An introduction to R

Scott Powers and Ryan Wang

January 2, 2014

The most important command

- R is a very well-documented language
- The most important function is help, as it allows you to access that documentation
- It can be accessed by help(function) or ?function for short e.g. type ?help to learn more about the help function
- Google is also a great resource and usually links to either stackoverflow or the R mailing list

Data frames

Let's start by reading in some real-life data

```
>   setwd('C:/GitHub/data-mining-intersession/day1/data')
>   nba <- read.csv('box_2012to2013.csv')
>   class(nba)
```

In R terminology, data typically comes in the form of a *data.frame* (try ?data.frame). Data frames are similar to spreadsheets or matrices that can store non-numeric values.

Data frames cont.

This is essentially the same as what you would see if you opened the .csv file in MS Excel

```
>   nba[1:5,1:5]
```

```
  X   game_id  pid         player pos
1 0 400277721 4270 Trevor Booker  PF
2 1 400277721 2426  Trevor Ariza  SF
3 2 400277721 2399  Emeka Okafor   C
4 3 400277721 4010    A.J. Price  PG
5 4 400277721 6580  Bradley Beal  SG
```

Columns correspond to *variables* e.g. column 5 indicates the number of minutes played. Rows correspond to *observations* e.g. row 1 corresponds to Trevor Booker's line on the box score.

Data frames cont.

This data frame contains 31,375 rows (observations) and 18 columns (variables)

```
> nrow(nba)
[1] 31375
> ncol(nba)
[1] 20
> dim(nba)
[1] 31375    20
```

Vectors

The fundamental unit of computation in R is the vector. Create a vector by combining values with the "c" function (see ?vector and ?c):

```
>   v1 <- c(1,2,3,4,5)
>   v2 <- c('a','b','c','d')
>   v1
[1] 1 2 3 4 5
>   v2
[1] "a" "b" "c" "d"
>   is.vector(v1)
[1] TRUE
```

Vectors cont.

Most functions involving vectors (as well as matrices) operate element-wise:

```
>   v1 + 3
[1] 4 5 6 7 8
>   v1 + v1
[1]  2  4  6  8 10
>   sqrt(v1)
[1] 1.000000 1.414214 1.732051 2.000000 2.236068
```

Vectors cont.

The ":" operator (see ?colon) creates sequences

```
> 1:5
[1] 1 2 3 4 5
```

The "runif" function (see ?runif) generates random numbers

```
> runif(5)
[1] 0.26595195 0.05518444 0.31894451 0.48705538 0.94638785
> runif(5, min=1, max=5)
[1] 4.231143 4.849250 1.329395 3.735416 2.646077
```

Vectors cont.

Boolean values (TRUE and FALSE) are very useful (we'll soon see an application)

```
> TRUE & FALSE
[1] FALSE
> !(!T & F)
[1] TRUE
```

Can also do vector operations

```
> 1:5 > 2
[1] FALSE FALSE  TRUE  TRUE  TRUE
```