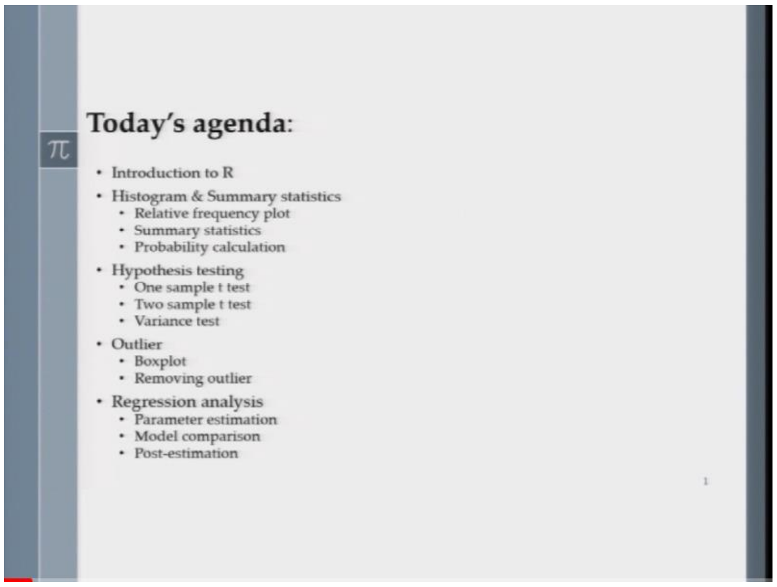
Applied Statistics and Econometrics

Professor Deep Mukherjee
Department of Economic Sciences
Indian Institute of Technology, Kanpur

Lecture 40

Hello friends. Welcome back to the lecture series on Applied Statistics and Econometrics. So, now we are in the last week of the course. And time passed like that. And finally, we have fallen short of enough time to teach you hands-on Excel or R software. So, we have only one and a half hours left in this course. So, I decided to give you some introduction on the R software because gradually the importance of R is becoming high and high in Applied Economic research and it is becoming popular also.

(Refer Slide Time: 01:00)



Today's agenda:

- Introduction to R
- Histogram & Summary statistics
  - Relative frequency plot
  - Summary statistics
  - Probability calculation
- Hypothesis testing
  - One sample t test
  - Two sample t test
  - Variance test
- Outlier
  - Boxplot
  - Removing outlier
- Regression analysis
  - Parameter estimation
  - Model comparison
  - Post-estimation

1

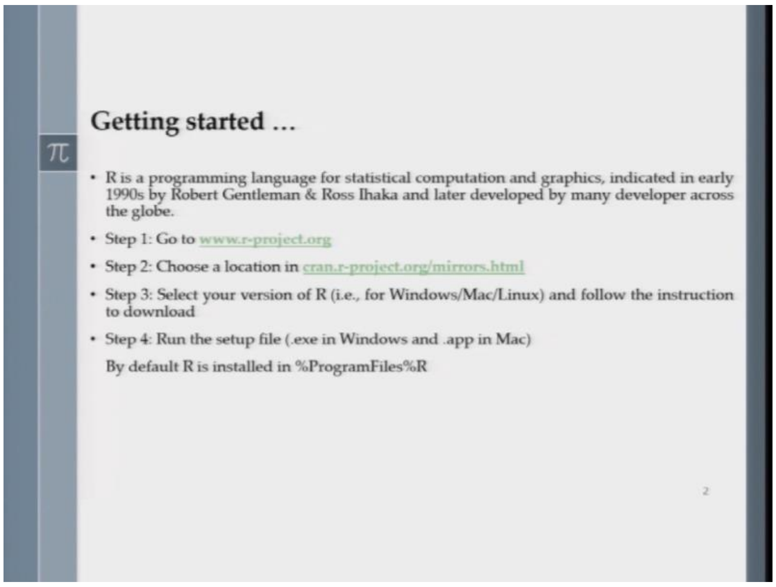
So, let us have a look at today's agenda items first. And there, I will start with an introduction to R and then I will talk about some basic exploratory data analysis. I will start by some graphing tools, important graphs like histogram, how to make them in R. Then I will talk about how to compute summary statistics, and then I will move on to the hypothesis testing component of the course and there are in I am planning to show one sample t test, two sample t test and Variance test. And then I will show you how to detect outliers. And finally, I will end today's lecture with regression analysis. So, I will show you what you a tutorial how to conduct a regression analysis.

So, in consultation with my TA for the course, I have decided to take a mixed mode. So, in real life lab classes what we do, we basically shared the screen with the students and then show them, if I write certain codes some software's platform then how the software is going to respond. So, if I want to conduct some statistical data analysis, say regression or hypothesis testing, what are the comments that I will write and then what will be the outcome of the software.

So, I have actually taken the screenshots of those very important lines, the codes or the commands, whatever you want to call them. And then I am also going to show you how the software is responding if I have written those commands and codes in the software platform. So, you are going to get a feel how you should interact with the R and it is not going to be only like that, I am also going to explain the results that I found from the R screen.

So, you will also get a very good recap or review of what we have discussed in this course. I am not going to show you all the statistical data analysts is what we have learned in theoretical terms in this software sessions, because we only have a couple of hours left. So, here I am going to be very selective and choosy. I am going to not talk about the most important things that I think you must try at your end once you are done with the course.

(Refer Slide Time: 03:52)



Getting started ...

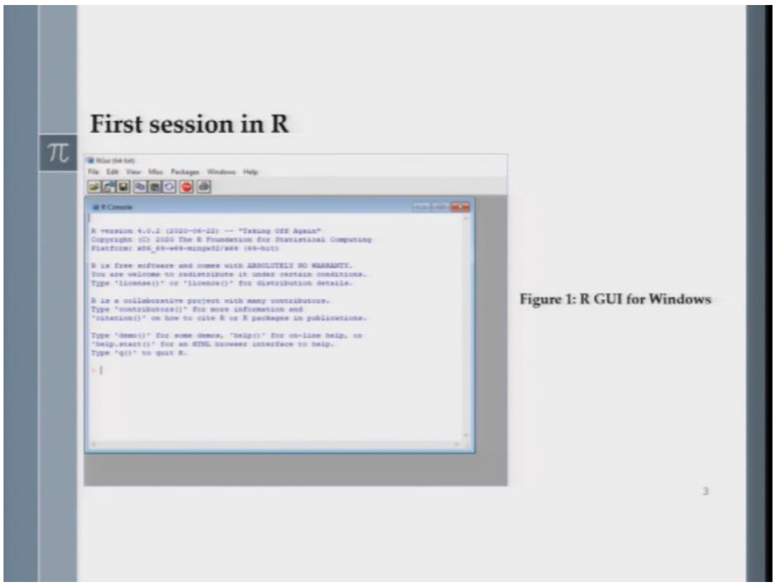
- R is a programming language for statistical computation and graphics, indicated in early 1990s by Robert Gentleman & Ross Ihaka and later developed by many developer across the globe.
- Step 1: Go to www.r-project.org
- Step 2: Choose a location in cran.r-project.org/mirrors.html
- Step 3: Select your version of R (i.e., for Windows/Mac/Linux) and follow the instruction to download
- Step 4: Run the setup file (.exe in Windows and .app in Mac)

  By default R is installed in %ProgramFiles%R

2

So, let us get started. R is a programming language and it was developed by Robert Gentleman and Ross Ihaka back in early 1990s and later many other coders and statisticians got into interested in their program and then gradually they have started contributing to this free software that got developed initially in 1990s. So, how to download the software. Once you have to conduct some data analysis with the software, you have to get a copy of the software. So, you go to this www.r-project.org link that I am showing here. And then you choose a location and then you select your version of R. So, you may be a windows user or a Mac Linux user. So, you choose your version for the R and then download. And then finally, after you downloaded the file, you need to run the set of file, of course. And then you are all set.

(Refer Slide Time: 04:57)



First session in R

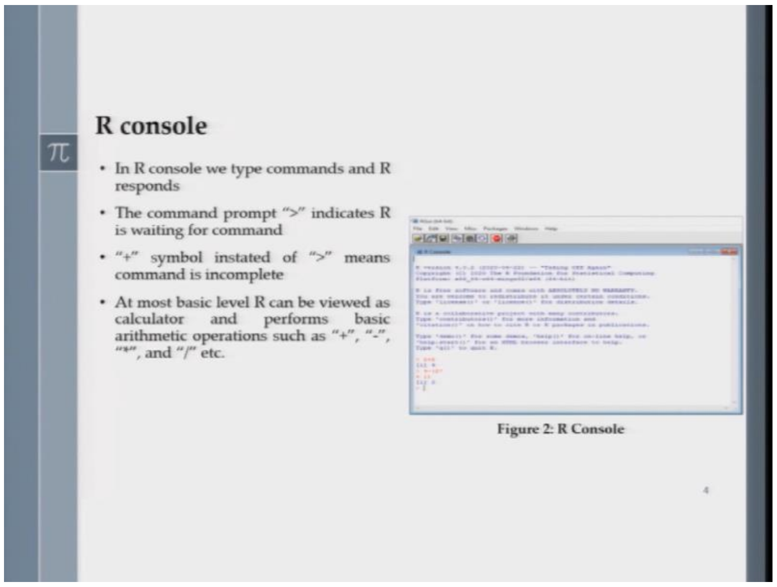
```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

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.

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.

> |
```

Figure 1: R GUI for Windows

3

So, here, I am showing you a screenshot of the first section in R. So, if you open the R window by clicking on the icon and after installing it, then you are going to see this kind of a picture. So, it is not very fancy, but still you can read that words and all here.

(Refer Slide Time: 05:22)



R console

- In R console we type commands and R responds
- The command prompt “>” indicates R is waiting for command
- “+” symbol instated of “>” means command is incomplete
- At most basic level R can be viewed as calculator and performs basic arithmetic operations such as “+”, “-”, “*”, and “/” etc.

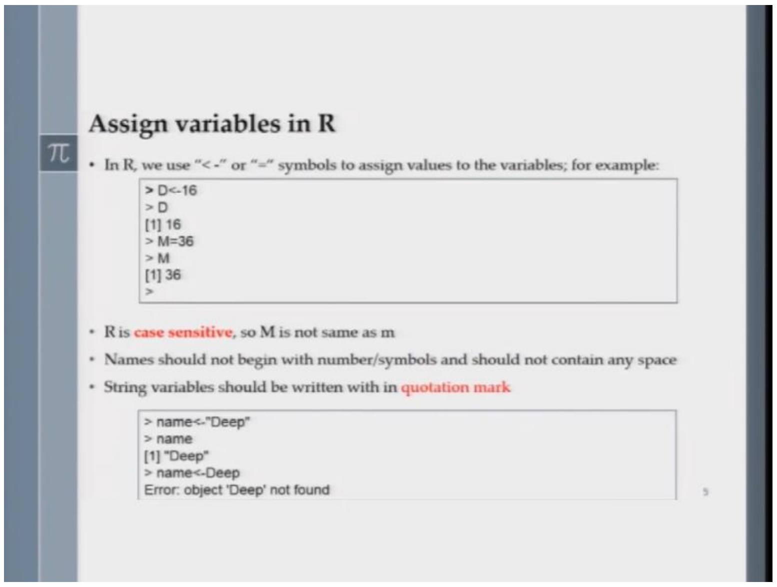
Figure 2: R Console

4

Then we have something called R console. This is a very important thing that you must understand before you start doing anything in R. So, in R console, we need to type all the commands and codes that we want to write in the software platform to conduct some data analysis and R actually responds to those codes and commands.

So, that command prompt starts with this arrow sign, this greater than sign. And that shows that R is waiting for your command. So, you have to now write something. Then there is a plus symbol that you can get and that is basically means that the command is incomplete. If you have written something incomplete command, R does not like it, and R shows you that plus sign. And then at most basic level R can be viewed as a calculator and it performs basic arithmetic operations, like addition, subtraction, multiplication, division. So, this is bare minimum that every software should do. R also does.

(Refer Slide Time: 06:37)



Assign variables in R

- In R, we use "< -" or "=" symbols to assign values to the variables; for example:

```
> D<-16
> D
[1] 16
> M=36
> M
[1] 36
>
```

- R is case sensitive, so M is not same as m
- Names should not begin with number/symbols and should not contain any space
- String variables should be written with in quotation mark

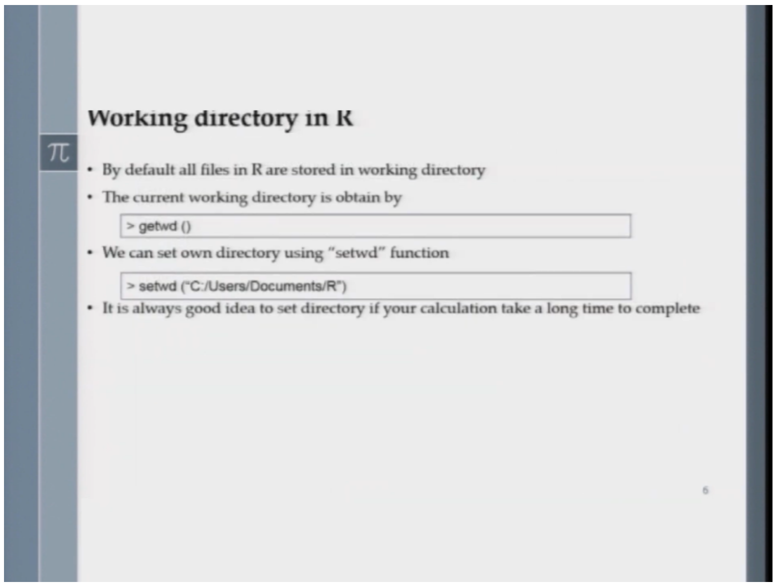
```
> name<-"Deep"
> name
[1] "Deep"
> name<-Deep
Error: object 'Deep' not found
```

5

Now, you have to then assign variables in the R platform so that you can compute certain things. And in R, you use this particular sign, less than sign and then minus sign or you can also use equal to sign. So, these symbols are used to assign the values to the variable. So, suppose, I want to declare a variable D and then I want to assign a value 16 to it. So, this is the way I am going to write. And then R will basically give me the value 16. Similarly, I have shown this for another case where I have used the other symbol to get the value.

And note that R is very, very case sensitive. So, this capital M is not same as small m. So, if you write capital M and then later you forgot, and then you write small m. Then R will not understand that. And some cautionary note. Names should not begin with number and symbols and should not contain any space. So, there are string variables. So, R can also handle string variables and they should be retained with quotation marks. So, like if I write someone's name, then that should be written within a quotation mark. If you do not use quotation, then you will get this kind of error message that you are seeing at the bottom of the box.

(Refer Slide Time: 08:07)



Working directory in R

- By default all files in R are stored in working directory
- The current working directory is obtain by

```
> getwd ()
```

- We can set own directory using "setwd" function

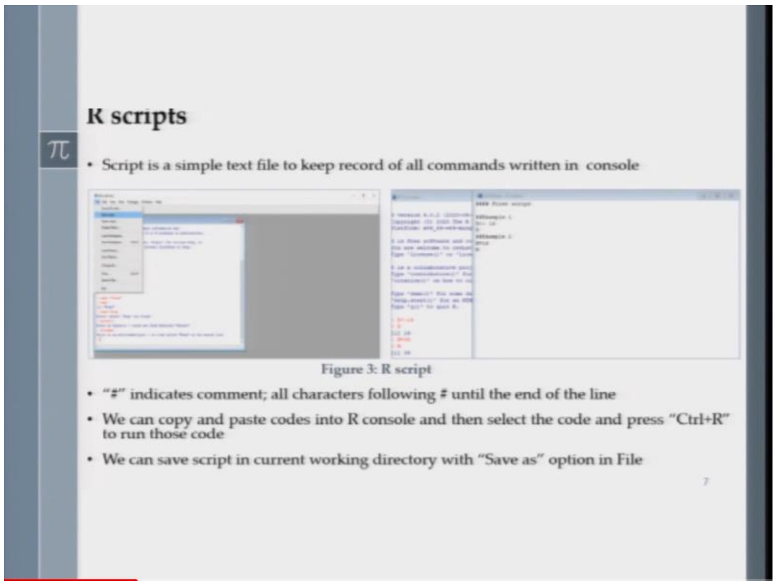
```
> setwd ("C:/Users/Documents/R")
```

- It is always good idea to set directory if your calculation take a long time to complete

6

Now, it is very important to open a working directory and that you should do next. So, by default, all the files in R are stored in this working directory and the current working directory you can obtain by writing this simple command. And you can also set your own directory, if you are a first time user of R by using this setwd function and I am showing you this code, how you can do it.

(Refer Slide Time: 08:42)



R scripts

- Script is a simple text file to keep record of all commands written in console

Figure 3: R script

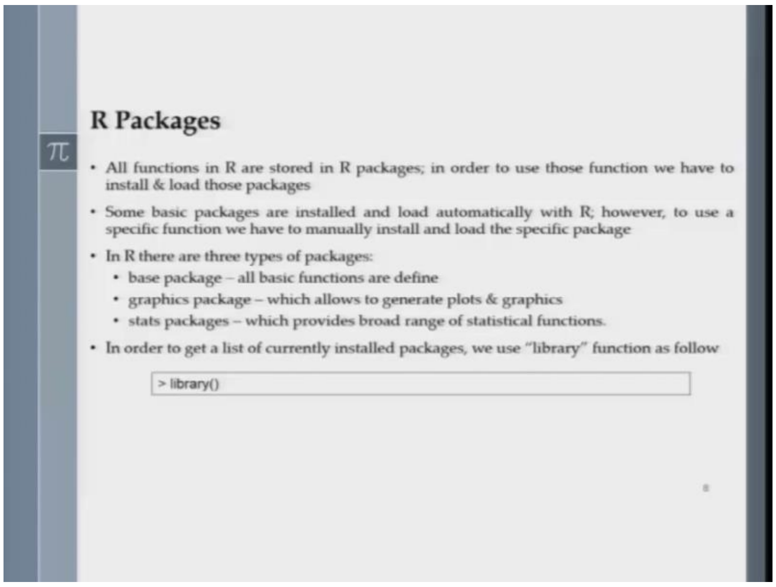
- "#" indicates comment; all characters following # until the end of the line
- We can copy and paste codes into R console and then select the code and press "Ctrl+R" to run those code
- We can save script in current working directory with "Save as" option in File

7

Now here in this slide, I am going to talk about the R scripts. Script is basically a simple text file to keep record of all the command that you have written in the console while you are interacting with R. And here I am showing you the screenshots. So, here the hashtag indicates comment and all characters following the hash sign until the end of the line. Basically, you can see these as a comment.

Suppose you have got some codes from outside the R environment, but it is written in the R programming language. Then what you do? You can copy paste those lines and paste them in the R console screen. And then you just press control R and the code should run, if there is no bug in it, of course. And you can also save the script in current working directory with save as option. So, these things you will learn gradually as you become more and more familiar with R.

(Refer Slide Time: 10:14)



R Packages

- All functions in R are stored in R packages; in order to use those function we have to install & load those packages
- Some basic packages are installed and load automatically with R; however, to use a specific function we have to manually install and load the specific package
- In R there are three types of packages:
  - base package – all basic functions are define
  - graphics package – which allows to generate plots & graphics
  - stats packages – which provides broad range of statistical functions.
- In order to get a list of currently installed packages, we use "library" function as follow

```
> library()
```

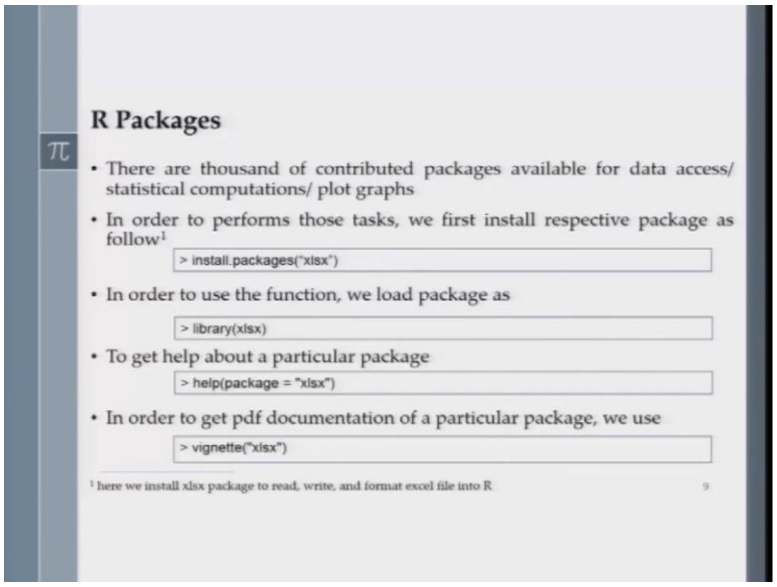
Now not that R is basically a software which works with different packages. And these packages are retained by different users and to conduct the same statistical analysis, there could be multiple packages available. Now, which one is best and which one is second best, I do not want to get into that discussion.

Now, some basic packages are already installed and loaded automatically when you download the R software. But if you need to use a very specific function, then you have to manually install. Remember that very specific functions may not be available with the downloaded version of R that you are working with. So, it is good idea to for search for the package, whether it is installed in your version of R or not, if it is not, then you need to manually download it.

So, in R there are three types of packages. The first type is base package. So, these are like all basic functions like addition, subtraction, multiplication. Then there are graphic packages, and these will allow us to generate plots. And then there are stats packages, which provides broad range of statistical functions. Now, in order to get a list of currently installed packages in your machine, you can use the library function. So, here I am showing how you can write library in our console, and then it will give you in return the list of currently installed packages in your system.

Note that when we are conducting statistical or econometric data analysis, generally the raw data is available to us in Excel format. And that is why it is very important that you must learn how to read the Excel file in any kind of software. So, in R also you can read Excel files and I am showing you what command you should write or what package you require to read an Excel file in R environment.

(Refer Slide Time: 12:00)



R Packages

- There are thousand of contributed packages available for data access/ statistical computations/ plot graphs
- In order to performs those tasks, we first install respective package as follow[1]

```
> install.packages("xlsx")
```

- In order to use the function, we load package as

```
> library(xlsx)
```

- To get help about a particular package

```
> help(package = "xlsx")
```

- In order to get pdf documentation of a particular package, we use

```
> vignette("xlsx")
```

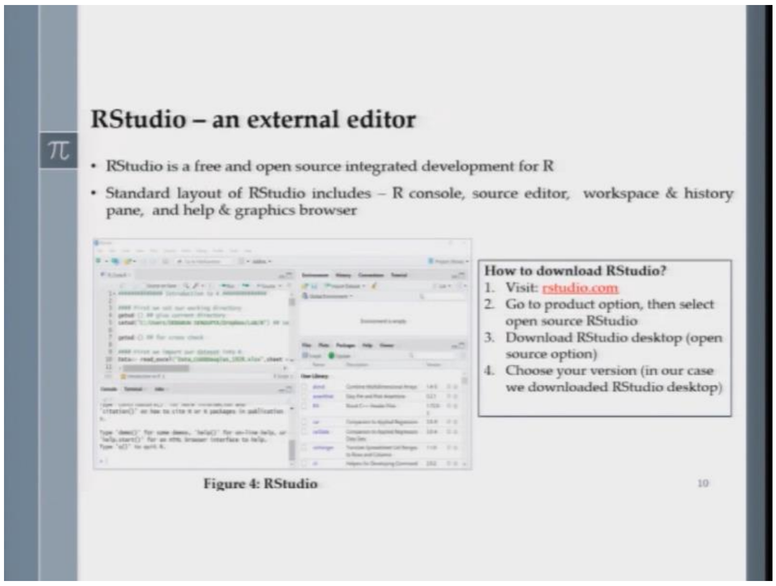
[1] here we install xlsx package to read, write, and format excel file into R

9

So, here you see, you need to first install a package and that is basically xlsx package and this package will help you to read, write and format Excel file in the R environment. So, you simply write this install dot packages within bracket, a xlsx command. And then if you enter you can actually install this package there. And then to, if you want to get help about a particular package say, xlsx itself.

So, then you can also write this help command and that will give you some screen here in front of you which will share some information from the R's side. Now, it is very interesting to note that for certain packages the help can also be downloaded in PDF file so that you can take a printout of that help file so that it helps you to read from the and then follow it step-by-step. So, for that also, we have a command. And that command is called vignette so if you write that, suppose, you want to learn more about the xlsx command. So, you write vignette xlsx and then, a PDF will pop up and you can download and print it.

(Refer Slide Time: 13:25)



RStudio – an external editor

- RStudio is a free and open source integrated development for R
- Standard layout of RStudio includes – R console, source editor, workspace & history pane, and help & graphics browser

How to download RStudio?
1. Visit: rstudio.com
2. Go to product option, then select open source RStudio
3. Download RStudio desktop (open source option)
4. Choose your version (in our case we downloaded RStudio desktop)

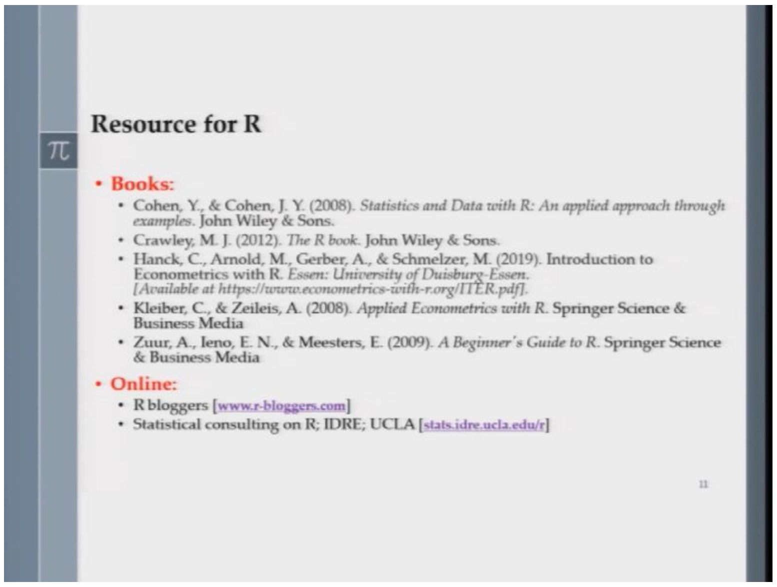
Figure 4: RStudio

10

Now, the major thing in R environment is Rstudio. It is an external editor, so it is basically standard layout where you can have R console, source editor, workspace, and history pane, and what, many things. So, here I am showing you the screenshot of Rstudio. It looks very messy and clumsy. But I think for a fee software this is good amount of sophistication that you find in Rstudio and how you can download Rstudio. Rstudio actually can be downloaded from rstudio.com and then you go to the product option, you can select for the open source Rstudio, download Rstudio desktop, and then choose your version.

It may be of great help if you have some resources to learn R like when I started learning software’s like LIMDEP or Stata or GAMS then manuals helped me a lot. And I think it is not a bad idea if you procure a copy of some resource book where it is shown with pictures and sentences, clearly written sentences that what you should do if you want to make R talk. And fortunately, there are many good textbooks or textbook kind manuals available for R. And I am going to refer some of them to you. So, you may get a copy of this, and then this will be of great help if you want to just open this book with you, in front of you and then practice R.

(Refer Slide Time: 15:30)



Resource for R

- **Books:**
  - Cohen, Y., & Cohen, J. Y. (2008). *Statistics and Data with R: An applied approach through examples*. John Wiley & Sons.
  - Crawley, M. J. (2012). *The R book*. John Wiley & Sons.
  - Hanck, C., Arnold, M., Gerber, A., & Schmelzer, M. (2019). Introduction to Econometrics with R. *Essen: University of Duisburg-Essen. [Available at https://www.econometrics-with-r.org/ITER.pdf].*
  - Kleiber, C., & Zeileis, A. (2008). *Applied Econometrics with R*. Springer Science & Business Media
  - Zuur, A., Ieno, E. N., & Meesters, E. (2009). *A Beginner's Guide to R*. Springer Science & Business Media
- **Online:**
  - R bloggers [www.r-bloggers.com]
  - Statistical consulting on R; IDRE; UCLA [stats.idre.ucla.edu/r]

11

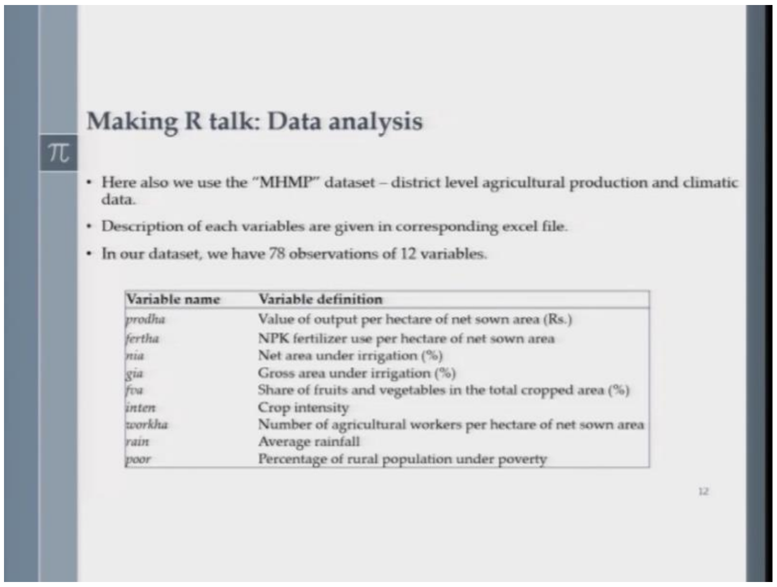
So, here I show five books, which I find a very useful if you want to learn R from a book written for students and out of these five, especially I suggest two books. The second one by Crawley, that name of the book is R book. And then the third one by Hanck Arnold. I like Hanck Arnold because it is available for free, so you can download this PDF book at free of cost. And I am also showing you where you have to go to download the copy of this book.

There are many online sources also and out of them, I am showing you these two which I find very useful and student friendly. One is R bloggers, and the other one is coming from university of California, Los Angeles. So, there is a website which is like stats.idre.ucla.edu. You can go and you can find many discussions on R.

Now, in today's lecture, I want to work with a simple dataset, and I would like to conduct very simple statistical data analysis that we have done in the first part of the course and first half of the second part of the course and for that actually I have got a dataset from Ministry of Agriculture, government of India.

So, regularly Ministry of Agriculture reports, crop yield data at district levels for various states in India. And I have downloaded such data for two states in India and they are Maharashtra and Madhya Pradesh. I have chosen these two states because these two states has many districts. So, just to increase the number of observations in my dataset, I have concentrated on these two states and I have actually made the arithmetic mean of the reported crop fields for two consecutive years, just to reduce the impact of random noise on the yield, because with this available data, a prior I do not have much information about the weather conditions in that area for that particular year, whether the area was suffering from flood or drought and all. My agricultural crop yield would be very sensitive to fluctuations in weather conditions and random shocks in weather.

(Refer Slide Time: 18:19)



Making R talk: Data analysis

- Here also we use the "MHMP" dataset – district level agricultural production and climatic data.
- Description of each variables are given in corresponding excel file.
- In our dataset, we have 78 observations of 12 variables.

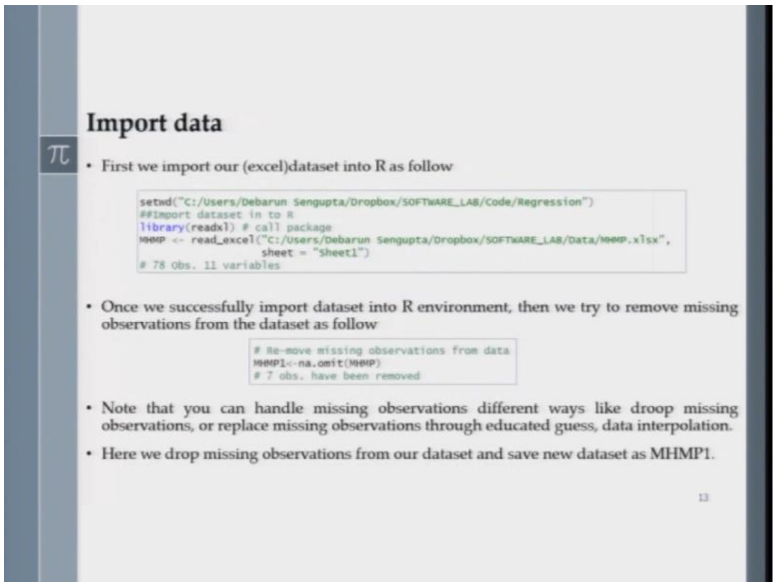
Variable name	Variable definition
prodha	Value of output per hectare of net sown area (Rs.)
fertha	NPK fertilizer use per hectare of net sown area
nia	Net area under irrigation (%)
gia	Gross area under irrigation (%)
fva	Share of fruits and vegetables in the total cropped area (%)
inten	Crop intensity
workha	Number of agricultural workers per hectare of net sown area
rain	Average rainfall
poor	Percentage of rural population under poverty

12

So, the name of the Excel file is MHMP and the description of the data I am showing you here. So, in total I have 78 observations. And these 78 observations have the data on 12 variables and these 78 by the way are districts in these two states, Maharashtra and Madhya Pradesh. So, let us have an idea about the data here. So, we have the value of output per hectare of net sown area in rupees. So, this is the monetary value of the production of crops. And then we have NPK fertilizer use per hectare, so this is input.

Now, net area under irrigation, this is institutional input. And we also have the share of fruits and vegetables in the total cropped area. There is also data on crop intensity and number of agricultural workers per hectare of net sown area is another important agricultural input in the agricultural production function. Then finally, I have some data on average rainfall and I have another socio-demographic variable in my dataset and that is percentage of rural population under poverty or below poverty line.

(Refer Slide Time: 19:39)



Import data

- First we import our (excel)dataset into R as follow

```
setwd("C:/Users/Debarun Sengupta/Dropbox/SOFTWARE_LAB/Code/Regression")
##Import dataset in to R
library(readxl) # call package
MHMP <- read_excel("C:/Users/Debarun Sengupta/Dropbox/SOFTWARE_LAB/Data/MHMP.xlsx",
                   sheet = "Sheet1")
# 78 Obs. 11 variables
```

- Once we successfully import dataset into R environment, then we try to remove missing observations from the dataset as follow

```
# Re-move missing observations from data
MHMP1<-na.omit(MHMP)
# 7 obs. have been removed
```

- Note that you can handle missing observations different ways like droop missing observations, or replace missing observations through educated guess, data interpolation.
- Here we drop missing observations from our dataset and save new dataset as MHMP1.

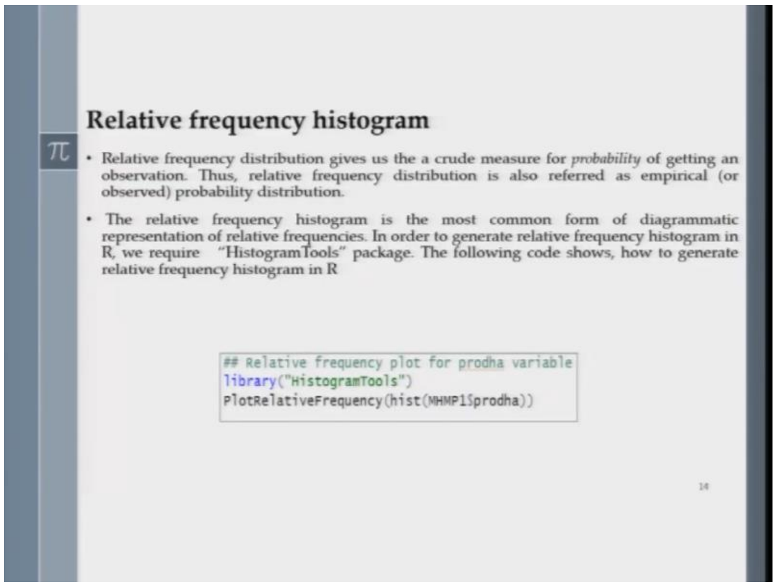
13

Now, we are going to play with this data. First, I need to import the data in the R environment. Suppose the data is stored in some working directory. And that is I am showing here in this setwd command, and then you import the dataset into R. So, that is basically showing here. So, how to import the data in R environment that I am showing here in the first box. Here, this second box in the slide is showing you what you need to write as command if you want to remove the missing observations from the dataset. So, after you dropped the missing observations from the dataset, then you may like to save it as a new dataset. And that is what we have done as MHMP1 file name.

Now, when you start talking about statistical data analysis and you have some kind of regression in, in your mind, you finally would like to end up doing some regression analyses to see whether some set of explanatory variables have any impact on a particular dependent variable or not. The best way to start is to by looking at the histogram of the variables on which you have quantitative data.

Why? Because by looking at the histogram, an associated concept called box plot, you can easily identify the outlier. I think in my humble opinion although you can check for outliers later on once you are done with your regression analysis, and we have discussed that in the course by looking at the residual values and all, but I think it is better, you try to figure out the outliers at the very beginning before you even start thinking about regression and all.

(Refer Slide Time: 21:44)



Relative frequency histogram

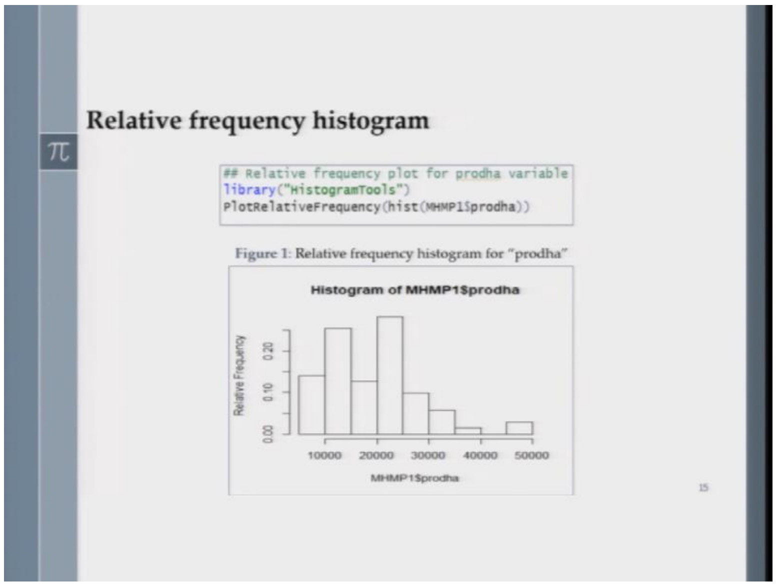
- Relative frequency distribution gives us the a crude measure for *probability* of getting an observation. Thus, relative frequency distribution is also referred as empirical (or observed) probability distribution.
- The relative frequency histogram is the most common form of diagrammatic representation of relative frequencies. In order to generate relative frequency histogram in R, we require "HistogramTools" package. The following code shows, how to generate relative frequency histogram in R

```
## Relative frequency plot for prodha variable
library("HistogramTools")
PlotRelativeFrequency(hist(MHMP1$prodha))
```

14

So, now we are going to first have a look at the relative frequency distribution, because it gives us the crude measure for probability of getting an observation. And I am showing you the command in the box at the bottom, what you need to write. So, that you can retrieve the relative frequency plot from R. And note that here, I am focusing on my output variable, so I am actually drawing the histogram for the prodha variable. So, that is basically the output variable in the dataset that I have collected.

(Refer Slide Time: 22:30)



Relative frequency histogram

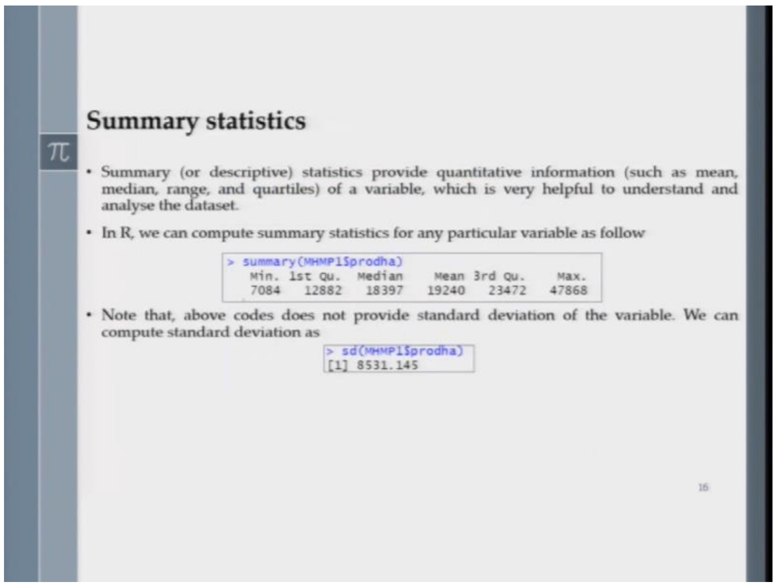
```
## Relative frequency plot for prodha variable
library("HistogramTools")
PlotRelativeFrequency(hist(MHMP1$prodha))
```

Figure 1: Relative frequency histogram for "prodha"

15

So, if I write that command that I am again showing in the box at the top of the slide, you see the histogram plot is generated, and that is my figure 1 here. So, you see that from this histogram, you can say that it is the output variable is not normally distributed and most likely it is positive distribution.

(Refer Slide Time: 22:54)



Summary statistics

- Summary (or descriptive) statistics provide quantitative information (such as mean, median, range, and quartiles) of a variable, which is very helpful to understand and analyse the dataset.
- In R, we can compute summary statistics for any particular variable as follow

```
> summary(MHMP1$prodha)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   7084   12882   18397   19240   23472   47868
```

- Note that, above codes does not provide standard deviation of the variable. We can compute standard deviation as

```
> sd(MHMP1$prodha)
[1] 8531.145
```

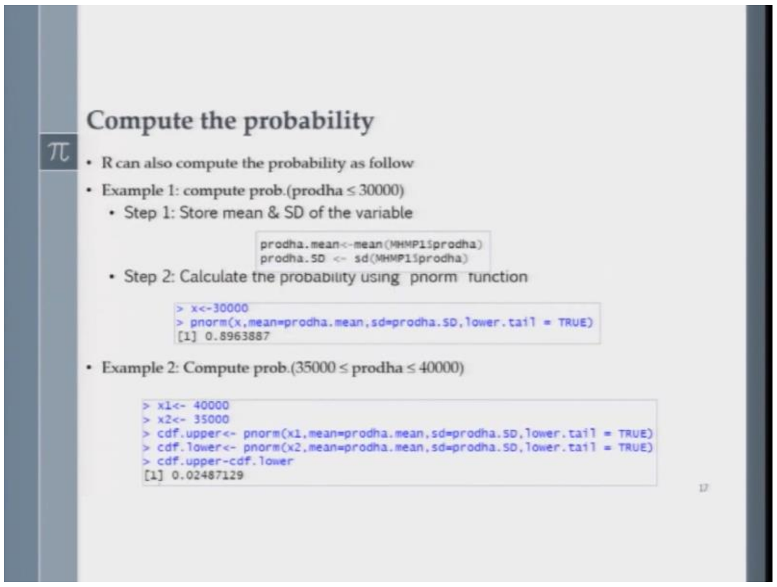
16

Now, after you have drawn the histogram for a variable, it is time to have a look at the summary statistics because summary statistics will give you a fair idea about the distribution. So, here I am showing you what you need to write in R console so that you can retrieve the summary statistics figure.

So, here command is summary. And then you have to write the name of the dataset, and then this dollar sign. After that, you need to write this prodha as the variable for which you are asking for summary statistics figures from R and then the R will in return give you the information on minimum value first quartile and median, mean, third quartile and the maximum value.

Note that here somehow it is not giving you data on standard deviation, but it does not mean that R cannot compute standard deviation for you. So, for that, you just have to write another command sd and then R will give you the value of the standard deviation as well. Note that, you do not forget to write the name of the data file from where R has to read the numbers and calculate for you. And then the name of the file and the name of the variable for which you are asking summary statistics should be differentiated by this dollar sign.

(Refer Slide Time: 24:24)



Compute the probability

- R can also compute the probability as follow
- Example 1: compute prob.(prodha $\leq$ 30000)
    - Step 1: Store mean & SD of the variable

```
prodha.mean<-mean(MHMP1$prodha)
prodha.SD <- sd(MHMP1$prodha)
```

    - Step 2: Calculate the probability using pnorm function

```
> x<-30000
> pnorm(x,mean=prodha.mean,sd=prodha.SD,lower.tail = TRUE)
[1] 0.8963887
```

- Example 2: Compute prob.(35000 $\leq$ prodha $\leq$ 40000)

```
> x1<- 40000
> x2<- 35000
> cdf.upper<- pnorm(x1,mean=prodha.mean,sd=prodha.SD,lower.tail = TRUE)
> cdf.lower<- pnorm(x2,mean=prodha.mean,sd=prodha.SD,lower.tail = TRUE)
> cdf.upper-cdf.lower
[1] 0.02487129
```

17

Now, let us compute probability. Because relative frequency distribution can be useful to give you proxy probability figures or crude measures of probability. So, here, I mean although the histogram is not looking like a a bell-shaped curve, but we can still forcefully assume normal distribution and get some probability numbers.

So, that is what I am showing you here. So, I am going to now show you how you can compute the probability that the crop yield or value of production, whatever you want to call it, actually you should call it the monetized value of production per hectare. If that variable follows a normal distribution, then what is the probability that this particular variable will take a value in some particular range.

So, for that, actually you have to save the standard deviation and mean of the relative frequency distribution. And then, you have to make use of these two numbers. You have to tell the software that look, these are the mean and standard deviations of my normal distribution and now you tell me what is the probability that these variables will take some value in a particular range.

So, here, how to store mean and standard deviation of the variable. So, here I am showing you what you need to write. So, it simple then for calculating the probability, you have to make use of the pnorm function. So, pnorm function actually gives you the probability from a normal distribution.

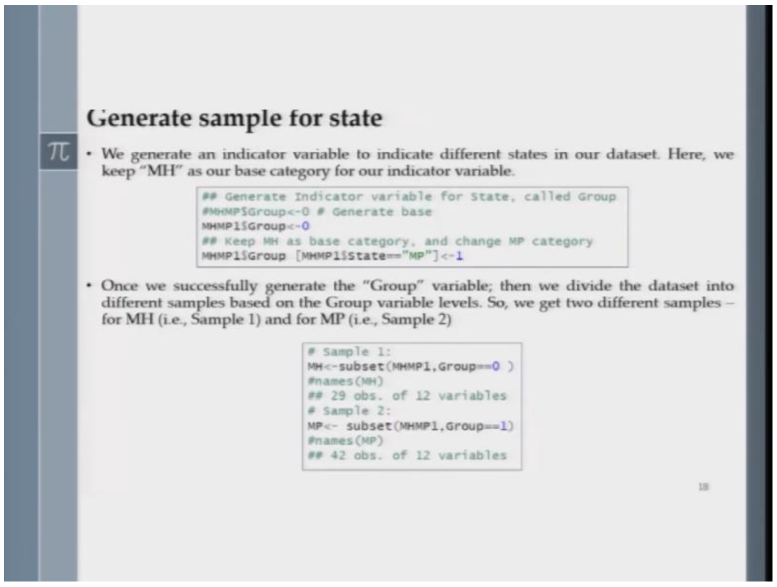
So, here in the second box, I am showing you what you need to write in R console. And then it will return you some number here. In the first example, I am showing you, if I say that x takes value less than 30,000, then you write this command, and it will return you a probability value of 0.89. Now, if you are interested to compute probability of observing a value of prodha between 35,000 and 40,000, then what to do? Then you define two different numbers first. x1, and x2. These will denote these two different limiting points of your range, 35,000 and 40,000 respectively.

And then you compute the upper value from the cdf and the lower value from the cdf. And then basically you finally write this command where you take the difference between the cdf dot upper and the cdf dot lower. So, cdf, I expect you remember, what do we mean by cdf at the end of the course. So, we have made use of this big capital phi if you remember, when we were computing normal probabilities from standard normal table. So, that cdf is basically mentioning that standard normal cdf.

And then finally, you see, you get a value of 0.0248. Now, we are going to conduct statistical data analysis and suppose I want to conduct some regression analysis where I want to throw a dummy variable or indicator variable for a state or I may be interested to see whether the means are, or the variances are varying across states or not.

So, for that actually, we need to split the data into two groups and for each state we should have one group. So, how to do that. So, for that, we have to now define an indicator variable in the R platform. And now I am going to show you the command, what you need to write if you want to create a dummy variable in R environment.

(Refer Slide Time: 28:30)



Generate sample for state

- We generate an indicator variable to indicate different states in our dataset. Here, we keep "MH" as our base category for our indicator variable.

```
## Generate Indicator variable for State, called Group
#MHMP$Group<-0 # Generate base
MHMP1$Group<-0
## Keep MH as base category, and change MP category
MHMP1$Group [MHMP1$State=="MP"]<-1
```

- Once we successfully generate the "Group" variable; then we divide the dataset into different samples based on the Group variable levels. So, we get two different samples – for MH (i.e., Sample 1) and for MP (i.e., Sample 2)

```
# Sample 1:
MH<-subset(MHMP1,Group==0 )
#names(MH)
## 29 obs. of 12 variables
# Sample 2:
MP<- subset(MHMP1,Group==1)
#names(MP)
## 42 obs. of 12 variables
```

18

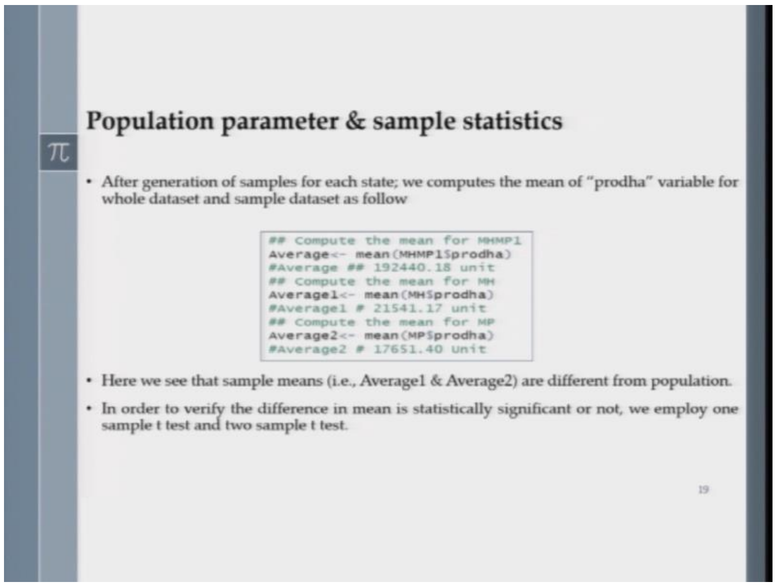
So, now I am showing here how you can generate this indicator variable named Group. So, here, you define a new variable in the dataset, and then you name it Group, and then you assign zero value to all of these observations corresponding to the different districts. And then you want to keep Maharashtra as the base category. So, then, you need to change the value for this particular variable.

So, now you go to the data and refer to MHMP1 dataset, and then there is a variable named State. So, for that actually you see two different categories MH and MP recorded in the dataset. So, then for the label or the category MP, for that state variable in MHMP1 dataset you declare that my Group variable will now take value one. And that is what I have done in the first box. Now, once you successfully generate the group variable, then we divide the dataset into different samples based on the group variables.

So, after successfully generating the group variable, we get two different samples. One for Maharashtra and let me say that this is my sample 1 and one for the Madhya Pradesh state and that I assume to be my sample number 2. So, now, I can actually write some further commands to see how many observations I have for each of these groups.

So, here, I can find out that by writing these commands that I am showing here at the box at the bottom. So, here, you write a command to retrieve that for group equal to 0 that is basically Maharashtra, you have 29 observations and for your sample 2, which is basically defined as a group equal to one, and that is basically for Madhya Pradesh, you have 42 observations.

(Refer Slide Time: 30:54)



Population parameter & sample statistics

- After generation of samples for each state; we computes the mean of "prodha" variable for whole dataset and sample dataset as follow

```
## Compute the mean for MHMP1
Average<- mean(MHMP1$prodha)
#Average ## 192440.18 unit
## Compute the mean for MH
Average1<- mean(MH$prodha)
#Average1 # 21541.17 unit
## Compute the mean for MP
Average2<- mean(MP$prodha)
#Average2 # 17651.40 unit
```

- Here we see that sample means (i.e., Average1 & Average2) are different from population.
- In order to verify the difference in mean is statistically significant or not, we employ one sample t test and two sample t test.

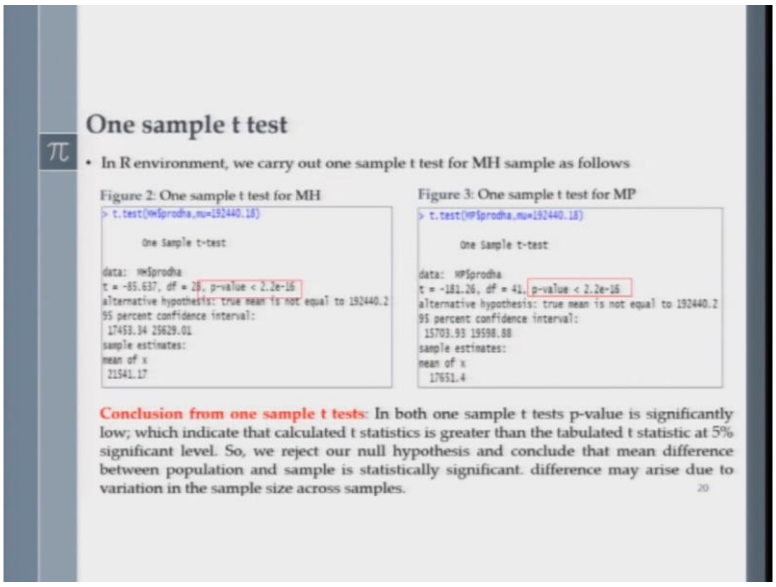
19

Now, I want to compute the population parameter and sample statistics from that data. So, again, we are showing this exercise for prodha variable. The monetary value of the crop output per hectare. And here, you have to write average command to get the value of the mean from the entire dataset. And I here assume that as I have data on only these two states, I assume that my population or universe consists of only these two states. And when I am looking at state level data for these two different Groups, I am treating them as samples, different samples drawn from that universal population, which consists of two different States.

So, if I now want to compare the means, population mean and the sample mean, etcetera. Let me compute the average from the entire dataset and that you can see, I have obtained by writing this command and I am denoting these variables as the name Average. And the mean of Maharashtra state is denoted by the variable name, Average1. And I am showing here, how you can get it computed in R and then finally, if I want to compute the mean for Madhya Pradesh then I create a new variable, say Average2 to store the mean value of Madhya Pradesh districts. And that also is done.

Now, I want to see whether the sample means, which are coming from two different Groups, Group1 and Group2 and they are named as Average1 and Average2 are different from the population mean. So, in the box above, I have shown you how you can get an idea about the population mean, and the sample means. Now note that, the sample means or the values for Average1 and Average2 are somewhat different from the average, when I have computed it over all the districts taken together. So, in order to verify whether these difference is statistically significant or not, now we have to employ one sample t test and two sample t test.

(Refer Slide Time: 33:30)



One sample t test

- In R environment, we carry out one sample t test for MH sample as follows

Figure 2: One sample t test for MH

```
> t.test(MH$prodha,mu=192440.18)

        One Sample t-test

data:  MH$prodha
t = -65.637, df = 28, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 192440.2
95 percent confidence interval:
 17453.34 25629.01
sample estimates:
mean of x
 21541.17
```

Figure 3: One sample t test for MP

```
> t.test(MP$prodha,mu=192440.18)

        One Sample t-test

data:  MP$prodha
t = -181.26, df = 41, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 192440.2
95 percent confidence interval:
 15703.93 19598.88
sample estimates:
mean of x
  17651.4
```

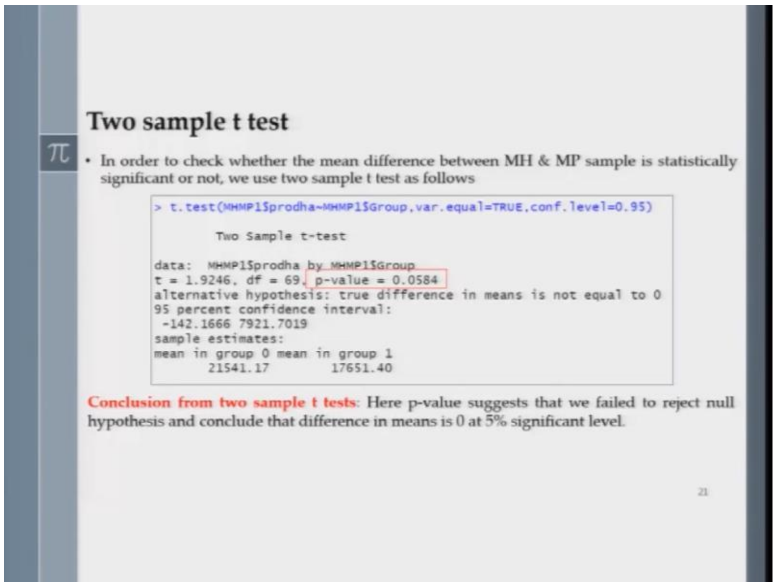
Conclusion from one sample t tests: In both one sample t tests p-value is significantly low; which indicate that calculated t statistics is greater than the tabulated t statistic at 5% significant level. So, we reject our null hypothesis and conclude that mean difference between population and sample is statistically significant. difference may arise due to variation in the sample size across samples.

20

So, now in this slide, I am showing you how you can conduct one sample test for both these groups. So, in figure 2, I am showing the case for Maharashtra and in figure three, I am showing the case of Madhya Pradesh. So, here you have to write a simple command, t dot test and note that within the parenthesis, you have to first mention the source of the data, the, and then you have to mention the population parameter, which value you are tasting for.

So, if you remember, when I computed the arithmetic mean or average for the entire dataset taken together, then I obtained this average figure 192440.18, so that I am using here now. And then one sample t test is conducted. You note that p value is reported. Similarly, I have conducted the t test for the for Madhya Pradesh in figure 2. And p value is also reported and I am highlighting p value here with this red box so that you can easily identify. Note that the p value is extremely small. So, basically, we reject our null hypothesis and then we can conclude that the mean difference between population and sample is statistically significant.

(Refer Slide Time: 35:10)



Two sample t test

- In order to check whether the mean difference between MH & MP sample is statistically significant or not, we use two sample t test as follows

```
> t.test(MHMP1$prodha~MHMP1$Group,var.equal=TRUE,conf.level=0.95)

        Two Sample t-test

data:  MHMP1$prodha by MHMP1$Group
t = 1.9246, df = 69, p-value = 0.0584
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -142.1666 7921.7019
sample estimates:
mean in group 0 mean in group 1
       21541.17        17651.40
```

Conclusion from two sample t tests: Here p-value suggests that we failed to reject null hypothesis and conclude that difference in means is 0 at 5% significant level.

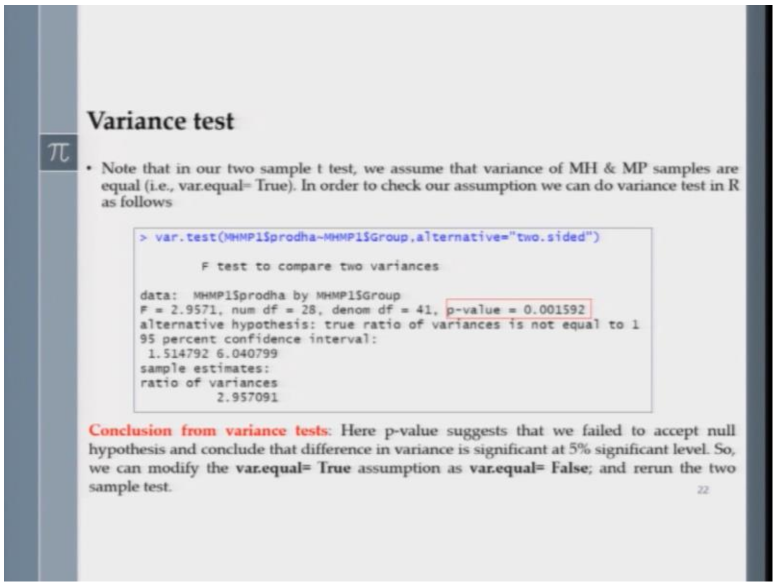
21

So, in the previous slide we have tested for the difference between the sample mean and the population mean, but we can also compare the means for two different groups. And for that, we need to conduct a two sample t test. So, I am going to show you how you can conduct it in R. Now, here in the box again, I am showing you the code, what you need to write to conduct a two sample t test and here you see again, the p value is reported.

Now, the p value is not that small, it is in fact on the borderline. So, it is 0.0584. Now, it is up to you. So, if you are a bit lenient, then you can ignore the third and fourth decimal point after the decimal point, the digit third and fourth. You can ignore the third and four digits after the decimal point. And then you can say that I have got a p value of 0.05, it is on the borderline case, or you can be a bit traditional you say, this is above 0.05. So, I am not going to reject my hypothesis. Of course, then your alpha that you have decided is 0.05. So, that is 5 percent level of significance.

Now, when we conduct two sample mean test, we assume that, of course to start with, we assume that the variance is same for two groups, but it may not be the case. So, we have to actually now check whether variance the same in two different groups or not. And for that also we can conduct a test in R.

(Refer Slide Time: 36:39)



Variance test

- Note that in our two sample t test, we assume that variance of MH & MP samples are equal (i.e., var.equal= True). In order to check our assumption we can do variance test in R as follows

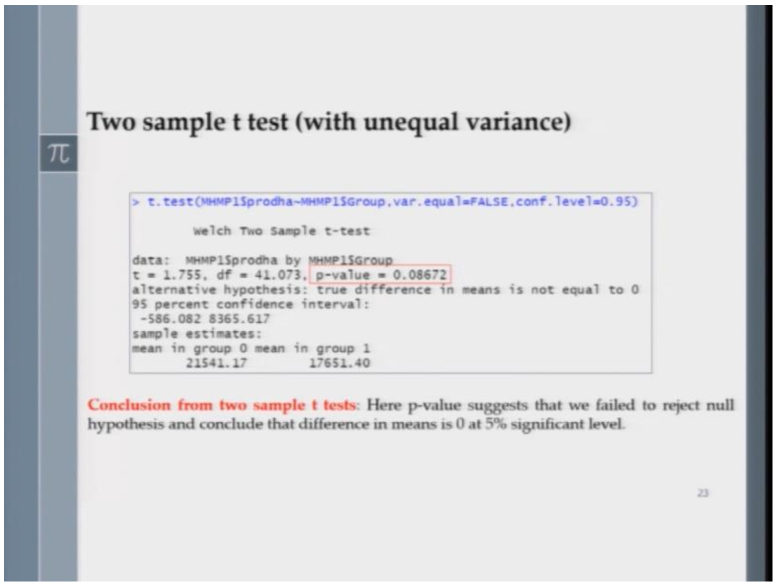
```
> var.test(MHMP1$prodha~MHMP1$Group,alternative="two.sided")

        F test to compare two variances

data:  MHMP1$prodha by MHMP1$Group
F = 2.9571, num df = 28, denom df = 41, p-value = 0.001592
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 1.514792 6.040799
sample estimates:
ratio of variances
          2.957091
```

Conclusion from variance tests: Here p-value suggests that we failed to accept null hypothesis and conclude that difference in variance is significant at 5% significant level. So, we can modify the **var.equal= True** assumption as **var.equal= False**; and rerun the two sample test.

22

So, here again in the box, I am reproducing the screenshot from R. I am showing you the code or command that you need to write that is in blue color as usual. And you see, it is an F test. if you remember your theoretical lessons in the course variance tests, of course an F test and here you see that p value is reported and the p value is quite small. So, as the p value is less than 0.05, you can reject the null hypothesis. So, as we are rejecting the null hypothesis, then we can modify our assumption that variance was equal. Now we can say that variance is not equal and hence we need to conduct another test.

(Refer Slide Time: 37:34)



Two sample t test (with unequal variance)

```
> t.test(MHMP1$prodha~MHMP1$Group,var.equal=FALSE,conf.level=0.95)

        Welch Two Sample t-test

data:  MHMP1$prodha by MHMP1$Group
t = 1.755, df = 41.073, p-value = 0.08672
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -586.082 8365.617
sample estimates:
mean in group 0 mean in group 1
       21541.17        17651.40
```

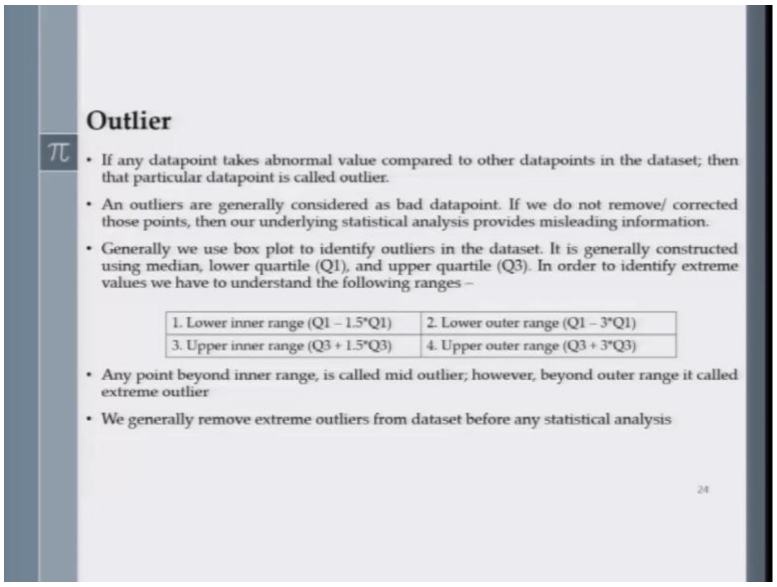
Conclusion from two sample t tests: Here p-value suggests that we failed to reject null hypothesis and conclude that difference in means is 0 at 5% significant level.

23

So, now I am going to show you how to conduct a two sample t test when you have this information that your variance is unequal. So, again, the code is written in the box, in blue color, the first line of course. And then you get a two sample t test figures thrown at you. And then you see the p value is 0.08. So, if you now set your alpha at 0.05, then again, you fail to reject your null hypothesis and conclude that difference in means, conclude that difference in means is 0 at 5 percent significance level.

Now, before you go to regression analysis, as I told you, I personally feel that it is not a bad idea to look at each variable by conducting some box plot analysis and then find out the outliers and then drop the outliers before you run the regression and that is what I am going to show you the next, how you can conduct outlier analysis before you proceed to regression analysis.

(Refer Slide Time: 38:45)



Outlier

- If any datapoint takes abnormal value compared to other datapoints in the dataset; then that particular datapoint is called outlier.
- An outliers are generally considered as bad datapoint. If we do not remove/ corrected those points, then our underlying statistical analysis provides misleading information.
- Generally we use box plot to identify outliers in the dataset. It is generally constructed using median, lower quartile (Q1), and upper quartile (Q3). In order to identify extreme values we have to understand the following ranges –

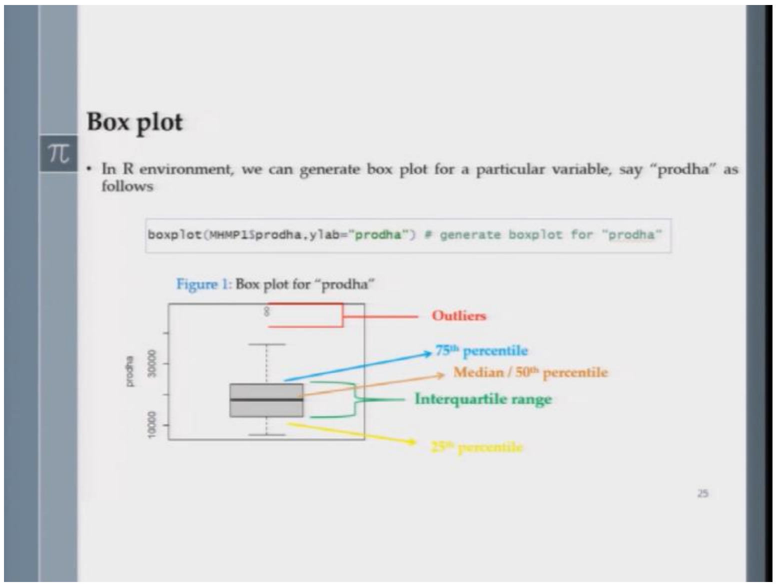
1. Lower inner range (Q1 – 1.5*Q1)	2. Lower outer range (Q1 – 3*Q1)
3. Upper inner range (Q3 + 1.5*Q3)	4. Upper outer range (Q3 + 3*Q3)

- Any point beyond inner range, is called mid outlier; however, beyond outer range it called extreme outlier
- We generally remove extreme outliers from dataset before any statistical analysis

24

So, here the outlier can be detected by looking at the box plot. So, if you remember the theoretical lessons, I told you that you have to talk about the interquartile range and here, I am reporting the four different ranges here, given in the box. And, if you have completely forgotten that, I suggest that you go back to the previous slides and revise because I do not have time to explain this concept here again, I am going to discuss the code only.

(Refer Slide Time: 39:23)



Box plot

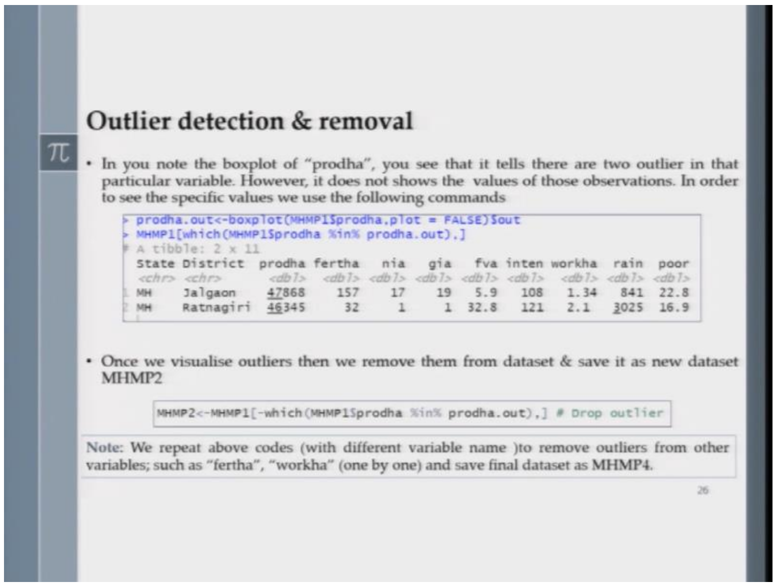
- In R environment, we can generate box plot for a particular variable, say "prodha" as follows

```
boxplot(MHMP1$prodha,ylab="prodha") # generate boxplot for "prodha"
```

Figure 1: Box plot for "prodha"

So, by writing the command that you see in the box, you can generate box plot for the output variable or the dependent variable prodha. And here you see the R software is proposing two observations as outliers, and these are the circles at the topmost end of the box that you see here. So, here in this diagram, I am pointing to the interquartile range and median, 75th and 25th percentile. So, you see, these two outliers are way far from the centre of that data and you can call them outliers.

(Refer Slide Time: 40:12)



Outlier detection & removal

- In you note the boxplot of "prodha", you see that it tells there are two outlier in that particular variable. However, it does not shows the values of those observations. In order to see the specific values we use the following commands

```
> prodha.out<-boxplot(MHMP1$prodha,plot = FALSE)$out
> MHMP1[which(MHMP1$prodha %in% prodha.out),]
# A tibble: 2 x 11
  State District  prodha fertha   nia   gia   fva inten workha  rain  poor
  <chr> <chr>      <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
1 MH    Jalgaon    47868    157    17    19   5.9   108   1.34   841  22.8
2 MH    Ratnagiri  46345     32     1     1  32.8   121   2.1   3025  16.9
```

- Once we visualise outliers then we remove them from dataset & save it as new dataset MHMP2

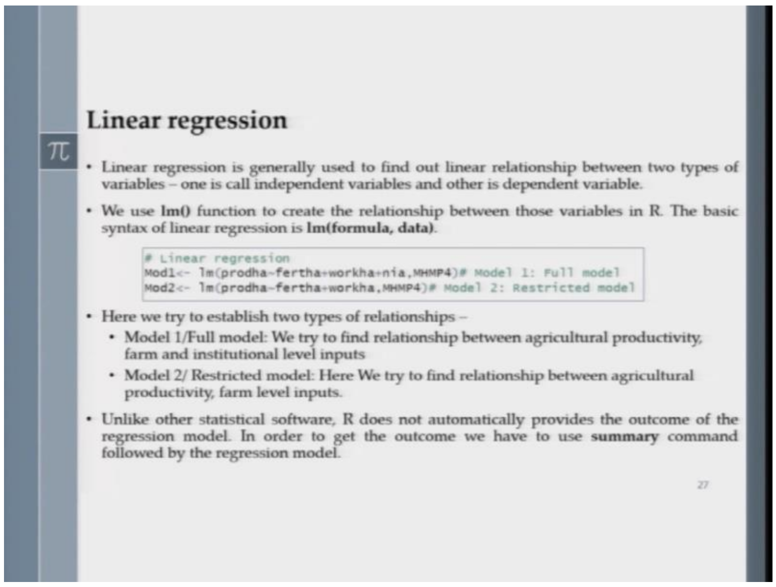
```
MHMP2<-MHMP1[-which(MHMP1$prodha %in% prodha.out),] # Drop outlier
```

Note: We repeat above codes (with different variable name)to remove outliers from other variables; such as "fertha", "workha" (one by one) and save final dataset as MHMP4.

26

So, now once you have detected some potential outliers two observations, then I think you should not take them to regression analysis, you should drop them at this point. And you can actually drop them out by writing some commands, but it is not a bad idea to have a look at this individual observation. So, you can also write couple of commands, lines to figure out which observations are potential outliers. So, here, I am showing you those commands. And if you type these two lines, then you will get those two potential outlier observations. And once you visualize the outliers, then you may want to remove them. And for that code is given in the second box. And then you are ready to run regression analysis with an outlier free dataset.

(Refer Slide Time: 41:08)



Linear regression

- Linear regression is generally used to find out linear relationship between two types of variables – one is call independent variables and other is dependent variable.
- We use **lm()** function to create the relationship between those variables in R. The basic syntax of linear regression is **lm(formula, data)**.

```
# Linear regression
Mod1<- lm(prodha~fertha+workha+nia,MHMP4)# Model 1: Full model
Mod2<- lm(prodha~fertha+workha,MHMP4)# Model 2: Restricted model
```

- Here we try to establish two types of relationships –
  - Model 1/Full model: We try to find relationship between agricultural productivity, farm and institutional level inputs
  - Model 2/ Restricted model: Here We try to find relationship between agricultural productivity, farm level inputs.
- Unlike other statistical software, R does not automatically provides the outcome of the regression model. In order to get the outcome we have to use **summary** command followed by the regression model.

27

So, now finally linear regression. So, now I want to talk about production function kind of function estimation. And I say that my output variable, which is the monetary value of agricultural production per hectare, that will depend on primarily three variables and one is fertilizer use per hectare, that is, fertha. The labour input per hectare that is, workha and then net irrigated area that is, nia.

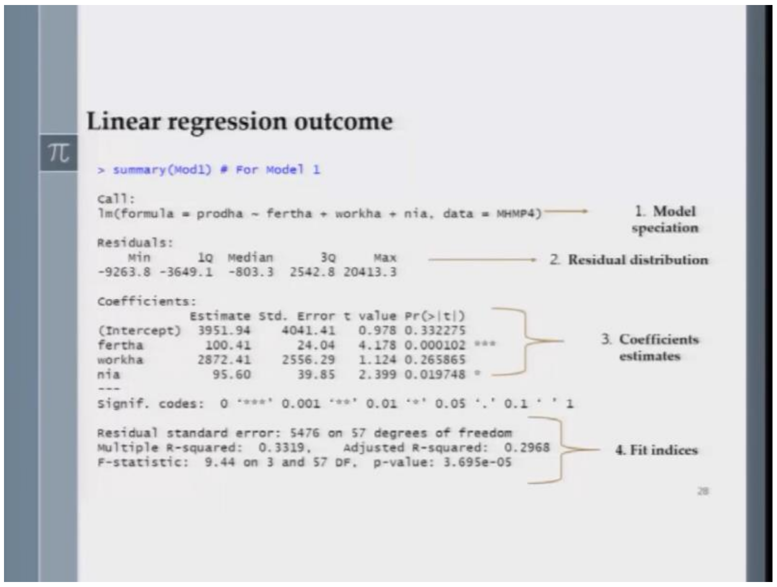
So, that is basically my full model. Now, in this full model, you know that there are two inputs which are coming from the farm households, which is basically the application of fertilizer and the labour input. And the other one is coming from institutional input, which is the net irrigated area because government has to provide some infrastructure so that farmer can use canal water for irrigation.

If there is no institutional infrastructure, then of course, farmers can also make use of pump set to extract groundwater and irrigate but overall, it is conceived that net irrigated area, if it is higher in a region, it will help farmer. So, the yield is expected to be higher for crops. So, we want to see whether this need irrigated area has any special influence on the variability of y or not, whether it has got enough statistical power to determine the levels of y or not. So, we are here going to compare restricted and unrestricted models. So, the unrestricted model is my full model, where I have three explanatory variables, including that nia variable and in the restricted model, I do not have that nia variable.

So, here, the Model 1 is full model and Model 2 is my restricted model. As I was explaining couple of seconds before. And you run two different linear regression models and you can name them Model 1 and Model 2 by putting that hash sign and then you type the name that you are giving to each of them. And that is basically Full model and Restricted model. You can see the codes clearly written in the box.

Now, I am showing you the summary sheet that R will throw at you once you conduct the linear regression analyses in R, when you write those commands that I have shown you in the earlier slide. What sort of an outcome R will present in front of you. And for that, let us have a look at the slide.

(Refer Slide Time: 44:05)



Linear regression outcome

```
> summary(Mod1) # For Model 1

Call:
lm(formula = prodha ~ fertha + workha + nia, data = MHMP4)

Residuals:
    Min      1Q  Median      3Q     Max 
-9263.8 -3649.1  -803.3  2542.8 20413.3 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3951.94    4041.41   0.978 0.332275    
fertha        100.41      24.04   4.178 0.000102 ***
workha       2872.41    2556.29   1.124 0.265865    
nia            95.60      39.85   2.399 0.019748 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5476 on 57 degrees of freedom
Multiple R-squared:  0.3319,	Adjusted R-squared:  0.2968 
F-statistic:  9.44 on 3 and 57 DF,  p-value: 3.695e-05
```

So, here you see, I am presenting the summary for my Model 1. So, you need to write that command, summary mod1 and now R will actually share all the results with you for this Model 1, which is the full model. So, you see the first line is talking about the model specification. It is giving you the dependent variable and all the independent variables. It is also showing you the data source from where it is coming. Then, the second part of the command is actually talking about the residual distribution.

So, it is providing the summary statistics for the residuals variable because you may be interested to see whether the residuals are following a normal distribution or not by looking at the histogram of the residuals and by making this chebyshev kind of rule that I spoke about earlier to test whether that rule is valid for this kind of dataset or not.

So, in the third block, you have the figures on the coefficients. These are the most important things. So, you see intercept is reported, then individual beta coefficients for input variables, fertha, workha and nia, they are all reported. And we see here that fertha is strongly significant, where nia is mildly least significant and workha is not significant at all.

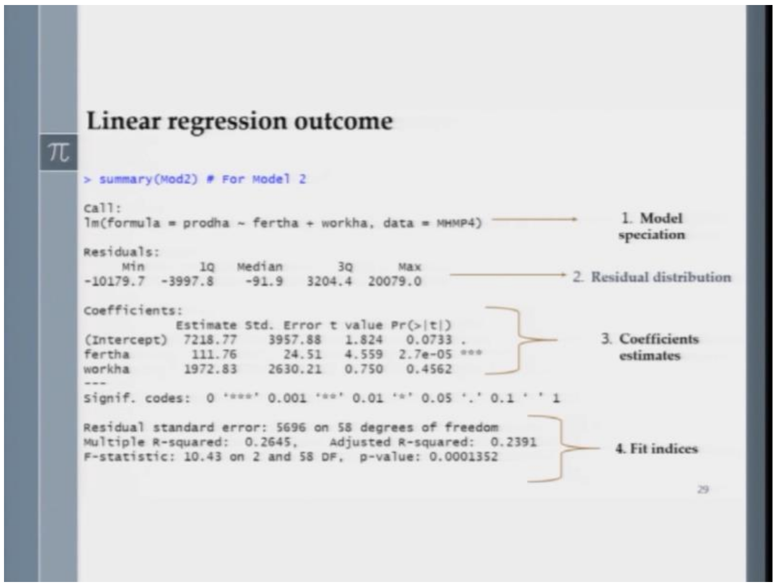
So, here, the significance levels of different variables are denoted by different symbols, like asterisks and dots. So, like if you see one asterisks, then you say that that particular regression coefficient is significant at 5 percent significance level. If you observe a dot, then that means that, that particular coefficient is significant at 10 percent level of significance.

And finally, you see at the bottom part of the summary statistic for regression, R is reporting the values for multiple R squared. And it is also reporting the value of adjusted R squared. Multiple R squared is not that great, but you generally do not expect very high R squared from cross sectional data with so few observations and so few variables.

Although it is not reporting the ANOVA table, you can still conduct that joint hypothesis testing, where you can jointly test the significance for all coefficients taken together because the F statistic value is reported. And let us have a look at the statistic value. So, here for the degrees of freedom 3 and 57, R is reporting you a p value, which is very small. So, you can say that you can reject the null hypothesis. So, you can say that jointly these three variables actually help you to determine the values of y. It helps you to explain the variations in y.

So, we are done with the discussion on the outcome of our regression Model 1. Now, if you remember, the objective was to do for a, the objective was to go for a model comparison. So, we, the end objective is to compare the restricted model with the unrestricted model and judge, which one is the better. So, here, this Model 1 is basically the unrestricted model where we have used the farm level inputs and the institutional input. So, we are having three explanatory variables here.

(Refer Slide Time: 48:08)



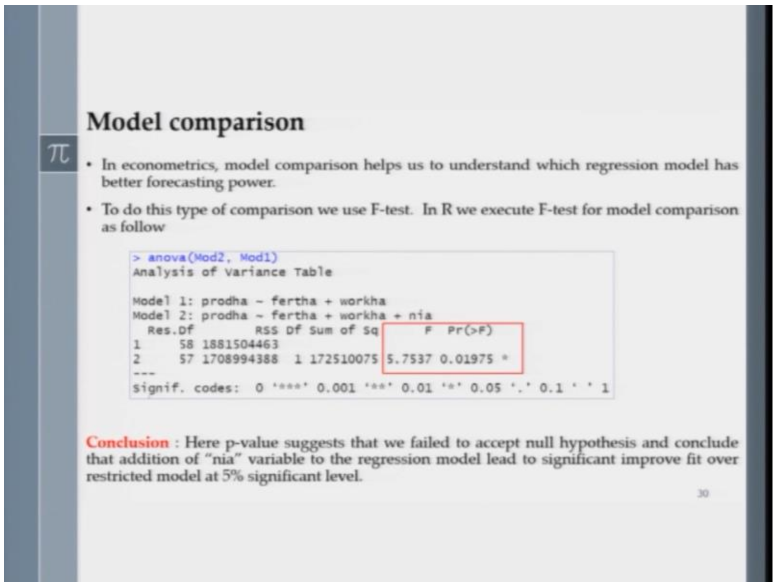
Linear regression outcome

```
> summary(Mod2) # For Model 2

Call:
lm(formula = prodha ~ fertha + workha, data = MHMP4)

Residuals:
     Min       1Q   Median       3Q      Max
-10179.7  -3997.8    -91.9   3204.4  20079.0

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  7218.77    3957.88   1.824   0.0733 .
fertha        111.76      24.51   4.559  2.7e-05 ***
workha       1972.83    2630.21   0.750   0.4562
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5696 on 58 degrees of freedom
Multiple R-squared:  0.2645,    Adjusted R-squared:  0.2391
F-statistic: 10.43 on 2 and 58 DF,  p-value: 0.0001352
```

29

Now let us move on to the restricted model case. That is our Model 2. And here in the restricted model, you see, I have dropped the institutional input, which is basically the net irrigated area, nia. And the regression outcome is in front of you. So, I have already explained to you how to look at a regression table in R so nothing new. But here note that although we are dropping one explanatory variable, but still the regression insignificant because the p value corresponding to the F statistic is quite low. So, fit indices are not that bad.

(Refer Slide Time: 48:45)



Model comparison

- In econometrics, model comparison helps us to understand which regression model has better forecasting power.
- To do this type of comparison we use F-test. In R we execute F-test for model comparison as follow

```
> anova(Mod2, Mod1)
Analysis of Variance Table

Model 1: prodha ~ fertha + workha
Model 2: prodha ~ fertha + workha + nia
  Res.Df        RSS Df Sum of Sq      F  Pr(>F)
1     58 1881504463
2     57 1708994388  1 172510075 5.7537 0.01975 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

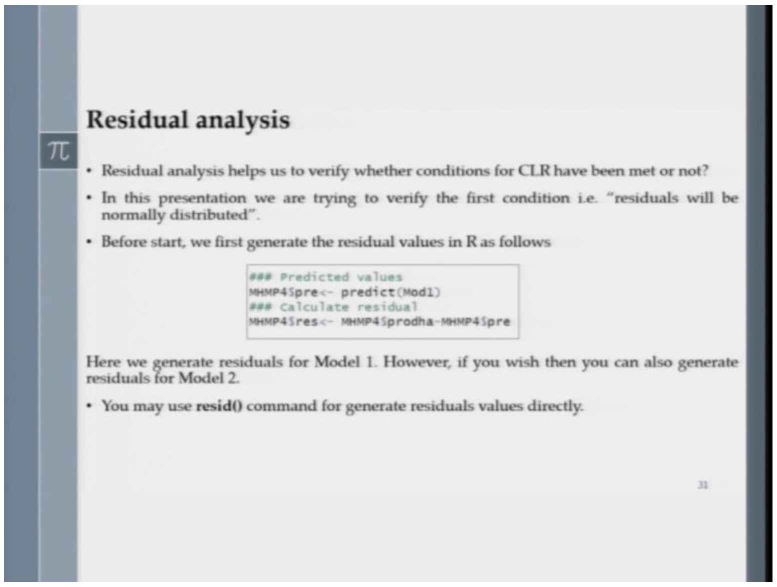
Conclusion : Here p-value suggests that we failed to accept null hypothesis and conclude that addition of "nia" variable to the regression model lead to significant improve fit over restricted model at 5% significant level.

30

So, now let us move on to the model comparison story. So, now how to do a model comparison? We have to run the F-test. That's what we have studied in the second part of the course, we have to conduct F-test if we want to compare the performance of restricted versus unrestricted models. So, here I am showing you how you can execute F-test for model comparison. So, here you see, I have stored the results for Model 1 and Model 2. And now I am calling those saved results. I am writing this anova command here and now it is showing me the analysis of variance table for Model 1 and Model 2.

And finally, you see, the F-statistic is computed. Its value is 5.7537. And for the degrees of freedom, the corresponding p value is 0.019. So, the p value is very small. It is less than 0.05. If we set a significance level of 5 percent, so actually we reject the null hypothesis. So, as we fail to accept the null hypothesis, we can conclude that the addition of the input variable, nia to 30 regression model lead to significant improvement in fit over the restricted model. So, basically, we require that nia variable back in the regression model, if we want to improve the fit of the model. So, basically the unrestricted model is doing a better job compared to the restricted model.

(Refer Slide Time: 50:23)



Residual analysis

- Residual analysis helps us to verify whether conditions for CLR have been met or not?
- In this presentation we are trying to verify the first condition i.e. "residuals will be normally distributed".
- Before start, we first generate the residual values in R as follows

```
### Predicted values
MHMP4$pre<- predict(Mod1)
### Calculate residual
MHMP4$res<- MHMP4$prodha-MHMP4$pre
```

Here we generate residuals for Model 1. However, if you wish then you can also generate residuals for Model 2.

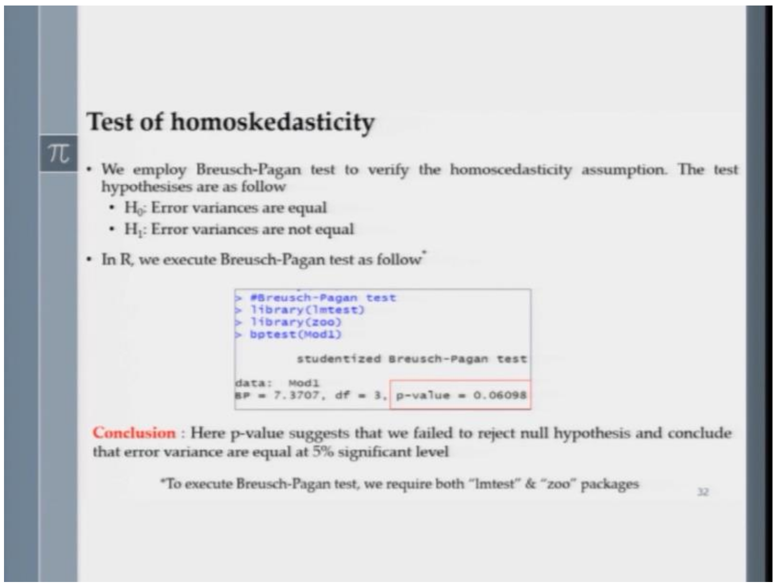
- You may use **resid()** command for generate residuals values directly.

31

Now, we want to bring your attention to residual analysis after running the regression never, ever forget to conduct residual analysis. And here I am showing you a couple of slides, how to conduct the residual analysis. So, here, first of all, you have to get the predicted value from your regression. So, here I am showing you the command, how you can generate the predicted value from your Model 1. And then, I am also showing you the code, how you can calculate the residual from that model. So, after we, I have generated the residuals for Model

1, we can also do the same for Model 2. Note that, we can also make use of the resid command for generating the residual values directly.

(Refer Slide Time: 51:18)



Test of homoskedasticity

- We employ Breusch-Pagan test to verify the homoscedasticity assumption. The test hypothesises are as follow
  - H_0: Error variances are equal
  - H_1: Error variances are not equal
- In R, we execute Breusch-Pagan test as follow*

```
> #Breusch-Pagan test
> library(lmtest)
> library(zoo)
> bptest(Mod1)

        studentized Breusch-Pagan test

data:  Mod1
BP = 7.3707, df = 3, p-value = 0.06098
```

Conclusion : Here p-value suggests that we failed to reject null hypothesis and conclude that error variance are equal at 5% significant level

*To execute Breusch-Pagan test, we require both "lmtest" & "zoo" packages

32

So, now we are going to talk about test of homoskedasticity because many times there could be heteroskedasticity issue and especially in the context of cross-section data, prevalence of heteroskedasticity problem. And we all know that cross-sectional data most of the times it is plagued by the issue of heteroskedasticity so better we check whether our residuals show any heteroskedasticity or not, because without testing for that, we cannot be sure that the generated residual obeys all the classical linear regression model assumptions.

So, now we are going for the test of heteroscedasticity or homoskedasticity by employing Breusch-Pagan test and this test starts with setting null hypotheses as error variances are equal against the alternative hypothesis that error variances are not equal. And now here I am showing you how one can execute the Breusch-Pagan test in R platform.

Now, note that to execute the Breusch-Pagan homoskedasticity or heteroskedasticity test, we require two packages, and we require both of them, lmtest and zoo. So, do not forget to install these two packages and check with the library command whether they are properly installed or not. So, first you call these two packages by this library command, and then you write this command, bptest and this bptest is done on Model 1.

So, here you get the value of the Breusch-Pagan test statistic and for 3 degrees of freedom, p value is also reported and you see the p value is small, but it is above 0.05. So, at 5 percent significance level, we actually fail to reject the null hypothesis. So, as we are failing to reject the null hypothesis, then we can conclude that error variances are equal at 5 percent significance level.

So, with this, we end our discussion on use of R conducting linear regression analysis utilizing cross sectional data. So, in the next two lectures, we are going to talk about the handling of other types of data and other types of econometric models. So, we will continue our discussion with R in the next lecture. Thank you.