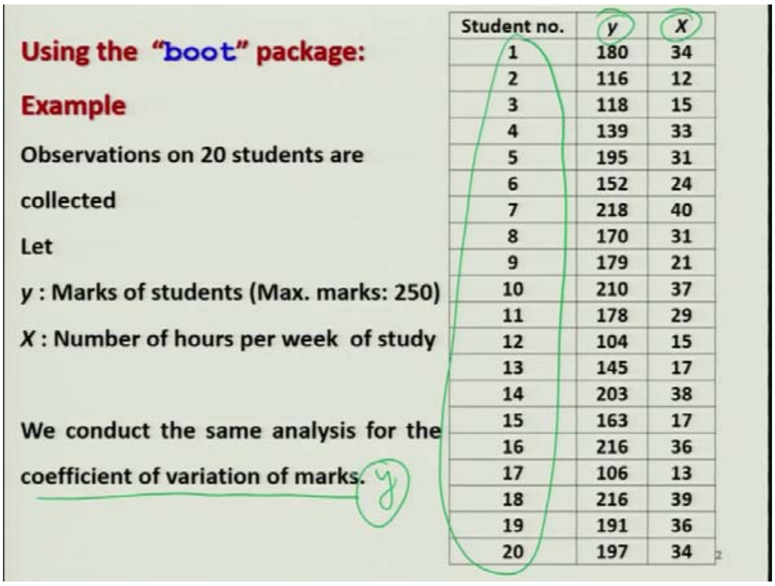
Just to make you comfortable what I have done that in the earlier example, I had considered the bootstrap estimator for the correlation coefficient, but now in this case, I will take a new statistic, I will be using coefficient of variation and coefficient of variation is defined as the ratio of standard deviation and mean.

So, this is also an important criterion in data sciences to where you can take the conclusion when you have more than one data sets. So, but in case if I ask you that how you will find out the bias or standard error or confidence interval for the coefficient of variation, it is difficult to find out the exact expressions or the finite sample expressions.

Now, this bootstrap methodology comes to help us and using the bootstrap methodology, I will try to show you here through this example that how we can compute the value of the coefficient of variation, its standard error as well as coefficient confidence interval using the bootstrap techniques using the package boot.

And similarly, well if you want to use the package bootstrap also to construct the confidence interval based on the t method that goes straight forward without any further discussion and I am confident that you will be able to do it. So, I will try to explain you all the things over here. So, let us begin our lecture.

(Refer Slide Time: 02:57)



Using the "boot" package:

Example

Observations on 20 students are collected

Let

y : Marks of students (Max. marks: 250)

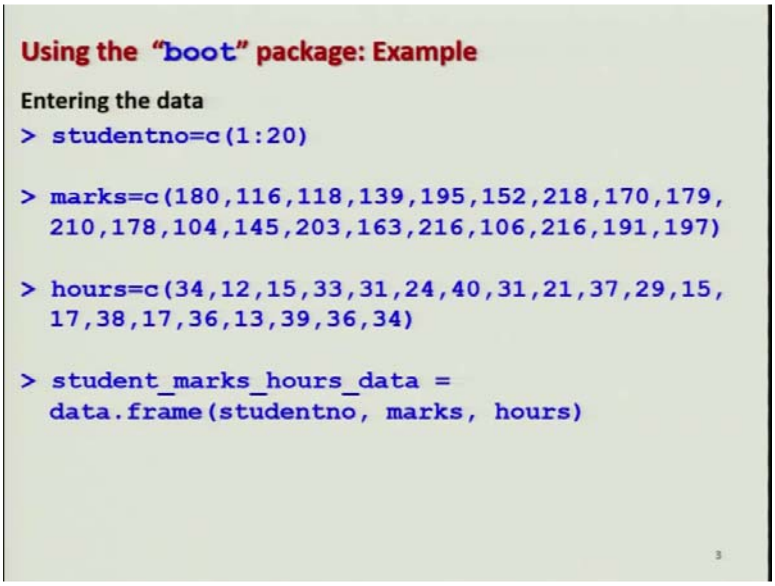
X : Number of hours per week of study

We conduct the same analysis for the coefficient of variation of marks. y

Student no.	y	X
1	180	34
2	116	12
3	118	15
4	139	33
5	195	31
6	152	24
7	218	40
8	170	31
9	179	21
10	210	37
11	178	29
12	104	15
13	145	17
14	203	38
15	163	17
16	216	36
17	106	13
18	216	39
19	191	36
20	197	34

So, you can recall that I had considered this example earlier where I had taken 20 students and I had recorded their marks as y and the number of hours they studied as x and earlier, we had done the analysis for the correlation coefficient, but in this case, we are going to consider the coefficient of variation for marks, right. So, this here y and, if you want to do it for x, exactly on the same lines you can do it.

(Refer Slide Time: 03:29)



Using the "boot" package: Example

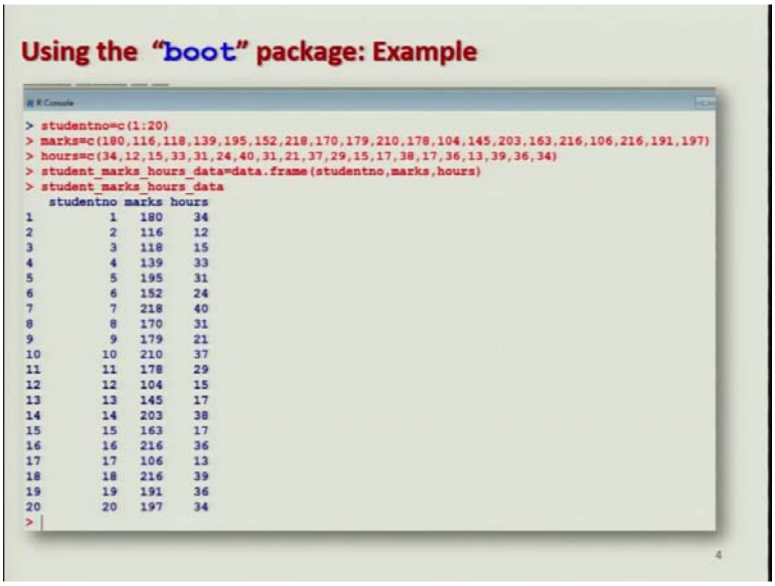
Entering the data

```
> studentno=c(1:20)

> marks=c(180,116,118,139,195,152,218,170,179,
  210,178,104,145,203,163,216,106,216,191,197)

> hours=c(34,12,15,33,31,24,40,31,21,37,29,15,
  17,38,17,36,13,39,36,34)

> student_marks_hours_data =
  data.frame(studentno, marks, hours)
```

3

So, you can see here, here I have created the data set just like in the earlier case, right.

(Refer Slide Time: 03:39)



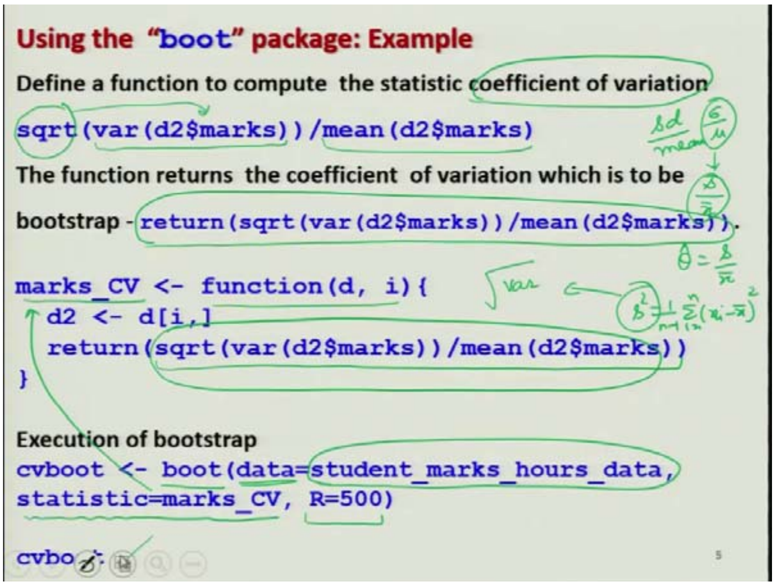
Using the "boot" package: Example

```
> studentno=c(1:20)
> marks=c(180,116,118,139,195,152,218,170,179,210,178,104,145,203,163,216,106,216,191,197)
> hours=c(34,12,15,33,31,24,40,31,21,37,29,15,17,38,17,36,13,39,36,34)
> student_marks_hours_data=data.frame(studentno,marks,hours)
> student_marks_hours_data
   studentno marks hours
1          1   180    34
2          2   116    12
3          3   118    15
4          4   139    33
5          5   195    31
6          6   152    24
7          7   218    40
8          8   170    31
9          9   179    21
10        10   210    37
11        11   178    29
12        12   104    15
13        13   145    17
14        14   203    38
15        15   163    17
16        16   216    36
17        17   106    13
18        18   216    39
19        19   191    36
20        20   197    34
> 
```

4

And this is the screenshot of the data frame which I have created. So, this I already have explained you couple of times. So, I will not discuss it much.

(Refer Slide Time: 03:48)



Using the "boot" package: Example

Define a function to compute the statistic coefficient of variation

```
sqrt(var(d2$marks))/mean(d2$marks)
```

$\frac{sd}{mean}$ $\frac{\sigma}{\mu}$ $\frac{s}{\bar{x}}$

The function returns the coefficient of variation which is to be bootstrap - `return(sqrt(var(d2$marks))/mean(d2$marks))`.

$\hat{\theta} = \frac{s}{\bar{x}}$

```
marks_CV <- function(d, i){
  d2 <- d[i,]
  return(sqrt(var(d2$marks))/mean(d2$marks))
}
```

$\sqrt{var}$ $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2$

Execution of bootstrap

```
cvboot <- boot(data=student_marks_hours_data,
statistic=marks_CV, R=500)

cvbo
```

5

Now, I come to my objective. My objective is that I want to compute the bootstrap estimates for the coefficient of variation, right. Coefficient of variation is defined as say standard deviation divided by mean. So, if you try to take the say this normal population with mean μ and variance σ^2, then it is defined as a σ/μ, but again it depends on the unknown parameters so, it has to be estimated and there is no estimator which can directly estimate the coefficient of variation.

So, one possibility is that I can just replace this population value by their sample variance. For example, σ can be replaced by here s, the variance based on the sample and μ can be replaced by the sample mean. So, this becomes our estimator, and this is something like our $\hat{\theta}$ as $s/\bar{x}$ where your $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$, right. So, this is what I want to do.

So, essentially if you try to see here, what I want? I want here the square root of the variance and we know that this s square quantity can be directly computed using the R function var. So, what I need? I need only here the square root of this thing. So, what I try to do here? That I try to define my coefficient of variation for the given data.

So, this you can see here, this is the variance of marks and this is the mean of marks, but now I need the standard deviation. So, I am trying to take the square root of the variance of marks, right. So, this is to be estimated. And we have to write a function which can return this value because this function has to be feeded in the boot function.

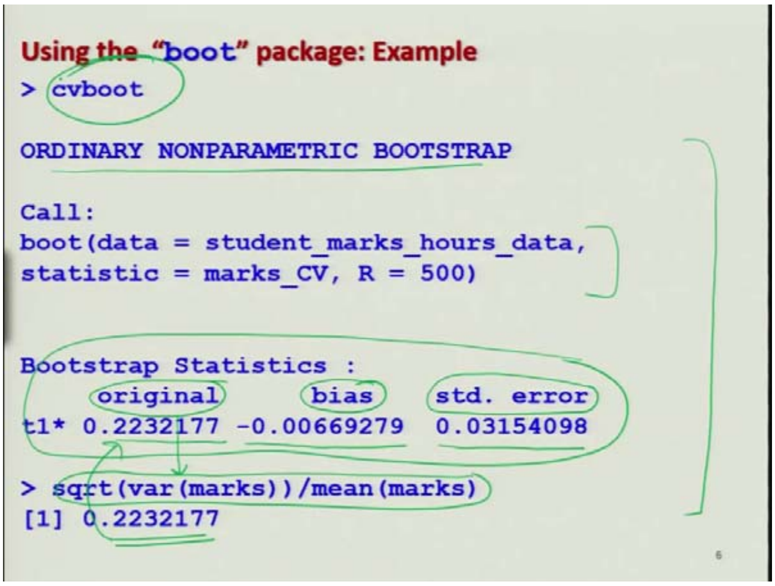
So, I will be using this command exactly in the same way as we did in the case of correlation coefficient and now first, I write try to write down that function. So, the function I have given a name marks_CV that is coefficient of variation and this is a function of two arguments d and i and this example exactly on the same way as we have defined the d and i in the earlier case.

So, but now instead of correlation coefficient, I am writing here square root of variance divided by mean, right. So, what will happen? We have the data, from that data a bootstrap sample will be generated and for that given bootstrap sample, this quantity will be computed by this function, right ok.

So, now, I come to the execution of bootstrap so, I will be using here the same command boot the what is here? Data. Data is the same, the data frame in which we have given the marks as well as hours so, this is here the name of the data frame which we are going to use and from where we are going to supply the data for the bootstrapping and now, what is the; what is the statistic which we want to compute? This is here marks dot CV, right.

And definitely, I would like to generate the bootstrap sample so, I am writing here R is equal to 500 that ok means I will be estimating all this thing based on 500 boot sample and then, I try to get the outcome here with the cvboot.

(Refer Slide Time: 07:54)



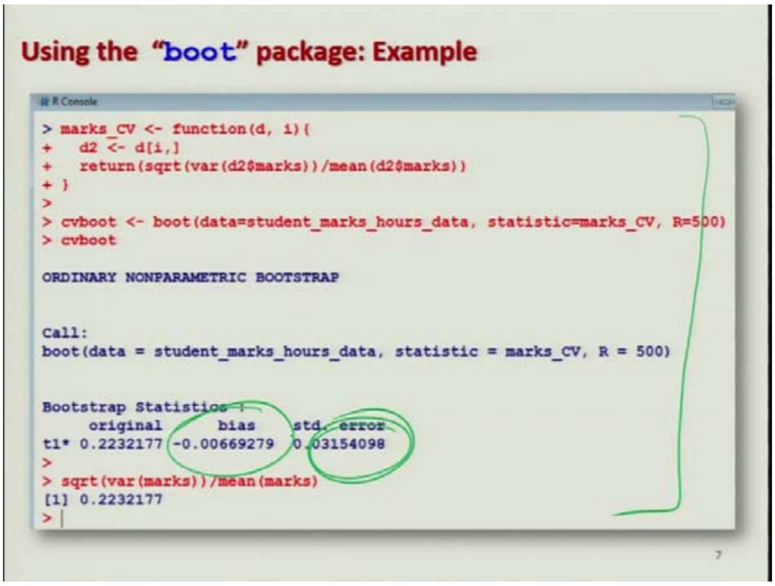
Using the "boot" package: Example

```
> cvboot

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = student_marks_hours_data,
statistic = marks_CV, R = 500)

Bootstrap Statistics :
     original        bias    std. error
t1* 0.2232177 -0.00669279  0.03154098

> sqrt(var(marks))/mean(marks)
[1] 0.2232177
```

So, now, if you try to do it here, this is the outcome which you will be getting here. So, the outcome framework is similar to the earlier case also. So, this is giving us the ORDINARY NON-PARAMETRIC BOOTSTRAP, this is the command that we have used and here is the main outcome. So, you can see here that this is the original value means if you try to find out the coefficient of variation based on the original data, original sample, this will come out to be like this 0.223.

And the bootstrap bias of coefficient of variation is indicated here which is minus 0.00669279 and the standard error of the coefficient of variation which is actually the bootstrap standard error, it is coming out to be 0.03154098 and just to make you confident that whatever is this original is the coefficient of variation which is computed on the basis of the original sample so, I have computed it separately from the given data sets and you can see here that both the numbers are matching, right ok.

(Refer Slide Time: 09:06)

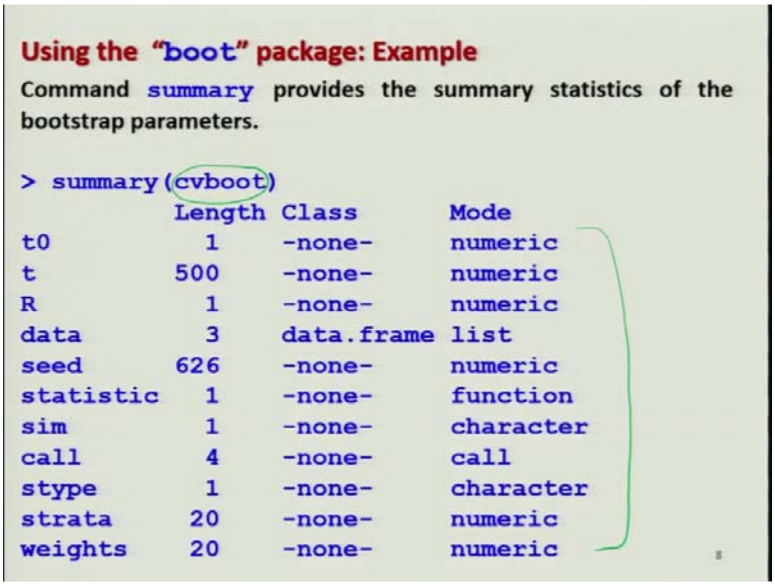


Using the "boot" package: Example

```
> marks_CV <- function(d, i){
+   d2 <- d[i,]
+   return(sqrt(var(d2$marks))/mean(d2$marks))
+ }
>
> cvboot <- boot(data=student_marks_hours_data, statistic=marks_CV, R=500)
> cvboot

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = student_marks_hours_data, statistic = marks_CV, R = 500)


Bootstrap Statistics :
     original       bias    std. error
t1* 0.2232177 -0.00669279  0.03154098
>
> sqrt(var(marks))/mean(marks)
[1] 0.2232177
> |
```

So, and this is the screenshot of the same operation which I have shown you here ok.

(Refer Slide Time: 09:14)



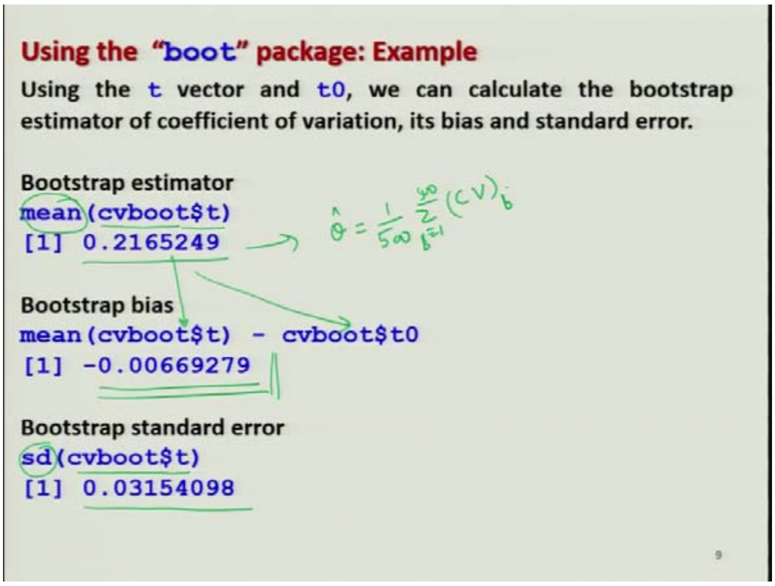
Using the "boot" package: Example

Command summary provides the summary statistics of the bootstrap parameters.

```
> summary(cvboot)
          Length Class      Mode
t0          1    -none-     numeric
t         500    -none-     numeric
R           1    -none-     numeric
data        3    data.frame list
seed      626    -none-     numeric
statistic   1    -none-     function
sim         1    -none-     character
call        4    -none-     call
stype       1    -none-     character
strata     20    -none-     numeric
weights    20    -none-     numeric
```

Now, you can see here, now I will try to find out the summary of this boot object. So, I have stored the outcome in the cvboot, you can see here cvboot and so, the summary will give you all the related information which has been used in the bootstrap computations, right so, right. So, that is the same thing which we have done earlier.

(Refer Slide Time: 09:39)



Using the "boot" package: Example

Using the t vector and t0, we can calculate the bootstrap estimator of coefficient of variation, its bias and standard error.

Bootstrap estimator

```
mean(cvboot$t)
[1] 0.2165249
```

$\hat{\theta} = \frac{1}{500} \sum_{b=1}^{500} (CV)_b$

Bootstrap bias

```
mean(cvboot$t) - cvboot$t0
[1] -0.00669279
```

Bootstrap standard error

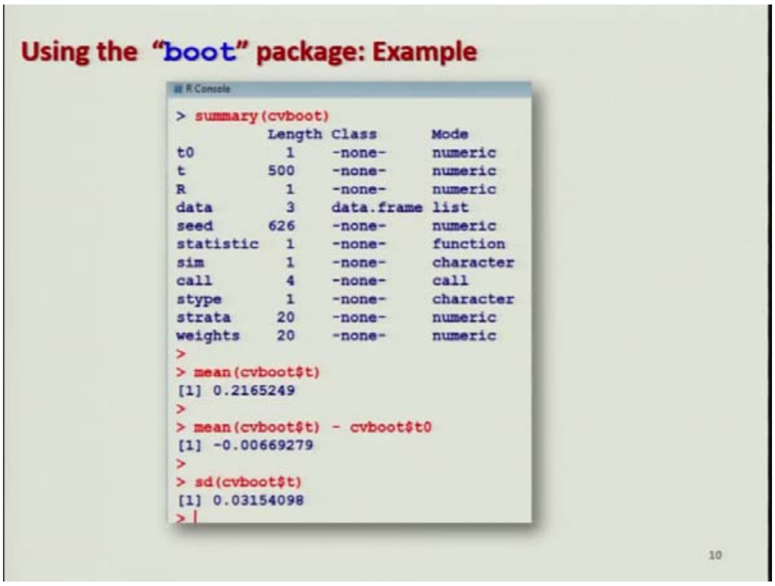
```
sd(cvboot$t)
[1] 0.03154098
```

So, now, I come that I want to find out the bootstrap estimator of coefficient of variation. So, I try to use the boot object that we have a store cvboot followed by dollar and t. So, this will and then, we try to find out the mean of the values. So, this comes out to be 0.2165. So, this is essentially the sort of $\hat{\theta}$ which is based on 1 upon 500 values i goes from 1 to here 500 based on the cv which is computed for ith sample or I may you I should use here b which is our usual symbol, right.

And if you want to find out the bootstrap bias so, I can take out this, I can consider this value minus cvboot dollar t naught that was given in the data and you can obtain here the bias so, you can verify that this value is the same value which you had obtained here, here in the software outcome. So, there should not be any doubt.

And similarly, if you separately want to find out the bootstrap standard error of the objective cvboot so, for the same bootstrapped value which have been stored under the cvboot dollar t, I can find out simply the standard deviation so, this will come out to be 0.03154098 and you can compare it here, this is the same thing which we have obtained here, right ok. So, this is how you can find out the estimator, its bias as well as the standard error, right.

(Refer Slide Time: 11:31)



Using the "boot" package: Example

```
> summary(cvboot)
          Length Class      Mode
t0          1    -none-     numeric
t         500    -none-     numeric
R           1    -none-     numeric
data        3    data.frame list
seed      626    -none-     numeric
statistic   1    -none-     function
sim         1    -none-     character
call        4    -none-     call
stype       1    -none-     character
strata     20    -none-     numeric
weights    20    -none-     numeric
>
> mean(cvboot$t)
[1] 0.2165249
>
> mean(cvboot$t) - cvboot$t0
[1] -0.00669279
>
> sd(cvboot$t)
[1] 0.03154098
> |
```

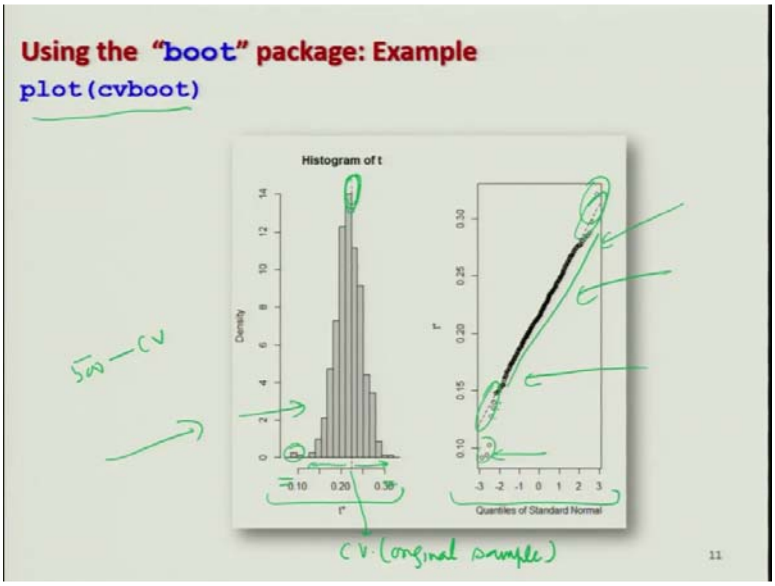
10

And this is the screenshot of the outcome which I have shown you here because this screenshot is not going to be the same when I try to do the same thing on the R console because then my bootstrap sample will be different unless and until I use the same seed value, ok.

(Refer Slide Time: 11:51)

Using the "boot" package: Example

plot(cvboot)



11

Now, I come on the aspect of confidence interval. So, what I try to do? First, I try to have a plot, from the plot I can have an idea whether we do not have any extreme observation or something like this. So, the command plot cvboot will give us a plot like this one which consists of a histogram and a Q-Q plot, quantile-quantile plot.

So, whatever 500 values I have obtained for the cv, right, these have these values have been plotted in this histogram here and the normal quantile-quantile plots, they have been obtained and plotted here in this graph so, every this dot is indicating the value, the corresponding value and you can see here that in the case of histogram, this is here the dotted value. So, dotted value is trying to indicate that what is the value of for example, here, the value of coefficient of variation based on the original sample.

So, you can see here that if this is the value not much variation is there between the true value which is obtained on the basis of original sample and the bootstrap value, you can see here that they are going up to 0.1 to 0.3 and the average value is and it is the value around 0.22 or so, right. So, you can see here, there are some values which are here, which are missing, but you can generate those values if you try to increase the number of bootstrap samples. So, you can see here that it is not actually bad.

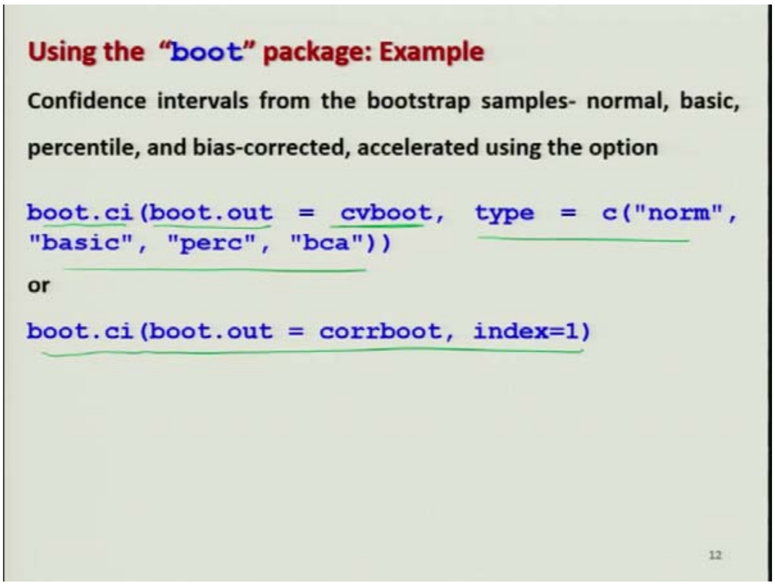
Now, in case you if; in case if you try to look into the Q-Q plot here, you can see here this is here the line which is passing through here something like this and all the points, I will remove this line and all the points which is which are clustered here, these are the 500 values of the coefficient of variation obtained from each of the bootstrap sample, they are also lying almost on the line, there are some values here which are here and here which are away from the line.

But, but that will not make much difference, but I would suggest you that you try to have a look on these numbers that why they are going so away from the point and then, based on that, you have to take a proper conclusion. So, that may require data cleaning also sometime or if you try to generate a new sample, possibly your things may change.

So, looking at this quantile-quantile plot, I can be assured that my samples for coefficient of variation are coming from a population which can be considered as normal because what will really happen that suppose if I come in an abstract way so, then we have to consider a population which is normal.

And this population is normal with respect to the coefficient of variation and from there, we have to draw a large number of sample and then I have to plot this Q-Q plot. But that is not possible for us. So, we are trying to draw the sample from an from a sample and then, I am trying to just compare whether the bootstrap sample which we have drawn from the original sample, they can be treated as if they are coming from a normal population or not. So, this so, we have a reason to be happy here that this assumption is satisfied. So, your confidence interval will be reasonably good.

(Refer Slide Time: 15:34)



Using the "boot" package: Example

Confidence intervals from the bootstrap samples- normal, basic, percentile, and bias-corrected, accelerated using the option

```
boot.ci(boot.out = cvboot, type = c("norm",
"basic", "perc", "bca"))
```

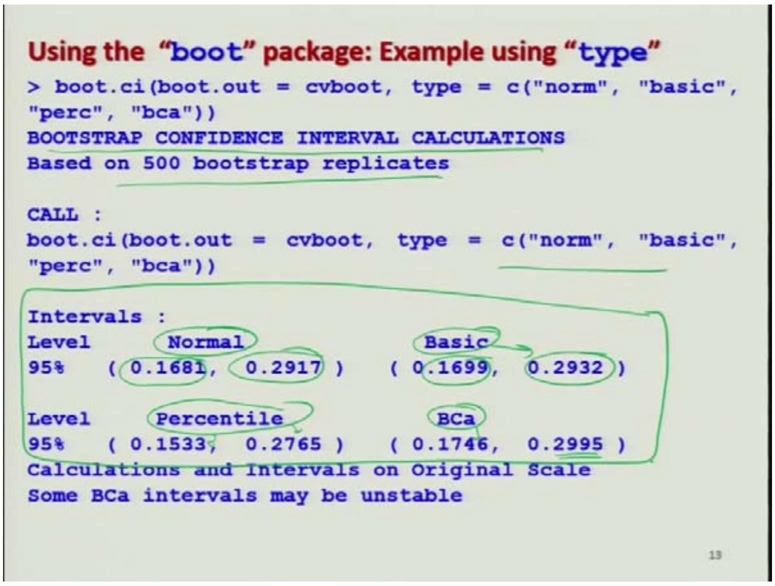
or

```
boot.ci(boot.out = corrboot, index=1)
```

12

So, now, I try to construct the confidence interval, I use the command boot dot ci. So, boot dot out that is required, this is the object where I have stored the outcome of coefficient of variation and then, I am giving here the type equal to means all four type of confidence interval or I can use here the alternative command boot dot ci with index is equal to 1, you can use means any one of the two.

(Refer Slide Time: 16:13)



Using the "boot" package: Example using "type"

```
> boot.ci(boot.out = cvboot, type = c("norm", "basic",
"perc", "bca"))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, type = c("norm", "basic",
"perc", "bca"))

Intervals :
Level       Normal               Basic
95%    ( 0.1681,  0.2917 )   ( 0.1699,  0.2932 )

Level      Percentile            BCa
95%    ( 0.1533,  0.2765 )   ( 0.1746,  0.2995 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
```

13

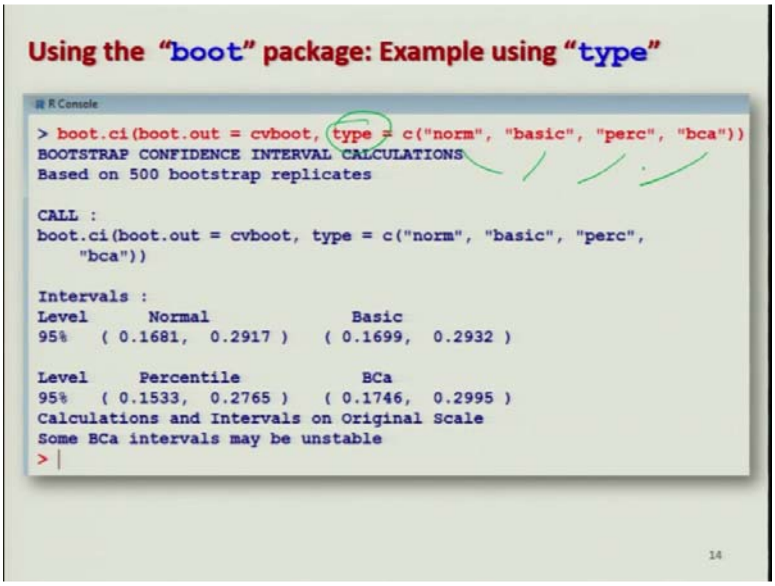
And if you try to execute it on the R console, you will get here this type of outcome and you can see here so, this is saying that Bootstrap Confidence Interval Calculation based on 500 bootstrap replicates and this is the command and here is the outcome of the confidence interval.

So, this confidence interval here is Normal and it is saying that the lower confidence limit is 0.1681 and upper confidence limit is 0.2917. Similarly, the Basic confidence interval is 0.1699 and upper limit is 0.2932. The confidence interval based on the Percentile method this is 0.1533, 2.2765 and the confidence interval based on the BCa, this is from 0.161746 to 0.2995.

And you can see here one very important thing, you can see here all the confidence interval which you have obtained they are more or less similar, there is not much difference and one of the reason that why this difference is not much because from this Q-Q plot, you can see here that the distribution of the bootstrap value is not actually bad this is reasonably good.

And this is comparable with normal which is a symmetric curve and that is a smooth curve that is why this confidence intervals are coming out to be nice. So, this is how you try to take different types of conclusion from the same set of values, right.

(Refer Slide Time: 17:54)



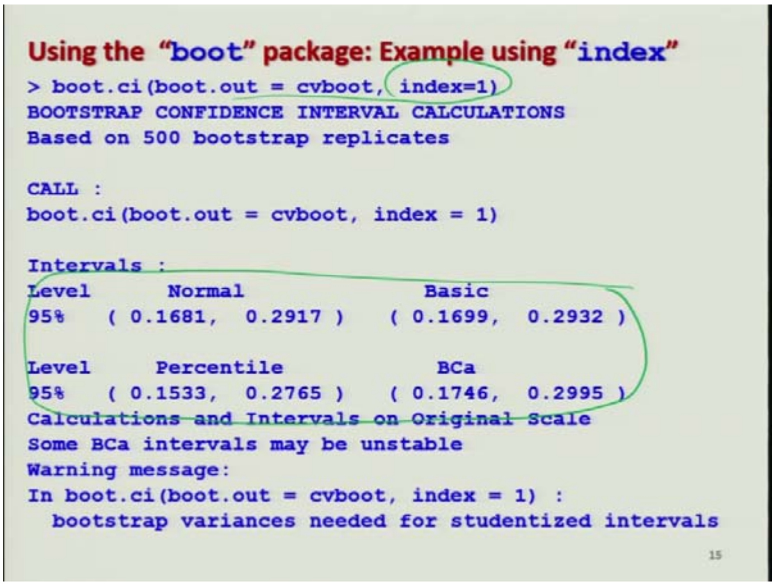
Using the "boot" package: Example using "type"

```
> boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc", "bca"))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc",
    "bca"))

Intervals :
Level      Normal              Basic
95%   ( 0.1681,  0.2917 )   ( 0.1699,  0.2932 )

Level     Percentile            BCa
95%   ( 0.1533,  0.2765 )   ( 0.1746,  0.2995 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
> |
```

14

And this is here the screenshot of the same outcome ok.

(Refer Slide Time: 18:02)



Using the "boot" package: Example using "index"

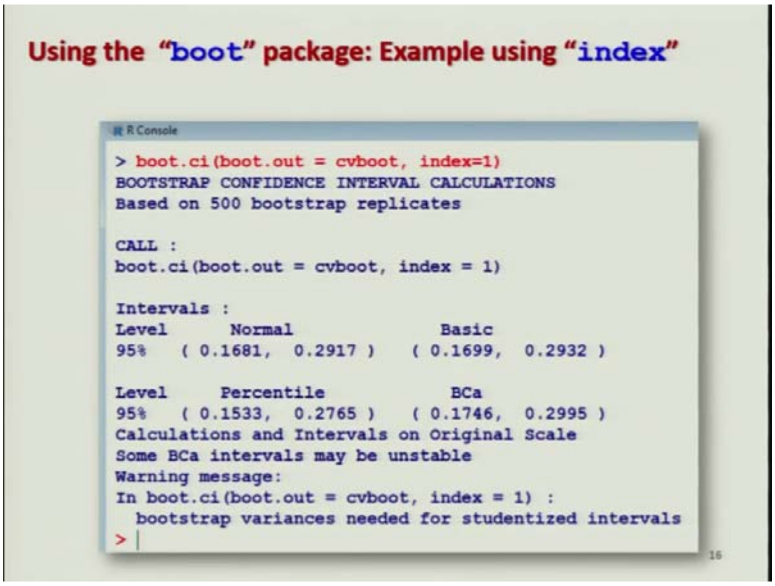
```
> boot.ci(boot.out = cvboot, index=1)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, index = 1)

Intervals :
Level      Normal              Basic
95%   ( 0.1681,  0.2917 )   ( 0.1699,  0.2932 )

Level     Percentile            BCa
95%   ( 0.1533,  0.2765 )   ( 0.1746,  0.2995 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
Warning message:
In boot.ci(boot.out = cvboot, index = 1) :
  bootstrap variances needed for studentized intervals
```

15

So, and in case if you try to use the alternative command means index equal to 1 so, I try to do the same command with here index equal to 1 and you can see here that these are the four types of confidence interval that you have obtained which are the same thing. So, this is your choice which one you want to use here.

In the index equal to 1, you have no option to obtain the individual confidence interval whereas, in the earlier command, you have an; you have an option that when you are trying to define here the here type, you can choose only one, two or all depending on your need, right.

(Refer Slide Time: 18:43)

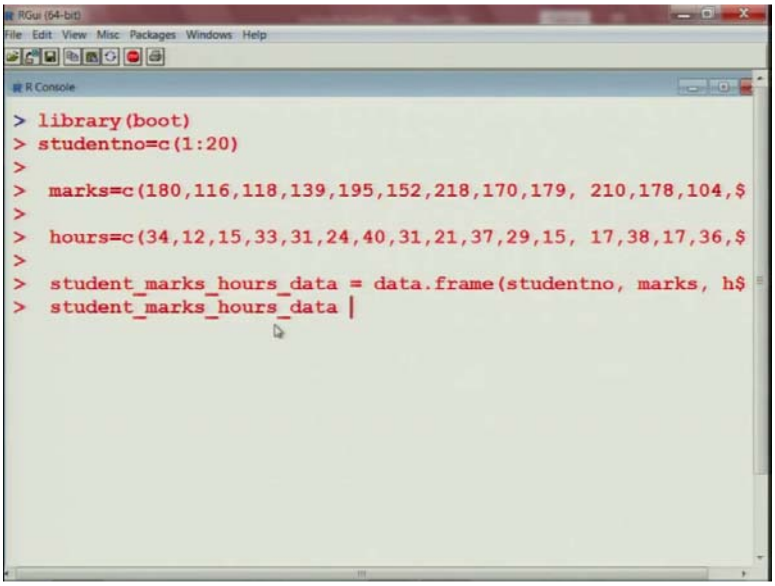


Using the "boot" package: Example using "index"

```
> boot.ci(boot.out = cvboot, index=1)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, index = 1)

Intervals :
Level      Normal              Basic
95%   ( 0.1681,  0.2917 )   ( 0.1699,  0.2932 )

Level     Percentile            BCa
95%   ( 0.1533,  0.2765 )   ( 0.1746,  0.2995 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
Warning message:
In boot.ci(boot.out = cvboot, index = 1) :
  bootstrap variances needed for studentized intervals
> 
```

So, and this is the screenshot of the same outcome when I am using index.

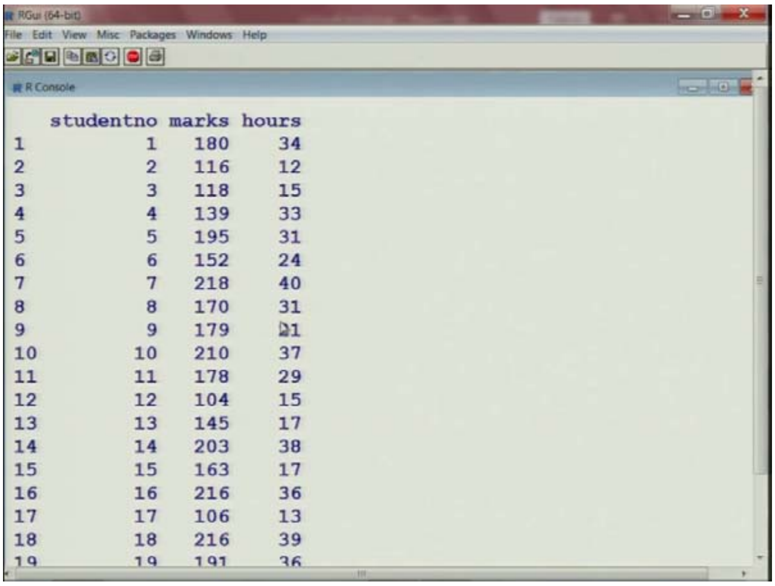
(Refer Slide Time: 19:06)



```
> library(boot)
> studentno=c(1:20)
>
>  marks=c(180,116,118,139,195,152,218,170,179, 210,178,104,$
>
>  hours=c(34,12,15,33,31,24,40,31,21,37,29,15, 17,38,17,36,$
>
>  student_marks_hours_data = data.frame(studentno, marks, h$
>  student_marks_hours_data 
```

So, now, I try to come to the R console, and I will try to take the same example and I will try to show you how the results are obtained directly on the R console. So, let us try to look into here. So, first let me load the Library boot. So, this package is already installed on my computer and then, I try to create the data set. So, you can see here this is the data set which we are going to use same thing what we have used earlier.

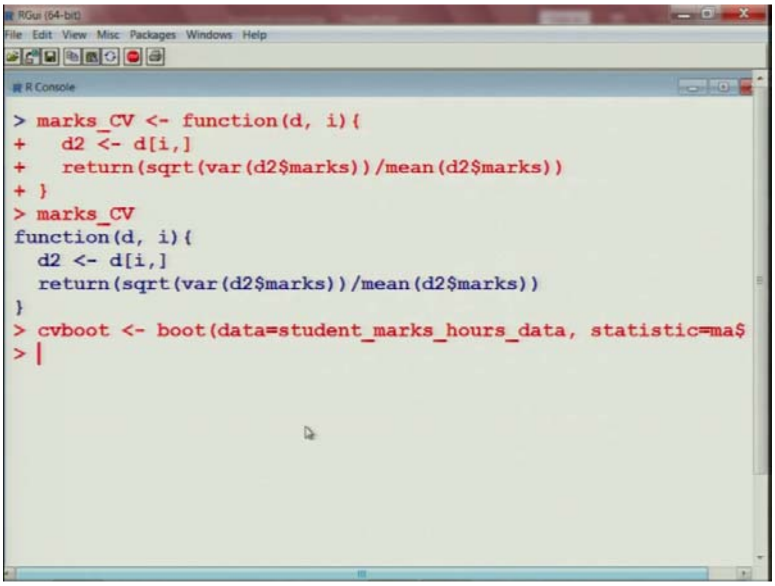
(Refer Slide Time: 19:24)



```
   studentno marks hours
1          1   180    34
2          2   116    12
3          3   118    15
4          4   139    33
5          5   195    31
6          6   152    24
7          7   218    40
8          8   170    31
9          9   179    21
10        10   210    37
11        11   178    29
12        12   104    15
13        13   145    17
14        14   203    38
15        15   163    17
16        16   216    36
17        17   106    13
18        18   216    39
19        19   191    36
```

You can see here is the same data set.

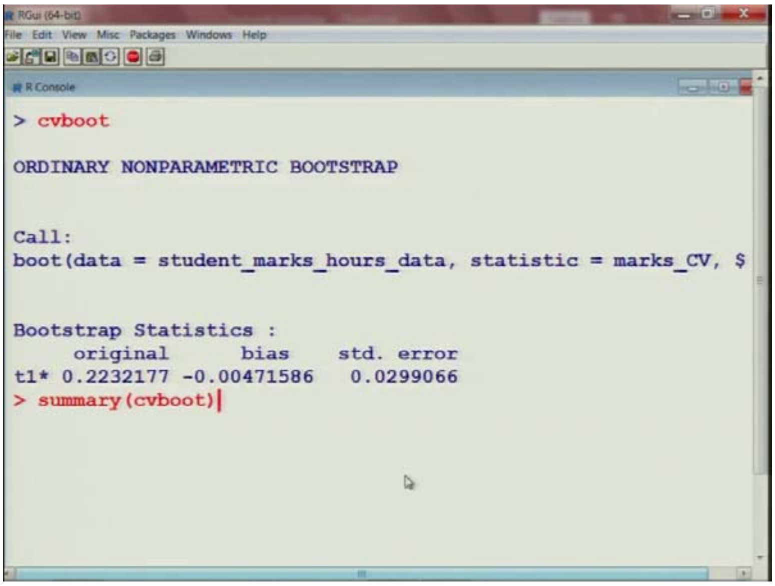
(Refer Slide Time: 19:40)



```
> marks_CV <- function(d, i){
+   d2 <- d[i,]
+   return(sqrt(var(d2$marks))/mean(d2$marks))
+ }
> marks_CV
function(d, i){
  d2 <- d[i,]
  return(sqrt(var(d2$marks))/mean(d2$marks))
}
> cvboot <- boot(data=student_marks_hours_data, statistic=ma$
> |
```

Now, I come to create my this program, I try to this function for computing the coefficient of variation, I try to copy it here. So, you can see here this is the program function which is going to compute the coefficient of variation for the given set of data.

(Refer Slide Time: 20:03)

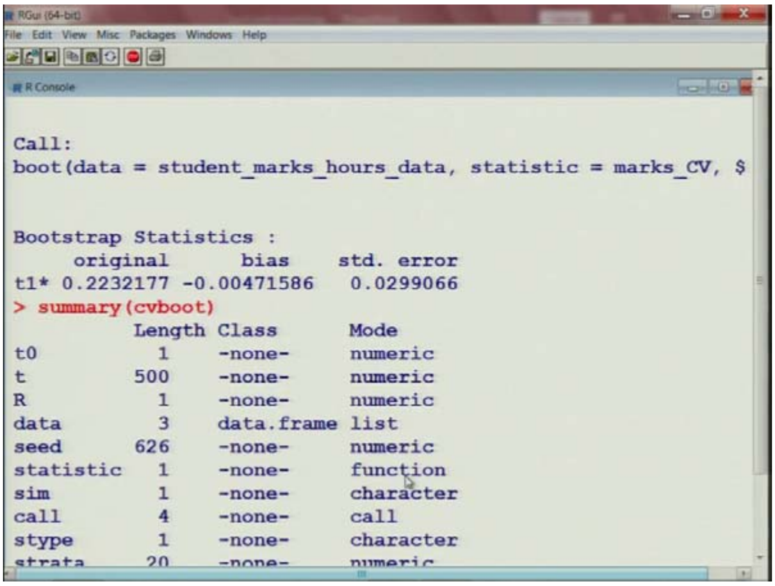


```
> cvboot

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = student_marks_hours_data, statistic = marks_CV, $


Bootstrap Statistics :
     original      bias    std. error
t1* 0.2232177 -0.00471586   0.0299066
> summary(cvboot)
```

And now, I try to use here the boot command. So, you can see here this is now here the outcome. So, I can clear the screen so that I can show you clearly. So, this is you can see this is the outcome and this is here the Bootstrap Statistics, the value of the estimator that means, the original value of this of the coefficient of variation based on the original sample.

This is here the bias which is the bootstrap bias of 500 values of coefficient of variation based on the 500 bootstrap samples and this is the bootstrap standard error which is again based on the 500 values of bootstrap samples, right. And now, if you want to find out here the mean of this thing so, if you want to see the summary of your cvboot, you can see here, it is here.

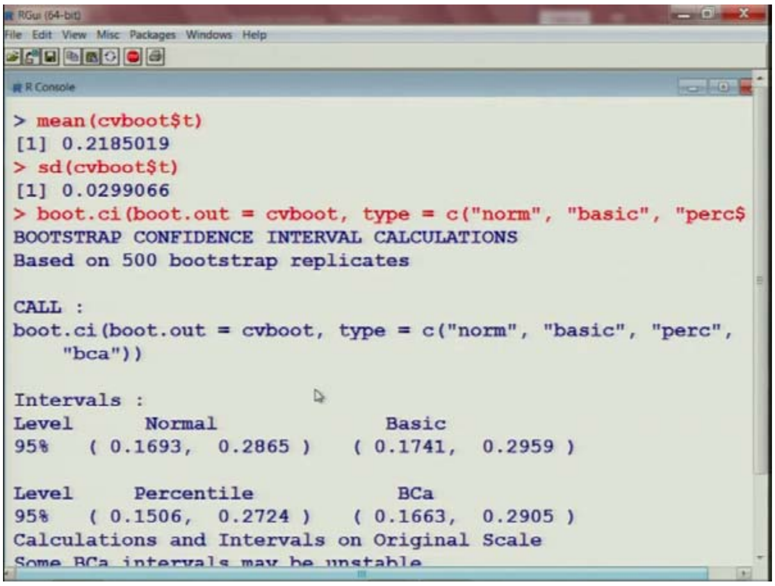
(Refer Slide Time: 20:58)



```
Call:
boot(data = student_marks_hours_data, statistic = marks_CV, $


Bootstrap Statistics :
     original      bias    std. error
t1* 0.2232177 -0.00471586   0.0299066
> summary(cvboot)
          Length Class      Mode
t0          1    -none-     numeric
t         500    -none-     numeric
R           1    -none-     numeric
data        3    data.frame list
seed      626    -none-     numeric
statistic   1    -none-     function
sim         1    -none-     character
call        4    -none-     call
stype       1    -none-     character
strata     20    -none-     numeric
```

Well, it is going to give you seed is equal to 626 and other information, right.

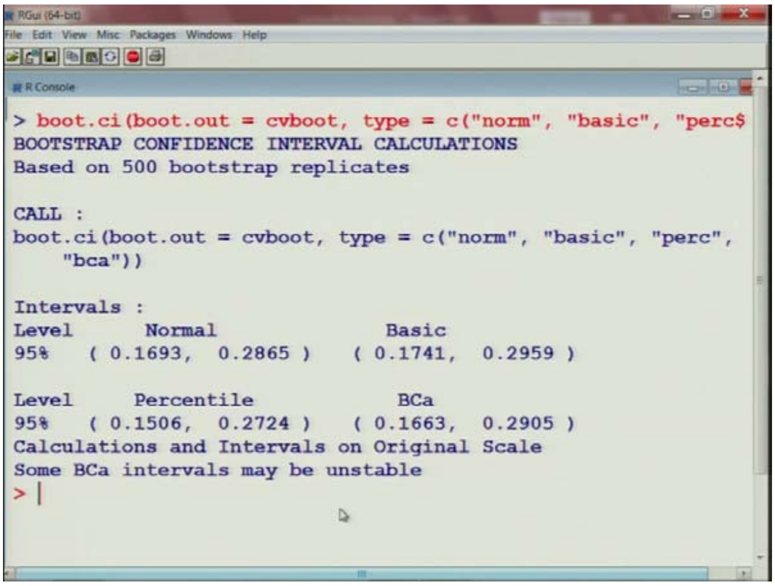
(Refer Slide Time: 21:07)



```
> mean(cvboot$t)
[1] 0.2185019
> sd(cvboot$t)
[1] 0.0299066
> boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc$
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc",
    "bca"))

Intervals :
Level      Normal              Basic
95%   ( 0.1693,  0.2865 )   ( 0.1741,  0.2959 )

Level     Percentile            BCa
95%   ( 0.1506,  0.2724 )   ( 0.1663,  0.2905 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
```

And if you want to find out the value of the bootstrap estimate of coefficient of variation, this is here like this and if you want to find out the standard deviation of the 500 bootstrap values of coefficient of variation, this is obtained here like this, right ok. Now, if you come and try to find out the bootstrap confidence interval estimate of intervals, then this comes out to be here like this. So, I can clear the screen so that you can see it clearly well.

(Refer Slide Time: 21:44)



```
> boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc$
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 500 bootstrap replicates

CALL :
boot.ci(boot.out = cvboot, type = c("norm", "basic", "perc",
    "bca"))

Intervals :
Level      Normal              Basic
95%   ( 0.1693,  0.2865 )   ( 0.1741,  0.2959 )

Level     Percentile            BCa
95%   ( 0.1506,  0.2724 )   ( 0.1663,  0.2905 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
> |
```

So, as you can see here, these are the four confidence intervals which we have obtained and similarly, if you want to compute the bootstrap t confidence interval also, you can just use the package bootstrap and exactly just use on the same line. So, this is how you can see here this is the similar outcome that you had obtained, right, and this is here the outcome on the R console, right ok.

So, now, we have understood about the bootstrap methodology also and now, if you try to connect, unless and until you have studied the sampling theory properly, you have under you have not if you have not understood the basic concept of simple random sampling, do you think that you can clearly understand the theory and philosophy of this bootstrap? Yes.

If you simply want to use it just like a compounder, I will say just try to learn the commands and you can use it, but if you have to write a new program or if you want to write a good program where you can be confident that yes, good values are estimated properly, then you need to learn the statistical concept.

And now, this bootstrap methodology is in your hand. Now, there should not be any question in your life as a data scientist that you cannot at least approximately estimate anything, you can find whatever you want. Although, I have taken here the examples of only estimation, confidence interval estimation, point estimation and standard error, but based on the bootstrap methodology, you can also conduct the test of hypothesis. Although, I have not considered here, but that is not difficult because in the test of hypothesis, you are simply trying to compute the value of the statistics for the given set of data and you try to compare it with some critical value based on either normal distribution, t distribution, chi square distribution or f distribution.

So, as you have say simulated the distribution function, you can again create the distribution of the required statistics with the bootstrap samples and it is not only that, you can conduct the test of hypothesis for any statistics. So, at all those places where as a mathematical statistician, the results are not available, the bootstrap methodology will help you and I am not restricting you to only test of hypothesis, but rather you can do many more things.

And if you want to learn those things, my very honest suggestion is that try to look into the books and I said my objective here in this course to take out the fear from your heart that statistics is difficult or data science is difficult well, data science has different components and you have to learn all of them, but a statistic plays a very important and major role.

So, after this statistics should not play a fear factor in your career and it should not be create be creating any hindrance to become a successful data scientist. So, you learn, you practice, and I will come with a new topic in the next lecture.

Till then, good luck, God bless you and take care, goodbye.