Nicholas Eubank
Assistant Research Professor

MIDS NUMBER
Fall & 2019

Programming for Data Science

1 Course Description

The aim of this course is to provide an introduction to the principles and concepts of programming. While there will be many similarities between this course and an introductory computer science course, this course is designed specifically for data science, and as such will emphasize methods for analyzing real world data rather than the "software development" skills (learning to write applications and programs) often taught in computer science introductory courses.

The course will be focused in large part on teaching students R, an extremely popular statistical programming language. However, this is not a course *about* R; rather, while we will use R in this course, we will be doing so as a means of teaching generalizable principles that will apply in any programming language.

In addition to working with R, this course will also provide training in a number of ancillary tools that are often overlooked in the training of data scientists, but which are absolutely critical to the day-to-day life of a data scientist, including:

- Git and Github (for collaboration and project management)
- The Command Line / Terminal
- Getting Help Online (no seriously – there's more to it than you may think!)
- Troubleshooting problems and debugging

2 Learning Goals

By the end of Programming for Data Science, our goal is for you to be able to do all of the following:

R Learning Goals

The following are our R-specific learning goals. Learning these skills in R will make learning to do these things in other language much easier, but our focus in this class will just be to empower you to do these things in R.

- Load data of various formats into R

- Manipulate the data in basic ways (like tabulating results)
- Clean and merge *real world dirty data* for analysis

General Programming Goals

These are goals we will explore in R, but which are inherently more general, and the skills you learn here apply as-learned in other languages and contexts.

- Organize your workflow for a project
- Program in a manner that minimizes the likelihood of mistakes, and maximizes the likelihood that when mistakes occur, you will catch them
- Find and fix problems (debug) your code

Ancillary Programming Skills

These are skills that are rarely taught in classes, but which are absolutely critical to being a successful data scientists in the real world.

- Collaborate with others using git and github
- Comfortably do basic operations in the command line / terminal.
- Find help online when you get stuck

3 Required Programming Background

None.

We have *absolutely no expectations that students will have any experience with programming!* This is an introductory course, and save MIDS bootcamp programs, we fully recognize that many students have never worked with statistical software. That's FINE! If you know how to use google and email, you have plenty of experience – everything else we'll take care of.

If do have experience with a programming language, however, worry not: in my experience, most people who learned to use tools like R, Stata, or Matlab did so in a somewhat haphazard manner. They were given some code, they learned to emulate it, and they can now stitch together code that does what they want. But most students were never taught any of the organizing principles of programming. So if you are one of those students, you may find parts of this course easier than other students, but you will still come away with a new, deeper understanding of these tools that should make you more comfortable and productive in your life as a data scientist.

Note that at times during this course, I may make statements about how the tools we're learning (like R) compare to different tools (like Stata or Python) if there are students with experience in these other tools in the class. However, if we make those comparisons, it is only to help those students the traps one can fall in if one's background is in another language. It is in no way because we *expect* all students to have experience with other tools.

4 R

In this class we will be focused on teaching you to use a program for statistical analysis called *R* (yup, just the letter).

Why R? Because it's currently one of the two most-used programs in data science (the other being Python, which we'll work with in Advanced Programming for Data Science), which means there is a good chance you'll be called upon to use it when working in teams. Moreover, it's a much easier tool to get started with than a language like Python.

It is worth emphasizing that we're not learning R because it is necessarily the "the best" language. The reality is that there are *lots* of tools for statistical programming, and each has its own strengths and weaknesses (e.g. R, Stata, SPSS, Python, Julia, Matlab, etc., etc.). People often develop strong opinions about which language is *best*, and sometimes pass judgement on people who use other languages. Every programming language has its strengths and weaknesses, and what is "best" depends on your use-case (the types of things you are using the language to do). This is true not only because languages themselves have strengths and weaknesses, but also because the tools and packages that have been created for use in different languages differ (e.g. people just haven't made a good package for doing geo-spatial work in Julia yet, for example). And if you're working on teams, you'll also have to make decisions based on the backgrounds of your tool sets. All of which is to say: there is no single *best* language for all purposes. But R is a very popular, strong, general purpose language, so will serve as a great starting point.

As a result, over the course of your career you may find yourself gravitating to one tool or another as required by your research. But in providing you with a firm foundation in a very popular language like R, you will not only be learning a tool that will allow you to do most everything you'll want to do in graduate school, but you will also be providing yourself with a solid foundation in *generalizable* skills that you will find useful if you later change platforms.

5 Class Organization

In this class we will be "flipping the classroom" – that means that most weeks, you will be **required** to review tutorials between classes so we can spend our class time doing hands-on programming exercises in an environment where help will be available. These tutorials will not generally be very long, and I **strongly** recommend that while you read through them you do so with an open programming session so you can just play around a little, trying out the things you learn. The research on learning to program is exceedingly clear on this point: **the only way to learn to program is to actually program**, so the more time you spend playing with the tools we are using, making mistakes, and troubleshooting, the more you will learn.

6 Course Schedule

PART I. R

Week 1: Intro to R

Class 1: Getting to Know R

- Read Before Class: Welcome to R!

Class 2: Vectors

- Read Before Class: Vectors

Week 2: Intro to R (Continued); Cleaning and Manipulation

Class 1: Datasets

- Read Before Class: dataframes

Class 2: Manipulating Data

- Read Before Class: Manipulating Data

Week 3: Merging Data; Plotting

Class 1: Merging Data

- Read Before Class: Merging Data

Class 2: Plotting

- Read Before Class: Plotting Data

Week 4: Loops and Functions

Class 1: For-Loops

- Read Before Class: Loops

Class 2: Functions

- Read Before Class: Functions

Week 5: Collapsing and Reshaping; Functions

Class 1: Collapsing and Reshaping

- Read Before Class: Collapsing
- Read Before Class: Reshaping

Class 2: Functions

- Read Before Class: Functions

Week 6: R Wrap-Up

Class 1: Lists

- Read Before Class: Lists

Class 2: To Be Decided...

PART II. The Tools of Data Science No One Taught You

Week 8: The Terminal / Command Line

Class 1: Command Line Basics

- Read: What is the terminal?
- Do: DataCamp Intro to Shell for Data Science, Parts 1 (Modifying Files and Directories) and 2 (Manipulating Data)

Class 2: R & Anaconda

- Find intro to anaconda or something??

Week 9: Git and Github

Class 1: Git Basics

- Read: Git and Github
- Do: Git-It Tutorial
- Maybe: Software carpentry?

Class 2: Collaborating on Github

- Find materials!

Week 10: Getting help online; Jupyter Labs

Week 11: Debugging and Troubleshooting

Week 12: Workflow management

PART III. Programming, a CS Perspective

- Week 13: Defensive Programming, Decomposition
  - Read: Defensive Programming
  - Do: Exercises??
- Week 14: More on Data Types (Floats, Ints, Strings, etc.)

  - Read: Data Types
  - Do: Exercises??

PART IV. Other Languages

- Week 15: Trade-offs of Different languages; Python
- Week 16: Python Libraries for Data Science (numpy, pandas)