Basic R Notebook: Lecture 1

Bikram

Preliminaries
Use `ctrl+alt+i` to create the R environment (thanks, Soo Han Soon)

Returns the current working directory

```r
getwd()
```

```
## [1] "/Users/Bikram/Dropbox/Courses/The Analytics Edge -- Fall 2020/Week 1/R code"
```

Provides help on a specific function

```r
help(getwd)
?getwd
```

Sets the current working directory

```r
setwd("/Users/Bikram/Google Drive/AnalyticsEdge/Week 1/R code")
```

Lists the files in a directory

```r
dir("/Users/Bikram/Google Drive/AnalyticsEdge/Datasets/")
```

```
##  [1] "Baseball.csv"
##  [2] "Chapter11_PageRankExercise_GoogleVotesComputation.xlsx"
##  [3] "Chapter11_PageRankExercise_VoteGraph.pdf"
##  [4] "ClassAssignments.xlsx"
##  [5] "ClimateChange.csv"
##  [6] "Crime.csv"
##  [7] "DailyKos.csv"
##  [8] "Edges.csv"
##  [9] "Elantra.csv"
## [10] "Framingham.csv"
## [11] "Gerrymandering.xlsx"
## [12] "HubwayTrips.csv"
## [13] "Letters.csv"
## [14] "Loans.csv"
## [15] "NasdaqReturns.csv"
## [16] "Parole.csv"
## [17] "Polling.csv"
## [18] "Quality.csv"
## [19] "SelectingHotels.xlsx"
## [20] "StateData.csv"
## [21] "Stevens.csv"
## [22] "Unemployment.csv"
## [23] "Users.csv"
## [24] "WHO.csv"
## [25] "Wikipedia.csv"
## [26] "Wine.csv"
## [27] "WineTest.csv"
```

Lists objects stored in the R workspace

```
ls()
```

```
## character(0)
```

Assigns a number to a variable

```
x<-40
x
```

```
## [1] 40
```

(Alternative to) assigning a number to a variable. Mostly '<-' is preferred for assignment.

```
x=40
x
```

```
## [1] 40
```

Common functions - exponential, inverse, power, addition

```
exp(x)
```

```
## [1] 2.353853e+17
```

```
1/x
```

```
## [1] 0.025
```

```
x^3
```

```
## [1] 64000
```

```
y<-x+6
y
```

```
## [1] 46
```

Remove a variable from the workspace

```
ls()
```

```
## [1] "x" "y"
```

```
rm(y)
ls()
```

```
## [1] "x"
```

Numbers and vectors

Concatenates (combines) numbers to form a vector

```
x<-c(1,-2,3,5,pi)
x
```

```
## [1]  1.000000 -2.000000  3.000000  5.000000  3.141593
```

Accessing specific elements of the vector

```
x[3]
```

```
## [1] 3
```

Applying operations to the vector - term by term inverse, concatenate vectors, exponentiation

```r
1/x
```

```
## [1]  1.0000000 -0.5000000  0.3333333  0.2000000  0.3183099
```

```r
exp(x)
```

```
## [1]   2.7182818   0.1353353  20.0855369 148.4131591  23.1406926
```

```r
y<-c(x,0,x)
y
```

```
##  [1]  1.000000 -2.000000  3.000000  5.000000  3.141593  0.000000  1.000000
##  [8] -2.000000  3.000000  5.000000  3.141593
```

You can overload the sum operator by recycling the shorter vector - mathematically adding vectors of different sizes are not permitted

```r
x
```

```
## [1]  1.000000 -2.000000  3.000000  5.000000  3.141593
```

```r
y
```

```
##  [1]  1.000000 -2.000000  3.000000  5.000000  3.141593  0.000000  1.000000
##  [8] -2.000000  3.000000  5.000000  3.141593
```

```r
x+y
```

```
## Warning in x + y: longer object length is not a multiple of shorter object
## length

##  [1]  2.000000 -4.000000  6.000000 10.000000  6.283185  1.000000 -1.000000
##  [8]  1.000000  8.000000  8.141593  4.141593
```

Finding the maximum and minimum elements and identifying the location (index) of the first max and all max

```r
max(x)
```

```
## [1] 5
```

```r
min(y)
```

```
## [1] -2
```

```r
which.max(x)
```

```
## [1] 4
```

```r
which(x==min(x))
```

```
## [1] 2
```

Other operations - sum, product, mean, variance, standard deviation

```r
sum(x)
```

```
## [1] 10.14159
```

```r
prod(x)
```

```
## [1] -94.24778
```

```r
mean(x)
```

```
## [1] 2.028319
```

```
sd(x)
```

```
## [1] 2.659851
```

```
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -2.000   1.000   3.000   2.028   3.142   5.000
```

For a vector, it provides a six number summary including min, max, mean, 1st quartile, median and 3rd quartile. This can be used with other objects too.

Maximum entry in a concatenated vector

```
max(x,y)
```

```
## [1] 5
```

Parallel maximum returns a vector of length equal to the longest argument that contains in each element, the largest element in that position of any of the vector

```
?pmax
x
```

```
## [1]  1.000000 -2.000000  3.000000  5.000000  3.141593
```

```
y
```

```
##  [1]  1.000000 -2.000000  3.000000  5.000000  3.141593  0.000000  1.000000
##  [8] -2.000000  3.000000  5.000000  3.141593
```

```
pmax(x,y)
```

```
## Warning in pmax(x, y): an argument will be fractionally recycled
```

```
##  [1]  1.000000 -2.000000  3.000000  5.000000  3.141593  1.000000  1.000000
##  [8]  3.000000  5.000000  5.000000  3.141593
```

Remove all variables from the workspace

```
ls()
```

```
## [1] "x" "y"
```

```
rm(list=ls())
ls()
```

```
## character(0)
```

Differences in the assignment using <- and ==. Try exp(a=1:5) and exp(a<-1:5)

```
exp(a<-1:5)
```

```
## [1]   2.718282   7.389056  20.085537  54.598150 148.413159
```

```
#exp(a=1:5)
```

Generating vectors using a variety of commands

```
x<--3:8
x
```

```
##  [1] -3 -2 -1  0  1  2  3  4  5  6  7  8
```

```
seq(-3,8,0.2)
```

```
##  [1] -3.0 -2.8 -2.6 -2.4 -2.2 -2.0 -1.8 -1.6 -1.4 -1.2 -1.0 -0.8 -0.6 -0.4 -0.2
## [16]  0.0  0.2  0.4  0.6  0.8  1.0  1.2  1.4  1.6  1.8  2.0  2.2  2.4  2.6  2.8
## [31]  3.0  3.2  3.4  3.6  3.8  4.0  4.2  4.4  4.6  4.8  5.0  5.2  5.4  5.6  5.8
## [46]  6.0  6.2  6.4  6.6  6.8  7.0  7.2  7.4  7.6  7.8  8.0
```

```r
rep(x,times=3)
```

```
##  [1] -3 -2 -1  0  1  2  3  4  5  6  7  8 -3 -2 -1  0  1  2  3  4  5  6  7  8 -3
## [26] -2 -1  0  1  2  3  4  5  6  7  8
```

```r
rep(x,each=3)
```

```
##  [1] -3 -3 -3 -2 -2 -2 -1 -1 -1  0  0  0  1  1  1  2  2  2  3  3  3  4  4  4  5
## [26]  5  5  6  6  6  7  7  7  8  8  8
```

Returns logical vector based on the check

```r
x
```

```
##  [1] -3 -2 -1  0  1  2  3  4  5  6  7  8
```

```r
x>1
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

Dealing with missing entries

```r
is.na(x)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```r
?is.na
y<-c(x,NA)
y
```

```
##  [1] -3 -2 -1  0  1  2  3  4  5  6  7  8 NA
```

```r
is.na(y)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [13]  TRUE
```

END OF LECTURE 1.