Syllabus for "Fundamentals of Digital Archeology"

- **Course:** [COSCS-445/COSCS-545]
- T/Th 4:05-5:20
- **Instructors:** Audris Mockus: audris@utk.edu and Rhema Linder: rlinder@utk.edu
- **TAs:** Ben Klein: bklein3@vols.utk.edu

Need help?

There are no stupid questions. However, it may be worth going over the following steps:

1. Think of what the right answer may be.
2. Search online: stack overflow, etc.
    - Code snippets: On GH gist.github.com or, if anyone contributes, for this class
    - Answers to questions: Stack Overflow
3. Look through issues.
4. Post the question as an issue.
5. Ask an instructor: email for 1-on-1 help, or set up a time to meet.

Objectives

The course will combine theoretical underpinning of big data with intense practice. In particular, approaches to ethical concerns, reproducibility of the results, absence of context, missing data, and incorrect data will be both discussed and practiced by writing programs to discover the data in the cloud, to retrieve it by scraping the deep web, and by structuring, storing, and sampling it in a way suitable for subsequent decision making. At the end of the course students will be able to discover, collect, and clean digital traces, to use such traces to construct meaningful measures, and to create tools that help with decision making.

Expected Outcomes

Upon completion, students will be able to discover, gather, and analyze digital traces, will learn how to avoid mistakes common in the analysis of low-quality data, and will have produced a working analytics application.

In particular, in addition to practicing critical thinking, students will acquire the following skills:

- Use Python and other tools to discover, retrieve, and process data.
- Use data management techniques to store data locally and in the cloud.
- Use data analysis methods to explore data and to make predictions.

Course Description

A great volume of complex data is generated as a result of human activities, including both work and play. To exploit that data for decision making it is necessary to create software that discovers, collects, and integrates the data.

Digital archeology relies on traces that are left over in the course of ordinary activities, for example the logs generated by sensors in mobile phones, the commits in version control systems, or the email sent and the documents edited by a knowledge worker. Understanding such traces is complicated in contrast to data collected using traditional measurement approaches.

Traditional approaches rely on a highly controlled and well-designed measurement system. In meteorology, for example, the temperature is taken in specially designed and carefully selected locations to avoid direct sunlight and to be at a fixed distance from the ground. Such measurement can then be trusted to represent these controlled conditions and the analysis of such data is, consequently, fairly straightforward.

The measurements from geolocation or other sensors in mobile phones are affected by numerous (yet not recorded) factors: was the phone kept in the pocket, was it indoors or outside? The devices are not calibrated or may not work properly, so the corresponding measurements would be inaccurate. Locations (without mobile phones) may not have any measurement, yet may be of the greatest interest. This lack of context and inaccurate or missing data necessitates fundamentally new approaches that rely on patterns of behavior to correct the data, to fill in missing observations, and to elucidate unrecorded context factors. These steps are needed to obtain meaningful results from a subsequent analysis.

The course will cover basic principles and effective practices to increase the integrity of the results obtained from voluminous but highly unreliable sources.

- Ethics: legal aspects, privacy, confidentiality, governance
- Reproducibility: version control, ipython notebook
- Fundamentals of big data analysis: extreme distributions, transformations, quantiles, sampling strategies, and logistic regression
- The nature of digital traces: lack of context, missing values, and incorrect data

Prerequisites

Students are expected to have basic programming skills, in particular, be able to use regular expressions, programming concepts such as variables, functions, loops, and data structures like lists and dictionaries (for example, COSC 365)

Being familiar with version control systems (e.g., COSC 340), Python (e.g., COSC 370), and introductory level probability (e.g., ECE 313) and statistics, such as, random variables, distributions and regression would be beneficial but is not expected. Everyone is expected, however, to be willing and highly motivated to catch up in the areas where they have gaps in the relevant skills.

All the assignments and projects for this class will use github and Python. Knowledge of Python is not a prerequisite for this course, provided you are comfortable learning on your own as needed. While we have strived to make the programming component of this course straightforward, we will not devote much time to teaching programming, Python syntax, or any of the libraries and APIs. You should feel comfortable with:

1. How to look up Python syntax on Google and StackOverflow.
2. Basic programming concepts like functions, loops, arrays, dictionaries, strings, and if statements.
3. How to learn new libraries by reading documentation and reusing examples.
4. Asking questions on StackOverflow or as a GitHub issue.

Requirements

These apply to real life, as well.

- Must apply "good programming style" learned in class
  - Optimize for readability
- Bonus points for:
  - Creativity (as long as requirements are fulfilled)

Team Tips

- Agree on an editor and environment that you're comfortable with.
- The person who's less experienced/comfortable should have more keyboard time.
- Switch who's "driving" regularly.
- Make sure to save the code and send it to others on the team.

Evaluation

- Class Participation – 15%: students are expected to read all material covered in a week and come to class prepared to take part in the classroom discussions (online). Asking and responding to

other student questions (issues) counts as a key factor for classroom participation. With online format and collaborative nature of the projects, this should not be hard to accomplish.

- Assignments - 40%: Each assignment will involve writing (or modifying a template of) a small Python program.
- Project - 45%: one original project done alone or in a group of 2 or 3 students. The project will explore one or more of the themes covered in the course that students find particularly compelling. The group needs to submit a project proposal (2 pages IEEE format) approximately 1.5 months before the end of term. The proposal should provide a brief motivation of the project, detailed discussion of the data that will be obtained or used in the project, along with a time-line of milestones, and expected outcome.

Other considerations

As a programmer you will never write anything from scratch, but will reuse code, frameworks, or ideas. You are encouraged to learn from the work of your peers. However, if you don't try to do it yourself, you will not learn. Deliberate practice (activities designed for the sole purpose of effectively improving specific aspects of an individual's performance) is the only way to reach perfection.

Please respect the terms of use and/or license of any code you find, and if you re-implement or duplicate an algorithm or code from elsewhere, credit the original source with an inline comment.

Resources

Materials

This class assumes you are confident with this material, but in case you need a brush-up...

- Python for beginners and Python Dictionaries

Other

- Mining the Social Web, 2nd Edition

Databases

- A MongoDB Schema Analyzer. One JavaScript file that you run with the mongo shell command on a database collection and it attempts to come up with a generalized schema of the datastore. It was also written about on the official MongoDB blog.

R and data analysis

- Modern Applied Statistics with S (4th Edition) by William N. Venables, Brian D. Ripley. ISBN0387954570
- R
- Code School
- Quick-R

Tutorials written as ipython-notebooks

- python-data-cleaning
- python tutorial for engineers

GitHub

- Git and GitHub
  - Official GitHub Help
  - GitHub Issues
  - Recommended resources
- GitHub Pages
  - Official site
  - Thinkful guide