R Programming Notes using LaTeX

Arvind Kumar

July 1, 2020

Abstract

R is a language and environment for statistical computing and graphics. This essay give brief introduction to R. The entire essay is written using LaTeX

Introduction

Always play with the code while learning,programming fundamentals always need to come first: the better you understand them, the easier it is to learn more advanced concepts. For college exams and technical interviews you will have to code-by-hand, because not only is this good for learning, but it's universally known to be the ultimate test for a programmer's proficiency.R is a programming language developed by Ross Ihaka and Robert Gentleman in 1993. R possesses an extensive catalog of statistical and graphical methods. It includes machine learning algorithms, linear regression, time series, statistical inference to name a few. Most of the R libraries are written in R, but for heavy computational tasks, C, C++ and Fortran codes are preferred.

1 Basics

There is an art to problem solving, and the only way to get practice is to get out there and start solving problems. Form your question, get data, explore and analyse your data, and communicate your results.

1.1 What is data?

First up, we'll look at the Cambridge English Dictionary, which states that data is: Information, especially facts or numbers, collected to be examined and considered and used to help decision-making. Second, we'll look at the definition provided by Wikipedia, which is: A set of values of qualitative or quantitative variables. A more complex data source to analyse are images/videos. There is a wealth of information coded in an image or video, and it is just waiting to be extracted. An example of image analysis that you may be familiar with is when you upload a picture to Facebook and not only does it automatically recognize faces in the picture, but then suggests who they may be. A fun example you can play with is the Deep Dream software that was originally designed to detect faces in an image, but has since moved on to more artistic pursuits.

1.2 Installing R and RStudio

Install R and can open the R interface to input code, but there are other ways to interface with R - and one of those ways is using RStudio o expand upon R's basic functionality, people have developed packages.A package is a collection of functions, data, and code conveniently provided in a nice, complete format for you.A package is not to be confused with a library (these two terms are often conflated in colloquial speech about R). A library is the place where the package is located on your computer. To think of an analogy, a library is, well, a library... and a package is a book within the library. The library is where the books/packages are located.

1.3 What are repositories?

A repository is a central location where many developed packages are located and available for download.

1.3.1 There are three big repositories

1. CRAN (Comprehensive R Archive Network):R's main repository (more than 12,100 packages available!
2. BioConductor:A repository mainly for bioinformatic-focused packages
3. GitHub:A very popular, open source repository (not R specific!)

1.4 What is an R Project?

When you make a Project, it creates a folder where all files will be kept, which is helpful for organizing yourself and keeping multiple projects separate from each other.

1.5 What is version control?

Version control is a system that records changes that are made to a file or a set of files over time. What are the benefits of using version control? Without version control, one might be keeping multiple, very similar copies of a file Git is a free and open source version control system. We've spent a lot of time getting R and RStudio working, learning about projects and version control - you are practically an expert at this! There is one last major functionality of R/RStudio that we would be remiss to not include in your introduction to R -Markdown!

1.6 What is R Markdown?

R Markdown is a way of creating fully reproducible documents, in which both text and code can be combined. In fact, these lessons are written using R Markdown! That's how we make things:bullets, bold, italics, links, run inliner code

1.7 The main types of data science analysis

There are, broadly speaking, six categories in which data analyses fall.

1.7.1 Descriptive Analysis

Descriptive Analysis will generate simple summaries about the samples and their measurements. You may be familiar with common descriptive statistics: measures of central tendency (eg: mean, median, mode) or measures of variability (eg: range, standard deviations or variance).

1.7.2 Exploratory

The goal of exploratory analysis is to examine or explore the data and search relationships that weren't previously known.

1.7.3 Inferential Analysis

is commonly the goal of statistical modelling, where you have a small amount of information to extrapolate and generalize that information to a larger group.

1.7.4 PredictiveEssentially

you are using current and historical data to get patterns and predict the likelihood of future outcomes.

1.7.5 Causal Analysis

Fills that gap; the goal of causal analysis is to see what happens to one variable when we manipulate another variable - looking at the cause and outcome of a relationship.

1.7.6 Mechanistic Analyses

are not nearly as commonly used as the previous analyses - the goal of mechanistic analysis is to understand the exact changes in variables that lead to exact changes in other variables. These analyses are exceedingly hard to use to infer much, except in simple situations or in those that are nicely modeled by deterministic equations.

1.8 Challenges of working with big data

1. It is big
2. It is constantly changing and updating
3. The variety can be overwhelming
4. It is messy

1.9 Advantage of working with big data

So with all of these challenges, why don't we just stick to analysing smaller, more manageable, curated datasets and arriving at our answers that way? Sometimes questions are best addressed using these smaller datasets, but many questions advantage from having lots and lots of data, and if there is some messiness or inaccuracies in this data, the sheer volume of it negates the reason of these small errors. So we are able to get closer to the truth even with these messier datasets. Additionally, when you have data that is constantly updating, while this can be a challenge to analyse, the ability to have real time, up to date information allows you to do analyses that are accurate to the current state and make on the spot, rapid, informed predictions and decisions.

2 How to program in R and how to use R for effective data analysis

Roger D. Peng: I don't think anyone actually believes that R is designed to make everyone happy. For me, R does about 99

2.1 R Nuts and Bolts

The primary R system is available from the Comprehensive R Archive Network15, also known as CRAN .R functionality is divided into a number of packages.

2.2 Entering Input

2.2.1 Evaluation

When a complete expression is entered at the prompt, it is evaluated and the result of the evaluated expression is returned. The result may be auto-printed.

```
# Create a sequence of numbers
X = 2:10
# Display basic statistical measures
summary(X)
```

3 R Objects

R has five basic or "atomic" classes of objects:

1. character
2. numeric (real numbers)
3. integer
4. complex
5. logical (True/False)

vector can only contain objects of the same class. Numbers in R are generally treated as numeric objects (i.e. double precision real numbers) so something like "1.00" or "2.00" If you explicitly want an integer, you need to specify the L suffix. So entering 1 in R gives you a numeric object; entering 1L explicitly gives you an integer object.

4 Attributes

R objects can have attributes, which are like metadata for the object. These metadata can be very useful in that they help to describe the object. For example, column names on a data frame help to tell us what data are contained in each of the columns. Some examples of R object attributes are

1. names
2. dimnames
3. dimensions (e.g. matrices, arrays)
4. class (e.g. integer, numeric)
5. length
6. other user-defined attributes/metadata

Attributes of an object (if any) can be accessed using the attributes() function. Not all R objects contain attributes, in which case the attributes() function returns NULL.

4.1 Creating Vectors

The c() function can be used to create vectors of objects by concatenating things together.

```
> x <- c(0.5, 0.6)
> x <-c(TRUE,FALSE)
```

When different objects are mixed in a vector, coercion occurs so that every element in the vector is of the same class. implicit coercion. Explicit Coercion Objects can be explicitly coerced from one class to another using the as.* functions, if available.

```
> x <- 0:6
> class(x)
[1] "integer"
> as.numeric(x)
[1] 0 1 2 3 4 5 6
> as.logical(x)
[1] FALSE TRUE TRUE TRUE TRUE TRUE TRUE > as.character(x)
[1] "0" "1" "2" "3" "4" "5" "6"
```

4.2 Matrices

Matrices are vectors with a dimension attribute. The dimension attribute is itself an integer vector of length 2 (number of rows, number of columns)

```
> m <- matrix(nrow = 2, ncol = 3)
>m
[,1] [,2] [,3] [1,] NA NA NA [2,] NA NA NA
> dim(m)
[1] 2 3
> attributes(m)
$dim
[1] 2 3
> y <- c(1.7, "a") ## character
```

4.3 Lists

Lists are a special type of vector that can contain elements of different classes. Lists can be explicitly created using the list() function, which takes an arbitrary number of arguments.

```
> x <- list(1, "a", TRUE, 1 + 4i)
>x
[[1]] [1] 1
```

4.4 Factors

Factors are used to represent categorical data and can be unordered or ordered. One can think of a factor as an integer vector where each integer has a label.

```
> x <- factor(c("yes", "yes", "no", "yes", "no"))
>x
[1] yes yes no yes no
```

4.5 Missing Values

Missing values are denoted by NA or NaN for q undefined mathematical operations.

1. is.na() is used to test objects if they are NA
2. is.nan() is used to test for NaN

```
> ## Create a vector with NAs in it
> x <- c(1, 2, NA, 10, 3)
> ## Return a logical vector indicating which elements are NA
> is.na(x)
[1] FALSE FALSE TRUE FALSE FALSE
```

4.6 Data Frames

Data frames are used to store tabular data in R. package dplyr35 has an optimized set of functions designed to work efficiently with data frames. every element of the list has to have the same length. data frames can store different classes of objects in each column. Data frames are usually created by reading in a dataset using the read.table() or read.csv().

```
> x <- data.frame(foo = 1:4, bar = c(T, T, F, F))
>x
foo  bar
1  1 TRUE 2  2 TRUE
          .        3  3 FALSE
          .        4  4 FALSE
> nrow(x)
[1] 4
> ncol(x)
[1] 2
```

4.7 Names

Useful for writing readable code and self-describing objects.

```
> x <- 1:3
> names(x)
NULL
> names(x) <- c("New␣York", "Seattle", "Los␣Angeles")
>x
   New York      Seattle Los Angeles
123 > names(x)
[1] "New␣York" "Seattle" "Los␣Angeles"
```

5 Adding ggplot 2 example figures

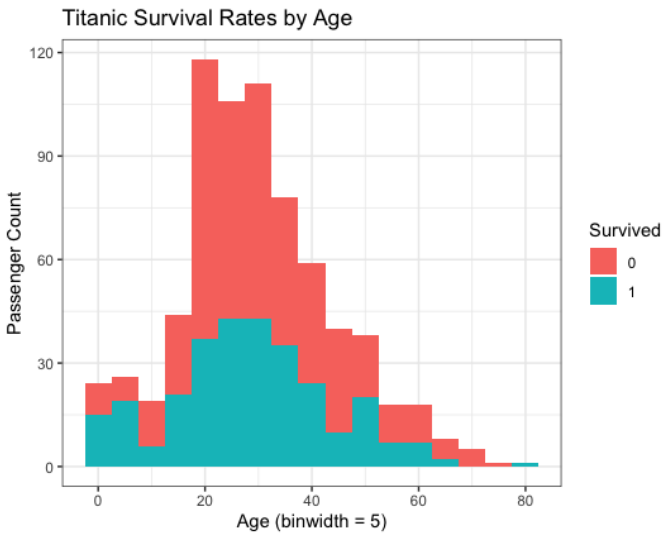


Figure 1: Titanic Survival Rates by Age using histogram

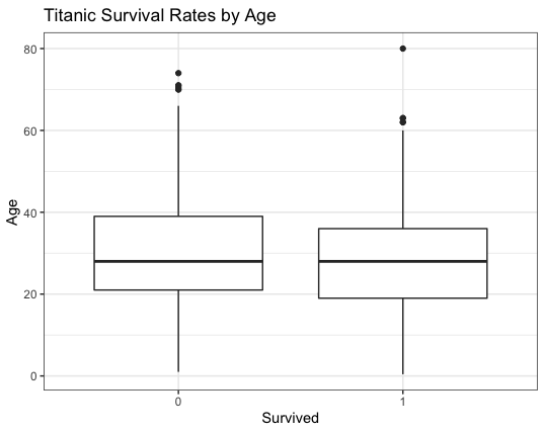
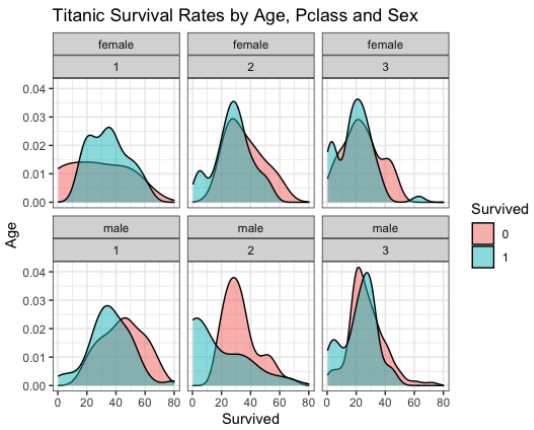


Figure 2: Titanic Survival Rates by Age using box and whisker plot

[Fig.a]



[Fig.b]

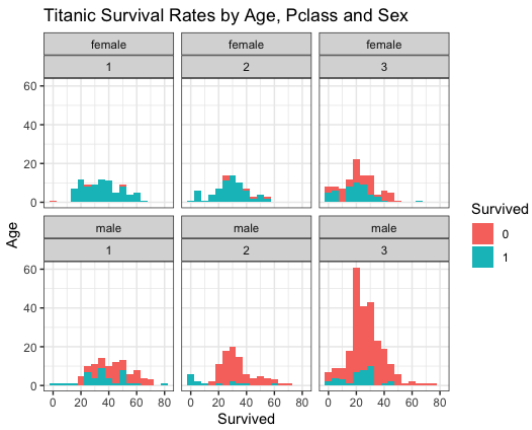


Figure 3: Titanic Survey Rates by Age,Pclass and Sex using density plot and histogram

6 Conclusion

LaTeX is quite simple, anyone who is writing non-trivial documents and is tired of being let down by the performance of the current crop of word processors. If you are in academia, you really ought to be using it! Anybody writing anything maths related will not find a richer and better quality system. For example, even WikiPedia use LaTeX for rendering any formulas that appear on their site.

LaTeX isn't for people who are too lazy or dislike change! I personally found the investment paid off because LaTeX allows me to produce my documents at a greater pace. I know that the enterprise will not be interested as Word is so ingrained, even though their business reports would look so much nicer. Their loss! For everyone else, it's time to give it a fair try, just so that you compare and contrast, then decide which does the job best for your needs.

References

Roger D. Peng. *R Programming for Data Science*
https://www.overleaf.com