Contents

Data Scientist Toolbox

John's Hopkins University
Jeff Leek, Roger Peng, Brian Caffo

April 2, 2016

Part I

Week 1: What do data scientists do?

- define the question
- identify the ideal data set
- gather data (from database or web)
- clean the data
- Perform exploratory analysis (plots, clustering)
- Statistical prediction/modeling
- interpret/challenge results
- synthesize/write up results
- create reproducible code
- distribute results to others (interactive graphs, etc...)

Part II

Command Line Interface (CLI)

Use **Git Bash** from **Git** for windows to open a terminal.

0.1 Command Line Interface

CLI commands follow this recipe: `command -flags arguments` :

`command`	the CLI command which does a specifc task
`flags`	flags
`arguments`	binary

List of CLI commands:

`pwd`	print working directory
`ls`	list files and folders in current directory
`ls -a`	list hidden/unhidden files/folders
`ls -al`	list details for hidden/unhidden files/folders
`cd folder`	go to *folder*
`cd..`	change directory
`mkdir fldr`	create folder *fldr*
`touch test_file`	Create an empty file
`cp fname fldr`	copy file *fname* in folder *fldr*
`cp -r fldr1 fldr2`	copy folder *fldr1* to *fldr2* (`r` flag stands for recursive)
`rm -r More_docs`	Delete entire directories and its content
`mv fname fldr1`	move file *fname* to *fldr1*
`mv fname renfname`	rename file *fname* to *renfname*
`echo Hello World!`	print
`date`	current date

Part III

Git(local) and GitHub(remote)

0.2 Configuration: Username and Email

Open **Git Batch**:

```
$ git config --global user.name "Your Name Here"
$ git config --global user.email "Your email@Here.com"
#Confirm config
$ git --list #Return: (email/username)
```

0.3 Create a GitHub Repository

0.3.1 Start a repository from scratch

1)	go to **Github.com** and click on new
2)	create a repository name and a brief description
3)	select **Public**
4)	check the box next to: "*Initialize this repository with a README*"

0.3.2 Create a local copy

1)	open **GitBash**
2)	create a directory where to store copy of the repository
3)	navigate to the directory
4)	check the box next to: "Initialize this repository with a README"
5)	initialize a local Git repository in this directory `$ git init`
6)	Point your local repository at the remote repository: `$ git remote add origin https://github.com/username/name_repo.git`

0.3.3 "Fork" another user's repository

Make a copy of the repository of someone else's:

1)	Go to the repository and click on Fork
2)	Make a local copy by "cloning"
3)	`$ git clone https://github.com/username/name_repo.git`

0.3.4 add to the index

Adding (new files) to local repository

`git add.`	adds all new files to be tracked by Git
`git add -u`	update tracking for files that changed names or were deleted
`git add A`	does both

0.3.5 Commiting

Commit to be saved as an intermediate version.

`git commit -m "message"`	*message* is a description of what you did This only update local repo

0.3.6 Pushing

Update on remote (Github)

```
$ git push
```

0.3.7 Branches

Sometime, you are working on a project with a version being used by many people. You may not want to edit that version, you can create a **branch**:

`git checkout -b branchename`	Create a branch
`git branch`	check what branch you are on
`git checkout master`	switch to master branch

0.3.8 Pull requests

If you "fork" someone's repo or have multiple branches you will both be working separately. If you want to merge in your changes into the other branch:

1)	Go to your branch
2)	Pull request (the request will be send to all parties)

Part IV

Markdown: .md

```
## This is a 2ndary heading
### This is a 3rdary heading
```

Part V

R for statistical computing

Program: R and R-studio

```
?rnorm  %access help file for function 'rnorm'
help.search("rnorm") %search help file
```

0.4 R-packages

```
a <- available.packages()  % check available packages
head(rownames(a),3) % show the names of the 1st 3 packages
install.packages("arg")  % install new package of name "arg"
install.packages(c("pack1", "pack2", "pack3"))  % install multiple packages
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%go to Install packages in R-studio and load R-packages
library(packagename}
search()  % show all functions in package
```

0.4.1 RTools

R-Tools is a collection of tools to build R-packages in Windows.

1)	Open R Studio
2)	Install the devtools R package
3)	`find.package("devtools")` to check if package is installed ('True')
4)	`install.packages("devtools")`
5)	`library(devtools)` to load the package