Probability and Statistics with R Software

Installation and working with R software

Dr. Pooja Bansal

Guru Gobind Singh Indraprastha University

May 17, 2022

What is R

- R is a software for data analysis, statistical computing, data manipulation.
- simple as well as complicated calculations are possible.
- R is a free (open source) software, therefore its not a black box.
- R supports many free packages which helps the data scientists and analysts.
- It have many built in packages and it allows for coding as well.
- You can prepare your own packages and contribute.
- The commands can be saved, run and stored in script files.
- Graphics can also be stored in jpg, png, pdf etc. formats.

Installing Packages and Libraries

- The base R package contains programs for basic calculation.
- It does not contain some of the libraries for advance statistical work.
- Specific requirements are met by downloading special packages.
- Examples: To install the package "ggplot", run the following command on R console
  >install.packages("ggplot2")
  This command installs package ggplot2
- After downloading the package run the following command
  >library(ggplot)
  to load the package in R.

Installing Packages and Libraries

- There are some base packages which are not required to be installed. You just need to load the library
  Example: "MASS" package is in base package. Run the following command to load it
  >library(MASS)
- For help regrading any package run the following command
  >library(help=MASS)

Cleaning Up Windows

- Command for removing variable x:

  ```
  >rm(x)
  ```

  To remove all variables rm(list=ls())

- Example:

  ```
  >x=10
  >y=3
  >z=13
  >rm(x) #will removes the variable x
  >rm() #will remove all the variables
  ```

- To clear the Console screen, press "Ctrl+L" key on keyboard.

- To quit R software press

  ```
  >q()
  ```

Working with R studio

- Make coding and execution of the programs easier.
- We have four windows which help in execution of different things.
- Window 1: Script file, where we write our commands
  Window 2: Console, where the commands are executed
  Window 3: Contains History, variable values, helps in downloading data.
  Window 4: Show graphics, help in installing packages and library, information about a package etc.
- Example

```
>x=1
>y=2
>z=x+y
>z
>plot(x,y)
```

Calculations with R as Calculator

- Assignment operator are "$=$" and "$<-$". Example:

  $>$x$=$1

  $>$x$<-$1

  Both will assign value 1 to x. Initially only $<-$ was the assignment operator.
- "#" this is symbol for commenting, i.e., the command which we don't want to execute.
- $>$y$=$3*x #assigns 3x to y

  $>$z$=$x-y #assigns x-y to z
- "c(1,2,3,4)" combined the number 1,2,3,4 to a vector. Example:

  $>$w$=$c(1,2,3,4)# w is vector with 4 values. To use different values of w, we use w(1), w(2), w(3) and w(4).

  w(5) will give you error.
- No command like w$=$1,2,3,4 exists. It will give error.

Calculations with R as Calculator

- Blank space will not make any difference in commands.

  Example

  ```
  >w = c(1,2,3,4)
  ```

- Capital and small letters are different. Example:

  ```
  > x=20
  > X=15
  ```

  are two different variables.

- Basic operations:

  ```
  > 2+5 #addition
  > 6*7 #multiplication
  > 7-3 #subtraction
  > 10/2 #division
  > 2^3 #power
  > 2**3 #power
  ```

Frame Title

- For root, take half power
  >2^0.5
  > 2**0.5
- Negative powers are also possible. Example $\frac{1}{2^{0.5}}$ is
  >2^-0.5
- Note: Use brackets while using powers in fraction form. As multiple operators are solved using the rule of BODMAS Example
  >2^1/2 #gives ans 1 not 1.4142

Calculations with data Vectors

- Addition of a vector with constant

  >c(1,2,3,4)+6

  All the vector components will be increased by 6.
- Addition in data vectors, when data vectors have same number of elements

  >c(1,2,3,4,5)+c(5,6,7,8,9)
- when one data vector have number of elements which are multiple of number of elements of the other data vector.

  >c(1,2,3,4)+c(7,8)
- when no data vector has number of elements as multiple of number of elements of the other data vector, a warning message will appear

  >c(1,2,3,4)+c(7,8,9)
- The operation of subtraction, multiplication, division and power happens in similar manner.

Calculations with data Vectors

- Rule of BODMAS is applies in data vectors. Example:
  ```
  >c(1,21,3,4)*c(5,6)/c(8,9)-c(4,5,6,7)
  ```
- Maximum of a vector
  ```
  >max(2,5,7,-4)
  ```
- Minimum of a vector
  ```
  >min(2,5,7,-4)
  ```
- Absolute value
  ```
  > abs(-2)
  ```
- Square root
  ```
  >sqrt(4)
  ```
- Rounding off a number
  ```
  >round(2.7)
  >round(3.4)
  ```
- Floor
  ```
  >floor(2.4)
  ```

Frame Title

- Ceiling
  >ceiling(2.4)
- Sum of elements of vector
  >sum(1,2,3,4)
- Product of elements of vectors
  >prod(1,2,3,4)
- Other functions:
  log(),log10(),log2(),exp(),sin(),cos(),tan(),asin(),acos(),atan() etc.

Plots

- x,y are two data vectors, then plot(x,y) command plots y corresponding x.
- plot(x,y,type) will give a line plot, point plot, etc based on type chosen.
  type=”p”, the graphic is in form of point
  type=”l”, the graphic is in form of line
  type=”h”, the graphic is in form of histogram
  type=”s”, the graphic is in form of a stair case
- Add headings by ”main=”, subheadings by “sub=”, title for x-axis by “xlab=”.title for y-axis by “ylab=”

Counting Principal

- Binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

in R is written as

$>$choose(n,m)

$> factorial(n)/(factorial(m) * factorial(n-m))$

Example

- Marks out of 500 maximum marks
  marks $< -$c(337,316,334,327,340,360, 374,330,352, 353,370,380,384,398,413,428,430,438,439,450) Number of hours per week
  hours$< -$c(23,25,25,26,27,28,30,26,29,32,33,34, 35,38,39,42,43,44,45,45.5)
  Plot with marks on y-axis corresponding to hours on x-axis with labels and headings and with different line types.
- plot(hours,marks), plot(hours,marks,type="p"), plot(hours,marks,type="p", xlab="hours", ylab="marks")etc.

Counting Principal

- Binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

in R is written as

$>$choose(n,m)

$> factorial(n)/(factorial(m) * factorial(n-m))$

Relative Frequency and sampling

- Throwing a die 100 times:
  >sample(c(1,2,3,4,5,6))
  will draw randomly from 1,2,3,4,5,6 without replacement 6 times.
- To draw a sample of size 2, run the command
  >sample(c(1,2,3,4,5,6),size=2)
- The command >sample(c(1,2,3,4,5,6),size=7)
  will give an erros as this sampling is without replacement.
- For with replacement sampling, run
  >sample(c(1,2,3,4,5,6),size=7, replace=TRUE)

Probability

- Roll a dice 100 times and store the values
  >dice100=sample(c(1,2,3,4,5,6),size=100,replace=TRUE)
- length(dice100) will give length of the vector dice100
- length(dice100[(dice100==2)]) will give count of 2's
- length(dice100[(dice100==2)])/length(dice100) will give probability of 2
- Probability distribution table:
  >table(dice100)/length(dice100)
- Repeat with 1000, 10000 size

Probability A ∪ B and A ∩ B

- To find probability of occurrence of 2 or 6,
  > length(dice100[(dice100==2) | (dice100==6)])/length(dice100)
- For A ∩ B,
  length(dice100[(dice100==A) & (dice100==B)])/length(dice100)
- Toss a coin random experiment can be generated by considering occurance of head as 1 and tail as 0
  >sample(c(0,1), size=100, replace="TRUE")
  >sample(c("H","T"), size=100, replace="TRUE")

Sample Statistics

- Histogram of distribution:
  >hist(x)
- Mean:$= \frac{1}{n}\sum_{i=1}^{n} x_i$
  >mean(x)
- Variance:$= \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2$
  >var(x)
- Standard Deviation:
  >sd(x)
- Skewness: $= \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{3/2}}$
  >skewness(x)
- Kurrtosis:$= \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{2}}$
  >kurtosis(x)

Moments

- Install package "moments"
- Central moments: $\mu_r = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^r$
  when $r = 0$, $\mu_0 = 1$
- Non-central moments: $\mu_r^{'} = \frac{1}{n}\sum_{i=1}^{n}(x_i)^r$
  when $r = 0$, $\mu_0 = 1$
- For all moments
  $>$all.moments(x,order.max=2,central=TRUE)
  Default moments upto order 2, i.e., 0,1,2, are given and non-central moments are given if central=TRUE not specified.

Discrete Uniform Random Number

- Package “purrr” is required for generation of discrete uniform random variable.
- To generate random samples from discrete uniform random variable

  $>$ rdunit(n,b,a)
- Find mean, variance by mean(x), var(x)

Discrete Uniform Random Number

- Package “purrr” is required for generation of discrete uniform random variable.
- To generate random samples from discrete uniform random variable

  $>$ rdunif(n,b,a)
- Find mean, variance by mean(x), var(x)
- The command table() prepares the frequency table and table()/length() prepares the probability distribution table.