Introduction to R for beginners

Cheng-Bang Chen

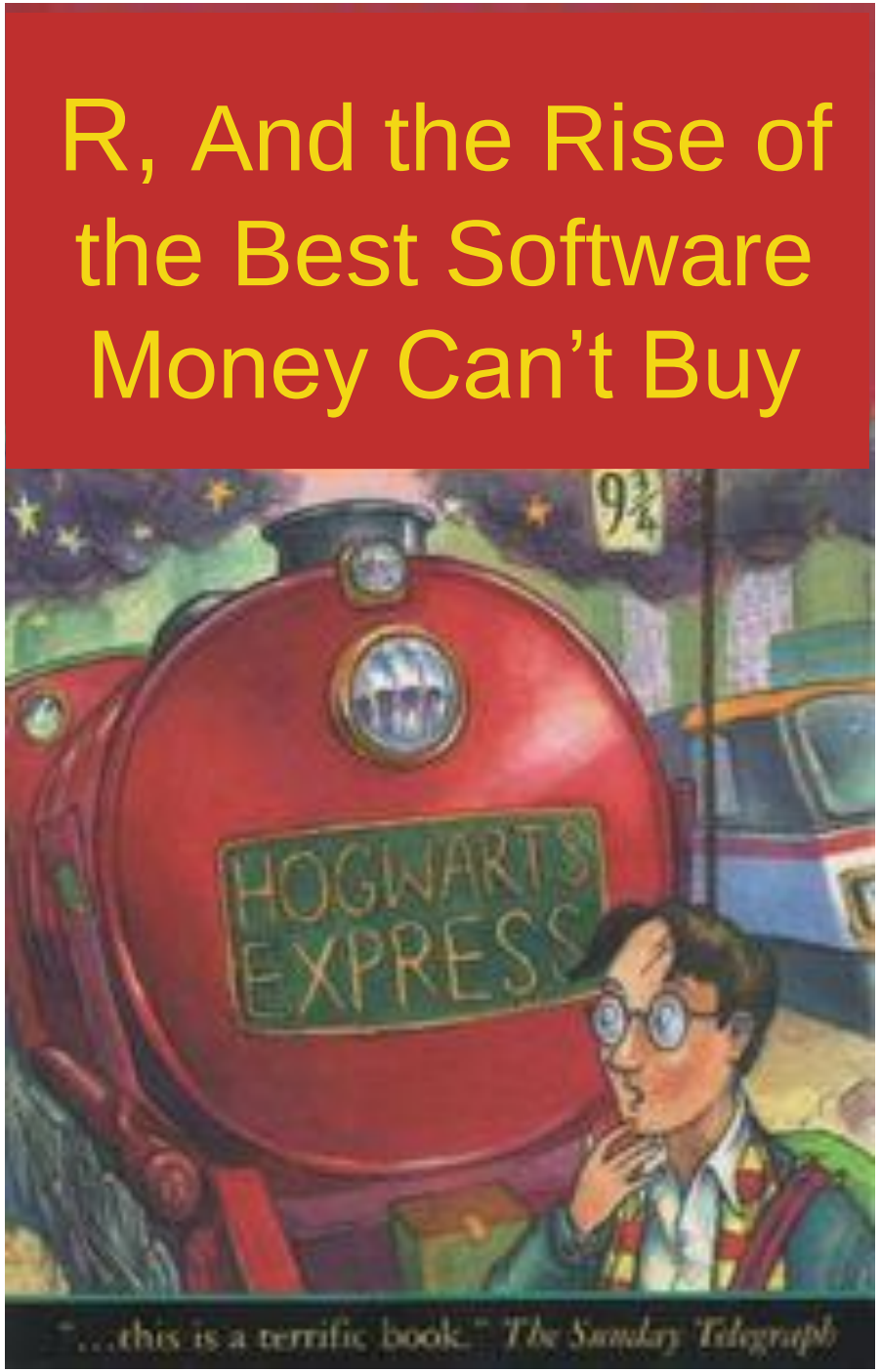
Dr. Soundar Kumara

Topics

- What is R? Why R?
- Install R & Rstudio
- Basic R commands
  - Getting Started
  - Basic Calculation
  - Data type in R
  - Basic statistic
  - Control flow
  - pakcage

R programming language is a lot like magic... except instead of spells you have functions.

- Matthew Keller

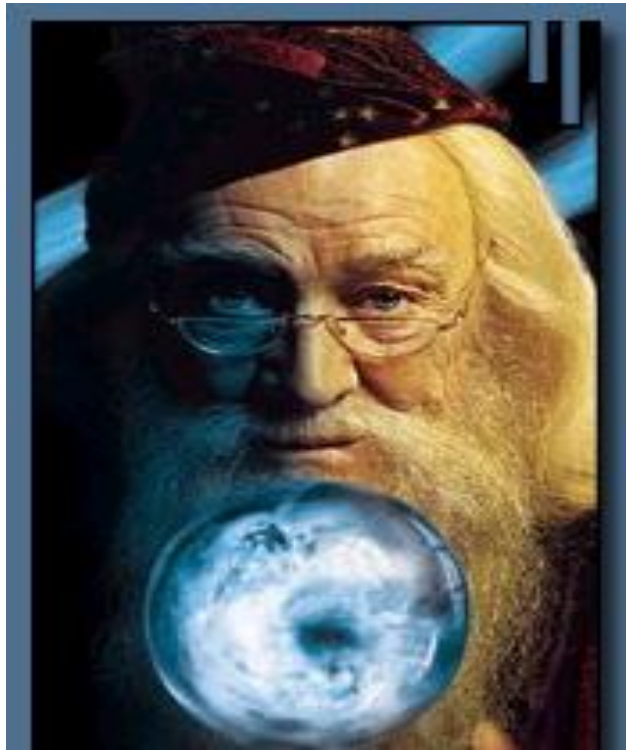


SPSS =

muggle

SPSS and SAS users are like muggles. They are limited in their ability to change their environment. They have to rely on algorithms that have been developed for them. The way they approach a problem is constrained by how SAS/SPSS employed programmers thought to approach them. And they have to pay money to use these constraining algorithms. - **Matthew Keller**

R =



wizard

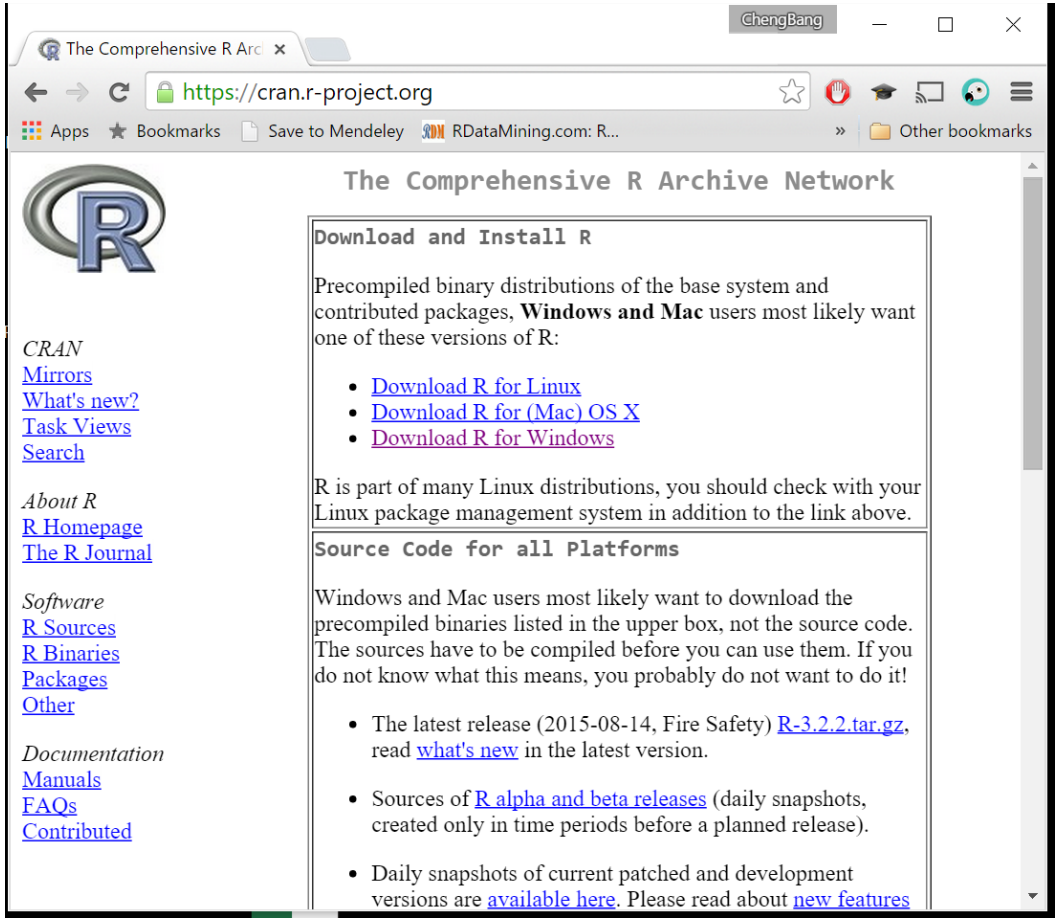
R users are like wizards. They can rely on functions (spells) that have been developed for them by statistical researchers, but they can also create their own. They don’t have to pay for the use of them, and once experienced enough (like Dumbledore), they are almost unlimited in their ability to change their environment.

- Matthew Keller

What is R?

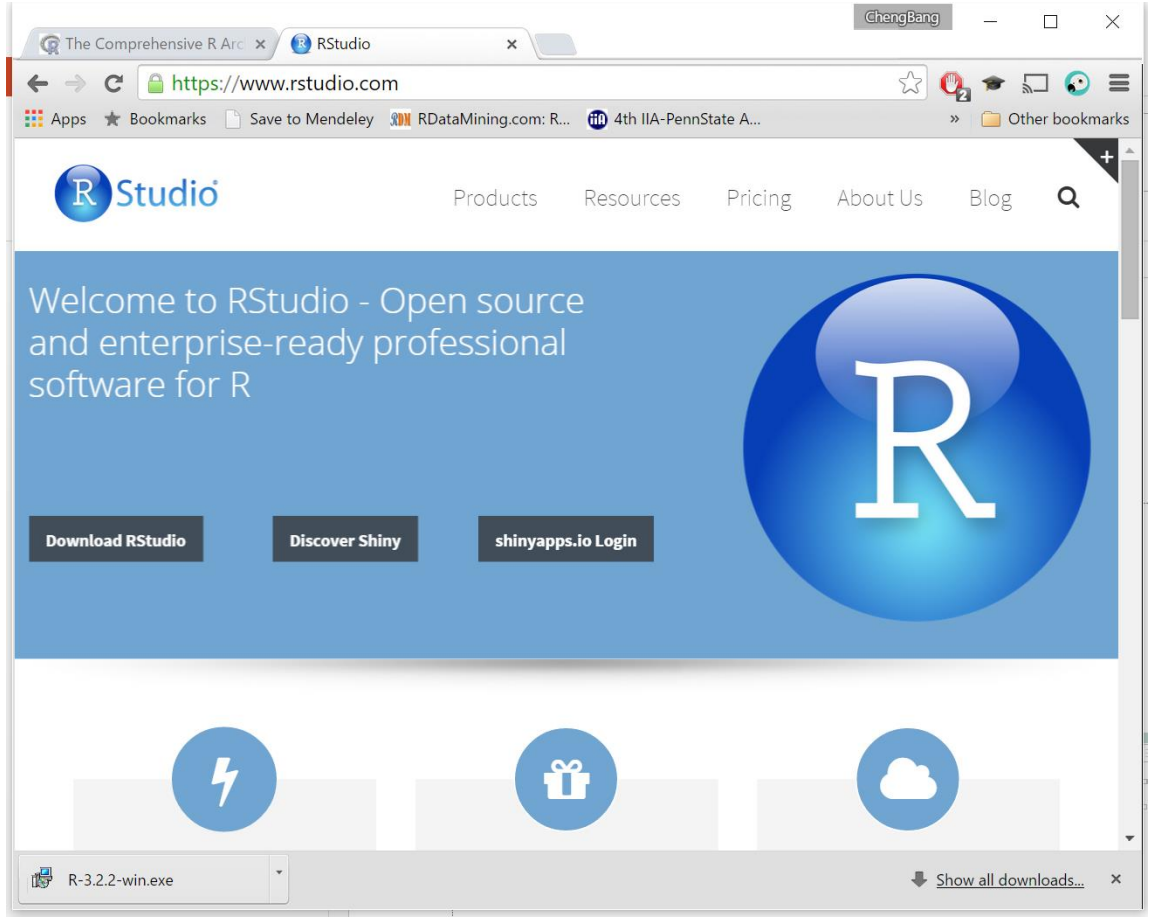
- The R statistical programming language is a free open source package based on the S language developed by Bell Labs.
- R is the leading tool for statistics, data analysis, and machine learning.
- The language is very powerful for writing programs.
- Many statistical functions are already built in.
- Contributed packages expand the functionality to cutting edge research.
- Since it is a programming language, generating computer code to complete tasks is required.
- R is a powerful language and environment for Statistical computing and graphics

Download R



- Go to https://cran.r-project.org
- Download the suitable version for your computer

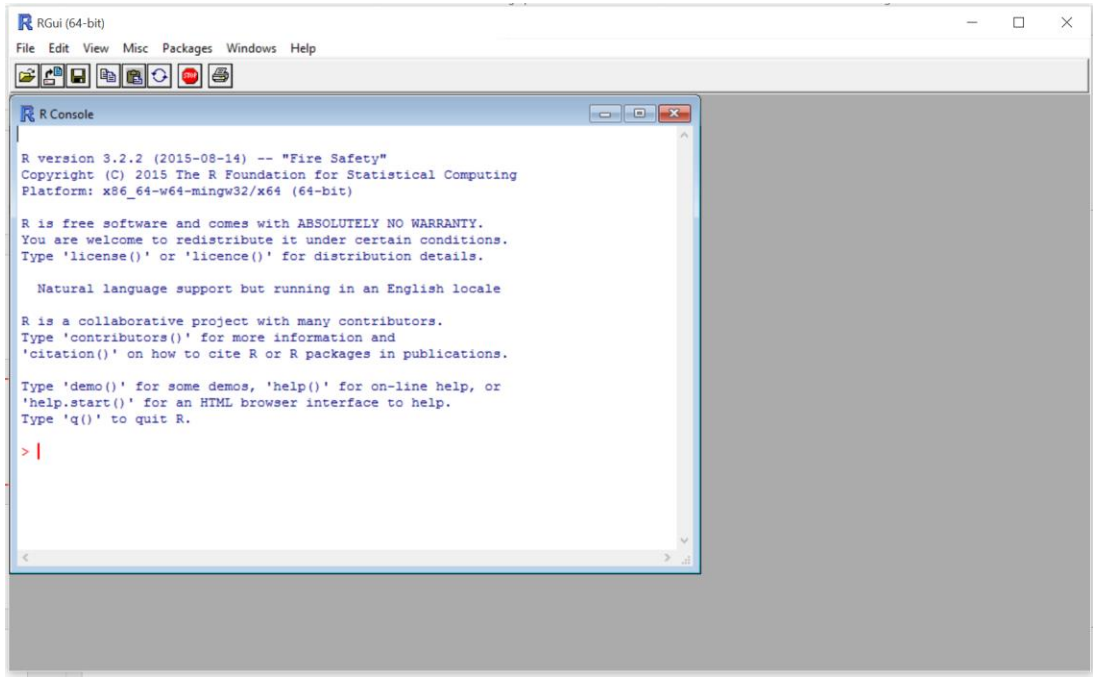
Download Rstudio



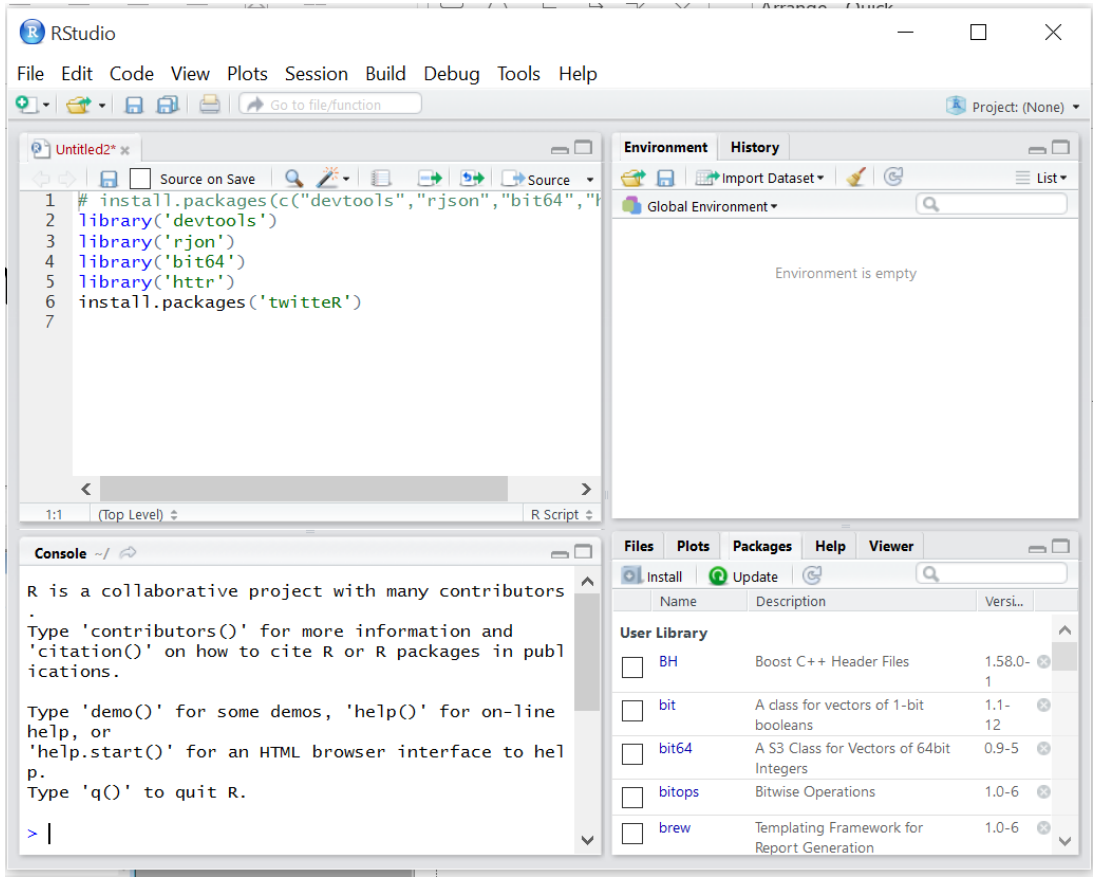
- Go to https://www.rstudio.com
- Download the suitable version for your computer (select the free version)

Install R and R studio

R console:



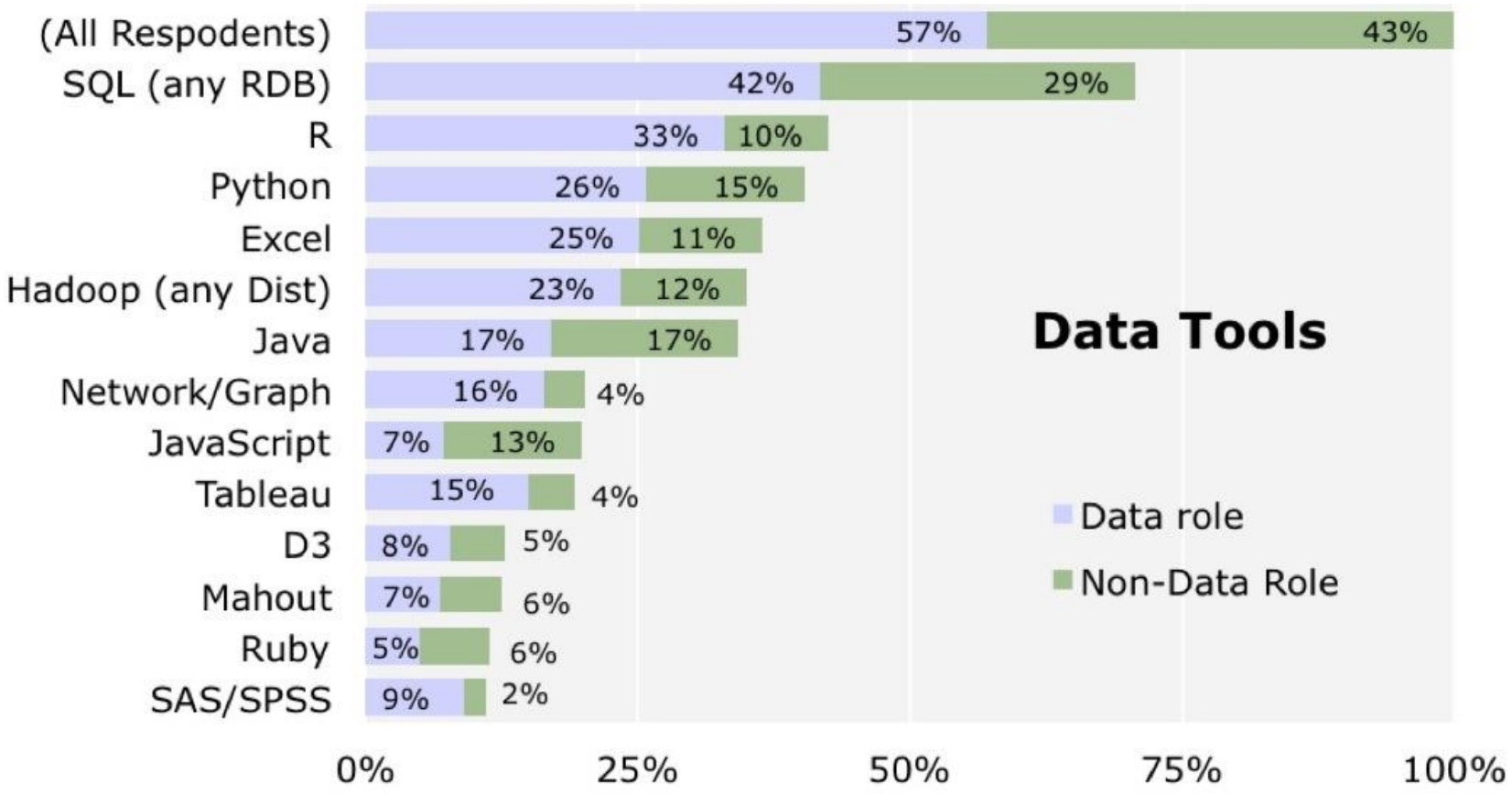
Rstudio:



* Rstudio integrates R and provides better UI (User Interface) and some extra options.

Data Science Salary Survey 2013

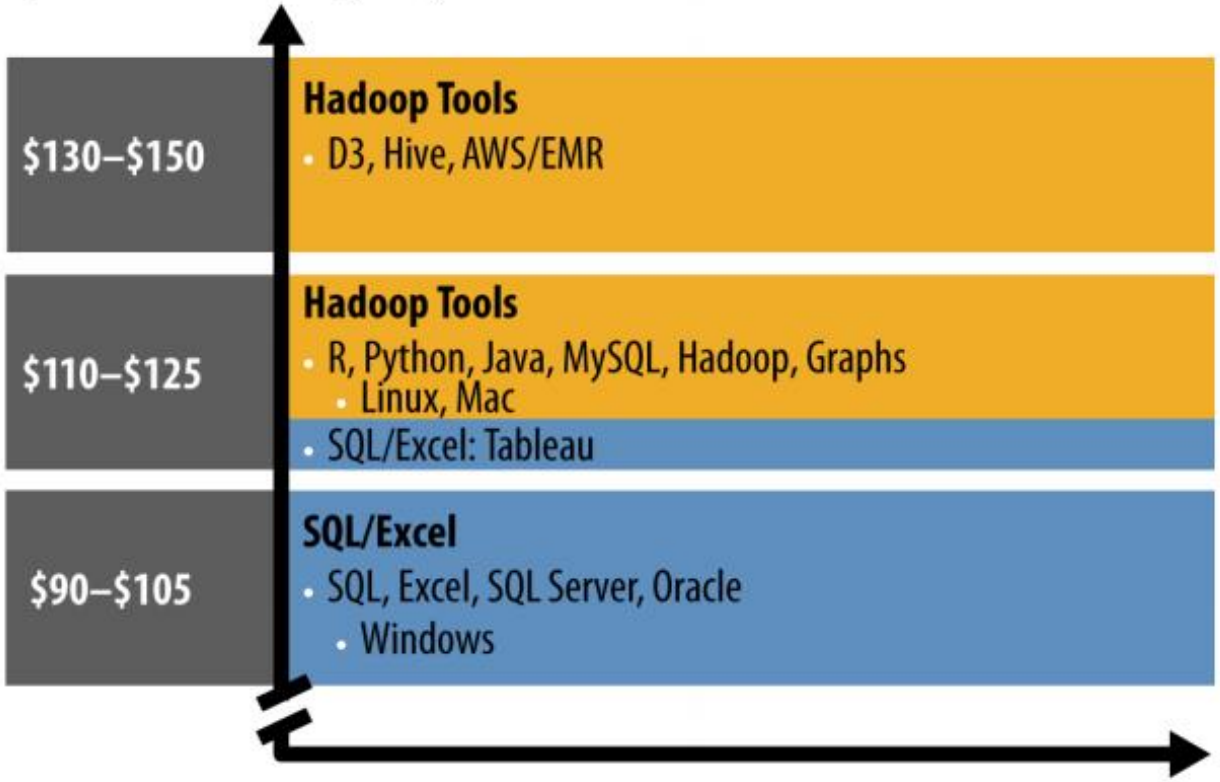
The usage rate for the most commonly used tools.



2013 Data Science Salary Survey

Salary & Tools

- **Two Clusters and Salary**
- **Newer, More Scarce Skills Pay Better**
- **Specialized Knowledge Pays Better**



- Supply and demand at work
- Few BO or Netezza users, high median salary
- Hadoop: Linux, R, Python, Java, D3, Mahout, Pig, Hive, MongoDB, Hbase, Libsvm, cassandra, Pentaho, Network/Graph
- SQL: Windows, Excel, SQL, Tableau, SQL Server, Oracle, DB2, Excel, VBA, BO, Netezza, SAS, Cognos

Layout of Rstudio

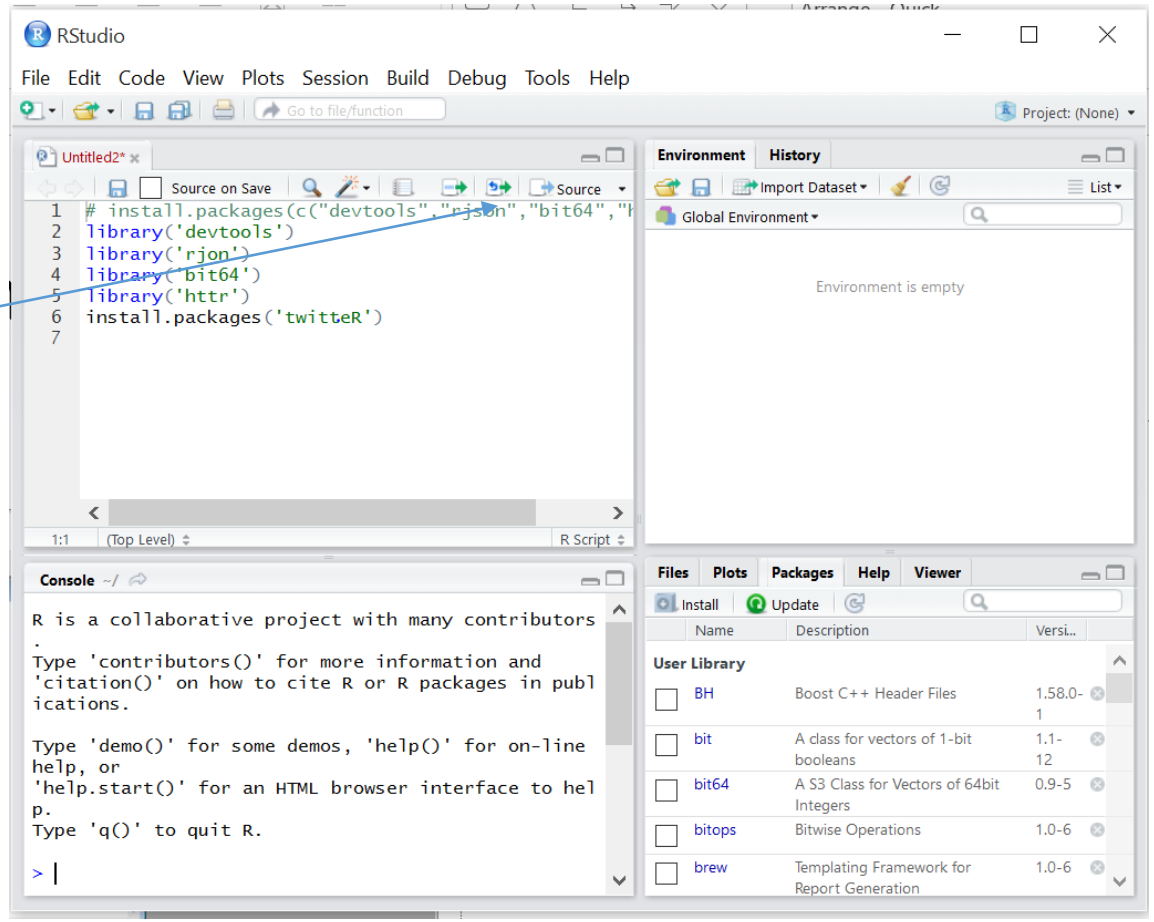
Script editor

[File] → [New] → [R script]
Collections of commands (scripts) can be edited and save.

Use [Run] or CTRL + ENTER to execute the selected commands.

R console

The commands are executed in this windows.



Workspace/history window

In this section, we can see which data and values R has in its memory.

You can view and edit the values by clicking on them

Go to [history] tag, we can see the executed commands

Files/Plots/Packages/Help/Viewer

In this section, we can open files, view plots (also previous plots), install/load packages,
Use the help function

Getting Started

- Basic assignment and operations.
- Arithmetic Operations:
  - +, -, *, /, ^ are the standard arithmetic operators.
- Matrix Arithmetic.
  - * is element wise multiplication
  - %*% is matrix multiplication
- Assignment
  - To assign a value to a variable use “<-”
- R is letter case sensitive

Getting Started

- How to use help in R?
  - R has a very good help system built in.
  - If you know which function you want help with simply use ?_____ with the function in the blank.
    - Ex: ?hist
    - Ex: ?pairs
  - If you don't know which function to use, then use help.search("_____").
    - Ex: help.search("histogram").

Basic Calculation

- **Calculation in the console:**
  - `2^10-24`
  - `(3/4+12/8)*15`

- **Variable**
  - `x=3`
  - `y=x+2`
  - `x<-2*x+2`

- `x<-3` is equal to `x =3`

Basic Data type

- **Scalar** (single number)
  - `x <-3`
  - `sqrt(x)`
- **Vector** (a row of numbers – 1 dimensional)
  - `a <- c(1,2,5.3,6,-2,4)` # numeric vector
  - `b <- c("one","two","three")` # character vector
  - `c <- c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE)` #logical vector
  - `d <- c(1,2,3,4,5,6)`
  - `x2 <- c(x,2*x,3*x)`
  - `x3 <- seq(0,102,3)`
  - `x4 <- sin(x3)`
- **Matrix** (like a table – 2 dimensional)
  - `m <-matrix(c(1,3,5,7,9,11), nrow=3, ncol=2)`
  - `n <- matrix(c(1,2,3,4,5,6), nrow=2, ncol=3 , byrow = TRUE)`
  - `m[3,2] # row 3 and column 2 of matrix m`
  - `n[,2] # whole column 2 of matrix n`
  - `n[1,] # 1st row of matrix n`

- Try:
  - `2*x`
  - `2*a + 1`
  - `a + d`
  - `2*m`
  - `m + t (n)`
  - `m[3,2]+n[1,1]`
  - `m %*% n`
  - `m+1`

Basic Data type

- **Data Frames**
  - A data frame, in brief, is a *matrix* with names above the columns.
  - You can call and use one of the columns without knowing in which position it is.

- Try:
- `tt$x`
- `tt$x+2-2/3*tt$y`
- `mean(tt$z)`

```
tt<-data.frame(x=c(11,12,14), y=c(19,20,21), z=c(10,9,7))
df<-data.frame(x1=c(101,102,103), y1=c("A", "A", "C"),z1=c(TRUE,TRUE,FALSE))
```

Read data/write data

- Import data:
  - read.csv('/your directory/filename.csv') or read.csv('\\your directory\\filename.csv')
  - Try:
    - Read the file 'IE330_HW5.csv' and assign to a variable hw5

      hw5<-read.csv('/your directory/IE330_HW5.csv')

- Export data:
  - write.csv(data object, '/your directory/filename.csv')
  - Try:
    - write hw5 to a csv file named test.csv

      write.csv(hw5, 'your directory/test.csv')
    - Is it the same as 'IE330_HW5.csv'?

      write.csv(hw5, 'your directory/test.csv', row.names = FALSE)

Plot the data

- plot(data)
- plot(x, y)
- Plot(x, y, xlab="…", ylab="…",main="…")

- Try:
  - `plot(hw5)`
  - `plot(hw5$x)`
  - `plot(hw5$x, hw5$y)`
  - `abline(v=10)`
  - `points(c(10,20,30),c(70,60,50),col="red")`

Plot the data

- hist(data,….)

- Try:
  - `hist(hw5$x)`
  - `hist(ht5$y)`
  - `Histhw5.x<-hist(hw5$x, breaks=20)`
  - `Histhw5.x$breaks`
  - `Histhw5.x$counts`
  - `Histhw5.y<-hist(hw5$y, breaks=seq(20,260,40))`

Basic statistic

- mean(data)
- sd(data)
- var(data)
- median(data)
- quantile(data, prob)
- cov(x,y)
- cor(x,y)
- lm(y~x)
- anova(object)
- length(array)
- dim(matrix or data frame)
- shapiro.test(x)

- Try:

```
mean(hw5$x)
sd(hw5$x)
var(hw5$x)
median(hw5$x)
quantile(hw5$x,.25)
cov(hw5$x, hw5$y)
cor(hw5$x, hw5$y)
FIT<-lm(hw5$y~hw5$x)
FIT2<-lm(y~x,data=hw5)
ANOVA<-anova(FIT)
length(hw5$x)
dim(hw5)
Shairo.test(x)
```

Regression

- lm(y~x)

- Try:
  - `Fit<-lm(hw5$y~hw5$x)`
  - `Fit$residual`
  - `Fit1<-lm(y~x, data=hw5)`
  - `Fit1$residual`
  - `anova(Fit)`

Basic data manipulation

- Data[criteria,] or Data[criteria]
  - Ex: show the data in hw5 that x>40
  - hw5[hw5$x>40,]
  - Ex: show the data in hw5 that x>40 and y<100
  - hw5[hw5$x>40 & hw5$y<100,]
  - hw5$x[hw5$x>=40]
  - hw5$x>=40
  - hw5>20

Control Flow

- if(cond) expr
- if(cond) cons.expr else alt.expr
- for(var in seq) expr

- Try:

```
A <- c(3,4,6,4,5)
B <- c(4,3,5,2,1)
C <- c()
for(i in 1:5){
      if(A[i]+B[i]>6){
            C[i]<-1}else{
            C[i]<-0
      }
}
```

function

<u>Define:</u>

myfun<-function(arg1, arg2,…){

procedures

}

<u>Apply:</u>

myfun(arg1, arg2,…)

- Try:

```
IE330<-function(x){
  if(x<=100 & x>=0){
    tmp<-ifelse(x>=60,"P","F")
  }else{
    tmp<-c("Check the input!")
  }
return(tmp)
}

IE330(78)
IE330(52)
IE330(101)
```

packages

- 1. Install the package at the first time using.
- 2. load the package
- 3. Call the function in the package.

- Ex.

```
data(iris)                # load data

install.packages('rgl') # install the package
library('rgl')            # load the package
plot3d(iris$Sepal.Length,
      iris$Sepal.Width,
      iris$Petal.Length,
      col = as.numeric(iris$Species))  # call the function in the package
```