LIS590 Data Science in the Humanities

Ted Underwood tunder@illinois.edu

Class meets LIS 341, Tuesday 10-12:50.

Office hours: LIS room 307, Tuesday 2-3 pm and by appointment

Course description:

Over the last several decades, forms of analysis once restricted to structured datasets have been escaping from the lab to cast light on other aspects of human life. Analysts are finding patterns in things that don't initially look like numeric data at all — things like product reviews, postings on social media, music, novels, and art.

These new modes of analysis are starting to make waves in the academic disciplines we group together as "the humanities." But the techniques that illuminate the history of music or literature also have applications outside a university: the unstructured character of humanistic data often makes it a close fit for the messy problems one encounters in business and journalism, for instance.

The goal of this course is first, to introduce students to the foundations of data science, and then to explore the special problems that emerge when those analytical methods are applied to human expression.

We'll start by reviewing the basics of programming in Python, but then move fairly rapidly toward real applications. So although the course doesn't assume previous programming background, people who are approaching programming for the first time will admittedly find the first two weeks very intense. I'm going to schedule a couple of extra office hours in the first two weeks to help people who may be struggling with early problem sets.

Assignments:

There will be nine homework assignments, due Saturday (or Sunday by noon in a pinch). Often the homework will be a coding assignment, but in some cases I'll ask you to write a brief reflection on one of the readings. We'll be learning how to think about human history as well as how to analyze data.

There will be a midterm exam, but it's really located two thirds of the way through the course, and it will do some of the summative work that might ordinarily be done by a final exam.

We don't have a final exam, because I want to leave the end of the course free for a final project. The goal in the final project is quite simply to do some interesting data science with a real-world dataset that includes unstructured data. (In other words, it's not just a table of numbers.) I can help you find material to support questions about history, literature, and music, but contemporary social and economic life are also fair game. Around the middle of the semester, we'll start talking about the final project, so you can have some work in progress to present on the last day of class. (Anticipate a five-minute presentation supported, optionally, by one or two slides.) The project itself is due on the day when a final exam would ordinarily be assigned for this course (which is TBA, but in the exam period). The shape of the research product can be flexible, but I'm envisioning something that looks (roughly) like a 1250-word blog post, supported by a richly documented Jupyter notebook.

Grading:

nine weekly homeworks, 50% (lowest homework score will be dropped)

midterm exam, 25%

final project, 25%

Schedule

Note also that in-class exercises and homework will be available at https://github.com/tedunderwood/LIS590DSH

Tuesday

Jan 17

What is data science? What can it achieve in the humanities?

Intro to the command line. Data types, lists, and methods. The variable explorer in Spyder. Interactive input and output. Jupyter notebooks.

- A fun example (you can skim): Thomas Dimson, "Emojineering."
- Optional reading (I'll cover key points in class): Daniel Rosenberg, "Data Before the Fact"
- The Django Girls, "Introduction to the Command-Line Interface."
- Homework 1.

Jan 24

Basics of data manipulation in Python.

Reading files. Conditional statements and iteration. Basics of data visualization using Pandas.

- David Zentgraf, "What Every Programmer Absolutely, Positively Needs to Know about Encodings and Character Sets to Work with Text."
- John Behrens, "Principles and Procedures of Exploratory Data Analysis"
- Homework 2.

Jan 31

Exploratory data analysis with Pandas.

Introducing Numpy vectors and Pandas data frames more systematically.

Exploring relations between variables.

First example of statistical learning: linear regression.

- "Introduction to hypothesis testing."
- Just glance at "Numpy Quickstart Tutorial."
- Read chapter 1-4 of Pandas tutorial more carefully.
- Homework 3.

Feb 7

Representing language geometrically.

Text mining depends heavily on geometrical reasoning about language; we need to understand those geometric abstractions, and understand their limits. At the same time, we'll practice defining functions in Python, and introduce concepts of statistical validity and confidence.

- Chaps 4 and 5 from Widdows, *Geometry and Meaning*.
- Benjamin Schmidt, "Comparing Corpuses by Word Use"
- Vincent Spruyt, "The Curse of Dimensionality in Classification"
- David Robinson, "Text Analysis of Trump's Tweets Confirms He Writes Only the Angrier (Android) Half."
- Homework 4.

Feb 14

The mathematics of classification, and reasons to be wary of it.

We'll build our first classifier, using a Naïve Bayes algorithm.

- Victor Powell, Conditional probability explained visually.
- The Arbital Guide to the Bayes Rule.
- Miriam Posner, July 27, 2015, "What's Next: The Radical, Unrealized Potential of Digital Humanities"
- Hastie, Tibshirani, and Friedman, *Elements of Statistical Learning* pp. 1-22. (You can semi-skim this, going easy on yourself; I will review the key elements in class. But I wanted to give you a link to this text, because it reviews the material in a systematic, principled way, and will be good to return to.)
- Homework 5.

Feb 21

Some practical foundations of machine learning.

Bias/variance tradeoffs, model evaluation, mechanics of using scikit-learn.

- Stephanie Yee and Tony Chu, "A Visual Introduction to Machine Learning"
- Leo Breiman, "Statistical Modeling: The Two Cultures"
- Just Section 6, "The Predictive Culture's Secret Sauce," from David Donoho, "Fifty Years of Data Science."
- Homework 6.

Feb 28 **Dimensionality reduction; unsupervised learning.**

We'll introduce clustering algorithms, as well as common methods of dimension reduction, and reflect on their humanistic applications.

- Victor Powell, "Principal component analysis explained visually."
- Alison et al., "Quantitative Formalism."
- Stahnke, et al. "Probing Projections."
- Stephen Marche, "Literature is Not Data."

Mar 7 **Machine learning and human interpretation.**

We will continue to explore the botanical garden of machine learning algorithms: from "decision trees" to "random forests." At the same time, we will start to focus more sharply on the challenges of integrating machine-learning epistemologies into human patterns of investigation and argument.

- Horton et. al. "Mining Eighteenth Century Ontologies" (below)
- Sylvain Parasie, "Data-Driven Revelation? Epistemological Tensions in Investigative Journalism" *Digital Journalism* 2015 (below).
- Galit Shmueli, "To Explain or To Predict?" *Statistical Science* 2010.
- Sculley and Bradley M. Pasanek, "Meaning and Mining."

Mar 1

How to anchor new patterns in human experience.

The advantage of data science — that it reveals patterns we haven't seen before — can also be a weakness. How do you interpret a pattern that has no connection to a familiar reference point? We'll consider case studies in sentiment analysis and social network analysis.

Sentiment analysis.

- David Bamman "Validity."

Other lexicon-based analysis.

- Twenge et al., "Increases in Individualistic Words and Phrases in American Books, 1960-2008."
- Jurafsky et al., "Linguistic Markers of Status in Food Culture."

Social network analysis.

- Marten Düring. "The Dynamics of Helping Behaviour for Jewish Fugitives During the Second World War."
- Dames et al. "Extracting social networks from literary fiction."

Mar 28

Entity extraction. Maps and geographic information.

We'll use NLTK to extract place names from text, and plot those names on a map. Review problems caused by multiple hypothesis testing.

- Matthew Wilkens, "The Geographic Imagination of Civil War-Era American Fiction."
- Cameron Blevins, "Space, Nation, and the Triumph of Region."
- Homework 7.

Apr 4

80% of the work: data munging.

Getting data, cleaning it, versioning it. Web scraping, rsync, git, deduplication, fuzzy matching.

- Katie Rawson and Trevor Muñoz, "Against Cleaning."
- Homework 8.

April 11

In-class midterm exam.

April 18 **Topic modeling.**

- David Blei, "Probabilistic Topic Models."
- David Mimno, "Computational Historiography."
- Chang et al., "Reading Tea Leaves: How Humans Interpret Topic Models."
- Homework 9.

April 25 **Emerging topics and future directions.**

We'll look both at some relatively new methods, esp. based on neural networks, and at open questions about the significance and social impact of data science.

Latent variable models.

- David Blei, "Build, Compute, Critique, Repeat: Data Analysis with Latent Variable Models."

Neural networks.

- Ryan Heuser, "Word Vectors in the Eighteenth Century" (whole series).
- Julien Despois, "Finding the Genre of a Song with Deep Learning."
- Gaytas et al., "A Neural Algorithm of Artistic Style" (skim).

Implications.

- Hanna Wallach, "Big Data, Machine Learning, and the Social Sciences."
- Andrew Piper, "There Will Be Numbers"

Bigger datasets (optional reading).

- Thomas Wiecki, "Easily distributing a parallel iPython notebook."

May 2 **Presentations of final projects**

TBA *Final projects due.*

Other syllabi consulted:

This course was importantly influenced by

Benjamin Schmidt, “Humanities Data Analysis”

David Bamman, “Deconstructing Data Science”

Victoria Stodden, “Introduction to Data Science”

David Mimno, “Text Mining for History and Literature”

Andrew Piper, “Cultural Analytics: The Computational Study of Culture”

Initial preparation for “Data Science in the Humanities”:

If you have access to a laptop, please bring it to class; we’ll generally spend the first half of the class on lecture/discussion, and the second half doing some hands-on exploration of the new ideas we have covered. Before or during the first class, you’ll need to install some free software on the computer you’re using: the Anaconda distribution of Python 3.5. If you’re able to install it before class, that will accelerate things a little. But I can help in class if you run into obstacles.

We’ll be writing programs in Python, and the Anaconda distribution of Python has the advantage of coming with many extensions that we’ll need for data science. It does matter that you get Python version 3 — and preferably 3.5. If you already have another version of Python on your computer, that won’t pose a problem; downloading Anaconda won’t automatically overwrite the other version.

Get Anaconda rather than Miniconda; it comes with more packages, and with a couple of useful editors you can use to write and run Python programs. We’ll be using Jupyter notebooks very heavily, but you may also want to explore Spyder.

I'm going to try to provide guidance that translates across Windows, as well as the Linux and OSX operating systems. But I'm more familiar with Linux/OSX, and people using Windows may encounter occasional points of friction. If you have a Windows machine and feel adventurous, you might consider creating a Linux virtual machine *inside* your computer, where you can use exactly the same instructions as everyone else. This is not required; it's just an option for people who feel like trying it. But if you want to explore it, here are some (relatively) straightforward instructions. If you do create a virtual machine, you'll want to install Anaconda (and create your course folder) *inside* the virtual machine.

Otherwise, if you're just installing Anaconda directly on your computer, it doesn't matter where you put it. The default installation location is a good choice.

It does matter (a little) where you put other files related to the course. I recommend creating a folder for the course relatively near the top of your folder hierarchy, so you don't have to write long "file paths." Something like "`/Users/alice/datahum`" or "`/Users/alice/Dropbox/datahum`" could be a good choice. (On Windows machines, these paths may begin with something like "C:") Notice that I'm keeping directory names short *and not including special characters like spaces* in the name. Although spaces are technically allowed, they create minor complications for you later on.

Underneath your main datahum folder, I recommend creating subfolders right away labeled `code`, `data`, and `results`. You may also wish to have other subfolders like `readings`. But the key is to separate your code from data (inputs) and results (outputs). Otherwise you'll end up with a long file list that's hard to scan visually.