Statistical Computing with R
Masters in Data Science 503 (S9)
First Batch, SMS, TU, 2021

Shital Bhandary

Associate Professor

Statistics/Bio-statistics, Demography and Public Health Informatics

Patan Academy of Health Sciences, Lalitpur, Nepal

Faculty, Data Analysis and Decision Modeling, MBA, Pokhara University, Nepal

Faculty, FAIMER Fellowship in Health Professions Education, India/USA.

Review Preview (Unit 2, Session 4)

- Data Mining
  - Descriptive
  - Predictive
  - Prescriptive
- Text Mining
  - Steps
  - Package
    - tm
    - quanteda
    - tidytext and tidyverse packages
- Big data in R
  - Subsampling
  - Hex and 2D Density plots
  - ff, ffbase, ffbase2 packages
  - ScaleR & RHadoop packages
  - Microsoft R and R Server
  - SparkR and sparklyr packages
  - R for Azure SQL and SQL server
- Closure of Unit 2
  - Questions
  - Queries

Working with large datasets in R:
(Chapter 5: Beginning Data Science in R – Thomas Mailund)

- The concept of Big Data refers to very large datasets, sets of sizes where you need **data warehouses** to store the data, where you typically need sophisticated algorithms to handle the data, and distributed computations to get anywhere with it.
- At the very least, we talk many gigabytes of data but also are often dealing with **terabytes or exabytes**.
- Dealing with Big Data is also part of data science. Working with large datasets and how to deal with data that slows down your analysis is very important knowledge and skill for data scientists.

Big Data/Large dataset: Chapter 5

- If we ignore the Big Data issue, what a large dataset is, depends very much on what you want to do with the data. That comes down to the complexity of what you are trying to achieve.
- The science of what you can do with data in a given amount of time, or a given amount of space (be it RAM or disk space or whatever you need), is called complexity theory and is one of the fundamental topics in computer science.
- **In practical terms, though, it usually boils down to how long you are willing to wait for an analysis to be done and it is a very subjective decision**.

Working with large datasets: Chapter 5

- **Subsample** your data before you Analyze the Full Dataset
- You very rarely need to analyze a complete dataset to get at least an idea of how the data behaves.
- Unless you are looking for very rare events, you will get as much a feeling for the data looking at a few thousands of data points as you would from looking at a few million.
- Here it is important that you pick a **random sample**. Randomizing might remove a subtle signal, but with the power of statistics, we can deal with **random noise**.
- It is much harder to deal with consistent biases we just don't know about.
- This is same as taking random sample from (target) population in research!

You can use “dplyr” package for sampling:

- iris %>% sample_n(size = 5) #Select random sample of size 5
- iris %>% sample_frac(size = 0.02) #Select 2% random sample

- You need your data in a form that “dplyr” can manipulate, and if the data is too large even to load into R, then you cannot have it in a data frame to sample from, to begin with.

- Luckily, dplyr has support for using data that is stored on disk rather than in RAM, in various backend formats, too.

- It is, for example, possible to connect a database to dplyr and sample from a large dataset this way.

Problems working with large datasets in R: Chapter 5

- Running out of memory during analysis
  - R can be very wasteful of RAM because R remembers more (than is immediately obvious) but shows less in the output
  - In R, all objects are immutable (unless we use a workaround)
  - Whenever you modify an object, you are actually creating a new object

- Too large to plot (Use Hex plot and/or 2-D density plots for this)
  - d <- data.frame(x = rnorm(10000), y = rnorm(10000))
  - d %>% ggplot(aes(x = x, y = y)) + geom_point() # Large/cluttered scatterplot
  - d %>% ggplot(aes(x = x, y = y)) + geom_hex() # Requires “hexbin” package
  - **d %>% ggplot(aes(x = x, y = y)) + geom_density_2d() # 2D Density plot**
  - d %>% ggplot(aes(x = x, y = y)) + geom_hex() + scale_fill_gradient(low = "lightgray", high = "red") + geom_density2d(color = "black") # Hex with 2-D Density plot

Problems working with large datasets in R: Chapter 5

- Too slow to analyze
  - Sub-sample the data
  - **Otherwise use need to pick analysis algorithms that work more efficiently**
  - Use lm() for fitting standard linear models
  - **Use biglm package to fit standard linear models in large datasets**
- "biglm" package provides a memory efficient linear model fitting (it avoids creating a model matrix that would have rows for each data point and solving equations for that) and functionality for updating the model in batches
- Defining the slice indices requires some arithmetic and after that we can extract subsets of the data using the slice() function from dplyr.

Example 1: slice from dplyr & library(biglm)
Loading required package: DBI (R Database Interface)

```
slice_size <- 10
n <- nrow(cars)
slice <- cars %>% slice(1:slice_size)
model <- biglm(dist ~ speed, data = slice)
        for (i in 1:(n/slice_size-1)) {
        slice <- cars %>% slice((i*slice_size+1):((i+1)*slice_size))
        model <- update(model, moredata = slice)
}
model
```

Large data regression model: biglm(dist ~ speed, data = slice)
Sample size = 50

biglm package models can be used without slices too but it makes more sense to work with the slices for fast computing in R!

I have not shown the model fit for now!

Example 2:

- library(ff)
  - ffcars <- as.ffdf(cars)
  - summary(ffcars)
- library(ffbase) #Another set of fast models for big data in R!
  - model <- bigglm(dist ~ speed, data = ffcars)
  - summary(model)
- library(ffbase2)
  - dplyr on ff
  - Available from github: https://github.com/edwindj/ffbase2
  - iris_f <- tbl_ffdf(iris)
  - cars_f <- tbl_ffdf(mtcars, src="./db_ff, name_cars")

dplyr backend for (relational) database:

- These systems require that you set up a server for the data, though, so a simpler solution, if your data is not already stored in a database, is to use LiteSQL. LiteSQL works just on your file system but provides a file format and ways of accessing it using SQL.

- You can open or create a LiteSQL file using the src_sqlite() function:
  - iris_db <- src_sqlite("iris_db.sqlite3", create = TRUE)

- You load a dataset into it using copy_to():
  - iris_sqlite <- copy_to(iris_db, iris, temporary = FALSE)

Pull a table with tbl and run query with dplyr:

- Once you have a connection to a database, you can pull out a table using tbl():
  - iris_sqlite <- tbl(iris_db, "iris“) #This is a direct process too!

- Then you can use dplyr functions to make a query to it:
  - iris_sqlite %>% group_by(Species) %>%
  - summarise(mean.Petal.Length = mean(Petal.Length))
- ## Source: query [?? x 2] ## Database: sqlite 3.33.0 [C:\Users\Dell\Documents\iris_db.sqlite3]
- ## Species mean.Petal.Length
- ## <chr> <dbl>
- ## 1 setosa 1.462
- ## 2 versicolor 4.260
- ## 3 virginica 5.552

Big Data Analytics with R: Utilize R to uncover hidden patterns in your Big Data (BOOK)

- **Big Data is possibly the scariest, deadliest and the most frustrating phrase which can ever be heard by a traditionally trained statistician or a researcher (as they can't do research/analysis using software e.g. SPSS)**
- **The initial problem lies in how the concept of Big Data is defined!**
- Some (maybe psychologists?) will try to convince you that even 100 MB is quite a big file or big enough to be scary. Some others (social scientists?) will probably say that 1 GB heavy data would definitely make them anxious.
- In fact, in many areas of medical science (such as human genome studies) file sizes easily exceed 100 GB each, and most industry data centers deal with data in the region of 2 TB to 10 TB at a time. **(For MS Excel – 5 GB?)**
- Leading organizations and multi-billion dollar companies such as Google, Facebook, or YouTube manage petabytes of information on a daily basis.
- **What is then the "threshold" to qualify data as Big?**

Working with Big Data in R: (from Big Data Analytics with R Book)

- Unleashing the power of R from within:
  - Use apply() family of functions instead of loop and pipes for data.frames
  - Use R package such as ff, ffbase, ffbase2 and bigmemory packages
  - Apply statistical methods to large R objects though biglm and ffbase packages
  - Enhance the speed of data processing with R libraries supporting parallel computing e.g. **parallel** (doParellel and forreach) and **boot** packages for the parallel computing and bootstrapping in R
  - Benefit from faster data manipulation methods available in the **data.table** package

Book Example: Bureau of Transformation Statistics (https://www.transtats.bts.gov/homepage.asp)

- flights.ff <- read.table.ffdf(file="flights_sep_oct15.txt", sep=",", VERBOSE=TRUE, header=TRUE, next.rows=100000, colClasses=NA)
  - csv-read=34.24sec ffdf-write=6.303sec TOTAL=40.543sec, 426.4 KB
- flights.table <- read.table("flights_sep_oct15.txt", sep=",", header=TRUE) – done in 32 seconds, data size = 101.9 MB

- read.table.ffdf() of 2013-2014 flights of 2 GB data) – done with 28 files of 516.5 KB size (456 seconds with only 380 MB RAM used)
- read.table() of 2013-2014 flights of 2 GB) data – done with 1.3 GB size single file (441 secondss with maximum of 4.85 GB RAM)

Use “data.table” package in R:

- A frequent criticism of R is its inefficiency in handling large datasets. That's where the R package data.table enters the scene. If your datasets have more than tens of thousands of rows, the data.table package is a must. (https://psrc.github.io/r-data-table/)
- The data.table is an alternative to R’s default data.frame to handle tabular data. The reason it’s so popular is because of the speed of execution on larger data and the **terse** syntax.
- **So, effectively you type less code and get much faster speed. It is one of the most downloaded packages in R and is preferred by Data Scientists.**
- It is probably one of the best things that have happened to R programming language as far as speed is concerned

Visit: https://www.machinelearningplus.com/data-manipulation/datatable-in-r-complete-guide/ for full tutorial on data.table package and its syntax!

Hadoop and MapReduce frameworks in R: You will need to know Linux commands!

- RHadoop package: It supports MapReduce, HDFS and Hbase database management directly from the console of R language (R Studio server is better!) [rmr2 and rhdfs of Rhadoop are used for MapReduce files!]

- The packages have been developed by **Revolution Analytics**, but due to the acquisition of Revolution Analytics by Microsoft, the latter has recently become the lead maintainer of the packages.

- All five R packages (rhdfs, rhbase, plyrmr, rmr2 and ravro) of RHadoop, their binary files, documentation, and tutorials, are available at a GitHub repository at https://github.com/RevolutionAnalytics/RHadoop/wiki

Computing with REvolution R / Microsoft R (Revolution Analytics – Wikipedia)

- Revolution Analytics was founded in 2007 as REvolution Computing providing support and services for R in a model similar to Red Hat's approach with Linux in the 1990s as well as bolt-on additions for parallel processing.
- Their core product, Revolution R, offered free to academic users and their commercial software would focus on **big data**, large scale multiprocessor (or "**high performance**") computing, and multi-core functionality. Norman H Nie was appointed as CEO in 2009, who sold SPSS to IBM for 2.1 billion dollars, left in 2012!
- Microsoft announced on January 23, 2015 that they had reached an agreement to purchase Revolution Analytics for an as yet undisclosed amount.

REvolution R: Early successes (Wikipedia)

- Adding parallel computing to R allowed the company to net large gains in speed for many common analytics operations and early clients like **Pfizer** took advantage of REvolution R to see large performance gains using R on computing clusters.

- While the improvements to core R were released under the GNU General Public License (GPL), REvolution provides support and services to customers of their commercial product and had considerable early success with life sciences and pharmaceutical companies.

ScaleR of REvolution Analytics: (Wikipedia)

- In 2010 Revolution Analytics introduced **ScaleR**, a package for Revolution R Enterprise designed to handle big data through a high-performance disk-based data store called XDF (not related to IBM's Extensible Data Format) and high performance computing across large clusters.
- The release of ScaleR marked a push away from consulting and services alone to custom code and ***a la carte*** package pricing.
- ScaleR also works with **Apache Hadoop** and other distributed file systems and Revolution Analytics has partnered with IBM to further integrate Hadoop into Revolution R.
- Packages to integrate **Hadoop** and **MapReduce** into open source R can also be found on the community package repository, CRAN

Changes in RevoR from Microsoft in 2015:

- Microsoft rebranded and renewed several Revolution Analytics products and offerings for Hadoop, Teradata Database, SUSE Linux, Red Hat, and Microsoft Windows.

- Microsoft made several of these R-based products free of charge for developers:
  - **Microsoft R Server**
  - Microsoft R Server Developer Edition
  - Microsoft Data Science Virtual Machine
  - **Microsoft R Open**

Revolution R Enterprise, R with sparkR and sparklyr package for R Studio and Azure/SQL:

- Revolution/Microsoft R Enterprise adds proprietary components e.g. scaleR to support statistical analysis of **Big Data**, and is sold as subscriptions for workstations, servers, Hadoop and databases.

- Single-user licenses are available free for academic users (I have it with VB GUI) as well as users competing in **Kaggle data mining** competitions.

- More on its blog site: https://blog.revolutionanalytics.com/

- R in Azure SQL and SQL server is also available with sparkR & sparklyr!
  - **https://spark.apache.org/docs/latest/sparkr.html**
  - **https://spark.rstudio.com/ #Connection to H2O is also possible with sparklyr!**
  - **https://cloudblogs.microsoft.com/sqlserver/2021/06/30/looking-to-the-future-for-r-in-azure-sql-and-sql-server/**

Big Data, Code Profiling and R plug-ins:

- Big Data (Volume, Variety and Velocity) analysis strategies in R:
  - https://rviews.rstudio.com/2019/07/17/3-big-data-strategies-for-r/
- Code Profiling and Optimization for Big Data:
  - http://www.columbia.edu/~sjm2186/EPIC_R/EPIC_R_BigData.pdf
- Pushing R Further (pdf file from Big Data Analytics with R book):
  - https://static.packt-cdn.com/downloads/5396_6457OS_PushingRFurther.pdf
- R/Python plug-in/integration for commercial statistical software e.g. SPSS is a norm now!
  - https://www.ibm.com/docs/en/spss-statistics/SaaS?topic=r-spss-statistics-essentials
  - https://www.stata.com/new-in-stata/pystata/

Question/Queries?

Thank you!

@shitalbhandary