post01-josia-yuan.Rmd

Josia Yuan

October 18, 2017

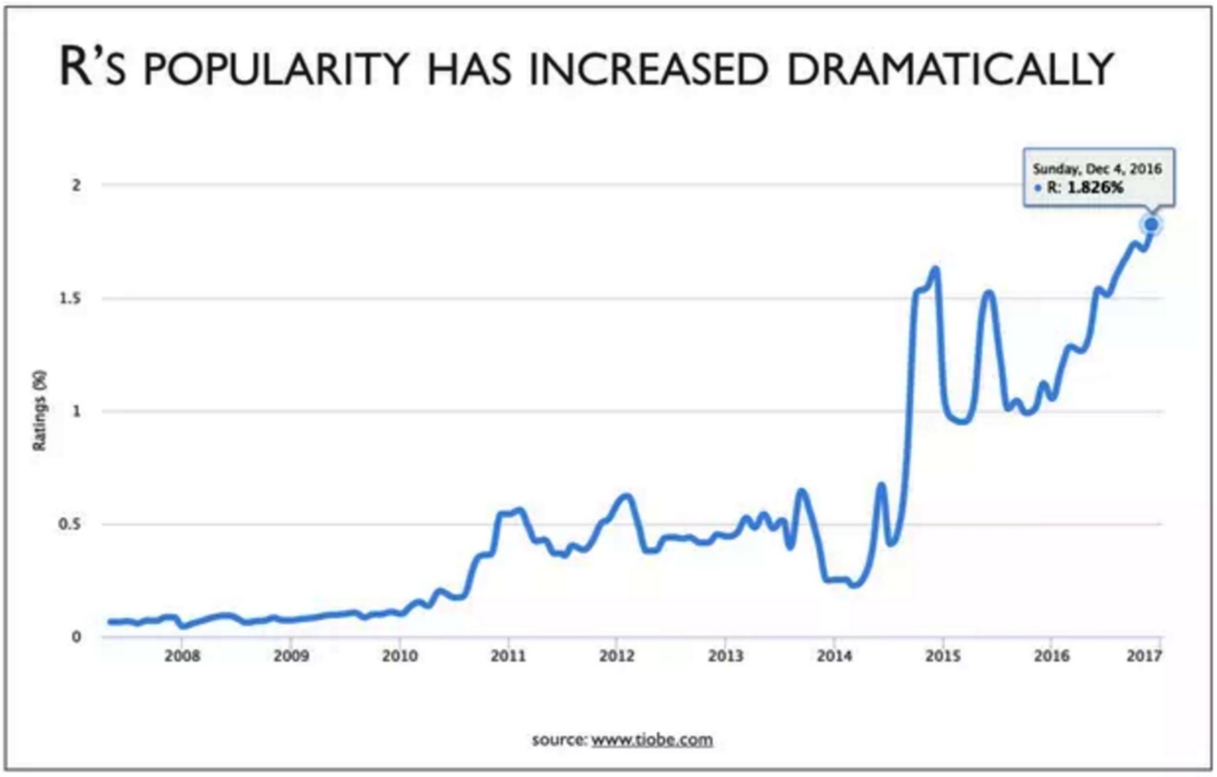
A Quick Look at the Pros and Cons of R

Introduction

As a beginner in R language, my first contact with R was both painful and inspiring. It was painful in the sense that I did not know anything about programming and I had to familiarize myself with the syntax and imbbedded intuitions. It was also inspiring becasue I had the opporutinity to explore different arguments,functions, and packages within R, while also evaluating R language in a more holisitic context. With this post, I want to share my own experiences as a beginner in picking up R and also incoporate others' opinions in the discussions. For others' opinions, I will draw in some online posts and resources that discuss the pros and cons of R respectively to give a more dynamic discussion.

Background

Accroding to a blog published by Sharp Sight Lab, R has gained its popularity over the year, a trend visible in several important surveys and programming language rankings. One of the ranking index "TIOBE" shows an increasing popularity of R as shown below.



TIOBE: R ranks high with consistent upward trend

While some recognize R's growing popularity, others also point out some shortcomings of R.The next section will give a closer look at how these opinions intervene with one another.I will start with my own experiences and delve into others' comments.

Discussion

Vectorization and Recycling

Personally speaking, one thing that I really like about R is its vectorization and recycling features.For a beginner who is not yet confortable with writing out each one of the customized commands, it is much easier for one to have an operation that executes on every element by itself without specification. A simple example would be addtion of value of 4 to each element in the vector "k".

```
#Demonstrate vectorization and Recycling
k <- c(1,2,3)
k_1 <- k + 4
k_1
```

```
## [1] 5 6 7
```

```
#Without vectorization
k <- c(1,2,3)
k_2 <- c(1+4,2+4,3+4)
k_2
```

```
## [1] 5 6 7
```

As shown above, instead of wrting out every expression of addition for each element in k (as demonstraed by k_2), the **vectorization** feature in R allows the user to **recycle** the operation and automatically have it executed on the entire vector of values at the same time (as demonstrated by k_1).

In regards to R's vectorization feature, Andrie de Vries and Joris Meys from "R for Dummies" gave the following insights:

> Programmers who are used to other languages often have trouble with this concept at first… Your natural reflex as a programmer may be to loop over all values of the vector and apply the function, but vectorization makes that unnecessary.

This shows that while beginners may find it easier to have vectorization, programmers familiar with other languages may find it quite troublesome.

R Packages

Another thing I have found great about R is its pleaseing visulization and manipulation of data. Stats 133 has covered packages like dplyr, ggplot2 and shiny app, with which users can customize almost every feature of the graphs rendered as well as process data more intuitively. Here I want to talk about another interesting package **lubridate** which makes it easier to work with dates and times.

```
#load the package
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.4.2
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
#You can use the package to change Time Zone
time <- ymd_hms("2017-10-28 09:00:00", tz = "Pacific/Auckland")
with_tz(time, "America/Chicago")
```

```
## [1] "2017-10-27 15:00:00 CDT"
```

The code chunck above illustrates how **lubridate** allows users to find the exact time within a different timezone. The package also has many other functions in regards to parsing, extracting information, and finding time intervals. More information to be found here.

With this example, I want to showcase how the multiple packages in R allow users extensive flexibility in dealing with different sorts of data. For visulization, R has ggplot2, ggvis, lattice, rCharts, googleVis; while for data processing R provides dplyr, tidyr, lubridate, and many more!

In regards to such flexibility, an article published by Paul Krill Why R? The pros and cons of the R language highlights some of R's strengths as the following:

> The vastness of package ecosystem is definitely one of R's strongest qualities – if a statistical technique exists, odds are there's already an R package out there for it…Any new research in the field probably has an accompanying R package to go with it from the get-go.

The comment above highlights the advantage of R's extensive range of packages as well as the flexibility which allows for even more add-on functions.

More of Experts' Comments on Cons of R

In his article Why R is Hard to Learn, Robert A. Muenchen illustrated some common complains about R as the following:

- Lack of Graphical User Interface (GUI): Like most other packages, R's full power is only accessible through programming. However unlike many others, it does not offer a standard GUI to help non-programmers do analyses.
- Lack of Consistent Naming Convention: You see names in: names, colnames
  row.names, rownames
  rowSums, rowsum
  rowMeans, (no parallel rowmean exists)
  browseURL, contrib.url, fixup.package.URLs
  package.contents, packageStatus
  getMethod, getS3method
- Inconsistent Syntax: Since everyone is free to add new capabilities to R, the resulting code for different R packages is often a bit of a mess.
- Too Many Ways to Select Variables: If variable x is stored in mydata and you want to get the mean of x, most software only offers you one way to do that, such as "VAR x;" in SAS. In R, you can do it in many different ways:
  summary(mydatax) summary(mydata"x")
  summary(mydata["x"])
  summary(mydata[,"x"])
  …

Quick note at Comparison with Python

Since Python is usually referred to as an important counterpart to R in data analysis, I want to provide some information about comparision between the two:

Accroding to the same blog published by Sharp Sight Lab, a common issue with Python is that:

> …many students get caught up in software development. That is, instead of learning statistics, data visualization, data manipulation, probability, etc, they end up spending their time learning about data structures, loops, flow-control, object oriented programming, and web frameworks. These skill areas can complement the core data science toolkit, but they are not data science topics in the sense that I'm using the term here. In fact, I recommend that most beginners learn software development contepts after learning basic data science subjects like data manipulation, visualization, analysis, etc.

Moreover, datacamp provides a graphical illustration of comparison between R and Python in terms of features. A one sentence summary I would like to highlight for R and Python is as such:

> "The closer you are to statistics, research and data science, the more you might prefer R."

> "The closer you are to working in an engineering environment, the more you might prefer Python."

In a word, depending on one's specific needs and backgrounds, R or Python may serve different usages and provide more advantages over the other.

Take-home Message

Through this post, I hope to provide a more dynamic discussion about the pros and cons of R so that audience can have more references when they evaluate R on their own. To summarize, a common consent among the R community is that R provides many useful packages for data/statistical analysis, while it might be troublesome or confusing sometimes due to its ineffcienciy in dealing with large data and data structures. In comparison to Python, R might be less intuitive yet has its own advantage of flexibility and visualization, while Python is better for engineering field.

Reference

1. R's Pros and Cons
2. Why R is hard to learn
3. Why R is the best data science language to learn today
4. HOW TO VECTORIZE YOUR FUNCTIONS IN R
5. Choosing R or Python for Data Analysis? An Infographic
6. Do more with dates and times in R with lubridate 1.3.0
7. ggvis basics
8. Data Visualization in R with lattice
9. Interactive Charts from R using rCharts

Processing math: 100% ng Your Data to Life with googleVis and R - Tutorial