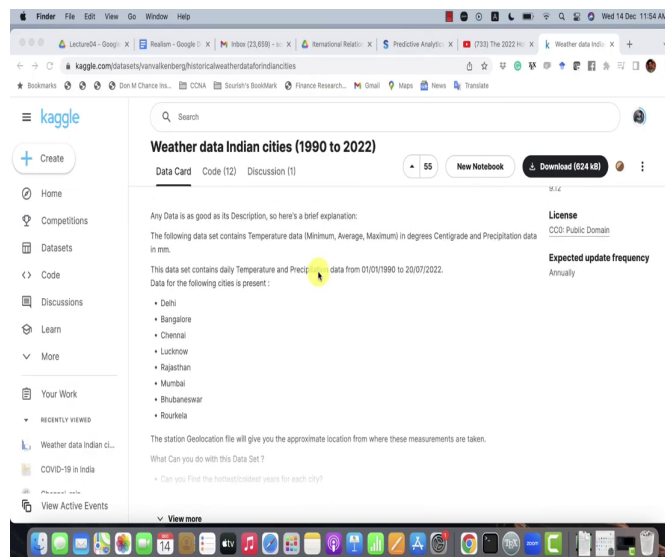


Predictive Analytics - Regression and Classification
Prof. Sourish Das
Department of Mathematics
Chennai Mathematical Institute

Lecture - 16
Hands-on with R Part-4

Welcome back to the part C of lecture 4. This part we are going to discuss the hands on, we are going to do some hands on of fitting linear models with feature engineering etcetera.

(Refer Slide Time: 00:36)



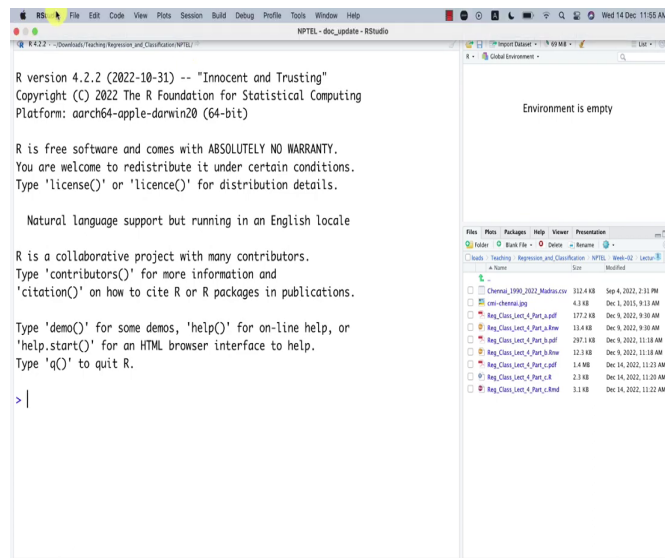
So, for this we are going to use the weather data of Indian cities from the Kaggle. I am going to share this data, this website this thing this page with you guys, as well as if you I am going to share the data. So, we are going to use the data for Chennai, you can try the same thing with other cities as well.

(Refer Slide Time: 00:55)

The screenshot shows a web browser displaying the Kaggle dataset page for "Weather data Indian cities (1990 to 2022)". The page title is "Weather data Indian cities (1990 to 2022)" and it has 55 data cards and 1 discussion. The dataset is categorized under "Weather and Climate". The main content area shows a directory titled "Temperature_And_Precipitation_Cities_IN (9 files)". Below the directory title, there is a section "About this directory" which states: "Here Each CSV file contains temperature and precipitation data, the location is the worst word in the file name. The column description is as follows: time: the date when measurements were taken temp: Average temperature in degree Centigrade. tmin: Minimum temperature in degree Centigrade." Below this text, there are four file cards showing the first few files in the directory: "Bangalore_1990_2022.csv" (390.23 kB), "Chennai_1990_2022.M..." (318.83 kB), "Delhi_NCR_1990_2022..." (322.3 kB), and "Lucknow_1990_2022.csv" (315.33 kB). On the right side, there is a "Data Explorer" panel showing a tree view of the files, with "Temperature_And_Precipitation_Cities_IN" expanded to show the individual CSV files. The browser's address bar shows the URL "kaggle.com/datasets/vanvaikariberg/historicalweatherdataforindiancities". The browser's taskbar at the bottom shows various application icons, and a small video feed of a person is visible in the bottom right corner.

So, here is all the data sets are available. So, we are going to use the Chennai data set. The data set is available from 1990 to 2022, ok.

(Refer Slide Time: 01:15)



R version 4.2.2 (2022-10-31) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: aarch64-apple-darwin20 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

```
> |
```

Environment is empty

File Name	Size	Modified
Chennai_1995_2022_Median.csv	312.4 KB	Sep 4, 2022, 2:15 PM
cmr-climate.jpg	4.3 KB	Dec 1, 2021, 9:15 AM
Reg_Class_Lect_4_Part_1.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_1.Rnw	13.4 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_3.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_3.Rnw	12.3 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_4.pdf	1.4 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_4.R	2.3 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_4.Rmd	3.1 KB	Dec 14, 2022, 11:20 AM

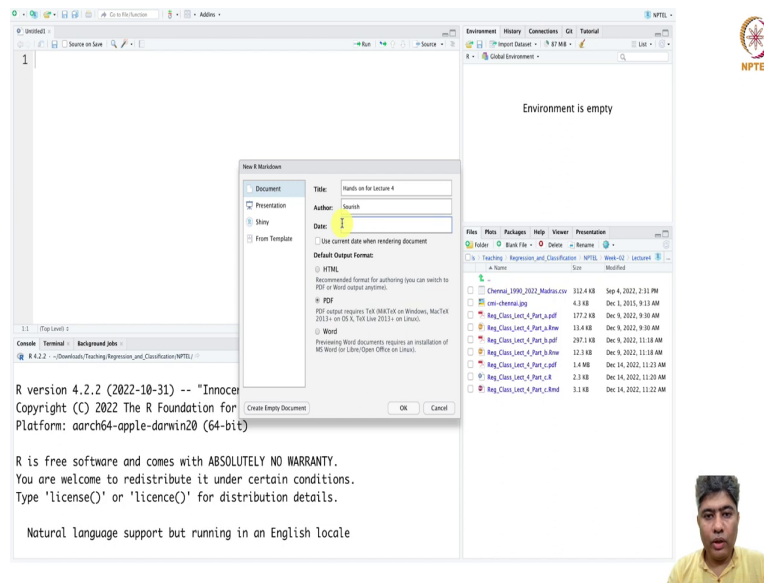


(Refer Slide Time: 01:17)

The screenshot displays the RStudio application window. The 'File' menu is open, showing options such as 'New Project...', 'New File...', 'Open File...', 'Recent Files', 'Open Project...', 'Import Dataset...', 'Save', 'Save As...', 'Save All', 'Print...', 'Close', 'Close All', 'Close Project', 'Quit Session...', 'R Notebook', 'R Markdown...', 'Shiny Web App...', 'Plumber API...', 'C File', 'Header File', 'Markdown File', 'HTML File', 'CSS File', 'JavaScript File', 'DS Script', 'Python Script', 'Shell Script', 'SQL Script', 'Stan File', 'Text File', 'R Source', 'R HTML', and 'R Documentation...'. The console window on the right shows the message 'Environment is empty'. The bottom right corner features a small video feed of a man in a yellow shirt.

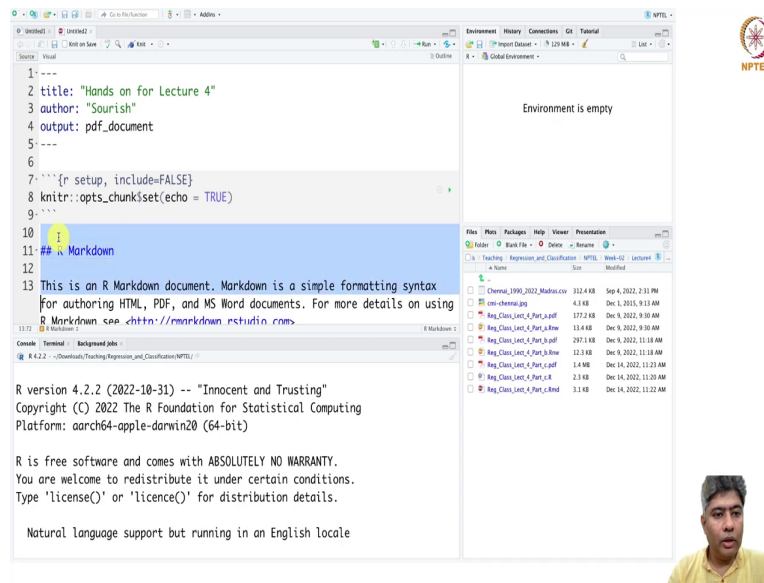


(Refer Slide Time: 01:22)



So, what we will do? We will first start R studio, ok. And in the R studio, first thing we will open is R script and an R markdown file, in the file, R markdown file.

(Refer Slide Time: 01:49)



The screenshot displays the RStudio environment with an R Markdown file open. The editor shows the following code:

```
1: ---
2: title: "Hands on for Lecture 4"
3: author: "Saurish"
4: output: pdf_document
5: ---
6:
7: {r setup, include=FALSE}
8: knitr::opts_chunkset(echo = TRUE)
9: ---
10:
11: ## R Markdown
12:
13: This is an R Markdown document. Markdown is a simple formatting syntax
    for authoring HTML, PDF, and MS Word documents. For more details on using
    R Markdown see <http://rmarkdown.rstudio.com>
```

The console output shows:

```
R version 4.2.2 (2022-10-31) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: aarch64-apple-darwin20 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale
```

The file explorer on the right shows a list of files:

File	Size	Modified
Chennai_1995_2022_MathSci.csv	312.4 KB	Sep 4, 2022, 2:13 PM
chir-chirneye.jpg	4.3 KB	Dec 1, 2021, 9:13 AM
Reg_Class_Lect_4_Part_1a.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_1a.Rnw	13.4 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_1a.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_1a.Rnw	12.3 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_1a.pdf	1.4 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_1a.R	2.3 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_1a.Rmd	3.1 KB	Dec 14, 2022, 11:20 AM

The NPTEL logo is visible in the top right corner, and a small video feed of a person is in the bottom right corner.

In the R markdown file, we will first write that hands on for lecture 4. We will choose PDF; we do not need, ok.

(Refer Slide Time: 02:02)

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

```
1 ---
2 title: "Hands on for Le
3 author: "Saurish"
4 output: pdf_document
5 ---
6
7 {r setup, include=FA
8 knitr::opts_chunkset(e
9 ...
10
```

A "Save File - Untitled2" dialog box is open in the center, showing the file name "Hands_on_Lecture_p4" and the location "NPTEL".

The console window at the bottom left shows the R version and platform information:

```
R version 4.2.2 (2022-10-31) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: aarch64-apple-darwin20 (64-bit)
```

Below this, a message states: "R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details. Natural language support but running in an English locale".

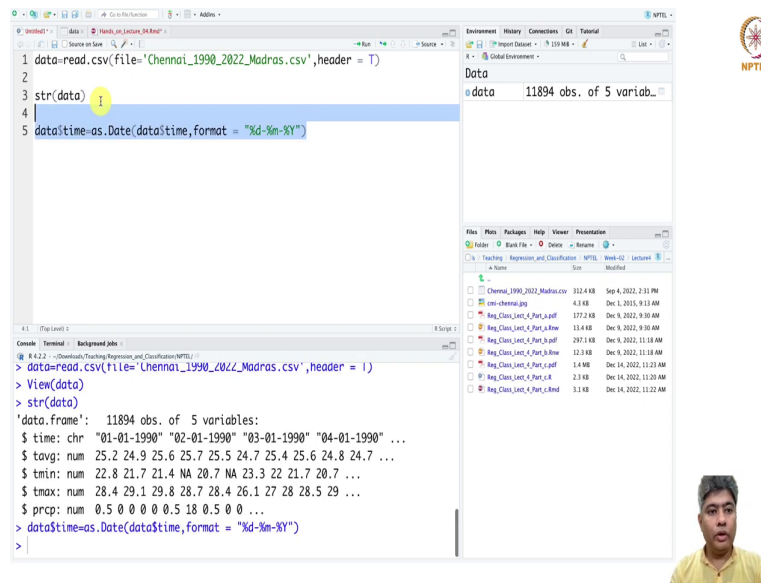
The environment pane on the right shows "Environment is empty".

The file browser on the right shows a list of files and folders:

File	Size	Modified
992_2022_Madras.csv	332.4 KB	Sep 4, 2022, 2:13 PM
999	4.3 KB	Dec 1, 2022, 9:13 AM
lec_4_Part_a.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
lec_4_Part_a.Rnw	13.4 KB	Dec 9, 2022, 9:30 AM
lec_4_Part_b.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
lec_4_Part_b.Rnw	12.3 KB	Dec 9, 2022, 11:18 AM
lec_4_Part_c.pdf	3.4 KB	Dec 14, 2022, 11:20 AM
lec_4_Part_c.R	2.3 KB	Dec 14, 2022, 11:20 AM
lec_4_Part_c.Rmd	3.1 KB	Dec 14, 2022, 11:20 AM



(Refer Slide Time: 02:17)



```
1 data=read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
2
3 str(data)
4
5 data$time=as.Date(data$time,format = "%d-%m-%Y")
```

Console Terminal Background jobs


```
> data=read.csv(file='Chennai_1990_2022_Madras.csv',header = 1)
> View(data)
> str(data)
'data.frame': 11894 obs. of 5 variables:
 $ time: chr "01-01-1990" "02-01-1990" "03-01-1990" "04-01-1990" ...
 $ tavg: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
 $ tmin: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
 $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
 $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 ...
> data$time=as.Date(data$time,format = "%d-%m-%Y")
>
```

Data

edata 11894 obs. of 5 variab...

Files

File Name	Size	Modified
Chennai_1990_2022_Madras.csv	332.4 KB	Sep 4, 2022, 2:13 PM
cm-chennai.jpg	4.3 KB	Dec 1, 2022, 9:13 AM
Reg_Class_Lect_4_Part_1.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_1.ppt	13.4 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_2.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_2.ppt	12.3 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_3.pdf	1.4 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_3.ppt	2.3 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_4.pdf	3.1 KB	Dec 14, 2022, 11:20 AM



And then, what we will do, we will just save them as a hands on for lecture 4. And the first thing we have to do is we have to read the data. So, data equal to read dot csv file equal to Chennai, 1990 to 2022, Madras dot csv header equals to true.

(Refer Slide Time: 02:44)

The screenshot displays the R Studio environment. The main window shows a data frame with 11894 observations and 5 variables. The console shows the following commands and output:

```
> data=read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
> View(data)
>
```

time	temp	min	max	precip
1 01-01-1990	25.2	22.8	28.4	0.5
2 02-01-1990	24.9	22.7	29.1	0.0
3 03-01-1990	25.8	23.4	29.8	0.0
4 04-01-1990	25.7	23.1	29.7	0.0
5 05-01-1990	25.5	23.7	29.4	0.0
6 06-01-1990	24.7	23.0	26.3	0.5
7 07-01-1990	25.4	23.3	27.0	18.0
8 08-01-1990	25.8	23.0	28.0	0.5
9 09-01-1990	24.8	22.7	28.5	0.0
10 10-01-1990	24.7	23.7	29.0	0.0
11 11-01-1990	24.5	23.0	28.8	0.0
12 12-01-1990	23.8	18.0	28.4	0.0
13 13-01-1990	23.1	17.1	23.3	0.0
14 14-01-1990	23.9	19.5	23.8	0.0
15 15-01-1990	23.3	17.4	30.5	0.0
16 16-01-1990	23.3	18.0	29.8	0.0
17 17-01-1990	23.4	18.1	28.9	0.0
18 18-01-1990	22.6	18.2	30.2	0.0
19 19-01-1990	23.0	17.2	30.4	0.0
20 20-01-1990	23.1	16.9	31.3	0.0



So, here is the data that we have seen. So, this is the first day of the data. So, if you look into the date, date, it is like day first, then month, then year. And we look into the structure of the data, it is the structure of the data is it is a data frame with 11894 observation and 5 variables.

And time is stored as character. So, we have to do it properly as a date fine, as a date column, we have to store it as a date column, data dollar time format should be percentage, date first, then percentage m, then percentage year.

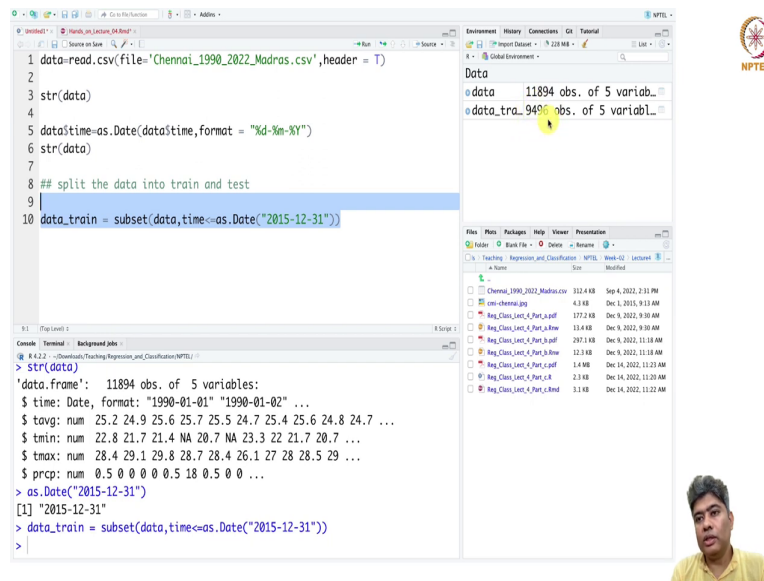
(Refer Slide Time: 03:44)

The screenshot displays the RStudio interface. The main window shows a data frame with 11894 observations and 5 variables. The variables are time, tavg, tmin, tmax, and prcp. The console shows the output of the following R code:

```
> str(data)
'data.frame': 11894 obs. of 5 variables:
 $ time: chr "01-01-1990" "02-01-1990" "03-01-1990" "04-01-1990" ...
 $ tavg: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
 $ tmin: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
 $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
 $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 ...
> data$time=as.Date(data$time,format = "%d-%m-%Y")
> View(data)
```



(Refer Slide Time: 03:50)



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

```
1 data-read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
2
3 str(data)
4
5 data$time<-as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 ## split the data into train and test
9
10 data_train = subset(data,time<=as.Date("2015-12-31"))
```

The console shows the output of the code:

```
> str(data)
'data.frame': 11894 obs. of 5 variables:
 $ time: Date, format: "1990-01-01" "1990-01-02" ...
 $ tavg: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
 $ tmin: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
 $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
 $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 0 ...
> as.Date("2015-12-31")
[1] "2015-12-31"
> data_train = subset(data,time<=as.Date("2015-12-31"))
>
```

The Data viewer shows the structure of the data:

```
data      11894 obs. of 5 variab...
data_tra_9496 obs. of 5 variabL...
```

The file explorer shows the following files:

File Name	Size	Modified
Chennai_1990_2022_Madras.csv	332.4 KB	Sep 4, 2022, 2:13 PM
chennai-chennai.jpg	4.3 KB	Dec 1, 2022, 9:13 AM
Reg_Class_Lect_4_Part_1.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_1.kw	13.4 KB	Dec 9, 2022, 9:30 AM
Reg_Class_Lect_4_Part_2.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_2.kw	12.3 KB	Dec 9, 2022, 11:18 AM
Reg_Class_Lect_4_Part_3.pdf	1.4 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_3.kw	2.3 KB	Dec 14, 2022, 11:20 AM
Reg_Class_Lect_4_Part_4.pdf	3.1 KB	Dec 14, 2022, 11:20 AM

Now, if I do that and if we just check, it is read as this way. And if we know structure of data is (Refer Time: 03:55) this. Now, you can see the time is read as date with format in this format. So, now, it is a date format, not character format. So, the first thing we will see is we will split the data into train and test, split the data into train and test. So, data train equals to subset, data, time as less than equal to as dot Date.

Now, we have to give it in this format. So, date, month and day; year, month and day. So, the first thing will be 2015, then month will be 12, and day will be 31. I hope this is fine and then let us try. So, now, in the train data set you can see 2000 90; 9496 observations are there.

(Refer Slide Time: 05:30)

The screenshot displays the RStudio interface. The main window shows a data frame with 11894 observations and 5 variables: time, tavg, tmin, tmax, and prcp. The console shows the following R code and output:

```
data_train = subset(data, time <= as.Date("2015-12-31"))
View(data_train)
```

The console output shows the structure of the data frame:

```
data_train: 11894 obs. of 5 variables:
 $ time: Date, format: "1990-01-01" "1990-01-02" ...
 $ tavg: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
 $ tmin: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
 $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
 $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 0 ...
> as.Date("2015-12-31")
[1] "2015-12-31"
```



(Refer Slide Time: 05:34)

The screenshot displays the RStudio interface. The main window shows a data frame with 11894 observations and 5 variables. The console shows the following R code and output:

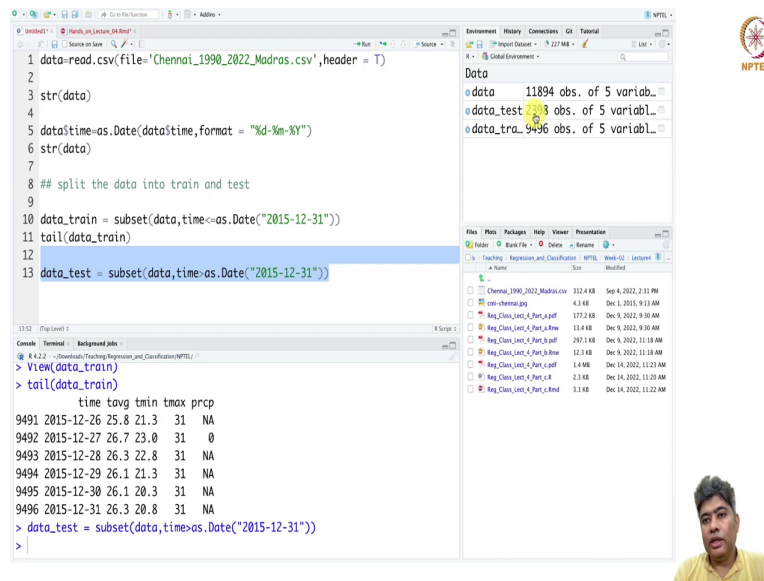
```
data_train = subset(data,time<=as.Date("2015-12-31"))
View(data_train)
```

The output in the console is:

```
data_train: 11894 obs. of 5 variables:
 $ time: Date, format: "1990-01-01" "1990-01-02" ...
 $ tavg: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
 $ tmin: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
 $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
 $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 0 ...
> as.Date("2015-12-31")
[1] "2015-12-31"
> data_train = subset(data,time<=as.Date("2015-12-31"))
> View(data_train)
```



(Refer Slide Time: 05:40)





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

```
1 data<-read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
2
3 str(data)
4
5 data$time<-as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 ## split the data into train and test
9
10 data_train = subset(data,time<=as.Date("2015-12-31"))
11 tail(data_train)
12
13 data_test = subset(data,time>as.Date("2015-12-31"))
```

The console shows the output of the `tail(data_train)` command:

```
> View(data_train)
> tail(data_train)
      time tavg tmin tmax prcp
9491 2015-12-26 25.8 21.3  31  NA
9492 2015-12-27 26.7 23.0  31   0
9493 2015-12-28 26.3 22.8  31  NA
9494 2015-12-29 26.1 21.3  31  NA
9495 2015-12-30 26.1 20.3  31  NA
9496 2015-12-31 26.3 20.8  31  NA
> data_test = subset(data,time>as.Date("2015-12-31"))
>
```

The file explorer on the right shows a list of files, including `Chennai_1990_2022_Madras.csv` and several PDF files related to regression and classification.



So, it is starting from 1990 and last day is 31st December, 2012. So, also if I just say train tail of the data train. So, up to the last day is 31st December 2015, 31st December 2015. This is the train data. So, same way we can define the test data.

(Refer Slide Time: 06:14)

The screenshot displays the RStudio interface. The main window shows a data frame with the following columns: 'time', 'avg', 'tmin', 'tmax', and 'prcp'. The console shows the following commands and output:

```
> tail(data_train)
      time avg tmin tmax prcp
9491 2015-12-26 25.8 21.3  31  NA
9492 2015-12-27 26.7 23.0  31   0
9493 2015-12-28 26.3 22.8  31  NA
9494 2015-12-29 26.1 21.3  31  NA
9495 2015-12-30 26.1 20.3  31  NA
9496 2015-12-31 26.3 20.8  31  NA

> data_test = subset(data, time >= Date("2015-12-31"))
> View(data_test)
```

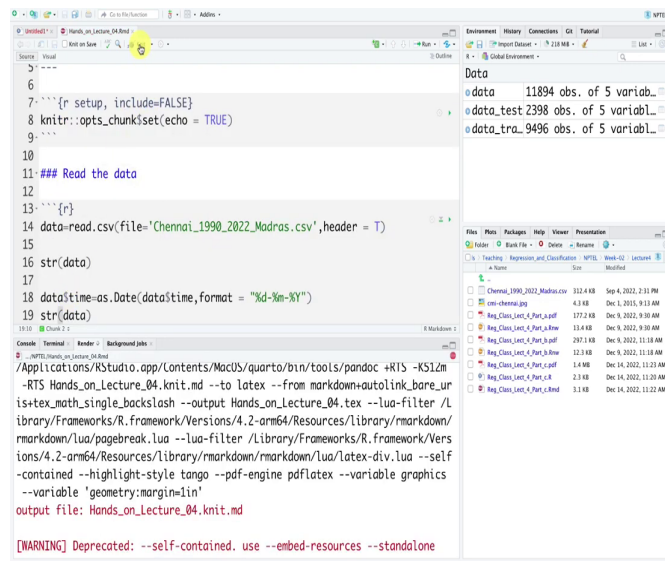
The file explorer shows a folder named 'Teaching Regression and Classification NPTEL' with the following files:

- Chemo_1995_2022_Matlab.csv (312.4 KB, Sep 4, 2022, 2:13 PM)
- cin-climate.csv (4.2 KB, Dec 1, 2021, 9:13 AM)
- Reg_Class_Lect_4_Part_1.pdf (177.2 KB, Dec 9, 2022, 9:30 AM)
- Reg_Class_Lect_4_Part_1.Rnw (13.4 KB, Dec 9, 2022, 9:30 AM)
- Reg_Class_Lect_4_Part_3.pdf (297.1 KB, Dec 9, 2022, 11:18 AM)
- Reg_Class_Lect_4_Part_3.Rnw (12.3 KB, Dec 9, 2022, 11:18 AM)
- Reg_Class_Lect_4_Part_4.pdf (1.4 KB, Dec 14, 2022, 11:20 AM)
- Reg_Class_Lect_4_Part_4.R (2.3 KB, Dec 14, 2022, 11:20 AM)
- Reg_Class_Lect_4_Part_4.Rmd (3.1 KB, Dec 14, 2022, 11:20 AM)



So, we will just take the same thing, instead of less than equal to we will make it simply greater than. If we do that test data has 2398 and the first day is 1st January 2016, is the first day of the data. And the last day of the data is 25th July 2022. So, this is what we are going to do now. What we will do? We will put it in the knitr.

(Refer Slide Time: 06:40)



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

```
6  
7: ```{r setup, include=FALSE}  
8 knitr::opts_chunkset(echo = TRUE)  
9: ```  
10  
11 ## Read the data  
12  
13 ```{r}  
14 data<-read.csv(file='Chennai_1990_2022_Madras.csv', header = T)  
15  
16 str(data)  
17  
18 data$time<-as.Date(data$time, format = "%d-%m-%Y")  
19 str(data)  
20  
21 ```
```

The console window shows the execution of the code, including the command to run the presentation:

```
15:10 | Chunk 2 |  
/Applications/RStudio.app/Contents/MacOS/quarto/bin/tools/pandoc +RTS -K512M  
-RTS Hands_on_Lecture_04.knit.md --to latex --from markdown+autolink_bare_ur  
is+tex_math_single_backslash --output Hands_on_Lecture_04.tex --lua-filter /L  
ibrary/Frameworks/R.framework/Versions/4.2-arm64/Resources/library/markdown/  
rmarkdown/Lua/pagebreak.lua --lua-filter /Library/Frameworks/R.framework/Vers  
ions/4.2-arm64/Resources/library/markdown/rmarkdown/Lua/latex-div.lua --self  
-contained --highlight-style tango --pdf-engine pdflatex --variable graphics  
--variable 'geometry:margin=1in'  
output file: Hands_on_Lecture_04.knit.md  
  
[WARNING] Deprecated: --self-contained. use --embed-resources --standalone
```

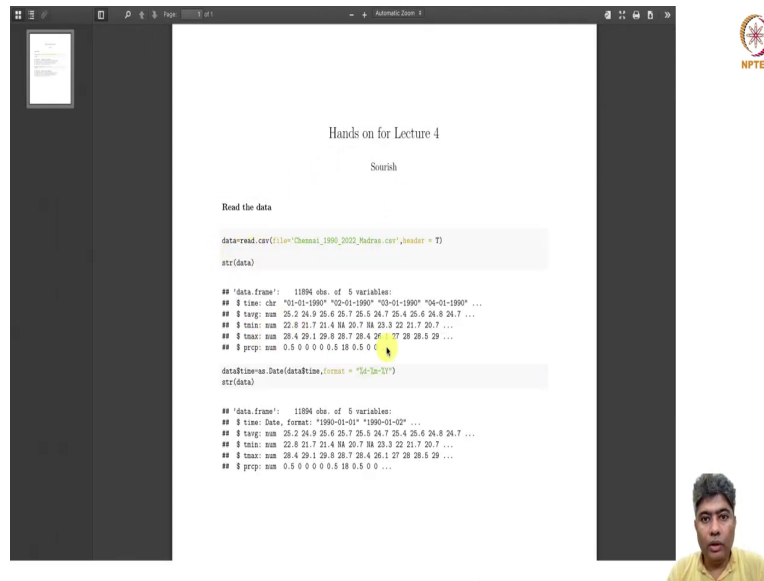
The file explorer window shows a list of files:

File Name	Size	Modified
Chennai_1990_2022_Madras.csv	332.4 KB	Sep 4, 2022, 2:13 PM
chennai-ggij	4.2 KB	Dec 1, 2022, 9:13 AM
Reg_Chennai_Part_1.pdf	177.2 KB	Dec 9, 2022, 9:30 AM
Reg_Chennai_Part_1.Rnw	13.4 KB	Dec 9, 2022, 9:30 AM
Reg_Chennai_Part_2.pdf	297.1 KB	Dec 9, 2022, 11:18 AM
Reg_Chennai_Part_2.Rnw	12.3 KB	Dec 9, 2022, 11:18 AM
Reg_Chennai_Part_3.pdf	1.4 KB	Dec 14, 2022, 11:20 AM
Reg_Chennai_Part_3.Rnw	2.3 KB	Dec 14, 2022, 11:20 AM
Reg_Chennai_Part_4.pdf	3.1 KB	Dec 14, 2022, 11:20 AM



So, first thing is we and we will keep going read the data. So, up to this we can just put it in the read the data part, so here, and then if we just run it.

(Refer Slide Time: 07:10)



Hands on for Lecture 4

Sourish



Read the data

```
dataread.csv(file="Chemist_1990_2022_hadron.csv",header = T)
str(data)
```

```
## 'data.frame': 11894 obs. of 5 variables:
## $ time: chr "01-01-1990" "02-01-1990" "03-01-1990" "04-01-1990" ...
## $ temp: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ time: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
## $ time: num 28.4 29.1 29.8 28.7 28.4 28.7 28.5 29 ...
## $ prep: num 0.5 0 0 0 0.5 18 0.5 0 ...
```

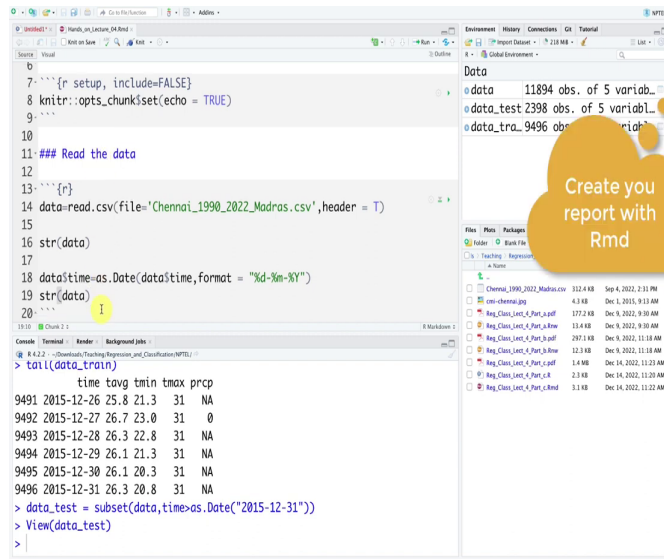
```
dataTime=as.Date(data$time,format = "%d-%m-%Y")
str(data)
```

```
## 'data.frame': 11894 obs. of 5 variables:
## $ time: Date, format: "1990-01-01" "1990-01-02" ...
## $ temp: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ time: num 22.8 21.7 21.4 NA 20.7 NA 23.3 22 21.7 20.7 ...
## $ time: num 28.4 29.1 29.8 28.7 28.4 28.1 27 28 28.5 29 ...
## $ prep: num 0.5 0 0 0 0.5 18 0.5 0 ...
```



So, here in the R markdown, nicely it is come in a PDF; you can create the report of your analysis alongside whatever you are doing.

(Refer Slide Time: 07:21)



```
7. ## {r setup, include=FALSE}
8. knitr::opts_chunkset(echo = TRUE)
9.
10.
11. ## Read the data
12.
13. ## {r}
14. data=read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
15.
16. str(data)
17.
18. data$time=as.Date(data$time,format = "%d-%m-%Y")
19. str(data)
20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.
31.
32.
33.
34.
35.
36.
37.
38.
39.
40.
41.
42.
43.
44.
45.
46.
47.
48.
49.
50.
51.
52.
53.
54.
55.
56.
57.
58.
59.
60.
61.
62.
63.
64.
65.
66.
67.
68.
69.
70.
71.
72.
73.
74.
75.
76.
77.
78.
79.
80.
81.
82.
83.
84.
85.
86.
87.
88.
89.
90.
91.
92.
93.
94.
95.
96.
97.
98.
99.
100.
```

```
> tail(data_train)
      time avg tmin tmax prcp
9491 2015-12-26 25.8 21.3    31  NA
9492 2015-12-27 26.7 23.0    31   0
9493 2015-12-28 26.3 22.8    31  NA
9494 2015-12-29 26.1 21.3    31  NA
9495 2015-12-30 26.1 20.3    31  NA
9496 2015-12-31 26.3 20.8    31  NA

> data_test = subset(data,time=as.Date("2015-12-31"))
> View(data_test)
```



Create your report with Rmd



(Refer Slide Time: 07:29)

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

```
14 data_read.csv(file="L1nennai_1990_c022_Madras.csv", header = 1)
15
16 str(data)
17
18 data$time=as.Date(data$time, format = "%d-%m-%Y")
19 str(data)
20 ````
21
22 ## Split the data into train and test
23 ````{r}
24 data_train = subset(data,time<=as.Date("2015-12-31"))
25 tail(data_train)
26
27 data_test = subset(data,time>as.Date("2015-12-31"))
28 head(data_test)
```

The console window shows the execution of the following command:

```
/Applications/RStudio.app/Contents/MacOS/quarto/bin/tools/pandoc +RTS -K512m  
-RTS Hands_on_Lecture_04.knit.md --to latex --from markdown+autolink_bare_ur  
is+tex_math_single_backslash --output Hands_on_Lecture_04.tex --lua-filter /L  
ibrary/Frameworks/R.framework/Versions/4.2-arm64/Resources/Library/markdown/  
rmarkdown/pagebreak.lua --lua-filter /Library/Frameworks/R.framework/Vers  
ions/4.2-arm64/Resources/Library/markdown/markdown/latex-div.lua --self  
-contained --highlight-style tango --pdf-engine pdflatex --variable graphics  
--variable 'geometry:margin=1in'  
output file: Hands_on_Lecture_04.knit.md
```

A warning message is displayed at the bottom of the console: `[WARNING] Deprecated: --self-contained. use --embed-resources --standalone`

The file explorer on the right shows a directory structure with files including `Chennai_1990_2022_Madras.csv`, `Chennai-Climate.jpg`, and several `Reg_Class_Lect_4_Part_*.pdf` files.



So, this is very important. When you are doing analysis you create the report as well whatever you are doing. So, this is a very handy thing in R. So, we load the data and then split the data into train and test R. So, ok we can just, we can just write it in this way and head data test.

(Refer Slide Time: 08:12)

Hands on for Lecture 4

Sourish

Read the data

```
dataread.csv(file="Chennai_1990_2022_hadras.csv",header = T)
str(data)
```

```
## 'data.frame':  11894 obs. of  5 variables:
## $ time: chr  "01-01-1990" "02-01-1990" "03-01-1990" "04-01-1990" ...
## $ tang: num  25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ tmin: num  22.8 21.7 21.4 20.7 20.7 20.3 22 21.7 20.7 ...
## $ tmax: num  28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
## $ prep: num  0.5 0 0 0 0.5 18 0.5 0 0 ...
```



```
data$time<-as.Date(data$time,format="'%d-%m-%Y'")
str(data)
```

```
## 'data.frame':  11894 obs. of  5 variables:
## $ time: Date, format: "1990-01-01" "1990-01-02" ...
## $ tang: num  25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ tmin: num  22.8 21.7 21.4 20.7 20.7 20.3 22 21.7 20.7 ...
## $ tmax: num  28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
## $ prep: num  0.5 0 0 0 0.5 18 0.5 0 0 ...
```

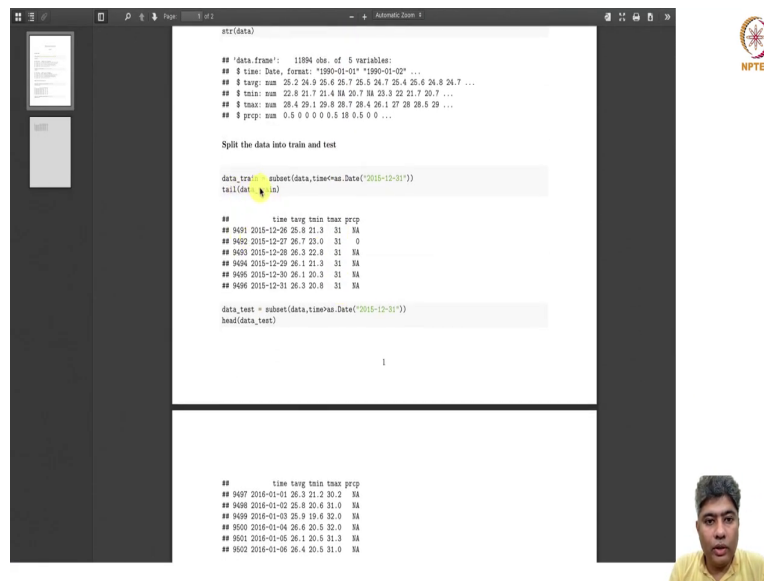
Split the data into train and test

```
data_train = subset(data,time<=as.Date("2010-12-31"))
tail(data_train)
```

```
##           time tang tmin tmax prep
## 9491 2015-12-26 25.8 21.3  31  NA
## 9492 2015-12-27 26.7 23.0  31  0
## 9493 2015-12-28 26.3 22.8  31  NA
## 9494 2015-12-29 26.1 21.3  31  NA
## 9495 2015-12-30 26.1 20.3  31  NA
## 9496 2015-12-31 26.3 20.8  31  NA
```



(Refer Slide Time: 08:15)



```
str(data)
## 'data.frame': 11894 obs. of 5 variables:
## $ time: Date, format: "1890-01-01" "1890-01-02" ...
## $ tang: num 25.2 24.9 25.6 25.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ tsun: num 22.8 21.7 21.4 18.8 20.7 18 20.3 22 21.7 20.7 ...
## $ tsax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
## $ prcp: num 0.5 0 0 0 0.5 18 0.5 0 0 ...

Split the data into train and test

data_train = subset(data, time<=as.Date("2015-12-31"))
tail(data_train)

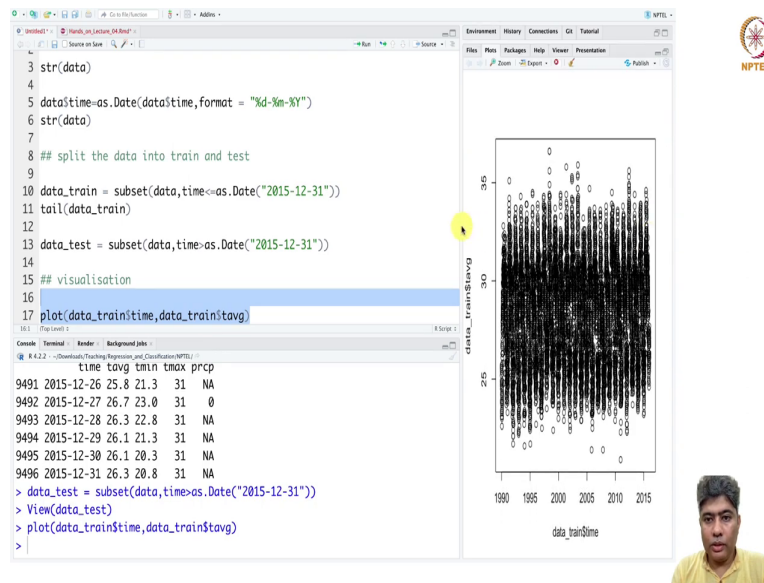
##           time tang tsun tsax prcp
## 9491 2015-12-26 25.8 21.3  31  NA
## 9492 2015-12-27 26.7 23.0  31  0
## 9493 2015-12-28 26.3 22.8  31  NA
## 9494 2015-12-29 26.1 21.3  31  NA
## 9495 2015-12-30 26.1 20.3  31  NA
## 9496 2015-12-31 26.3 20.8  31  NA

data_test = subset(data, time>=as.Date("2016-01-01"))
head(data_test)

##           time tang tsun tsax prcp
## 9497 2016-01-01 26.3 21.2 30.2  NA
## 9498 2016-01-02 25.8 20.6 31.0  NA
## 9499 2016-01-03 25.0 19.6 32.0  NA
## 9500 2016-01-04 26.6 20.5 32.0  NA
## 9501 2016-01-05 26.1 20.5 31.3  NA
## 9502 2016-01-06 26.4 20.5 31.0  NA
```

So, here is the splitting the data. We have trained data, we have defined, and here is the test data. Now, what we will do? We will do some visualization. Let us go there and in the main R script, let us do some visualisation, ok.

(Refer Slide Time: 08:30)





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

```
3 str(data)
4
5 data$time=as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 ## split the data into train and test
9
10 data_train = subset(data,time<=as.Date("2015-12-31"))
11 tail(data_train)
12
13 data_test = subset(data,time>as.Date("2015-12-31"))
14
15 ## visualisation
16
17 plot(data_train$time,data_train$tavg)
```

The console window shows the output of the code, including the structure of the data and the tail of the training data:

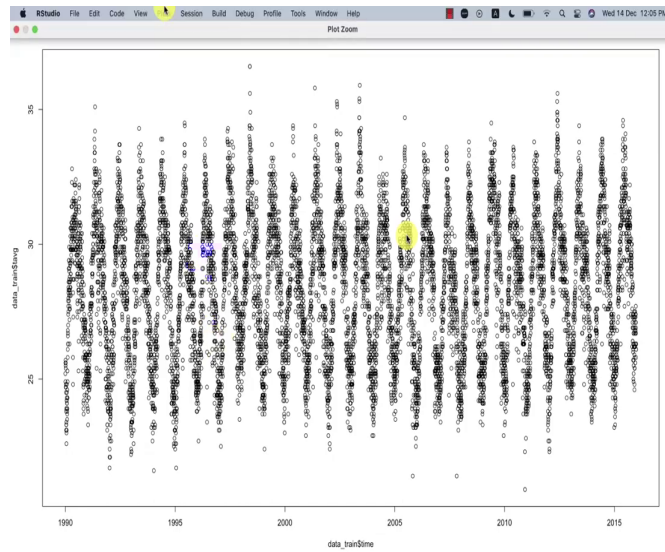
```
time tavg tmin tmax prcp
9491 2015-12-26 25.8 21.3 31 NA
9492 2015-12-27 26.7 23.0 31 0
9493 2015-12-28 26.3 22.8 31 NA
9494 2015-12-29 26.1 21.3 31 NA
9495 2015-12-30 26.1 20.3 31 NA
9496 2015-12-31 26.3 20.8 31 NA
> data_test = subset(data,time>as.Date("2015-12-31"))
> View(data_test)
> plot(data_train$time,data_train$tavg)
>
```

The plot window on the right shows a scatter plot titled "data_train\$time" vs "data_train\$tavg". The x-axis is labeled "data_train\$time" and ranges from 1990 to 2015. The y-axis is labeled "data_train\$tavg" and ranges from 25 to 35. The plot shows a dense cloud of points, with a yellow circle highlighting a specific point at approximately (2015-12-31, 26.3).



In the visualization, what we will do? We will take the from the train data, we will take the time on the x axis and data train as the average time, we will plot them.

(Refer Slide Time: 09:02)



So, you can see. So, this is how it looks like somewhat sinusoidal. So, what we will do? We will put a pch equal to 20, alright, yeah. It looks like somewhat sinusoidal. Every annual behavior is sort of there.

(Refer Slide Time: 09:25)

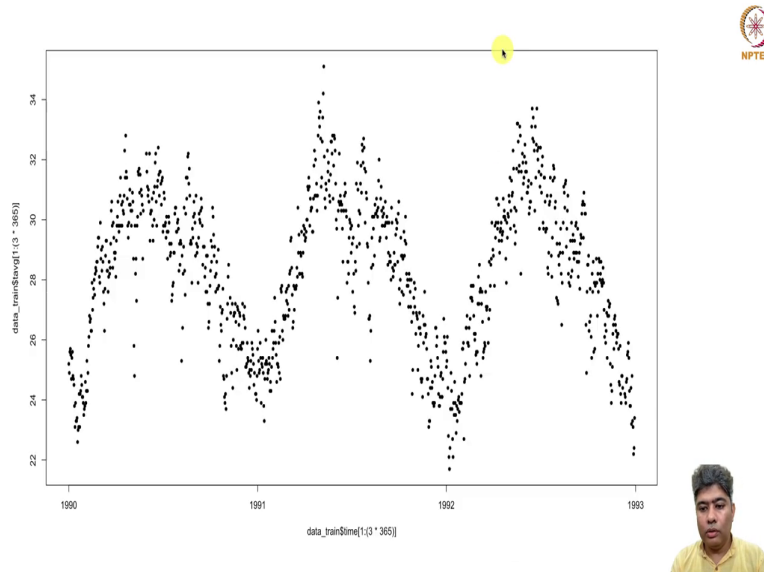
```
4
5 data$time<-as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 ## split the data into train and test
9
10 data_train = subset(data,time<=as.Date("2015-12-31"))
11 tail(data_train)
12
13 data_test = subset(data,time>as.Date("2015-12-31"))
14
15 ## visualisation
16
17 plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)]
18       ,pch=20)
```

Index	Date	avg	std	temp	
9495	2015-12-28	26.3	22.8	31	NA
9494	2015-12-29	26.1	21.3	31	NA
9495	2015-12-30	26.1	20.3	31	NA
9496	2015-12-31	26.3	20.8	31	NA

> data_test = subset(data,time>as.Date("2015-12-31"))
> View(data_test)
> plot(data_train\$time,data_train\$avg)
> plot(data_train\$time,data_train\$avg,pch=20)
> plot(data_train\$time[1:(3*365)],data_train\$avg[1:(3*365)]
+ ,pch=20)
>

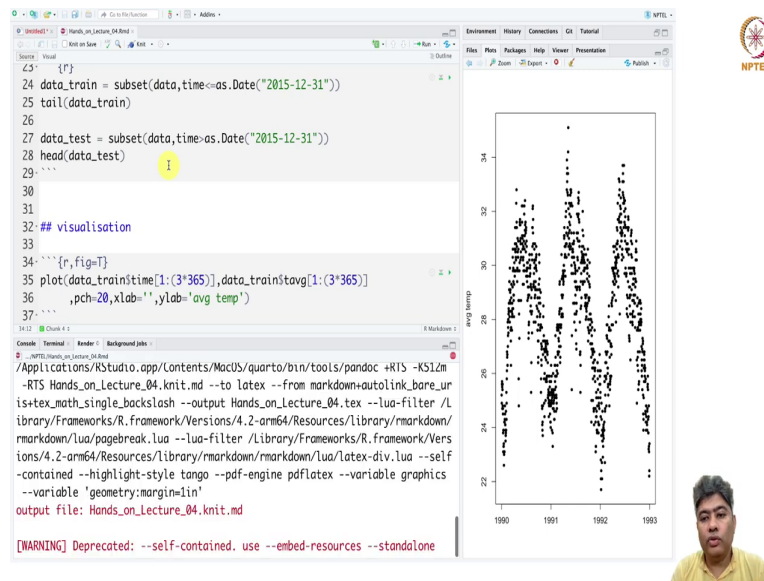
So, in order to just make sure, if it is truly that, we can (Refer Time: 09:28) is 2 into maybe 3 years of data we can plot, 3 into 365, yeah.

(Refer Slide Time: 09:44)



So, this is from 1990-91 is one cycle, 91 to 92 is another cycle, and 92 to 93 is another, 3 cycles, 3 years of data, and it is a sinusoidal behavior as expected.

(Refer Slide Time: 10:14)



The screenshot displays the RStudio interface. The left pane shows R code for data manipulation and visualization. The right pane shows a scatter plot of average temperature over time.

```
24 data_train = subset(data,time<=as.Date("2015-12-31"))
25 tail(data_train)
26
27 data_test = subset(data,time>as.Date("2015-12-31"))
28 head(data_test)
29 ...
30
31
32 ## visualisation
33
34 ...{r,fig=T}
35 plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)]
36       ,pch=20,xlab='',ylab='avg temp')
37 ...
```

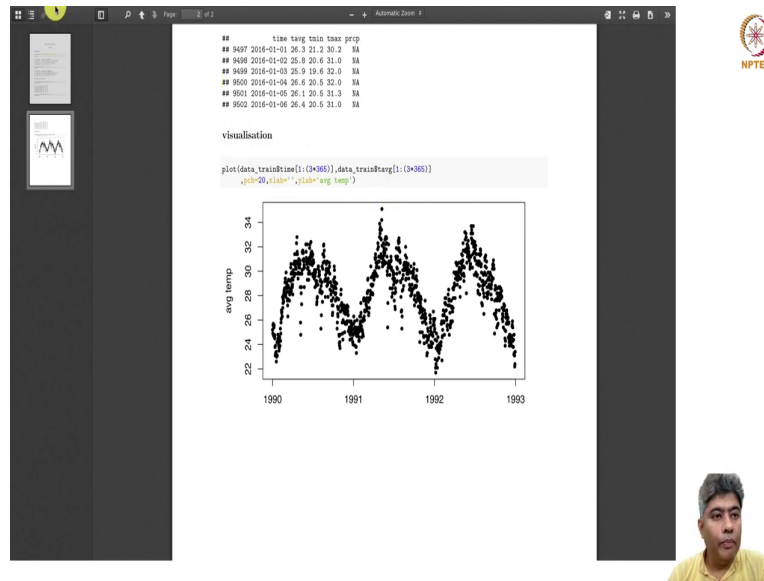
The scatter plot shows 'avg temp' on the y-axis (ranging from 22 to 34) and time on the x-axis (ranging from 1990 to 1993). The data points show a clear seasonal pattern with peaks around 34 and troughs around 22.

The console output shows the execution of the R code and the generation of the plot. A warning message is displayed at the bottom: [WARNING] Deprecated: --self-contained. use --embed-resources --standalone

Now, what we have is if you look into the plot the y lab has is bit, scriptic and x lab is also bit scriptic. What we can do? We can in xlab, we do not need to give anything some sort of self-explanatory, in the ylab in the y label, the label of y needs to be given is a average temperature, ok. So, this is average temperature.

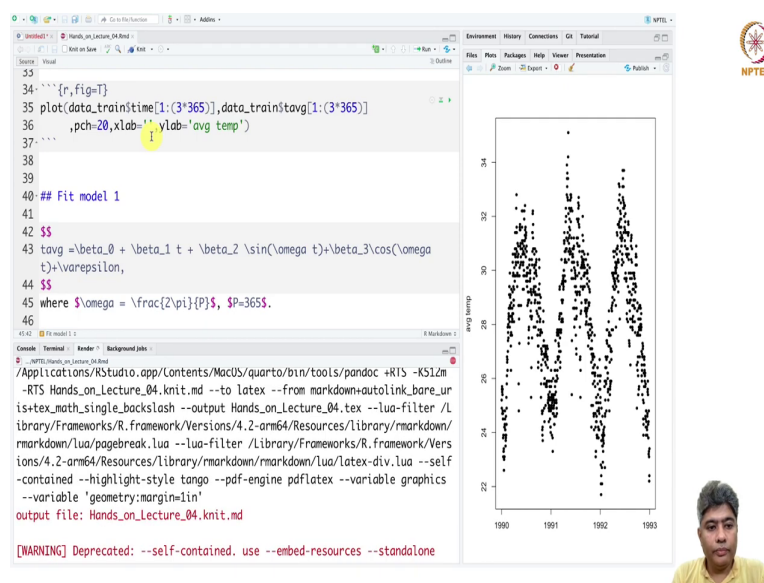
So, we will try to fit some kind of sinusoidal model in the for the average temperature. So, we can put it in R and in R markdown, we have to just say R. And here you have to say figure, fig equal true. So, that it will produce the figure as you wish.

(Refer Slide Time: 11:09)



So, now you can see it produces the figure as expected. If you do not give this figure equal to true, then it will not in (Refer Time: 11:23) reproduce the figure, ok.

(Refer Slide Time: 11:35)





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

```
34. ` ` ` [r,fig=T]
35 plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)]
36 ,pch=20,xlab='',ylab='avg temp')
37. ` ` `
38
39
40 ## Fit model 1
41
42 $$
43 tavg = \beta_0 + \beta_1 t + \beta_2 \sin(\omega t) + \beta_3 \cos(\omega t) + \epsilon,
44 $$
45 where $\omega = \frac{2\pi}{P}$, $P=365$.
46
```

The console window shows the execution of the following command:

```
/Applications/RStudio.app/Contents/MacOS/quarto/bin/tools/pandoc +RTS -K512m
-RTS Hands_on_Lecture_04.knit.md --to latex --from markdown+autolink_bare_ur
is+tex_math_single_backslash --output Hands_on_Lecture_04.tex --lua-filter /L
ibrary/Frameworks/R.framework/Versions/4.2-arm64/Resources/library/markdown/
rmarkdown/pagebreak.lua --lua-filter /Library/Frameworks/R.framework/Vers
ions/4.2-arm64/Resources/library/markdown/rmarkdown/latex-div.lua --self
-contained --highlight-style tango --pdf-engine pdflatex --variable graphics
--variable 'geometry:margin=1in'
output file: Hands_on_Lecture_04.knit.md
[WARNING] Deprecated: --self-contained. use --embed-resources --standalone
```

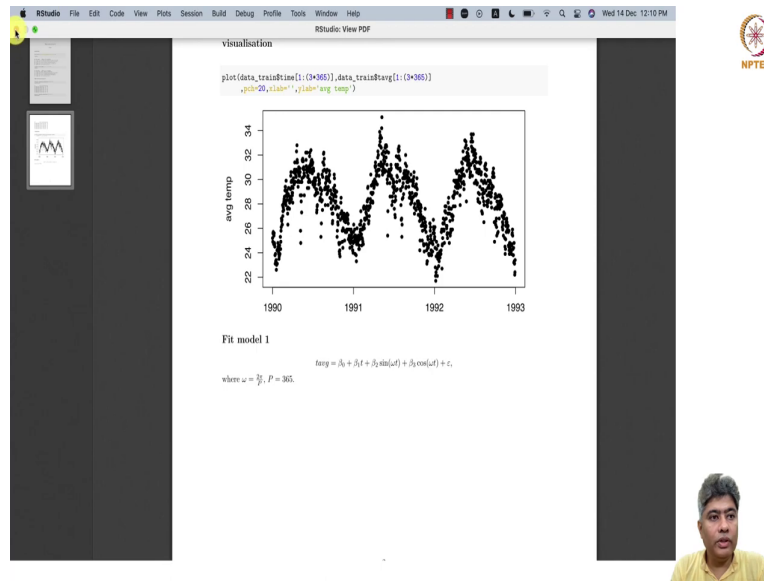
The plot window shows a scatter plot of 'avg temp' versus time (years). The x-axis ranges from 1990 to 1993, and the y-axis ranges from 22 to 34. The data points show a clear seasonal pattern with peaks around 34 and troughs around 22.



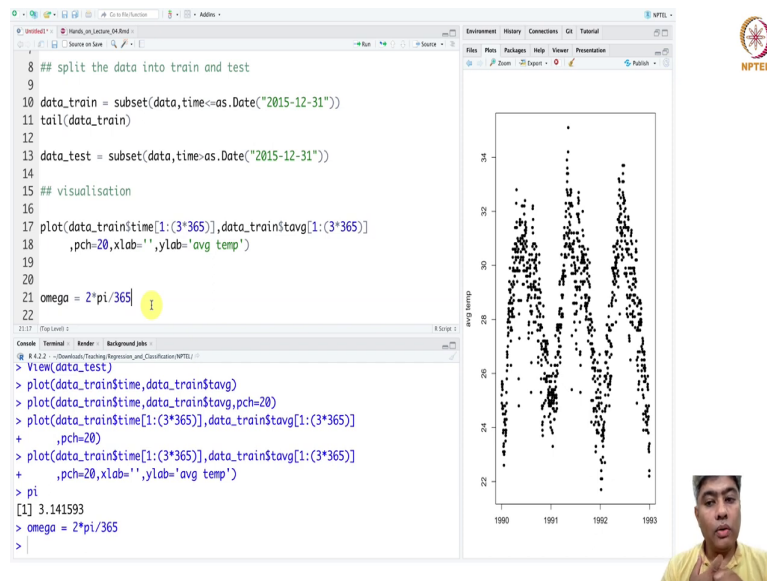
Next, we want to fit a model. So, what model we want to fit? Let us first fit a model, fit model, first model that we want to fit is some kind of sinusoidal. So, we can use in R markdown t average equals to β_0 plus $\beta_1 t$ plus $\beta_2 \sin(\omega t)$ plus $\beta_3 \cos(\omega t)$ plus some error, ok.

This is the model that we want to fit. But we have to at the same time we have to say something about the ω . So, ω , where ω is equal to $\frac{2\pi}{P}$ of p naught. This p is 365, this p is our daily data is 365, ok. So, this is the model that we want to fit, ok.

(Refer Slide Time: 13:56)



(Refer Slide Time: 14:04)



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

```
8 ## split the data into train and test
9
10 data_train = subset(data,time<=as.Date("2015-12-31"))
11 tail(data_train)
12
13 data_test = subset(data,time>as.Date("2015-12-31"))
14
15 ## visualisation
16
17 plot(data_train$time[1:(3*365)],data_train$avg[,1:(3*365)])
18 ,pch=20,xlab='',ylab='avg temp')
19
20
21 omega = 2*pi/365
22
```

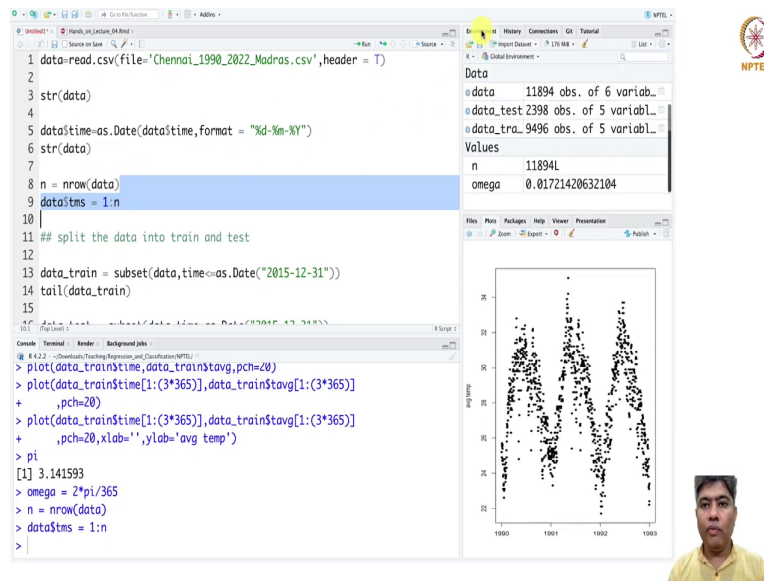
The console on the left shows the execution of the following commands:

```
> plot(data_train$time,data_train$avg)
> plot(data_train$time,data_train$avg,pch=20)
> plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)])
+ ,pch=20)
> plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)])
+ ,pch=20,xlab='',ylab='avg temp')
> pi
[1] 3.141593
> omega = 2*pi/365
>
```

The plot on the right is a scatter plot of average temperature (avg temp) over time. The x-axis represents years from 1990 to 1993, and the y-axis represents average temperature from 22 to 34. The data points show a clear seasonal pattern with annual oscillations. A small video inset of a person in a yellow shirt is visible in the bottom right corner of the RStudio window.

So, this is the model we want to fit. So, first thing we have to do is we have to first thing is omega equal to 2π by 365, π itself is defined as 3.141. So, omega would be like this. Now, we have to have, if I go there I have to have a time to be defined. So, what I will do is here and this time component, we will require both in train and test data set.

(Refer Slide Time: 14:54)



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

```
1 data-read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
2
3 str(data)
4
5 data$time=as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 n = nrow(data)
9 data$tms = 1:n
10
11 # split the data into train and test
12
13 data_train = subset(data,time<=as.Date("2015-12-31"))
14 tail(data_train)
15
```

The console window on the bottom left shows the following output:

```
> pi
[1] 3.141593
> omega = 2*pi/365
> n = nrow(data)
> data$tms = 1:n
>
```



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

Object	Details
data	11894 obs. of 6 variables
data_test	2398 obs. of 5 variables
data_train	9496 obs. of 5 variables

The Values pane shows the following values:

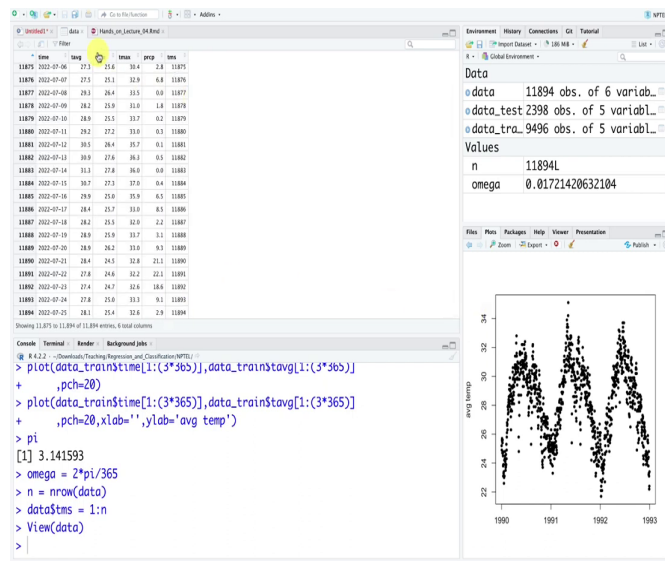
Variable	Value
n	11894
omega	0.01721420632104

The plot window on the right shows a scatter plot of average temperature (avg temp) over time (year). The x-axis ranges from 1990 to 1993, and the y-axis ranges from 24 to 28. The plot shows a clear seasonal pattern with temperature peaking around 28 in the summer and dipping around 24 in the winter.



So, what I am going to do is here before the training and test we are going to, I am going to tms, a new variable which is 1 is to n, where n is n row of data. Now, the problem will be if you take a look into the data tms, it is starting from 1 and it is going into 11894.

(Refer Slide Time: 15:31)



The screenshot displays the RStudio interface. The top-left pane shows a data table with columns 'time', 'avg', 'max', 'prec', and 'tms'. The top-right pane shows the 'Data' tab with summary statistics for 'data' (11894 obs. of 6 variables) and 'data_test' (2398 obs. of 5 variables). The bottom-left pane shows the R console with the following code and output:

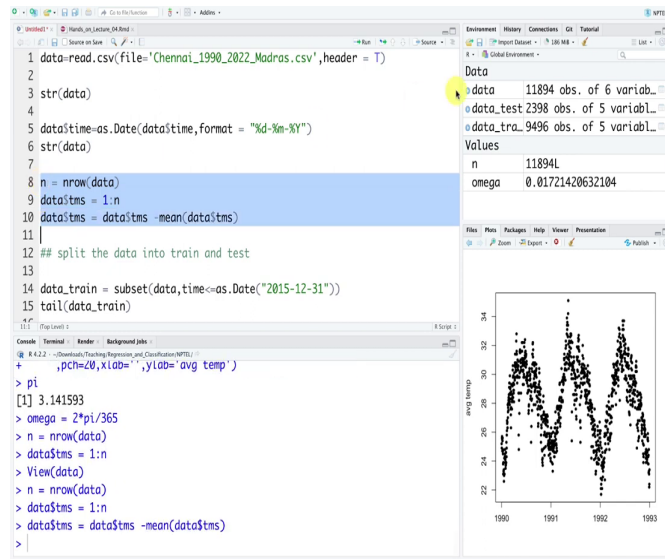
```
> plot(data_train$time[1:(3*365)], data_train$avg[1:(3*365)])
+
> plot(data_train$time[1:(3*365)], data_train$avg[1:(3*365)])
+
+ ,pch=20,xlab='', ylab='avg temp')
> pi
[1] 3.141593
> omega = 2*pi/365
> n = nrow(data)
> data$tms = 1:n
> View(data)
>
```

The bottom-right pane shows a scatter plot of 'avg temp' versus time from 1990 to 1993. The plot shows a clear seasonal pattern with two peaks and two troughs. The y-axis ranges from 22 to 34, and the x-axis ranges from 1990 to 1993. The data points are represented by small black circles.



So, it is better, if we just enter the time, it will just handling the data will be better, the coefficient will be little bit meaningful.

(Refer Slide Time: 15:48)



The screenshot displays the RStudio interface with the following components:

- Script Editor:** Contains R code for loading data from a CSV file, converting it to a date-time object, and splitting it into training and testing sets. The code includes:

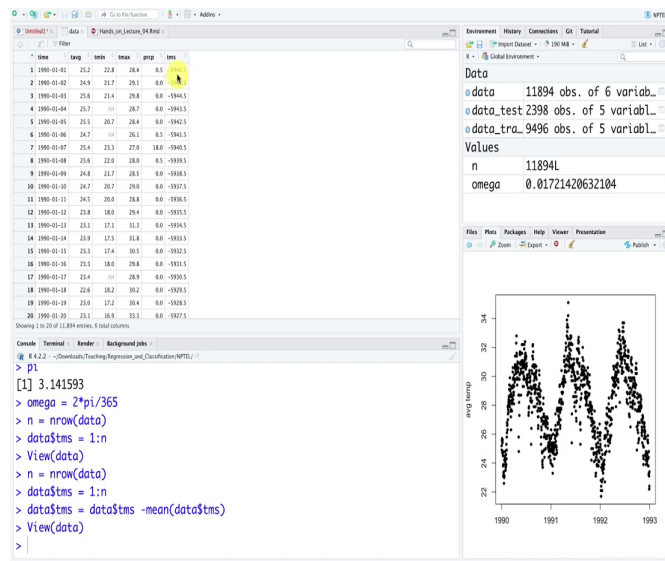
```
1 data-read.csv(file='Chennai_1990_2022_Madras.csv',header = T)
2
3 str(data)
4
5 data$time=as.Date(data$time,format = "%d-%m-%Y")
6 str(data)
7
8 n = nrow(data)
9 data$time = 1:n
10 data$time = data$time - mean(data$time)
11 |
12 ## split the data into train and test
13
14 data_train = subset(data,time<=as.Date("2015-12-31"))
15 tail(data_train)
```
- Console:** Shows the execution of the following commands and their outputs:

```
> pi
[1] 3.141593
> omega = 2*pi/365
> n = nrow(data)
> data$time = 1:n
> View(data)
> n = nrow(data)
> data$time = 1:n
> data$time = data$time - mean(data$time)
>
```
- Data Viewer:** Displays summary statistics for the 'data' object:

Values	
n	11894L
omega	0.01721420632104
- Plot:** A scatter plot titled 'avg temp' showing the relationship between time (x-axis, years 1990-1993) and average temperature (y-axis, 22-34). The plot shows a clear seasonal cycle with peaks in summer and troughs in winter.



(Refer Slide Time: 16:05)



The screenshot displays the RStudio interface. The top-left pane shows a data frame with columns: time, temp, min, max, precip, and rms. The data spans from 1990-01-01 to 1990-01-20. The top-right pane shows the 'Data' tab with a list of objects: data (11894 obs. of 6 variables), data_test (2398 obs. of 5 variables), and data_tra (9496 obs. of 5 variables). The 'Values' section shows n = 11894 and omega = 0.01721420632104. The bottom-left pane shows the R console with the following code and output:

```
[1] 3.141593
> omega = 2*pi/365
> n = nrow(data)
> data$time = 1:n
> View(data)
> n = nrow(data)
> data$time = 1:n
> data$time = data$time - mean(data$time)
> View(data)
>
```

The bottom-right pane shows a scatter plot of 'avg temp' (y-axis, 22 to 34) versus 'year' (x-axis, 1990 to 1993). The plot shows a clear seasonal cycle with peaks in 1991 and 1992 and troughs in 1990 and 1993.



So, what we will do? We just subtract the mean of these guys, ok. So, if I do that, then it is starting from minus 5946.5 and it is going ending at 5946.5. So, it is just shifting the horizon, shifting the time from you know centering at 0. Now, if I just run the whole thing, it will be no problem.

(Refer Slide Time: 16:50)

Sourish

Read the data

```
dataread.csv(file="Chennai_1990_2022_tms.csv",header = T)
str(data)

## 'data.frame': 11894 obs. of 5 variables:
## $ time: chr "01-01-1990" "02-01-1990" "03-01-1990" "04-01-1990" ...
## $ targ: num 25.2 24.9 25.6 26.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ tms: num 22.8 21.7 21.4 NA 20.7 NA 20.3 22 21.7 20.7 ...
## $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
## $ prep: num 0.5 0 0 0 0.5 18 0.5 0 0 ...

data$Time = as.Date(data$Time,format = "%d-%m-%Y")
str(data)

## 'data.frame': 11894 obs. of 5 variables:
## $ time: Date, format: "1990-01-01" "1990-01-02" ...
## $ targ: num 25.2 24.9 25.6 26.7 25.5 24.7 25.4 25.6 24.8 24.7 ...
## $ tms: num 22.8 21.7 21.4 NA 20.7 NA 20.3 22 21.7 20.7 ...
## $ tmax: num 28.4 29.1 29.8 28.7 28.4 26.1 27 28 28.5 29 ...
## $ prep: num 0.5 0 0 0 0.5 18 0.5 0 0 ...

n = nrow(data)
data$Time = 1:n
data$Time = data$Time - mean(data$Time)

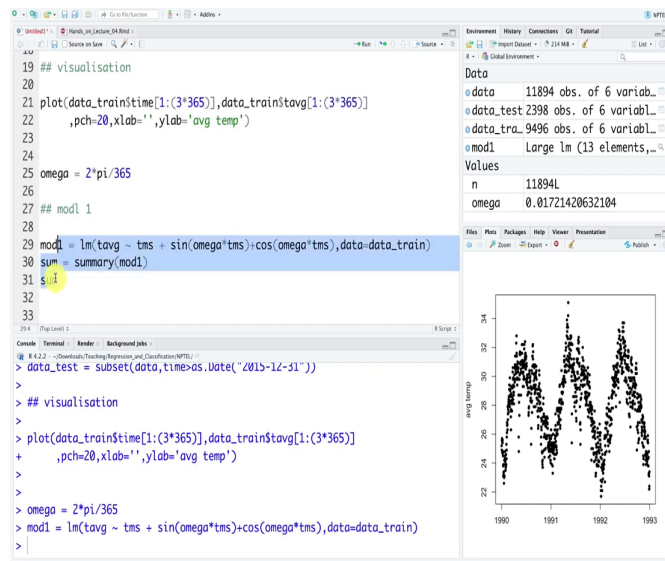
Split the data into train and test

data_train = subset(data,time<=as.Date("2015-12-31"))
tail(data_train)

##           time targ tmin tmax prep  tms
## 9491 2015-12-26 25.8 21.3  31  NA 3543.5
## 9492 2015-12-27 26.7 23.0  31  0 3544.5
## 9493 2015-12-28 26.3 22.8  31  NA 3545.5
## 9494 2015-12-29 26.1 21.3  31  NA 3546.5
## 9495 2015-12-30 26.1 20.3  31  NA 3547.5
## 9496 2015-12-31 25.3 20.8  31  NA 3548.5
```

So, we have to add this part in the markdown file, in the beginning. And now if I just run it, you can see the tms has taken care of, alright.

(Refer Slide Time: 17:02)



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

```
19 ## visualisation
20
21 plot(data_train$time[1:(3*365)], data_train$avg[1:(3*365)])
22     ,pch=20,xlab='', ylab='avg temp')
23
24
25 omega = 2*pi/365
26
27 ## mod 1
28
29 mod1 = lm(tavg ~ tms + sin(omega*tms) + cos(omega*tms), data=data_train)
30 sum = summary(mod1)
31 s[[i]]
32
33
```

The console window shows the execution of the following commands:

```
> data_test = subset(data, time>as.Date("2015-12-31"))
>
> ## visualisation
> plot(data_train$time[1:(3*365)], data_train$avg[1:(3*365)])
+   ,pch=20,xlab='', ylab='avg temp')
>
> omega = 2*pi/365
> mod1 = lm(tavg ~ tms + sin(omega*tms) + cos(omega*tms), data=data_train)
>
```

The Environment pane on the right shows the following objects:

Object	Class	Attributes
data	data.frame	11894 obs. of 6 variables
data_test	data.frame	2398 obs. of 6 variables
data_train	data.frame	9496 obs. of 6 variables
mod1	lm	Large lm (13 elements)

The Values pane shows the following values:

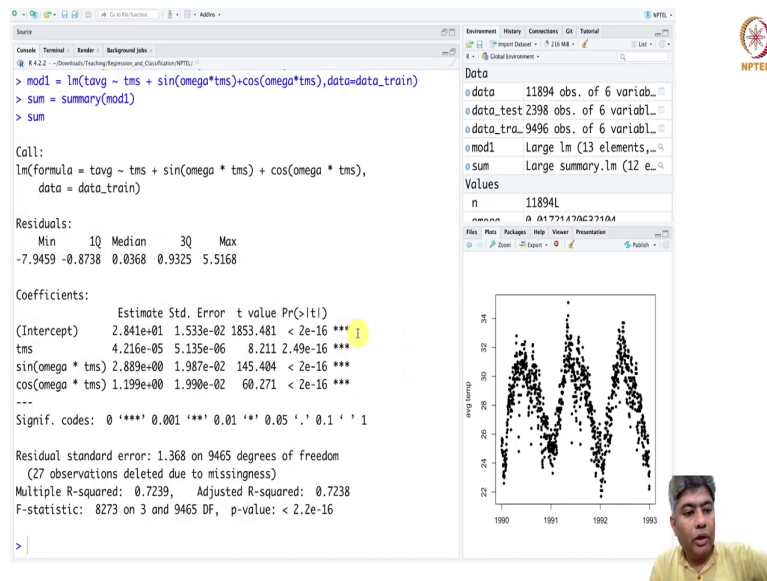
Variable	Value
n	11894L
omega	0.01721420632104

The plot window shows a scatter plot of average temperature (avg temp) over time (time). The x-axis ranges from 1990 to 1993, and the y-axis ranges from 22 to 34. The plot shows a clear seasonal pattern with two peaks and two troughs.



Next, what we will do? We will fit the model, first model. Model point mod 1 tms plus sin omega (Refer Time: 17:25). Now, we have to provide data equal to data train. And if we just put summary.

(Refer Slide Time: 18:11)

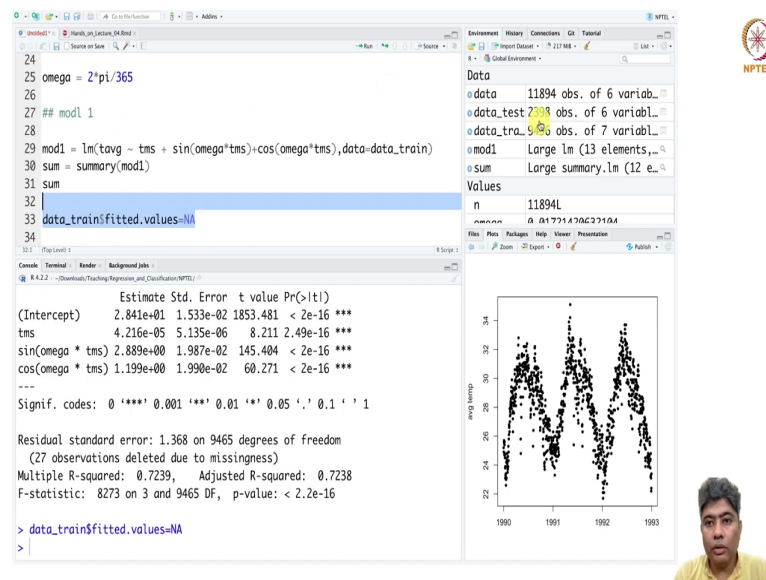


Now, this is the first model to fit. Now, what are the; let us spend understanding what; spend some time to understand what is going on here The first is intercept of the model that is beta naught and then second one is the this is the coefficient corresponding to t. So, if we look into the markdown file, so beta naught plus beta 1 t. So, this beta naught beta 1 times t is, this is the standard error, this is the t value and the p value is really small.

So, that means, with increasing time the average temperature in Chennai is increasing. This is sin omega t which has a significant effect, this is cos omega t, the this is the coefficient of the; so, beta 3, this is beta 2, and this is beta 3 and both of them are significant. Of course, sin cosine will have a significant effect, but the most important thing is trained is significant and it is positive, it is very small, it is very small point after 50421, so it is increasing at a very slow rate.

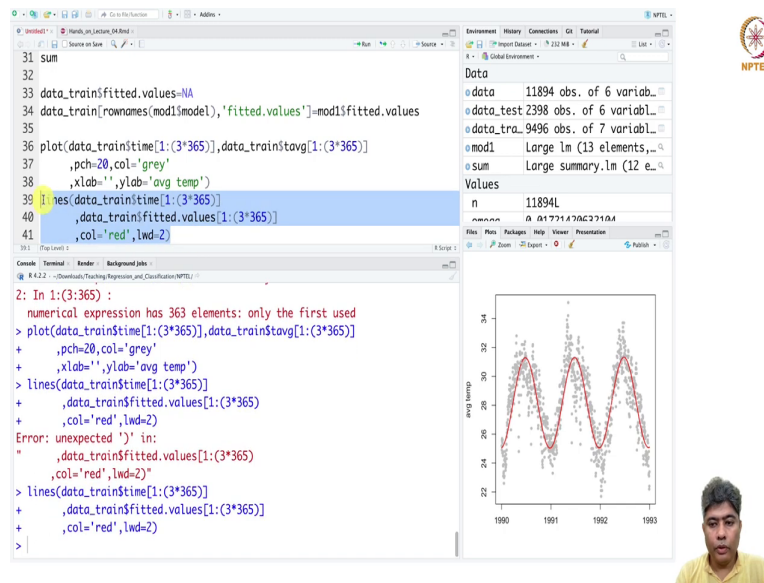
But we cannot ignore it. It is small enough, it is significant enough and it is saying that it is increasing over time.

(Refer Slide Time: 20:06)



Now, what we are going to do is we are going to plot the model, we have to see just how the fitting is working or not. So, before plot, what we will do? We will in the data train, what we will do we will just take the fitted values equal to NA. We are creating a new column with all the NAs.

(Refer Slide Time: 20:41)



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

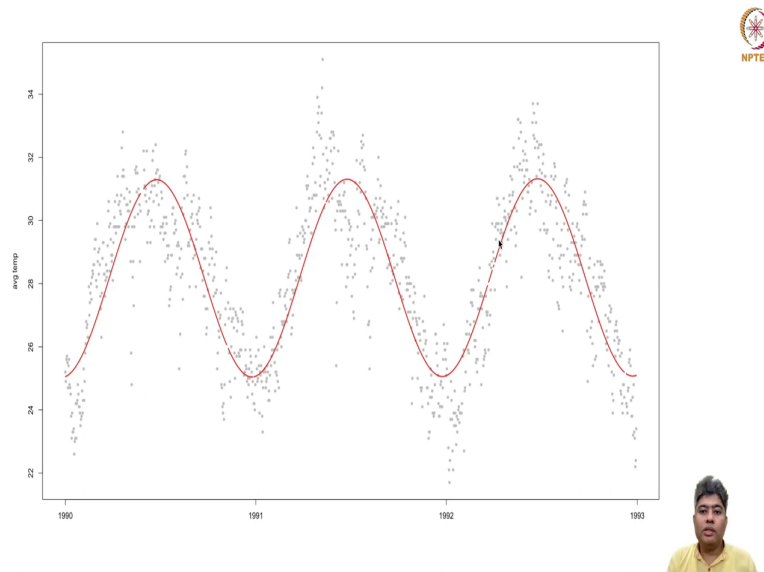
```
31 sum
32
33 data_train$fitted.values=NA
34 data_train[rownames(mod1$model),'fitted.values']=mod1$fitted.values
35
36 plot(data_train$time[1:(3*365)],data_train$avg[1:(3*365)]
37       ,pch=20,col='grey'
38       ,xlab='',ylab='avg temp')
39 lines(data_train$time[1:(3*365)]
40       ,data_train$fitted.values[1:(3*365)]
41       ,col='red',lwd=2)
```

The console shows the execution of the code, with an error message: "Error: unexpected ')' in: ",data_train\$fitted.values[1:(3*365)],col='red',lwd=2)". The plot on the right shows a scatter plot of average temperature (avg temp) over time (year) from 1990 to 1993. The y-axis ranges from 22 to 34, and the x-axis ranges from 1990 to 1993. The data points are grey dots, and a red line with a width of 2 is overlaid, showing a clear seasonal pattern with peaks around 32 and troughs around 24.

And then, data train sorry rownames and then model (Refer Time: 20:55) model. So, we have taken the, from the model we have taken the fitted values and we put it in the train data set. We extracted those values and; so, now, we can plot the train dollar, train data dollar, t average pch equal to 20, color equal to grey, x labels we can keep it empty, y lab equal to average temperature.

First, we are taking these, well we can take the first few years to just understand what is going on. First 3 years and then lines, we can just take this comma; instead of t average I want to fitted values, because color equal to say may be red, line width equal to 2. Now, we just, I have not close this here, yeah.

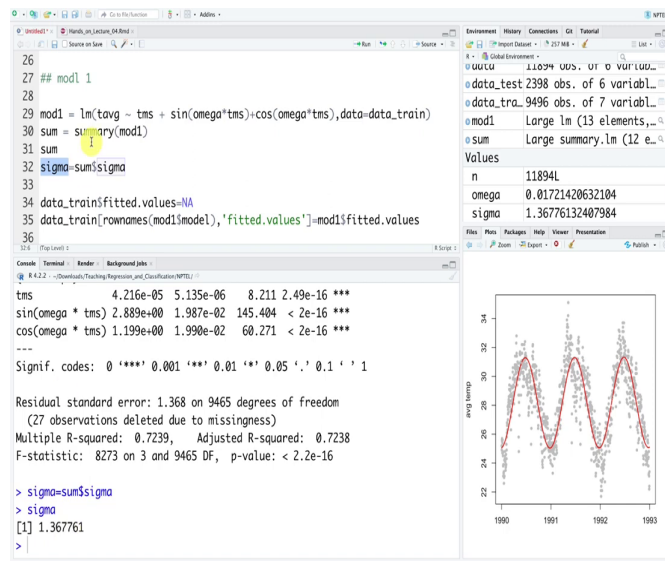
(Refer Slide Time: 23:48)



So, now if I just zoom it, you can see that the fitted model is doing very nicely between going through the sinusoidal way, going through the middle of the points. It is not doing pretty bad, but there are quite a few from far away, quite a few from bit down. So, this is maybe June, July where lot of rain happens. So, that time, during the rain the temperature drops. So, average temperature drops around that time.

Now, what we can do? We can take the from the summary; if you look into the summary this residual standard error is 1.38 which is actually the sigma.

(Refer Slide Time: 24:37)



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

```
26  
27 ## mod1  
28  
29 mod1 = lm(tavg ~ tms + sin(omega*tms)+cos(omega*tms),data=data_train)  
30 sum = summary(mod1)  
31 sum  
32 sigma=sum$sigma  
33  
34 data_train$fitted.values=NA  
35 data_train[rownames(mod1$model),'fitted.values']=mod1$fitted.values  
36
```

The console on the left shows the output of the model fit:

```
tms          4.216e-05  5.135e-06  8.211 2.49e-16 ***  
sin(omega * tms) 2.889e+00  1.987e-02 145.404 < 2e-16 ***  
cos(omega * tms) 1.199e+00  1.990e-02  60.271 < 2e-16 ***  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 1.368 on 9465 degrees of freedom  
(27 observations deleted due to missingness)  
Multiple R-squared:  0.7239,    Adjusted R-squared:  0.7238  
F-statistic: 8273 on 3 and 9465 DF,  p-value: < 2.2e-16  
  
> sigma=sum$sigma  
> sigma  
[1] 1.367761  
>
```

The environment pane on the right shows the loaded objects:


Object	Class	Attributes
data_test	data.frame	2398 obs. of 6 variables
data_train	data.frame	9496 obs. of 7 variables
mod1	lm	Large lm (13 elements)
sum	summary.lm	Large summary.lm (12 elements)

The plot pane on the right shows a scatter plot of 'avg temp' (y-axis, 22 to 34) versus year (x-axis, 1990 to 1993). The data points are grey dots, and a red line represents the fitted model. The plot shows a clear seasonal oscillation in temperature over the years.



So, we can take the sigma and summary we can calculate the sigma. So, this is our residual standard error 1.367.

(Refer Slide Time: 25:03)



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

```
37 plot(data_train$time[1:(3*365)], data_train$avg[1:(3*365)])
38 ,pch=20, col='grey'
39 ,xlab='', ylab='avg temp'
40 lines(data_train$time[1:(3*365)])
41 ,data_train$fitted.values[1:(3*365)]
42 ,col='red', lwd=2)
43
44 lines(data_train$time[1:(3*365)])
45 ,data_train$fitted.values[1:(3*365)]-1.96*sigma
46 ,col='brown', lwd=2, lty=2)
47
```

The console shows the following output:

```
Multiple R-squared: 0.7239, Adjusted R-squared: 0.7238
F-statistic: 8273 on 3 and 9465 DF, p-value: < 2.2e-16

> sigma=sum$sigma
> sigma
[1] 1.367761
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+ ,col='brown', lwd=2, lty=2)
Error in plot.xy(xy.coords(x, y), type = type, ...):
invalid color name 'brown'
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+ ,col='brown', lwd=2, lty=2)
>
```

The environment pane shows the following values:

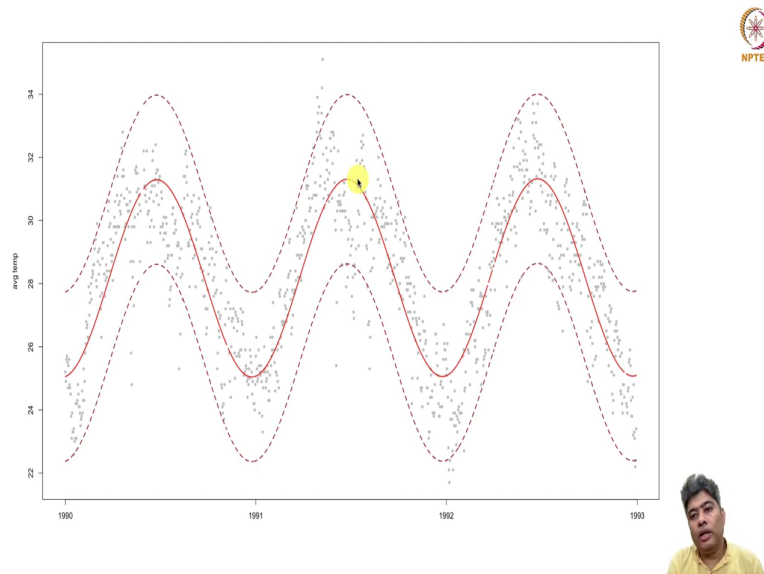
Variable	Value
n	11894
omega	0.01721420632104
sigma	1.36776132407984

The plot shows the average temperature (avg temp) over time (1990 to 1993). The data points are grey circles. A solid red line represents the fitted model. Two dashed brown lines represent the confidence bands, which are 1.96 times the standard error (sigma) away from the fitted line. The y-axis ranges from 22 to 34, and the x-axis ranges from 1990 to 1993.



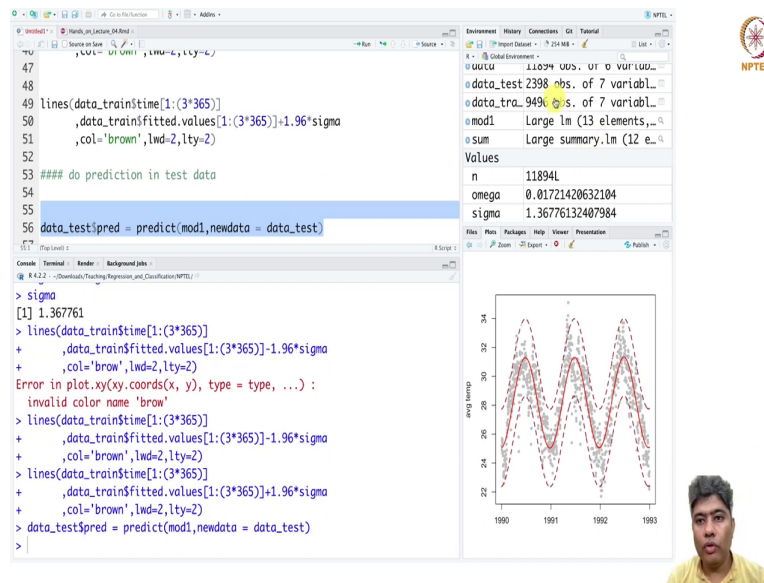
Now, what we can do? We can create a band 0.96 times sigma, if color maybe some different color, maybe brown and we take lty equal to 2. I have to take brown, yeah.

(Refer Slide Time: 25:35)



And similarly, we have this is lower bound, this gives us a lower bound you can see and similarly we can create an upper bound as well. So, now, you can see there is an upper bound, ok. And interestingly, you know looks like within the bound, all data points are nicely fitting in. So, it is a pretty decent model, it is a pretty decent model. Now, what we will do? We can just plot this. We can now fit, do the prediction in the, this fitting is inside the training data.

(Refer Slide Time: 26:17)



The screenshot displays the RStudio interface. The main editor window contains R code for model prediction. The console shows the execution of the code, including the calculation of the sigma value and the prediction of new data. A plot titled 'avg temp' is visible, showing a time series of average temperature from 1990 to 1993. The plot includes a solid red line representing the fitted model and dashed lines representing the confidence intervals. The y-axis ranges from 22 to 34, and the x-axis shows years from 1990 to 1993.

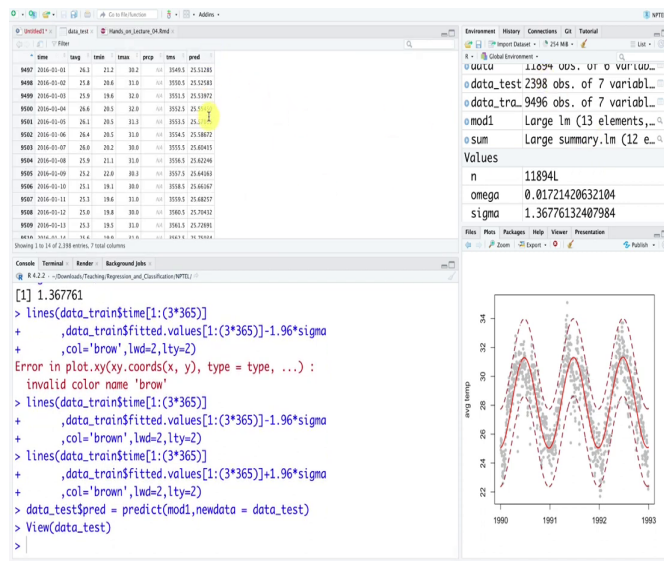
```
47 #CO2 = fitGAM1(LTMU~L,ty=C,
48
49 lines(data_train$time[1:(3*365)])
50 ,data_train$fitted.values[1:(3*365)]+1.96*sigma
51 ,col='brown',lwd=2,pty=2)
52
53 ### do prediction in test data
54
55
56 data_test$pred = predict(mod1,newdata = data_test)
```

```
> sigma
[1] 1.367761
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+ ,col='brown',lwd=2,pty=2)
Error in plot.xy(xy.coords(x, y), type = type, ...) :
  invalid color name 'brown'
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+ ,col='brown',lwd=2,pty=2)
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]+1.96*sigma
+ ,col='brown',lwd=2,pty=2)
> data_test$pred = predict(mod1,newdata = data_test)
>
```

Values	
n	11894L
omega	0.01721420632104
sigma	1.36776132407984

So, what we need to do is we need to look into the how it is doing in the test data. So, do prediction, do prediction in test data. So, what we will do? In data test dollar red equal to predict mod 1, new data equal to data test.

(Refer Slide Time: 27:05)



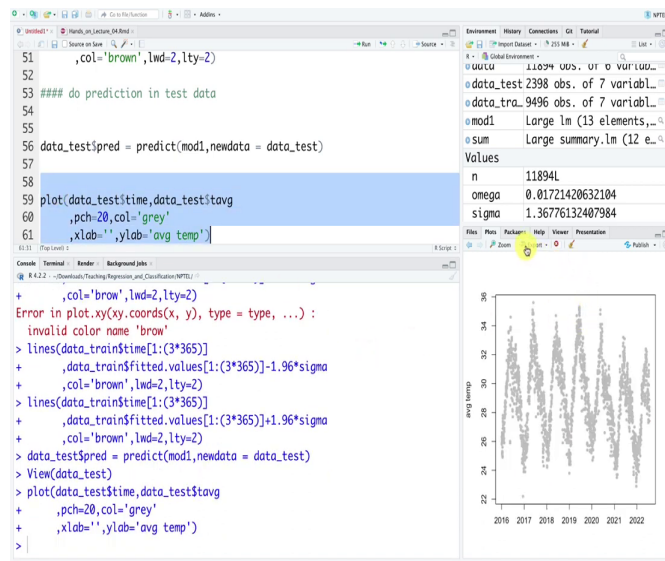
The screenshot displays the RStudio environment. The top-left pane shows a data table with columns: time, temp, min, max, precip, time, and pred. The console pane shows the following R code and output:

```
[1] 1.367761
> lines(data_train$time[1:(3*365)]
+       ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+       ,col='brown',lwd=2,pty=2)
Error in plot.xy(xy.coords(x, y), type = type, ...) :
  invalid color name 'brown'
> lines(data_train$time[1:(3*365)]
+       ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+       ,col='brown',lwd=2,pty=2)
> lines(data_train$time[1:(3*365)]
+       ,data_train$fitted.values[1:(3*365)]+1.96*sigma
+       ,col='brown',lwd=2,pty=2)
> data_test$pred = predict(mod1,newdata = data_test)
> View(data_test)
```

The top-right pane shows the environment with variables: data_test (2398 obs. of 7 variables), data_train (9496 obs. of 7 variables), mod1 (Large lm (13 elements)), and sum (Large summary.lm (12 elements)). The Values pane shows: n = 11894L, omega = 0.01721420632104, and sigma = 1.36776132407984. The bottom-right pane shows a plot of 'avg temp' from 1990 to 1993, with a red line representing the fitted model and dashed lines representing the confidence intervals.



(Refer Slide Time: 27:19)



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

```
51     ,col='brown',lwd=2,ly=2)
52
53 ### do prediction in test data
54
55
56 data_test$pred = predict(mod1,newdata = data_test)
57
58
59 plot(data_test$time,data_test$avg
60      ,pch=20,col='grey'
61      ,xlab='',ylab='avg temp')
```

The console on the bottom left shows the execution of the code, including an error message: "Error in plot.xy(xy.coords(x, y), type = type, ...) : invalid color name 'brow'". The plot on the right shows a scatter plot of 'avg temp' over time (2016-2022) with grey points and brown lines representing the model fit. The plot is zoomed in on the test data period (2016-2022).

The environment pane on the right shows the following objects:

- data_test: 2398 obs. of 7 variables
- data_train: 9496 obs. of 7 variables
- mod1: Large lm (13 elements)
- sum: Large summary.lm (12 elements)

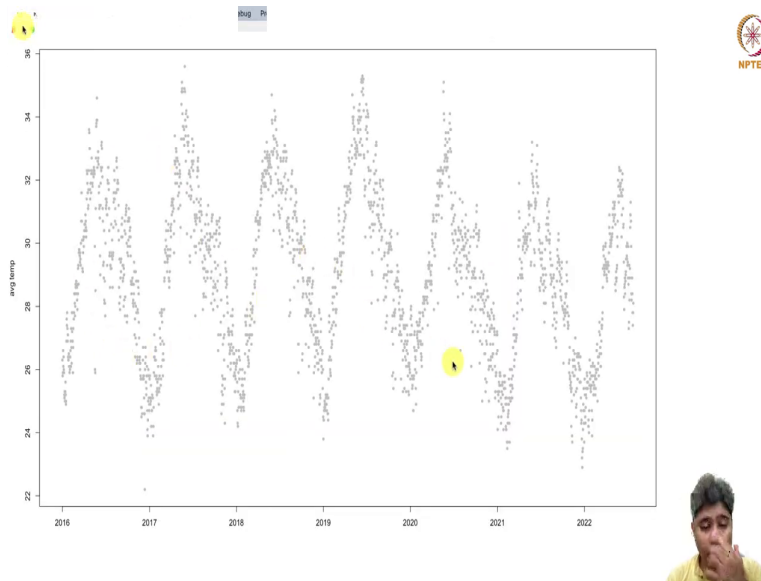
The console output shows the following commands and results:

```
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]-1.96*sigma
+ ,col='brown',lwd=2,ly=2)
> lines(data_train$time[1:(3*365)])
+ ,data_train$fitted.values[1:(3*365)]+1.96*sigma
+ ,col='brown',lwd=2,ly=2)
> data_test$pred = predict(mod1,newdata = data_test)
> View(data_test)
> plot(data_test$time,data_test$avg
+      ,pch=20,col='grey'
+      ,xlab='',ylab='avg temp')
```



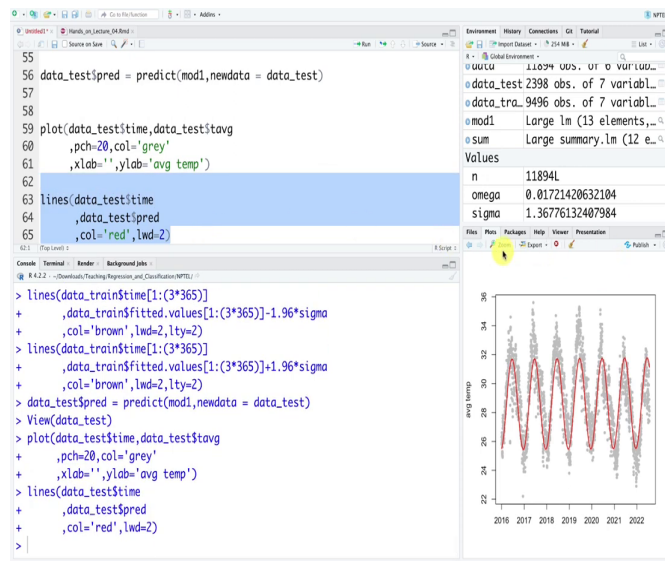
Now, if you go to the test data set, we got all the predicted values. And now similarly, we can do the plotting of first the test data set. And we do not need, we have only 5 years of data test data set, we do not need to keep it. And so, this is our test data set. Let us zoom it.

(Refer Slide Time: 27:40)



This is our original test data set. And now we can draw the lines here. So, this will be just, we have to follow it here in this way.

(Refer Slide Time: 28:01)



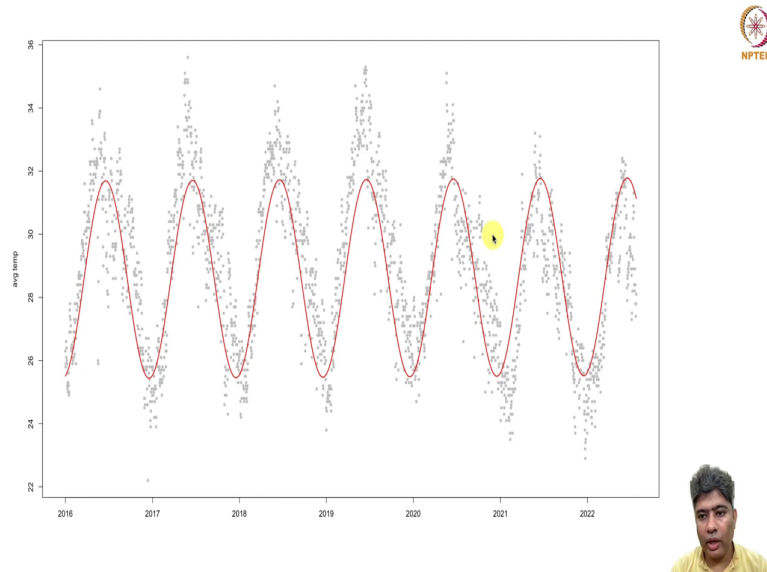
The screenshot displays the RStudio environment with the following components:

- Source Editor:** Contains R code for predicting and plotting data. Lines 63-65 are highlighted in blue.
- Environment:** Shows variables like `data_test` (2398 obs. of 7 variables), `data_tra` (9496 obs. of 7 variables), `mod1` (Large lm (13 elements, ...)), and `sum` (Large summary.lm (12 e...)).
- Values:** A table showing model parameters: `n` (11894), `omega` (0.01721420632104), and `sigma` (1.36776132407984).
- Console:** Shows the execution of the following R code:

```
> lines(data_train$time[1:(3*365)]  
+       ,data_train$fitted.values[1:(3*365)]-1.96*sigma  
+       ,col='brown',lwd=2,ly=2)  
> lines(data_train$time[1:(3*365)]  
+       ,data_train$fitted.values[1:(3*365)]+1.96*sigma  
+       ,col='brown',lwd=2,ly=2)  
> data_test$pred = predict(mod1,newdata = data_test)  
> View(data_test)  
> plot(data_test$time,data_test$avg  
+       ,pch=20,col='grey'  
+       ,xlab='',ylab='avg temp')  
> lines(data_test$time  
+       ,data_test$pred  
+       ,col='red',lwd=2)  
>
```
- Plot:** A line plot titled 'avg temp' showing average temperature from 2016 to 2022. The y-axis ranges from 22 to 36. The plot features grey points for observed data and a red line for the predicted values, with brown lines representing the 1.96-sigma confidence interval.

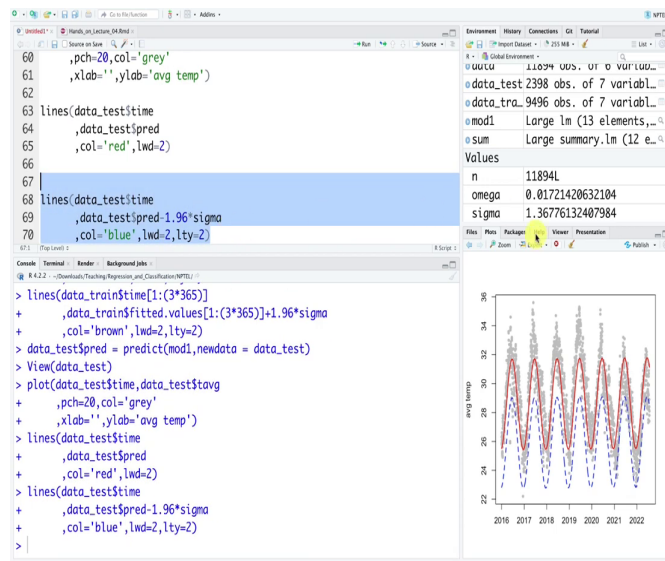


(Refer Slide Time: 28:16)



And instead of fitted values we have is pred, we do not need to keep it. So, looks like it has done ok, it has done ok, but we should do a prediction band. And let us see if it is inside the prediction band or not.

(Refer Slide Time: 28:35)



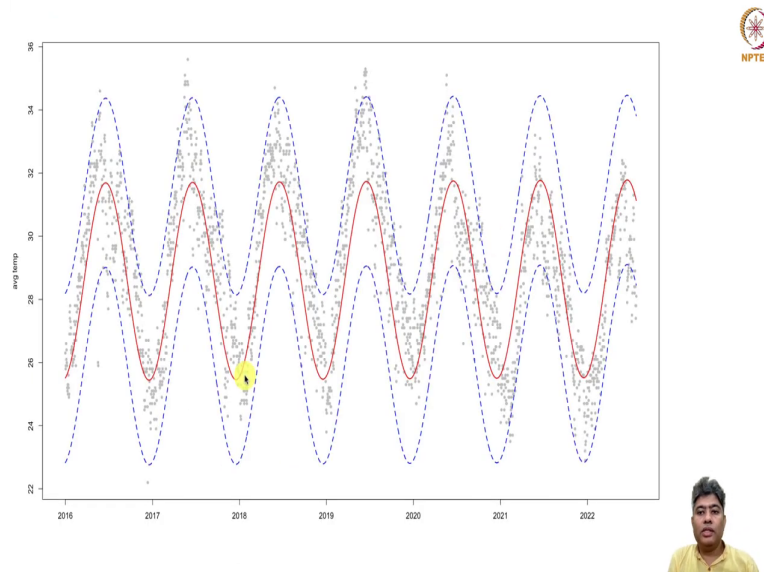
The screenshot displays the RStudio environment. The top-left pane shows R code with lines 60-70. Line 68 is highlighted in blue. The top-right pane shows the Environment window with variables: `data_test` (2398 obs. of 7 variables), `data_train` (9496 obs. of 7 variables), `mod1` (Large lm (13 elements...)), and `sum` (Large summary.lm (12 e...)). The Values section shows: `n` = 11894, `omega` = 0.01721420632104, and `sigma` = 1.36776132407984. The bottom-left pane shows the Console with the following R commands:

```
> lines(data_train$time[1:(3*365)]
+       ,data_train$fitted.values[1:(3*365)]+1.96*sigma
+       ,col='brown',lwd=2,lty=2)
> data_test$pred = predict(mod1,newdata = data_test)
> View(data_test)
> plot(data_test$time,data_test$avg
+       ,pch=20,col='grey'
+       ,xlab='',ylab='avg temp')
> lines(data_test$time
+       ,data_test$pred
+       ,col='red',lwd=2)
> lines(data_test$time
+       ,data_test$pred-1.96*sigma
+       ,col='blue',lwd=2,lty=2)
>
```

The bottom-right pane shows a plot of average temperature (avg temp) over time (2016-2022). The plot features a brown line representing the fitted model, a red line representing the predicted values, and a blue line representing the predicted values plus 1.96 standard deviations. The y-axis ranges from 22 to 36, and the x-axis shows years from 2016 to 2022. A small inset image of a person is visible in the bottom right corner of the RStudio window.



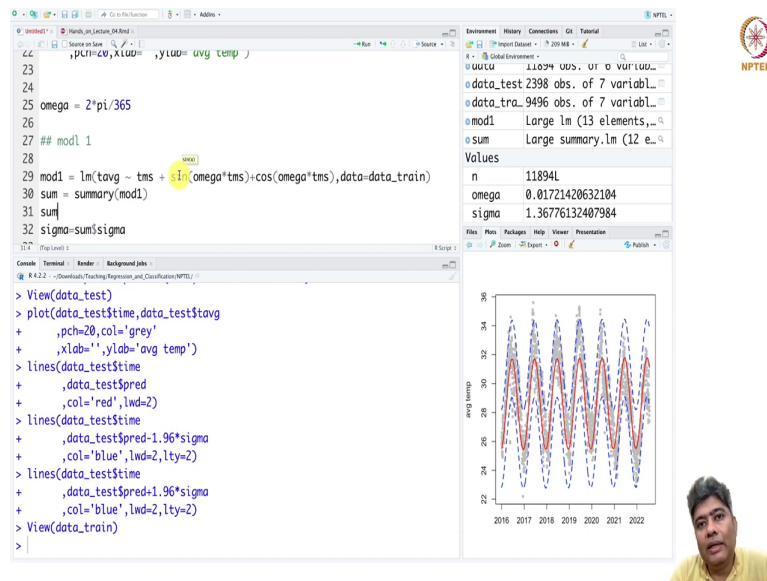
(Refer Slide Time: 28:54)



So, we have to just take this minus 1.96 times sigma. We would like to use some different color maybe blue with lty equal to 2. So, this is our lower band, this is our lower band and we have to do a upper band as well. So, all I have to do is here immediate plus and let us draw the train. So, in the test data set, most of our predictions are within the 95 percent predictive interval. So, we can say that this is a simple model with some sin cosine feature, and then it has done pretty good.

Now, little bit interesting point I would like to make here. So, if you see our data set, right our data set is in two-dimension effectively from a train point of view, we have effectively time and the average time, the dates, and the average temperature.

(Refer Slide Time: 30:02)



So, we have only two variable, average temperature and the time. From there we have created these, these two variables sin and cosine are our engineered feature, ok. These are our engineered feature, sin omega t and cos omega t are engineered feature. So, with the engineered feature we fit a nice model and which is doing pretty reasonable in terms of predicting the temperature.

So, we will stop here. And we will continue in on it, you can always ask me one question that why we are stopping at only two sin cosine Fourier engineered feature, Fourier feature. We can always go for higher order Fourier feature, like sin 2 omega t cos 2 omega t sin, 3 omega t cos 3 omega t.

In the next lectures, we will do those hands on with the higher dimension. But remember that more feature we will put, it will add more complexity in our model, our model will become

more high dimensional. So, the this has some advantage and disadvantage. We will talk about those model, increasing model complexity in the next lecture.

Thank you very much. See you in the next lecture.