PSYC 489D: Introduction to Data Science for Non-Majors

Instructor:
Alec Solway, Ph.D.
asolway@umd.edu

Fall 2018

Course description

The target audience for this class are psychology, neuroscience, and others majors interested in data science, but which are outside the typical data science track which includes computer science, statistics, etc. We start from scratch, and very minimal background is assumed (described below). A large number of industry and academic jobs require basic programming and data analysis skills. This class represents an introduction to both. Students will learn to program in R and will briefly be introduced to Python, the two most popular programming languages for data science. Common constructs shared by a variety of procedural programming languages will be emphasized. The second goal of this class is to explore basic statistics and probability theory from a computational perspective. Students will simulate toy data sets which they will then analyze, as well as work with real data. The class is highly hands-on with a large number of in-class and homework based projects. Expect to work a lot and move quickly. Because of the hands-on nature of the class, the overall focus is more on application and execution rather than strict theory.

Class meeting time and location

Tuesday & Thursday, 3:30-4:45pm, BPS 0140
Office hours and location

By appointment, and “pop up” office hours preceding homework and project deadlines to be announced in class.

Prerequisites

Understanding of introductory statistics, probability theory, and research methods at the level of PSYC 200 & PSYC 300, and one semester of Calculus (we will not rely on the latter, but this requirement is used as a measure of quantitative maturity). No background in programming is required.

Required textbook


Free online reference for introductory statistics and probability theory

As a prerequisite for this course, you are expected to have a basic understanding of introductory statistics and probability theory. However, if you've only taken a couple of classes and haven't applied the techniques beyond the classroom, you may be rusty. We will review everything we need in class, but it would be highly beneficial to have a written reference you can read before and after class. If you still have and like your introductory stats/probability theory textbook, feel free to use that. Assuming most people don’t, a good reference is:


This book also doubles as a second introduction to R, if you find something unclear in the main textbook.

Study strategies and academic integrity

The teaching philosophy in this class is centered around “learning by doing.” Most assignments and assessments are based on hands-on projects completed either in the classroom or at home. Putting learning into practice is a much more powerful way of understanding and remembering things than relying purely on one-shot
tests (although often times the latter cannot be avoided for practical reasons), be-
cause 1) you have to retrieve what you’ve learned in a context dependent manner,
and 2) you are repeatedly made aware of what you don’t know, and are forced to
go back and learn it. With this philosophy in mind, you are allowed to help each
other through high-level discussions on the assignments. However, each individual
is responsible for learning all of the material, and you have to code/write and turn
in your own individual answers.

Some units will have a reading associated with them. It is advised that you read this
material at least twice, once before the scheduled class(es) and once after, if you
truly want to understand it. It is impossible to completely understand a brand new
technical subject in one reading. On your first pass, you don’t actually know what
you need to learn or how to read the material, so you take most of it in passively.
This allows you to get a glimpse of the “big picture”: what the broad concepts are
and how they might relate to one another. Subsequent readings are required to
start to fill in the details and to learn the material in a more active fashion, where
you can predict what might come next and what you have yet to learn.

**Special needs and disabilities**

If you require special accommodations, please present current documentation from
the Disability Support Service (DSS) before the schedule adjustment deadline.
More information on University policies can be found at
http://www.counseling.umd.edu/DSS/.

**Grading**

*In-class assignments*

There will be a number of in-class assignments during the semester. These will
not be graded, but you have to hand them in to show that you did them and get
credit.

*Homeworks*

There will be five homeworks during the course of the semester.

*Projects*

There will be two larger projects, in lieu of a mid-term and final exam. You will
largely work on these at home, but also have ample class time to work on them
and ask questions.
The relative weight of these is as follows:

- In-class assignments 40% (divided evenly)
- Homeworks 25% (divided evenly, 5% each)
- Mid-term project 15%
- Final project 20%

**Attendance and late homework**

This is a hands-on course, thus, it will be extremely difficult to succeed without regular attendance. Many classes will have in-class assignments that are due by the end of the class. You have two free “passes” that allow you to turn in an in-class assignment (not a homework) up to 48 hours late. Beyond these two passes, written documentation (e.g. from a physician) is required to justify handing in an assignment late. Unexcused late homeworks will not be graded and will receive an automatic zero. You are highly encouraged to discuss any missed material and get feedback during office hours, since almost all later topics in this course build on earlier ones.

**University-wide policies**

Please see http://www.ugst.umd.edu/courserelatedpolicies.html.

**Course outline**

Subject to change!
• What is data science? Scope and bias of this class
• Variables and data types
• Algorithms, loops, and conditional statements
• Functions
• Probability
• A generative perspective on sampling distributions and t-tests
• Linear and logistic regression
• Programming in Python
• Object-oriented programming
• Generating publication quality plots
• Hierarchical models
• Numeric function optimization