Shraddha Manchekar

UID: 004945217

Memo

R

R is an open source programming language which is widely used among statisticians and data miners for developing statistical software and data analysis.

Features:

- Statistical features: R and its libraries implement a wide variety of statistical and graphical techniques, including linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering and others.
- R is highly extensible and for computationally intensive tasks C and C++ code can be linked and called at the run time.
- Also, R can produce publication quality graphs, including mathematical symbols.

Tips and tricks:

Download and install a package from CRAN → install.packages('package name')
Load a package into the session → library(packagename)
Find the current working directory → getwd()
Change the current working directory → setwd('path')

Vector Functions

- Sort(x) → return x sorted
- Table(x) → see counts of values
- Rev(x) → return x reversed
- Unique(x) → see unique values

Selecting vector elements

- X[4] → fourth element
- X[-4] → all but fourth element
- X[2:4] → elements 2 to 4
- X[c(1, 5)] → elements one and five
- X[x==10] → elements which are equal to 10
- X[x %in% c(1, 2, 5)] → elements in the set 1, 2, 5

Programming

For Loop

```
for (variable in sequence){
   Do something
}
```

Example

```
for (i in 1:4){
   j <- i + 10
   print(j)
}
```

While Loop

```
while (condition){
   Do something
}
```

Example

```
while (i < 5){
   print(i)
   i <- i + 1
}
```

If Statements

```
if (condition){
   Do something
} else {
   Do something different
}
```

Example

```
if (i > 3){
   print('Yes')
} else {
   print('No')
}
```

Functions

```
function_name <- function(var){
   Do something
   return(new_variable)
}
```

Example

```
square <- function(x){
   squared <- x*x
   return(squared)
}
```

Statistics

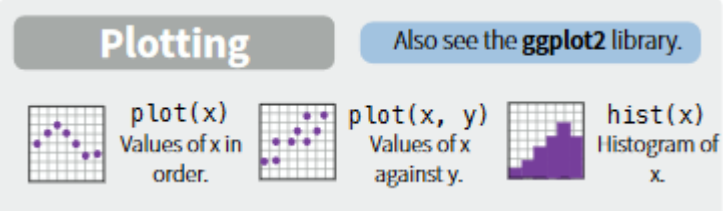
- lm(x~y, data = df) → linear model
- glm(x~y, data = df) → generalized linear model
- summary → get more detailed information out of a model

Distributions

Distributions

	Random Variates	Density Function	Cumulative Distribution	Quantile
Normal	rnorm	dnorm	pnorm	qnorm
Poison	rpois	dpois	ppois	qpois
Binomial	rbinom	dbinom	pbinom	qbinom
Uniform	runif	dunif	punif	qunif

Plotting



R Studio

RStudio is a free and open-source integrated development environment (IDE) for R.

Tips and tricks:

- Download: r-project.org
- Rstudio highlights the opening parenthesis or quotation mark when the cursor is placed directly following the closing parenthesis or quotation mark
- It is possible to directly copy and paste a plot image directly from the zoomed plot without saving the image first.
- Keyboard Shortcuts in R Studio.
    - Ctrl+Enter: to run current line or selection
    - Ctrl+Shift+S: to source the current file
    - Ctrl+Shift+F10: to restart the R session
    - Shift+F9: to toggle a breakpoint
    - F10: to execute next line
    - Shift+F4: to step into a function
    - Up/down arrow keys: to navigate the command history
    - Command history: Ctrl + up
    - Comment a block of code: Ctrl + Shift + C
    - Autocomplete: Tab

Rcpp

Rcpp is very easy to learn and use. It helps in improving the speed of a variety of tasks especially loops and function calls.

Tips:

- Add '#include<Rcpp.h>' to tell the computer to include the C++ library.
- 'using namespace Rcpp' tells the computer to look up any unknown commands in the Rcpp library.
- In Rcpp code, most of the statements end in a semi-colon.
- To add comments to our code, double forward slash should be used.
- [[Rcpp::export]] is a very important line of code that is added to a function to export it to our R environment so that we can call it from our R code.

- To source a cpp file, either select ‘source file’ option available in Rstudio or add the following code to the R file.
    - library(Rcpp)
    - sourceCpp (' File Name . cpp ')
- Rcout writes whatever follows to the console.
    - rcout<<”Hello world!”;
- std::endl is a useful command to jump to the next line while printing on screen.
- The most common Rcpp classes are the "NumericVector" and"NumericMatrix" classes.
- NumericVector v[10]: creates a numeric vector of size 10 with all elements as 0
- v.size(): returns the size of the numeric vector
- NumericMatrix mat(2,3): creates a matrix with 2 rows and 3 columns with 0 values
- Mat(1,2)=9: sets the element at row 2 and column 3
- mat.nrow(): returns number of rows in matrix
- mat.ncol(): returns the number of columns in matrix
- zeros<mat>(2,2): to create a 2x2 zero matrix
- join_rows(A, B): to combine the elements of two matrices row wise
- join_cols(A, B): to combine the elements of two matrices column wise

R parallel

Parallel computing in R is particularly useful If we want to solve larger problems that involve a lot of computation or if we want to work on something concurrently to save time.

Tips:

- R provides 2 packages for parallel computing: parallel and doparallel
- First, we use ‘makecluster’ method to create a cluster and allocate the number of cores available in the system.
- Then, we register will a parallel backend to use the parallel functionality using ‘registerDoParallel’
- The foreach package available in ‘doparallel’ allows the running of for loops in parallel. The %dopar% operator is used to specify that tasks should run in parallel.
- Also, it is important to close the cluster at the end of execution step so that core memory is released.

Example:

```
## Setup parallel backend to use multiple processors
cl <- makeCluster(6)
registerDoParallel(cl)
## Start program
results <- foreach(i = 1:numIters, .combine = cbind, .packages =
c("nloptr")) %dopar% {

  x   <- iris[which(iris[,5] != "setosa"), c(1,5)]
  ind <- sample(100, 100, replace=TRUE)
  result1 <- glm(x[ind,2]~x[ind,1], family=binomial(logit))
```

```
  ## Save results
  res[i]  <- coefficients(result1)

  ## Combine all of our output into a list
  result <- list(mean1 = coefficients(result1))
  result
}
## Stop the cluster manually
stopCluster(cl)
```

Python

Python is a widely used high-level programming language for general-purpose programming. It is an interpreted language which emphasizes on code readability and it's syntax allows programmers to express concepts in a fewer lines of codes.

Features:

- Easy-to-learn
- Easy-to-read
- A broad standard library support
- Object-oriented
- Scalable
- Portable

Tips and tricks:

Methods useful while working with a list

- Append(item)
- Count(item)
- Extend(list)
- Index(item)
- Insert(position, item)
- Pop(position)
- Remove(item)
- Reverse()
- Sort()

String methods

- Index(sub, start, end)
- Find(sub, start, end)
- Count(sub, start, end)
- Split(sep)
- Splitlines()
- Startswith(sub)

- Indexes and slices of a =[0, 1, 2, 3, 4, 5]
- Len(a) – 6
- a[-1] – [5]
- a[1:] – [1, 2, 3, 4, 5]
- a[: 5] – [0, 1, 2, 3, 4]

Reversing a string

- >>> a = "python"
  >>> print "Reverse is",a[::-1]
  Reverse is nohtyp

Store variables of a list in different variables

- >>> a = [1, 2, 3]
  >>> x, y, z = a
  >>> x
  1
  >>> y
  2
  >>> z
  3

Create a single string from all the elements in list above.

a = ["Code", "mentor", "Python", "Developer"]

```
>>> print " ".join(a)
Code mentor Python Developer
```

Swap two numbers with one line of code.

```
>>> a=7
>>> b=5
>>> b, a =a, b
>>> a
5
>>> b
7
```

SQL

SQL is a domain-specific language used in programming and designed for managing data held in relational database management system. SQL offers two main advantages: first, it introduced the concept of accessing many records with one single command; and second, it eliminates the need to specify *how* to reach a record, e.g. with or without an index.

Commands:

- CREATE DATABASE database_name – *create a database*

- DROP DATABASE database_name – *delete a database*
- SELECT column_name(s) FROM table_name – *select data from a table*
- SELECT * FROM table_name - *Select all data from a table*
  SELECT DISTINCT column_name(s) FROM table_name - *Select only distinct (different)* data from a table.
- SELECT column_name(s) FROM table_name
  WHERE column operator value
  AND column operator value
  OR column operator value
  AND (... OR ...) - *Select only certain data from a table.*
- SELECT column_1, ..., SUM(group_column_name)
  FROM table_name
  GROUP BY group_column_name
  HAVING SUM(group_column_name) condition value - HAVING... was added to SQL because the WHERE keyword could not be used against aggregate functions (like SUM), and without HAVING... it would be impossible to test for result conditions.
- SELECT column_1_name, column_2_name, ...
  FROM first_table_name
  INNER JOIN second_table_name
  ON first_table_name.keyfield = second_table_name.foreign_keyfield - The INNER JOIN returns all rows from both tables where there is a match. If there are rows in first table that do not have matches in second table, those rows will not be listed
- SELECT column_1_name, column_2_name, ...
  FROM first_table_name
  LEFT JOIN second_table_name
  ON first_table_name.keyfield = second_table_name.foreign_keyfield – *The LEFT JOIN returns all the rows from the first table, even if there are no matches in the second table. If there are rows in first table that do not have matches in second table, those rows also will be listed.*
- SELECT column_1_name, column_2_name, ...
  FROM first_table_name
  RIGHT JOIN second_table_name
  ON first_table_name.keyfield = second_table_name.foreign_keyfield - *The RIGHT JOIN returns all the rows from the second table, even if there are no matches in the first table. If there had been any rows in second table that did not have matches in first table, those rows also would have been listed.*
- SQL_Statement_1
  UNION
  SQL_Statement_2 - *Select all different values from SQL_Statement_1 and SQL_Statement_2*
- CREATE TABLE "table_name"
  ("column_1" "data_type_for_column_1",
  "column_2" "data_type_for_column_2",
  ...) - *Create a table in a database.*

Unix/Linux

- ls: to list contents of current working directory
- ls -a: to list all files in the directory including the hidden ones
- ls also supports a lot of other options for listing files in a directory.
- mkdir: to create a directory for holding files or other directories
- cd: to change the current directory to the specified one
- In unix, dot (.) means current directory, double dot (..) means parent directory and tilde (~) refers to home directory.
- pwd: to display the path of current working directory
- cp file1 file2: to copy file1 in the current directory to file2
- mv file1 file2: to rename file1 to file2 or to move file1 to another place
- rm and rmdir can be used to remove a file and directory respectively.
- cat: to display contents of a file on screen
- head: to write first few lines of a file to screen
- tail: to write last few lines of a file to screen
- grep keyword file: to search for keywords or patterns in a file
- wc file: to count number of words, characters or lines in a file
    - ➢ symbol can be used to redirect output of a command to a file
- >> symbol can be used to append standard output to a file
- < symbol can be used to redirect standard input from a file
- Pipe (|) symbol is used to pipe the output of one command to the input of another
- Sort command is used to sort contents of a list alphabetically or numerically.
- who: to view list of currently logged in users
        - is a wildcard character used to match zero or more characters in a file name.
- ? is a wildcard character used when you want to match just one character.
- To read online manual of a command, man command should be used.
- ls -l: to list access rights for all files in a directory
- chmod [options] file: to change access rights for the specified file
- ps: to list currently running processes on system
- jobs: to list currently running jobs
- sleep command can be used if you want to wait for a certain number of seconds before continuing
- command &: to run the command in the background
- kill pid: to kill a process with id as pid
- df: to find the amount of space left on the file system
- du: to list number of kB used by each subdirectory
- gzip command is used to compress a file for saving space.
- To search for files with specific attributes, find command can be used. It allows searching for files and directories based on name, date, size, etc.

Github

GitHub is a Web-based Git version control repository hosting service. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features.

Commands:

Configure tooling:

git config –global user.name "[name]" – sets the name you want attached to your commit transactions.

git config –global user.email "[email address]" – sets the email you want attached to your commits

git config –global color.ui auto – enables helpful colorization of command line output

Creating Repositories:

Git init [project.name] – creates a new local repository

Git clone [url] – downloads a project and its entire version history

Make changes:

Git status – lists all new or modified files to be committed

Git diff – shows the file differences

Git add [file] – snapshots the file in preparation for versioning

Git commit -m "[message]" – records the file snapshots permanently in version history

Group changes:

Git branch – lists all local branches in the current repository

Git branch [branch-name] – creates a new branch

Git checkout [branch name] – switches to the specified branch

Git merge [branch] – combines the specified branch's history into the current branch

Git branch -d [branch] – deletes the specified branch

Synchronize changes"

Git fetch [bookmark] – downloads all history from the repo bookmark

Git merge [bookmark]/[branch] – combines bookmarks's branch into current local branch

Git push [alias][branch] – uploads all local branch commits to github

Git pull – downloads bookmark history and incorporates changes.

Hoffman

Hoffman2 is UCLA's supercomputing cluster. It is the largest and most powerful cluster in the UC system. It is vital if you have huge data and/or heavy-duty computation. Hoffman2 is managed and operated by the IDRE Research Technology Group at UCLA. It has 1,200+ 64-bit nodes and 13,340 cores, over 50TB of memory

Getting a Hoffman2 account:

- Need to register with UCLA IDRE (https://www.hoffman2.idre.ucla.edu/getting-started/)
- Need a faculty sponsor (usually your advisor)
- IDRE gives workshops fairly frequently on using Hoffman

Connecting to Hoffman2:

- ***Unix, Linux, Mac users:***
  Use the 'ssh' command in the terminal to log in:
  ssh login-id@hoffman2.idre.ucla.edu
- ***Windows users***
  Can use an SSH client like "MobaXterm", "PuTTY", etc.
  Can use remote desktop "NoMachine"

Commands:

- Every user has a home directory with 20GB storage for general use
  - Type 'myquota' to see how much storage you have remaining
- You also have more storage for temporary use
  - 2TB in '/u/scratch'; keep for 7 days
- To use R on Hoffman2, type R.
- Similarly, type 'python' to use python on Hoffman2.

- Using interactive sessions:
  - Useful for doing more intensive computations
  - To request a compute node for interactive use, use the 'qrsh' command:
    qrsh -l h_rt=8:00:00,h_data=4G
    h_rt: Requested running time (8 hours).
    h_data: Requested memory (4GB).

- Submitting a job:
  - To submit a job to Hoffman2, use the 'qsub' command:
    qsub -N job1 -l h_data=8G,h_rt=23:00:00 -m bea thejob.sh
    -N job1: Name the job 'job1'. Can see the name when you type 'myjobs'
    -m bea: Define mailing rules. Email me when job '**b**'egins, '**e**'nds, and if the job is '**a**'borted

Tensorflow

Tensorflow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library used for machine learning applications such as neural networks. It is developed and used by Google for research and production.

- A tensor is a generalization of vectors and matrices to potentially higher dimensions.
- A tf.Tensor has the following properties:
    - A data type (float32, int32, or string)
    - A shape
- Special types of tensors: tf.Variable, tf.Constant, tf.Placeholder, tf.SparseTensor
- A placeholder is used as a container to send the outside data into our neural network. The data type of the placeholder is defined as tf.placeholder(tf.float32)
- We use session to perform sending and we employ feed_dict={} to designate the variables that will be sent in.
    - For e.g.: with tf.Session() as sess:

        Print(sess.run(output, feed_dict={input1: [7., input2: [2.]}))

- Main classes in Tensorflow:
    - Tf.Graph()
    - Tf.Operation()
    - Tf.Tensor()
    - Tf.Session()
- Some of the activation functions that can be used are relu, relu6, elu, softplus, softsign, dropout, bias_add, sigmoid, tanh, sigmoid_cross_entropu_with_logits, softmax, log_softmax, softmax_cross_entropy_with_logits, etc.
- TensorFlow Optimizers:
    - Tf.get_default_session()
    - Tf.get_default_graph()
    - Tf.reset_default_graph()
    - Ops.reset_default_graph()
    - Tf.device(“/cpu:0”)
    - Tf.name_scope(value)
    - Tf.convert_to_tensor(value)

SAS

SAS is a software suite developed by SAS Institure for advanced analytics, multivariate analyses, business intelligence, data management and predictive analytics. SAS is a software suite that can mine, alter, manage and retrieve data from a variety of sources and perform statistical analysis on it. SAS provides a graphical point-and-click user interface for non-technical users and more advanced options through the SAS language.

Commands:

- The command to import data from a file into a data set in SAS is as follows:
  - proc import out= work.data
  - datafile= "/folders/myfolders/study1/regression_auto.csv"
  - dbms=csv replace; getnames=yes; datarow=2;
  - run;
- Command to compute the correlation between x and y
  - proc corr data=data;
  - var x y;
  - run;
- To make a scatterplot of price (x-axis) and mpg (y-axis), we use sgplot → scatter command
  - proc sgplot data=data;
  - scatter X=price Y=mpg;
  - run;
- Make a box plot of mpg for foreign vs domestic cars;
  - proc sgplot data=data;
  - title 'MPG for Foreign and Domestic Cars';
  - vbox mpg / group=foreign;
  - run;
- Perform simple linear regression, y = mpg, x = price1 without including the intercept term
  - proc reg data=data;
  - title 'Linear Regression, y=mpg, x=price1';
  - model mpg = price1 / noint;
  - run;
- Perform linear regression, y = mpg, x1 = length, x2 = length^2;
  - proc glm data=data;
  - title 'Linear Regression, y=mpg, x1=length, x2=length^2';
  - model mpg = length length*length;
  - run;