A Short Tutorial on R

COSC 4335 Data Mining
Spring 2016
Can Cao

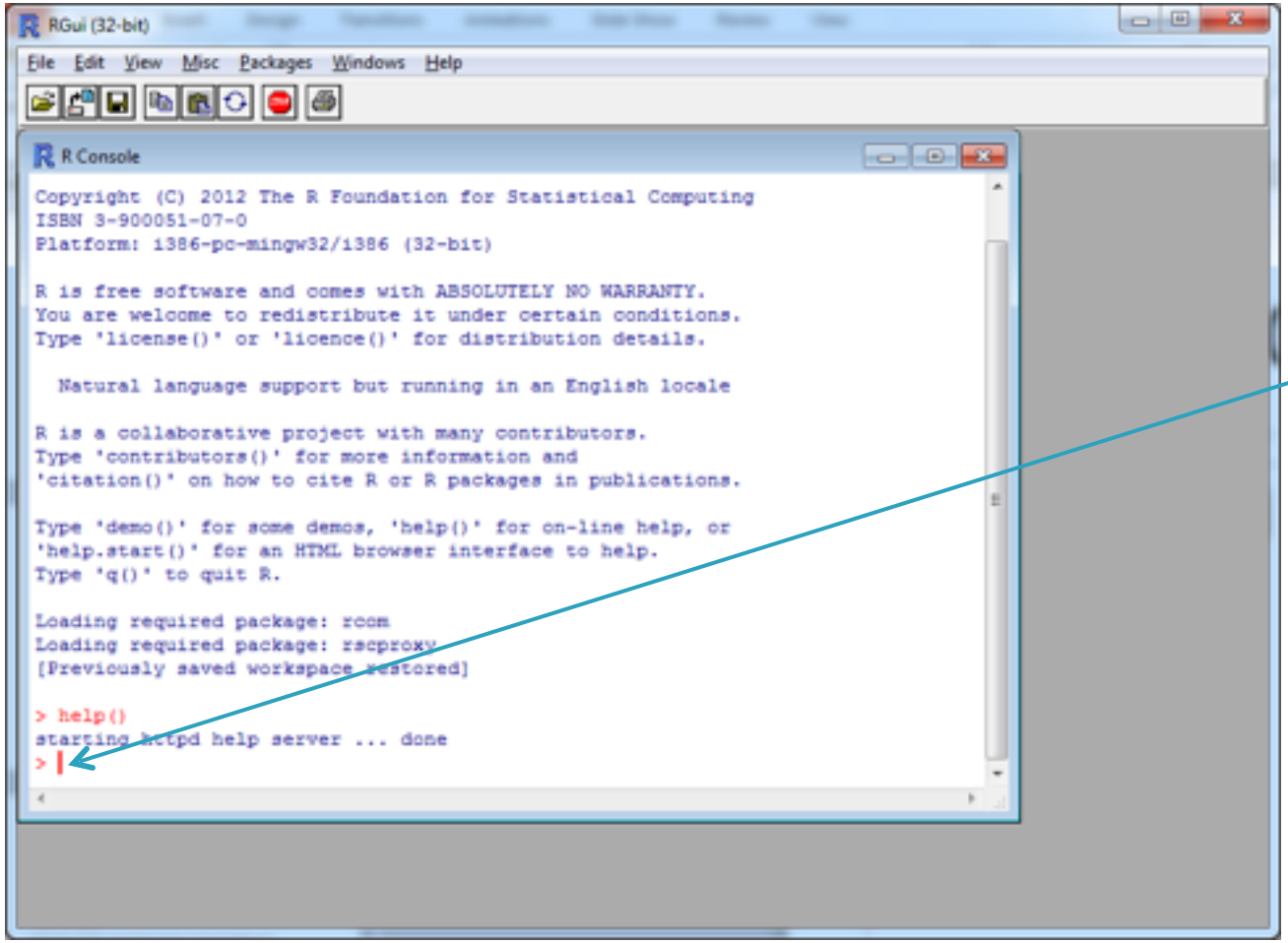
R… what is that?

- Free, open source statistical analysis software
- Competitor to commercial softwares like MATLAB and SAS – none of which are free
- Need to install only basic functionalities
- Install packages as and when needed – somewhat like Python
- Excellent visualization features!

Let's get started!

- Link for installation: http://cran.r-project.org/mirrors.html
- Choose a mirror link to download and install
- Choose default options for installation
- Don't worry about customizing startup options!
- Just click 'Okay', 'Next' and 'Finish'!

How R looks…


RGui (32-bit)
File Edit View Misc Packages Windows Help
R Console
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
Loading required package: rcom
Loading required package: rscproxy
[Previously saved workspace restored]
> help()
starting httpd help server ... done
>
Command line

Things to remember

- R is an interpreter
- Case sensitive
- Comment lines start with #
- Variable names can be alphanumeric, can contain '.' and '_'
- Variable names must start with a letter or '.'
- Names starting with '.' cannot have a digit as the second character
- To use an additional package, use library(<name>) e.g. library(matrixStats)
- Package you are trying to use must be installed before using

Things to remember

- Variable declaration not needed (like MATLAB)
- Variables are also called objects
- Commands are separated by ';' or newline
- If a command is not complete at the end of a newline, prompt shows '+'
- Use command q() to quit
- For help on a command, use help(<command>) or ?<command name>
- For examples using a command, use example(<command>)

Trying out…

- setwd(<folder_name>)
- source(<script_file>) – runs the R script <script_file>
- sink(<output_file>) – saves outputs to <output_file>
- sink() – restores output to console
- ls() – prints list of objects in workspace
- rm(<object1>, <object2>, …) – removes objects from workspace

```
> ls()
character(0)
> getwd()
[1] "/Users/ddiez"
> setwd("/Users/ddiez/Dropbox/rFunction/Videos")
> setwd("packageFiles")
> getwd()
[1] "/Users/ddiez/Dropbox/rFunction/Videos/packageFiles"
>
> source("lmPlot.R")
> ls()
[1] "lmPlot"
>
> setwd("..")
> getwd()
[1] "/Users/ddiez/Dropbox/rFunction/Videos"
```

Other things to know!

- Writing scripts - use Notepad/Notepad++ and save with extension .r OR use Rcmdr package

- Installing a package - Packages>Install package(s)... choose the package you want to install

Vectors in R

```
assign("x", c(10.4, 5.6, 3.1, 6.4, 21.7))
x
```

- c() concatenates vectors/numbers end to end

```
x<- c(10.4, 5.6, 3.1, 6.4, 21.7)
x

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

v<-2*x+y+1
v

min(v)

max(y)

range(x)

cat(v)
```

Sequences in R

v<-c(1:10)

w<-1:10

x<-2*1:10 (Note operator precedence here)

y<-seq(-4,7)

y<-seq(-4,7,2)

Arrays and Matrices

```
x <- array(1:20, dim=c(4,5))
x

i <- array(c(1:3,3:1), dim=c(3,2))
i

x[i] <- 0
#(1,3), (2,2), (3,1) are set to 0)
x

xy <- x %o% y #(outer product of x and y)
xy
```

Arrays and Matrices

```
seq1<-seq(1:6)
mat1<-matrix(seq1,2)
mat1
mat2<-matrix(seq1,2,byrow=T)
mat2
```

http://www.ats.ucla.edu/stat/r/library/matrix_alg.htm

- General use: matrix(data,nrow,ncol,byrow)

- Notice that ncol is redundant and optional!

Data Frames

- Efficient way to store and manipulate tables

e.g.

```
v1 = c(2, 3, 5)
v2 = c("aa", "bb", "cc")
v3 = c(TRUE, FALSE, TRUE)
df = data.frame(v1, v2, v3)
df
```

Factor

- A vector of categorical data

e.g.

```
> data = c(1,2,2,3,1,2,3,3,1,2,3,3,1)
> fdata = factor(data)
> fdata
 [1] 1 2 2 3 1 2 3 3 1 2 3 3 1
Levels: 1 2 3
```

Importing data

Skip today!

```
mydata <- read.table("c:/mydata.csv", header=TRUE, sep=",", row.names="id")
```

Path to the file
Note the use of / (slash) instead of \ (backslash)

Whether or not to treat the first row as header

Delimiter (default – White space)

Row names (optional)

mydata <-read.csv(<filename>, header = TRUE, row.names = <rows>)

Commonly used for CSV files

e.g.

```
mydata<-read.table("/Users/cancao/Downloads/sampledata1.txt")
ls()
```

Let's import a file

Download the file

http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data

Notice that the data is in CSV format and there are no headers!

```
mydata<-read.cvs(<path>, header = FALSE)
mydata<-read.csv("/Users/cancao/Downloads/adultdata.csv")
```

- You can also read xls files using read.xls which is a part of gdata package!

```
install.packages(pkgs="gdata")
library(gdata)
data <- read.xls(<path>)
data
```

Selecting/removing columns

- mydata<-data.frame(a=1:5,b=2:6,c=c(1,1,1,3,3))
- mydata[c(-1,-2)] #only shows the last column
- c
- 1 1
- 2 1
- 3 1
- 4 3
- 5 3
- > mydata
- a b c
- 1 1 2 1
- 2 2 3 1
- 3 3 4 1
- 4 4 5 3
- 5 5 6 3

Other manipulations

- Selecting observations

```
mySubset2<-mydata[1:4,]
mySubset2
```

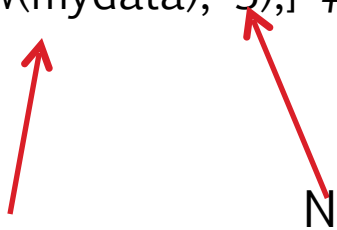
- Selecting observations based on variable values

```
mySubset3<-mydata[which(mydata$c==1 & mydata$a>2),] #display the rows where
   c==3 && a>2
mySubset3
```

Be careful when using multiple datasets!!!

Sampling, Splitting, Subsets

```
mySubset4<-mydata[sample(nrow(mydata), 3),] #randomly select samples
mySubset4
```



Number of samples

Total number of observations

```
mydata1<-split(mydata,mydata$c) #group data by class c!
mydata1
```

http://www.endmemo.com/program/R/split.php

Groups data of different classes together!

```
mySubset5<-subset(mydata,a==2 & c>0,select=c(a,c))
mySubset5
```

Most general form for manipulating data!

Good Manual on Data Import: http://cran.r-project.org/doc/manuals/R-data.html

```
> mydata1<-
split(mydata,mydata$c)
> mydata
  a b c
1 1 2 1
2 2 3 1
3 3 4 1
4 4 5 3
5 5 6 3
> mydata1
$1`
  a b c
1 1 2 1
2 2 3 1
3 3 4 1

$3`
  a b c
4 4 5 3
5 5 6 3
```

Other common operations…

- merge() - used for merging two or more data frames into a single frame
- cbind() – adds new column(s) to an existing data frame
- rbind() – adds new row(s) to an existing data frame

Visualization in R

- Different types of plots – helpful for data visualization and statistical analysis

- Basic plot

Download 'sampledata2' from https://sites.google.com/site/cosc4335datamining/resources

```
mydata<-read.csv(<path>,header=TRUE)
attach(mydata)
plot(x1,x2)
```

```
mydata<-read.csv("/Users/cancao/Downloads/sampledata2.txt", header=TRUE)
```

Visualization in R

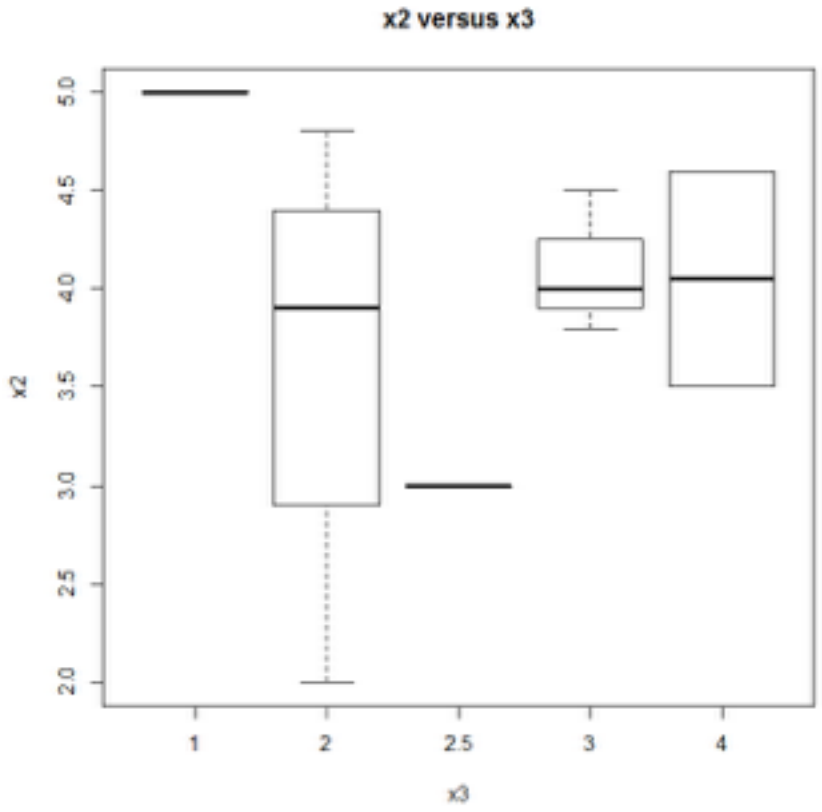
- abline() – adds one or more straight lines to a plot
- lm() – function to fit linear regression model

```
> x1<-c(1:5,1:3)
> x2<-c(2,2,2,3,6,7,5,1)
> abline(lm(x2~x1))
```

Visualization in R

- Boxplot

boxplot(x2~x3,data=mydata,main="x2 versus x3", xlab="x3",ylab="x2")

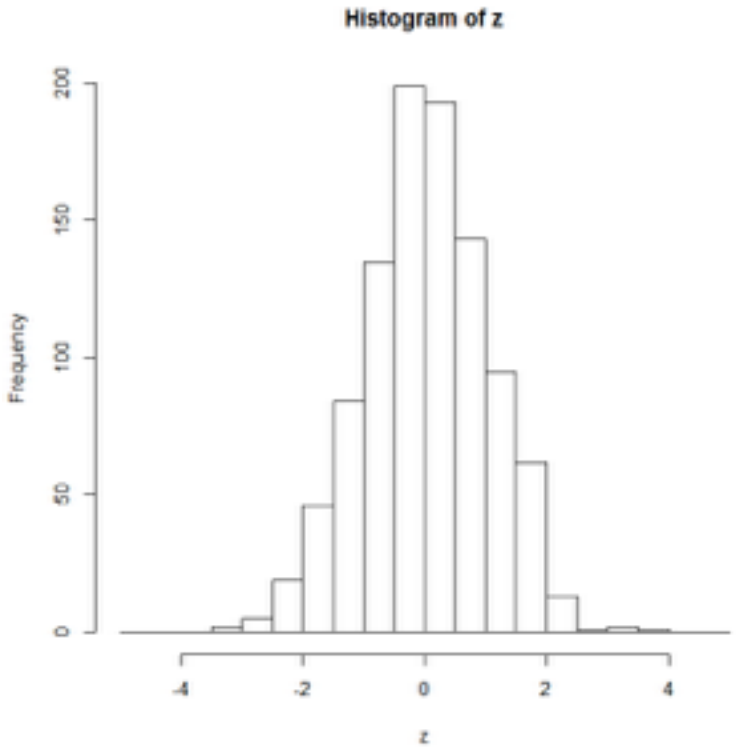


Visualization in R

- Histogram

```
z = rnorm(1000, mean=0,sd=1)
hist (z)
hist (z, breaks = 5)
bins = seq (-5, 5, by = 0.5)
hist (z, breaks = bins)
```

Generate 1000 samples of a standard normal random variable



Visualization in R

- scatterplot() – for 2D scatterplots
- scatterplot3d() – for 3D scatterplots
- plot3d() – for interactive 3D plots (rgl package)
- scatter3d() – another interactive 3d plot (Rcmdr package)
- dotchart() – for dot plots
- lines() – draws one or more lines
- pie() – for drawing pie chart
- pie3d() – for drawing 3D exploded pie charts (plotrix package)

Questions?