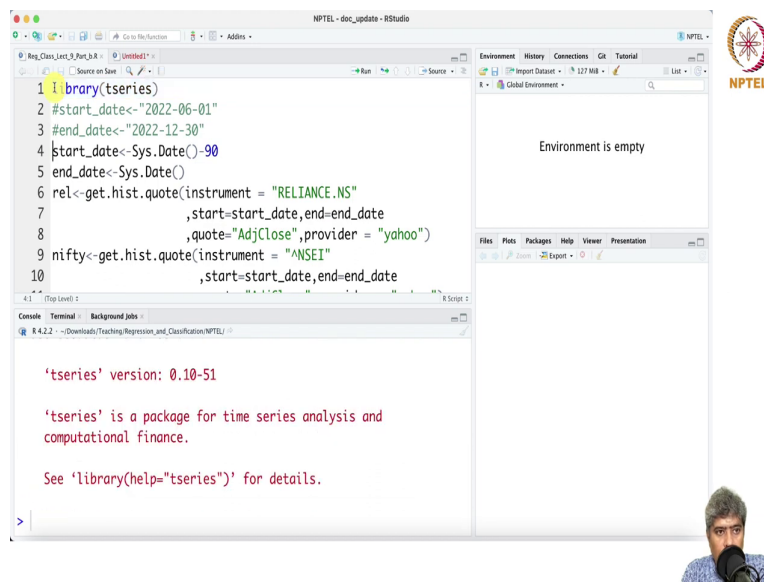


Predictive Analytics - Regression and Classification
Prof. Sourish Das
Department of Mathematics
Chennai Mathematical Institute

Lecture - 33
Hands on with R for Bootstrap Regression

Hello all welcome back to the last part or the final part of lecture 9. In this video, we will be doing some hands on, on how to do Bootstrap Regression for capital asset pricing model.

(Refer Slide Time: 00:31)



```
1 library(tseries)
2 #start_date<-"2022-06-01"
3 #end_date<-"2022-12-30"
4 start_date<-Sys.Date()-90
5 end_date<-Sys.Date()
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"
7                     ,start=start_date,end=end_date
8                     ,quote="AdjClose",provider = "yahoo")
9 nifty<-get.hist.quote(instrument = "ANSEI"
10                      ,start=start_date,end=end_date
```

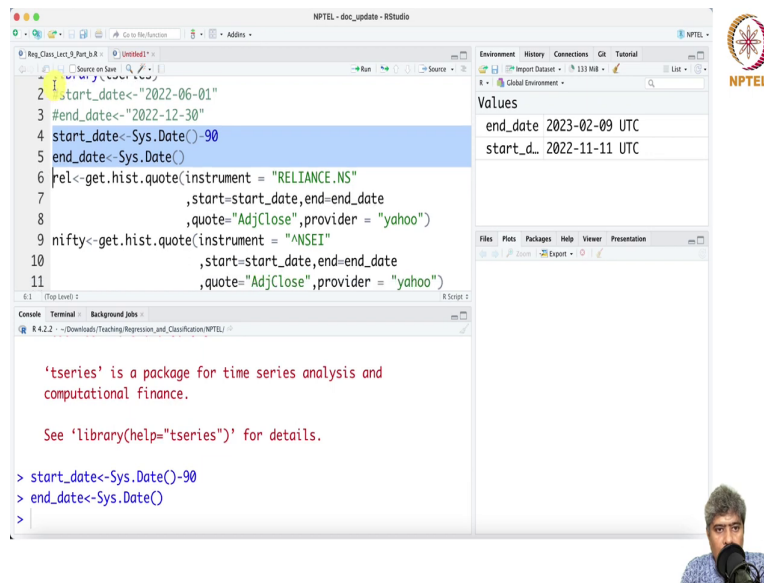
Environment is empty

'tseries' version: 0.10-51

'tseries' is a package for time series analysis and computational finance.

See 'library(help="tseries")' for details.

(Refer Slide Time: 00:48)



```
2 start_date<- "2022-06-01"
3 #end_date<- "2022-12-30"
4 start_date<- Sys.Date()-90
5 end_date<- Sys.Date()
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"
7                    ,start=start_date,end=end_date
8                    ,quote="AdjClose",provider = "yahoo")
9 nifty<-get.hist.quote(instrument = "ANSEI"
10                     ,start=start_date,end=end_date
11                     ,quote="AdjClose",provider = "yahoo")
```

Values

end_date	2023-02-09 UTC
start_d.	2022-11-11 UTC

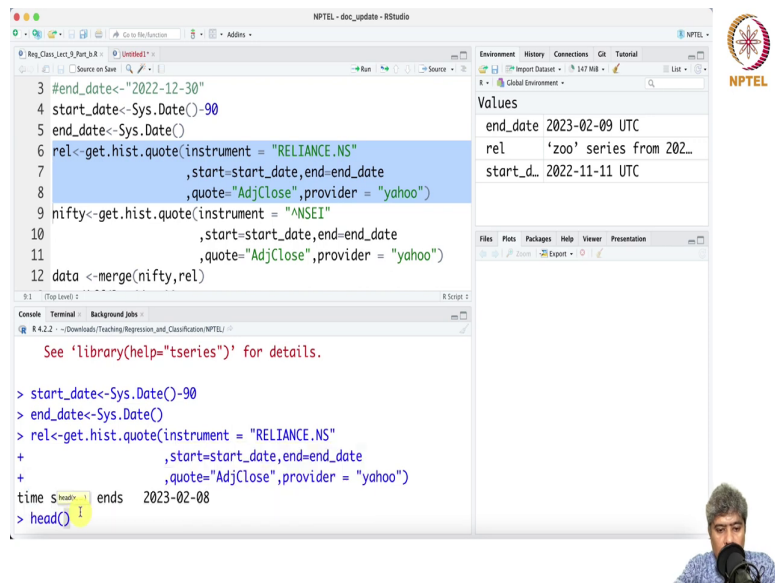
Console

```
6.1 (Top Level) >
R 4.2.2 -> [Downloads/Teaching/Regression_and_Classification/NPTEL] >
'tseries' is a package for time series analysis and
computational finance.

See 'library(help="tseries")' for details.
> start_date<-Sys.Date()-90
> end_date<-Sys.Date()
>
```

So, here in our this piece of code you have already seen, I will quickly remind you that you know we are going to use a T series package, then we are going to use system date or start date. Here we are then now this piece of code 9678 is going to download the data from internet from Yahoo Finance.

(Refer Slide Time: 01:01)



The screenshot displays the RStudio interface with the following R code in the editor:

```
3 #end_date<- "2022-12-30"  
4 start_date<-Sys.Date()-90  
5 end_date<-Sys.Date()  
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"  
7 ,start=start_date,end=end_date  
8 ,quote="AdjClose",provider = "yahoo")  
9 nifty<-get.hist.quote(instrument = "^NSEI"  
10 ,start=start_date,end=end_date  
11 ,quote="AdjClose",provider = "yahoo")  
12 data <-merge(nifty,rel)
```

The console output shows the execution of the code and the resulting data structure:

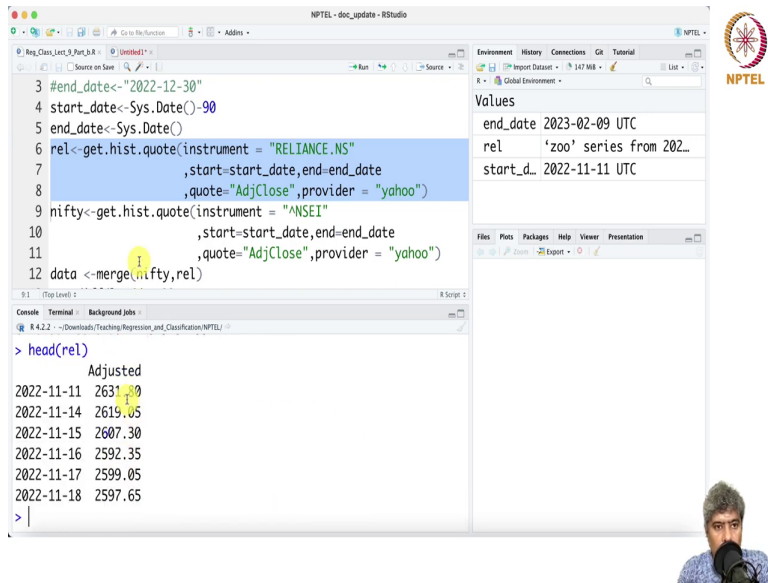
```
See 'library(help="tseries")' for details.  
> start_date<-Sys.Date()-90  
> end_date<-Sys.Date()  
> rel<-get.hist.quote(instrument = "RELIANCE.NS"  
+ ,start=start_date,end=end_date  
+ ,quote="AdjClose",provider = "yahoo")  
time series ends 2023-02-08  
> head()
```

The 'Values' pane on the right shows the following data:

end_date	rel	start_d.
2023-02-09 UTC	'zoo' series from 202...	2022-11-11 UTC

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 01:12)



The screenshot shows the RStudio interface with the following R code in the script editor:

```
3 #end_date<-"2022-12-30"  
4 start_date<-Sys.Date()-90  
5 end_date<-Sys.Date()  
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"  
7                       ,start=start_date,end=end_date  
8                       ,quote="AdjClose",provider = "yahoo")  
9 nifty<-get.hist.quote(instrument = "NSEI"  
10                      ,start=start_date,end=end_date  
11                      ,quote="AdjClose",provider = "yahoo")  
12 data <-merge(nifty,rel)
```

The console window shows the output of `head(rel)`:

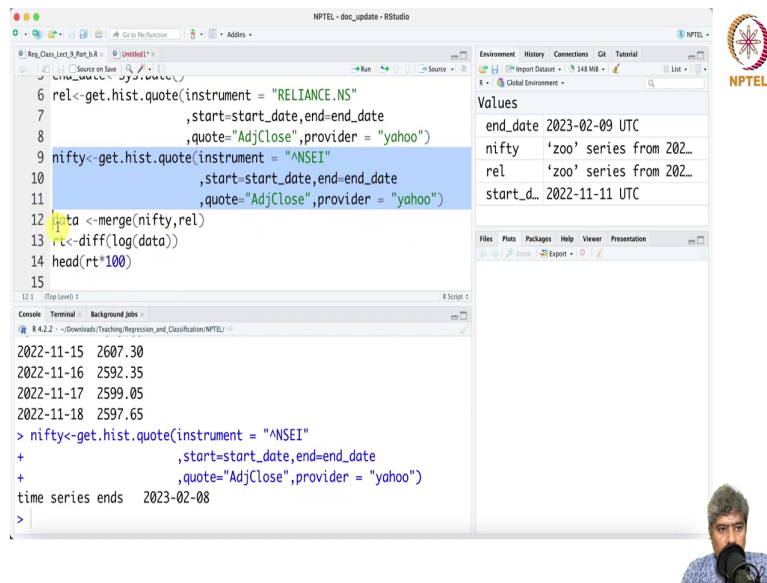
```
> head(rel)  
      Adjusted  
2022-11-11 2631.30  
2022-11-14 2619.05  
2022-11-15 2607.30  
2022-11-16 2592.35  
2022-11-17 2599.05  
2022-11-18 2597.65  
> |
```

The Values pane on the right shows the structure of the `rel` object:

```
Values  
end_date 2023-02-09 UTC  
rel      'zoo' series from 202_  
start_d... 2022-11-11 UTC
```

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 01:20)



```
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"
7                       ,start=start_date,end=end_date
8                       ,quote="AdjClose",provider = "yahoo")
9 nifty<-get.hist.quote(instrument = "NSEI"
10                      ,start=start_date,end=end_date
11                      ,quote="AdjClose",provider = "yahoo")
12 data <-merge(nifty,rel)
13 r<-diff(log(data))
14 head(rt*100)
15
```

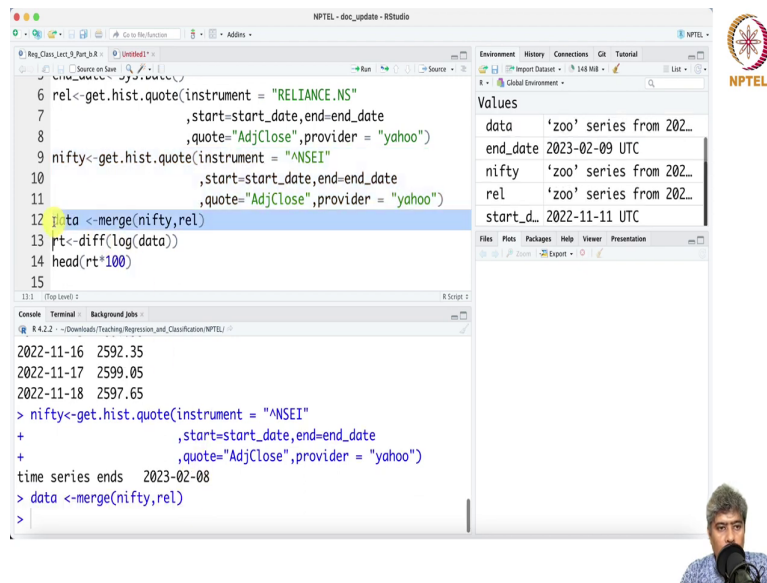
Values

end_date	2023-02-09 UTC
nifty	'zoo' series from 202...
rel	'zoo' series from 202...
start_d...	2022-11-11 UTC

```
2022-11-15 2607.30
2022-11-16 2592.35
2022-11-17 2599.05
2022-11-18 2597.65
> nifty<-get.hist.quote(instrument = "NSEI"
+                       ,start=start_date,end=end_date
+                       ,quote="AdjClose",provider = "yahoo")
time series ends 2023-02-08
>
```

So, here we have data is downloaded from Yahoo Finance head rel. So, here we can see then NIFTY 50 data will be downloaded adjusted close prices as of 8th February 2023 all the data has come.

(Refer Slide Time: 01:28)



The screenshot shows the RStudio interface with the following R code in the editor:

```
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"  
7                       ,start=start_date,end=end_date  
8                       ,quote="AdjClose",provider = "yahoo")  
9 nifty<-get.hist.quote(instrument = "NSEI"  
10                      ,start=start_date,end=end_date  
11                      ,quote="AdjClose",provider = "yahoo")  
12 data <-merge(nifty,rel)  
13 rt<-diff(log(data))  
14 head(rt*100)  
15
```

The console output shows the execution of the code:

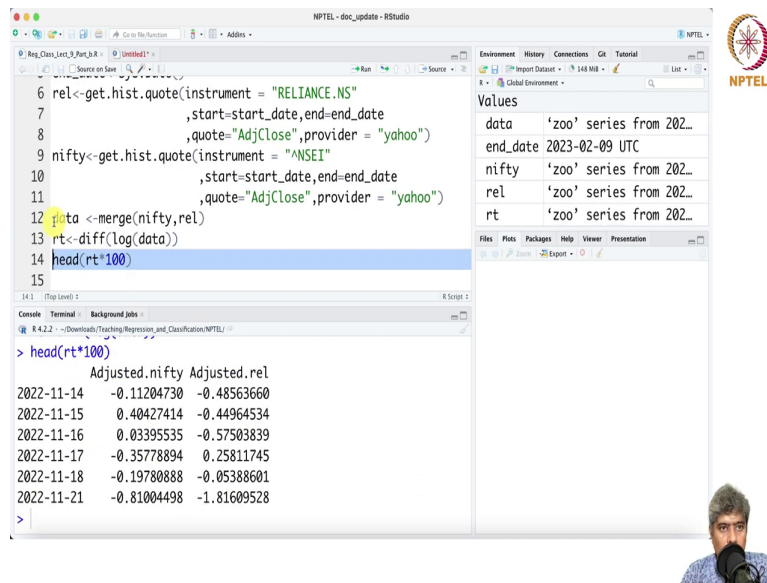
```
2022-11-16 2592.35  
2022-11-17 2599.05  
2022-11-18 2597.65  
> nifty<-get.hist.quote(instrument = "NSEI"  
+                       ,start=start_date,end=end_date  
+                       ,quote="AdjClose",provider = "yahoo")  
time series ends 2023-02-08  
> data <-merge(nifty,rel)  
>
```

The 'Values' pane on the right displays the structure of the merged data:

```
Values  
data 'zoo' series from 202...  
end_date 2023-02-09 UTC  
nifty 'zoo' series from 202...  
rel 'zoo' series from 202...  
start_d_ 2022-11-11 UTC
```

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 01:37)



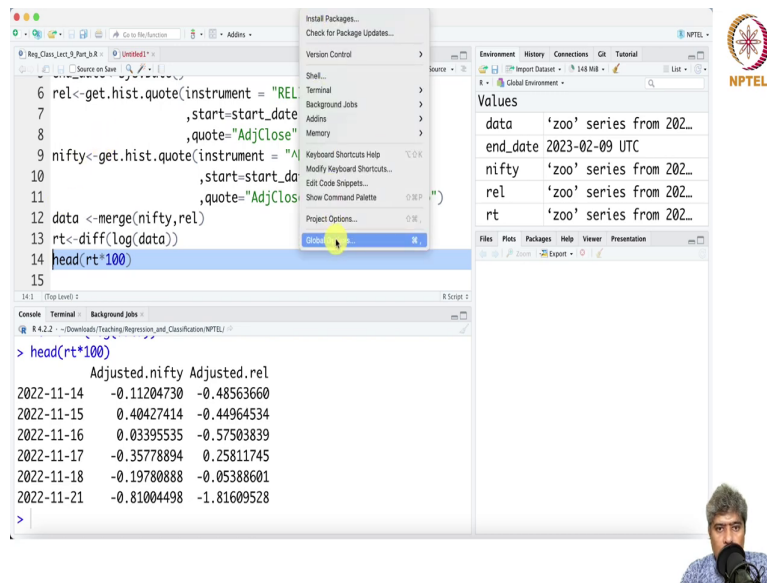
```
6 rel<-get.hist.quote(instrument = "RELIANCE.NS"
7                       ,start=start_date,end=end_date
8                       ,quote="AdjClose",provider = "yahoo")
9 nifty<-get.hist.quote(instrument = "ANSEI"
10                       ,start=start_date,end=end_date
11                       ,quote="AdjClose",provider = "yahoo")
12 data <-merge(nifty,rel)
13 rt<-diff(log(data))
14 head(rt*100)
15
```

Adjusted.nifty Adjusted.rel

Date	Adjusted.nifty	Adjusted.rel
2022-11-14	-0.11204730	-0.48563660
2022-11-15	0.40427414	-0.44964534
2022-11-16	0.03395535	-0.57503839
2022-11-17	-0.35778894	0.25811745
2022-11-18	-0.19780888	-0.05388601
2022-11-21	-0.81004498	-1.81609528

Then we are going to merge these data here, in line 13 we are going to calculate the log return. Here is the log returns in percentage term.

(Refer Slide Time: 01:14)



The screenshot displays the RStudio environment. The script editor on the left contains the following R code:

```
6 rel<-get.hist.quote(instrument = "REL",start=start_date,quote="AdjClose")
7
8
9 nifty<-get.hist.quote(instrument = "NIFTY",start=start_date,quote="AdjClose")
10
11
12 data <-merge(nifty,rel)
13 rt<-diff(log(data))
14 head(rt*100)
15
```

The console on the bottom left shows the execution of the code and the resulting output:

```
> head(rt*100)
      Adjusted.nifty Adjusted.rel
2022-11-14 -0.11204730 -0.48563660
2022-11-15  0.40427414 -0.44964534
2022-11-16  0.03395535 -0.57503839
2022-11-17 -0.35778894  0.25811745
2022-11-18 -0.19780888 -0.05388601
2022-11-21 -0.81004498 -1.81609528
```

The Environment pane on the right shows the objects created in the workspace:

Object	Value
data	'zoo' series from 202...
end_date	2023-02-09 UTC
nifty	'zoo' series from 202...
rel	'zoo' series from 202...
rt	'zoo' series from 202...

The NPTEL logo is visible in the top right corner of the RStudio window. A small video inset of a person speaking is located in the bottom right corner of the slide.

(Refer Slide Time: 01:46)

The screenshot displays the RStudio environment. The script editor contains the following R code:

```
6 rel<-get.hist.quote(instr
7 ,star
8 ,quot
9 nifty<-get.hist.quote(ins
10 ,st
11 ,qu
12 data <-merge(nifty,rel)
13 rt<-diff(log(data))
14 head(rt*100)
15
```

The console shows the output of the `head(rt*100)` command:

```
> head(rt*100)
      Adjusted.nifty Adj
2022-11-14 -0.11204730 -0.
2022-11-15  0.40427414 -0.
2022-11-16  0.03395535 -0.
2022-11-17 -0.35778894  0.
2022-11-18 -0.19780888 -0.
2022-11-21 -0.81004498 -1.81609528
```

An 'Options' dialog box is open, showing the 'General' tab. The 'R Sessions' section includes options for restoring projects and source documents at startup. The 'Workspace' section includes options for restoring RData and saving workspace on exit. The 'History' section includes options for always saving history and removing duplicate entries. The 'Other' section includes options for wrapping around navigation and receiving update notifications.

The NPTEL logo is visible in the top right corner of the RStudio window. A small video thumbnail of a person speaking is located in the bottom right corner of the slide.

(Refer Slide Time: 01:47)

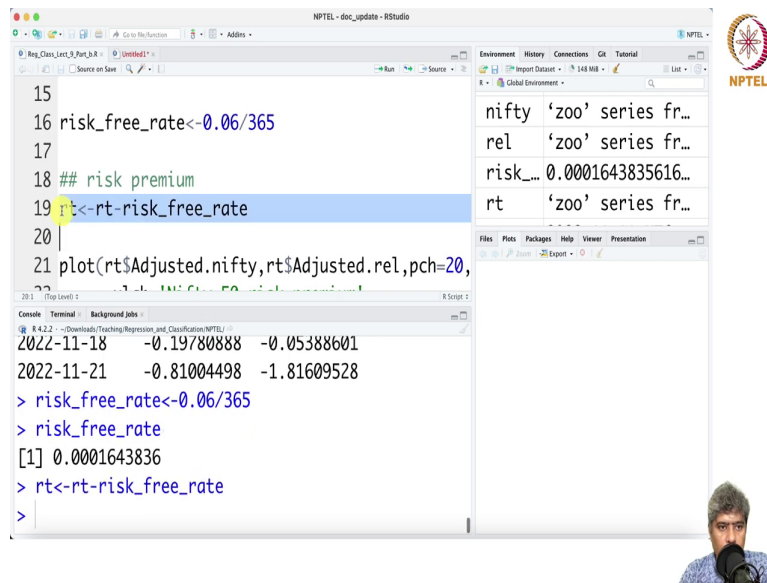
The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for data manipulation and plotting. The code includes: `rel <- get.hist.quote(instr, start, quote, nifty <- get.hist.quote(instr, start, quote, data <- merge(nifty, rel) rt <- diff(log(data)) head(rt*100)` and a function definition: `plot <- function(x) { if (is.function(x) && !is.null(attr("y"))) { if (missing(y)) y <- NULL # check for ylab hasylab <- function(name) { all(is.na(c(pmatch(name, "ylab")))) if (hasylab(...)) plot.function(x) else`
- Console:** Shows the output of `head(rt*100)` as a table of adjusted returns for dates from 2022-11-14 to 2022-11-21.
- Options Menu:** The 'Appearance' tab is selected, showing settings for 'Editor font size' (set to 18) and 'Editor theme' (set to Modern).
- Terminal:** Shows the R prompt `>` and the command `head(rt*100)`.
- Background Jobs:** A table showing background jobs for the `head(rt*100)` command.

	Adjusted.nifty	Adjusted
2022-11-14	-0.11204730	-0.11204730
2022-11-15	0.40427414	-0.11204730
2022-11-16	0.03395535	-0.11204730
2022-11-17	-0.35778894	-0.11204730
2022-11-18	-0.19780888	-0.11204730
2022-11-21	-0.81004498	-1.81609528

Let me increase the font size bit maybe I will use 24 apply ok; alright.

(Refer Slide Time: 01:58)



The screenshot shows the RStudio interface. The script editor contains the following R code:

```
15  
16 risk_free_rate<-0.06/365  
17  
18 ## risk premium  
19 rt<-rt-risk_free_rate  
20  
21 plot(rt$Adjusted.nifty,rt$Adjusted.rel,pch=20,
```

The console shows the following output:

```
2022-11-18 -0.19780888 -0.05388601  
2022-11-21 -0.81004498 -1.81609528  
> risk_free_rate<-0.06/365  
> risk_free_rate  
[1] 0.0001643836  
> rt<-rt-risk_free_rate  
>
```

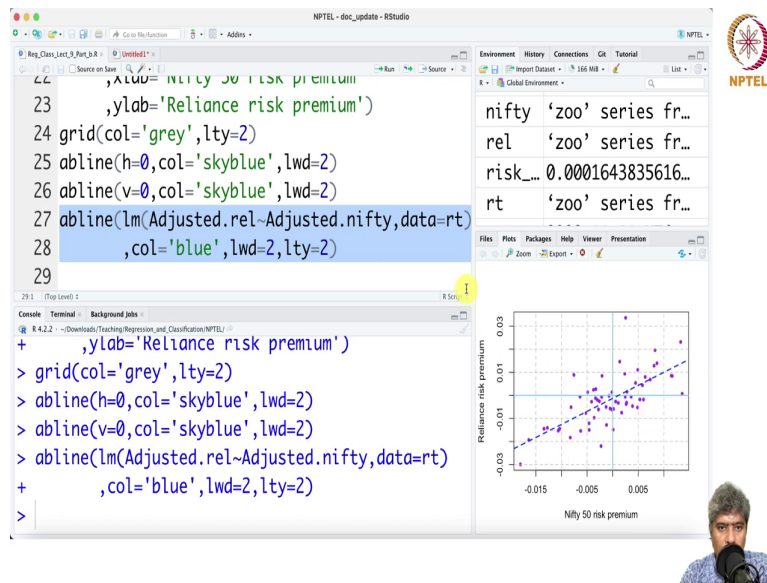
The Environment pane on the right shows the following objects:

```
nifty 'zoo' series fr...  
rel 'zoo' series fr...  
risk_... 0.0001643835616...  
rt 'zoo' series fr...
```

The NPTEL logo is visible in the top right corner of the RStudio window.

And now I am using risk free rate 6 percentage, annualized 6 percentages. So, I am dividing it to by 365, so this is the daily risk-free rate. Then I am calculating subtracting risk free rate from the return log return.

(Refer Slide Time: 02:24)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
23     ,ylab='Reliance risk premium')
24   grid(col='grey',lty=2)
25   abline(h=0,col='skyblue',lwd=2)
26   abline(v=0,col='skyblue',lwd=2)
27   abline(lm(Adjusted.rel~Adjusted.nifty,data=rt)
28     ,col='blue',lwd=2,lty=2)
29
```

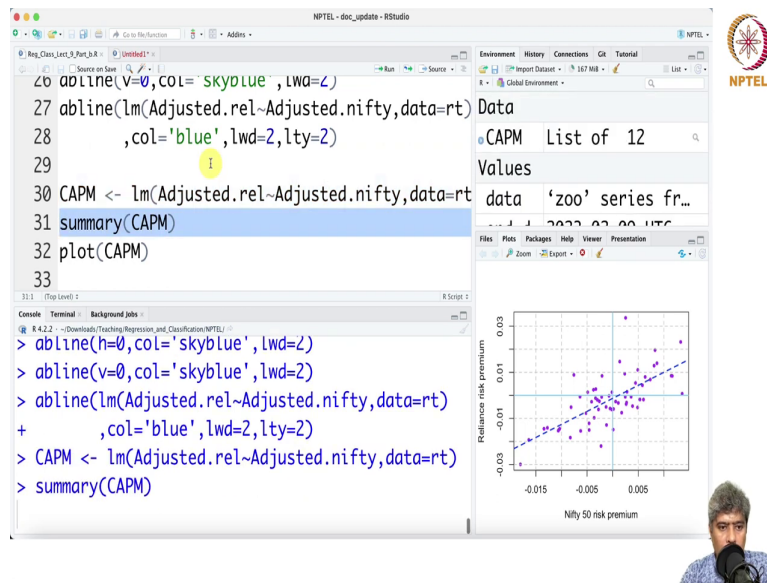
The console window shows the execution of these commands:

```
+     ,ylab='Reliance risk premium')
> grid(col='grey',lty=2)
> abline(h=0,col='skyblue',lwd=2)
> abline(v=0,col='skyblue',lwd=2)
> abline(lm(Adjusted.rel~Adjusted.nifty,data=rt)
+     ,col='blue',lwd=2,lty=2)
>
```

The Environment pane on the right lists variables: nifty 'zoo' series fr..., rel 'zoo' series fr..., risk_... 0.0001643835616..., and rt 'zoo' series fr... The bottom right pane shows a scatter plot of 'Reliance risk premium' (y-axis) versus 'Nifty 50 risk premium' (x-axis). The plot features a blue regression line, a grey grid, and two skyblue lines representing the x=0 and y=0 axes.

So, calculating risk premium and then drawing the plots of risk premium grids are being drawn. And then here are some ab lines and then. So, the fitted line this is the fitted line using OLS method.

(Refer Slide Time: 02:44)



The screenshot displays the RStudio interface. The script editor contains the following R code:

```
26 abline(v=0, col='skyblue', lwd=2)
27 abline(lm(Adjusted.rel~Adjusted.nifty,data=rt)
28       , col='blue', lwd=2, lty=2)
29
30 CAPM <- lm(Adjusted.rel~Adjusted.nifty,data=rt)
31 summary(CAPM)
32 plot(CAPM)
33
```

The console shows the execution of the following commands:

```
> abline(h=0, col='skyblue', lwd=2)
> abline(v=0, col='skyblue', lwd=2)
> abline(lm(Adjusted.rel~Adjusted.nifty,data=rt)
+       , col='blue', lwd=2, lty=2)
> CAPM <- lm(Adjusted.rel~Adjusted.nifty,data=rt)
> summary(CAPM)
```

The environment pane shows the 'Data' tab with 'CAPM List of 12' and 'Values' sections. The plot pane displays a scatter plot with 'Reliance risk premium' on the y-axis and 'Nifty 50 risk premium' on the x-axis. The plot includes a blue regression line and a vertical skyblue line at x=0. The NPTEL logo is visible in the top right corner, and a small video feed of the presenter is in the bottom right corner.

So, here I am fitting CAPM using OLS method and here is the summary statistics, so ok.

(Refer Slide Time: 02:52)



The screenshot displays an RStudio session with the following R code in the script editor:

```
26 abline(v=0, col='skyblue', lwd=2)
27 abline(lm(Adjusted.rel~Adjusted.nifty,data=rt)
28       , col='blue', lwd=2, lty=2)
29
30 CAPM <- lm(Adjusted.rel~Adjusted.nifty,data=rt)
31 summary(CAPM)
32 plot(CAPM)
33
```

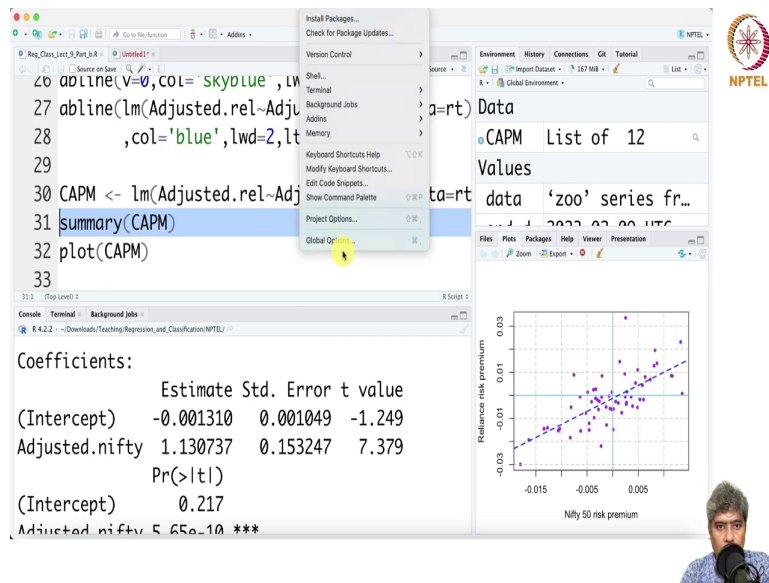
The console output shows the following results:

```
(Intercept) -0.001510  0.001049 -1.249
Adjusted.nifty  1.130737  0.153247  7.379
Pr(>|t|)
(Intercept)  0.217
Adjusted.nifty 5.65e-10 ***
---
Signif. codes:
```

The environment pane shows the 'CAPM' object as a list of 12 values. The plot pane displays a scatter plot with 'Reliance risk premium' on the y-axis and 'Nifty 50 risk premium' on the x-axis. The plot includes a blue regression line and a vertical skyblue line at x=0. The data points are purple dots showing a positive correlation.



(Refer Slide Time: 02:57)



The screenshot displays the RStudio environment. The script editor contains the following R code:

```
26 abline(v=0, col='skyblue', lw=2)
27 abline(lm(Adjusted.rel~Adj, data=rt))
28     , col='blue', lwd=2, lty=1)
29
30 CAPM <- lm(Adjusted.rel~Adj, data=rt)
31 summary(CAPM)
32 plot(CAPM)
33
```

The console shows the output of the `summary(CAPM)` command:

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.001310  0.001049  -1.249    0.217
Adjusted.nifty  1.130737  0.153247   7.379 5.65e-10 ***
```

The environment pane shows the loaded data: `data` (a 'zoo' series from 2002 Q3 to 2016 Q4) and `CAPM` (a list of 12 objects). The plot pane displays a scatter plot of 'Reliance risk premium' (y-axis, ranging from -0.03 to 0.03) versus 'Nifty 50 risk premium' (x-axis, ranging from -0.015 to 0.005). The plot includes a blue regression line and a vertical skyblue line at x=0. A small inset image of a person speaking is visible in the bottom right corner of the RStudio window.

(Refer Slide Time: 03:01)

The screenshot shows the RStudio interface with the following elements:

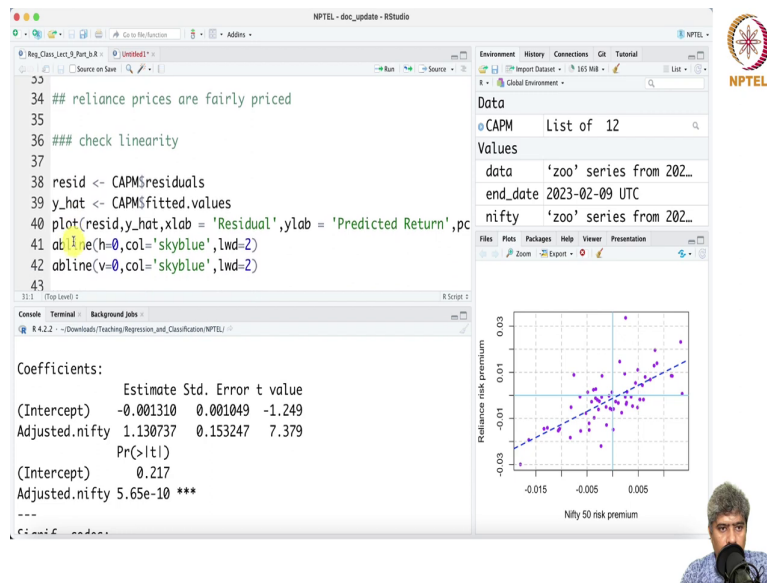
- Code Editor:** Contains R code for fitting a linear model and plotting it. The code includes:

```
26 abline(v=0, col='blue')
27 abline(lm(Adjusted...
28 , col='blue')
29
30 CAPM <- lm(Adjusted...
31 summary(CAPM)
32 plot(CAPM)
33
```
- Console:** Displays the output of the `summary(CAPM)` command:

```
Coefficients:
              Estima
(Intercept)  -0.0013
Adjusted.nifty 1.1307
Pr(>|t|)
(Intercept)   0.217
Adjusted.nifty 5.65e-10 ***
```
- Environment:** Shows a 'List of 12' objects, including a 'zoo' series.
- Plot:** A scatter plot titled 'Nifty 50 risk premium' showing a positive linear relationship between two variables. The x-axis ranges from -0.015 to 0.005, and the y-axis ranges from -0.015 to 0.005. A blue regression line is overlaid on the data points.
- Options Panel:** An 'Options' dialog box is open, showing settings for 'Editor font size' (set to 24) and 'Editor font' (set to Monaco).
- Background:** The NPTEL logo is visible in the top right corner, and a small video feed of a person is in the bottom right corner.

So, one problem is if we make the font size too big then it will not come as nicely as we would like to see it. So, now it like comes a bit nicely.

(Refer Slide Time: 03:10)



The screenshot shows an RStudio window with the following code in the script editor:

```
34 ## reliance prices are fairly priced
35
36 ### check linearity
37
38 resid <- CAPM$residuals
39 y_hat <- CAPM$fitted.values
40 plot(resid,y_hat,xlab = 'Residual',ylab = 'Predicted Return',pc
41 abline(h=0,col='skyblue',lwd=2)
42 abline(v=0,col='skyblue',lwd=2)
43
```

The console output shows the following coefficients:

```
Coefficients:
              Estimate Std. Error t value
(Intercept)  -0.001310  0.001049  -1.249
Adjusted.nifty 1.130737  0.153247  7.379
              Pr(>|t|)
(Intercept)    0.217
Adjusted.nifty 5.65e-10 ***
---
Signif. codes:  0. '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The plot shows a scatter plot of Reliance risk premium (y-axis) versus Nifty 50 risk premium (x-axis). The y-axis ranges from -0.03 to 0.03, and the x-axis ranges from -0.015 to 0.005. A blue regression line is shown, along with two vertical and horizontal skyblue lines at zero.

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 03:17)

The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for calculating residuals and plotting them. Line 38 is highlighted in yellow.
- Environment:** Lists variables: `rel` (zoo series), `resid` (Named num), `risk_fr...` (0.000164383561643836), `rt` (zoo series), `start_d...` (2022-11-11 UTC), and `y_hat` (Named num).
- Console:** Shows regression statistics: Residual standard error: 0.008232 on 60 degrees of freedom; Multiple R-squared: 0.4757, Adjusted R-squared: 0.467; F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10. It also shows the execution of the residual calculation and plotting commands.
- Plots:** A scatter plot titled 'Residual' vs 'Predicted Return' with purple points. The x-axis ranges from -0.02 to 0.03, and the y-axis ranges from -0.020 to 0.010.
- UI Elements:** NPTEL logo in the top right and a small video feed of a person in the bottom right corner.

And then we can take the residuals and then plot the residuals and then can do the Bartels rank test and then we did Breusch Pagan test, rank test and Breusch Pagan test everything was quite good.

(Refer Slide Time: 03:20)

The screenshot displays the RStudio interface with the following content:



```
40 plot(resid,y_hat,xlab = 'Residual',ylab = 'Predicted Return',pch=2)
41 abline(h=0,col='skyblue',lwd=2)
42 abline(v=0,col='skyblue',lwd=2)
43 |
44 ## Rank test for Randomness
45 library(randtests)
46 randtests::bartels.rank.test(resid)
47 ## Looks like, assumptions of randomness is okay!
48
49
50 ## Rauch-Donn Test for homoskedasticity
```

Environment: Global Environment
rel 'zoo' series from 202...
resid Named num [1:62] -0.0...
risk_fr_ 0.000164383561643836
rt 'zoo' series from 202...
start_d_ 2022-11-11 UTC
y_hat Named num [1:62] -0.0...

Files Plots Packages Help Viewers Presentation
Zoom Export

Console: Terminal Background Jobs
R 4.2.2 - Downloads/Teaching/Regression_and_Classification/NPTEL/
F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10
> resid <- CAPMS\$residuals
> y_hat <- CAPMS\$fitted.values
> plot(resid,y_hat,xlab = 'Residual',ylab = 'Predicted Return',pch=2
0,col='purple')
> abline(h=0,col='skyblue',lwd=2)
> abline(v=0,col='skyblue',lwd=2)
>

The plot shows a scatter of purple points around a horizontal skyblue line at y=0. The x-axis is labeled 'Residual' and ranges from -0.02 to 0.03. The y-axis is labeled 'Predicted Return' and ranges from -0.020 to 0.010.



(Refer Slide Time: 03:28)

The screenshot displays the RStudio environment. The script editor on the left contains the following R code:

```
46 randtests::bartels.rank.test(resid)
47 ## Looks like, assumptions of randomness is okay!
48
49
50 ## Breusch-Pagan Test for homoskedasticity
51
52 library(lmtest)
53 lmtest::bptest(CAPM)
54
55 ## Looks like, assumptions of homoscedasticity is okay!
```

The console on the bottom left shows the execution of the test:

```
> lmtest::bptest(CAPM)



studentized Breusch-Pagan test

data: CAPM
BP = 0.35063, df = 1, p-value = 0.5538
```

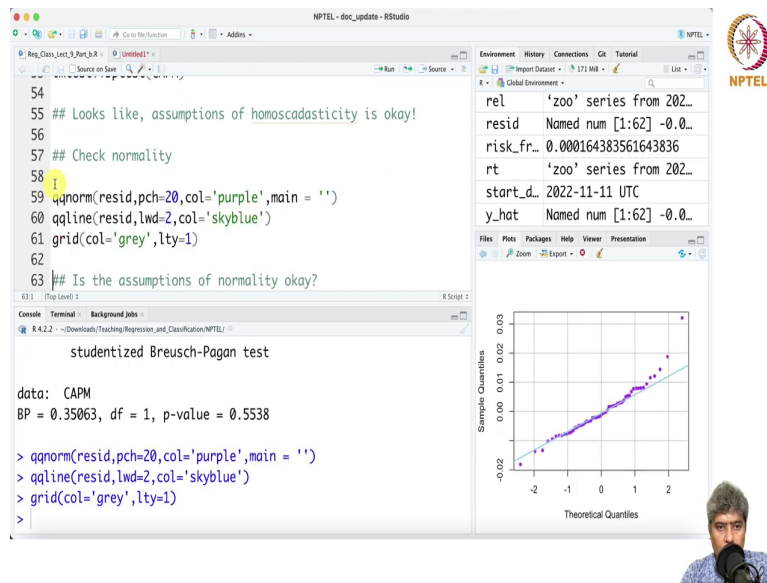
The environment pane on the right lists the following objects:

- rel 'zoo' series from 202...
- resid Named num [1:62] -0.0...
- risk_fr... 0.000164383561643836
- rt 'zoo' series from 202...
- start_d... 2022-11-11 UTC
- y_hat Named num [1:62] -0.0...

A residual plot is shown in the bottom right corner, with 'Predicted Return' on the y-axis and 'Residual' on the x-axis. The plot shows a scatter of points around a horizontal line at zero, indicating that the residuals are randomly distributed.



(Refer Slide Time: 03:34)



The screenshot displays the RStudio interface. The editor window contains R code for a normality test:

```
54  
55 ## Looks like, assumptions of homoscedasticity is okay!  
56  
57 ## Check normality  
58  
59 qqnorm(resid,pch=20,col='purple',main = '')  
60 qqline(resid,lwd=2,col='skyblue')  
61 grid(col='grey',lty=1)  
62  
63 ## Is the assumptions of normality okay?
```

The console window shows the output of the test:

```
studentized Breusch-Pagan test  
  
data: CAPM  
BP = 0.35063, df = 1, p-value = 0.5538  
  
> qqnorm(resid,pch=20,col='purple',main = '')  
> qqline(resid,lwd=2,col='skyblue')  
> grid(col='grey',lty=1)  
>
```

The Environment pane on the right shows the following objects:

rel	'zoo' series from 202...
resid	Named num [1:62] -0.0...
risk_fr...	0.000164383561643836
rt	'zoo' series from 202...
start_d...	2022-11-11 UTC
y_hat	Named num [1:62] -0.0...

The Plots pane shows a Q-Q plot with 'Sample Quantiles' on the y-axis and 'Theoretical Quantiles' on the x-axis. The data points are purple dots, and a blue line represents the theoretical normal distribution. The points generally follow the line, indicating approximate normality.

The NPTEL logo is visible in the top right corner of the RStudio window.

Then when we did a test for normality it was kind of get rejected and that is where the whole problem starts.

(Refer Slide Time: 03:44)

The screenshot shows an RStudio window with the following components:

- Script Editor:** Contains R code for a bootstrap regression analysis:

```
69  
70 ## Bootstrap regression  
71  
72 ## paired sampling method  
73 ## rt = a + b rm + e; e ~ F(), E(e)=0, Var(e)=s^2  
74 set.seed(6587)  
75 rt1<-data.frame(rt)  
76 n <- nrow(rt1)  
77 B<-1000 ## Bootstrap simulation size  
78 beta_star<-matrix(NA,nrow=B,ncol = 2)
```
- Environment:** Lists variables: rel ('zoo' series from 202...), resid (Named num [1:62] -0.0...), risk_fr... (0.000164383561643836), rt ('zoo' series from 202...), start_d... (2022-11-11 UTC), y_hat (Named num [1:62] -0.0...).
- Console:** Shows the execution of `stats::ks.test(resid, 'pnorm')` and its output:

```
Exact one-sample Kolmogorov-Smirnov test  
  
data: resid  
D = 0.49277, p-value = 2.054e-14  
alternative hypothesis: two-sided
```
- Plots:** A Q-Q plot with 'Sample Quantiles' on the y-axis (ranging from -0.02 to 0.03) and 'Theoretical Quantiles' on the x-axis (ranging from -2 to 2). The data points closely follow a diagonal reference line, indicating a normal distribution.

So, we decided to do a bootstrap regression first we are here we are going to apply paired resampling method or paired bootstrap regression method. So, here we are assuming risk premium equal to $a + b \text{ market premium} + \text{error}$. And we are assuming this idiosyncratic return or E is following some distribution. We do not know what is that distribution?

Some distribution F with expected value of E to be 0 and variance of E to be σ^2 , but we do not know what is that distribution is this could be any distribution.

(Refer Slide Time: 04:27)

The screenshot displays the RStudio interface with the following content:

```
73 ## paired resampling method
74 ## rt = a + b * rm + e; e ~ F(), E(e)=0, Var(e)=s^2
74 set.seed(6587)
75 r1<-data.frame(rt)
76 n1<- nrow(r1)
77 B<-1000 # Bootstrap simulation size
78 beta_star<-matrix(NA,nrow=B,ncol = 2)
79 colnames(beta_star)<-c('alpha','beta')
80 R.squared_star.pair<-rep(NA,B)
81 for(b in 1:B){
82   id_star<-sample(1:n1,n,replace = TRUE)
83   beta_star[b,]<-c(mapply(function(x,y){
84     lm_resid<-resid(lm(r1[id_star,]~r1[id_star,]
85     beta_star[b,]<-c(mapply(function(x,y){
86     R.squared_star.pair[b]<-cor(beta_star[b,],R.squared_star.pair[b])
87   }
88 }
89 }
90 }
91 }
```

Environment: Global Environment

rt1: 62 obs. of 2 variables

Values:

- data: 'zoo' series from 202...
- end_date: 2023-02-09 UTC
- nifty: 'zoo' series from 202...
- rel: 'zoo' series from 202...

Files: Plots: Packages: Help: View: Presentation

Console: Terminal | Background Jobs



R 4.2.2 - (Downloads)Teaching/Regression_and_Classification/NPTEL/

Exact one-sample Kolmogorov-Smirnov test

data: resid
D = 0.49277, p-value = 2.054e-14
alternative hypothesis: two-sided

```
> set.seed(6587)
> r1<-data.frame(rt)
>
```

Sample Quantiles vs Theoretical Quantiles plot showing a linear relationship between the two axes, indicating a normal distribution fit.



(Refer Slide Time: 04:34)

The screenshot shows an RStudio window titled "NPTEL - doc_update - RStudio". The script editor contains the following R code:

```
74 set.seed(6587)
75 rt1<-data.frame(rt)
76 n <- nrow(rt1)
77 B<-1000 ## Bootstrap simulation size
78 beta_star<-matrix(NA,nrow=B,ncol = 2)
79 colnames(beta_star)<-c('alpha','beta')
80 R.squared_star.pair<-rep(NA,B)
81 for(b in 1:B){
82   id_star<-sample(1:n,n,replace = TRUE)
83   rt_star<-rt1[id_star,]
```

The console shows the execution of the code:

```
> n <- nrow(rt1)
> n
[1] 62
> beta_star<-matrix(NA,nrow=B,ncol = 2)
Error in matrix(NA, nrow = B, ncol = 2) : object 'B' not found
> B<-1000 ## Bootstrap simulation size
> beta_star<-matrix(NA,nrow=B,ncol = 2)
> colnames(beta_star)<-c('alpha','beta')
>
```

The Environment pane on the right shows the following data objects:

- beta_star_logi [1:1000, 1:2] N...
- CAPM list of 12
- rt1 62 obs. of 2 variabl...

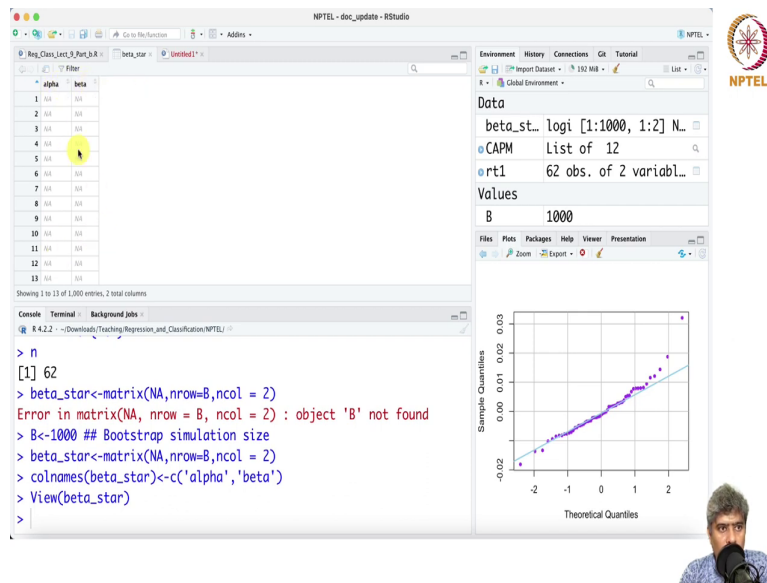
The Values pane shows B = 1000.

A Q-Q plot is displayed in the bottom right, with "Sample Quantiles" on the y-axis (ranging from -0.02 to 0.03) and "Theoretical Quantiles" on the x-axis (ranging from -2 to 2). The plot shows a series of purple dots following a blue diagonal line, indicating a normal distribution fit.

The NPTEL logo is visible in the top right corner of the RStudio window.

So, we put it as a data frame here is the number of samples is 62, we have taken bootstrap simulation size to be 1000, you can increase it to any number 10,000 or anything. And then I am defining a beta star a matrix of sample simulation size n and number of columns is beta 2, because I am going to estimate alpha and beta. So, oh sorry I have to run this ok.


(Refer Slide Time: 05:03)




The screenshot displays the RStudio interface. The top-left pane shows a data frame with columns 'alpha' and 'beta', containing 13 rows of NA values. The bottom-left pane shows the R console with the following code and output:

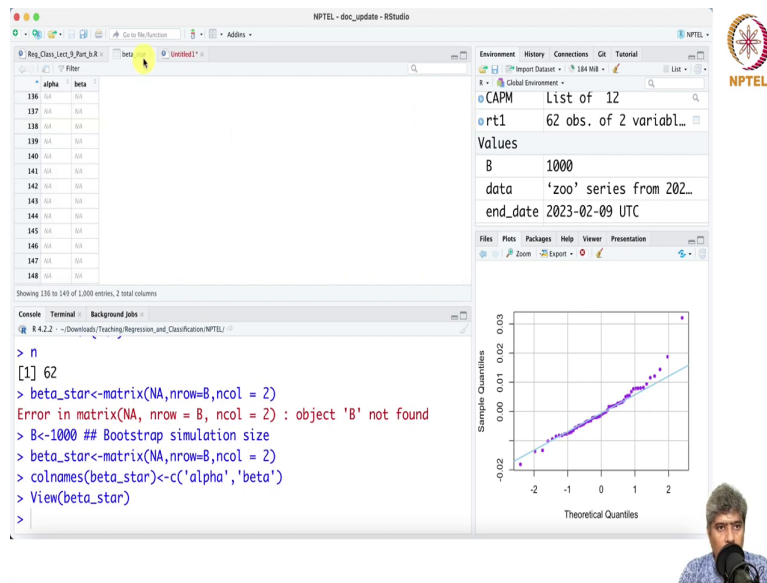
```
> n  
[1] 62  
> beta_star<-matrix(NA,nrow=B,ncol = 2)  
Error in matrix(NA, nrow = B, ncol = 2) : object 'B' not found  
> B<-1000 ## Bootstrap simulation size  
> beta_star<-matrix(NA,nrow=B,ncol = 2)  
> colnames(beta_star)<-c('alpha','beta')  
> View(beta_star)  
>
```

The right-hand pane shows the Environment window with variables 'beta_star_logi [1:1000, 1:2] N...', 'CAPM List of 12', and 'rt1 62 obs. of 2 variabl...'. Below this is a plot of Sample Quantiles versus Theoretical Quantiles, showing a linear relationship between the two axes, indicating a normal distribution fit.





(Refer Slide Time: 05:15)



The screenshot displays the RStudio interface. The top-left pane shows a data table with columns 'alpha' and 'beta' and rows 136 through 148. The top-right pane shows the Environment tab with a variable 'beta' of type 'double' and size 1. The bottom-left pane shows the Console with the following R code and output:

```
> n  
[1] 62  
> beta_star<-matrix(NA,nrow=B,ncol = 2)  
Error in matrix(NA, nrow = B, ncol = 2) : object 'B' not found  
> B<-1000 ## Bootstrap simulation size  
> beta_star<-matrix(NA,nrow=B,ncol = 2)  
> colnames(beta_star)<-c('alpha','beta')  
> View(beta_star)  
>
```

The bottom-right pane shows a Q-Q plot with 'Sample Quantiles' on the y-axis and 'Theoretical Quantiles' on the x-axis. The plot shows a positive linear relationship between the two axes, with a blue regression line and purple data points. A small video inset of a person speaking is visible in the bottom right corner of the RStudio window.

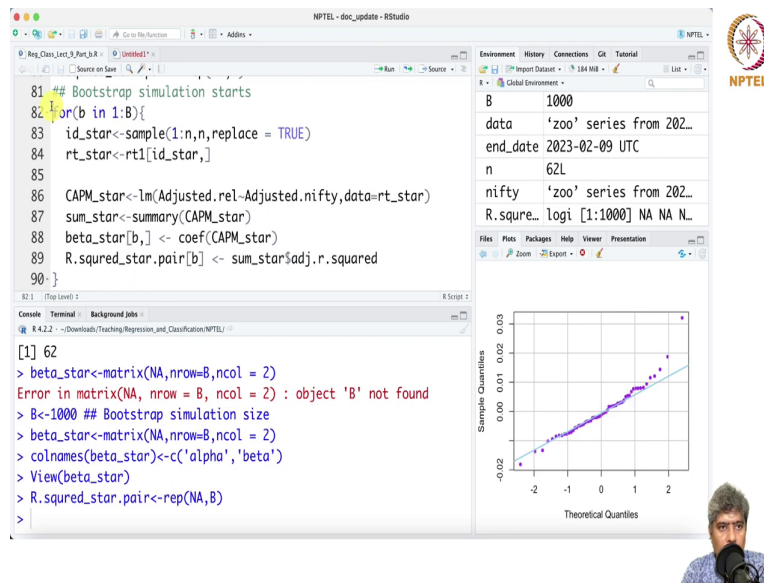
Now, if you see I have created this matrix with column name alpha beta and it has size of it has basically if you just go up 10th about 1000 rows number of simulation.

(Refer Slide Time: 05:22)

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for data preparation and simulation. Line 80 is highlighted: `R.squared_star.pair<-rep(NA,B)`.
- Environment:** Lists objects: `B` (1000), `data` ('zoo' series from 202...), `end_date` (2023-02-09 UTC), `n` (62L), `nifty` ('zoo' series from 202...), and `R.squared_star.pair` (logi [1:1000] NA NA N...).
- Console:** Shows the execution of the code and an error message: `Error in matrix(NA, nrow = B, ncol = 2) : object 'B' not found`. The error occurs because the variable `B` is not yet defined in the environment at the time of execution.
- Plots:** A Q-Q plot titled "Sample Quantiles" vs "Theoretical Quantiles" is shown, with data points following a linear trend.
- NPTEL Logo:** Located in the top right corner.
- Speaker:** A small video feed of a person speaking is visible in the bottom right corner.

(Refer Slide Time: 05:33)



The screenshot shows the RStudio interface. The script editor contains the following R code:

```
81 ## Bootstrap simulation starts
82 for(b in 1:B){
83   id_star<-sample(1:n,n,replace = TRUE)
84   rt_star<-rt1[id_star,]
85
86   CAPM_star<-lm(Adjusted.rel-Adjusted.nifty,data=rt_star)
87   sum_star<-summary(CAPM_star)
88   beta_star[b,] <- coef(CAPM_star)
89   R.squared_star.pair[b] <- sum_star$adj.r.squared
90 }
```

The console shows the execution of the code, with an error message:

```
[1] 62
> beta_star<-matrix(NA,nrow=B,ncol = 2)
Error in matrix(NA, nrow = B, ncol = 2) : object 'B' not found
> B<-1000 ## Bootstrap simulation size
> beta_star<-matrix(NA,nrow=B,ncol = 2)
> colnames(beta_star)<-c('alpha','beta')
> View(beta_star)
> R.squared_star.pair<-rep(NA,B)
>
```

The Environment pane shows the following objects:

B	1000
data	'zoo' series from 202...
end_date	2023-02-09 UTC
n	62L
nifty	'zoo' series from 202...
R.squared...	logi [1:1000] NA NA N...

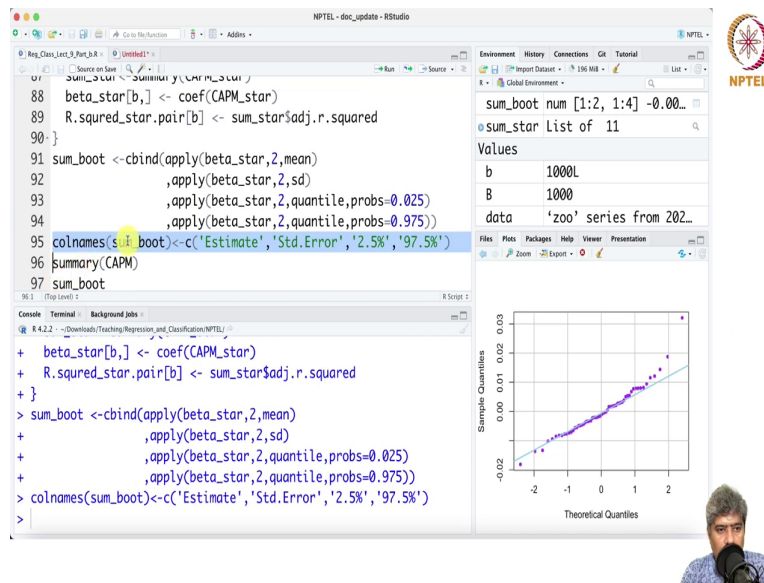
The Plots pane shows a Q-Q plot with 'Sample Quantiles' on the y-axis and 'Theoretical Quantiles' on the x-axis. The data points are purple dots, and a blue line represents the theoretical distribution. The points generally follow the line, indicating a normal distribution, but there is a slight deviation at the upper end.

The NPTEL logo is visible in the top right corner of the RStudio window.

Now, I am going to also create another place for R square I am calling it R square star. Then here my loop start bootstrap regression bootstrap simulation starts bootstrap simulation starts. So, first from 1 is to n id I am going to draw ransom's id I am going to draw the random ids and those ids I am going to take the samples random samples from the samples itself.

So, rt 1 is the sample itself from there I am going to draw random samples and those are that is the rt star and from the rt star I am going to you know fit the model and from the model I am going to calculate the summary and from the summary I am going to I am extracting the coefficients alpha beta ok and the adjusted R squared I am keeping it here.

(Refer Slide Time: 06:40)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
88 beta_star[b,] <- coef(CAPM_star)
89 R.squared_star.pair[b] <- sum_star$adj.r.squared
90 }
91 sum_boot <- cbind(apply(beta_star,2,mean)
92                   ,apply(beta_star,2,sd)
93                   ,apply(beta_star,2,quantile,probs=0.025)
94                   ,apply(beta_star,2,quantile,probs=0.975))
95 colnames(sum_boot) <- c('Estimate', 'Std.Error', '2.5%', '97.5%')
96 summary(CAPM)
97 sum_boot
```

The console window shows the execution of this code:

```
+ beta_star[b,] <- coef(CAPM_star)
+ R.squared_star.pair[b] <- sum_star$adj.r.squared
+ }
> sum_boot <- cbind(apply(beta_star,2,mean)
+                   ,apply(beta_star,2,sd)
+                   ,apply(beta_star,2,quantile,probs=0.025)
+                   ,apply(beta_star,2,quantile,probs=0.975))
> colnames(sum_boot) <- c('Estimate', 'Std.Error', '2.5%', '97.5%')
>
```

The Environment pane on the right shows the following objects:

- sum_boot num [1:2, 1:4] -0.00...
- sum_star List of 11
- Values: b (1000L), B (1000), data ('zoo' series from 202...)

A Q-Q plot is visible in the bottom right corner, with 'Sample Quantiles' on the y-axis and 'Theoretical Quantiles' on the x-axis. The plot shows a strong linear relationship between the two axes, indicating that the data is approximately normally distributed. A small inset image of a person speaking is located in the bottom right corner of the RStudio window.

So, let me just run this piece of code and it is run in almost no time and then I am going to. So, summarize the bootstrap statistics here so few and then estimate standard error and 25 percent.

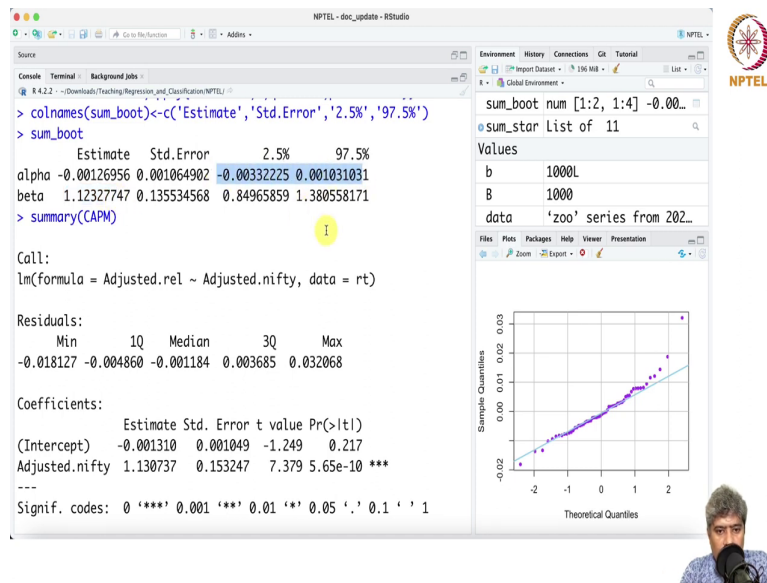
(Refer Slide Time: 06:57)

The screenshot displays the RStudio interface with the following components:

- Source Editor:** R code for bootstrapping CAPM parameters. Lines 91-99 show the calculation of `sum_boot` using `apply` and `cbind`, followed by a `summary` call and a histogram for the alpha parameter.
- Environment:** Shows the `sum_star` list with values for `b` (1000L) and `B` (1000).
- Console:** Displays regression statistics: Adjusted R-squared: 1.130737, Multiple R-squared: 0.4757, Adjusted R-squared: 0.467, and F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10.
- Plots:** A Q-Q plot of Sample Quantiles vs Theoretical Quantiles, showing a linear relationship with a blue regression line.

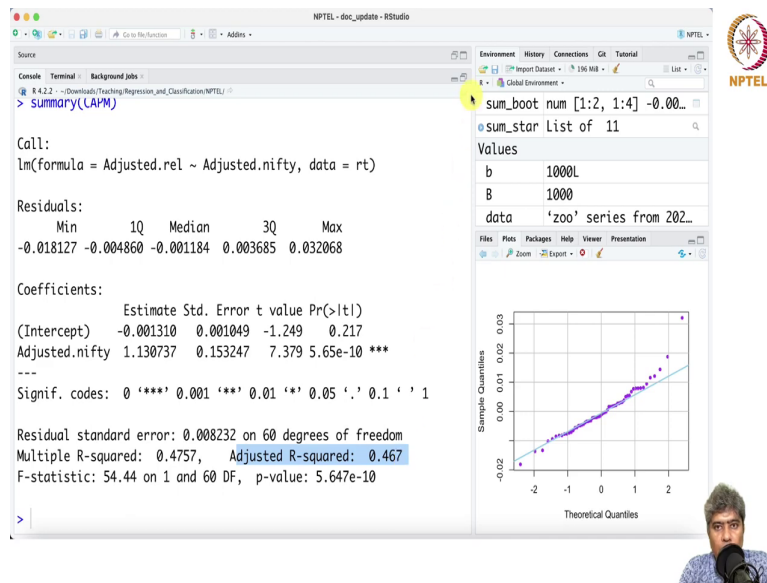
So, now if you run this, so estimates are like negative 0.0012 and 0.12. So, now you can see the alpha is containing the 0 the confidence interval this is the confidence interval for alpha and it contain 0. And similarly, this is the confidence interval for beta it contains 1. So, for this is for reliance and remember that summary statistics for CAPM this is the OLS thing.

(Refer Slide Time: 07:28)



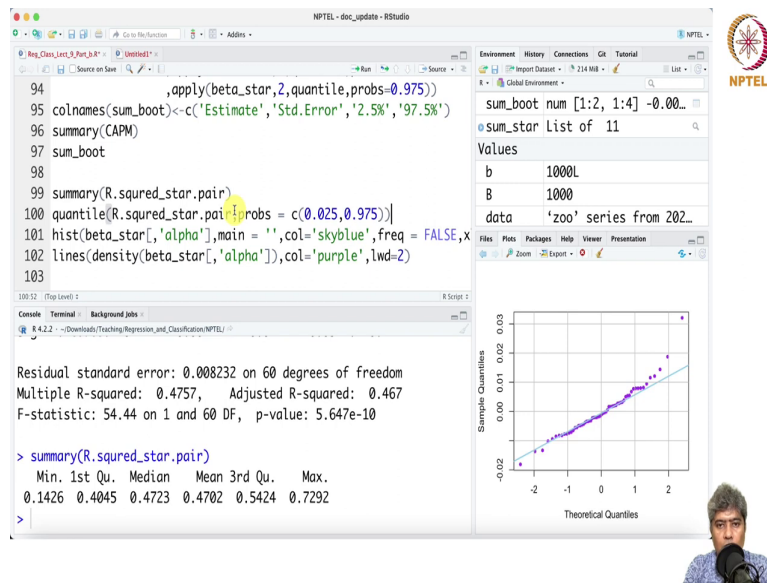
So, if you now carefully look into it that OLS estimate is 0.0013 and if you round it of to 4 decimal places it is actually negative 0.0013, beta is 1.1307 1 this is beta here 1.12327 the this is the coefficient.

(Refer Slide Time: 07:56)



So, now since the underlying distribution does not or underlying model assumption does not hold good, so these inference based on t value and p value is not valid inference. But here I am assuming that the underlying distribution is just unknown and I am doing a non-parametric test. So, these inference is valid though there is not much change in my inference, but this inference is valid.

(Refer Slide Time: 08:41)



The screenshot displays an RStudio window with the following components:

- Script Editor:** Contains R code for calculating bootstrap quantiles for the adjusted R-squared statistic. The code includes `apply`, `colnames`, `summary`, `summary`, `summary`, `quantile`, `hist`, and `lines` functions.
- Environment:** Shows variables like `sum_boot` and `sum_star`.
- Console:** Displays regression statistics: Residual standard error: 0.008232 on 60 degrees of freedom, Multiple R-squared: 0.4757, Adjusted R-squared: 0.467, and F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10. It also shows the output of `summary(R.squared_star.pair)`.
- Plots:** A Q-Q plot titled "Sample Quantiles" vs "Theoretical Quantiles" showing a linear relationship between the two, indicating approximate normality.

Console Output:

```
Residual standard error: 0.008232 on 60 degrees of freedom
Multiple R-squared: 0.4757, Adjusted R-squared: 0.467
F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10

> summary(R.squared_star.pair)
  Min. 1st Qu. Median Mean 3rd Qu. Max.
0.1426 0.4045 0.4723 0.4702 0.5424 0.7292
```

Here the adjusted R square is given 0.467, but what we can do is essentially we can just say summary of this. So, this is the bootstrap statistics for the R square adjusted R square and we can quantile say we can create a quantile of R squared with probability equal to 0.025 comma 0.975.

(Refer Slide Time: 09:10)

The screenshot shows an RStudio window with the following content:

```
94     ,apply(beta_star,2,quantile,probs=0.975))
95 colnames(sum_boot)<-c('Estimate','Std.Error','2.5%','97.5%')
96 summary(CAPM)
97 sum_boot
98
99 summary(R.squared_star.pair)
100 quantile(R.squared_star.pair,probs = c(0.025,0.975))
101 hist(beta_star[, 'alpha'],main = '',col='skyblue',freq = FALSE,x
102 lines(density(beta_star[, 'alpha']),col='purple',lwd=2)
103
```

Environment: Global Environment

sum_boot num [1:2, 1:4] -0.00...

sum_star List of 11

Values

b	1000L
B	1000

data 'zoo' series from 202...

Files: Files Packages Help Viewers Presentation

99:28 (Top Level) R Script

Console Terminal Background Jobs

R 4.2.2 - Downloads Regression_and_Classification/NPTEL

F-statistic: 54.44 on 1 and 60 DF, p-value: 5.647e-10

```
> summary(R.squared_star.pair)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1426 0.4045  0.4723  0.4702  0.5424  0.7292
> quantile(R.squared_star.pair,probs = c(0.025,0.975))
 2.5%    97.5%
0.2731307 0.6560612
>
```

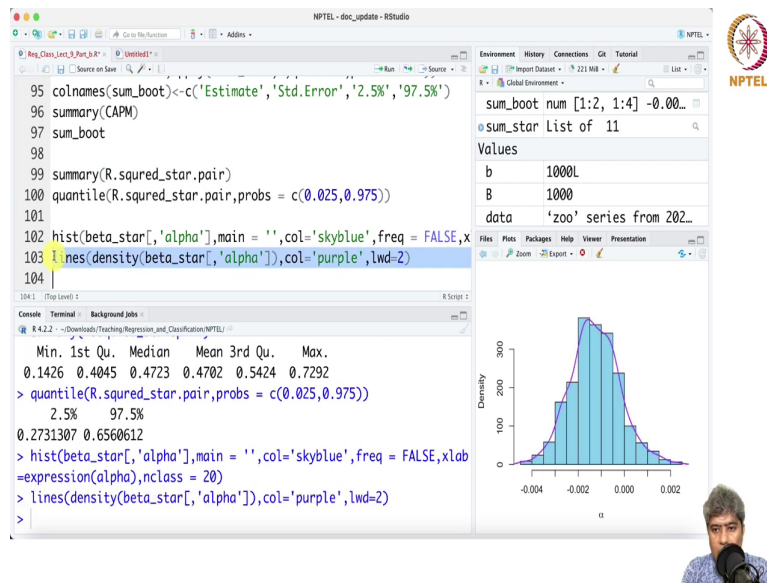
Sample Quantiles

Theoretical Quantiles

NPTEL

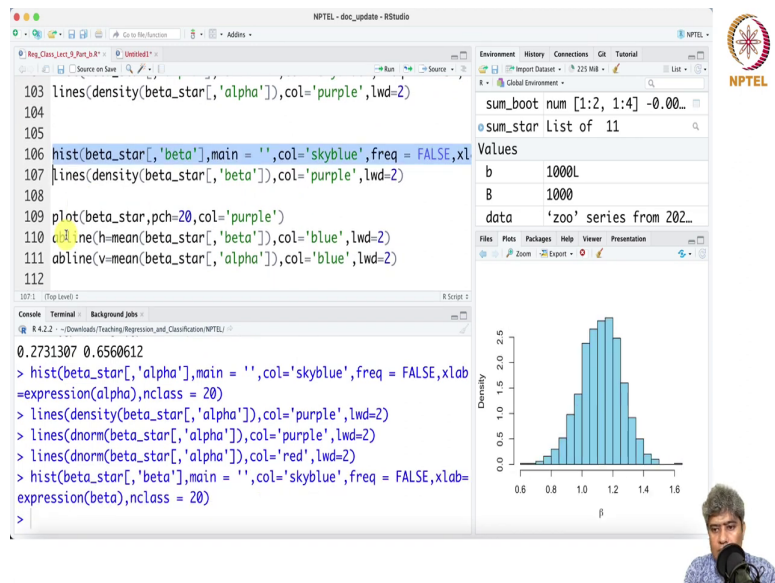
Now, this gives me a 95 percent confidence interval for adjusted R square. Now, in typically it is very difficult to get a any statistical inference for R square, but using bootstrap we can calculate a 95 percent confidence interval for adjusted R square.

(Refer Slide Time: 09:34)



So, this is a very interesting thing. So, you can here we have the alpha the sampling this is bootstrap sampling distribution of alpha and here is the density that we are drawing and then if you would have lines.

(Refer Slide Time: 09:57)



The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for density estimation and plotting:

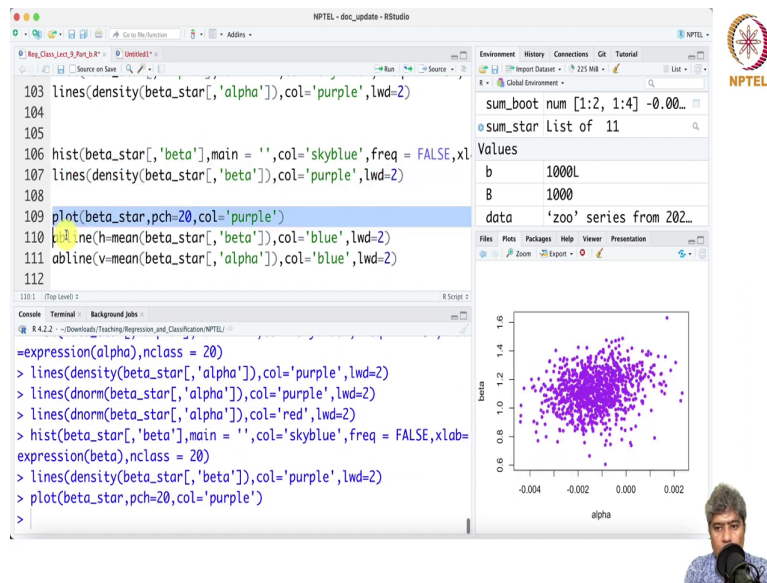
```
103 lines(density(beta_star[, 'alpha']), col='purple', lwd=2)
104
106 hist(beta_star[, 'beta'], main='', col='skyblue', freq = FALSE, xlab=expression(beta), nclass = 20)
107 lines(density(beta_star[, 'beta']), col='purple', lwd=2)
108
109 plot(beta_star, pch=20, col='purple')
110 abline(h=mean(beta_star[, 'beta']), col='blue', lwd=2)
111 abline(v=mean(beta_star[, 'alpha']), col='blue', lwd=2)
112
```
- Console:** Shows the execution of the code:

```
0.2731307 0.6560612
> hist(beta_star[, 'alpha'], main='', col='skyblue', freq = FALSE, xlab=expression(alpha), nclass = 20)
> lines(density(beta_star[, 'alpha']), col='purple', lwd=2)
> lines(dnorm(beta_star[, 'alpha']), col='purple', lwd=2)
> lines(dnorm(beta_star[, 'alpha']), col='red', lwd=2)
> hist(beta_star[, 'beta'], main='', col='skyblue', freq = FALSE, xlab=expression(beta), nclass = 20)
>
```
- Environment:** Lists objects in the workspace:

```
sum_boot num [1:2, 1:4] -0.00...
sum_star List of 11
```
- Plots:** A histogram showing the density of the 'beta' variable. The x-axis is labeled with the mathematical expression β and ranges from 0.6 to 1.6. The y-axis is labeled 'Density' and ranges from 0.0 to 2.5. The histogram bars are light blue. A vertical blue line is drawn at approximately $\beta = 1.1$, and a horizontal blue line is drawn at approximately $\text{Density} = 0.65$.

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 10:03)



The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for plotting density functions and a scatter plot. The code includes:

```
103 lines(density(beta_star[, 'alpha']), col='purple', lwd=2)
104
106 hist(beta_star[, 'beta'], main = '', col='skyblue', freq = FALSE, xlab='beta')
107 lines(density(beta_star[, 'beta']), col='purple', lwd=2)
108
109 plot(beta_star, pch=20, col='purple')
110 abline(h=mean(beta_star[, 'beta']), col='blue', lwd=2)
111 abline(v=mean(beta_star[, 'alpha']), col='blue', lwd=2)
112
```
- Console:** Shows the execution of the code, including the command `expression(alpha, nclass = 20)` and the resulting plot commands.
- Environment:** Lists objects in the environment, including `sum_boot num [1:2, 1:4] -0.00...` and `sum_star List of 11`.
- Plots:** A scatter plot is visible in the bottom right, showing a cloud of purple points with a horizontal blue line at approximately $y = 1.0$ and a vertical blue line at approximately $x = 0.0$. The x-axis is labeled 'alpha' and the y-axis is labeled 'beta'.

And similarly, we can do the histogram of the this is the beta and this is the lines.

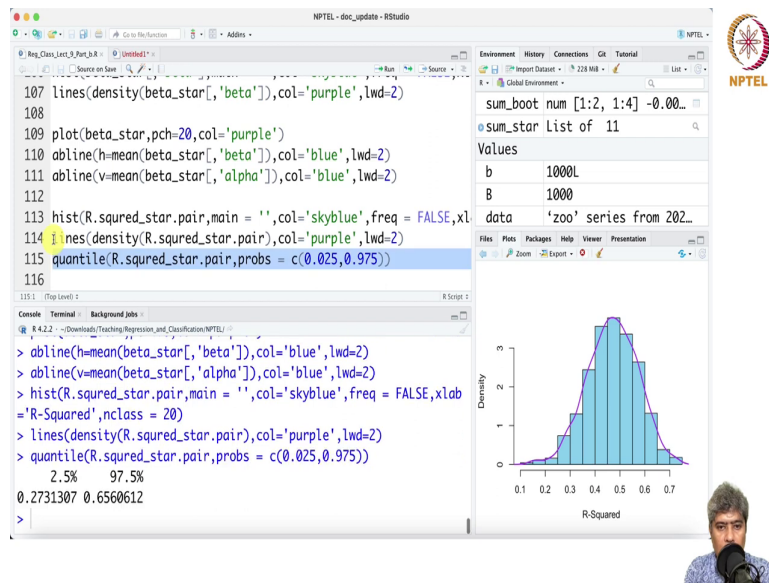
(Refer Slide Time: 10:15)

The screenshot displays an RStudio interface with the following components:

- Script Editor:** Contains R code for generating β_{star} and α_{star} . The code includes:
 - Line 103: `lines(density(beta_star[, 'alpha']), col='purple', lwd=2)`
 - Line 106: `hist(beta_star[, 'beta'], main = '', col='skyblue', freq = FALSE, xlab = expression(beta), nclass = 20)`
 - Line 109: `plot(beta_star, pch=20, col='purple')`
 - Line 110: `abline(h=mean(beta_star[, 'beta']), col='blue', lwd=2)`
 - Line 111: `abline(v=mean(beta_star[, 'alpha']), col='blue', lwd=2)`
- Console:** Shows the execution of the code, including the command `lines(dnorm(beta_star[, 'alpha']), col='purple', lwd=2)` and the resulting plot command `plot(beta_star, pch=20, col='purple')`.
- Environment:** Lists objects in the workspace, including `sum_boot num [1:2, 1:4] -0.00...` and `sum_star List of 11`.
- Plots:** A scatter plot showing the joint distribution of α (x-axis, ranging from -0.004 to 0.002) and β (y-axis, ranging from 0.6 to 1.6). The data points are purple dots, and the plot includes a horizontal blue line at $\beta \approx 1.0$ and a vertical blue line at $\alpha \approx 0.000$. The plot is titled "beta" on the y-axis and "alpha" on the x-axis.

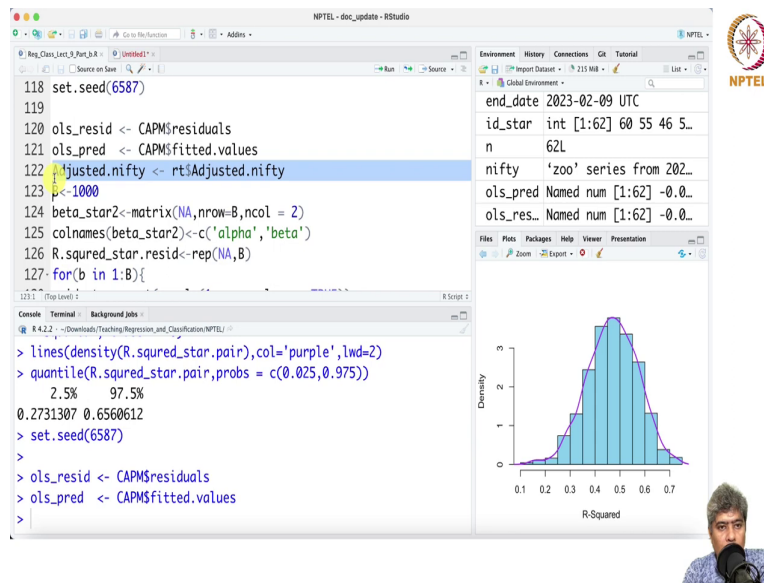
If we do the beta star this is the distribution between joint distribution of sampling distribution of alpha and beta. And we see that there is not much correlation that we are seeing between alpha and beta which is a good actually.

(Refer Slide Time: 10:26)



So, here is the sampling distribution of bootstrap sampling distribution of R square here is the lines density; kernel density here is the R square.

(Refer Slide Time: 10:41)



The screenshot displays the RStudio interface with the following code in the editor:

```
118 set.seed(6587)
119
120 ols_resid <- CAPM$residuals
121 ols_pred <- CAPM$fitted.values
122 njusted.nifty <- rt$Adjusted.nifty
123 b <- 1000
124 beta_star2 <- matrix(NA, nrow=B, ncol = 2)
125 colnames(beta_star2) <- c('alpha', 'beta')
126 R.squared_star.resid <- rep(NA, B)
127 for(b in 1:B){
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

The console shows the following output:

```
> lines(density(R.squared_star.pair), col='purple', lwd=2)
> quantile(R.squared_star.pair, probs = c(0.025, 0.975))
 2.5% 97.5%
0.2731307 0.6560612
> set.seed(6587)
>
> ols_resid <- CAPM$residuals
> ols_pred <- CAPM$fitted.values
>
```

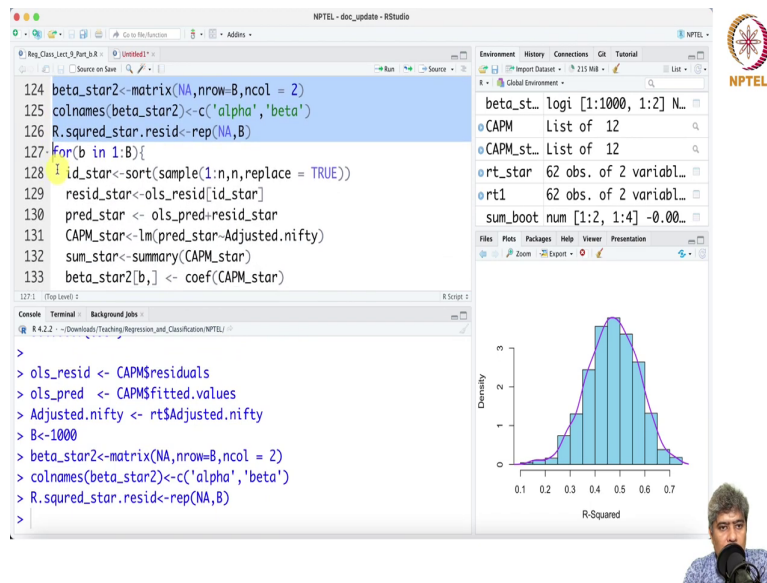
The Environment pane shows the following variables:

Variable	Class	Value
end_date	2023-02-09 UTC	
id_star	int [1:62]	60 55 46 5...
n	62L	
nifty	'zoo' series from 202...	
ols_pred	Named num [1:62]	-0.0...
ols_res...	Named num [1:62]	-0.0...

The Plots pane shows a density plot of R-squared values. The x-axis is labeled 'R-Squared' and ranges from 0.1 to 0.7. The y-axis is labeled 'Density' and ranges from 0 to 3. The plot shows a distribution of R-squared values with a purple density curve overlaid on a histogram.

Now, we are going to do the residual resampling or Residual Bootstrap Progression ok and from the CAPM we just take the OLS residual and OLS predicted values and then this is the adjusted nifty and same way we define the matrixes.

(Refer Slide Time: 11:11)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
124 beta_star2<-matrix(NA,nrow=B,ncol = 2)
125 colnames(beta_star2)<-c('alpha','beta')
126 R.squared_star.resid<-rep(NA,B)
127 for(b in 1:B){
128   id_star<-sort(sample(1:n,n,replace = TRUE))
129   resid_star<-ols_resid[id_star]
130   pred_star <- ols_pred+resid_star
131   CAPM_star<-lm(pred_star-Adjusted.nifty)
132   sum_star<-summary(CAPM_star)
133   beta_star2[b,] <- coef(CAPM_star)
```

The Environment pane on the right shows the following objects:

- beta_star2: logi [1:1000, 1:2] N...
- CAPM_star: List of 12
- CAPM_star.resid: List of 12
- rt_star: 62 obs. of 2 variabl...
- rt1: 62 obs. of 2 variabl...
- sum_boot num [1:2, 1:4] -0.00...

The Console window shows the execution of the code:

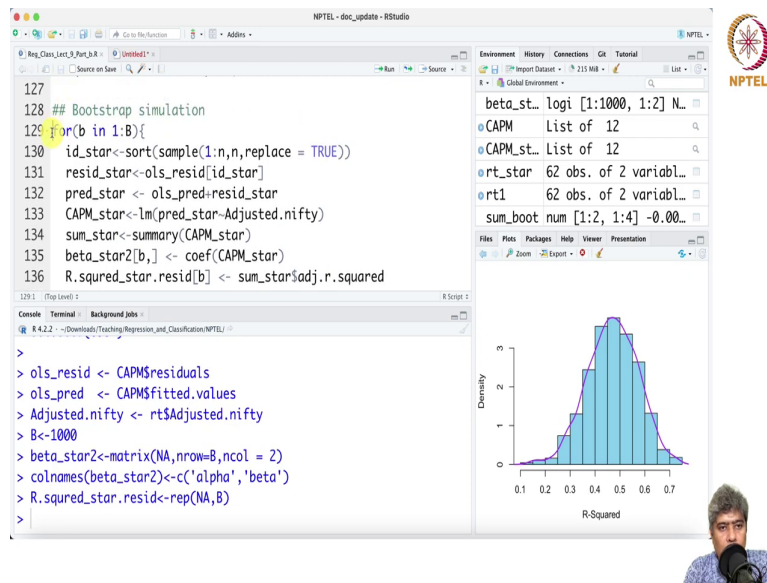
```
>
> ols_resid <- CAPM$residuals
> ols_pred <- CAPM$fitted.values
> Adjusted.nifty <- rt$Adjusted.nifty
> B<-1000
> beta_star2<-matrix(NA,nrow=B,ncol = 2)
> colnames(beta_star2)<-c('alpha','beta')
> R.squared_star.resid<-rep(NA,B)
>
```

A histogram is displayed in the bottom right corner, showing the distribution of R-squared values. The x-axis is labeled 'R-Squared' and ranges from 0.1 to 0.7. The y-axis is labeled 'Density' and ranges from 0 to 3. The histogram bars are light blue, and a red normal distribution curve is overlaid on the data.

The NPTEL logo is visible in the top right corner of the RStudio window. A small video feed of the presenter is located in the bottom right corner of the slide.

Now, here we have starting the bootstrap statistic Bootstrap simulation.

(Refer Slide Time: 11:18)



The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for a bootstrap simulation:

```
127  
128 ## Bootstrap simulation  
129 for(b in 1:B){  
130   id_star<-sort(sample(1:n,n,replace = TRUE))  
131   resid_star<-ols_resid[id_star]  
132   pred_star <- ols_pred+resid_star  
133   CAPM_star<-lm(pred_star-Adjusted.nifty)  
134   sum_star<-summary(CAPM_star)  
135   beta_star2[b,] <- coef(CAPM_star)  
136   R.squared_star.resid[b] <- sum_star$adj.r.squared
```
- Environment:** Lists objects: `beta_star2` (logi [1:1000, 1:2] N...), `CAPM_star` (List of 12), `CAPM_star2` (List of 12), `rt_star` (62 obs. of 2 variabl...), `rt1` (62 obs. of 2 variabl...), and `sum_boot num` ([1:2, 1:4] -0.00...).
- Console:** Shows the execution of the following commands:

```
> ols_resid <- CAPMS$residuals  
> ols_pred <- CAPMS$fitted.values  
> Adjusted.nifty <- rt$Adjusted.nifty  
> B<-1000  
> beta_star2<-matrix(NA,nrow=B,ncol = 2)  
> colnames(beta_star2)<-c('alpha', 'beta')  
> R.squared_star.resid<-rep(NA,B)  
>
```
- Plots:** A density plot of R-squared values. The x-axis is labeled "R-Squared" and ranges from 0.1 to 0.7. The y-axis is labeled "Density" and ranges from 0 to 3. The plot shows a distribution of R-squared values with a peak around 0.5.
- UI Elements:** The NPTEL logo is visible in the top right corner. A small video feed of a person speaking is located in the bottom right corner.

Now, here is the id star I am from the OLS residual I am just simulating the residuals, then from the predicted ols predicted value you just take the predicted star predicted value, because $X\hat{\beta}$ is nothing but the ols predicted value right and then you just add the error and that will give you the Y star. And then based on the Y star you compute the again fitted value and CAPM star and then all you do from there you compute the coefficients and from the and adjusted R square.

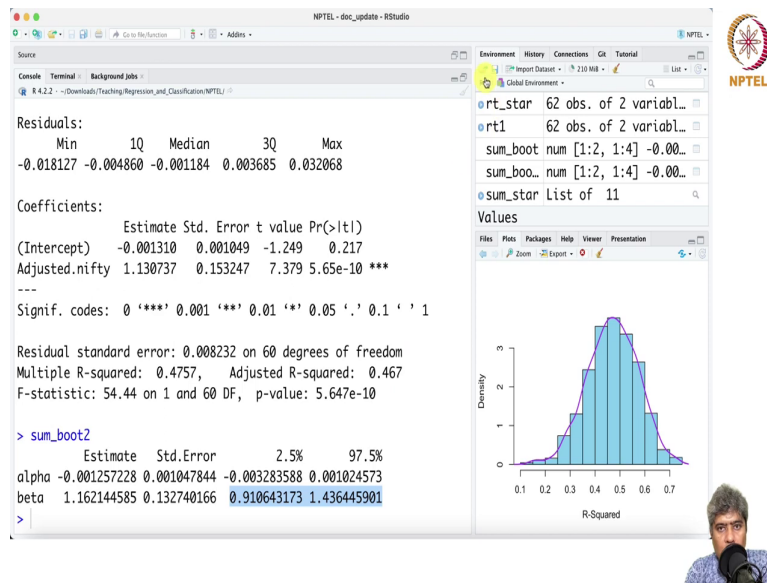
(Refer Slide Time: 12:20)

The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for a simulation. Line 142 is highlighted: `colnames(sum_boot2) <- c('Estimate', 'Std.Error', '2.5%', '97.5%')`. Other lines include `summary(CAPM)` and `sum_boot2`.
- Environment:** Lists objects: `rt_star` (62 obs. of 2 variables), `rt1` (62 obs. of 2 variables), `sum_boot` (matrix), `sum_boo...` (matrix), and `sum_star` (List of 11).
- Console:** Shows the execution of the code from the source editor, including `summary(CAPM)` and the `sum_boot2` object.
- Plots:** A histogram showing the distribution of R-squared values. The x-axis is labeled 'R-Squared' and ranges from 0.1 to 0.7. The y-axis is labeled 'Density' and ranges from 0 to 3. A normal distribution curve is overlaid on the histogram.
- NPTEL Logo:** Located in the top right corner of the RStudio window.
- Speaker:** A small video feed of a person speaking is visible in the bottom right corner.

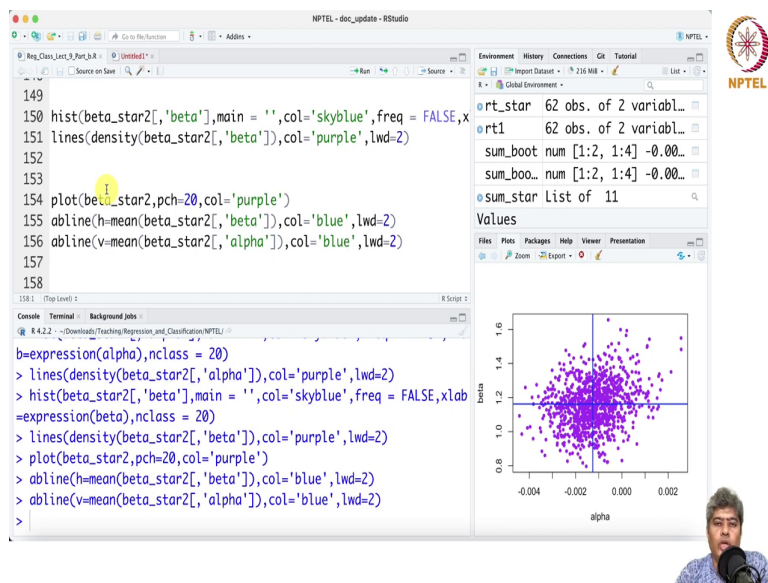
So, same way you just do that it is very fast for 1000 simulation it does very quickly actually.

(Refer Slide Time: 12:33)



So, here is the estimates are pretty much very close to similar previously and confidence interval also does not change much.

(Refer Slide Time: 12:48)



So, residual bootstrap holds pretty much similar not much change because and this is the alpha distribution of alpha, this is the distribution of beta, this is the joint distribution of alpha and beta when we are doing residual bootstrap regression. So, that is how we do paired bootstrap statistics or bootstrap regression and residual bootstrap regression.

If sometimes what happens as you know that you fit the model and what happens is you fit a model you did a very good job with modeling and do a reasonable predict prediction also. Suppose you have achieved a descent R squared very reasonably low RMSE, but and most likely you are ok to you know accept the model to the production.

Now, people can still re-question that if the model assumption does not hold good. Now, often time what I have seen that if this is the if the case is that randomness is ok and homoscedasticity is ok. Then and but the it is the distribution that is the problem because

distribution is always a very strong assumption; then I will always recommend you to resort to bootstrap.

So, if you do the bootstrapping what happens is you still work with the same model only the statistical testing of hypothesis and the inferences. Now, you are doing with the bootstrap distribution instead of assuming residual is normal. But rest of the assumptions is perfectly fine and if even the homoscedasticity does not hold good you can still work with paired bootstrap statistics and it will still work.

So, bootstrap regression is a big help for the statistician in case you have to do a statistical inference, that particularly if the model prediction is good. If the model prediction is not good then of course, you have to figure out a better model in that case.

But if the model prediction is good if the model is doing reasonable prediction in the out of the sample, if the model is doing if the RMSE is pretty low, if the R square adjusted R square is reasonably acceptable range for the given domain. Then just for inference you may find that the normality assumption is too strong assumption.

In that case you can do a statistical inference using bootstrap statistics and still you can go on with that prediction of the model, that predictive model because that predictive model do reason if it does do if it does good with prediction good prediction, then inference can be done using bootstrap sampling distribution. So, you will be good in that sense. You can salvage that situation when the model check assumption is not holding good.

As long as the IID sample assumption is working fine. You can salvage that situation using simple bootstrap statistic technique. So, with that I will stop here for this lecture and looking forward to the next lecture. I hope you enjoy this hands on session it is a very useful I find it in my many real life project I find it very useful this technique. So, I hope you will enjoy it and you will be able to use it and for your project experience.

So, thank you very much and see you in the next topic and next video next lecture.

