Course #1 In Data Science: Data Science Toolbox

# Table of Contents

[Table of Contents 1](#_Toc400866424)

[Week #1: Intro & Overview 1](#_Toc400866425)

[Week #2: Git & R Packages 3](#_Toc400866426)

[Lecture: CommandLine Interface (CLI) 3](#_Toc400866427)

[Lectures 2 & 3, 4: Intro to Git & GitHub 5](#_Toc400866428)

[Creating a GitHub Repository: 5](#_Toc400866429)

[To update /save a file onto GIthub from Git Bash: 6](#_Toc400866430)

[Lecture 5: Basic Git Commands: 7](#_Toc400866431)

[Pull Request – feature of Github 8](#_Toc400866432)

[Lecture 6: Basic Markdown 8](#_Toc400866433)

[Lectures 7-8: Installing R Packages & RTools 8](#_Toc400866434)

[Loading R packages 9](#_Toc400866435)

[Week #3: Conceptual 10](#_Toc400866436)

[Lecture #1: Topes of Data Science Questions 10](#_Toc400866437)

[Lecture #2: What is Data? 11](#_Toc400866438)

[Lecture #3: What is Big Data? 11](#_Toc400866439)

[Lecture #4: Experimental Design 11](#_Toc400866440)

[Issues to Remember: 12](#_Toc400866441)

[Summary: 13](#_Toc400866442)

# Week #1: Intro & Overview

What a Data Scientist Does:

1. Define the Question
2. Define the ideal data set
3. Determine what data you can access
4. Obtain the data
5. Clean the data
6. Exploratory data analysis
7. Statistical prediction/modeling
8. Interpret results
9. Challenge results
10. Synthesize/write up results
11. Create reproducible code
12. Distribute Results

Tools We’ll use:

1. R
2. RStudio
3. Github

Getting Help:

1. Use Forum
2. Help in R
   1. ?rnorm – gets you help file for this function (rnorm function)
   2. Help.search(“rnorm”) – don’t have to get function name exactly right
   3. Args(“rnorm”) – will show you arguments for a function
3. If type function in top, you get arguments below and link to help card.
4. How to ask an R question:
   1. What steps will reproduce the problem?
   2. What is the expected output?
   3. What did you see instead?
   4. What version of the product (e.g., packages, etc) are you using?
   5. What OS are you using?
5. How to ask a Data Analysis Question
   1. What question are you trying to answer?
   2. What steps/tools did you use to answer it?
   3. What did you expect to see?
   4. What did you see instead?
   5. What other solutions have you thought about?
6. Where to look for R types of questions? (see also <http://bit.ly/Ufaadn>)
   1. Search archives of class forum
   2. Read manual / help files
   3. Search Web
   4. Ask friend
   5. Post to forums
   6. Post to the R mailing list or StackOverflow (with the “[r]” tag).
7. Where to look for Data Analysis/Statistics
   1. Search archive of class forums
   2. Search Web, in Google search for “[data type] data analysis” or “[data type] R package”.
   3. Post to class forum
   4. Post to CrossValidated
   5. Know what your type of data are called:
      1. Biostatistics – Medical Data
      2. Data Science for web data
      3. Machine learning for data from compsci
      4. Natural language processing for data from texts
      5. Signal processing for data from electrical signals
      6. Business analytics for data on customers
      7. Econometrics for economic data
      8. Statistical process control for data about industrial processes

# Week #2: Git & R Packages

## Lecture: CommandLine Interface (CLI)

Root directory is represented by a slash

A home directory is represented by a tilde ~

Working directory is whatever directory you’re in at the time

We’ll use “Git bash” from “Start Menu-> Search for Git bash -> Open it.

Prompt is your name, followed by $... e.g., eric$

If type pwd after your prompt e.g., “eric$ pwd”, that will give you your path pwd = print working directory

Structure of commands in CLI: ***Commands flags arguments***

**Command** = specific task

**Flags** = options to give to command, preceded by a –

**Arguments** = what the command will modify

e.g., pwd is a command, but it requires no flags or arguments.

Clear will clear all commands

Eric$ clear

Ls will lists files and folders in current directory

Ls –a lists hidden and unhidden files and folders

Ls –al lists details for hidden and unhidden files and folders

a and al are flags bc they are preceeded by a dash

cd Music/Debussy Change directory, list directory you want to go to after

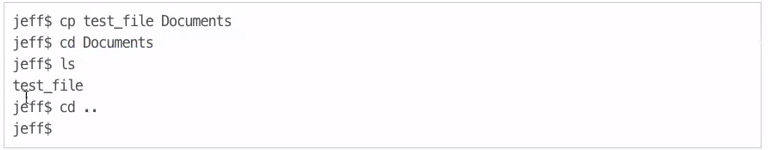
cd Without arguments takes you to home directory

cd .. goes to one level above your current directory

mkdir Make directory… create new folder just add the name of directory after

touch creates empty test file “test\_file”

cp is copy a file into another file. Arguments are the file you want to copy + directory you ant to put it in

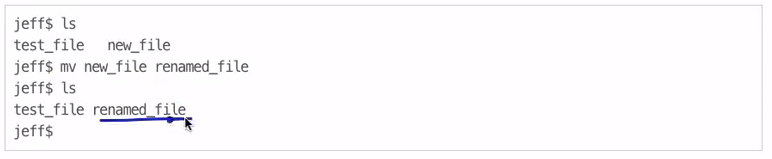


Cp –r Used to coy contents of directories. E.g., cp –r Documents More\_docs will copy contents fo “Documents” and place them in “More\_Docs”

Rm Remove… it deletes. Type rm and name of file,

Rm –r Will remove all files of entire directory. No undo, so be careful.

Mv This is move files between directories type mv *namefile foldername*. This command can also be used to rename files like so:



Echo will print whatever arguments you provide.

Date will print the date

**Summary of Commands:**

|  |  |
| --- | --- |
| Pwd | Print working directory |
| Clear | Clears screen |
| Ls | Lists file in working director |
| Cd | Change working directory |
| Mkdir | Creates new directory |
| Touch | Creates new file |
| Cp | Copies of file |
| Rm | Deletes a file or all files in a directory if used with –rrm - |
| Mv | Move a file or rename a file |
| Date | Print date |
| Echo | Print a particular command |
|  |  |

## Lectures 2 & 3, 4: Intro to Git & GitHub

Git = Free Open Source Version Control System

Stored on local machine and use

GitHub is

* web based hosting service for software development that uses GIT version control.
* Allows you to push/pull local repositories to Web.
* Gives you homepage of all your repositories.

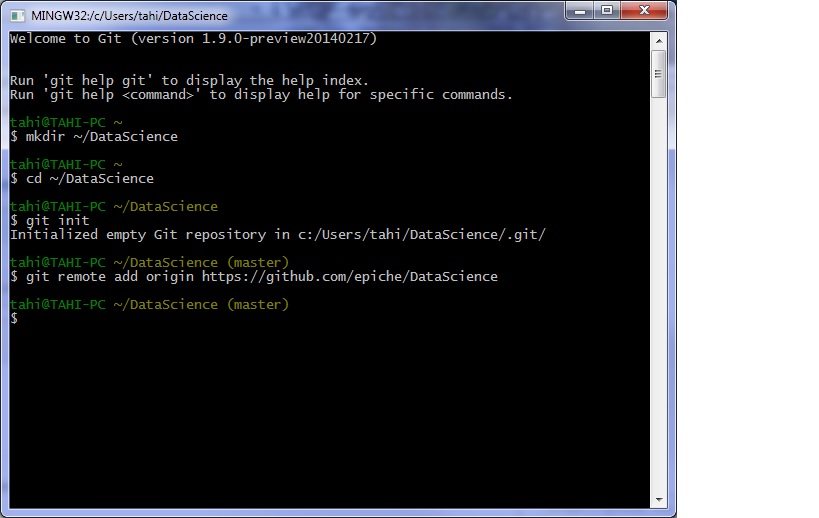
GitHub Account:

Username: epiche

Email: [ericfpiche@yahoo.ca](mailto:ericfpiche@yahoo.ca)

PW: asshole46

### Creating a GitHub Repository:



Fork: You can fork someone else Director, i.e., take what they have, split off from their version and hve your own in GitHub. <https://help.github.com/articles/fork-a-repo>

Next you need to create a local version so you can modify it in your local GIT. You do this by cloning. To clone in your current repository:

$ git clone <https://github.com/epiche/reponametoclonehere.git>

Later we’ll learn how to push changes you make up to the main repository.

Tutorials:

* <https://help.github.com/articles/for-a-repo>
* <http://git-scm.com/book/en/git-basics-getting-a-git-repository>

### To update /save a file onto GIthub from Git Bash:

From top-level directory, type

Cd ~/datasciencecoursera

You can then do pwd to test that you’re in the right working directory. Then type

git init

… and you’ll get message saying “Reinitialized existing GIT reporsitory in c:/Users/Eric/datasciencecoursera/.git/

Now that you are here… you can do a list files to see what’s in the local repo:

You can also do

Git status

And it will print files in your directory and if they are tracked or not. Currently, I have DummyR-Test\_Oct0814.r that is an untracked file. This is the one I need to commit to my GitHub repo

Connect to Github account:

Git remote add origin <https://github.com/epiche/datasciencecoursera.git>

(This creates a “remote” folder called “origin”

And double check to make sure it knows:

git remote -v

::::::::::::RECAP::::::::::

Type:

git status

This will list out all the files in your working directory.

Then type:

git add .

This adds in all of your files & changes so far to a staging area.

Then type:

git commit -m "first commit"

This commits all of your files, adding the message “first commit”

Next type:

git remote add origin https://github.com/username/rails-girls.git

Your GitHub Repository page will list the repository URL, so feel free to copy and paste from there, rather than typing it in manually. You can copy and paste the link from your GitHub repository page by clicking the clipboard icon next to the URL.

This creates a remote, or connection, named “origin” pointing at the GitHub repository you just created.

Then type:

git push -u origin master

This sends your commits in your “master” branch to GitHub

Congratulations your app is on GitHub! Go check it out by going to the same url you used above: https://github.com/username/rails-girls (without the .git part)

If you want to continue making changes and pushing them to GitHub you’ll just need to use the following three commands:

git add .

git commit -m "type your commit message here"

git push origin master

## Lecture 5: Basic Git Commands:

Here’s **a great tutorial** on installing GIT, setting up GitHub and how the two work together

Part 1: <http://readwrite.com/2013/09/30/understanding-github-a-journey-for-beginners-part-1#awesm=~oCp3dxAqMPX0gB>

Part 2: <http://readwrite.com/2013/10/02/github-for-beginners-part-2#awesm=~oCp7bwAlvJUvDh>

There are some good resources at the end of the 2nd tutorial.

These add files to local repo under version control.

You should do all these (add to index) before committing

|  |  |
| --- | --- |
| Git add . | Adds a new file |
| Git add –u | Updates tracking for files that changed names or were deleted |
| Git add -A | Does both of the previous |

Once you’ve indexed then you can commit them to your local repo with:

Git commit –m “message”

Message should be a useful summary of the changes being saved. So write something useful in between the quotes.

This is only local. To move to github you:

|  |  |
| --- | --- |
| Git push | Pushes files (local commits) to the remote (github) |
| Git checkout –b branchname | If want to work on existing project, you create a branch |
| Git branch | To see what branch you are on |
| Git checkout master | To switch back to master branch type |

### Pull Request – feature of Github

If you work on abranch after you forked someone’s repo you may want to merge back, so you need to send a “pull request”

Go to the repo and submit a request.

Git Documentation: <http://git-scm.com/doc>

Github help <https://help.github.com>

Or use Google or Stack Overflow

## Lecture 6: Basic Markdown

It’s a test file w/ extension .md

## this is a secondary heading

### this is a tertiary heading

\* Use asterisk to create lists with bullets

Markdown resources:

* <http://daringfireball.net/projects/markdown/>
* Click the MD button in Rstudio

## Lectures 7-8: Installing R Packages & RTools

Install.packages(“slidify”)

This command installs slidify by downloading it from CRAN (Comprehensive R Archive Network).

Any package that this depends on will also be installed by default

To install multiple packages:

Install.packages(c(“solidify”, “ggplot2”, “devtools”))

Can also install in Rstudio: Tools->Install Packages

Pick repository->Package from list

There’s also a section on installing packages from Bioconductor, but I left off notes

### Loading R packages

Gotta tell R to load packages to get access to functions:

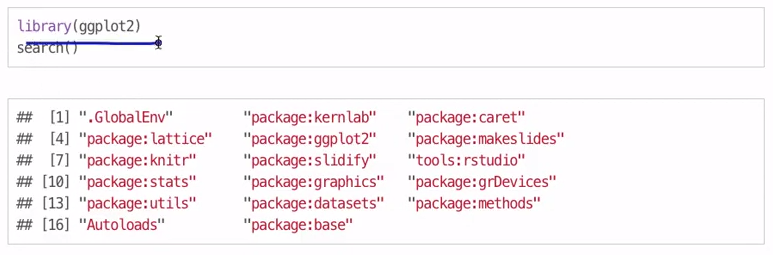
Library(ggplot2)

This is command to load ggplot2 package

Don’t put package names in quotes

Some packages give you message when loaded, some don’t

After you load package, the functions exported by that package will be attached ot tehtop of the search() list (after the workspace)



Installing RTools

Collection of tools to build packages in windows. Won’t be doing that here, but will later in course.

Download here:

<http://cran.r-project.org/bin/windows/rtools/>

# Week #3: Conceptual

## Lecture #1: Topes of Data Science Questions

In order of difficulty:

* **Descriptive:** 
  + Goal: First Kind of analysis, usually on census data
  + Description and interpretation are different steps
  + Description cannot be generalized without additional statistical modeling
* **Exploratory:**
  + Goal: Looking for relationships you didn’t know about
  + Not final say in analysis, but a start… Good for defining future studies
  + Not to be used for generalizing/predicting
  + Correlation does not imply causation
* **Inferential:**
  + Goal: Use small **sample** of data to say something about a bigger population
  + Inference is common goal of statistical models
  + Inference involves estimating both the quantity you care about and your uncertainty about your estimate
  + Inference depends heavily on both the population and the sampling scheme
* **Predictive:**
  + Goal: Use data on some objects to predict values for another object
  + If X predicts Y, it does not mean that X causes Y
  + Accurate prediction depends heavily on measuring the right variables
  + There are better and worse prediction models, but more data and a simple model works really well.
  + Prediction is really hard, especially about future references
* **Causal:**
  + Goal: To find out what happens to one variable when you make another variable change
  + Usually need randomized studies to ID causation
  + There are approaches for inferring causation for non-randomized studies, but they are complicated and sensitive to assumptions
  + Causal relationships are usually identified as average effects, but may not apply to every individual. So if we do X to a population, on average we’ll see effect of Y… but this does not apply to individuals.
  + Causal models are usually the gold standard for data analysis.
* **Mechanistic:** 
  + Goal: Understand the exact changes in variables that lead to changes in other variables for individual objects
  + Incredibly hard to infer, except in simple situations. Rare in data science.
  + Usually modeled by a deterministic set of equations (physical/engineering science)
  + Generally the only random component of the data is measurement error
  + If the equations are known but the parameters are not, they may be inferred with data analysis.

## Lecture #2: What is Data?

Data are values of qualitative or quantitative variables, belonging to a set of items

* Source: Wikipedia, entry on “data”

**Set of Items**: sometimes called a population, the set of objects you are interested in…

**Variables**: Measurements or characteristics of an item, height, pages on a site, gender

**Quantitative**: continuous scale, like height, weight, blood pressure

**Qualitative**: sex, treatment, country origin

What does data look like?

* Raw data is very messy
* Can be from an API (structured data)
* Medical records (text file, not formatted for your needs, usually extract what you need
* Data can be video, audio file (go to DarwinTunes for project on evolution of music).
* Data.gov has all kinds of data, from csv to excel to raw data.
* Data never looks like how you would like it.

Data is the 2nd most important thing in data science

* 1st most important thing is the question
* Data will limit or enable the questions. You might not be able to answer your question with the data you have, so you may have to modify your question.
* **KEY ISSUE:** **But having data can’t save you if you don’t have a question.**

## Lecture #3: What is Big Data?

Some data are so big, you can’t analyze on your machine, you need cloud to store data.

Most questions we try to answer don’t need the power associated with “Big Data”.

WE can now collect much more data, more cheaply than ever before.

Regardless of big or small, you need the RIGHT data:

*“The data may not contain the answer. The combination of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data…”*

* Tukey

## Lecture #4: Experimental Design

Must have a good and clearly stated analysis plan (statistical method)

Must have a plan to share your data and code (github and or figshare)

* See “The Leek group guide to data sharing: How to share data with a statistician”: <https://github.com/jtleek/datasharing>

Step #1: Formulate your question in advance

* Must ask a specific question
* E.g., does changing the text on the website improve donations?

Statistical inference: Take a sample from the overall population with a probability argument and show them the new treatment. Use descriptive statistics to see what donation levels were. Then use inferential statistics to see if findings are generalizable to broader population.

### Issues to Remember:

**Variability:** You look for small variability for results inside of each treatment, but for big difference between the two treatments, i.e., results within should be nicely clustered, but results between two groups should be different.

**Confounding**: Two variables that correlate are not related, but there is a third variable (confounder) that is confounding the relationship between your two variables. Must look at other variables that might be in your “model”… finally… “Correlation is not causation” Also, called Spurious Correlation.

You can deal with Confounders with:

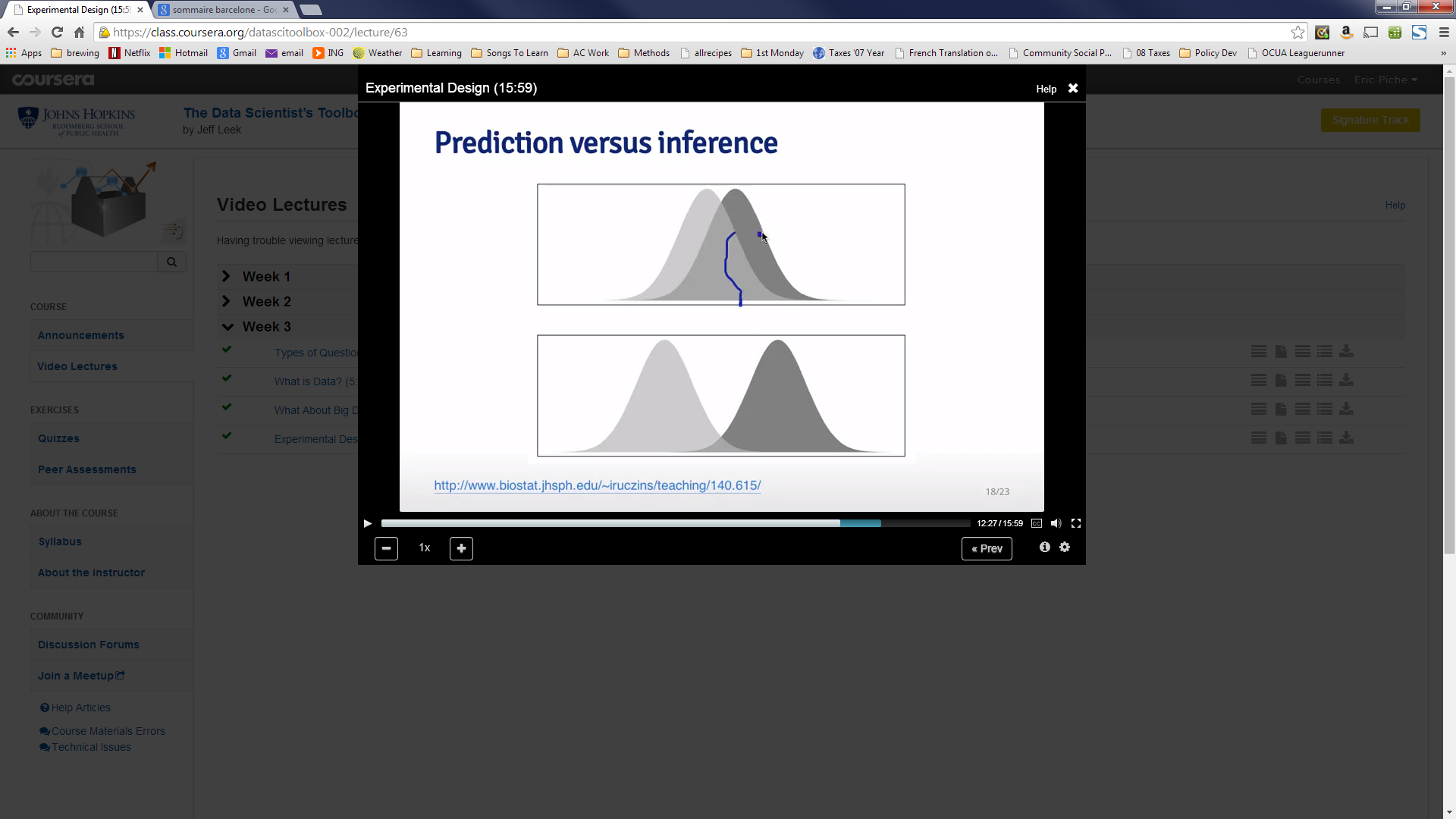
* **Fix** the variable. Get rid of it.
* **Stratify**: If you are testing a sign-up phrases and have to website colours, try each phrase on each colour to “Stratify” that way you can isolate the effect.
* **Randomize**: If can’t fix or stratify, then randomize the variable. Use a program to randomly assign the treatment to sample.

Prediction Study:

* Not well explained, but seemed to be where you take two samples from a population where you KNOW where they fall with regards to a treatment and then you “reverse engineer” a function that will accurately predict the results you have observed.

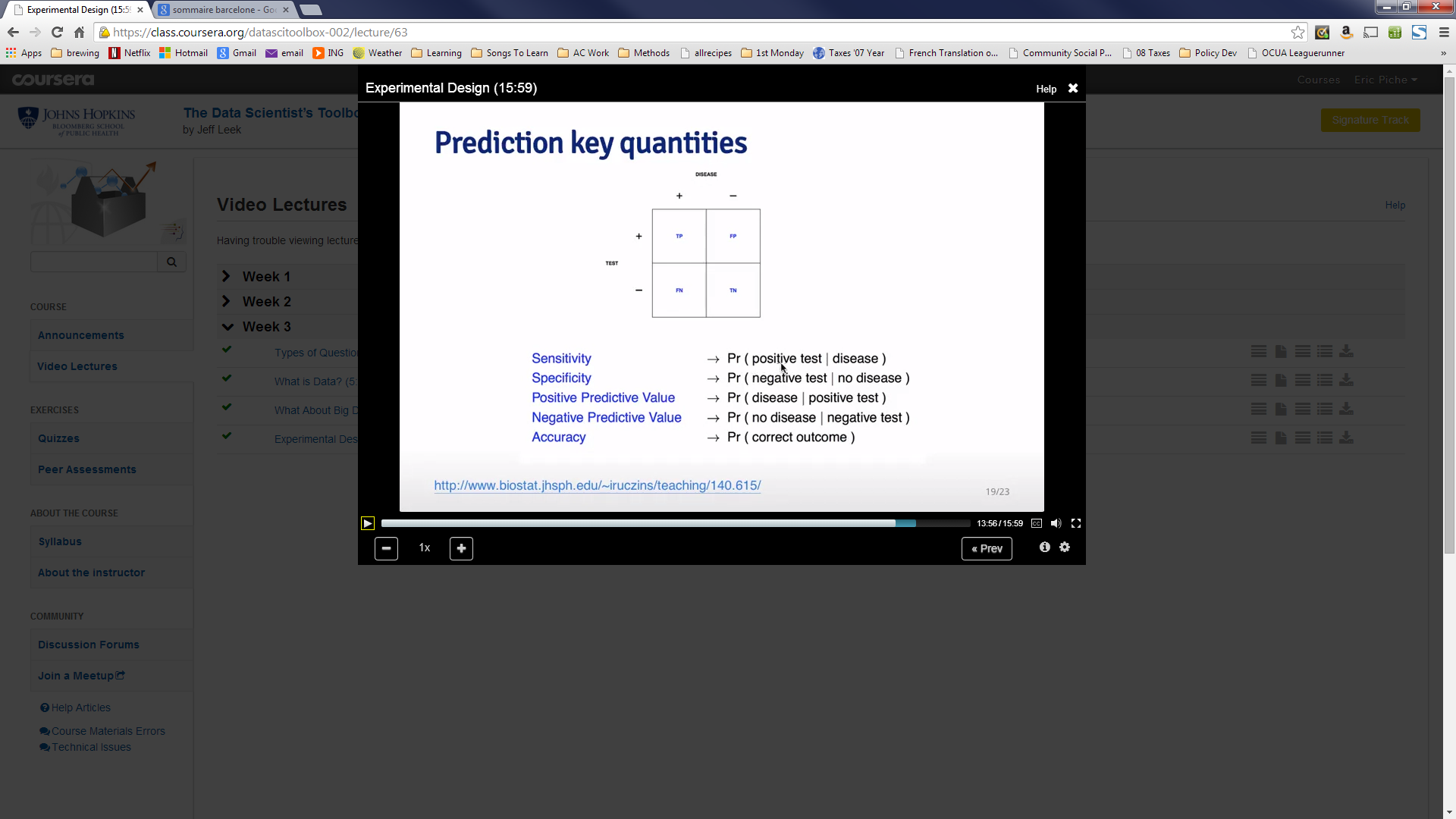
Prediction vs. Inference:

* Prediction harder than inference.
* You need good variation between the means of both populations. The top image has too much overlap, so if you had an observation along the blue line, you could not easily tell which population it came from. But on the below graph you could b/c there is little overlap. You need the populations separated for Prediction.



Prediction Key Quantities:

* You need to understand the probabilities listed below.
* “Pr” below stands for probability



Beware Data Dredging: This is where you don’t find the exact result you want initially, so you keep “dredging” the data until you find something that you like. Like the “Jelly Beans Cause Acne” example…

### Summary:

* Good experiments…
  + Have replication
  + Measure variability
  + Generatze to the problem you care about
  + Are transparent
* Prediction is not inference
  + Both can be important but it depends of what your application is.
* Beware data dredging