Stats-202A

Final Project

Devanshi Patel (UID: 504945601)

R

- R is an open source programming language and software environment for statistical computing and graphics.
- In this course, we used Rstudio as an environment to interact with R.
- It is very easy to get help in R. Just use ?func_name or help(func_name) to display documentation related to the function.
- install.packages(pkg_name) is used to install any package in R.
- library(libname) is used to load the package into the session and make all its functions available for use.
- getwd(): to get the current working directory
- setwd(): to change the current working directory
- The indices of elements in R start from 1 unlike Python.
- Assigning values to variables can be done with <- or =
- It is possible to create vectors of numbers using the function c() which stands for concatenation. For eg: vec <- c(1,2)
- Elements can be accessed using []. For eg: vec[1] will return 1.
- length() returns the vector length. For eg: length(vec) returns 2.
- seq and rep are some useful functions for generating vectors. seq generates numbers in a sequence separated by a difference and rep is used to generate repetition of numbers.
  seq(1, 10, 2) returns 1 3 5 7 9
  rep(c(1,2),2) returns 1 2 1 2
- NULL is the non-existent value in R while NA is the missing place holder.
- It is possible to select subsets of elements using vectors of logicals.
- It is also possible to give names to elements of vectors and then the elements can be accesses using the defined names.
  For eg: vec <- 1:3
  names(vec) <- c("A", "B", "C")
  vec("B") #returns 2
- Matrices are vectors with a number of rows and number of columns attribute and they can be created as follows:
  mat <- matrix(vec, nrow=2, ncol=5)
- t(): to compute the transpose
- solve(): to compute the inverse of a matrix
- norm(a): to compute matrix norm
- sort(a): returns a sorted list of elements
- sum(a): to take sum of all elements in a
- min(a): to return the minimum value in a
- max(a): to return the maximum value in a
- dim(): to get the dimensions of a matrix
- a[1,]: returns the first row of the matrix

- a[-1,]: returns all rows except the first one
- a[,2]: returns the second column of a matrix
- A %*% B: to multiply matrices A and B
- apply allows you to apply a function across a dimension of a matrix. For eg: apply(matrix1, 1 mean) computes mean values across rows
- cbind(a, b): to combine the arguments by rows
- rbind(a, b): to combine the arguments by columns
- Lists are very useful in R for data manipulation tasks. To create a list, use the following code:
  mylist <- list(A=1, B=2, C=3)
- data.frame is the core data type in R to hold different types of data.
  myframe <- data.frame(col1 = vec1, col2 = mat1)
- For loops are available in R to repeat some tasks or iterate through elements.
  1. for(i in 1:length(A)) {
     #dosomething }
  2. for(elem in A) {
     #dosomething }
- R also provides while loops and conditional statements similar to other programming languages.
- Functions can be created as follows:
  myfunc <- function(a) {
  #dosomething
  }
- The variables defined inside the function are local to that function.
- Generating plots in R is equally easy. It is also possible to specify the type of plot, limits of x and y axes, labels, legends and titles.
- plot(x, y): to generate a plot of x vs y
- hist(x): to generate a histogram of frequencies of x
- abline(x, y): to draw a line of slope y and intercept x
- abline(h=y): to draw a horizontal line at ordinate y
- lines(x, y): to add information to an existing plot
- lm(formula): to fit linear models having a certain formula
- glm(formula, family=): to fit generalized linear models specified by giving a symbolic description of the linear predictor and a description of the error distribution; ‘family’ is a description of the error distribution and link function to be used in the model
- rnorm(n, mean=0, sd=1): to generate a Gaussian (normal) distribution
- R also provides a way to create packages so that it is easy to collaborate with other people and to avoid writing the same piece of code again. A package is a convention for organizing files into directories.
- A package stored on disk contains source (actual code), bundle (tar.gz file) and binary (single compresses file for OS).
- Every package in R must have a description file that mentions the contents of the package and its usage.
- After creating a new R package project, the R files should be added to the R directory.
- To load all the files in the memory, use devtools:load_all() function.
- The man directory in package contains the documentation for all the functions. Roxygen package available for R makes it easy to add documentation for each function beside its definition.

- Once comments have been added to functions, run devtools::document() to generate corresponding .rd files and place them in man directory.

RStudio

- Rstudio is the most popular IDE for working with R. It makes it very easy to write, run and debug R code.
- It also has support for version control that allows collaboration on R projects.
- A useful feature provided by Rstudio is the ability to run either a single line or entire file or chunks of code by just selecting it.
- Another feature is that it allows re-running previous code by just clicking on that option. It also allows sourcing contents of the current document with or without echo.
- Code diagnostics are easily understandable and can be viewed by hovering over them in the margin.
- It also supports syntax highlighting and code completion using tabs to finish function names, paths, arguments, etc.
- Rstudio also displays environment variables, data, values and functions on the right pane. This is of great help while debugging the code.
- It also provides a file browser that has been keyed to our working directory. This makes it very convenient to browse among files and select one for opening.
- It also provides a separate section for generated plots, packages and help pages.
- In debug mode, it is possible to establish breakpoints in certain parts of code and when the breakpoint is hit, the execution stops and the user can examine the values of variables. It further provides options like step through, step over, resume and quit.
- Rstudio makes generation of packages extremely easy and time-saving.
- Some of the keyboard shortcuts that I learnt during this course are stated below:
    - 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

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.
- To get started with it, we just need to create a new C++ file in Rstudio.
- We need to 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.
- A major difference in Rcpp code is that 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.
- Thus, all functions should definitely have this line of code if we want to export them to R.
- To source a cpp file, we can either select 'source file' option available in Rstudio or add the following code to the R file.
  library(Rcpp)
  sourceCpp (' Fi le Name . cpp ')
- It is very important to make sure that the cpp file has been saved in our working directory.
- evalCpp() can be used if we want to evaluate a single C++ expression.
- cppFunction() creates, compiles and links a C++ file.
- Following are some common data types in C++:
  - bool (Boolean, i.e., true or false), Bits: 1
  - char (Character, i.e., letter), Bits: 8
  - short (Short integer), Bits: 16
  - int (Integer), Bits: 32
  - float (Floating point number), Bits: 32
  - double (Long float), Bits: 64
  - long double (Very long float), Bits: 80
- The most common Rcpp classes are the "NumericVector" and"NumericMatrix" classes.
- The most common Armadillo classes are "mat" (matrices)and "vec" (vectors). Armadillo is the most popular C++ linear algebra library.
- Rcout is a very popular statement in Rcpp that writes whatever follows to the console.
  For eg: rcout<<"Hello world!";
- std::endl is a useful command to jump to the next line while printing on screen.
- Another important difference from R is that the string indices start from 0 instead of 1.
- We should be careful in assigning values to a variable based on its type. Avoid mixing different datatypes.
- We always need to declare every single variable and specify its type before using it.
- When declaring a function, we need to identify the data types of the parameters that are passed to the function, and the type of variable that our function returns.
- If a function doesn't return a value, we should give it the blank data type void and end the function with simply return;
- We can have our Rcpp function return multiple variables by using the Rcpp data type "List".
- We can also use the "ternary operator", ?:, to implement an if-else statement. In the following code, "if d is less than zero, then, return -1; else if d is greater than 0, return 1; else return 0.
  For eg: double sign(double d){
  return d<0?-1: d>0?1:0
  }
- Thus, if-else statements can be written in a compact way using the ternary operator.
- C++ also provides a math library that can be included using '#include<cmath.h>' command.
- 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
- We should avoid assigning a "one-dimensional" numeric matrix to a numeric vector. This will likely result in an error. We should work with either matrices or vectors and avoid mixing them up.
- The Rcpp classes mentioned above do not provide us with the tools to avoid element-wise implementations of linear algebra operations (e.g., matrix multiplication) so, we can use the C++ Linear Algebra library armadillo instead.
- Primary classes in armadillo include mat and vec; the elements are of type double.
- 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.
- Not any large program can be parallelized. It needs careful consideration and planning.
- The decision depends on factors such as available resources, time required to completely run the program and whether the tasks in the program are capable of running in parallel.
- R provides 2 packages for parallel computing: parallel and doparallel.
- The first step is to create a cluster using 'makeCluster' and allocate it the number of cores available in the system.
- To start using parallel functionality, we need to register with a parallel backend to use.
- 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.
- It also includes a parameter .combine which is used to specify the kind of output needed. Using .combine=c gives a vector output while .combine=rbind creates a matrix.
- It is extremely important to close the cluster at the end of execution step so that core memory is released.
- Usage of these packages provides speedup in the performance free of charge.

Python

- I had not used Python much before taking this course. But after using it in the course, I have found to be very easy to understand and use.
- Python is a case-sensitive, implicitly types object oriented programming language.
- The most important thing to remember about Python is the importance of whitespace. It matters a lot and code won't run properly is indentation is incorrect. To begin a block of code, use indentation and to end the block outdent it.
- Single line comments can be added to Python code using #.
- It is very easy to get help in Python. It can be done by calling help(<object>).
- Equals ("=") sign is used to assign values to variables and equality testing is done using two equals ("==") sign.

- One trick that many people don't know of is that Python allows usage of multiple variables on one line.
- An important point to be noted is that array indices begin from 0. Negative indices can also be used and they count from the end to beginning i.e. -1 is last item.
- Array ranges can be indexed using colon (:) and this indexing is inclusive-exclusive, so if we specify [2:5], it will return items from index 2 to 4.
- The string formatting in Python is very similar to the one in C. It uses % to add elements of a tuple in a string. Also, strings can use single or double quotes.
- To concatenate two or more strings, "+" operator can be used.
- A tuple is a useful data structure to store a group of values separated by comma. This can be used when we want to return multiple values from a function.
- Dictionary is another important data structure that can be used to store key-value pairs with unique keys. It supports finding and accessing any element in constant time.
- Another easy to use data structure is list which is like one-dimensional array. It is very flexible and provides many built-in functions for manipulation. For eg: list.count(a), list.append(b), list.remove(b), list.index(a), list.reverse(), list.sort()
- Python supports flow control statements like if, for and while. "For" is a very useful statement for enumerating through elements of a list.
- To obtain a list of numbers, range(<number>) can be used.
- Functions are declared using def keyword. Optional arguments are specified at the end of function declaration together with their default values.
- Parameters in functions are passed by reference so we should be careful and make a copy of the parameters if we don't want to change them. The immutable types in Python cannot be changed in the caller function by the callee.
- Numpy is a very powerful library available for scientific computing in Python. It provides a high-performance multi-dimensional array and tools for working with these arrays. It provides functions like append, insert, delete, mean, median, standard deviation, etc.
- A numpy array is a grid of values of same type. The number of dimensions is the rank of the array and the shape is a tuple of integers giving the size of the array along each dimension.
  Example:

```
import numpy as np
A = np.array([1,2,3])          #creates a rank 1 array
print(A[1], A[2])              #prints "2 3"
A[0] = 5                       #change an element in array
```

- There are many other options to create arrays of specific type.
  - A = np.zeros((2,2)) – creates an array of all zeros
  - B = np.ones((3,2)) – creates an array of all ones
  - C = np.eye(2) – creates a 2x2 identity matrix
  - D = np.random.random((2,2)) – creates an array with random values
- Numpy arrays can be sliced similar to lists. For eg: b = a[:2, 1:3] will generate a subarray consisting of first 2 rows and columns 1 and 2. Specifying just colon (:) will fetch all elements along that dimension.
- One very useful function provided by numpy is summation of array elements.
  - np.sum(x) – computes sum of all elements of x

    - np.sum(x, axis=0) – computes sum of each column of x
    - np.sum(x, axis=1) – computes sum of each row of x
- It is very easy to compute transpose of a matrix. For eg: a = b.T will store the transpose of b in a. An alternate way to do this is: a = np.transpose(b).
- It is also possible to change the shape of data y using reshape function. For eg: a.reshape(2,4)
- Concatenate function can be used to combine arrays. To stack array s vertically, vstack function can be used and to stack them horizontally hstack can be used.
- Matplotlib is a very useful library available for plotting. The function plot can be used to plot 2D data. It is also possible to plot multiple lines of data at once and add labels, title and legends to it.
  Example:
  import matplotlib.pyplot as plt
  plt.plot(x,y)
  plt.title('My plot')
  plt.show()
- Scikit-learn is a powerful library for implementing various ML, cross-validation and visualization algorithms using a unified interface.
  For example:
  from sklearn import datasets as ds
  iris = ds.load_iris() #loads iris dataset into the variable

SQL

- CREATE DATABASE dbname: to create a database named dbname
- CREATE TABLE table1 (id int, name varchar): to create a table with 2 columns
- ALTER TABLE: to add or delete columns in an existing table
- DROP TABLE: to delete a table
- INSERT INTO table VALUES(v1, v2): to insert new rows into a table
- UPDATE table SET col1 = newval: to update one or more columns in a table
- TRUNCATE TABLE: to delete contents of a table
- SELECT: to select all or particular columns from a table
- ORDER BY: sort the selected data from table according to certain rows
- GROUP BY: to perform aggregate functions on groups of column values
- INNER JOIN: to return all rows from both tables where there is a match
- LEFT JOIN: to return all rows from first table even if there is no match in second table
- RIGHT JOIN: to return all rows from second table even if there is no match in first table
- AVG(col): to compute average value of a group in column
- SUM(col): to calculate sum of all values in a group
- UNION: to combine rows from results of two queries
- INTERSECT: to return the intersection of two queries
- LIKE: to query rows in a table using pattern matching

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 very useful tool for keeping track of programming projects, papers and data analysis.
- Git config –global user.name “name”: sets the name to be attached to commits
- Git config –global user.email “email-id”: sets the email address to be attached to commits
- Git init [project name]: to create a new local repository from scratch
- Git clone url: to download from an existing directory
- Git status: to list out new or modified files that have not been committed yet
- Git diff: to view changes to the files that have not staged yet
- Git diff commit1 commit2: to find the difference between two commit ids
- Git log: to view the entire change history
- Git branch: to list all local branches
- Git checkout branch1: to switch to a branch and update working directory
- Git add [file]: stages the file and makes it ready for commit
- Git commit -m “message”: commit all staged files to versioned history
- Git fetch: to get all the latest changes from origin
- Git pull: to fetch the latest changes from origin and merge
- Git push: to push the local changes to the origin
- Git branch new_branch: to create a new branch with mentioned name
- Git rm [file]: deletes the file from working directory and stages the deletion

Hoffman

- It is the most powerful cluster in the UC system that provides 1200+ nodes, 13,340 cores and 50 TB memory.
- It is useful if we have huge amount of data and/or we want to perform high performance computation.
- To connect to Hoffmann2, Unix/Linux/Mac user can use the ‘ssh’ command in the terminal as follows:
  ssh login-id@hoffman2.idre.ucla.edu
- Windows user can use an SSH client like “MobaXterm”, “PuTTY”, or remote desktop “NoMachine”.
- ‘Login node’ is assigned to the user just after login and it can be used manage files or maneuver around.
- ‘Compute node’ is obtained upon request using ‘qrsh’ command. Whenever a job is submitted to scheduler, it is executed on a ‘compute node’.
- ‘Login node’ can never be used to perform computations as opposed to ‘compute node’.
- A user is given 20 GB of storage in home directory. If additional temporary storage is needed, upto 2TB of storage is available for 7 days in ‘/u/scratch’.
- If a job attempts to use more memory than what had been requested, it will be automatically terminated by scheduler.
- Module avail R/Python: to view different versions of R or Python available on Hoffmann2
- Module load R/3.2.1: to load a specified version of R
- To request a compute node for interactive use, use the ‘qrsh’ command as follows:
  qrsh -l h_rt=8:00:00,h_data=4G

[h_rt is requested running time and h_data is requested memory]

- To submit a job to Hoffman2, use the 'qsub' command as follows:
  qsub -N job1 -l h_data=8G,h_rt=23:00:00 -m bea thejob.sh
  where -N job1 is to name the job 'job1' and -m bea is to define mailing rules to send an email when job '**b**'egins, '**e**'nds, and if the job is '**a**'borted
- myjobs: to list the names of currently running jobs
- cwd: to change the working directory to where you currently are in the file system

TensorFlow

- Tensorflow is an open source Python based toolkit for developing neural networks.
- It is graph based in structure where each node represents mathematical operation and each edge represents data flow between nodes.
- This framework is used to define and run computations involving tensors.
- A tensor is a generalization of vectors and matrices to potentially high dimensions.
- Each tensor has a datatype (float32, int32, string) and a shape. It might also belong to one of the special types.
- Placeholder is one such special type used when we need to send outside data into our neural networks.
  Eg: input1 = tf.placeholder(tf.float32)
- Once the graph for dataflow is defined, we need to create a session to run its parts. There are two ways to create a session. The only difference being that we need to close the session manually using sess.close() if we use the first method.
  1. sess = tf.session()
  2. with tf.session() as sess:
- Session is also useful for sending data into placeholder by using feed_dict as follows:
  sess.run(output, feed_dict={input1: [3.], input2: [6.]})
- TensorBoard is a suite of visualization tools that makes it easier to understand, debug, and optimize TensorFlow programs. It is useful to visualize our TensorFlow graph, plot quantitative metrics about the execution, and show additional data like images that pass through it.
- To build any neural network, first we need to import the needed modules and define functions to add layers. The next step is to initialize variables and create a session to perform operations.
- Finally, we can build the NN , send data using placeholders and then train and test it to measure its accuracy.
- tf.constant(2.0, name="const"): to create a Tensorflow constant named const
- tf.variable(1.0, name="a"): to create a Tensorflow variable named a with value 1
- tf.add(a, b, name="c"): to add values in 2 variables and store the result in c
- tf.global_variables_initializer(): to setup the variable initialization
- tf.matmul(a, b): to execute matrix multiplication between a and b
- tf.nn.relu(output_layer): to apply relu activation function to output of previous layer in NN
- If we want to assign probabilities to an object being one of several different things, softmax is used, because softmax gives us a list of values between 0 and 1 that add up to 1.
- A softmax regression has two steps: first we add up the evidence of our input being in certain classes, and then we convert that evidence into probabilities.
- tf.nn.softmax(output_layer): to apply softmax activation function to output of previous layer

- Cross-entropy is measuring how inefficient our predictions are for describing the truth and it can be implemented using following code. Here, tf.reduce_mean computes the mean over all the examples in the batch.
  tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=yhat, labels=y)):
- Tensorflow uses backpropagation algorithm to efficiently determine how our variables affect the loss we want to minimize. Then it can apply our choice of optimization algorithm to modify the variables and reduce the loss.
- tf.train.GradientDescentOptimizer(0.5).minimize(crossentropy): to minimize crossentropy using the gradient descent algorithm with a learning rate of 0.5
- tf.argmax is an extremely useful function which gives the index of the highest entry in a tensor along some axis.
- tf.equal(tf.argmax(prediction, 1), tf.argmax(y, 1)): to check if our prediction matches the truth.

SAS

- The following command can be used to import data from a file into a dataset in SAS

```
proc import out= work.data
datafile= "/path/to/file/file1.csv"
dbms=csv replace; getnames=yes; datarow=2;
run;
```

- To compute correlation between two columns of a dataset use ‘proc corr’ as shown below:

```
proc corr data=data;
var col1 col2;
run;
```

- The CORR procedure generates ‘Simple Statistics’ based on non-missing values, and ‘Pearson Correlation Coefficient’, an index that quantifies the linear relationship between a pair of variables.
- Insignificant p-value indicates the lack of linear relationship between the two variables.
- SGPLOT can be used to create statistical graphics such as histograms, lineplots and regression plots. To make a scatterplot of two columns of a dataset, use ‘sgplot’ as shown below. It is possible to add title, legends and labels to X-axis and Y-axis.

```
title 'Scatterplot';
proc sgplot data=data;
scatter x=col1 y=col2;
run;
```

- The BOXPLOT procedure creates side-by-side box-and-whiskers plots of measurements organized in groups. This plot displays the mean, quartiles, and minimum and maximum observations for a group.
- To make a box plot of a column with respect of grouping of values in another column, following command is used.

```
proc boxplot data=data;
plot col1*col2;
```

- PROC REG can be used for regression analysis to estimate the values of coefficients.

- To perform simple linear regression on two columns, use following command. Here 'noint' instructs sas not to include the intercept term. yval represents the dependent variable and xval represents the independent variable. The 'model' statement is used as we want to fit a model to our data.

```
proc reg data=data;
model yval = xval /noint;
run;
```

- The GLM procedure shown below uses the method of least squares to fit general linear models. If we want to perform polynomial regression, GLM needs to be used.

```
proc glm data = data;
model yval = x1 x1*x1;
run;
```