Syllabus: Data Bootcamp

ECON-UB.0232 | Fall 2017

Revised: August 22, 2017

Data Bootcamp is about nuts and bolts data analysis. You will learn about economic, financial, and business data, and enough about computer programming to work with it effectively. Applications include some or all of: leading economic indicators; emerging market country indicators; bond and equity returns; stock options; education and income; income by zip code; long tail sales data; innovation diffusion curves; and many others.

We will use Python, a popular high-level computer language that's widely used in finance, consulting, technology, and other parts of the business world. "High-level" means it's less painful than most (the hard work is done by the language), but it's a serious language with extensive capabilities.

"Data analysis" means primarily graphical descriptions that summarize data in ways that are helpful and informative. "Bootcamp" is a reminder that expertise takes work. Don't worry, it's worth it. **You will be more valuable to current and future employers.** And you will be able to do more things more efficiently than friends who rely on Excel.

If you're not convinced—or even if you are—we have a more elaborate sales pitch on the course website.

About the Instructor

Professor Michael Waugh
Email: mwaugh@stern.nyu.edu
Office: KMC 7-74
Phone: 212-998-0288
Office hours: TBA, Monday and Wednesday.

Where and When

Meeting times: Monday and Wednesday 330-445pm

Meeting place: KMC 5-140

Requirements

There are no prerequisites. **We welcome students with no prior programming experience** and have designed the course with them in mind. What you need is the **courage** to take on

a challenge and the **patience** to fix computer programs that don't work. That's a regular occurrence, even for experts.

Our one requirement is that **you must bring a laptop computer to class**. It should be your own computer, or at least one you can install new programs on. We will use it constantly in class, writing and correcting short programs.

Getting help

This course has a strong **support system** to help you when you run into problems—and anyone who codes runs into problems. We have a myself and a teaching fellow to help. Below is the contact info for our teaching fellow this semester.

Teaching Fellows

Felipe Alves
Email: falves@stern.nyu.edu
Office hours: TBA

Guess what. . . you also have classmates that can help!!! You can post questions (and maybe answer your classmates questions) on our discussion group (see below). Felipe and I will also monitor the discussion group.

The bottom line: **If you're stuck, ask for help**. Really. Don't be a hero, ask for help.

Course website and discussion groups

Everything you need, including this document, is posted on the **course website**:

https://nyu.data-bootcamp.com/undergrad_outline/

The **book** has its own site,

https://www.gitbook.com/book/nyudatabootcamp/data-bootcamp/details

There's a link to it on the course site. We will not use NYU Classes.

We have set up **announcement and discussion groups** (Google groups) to make announcements and post questions. We will use this to post all assignments and answer keys. If you have coding or other questions, we encourage you to post them there. If you know the answer to someone else's question, we **strongly** encourage you to post that, too. The links are

https://groups.google.com/d/forum/databootcamp_fall2017_undergrad

You can access the same links from the course website. Follow the instructions to join under whatever email address you plan to use.

Deliverables and grades

The course divides naturally into two parts. The first part is an introduction to those aspects of the Python programming language useful for data analysis. We cover this material with as many applications to real data as we can think of. It ends with an exam. The second part covers advanced topics and ends with a project of your own. The goal is for you to have a piece of work you can show potential employers to illustrate your quantitative skill set. Both parts include a number of graded deliverables. The idea is to **do some work all the time rather than lots of work once in a while.** We don't believe the latter will work.

Graded work includes:

- **Code Practice.** There are three such assignments in the first month. We encourage you do to all of them — they're good practice — but your grade will be based on the best two. We find that people who finish these assignments tend to keep up with the material better and these are easy points to get in terms of grades. We will also distribute Optional Code Practice "assignments," but they will not be collected or graded. They are, however, a good way to develop your coding skills.

  These assignments have questions that range from easy to moderately difficult, the latter marked challenging. They're all good practice, but if you're new to programming you might want to skip the challenging ones. If you get 70-80 percent of them, you're on the right track.

- **Examination.** The exam will cover material from the first half of the course. It takes 75 minutes and will be held in class. It will involve coding. **Bring your computer.** We suggest you also **bring one page of notes**. If the internet is working you will be able to look things up on there. If not, your notes will save you. It's also a good study tool: When you decide what to include, you'll be organizing your thoughts about what you've learned.

- **Project.** We work our way up to the project one step at a time, starting with idea generation and ending with a **professional piece of data collection and analysis** that you can share with potential employers. The structure of the project is laid out in a separate document. We have found that the quality of final project had a surprisingly low correlation to previous programming experience. A little thought and effort go a long way in creating an interesting project.

Due dates are posted on the course website. Assignments, whether code practice or components of the project, are due at the start of class on the specified dates.

Dates are not negotiable. Anything handed in late will get a grade of zero.

All your work should be clean and professional. Your grade depends on it.

Final grades will be computed from

Code Practice (best two of three)	20%
Exam	30%
Project	50%

Final grades are not subject to any fixed distribution or curve. The number of A grades, for example, will depend only on your performance in the course. If you make a good-faith effort, we expect it to be hard to get less than a B. We are the sole judges of what constitutes good-faith effort.

Recommended work habits

Python is not something you can learn from reading a book and attending lectures. You need to **write programs** — the more the better — to understand how they work. Think about how you'd learn to play basketball or soccer; reading and listening to lectures aren't enough, you need to do it. We'll do a lot of programming in class, but it's essential that you follow up outside of class. Here's how.

Write & Review. After each topic, we recommend you:

- *Write:* Shortly after class, write down everything you remember without looking at your notes or the book. Note things you don't understand—gaps, we call them.
- *Review:* Read the relevant section of the book. Fill in the gaps. Ask for help with anything you still don't understand.

Practice. For the first half of the term, each topic has an assignment that covers the same material. We suggest you do them, even the ones that aren't graded.

We also recommend you **practice coding** whenever you have the chance. Start small. Write short programs to do anything that crosses your mind. Use Python to do things you would ordinarily do in Excel. Try doing assignments from other courses in Python. At first this will be more work than doing it by hand or in Excel, but once you have some experience it will typically be easier in Python. Even if that's not the case, the practice will expand your skill set.

Pacing

The course is designed to be cover material at whatever pace the class is capable of. The topics should take roughly one week each, but we can scale that up or down as needed. If you're an expert, don't worry, we'll cover a lot of material either way.

Other questions

You can find answers to common questions on the course website. You can also post questions on the discussion group; see the link above and on the course website. For others, email me or the teaching fellow.

Viewing and printing pdf files

This document and most others in the course are pdf's that we hand out in class and post on GitHub. If you view them online, the links won't work; that's an unfortunate feature of GitHub's pdf viewer. You should instead download them to your own computer and open them there. To download them, click on the Raw button above the document. To view them, use Adobe Acrobat Reader or the equivalent (FoxIt, Sumatra, etc). The links should then work.

For Mac OS users, Preview has this problem and others; for example, links don't work and some fonts don't display or print. The solution is to install Adobe Acrobat Reader or the equivalent and set it up as your default for pdf files.

Policies

Ethics, disabilities, and many other things are governed by NYU and Stern policies. If you have questions about them, please ask.

On graded work: You may discuss assignments with anyone (in fact, we encourage it), but anything you submit, including your code, should be your own. Exams should be entirely your own work.

On disabilities: If you have a qualified disability that requires academic accommodation, please contact the Moses Center for Students with Disabilities (CSD, 212-998-4980) and ask them to send us a letter verifying your registration and outlining the accommodation they recommend. If you need to take an exam at the CSD, you must submit a completed Exam Accommodations Form to them at least one week prior to the scheduled exam time to be assured accommodation.