Data Visualization

Prepared By Deepali Sonawane, Assistant Professor, SIOM

Concept of Data Visualization

- Data visualization means the **graphical representation** of data.
- It involves producing **images, charts, graphs** which shows relationships among the represented data to viewers.
- Data visualization uses statistical graphs, plots, information graphics and other tools to communicate information **clearly and efficiently**.
- Effective visualization helps users to **analyze** and **interpret** the data.
- The main goal of data visualization is to communicate information clearly and effectively through **graphical means**.
- Data visualization can be used for a **variety of purposes**: dashboards, annual reports, sales and marketing material etc.

Data Visualization tools

- Data visualization is the **graphical representation** of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to **see and understand** trends, outliers, and patterns in data.
- Data visualization tools provide data visualization designers with an easier way to create visual representation **of large data sets**.
- The best data visualization tools in the market have few things in **common** like ease of use, excellent documentation, tutorials and designed in ways that feel intuitive to the user.
- Eg. Google Charts, Tableau, Power BI, R, JupyteR etc.

Exploratory Data Analysis (EDA)

- Exploratory Data Analysis (EDA) is an approach **to analyze the data** using visual techniques.
- It is used to discover trends, patterns, or to check assumptions with the help of **statistical summary** and **graphical representations**.
- EDA is an approach to **analyzing** data sets, to **summarize** their main characteristics with visual methods.
- EDA was developed to encourage statisticians to **explore the data**, and possibly formulate hypotheses that could lead to new **data collection and experiments**.

Data Cleaning and Data inspection

- **Data cleaning** is the process of **fixing or removing** incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.
- Data cleaning involves the **detection** and **removal** or **correction of errors** and inconsistencies in a data set or database due to the **corruption or inaccurate** entry of the data.
- To ensure that you're dealing with the **right information** you need a clear view of your data at every stage of the transformation process.
- Data Inspection meets this need: it is the act of viewing data for **verification and debugging purposes**, before, during, or after a translation.

grep() function

- The grep() function searches for matches of certain character pattern in a vector of character strings and returns the **indices** that yielded a match.
- Syntax : grep("char",x)
- Example 1:

```
x<-c("banana","papaya","orange","grapes")
grep("n",x)
```

- Output : # 1 3
- Example 2:

```
d<-read.csv("emp.csv")
grep('Rahul',d$Name)
```

grepl() function

- The grepl() function searches for matches of certain character pattern in a vector of character strings and returns **a logical vector** indicating which elements of the vector contained a match.
- Syntax : grepl("char",x)
- Example 1:

  x<-c("banana","papaya","orange","grapes")

  grepl("n",x)
- Output : # TRUE FALSE TRUE FALSE
- Example 2:

  d<-read.csv("emp.csv")

  grepl('Rahul',d$Name)

sub() and gsub() function

- sub() and gsub() function in R are **replacement functions**, which replaces the occurrence of substring with other substring.
- sub() function in R replaces the **first instance** of a substring.
- gsub() function in R replaces **all the instances** of a substring.
- Syntax : sub(old,new,string)
- Example 1:

  s<-"India is my country. India is beautiful."

  sub("India","Bharat",s)
- Output : # Bharat is my country. India is beautiful.
- Example 2:

  s<-"India is my country. India is beautiful."

  gsub("India","Bharat",s)
- Output : # Bharat is my country. Bharat is beautiful.

summarize() function

- summarize() function reduces a **data frame to a summary** of just one vector or value.
- Many times these summaries are calculated by grouping observations using a factor or categorical variables.
- Example:

```
install.packages("dplyr")
library('dplyr')
code=c("a","b","c","d","e","f")
qty=c(200,300,500,150,700,600)
d=data.frame(code,qty)
d
s=summarize(d,mean(qty))
s
```

Graphical functions in R for data visualization

- Statistical analysis is usually based on **numerical techniques**.
- These techniques are based on assumptions about the data being used. One way to determine if data confirm to these assumptions is the **graphical data analysis** with R because a graph can provide many insights into the properties of the plotted dataset.
- R has extensive facilities for **producing graphs**.
- There are both **low and high** level graphical facilities.
- The low level graphical facilities provide basic **building blocks** which can be used to build up graphs step by step.
- While the high level facilities provide a variety of pre-assembled graphical displays.

Plotting with Base Graphics

Line plots

- A Line chart is a graph that connects a series of points by drawing line segments between them.
- Line chart are usually used in identifying the trends in data.
- The plot() or lines() function in R is used to create the line graph.
- Syntax: plot(x,type,col,xlab,ylab,main)

  where, x : vector containing numerical values

  type : takes value "p" to draw only the points,

  "l" to draw only the lines and

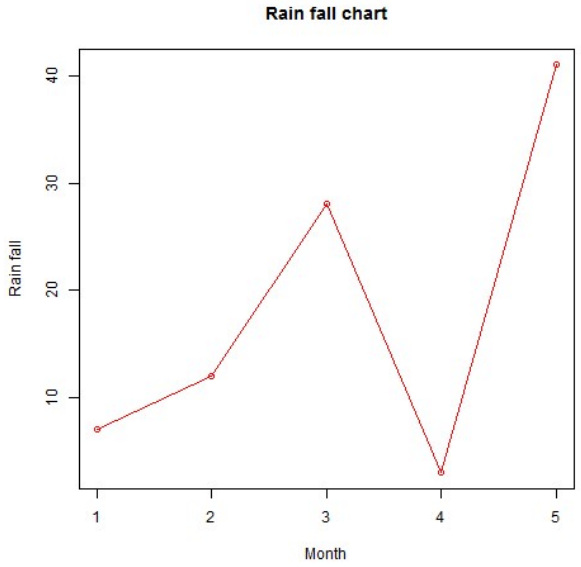
  "o" to draw both points and lines

  col : to give colors to both the points and lines

  xlab : the label for x axis

  ylab : the label for y axis

  main : the title of the chart



- Example: x = c(7,12,28,3,41)

plot(x,type="o",col="red",xlab="Month",ylab="Rain fall",main="Rain fall chart")

Bar plots

- A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable.
- R uses the function barplot() to create bar charts.
- R can draw both vertical and horizontal bars in the bar chart.
- Syntax: barplot(x, xlab,ylab,main,names.arg,col,border,horiz)

  where, x : vector or matrix containing numerical values

  xlab : the label for x axis

  ylab : the label for y axis

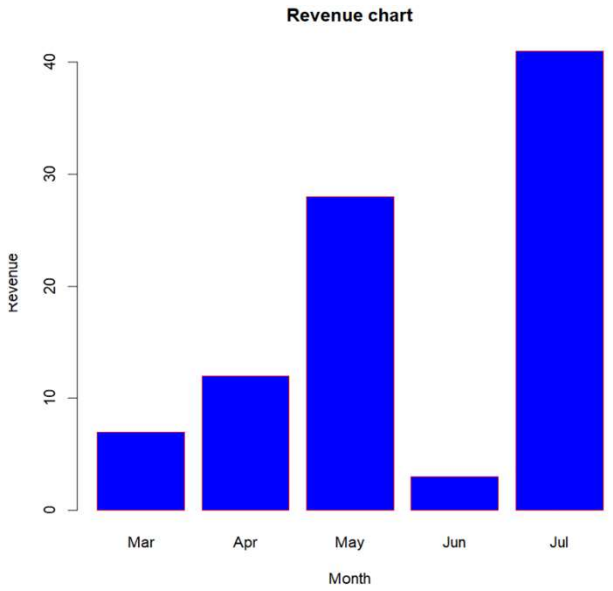
  main : the title of the chart

  names.arg : vector of names appearing under each bar

  col : to give color to the bars

  border : to specify border color

  horiz : contains TRUE /FALSE value



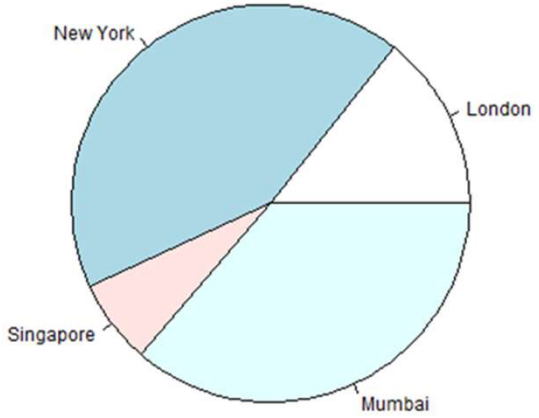
- Example: x = c(7,12,28,3,41)

  M = c("Mar","Apr","May","Jun","Jul")

barplot(x,names.arg=M,xlab="Month",ylab="Revenue",col="blue",main="Revenue chart",border="Red")

Pie Chart

- A pie chart is a representation of values as slices of a circle with different colors.
- The function pie() is used to plot pie chart.
- Syntax: pie(x,labels,radius,main,col,border,clockwise)
  where, x : vector containing numerical values
  labels : to give description to the slices
  radius : indicate the radius of the circle of the pie chart
  main : the title of the chart
  col : to set color to the palette
  border : to specify border color
  clockwise : logical value indicating slices are drawn clockwise or anti clockwise



Example: x=c(21,62,10,53)
ls=c("London", "New York", "Singapore","Mumbai")
pie(x,ls,main="City Pie chart", col=c("red,","Green","Blue","Yellow"),radius = 2.0)

Table plot

- Plotting graphs from tabular data is commonly done in data analytics.
- It gives a better interpretation of the tabular data and gives insights into interesting properties related to the data.
- Syntax: plot(table_name,type,ylim,lwd,xlab,ylab,main)

where, type : plotting type (p-points, s-stair steps, l-lines, h-histogram, b- points joined by lines)

ylim : range of y axis

lwd : line width for bars

xlab : x axis label

ylab : y axis label

main : the title of the chart

Example: x=rnorm(100)

t = table(x)

plot(t,lwd=3,xlab="Year",ylab="Population",main="Table Plot")

Histogram

- Histogram is a type of bar chart which shows the frequency of the number of values which are compared with a set of values ranges.
- The function hist() is used to plot histogram.
- Syntax: hist(v,main,xlab,xlim,ylim,breaks,col,border)

  where, v : vector containing numerical values

  main : the title of the chart

  xlab : the label for x axis

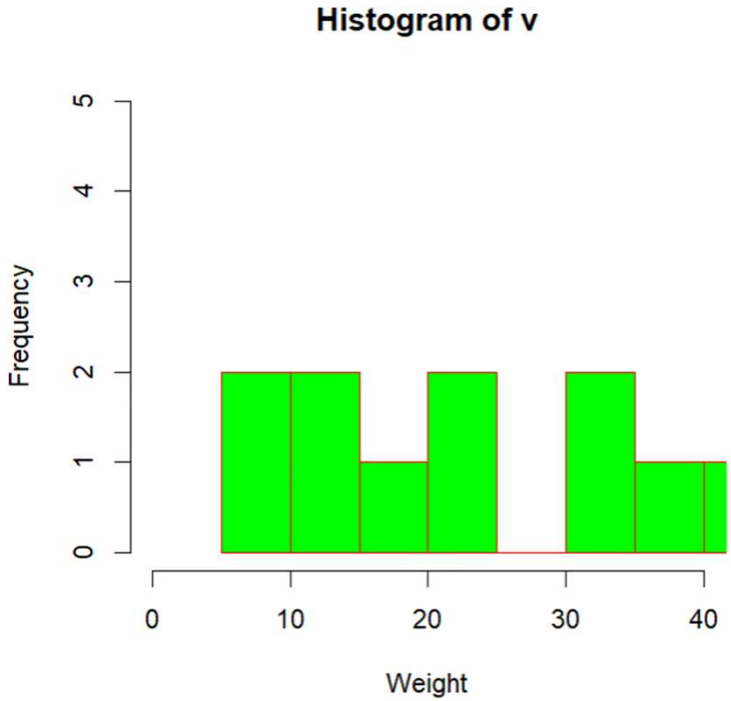
  xlim : to specify the range of values on the x-axis

  ylim : to specify the range of values on the y-axis

  breaks : used to mention the width of each bar

  col : to set color to the bars

  border : to specify border color



Example: v=c(9,13,21,8,36,22,12,41,31,33,19)

hist(v,xlab="Weight",col="green",border="red",xlim=c(0,40),ylim=c(0,5), breaks=5)

Density Plot

Example:

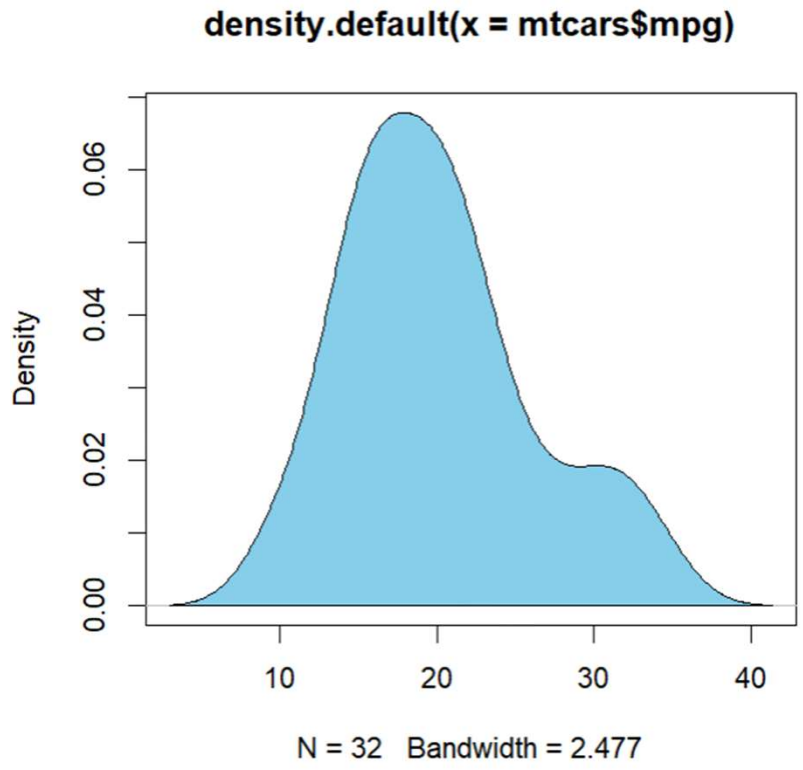
#mtcars is Dataset from R and mpg is a field of mtcars dataset

d<-density(mtcars$mpg)

d

plot(d,col="red")

polygon(d,col="skyblue",border="black")



Plotting with Lattice Graphics

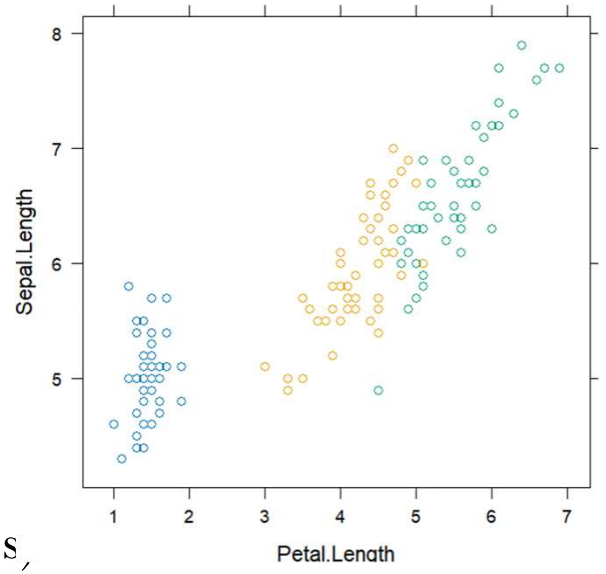
- **Lattice** is a powerful and high level **data visualization package** in R.
- It is designed with an emphasis on **multivariate data** and in particular allows easy conditioning to produce **small multiple plots**.
- The lattice package is an implementation of the multi-panel used to visualize the **dependencies and interactions** between multiple variables.
- It provides the ability to display **multivariate relationships** and it improves on the base R graphics.
- install.packages("lattice")

  library("lattice")
- graph_type(formula,data=)

Scatter Plot / Time series plot

- The function **xyplot()** is used to produce **scatter plots** or **time-series plots**.
- Syntax : xyplot(y~x,data)
- Example:

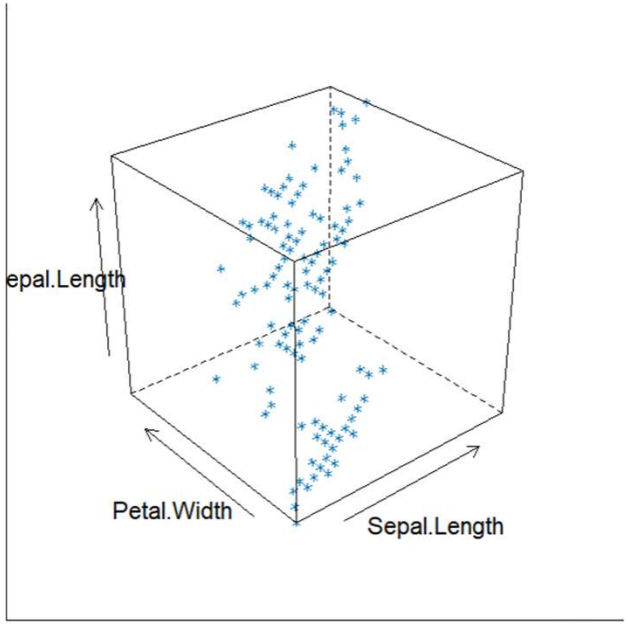
```
install.packages('lattice')
library('lattice')
mydata=iris #iris is R dataset
xyplot(Sepal.Length~Petal.Length,data=mydata)
#Color by group
xyplot(Sepal.Length~Petal.Length,data=mydata,group=Species,
#To show the points, grid and smoothing line
xyplot(Sepal.Length~Petal.Length,data=mydata,group=Species,
type=c("p","g","smooth"))
```



3D Scatter Plot

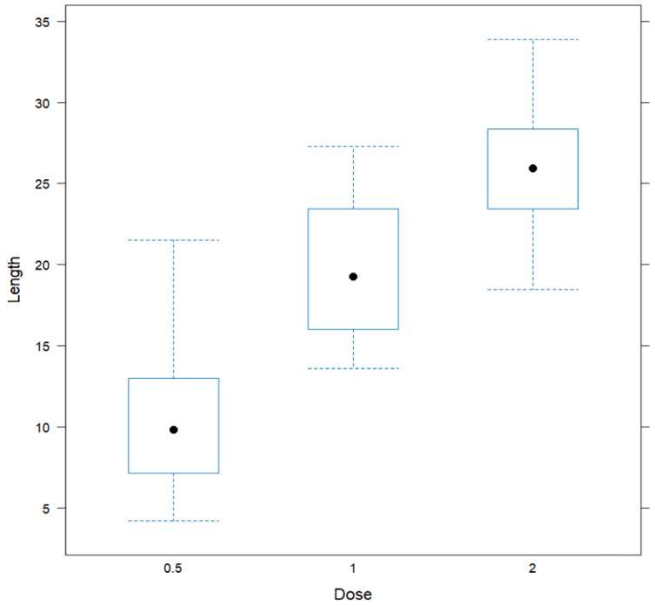
- The function **cloud()** is used to produce **3D scatter plots.**
- Syntax : cloud(y~x,data)
- Example:

```
install.packages('lattice')
library('lattice')
cloud(Sepal.Length~Sepal.Length*Petal.Width,data=iris)
```



Box Plot

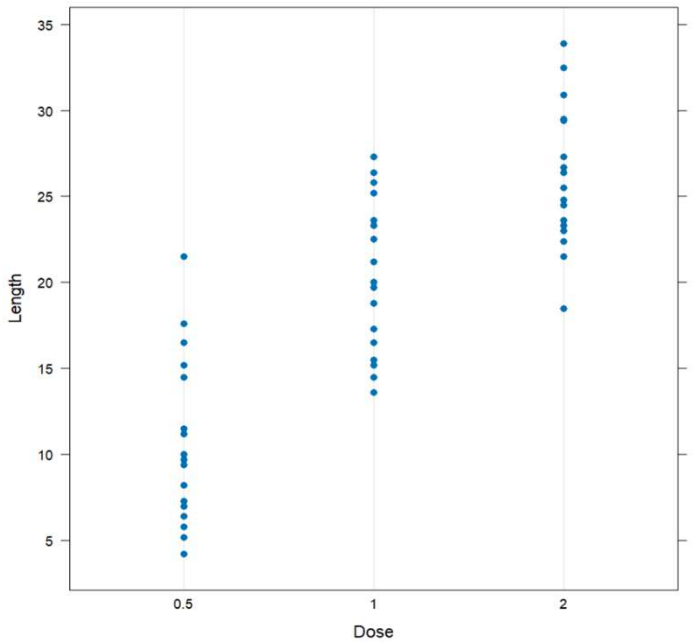
- The function **bwplot()** is used to produce **box plots.**
- Syntax : bwplot(y~x,data)
- Example:




```
install.packages('lattice')
library('lattice')
ToothGrowth$dose<-as.factor(ToothGrowth$dose)
head(ToothGrowth)
bwplot(len~dose,data=ToothGrowth, xlab="Dose",ylab="Length")
```

Dot Plot

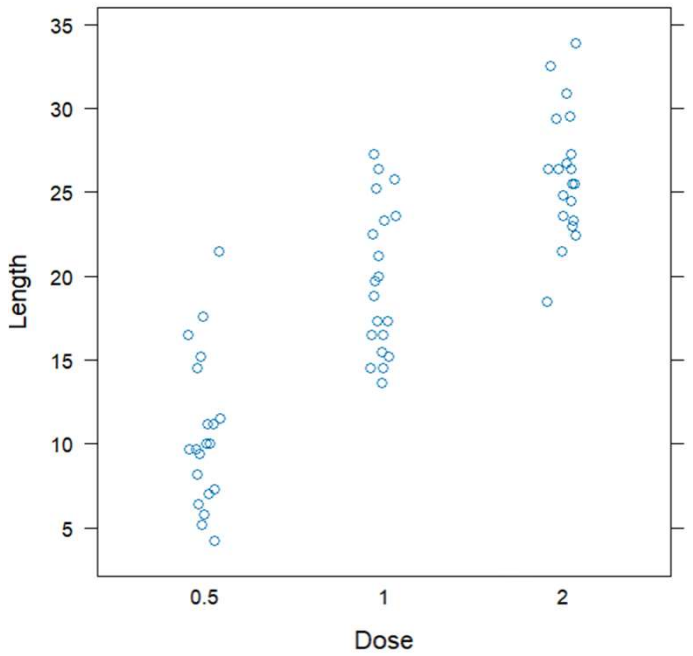
- The function **dotplot()** is used to produce **dot plots.**
- Syntax : dotplot(y~x,data)
- Example:




```
install.packages('lattice')
library('lattice')
ToothGrowth$dose<-as.factor(ToothGrowth$dose)
head(ToothGrowth)
dotplot(len~dose,data=ToothGrowth, xlab="Dose",ylab="Length")
```

Strip Plot

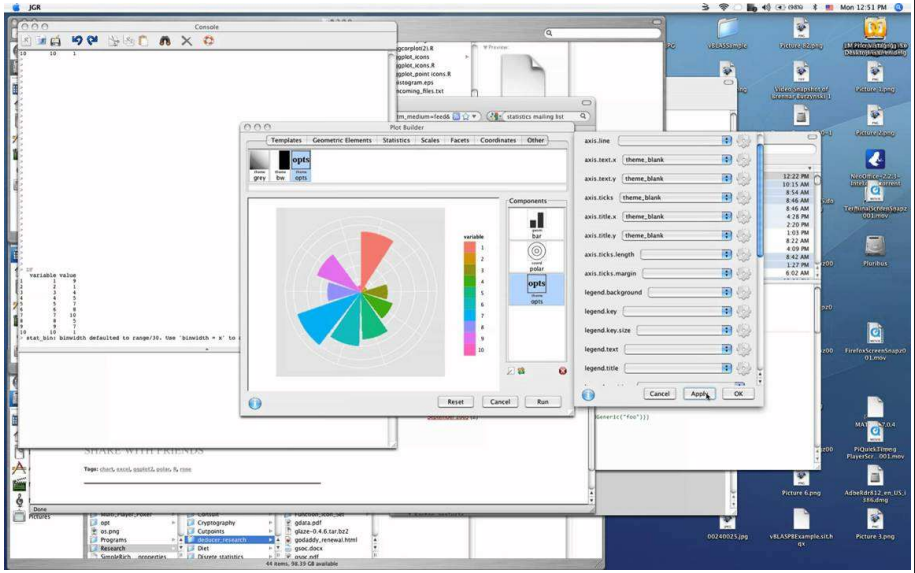
- The function **stripplot()** is used to produce **strip plots.**
- Syntax : stripplot(y~x,data)
- Example:




```
install.packages('lattice')
library('lattice')
ToothGrowth$dose<-as.factor(ToothGrowth$dose)
head(ToothGrowth)
stripplot(len~dose,data=ToothGrowth, xlab="Dose",ylab="Length",jitter.data=TRUE)
```

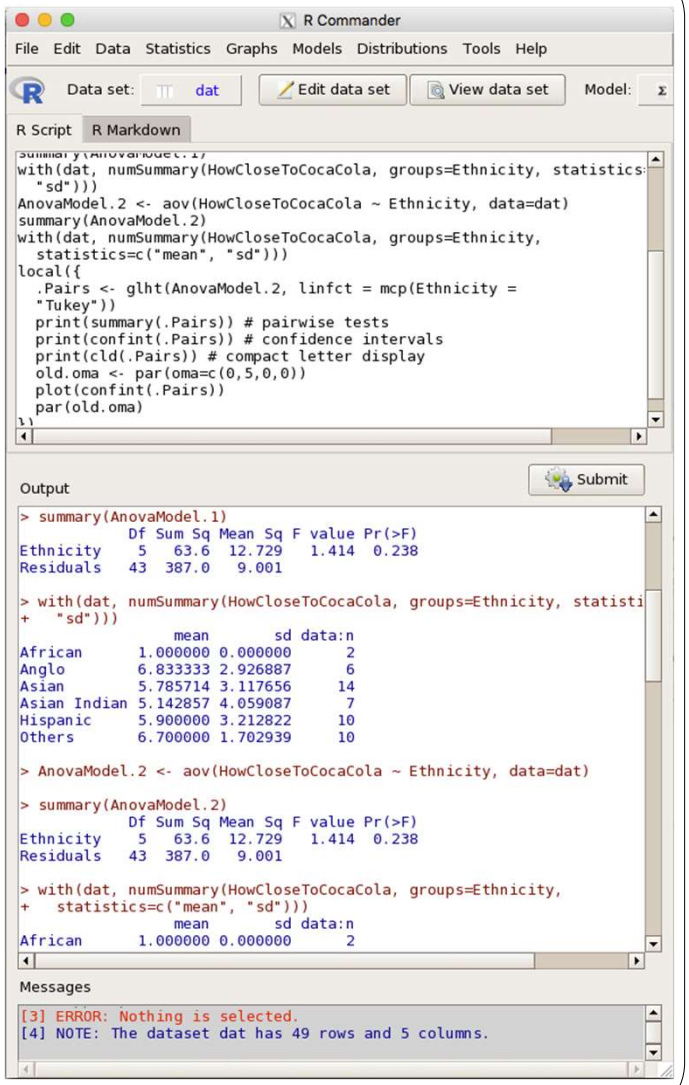
R Deducer

- **R** is the actual software environment that all the data analysis is operated.
- **RStudio** is an integrated development environment (IDE) for R.
- **Deducer** is a free and open source Graphical User Interface (GUI) for R software that provides **beginners** a way to point and click their way through **analysis**.
- **Deducer** is a cross-platform graphical data analysis system.
- It uses **menus and dialogs** to guide the user efficiently through the data manipulation and analysis process.
- It also has an excel like **spreadsheet** for easy data frame visualization and editing.



R Commander (Rcmdr)

- **R Commander** is a GUI for the R programming language, licensed under the GNU General Public License.
- It looks and works similarly to **SPSS GUI** by providing menu to analytics and graphical methods and display for each analysis run the underlying R code.
- R Commander is used as a suggested learning environment for a number of R-centric academic statistics **students and scientists**.
- R Commander is a graphical user interface for R, a statistical environment comparable to **SPSS or SAS**.
- R commander was developed as an **easy to use** graphical user interface (GUI) for R (freeware statistical programming language) and was developed by **Prof. John Fox** to allow the teaching of statistics courses and removing the barrier of software complexity from the process of learning statistics.



Introduction to Spatial Analysis

- **Spatial analysis** is a type of **geographical analysis** which seeks to explain patterns of human behavior and its spatial expression in terms of mathematics and geometry, that is **locational analysis**.
- New methodologies of spatial analysis include **geocomputation and spatial statistical** theory.
- Spatial analysis allows you to solve **complex location oriented problems** and better understand where and what is occurring in your world.
- Spatial analysis or spatial statistics includes any of the formal techniques which studies entities using their **topological, geometric or geographic properties**.

Thank you