CMN 152V SOCIAL SCIENCE WITH ONLINE DATA
WINTER 2020
***INSTRUCTOR*: PROF. SETH FREY**

CLASS INTERACTION

— Office hours: TBA
— For content questions, use Piazza
— For personal questions, send a private message on Piazza or Canvas, to ask questions or set up a meeting.
— Canvas Announcements will be common and important. Be sure that you have them activated.

OVERVIEW

Online social science methods use technology to push the bounds of what we can know about human interaction and communication. The digitization of communication has provided a wealth of data that is transforming the social sciences. But there's a gap between the important questions about human nature and the raw messy reality of code and data. In this class we will bridge the gap, learning enough of a popular programming language—Python—to discover things about ourselves, our friends, each other, and society, as we interact with the Internet in code. No prior skills are assumed.

LEARNING OBJECTIVES

By successfully completing this course, students should be
— Capable of thinking in code
— Able to write, skim, and tweak code
— Doing useful things on the Internet with code
— Inspired to want to code

COURSE CONTENT

WEEKLY GOALS

Each week has four goals:

Teaching a "coding ecosystem" concept,

Teaching a code concept and putting it into practice,

Illustrating the use of a web API, and

Providing an insight into society or provoking curiosity about it.

These goals recapitulate the learning objectives. Towards these goals, each week will include a code lesson, a web science demo, topic lectures, and a task requiring either programming or scientific hypothesis generation.

CODE ECOSYSTEM TOPIC LECTURES

Format. The code ecosystem topics lectures will be short video lectures

Purpose. These lectures will complement the "hard skill" programming pedagogy in other course content with "soft skill" insights into programming practice.

Assessment. This content will be assessed with topic questions embedded into video content

CODE LESSON

Format. Each lesson will be presented in an interactive "code notebook" and accompanied with a video walkthrough.

Purpose. The code lesson will teach basic programming concepts.

Assessment. Embedded in the code notebooks will low-stakes (multiple attempts) interactive formative micro-assessments, for fast high-granularity feedback. Complete notebooks will be submitted and graded for completion and correctness.

WEB SCIENCE DEMOS

Format. Each demo will be presented in a code notebook and accompanied with a video walkthrough.

Purpose. The web science demo will show complex code in action performing useful tasks in interaction with the Internet. These will also provide data-driven insights into a social or cultural phenomenon.

Assessment. Each will be assessed with brief webcam/screencast video submissions showing each student's ability to edit key demo functionality on the fly

CODE GENERATION AND HYPOTHESIS GENERATION

Format. Every other week, students will either generate code in problem sets, or hypotheses about web data. The problem sets will be administered in code notebooks. The hypothesis generation exercises, administered in Canvas, will elicit student's reasoning about social data from the week's content and ask students to generate alternative hypotheses to explain some aspect of the data, and suggest supplementary data that would distinguish those hypotheses.

Purpose. The coding exercises will test students' ability to author code from scratch. The hypothesis generation exercises will give students experience thinking about online data and its relationship to scientific inquiry.

Assessment. The programming exercises will be submitted and automatically graded for correctness, completion, and correct manner of completion. The hypothesis generation exercises will be manually graded by rubric

EXAMS

Format. There will be a mid-term and final exam. The content of these exams will be drawn primarily from the formative assessment questions embedded into the code lessons, with additional questions about the demo, topic, and problem set content.

Purpose. The mid-term and final will provide summative assessment of each of the above, with a focus on the lesson content's basic programming pedagogy.

Assessment. Both exams will be proctored. The best exam preparation is to review the interactive micro-assessments in the code lessons

COURSE MODULE CONTENT AND GOALS

MODULE ONE

Coding ecosystem topics

"The point of this course"
"Why code?"
"Generating hypotheses and conceiving tests"
Todo: View in Playposit, answering embedded questions

Code lesson

CODE NOTEBOOKS
Objectives
- Load a code notebook and download it
- Explore blocks of text and code
- Executing code
- How to copy, pasting, and edit code

Todo: Complete all micro-assessments and submit notebook

Web science demo

SMORGASBORD
Insight: *You can use code to interact with the Internet*
Todo: Record video showing targeted code edits

Hypothesis generation

TRANSLATION QUALITY
Todo: In the Code Lesson, automatic translations from some languages were worse than from others. Generate at least two hypotheses for why translations from two different languages might differ in quality. Discuss what additional data would make it possible to distinguish these alternative explanations.

COURSE CALENDAR

Week	Content
1	Module 1
2	Module 2
3	Module 3
4	Module 4
5	Exam prep and exam
6	Module 5
7	Module 6
8	Module 7
9	Module 8
10	Exam prep and exam

MODULE TWO

Coding ecosystem topics

"Thinking in code"
"Feeling worthy to code"
"Setting up capture for webcam and screen"
Todo: View in Playposit, answering embedded questions

Code lesson

PYTHON AS CALCULATOR
Objectives
- Chain arithmetic operations, like in elementary algebra
- Build sequences of commands and their output

Create and edit variables
Learn the purpose and syntax of modules
Bringing it together: Bike wheel picture frames

Todo: Complete all micro-assessments and submit notebook

Web science demo

KITTENS

Insight: *Kitten meme detection*

Todo: Record video showing code edits

Code Generation

PIZZA IN METRIC

Todo: Complete and submit

MODULE THREE

Coding ecosystem topics

"Getting help"

"Why Python?"

Todo: View in Playposit, answering embedded questions

Code lesson

PYTHON HAS DIFFERENT KINDS OF THINGS

Objectives

Discover more types of data beyond numbers
Represent text in code
What to do about errors
How to compare things
Bringing it together: Shake it off

Todo: Complete all micro-assessments and submit notebook

Web science demo

TWITTER API

Insight: *Your can explore social media in code*

Todo: Record video showing code edits

Hypothesis generation

TWEET GEOLOCATION

Todo: Comparing between San Francisco's Pier 39 and Japanese Tea Garden in the Web Science Demo, more tweets came from one than the other. Generate at least two hypotheses for why one location would have higher tweet traffic. Discuss what additional data would make it possible to distinguish these alternative explanations.

MODULE FOUR

Coding ecosystem topics

"How to 'debug' code"

"How to prepare for the exams"

Todo: View in Playposit, answering embedded questions

Code lesson

COLLECTIONS OF THINGS IN PYTHON

Objectives

Make lists of things and access those things
Change and grow lists
Recognize strings as lists of characters

Transform lists

Bringing it together: The history of the western calendar

Todo: Complete all micro-assessments and submit notebook

Web science demo

WEB HISTORY

Insight: *Which is bigger, the number of sites you watch, or the number of sites watching you?*

Todo: Record video showing targeted code edits

Code Generation

TEETHING

Todo: Complete and submit

MODULE FIVE

Coding ecosystem topics

"Thinking through code flow"

"Reading documentation"

Todo: View in Playposit, answering embedded questions

Code lesson

CODE FLOW

Objectives

Change data by doing something on every part of a list

Grow data by running the same line of code many times

Filter data that satisfies conditions

Bringing it together: Cleaning data

Todo: Complete all micro-assessments and submit notebook

Web science demo

WIKIPEDIA

Insight: *What is the probability that you will become famous?*

Todo: Record video showing targeted code edits

Hypothesis generation

FAME OVER TIME

Todo: Rerunning the code in the Web Science Demo for each year of the last century will produce the probabilities of becoming famous for people born in each year. Generate at least two hypotheses for why someone born a century ago would have a higher probability of becoming notable on Wikipedia, compared to someone born two decades ago. Discuss what additional data would make it possible to distinguish these alternative explanations.

MODULE SIX

Coding ecosystem topics

"How to skim code"

Todo: View in Playposit, answering embedded questions

Code lesson

DOING ONE THING IF ANOTHER

Objectives

Change data by doing something on every part of a list

Grow data by running the same line of code many times

Filter data that satisfies conditions

Bringing it together: Anagram detector

Todo: Complete all micro-assessments and submit notebook

Web science demo

GUTENBERG

Insight: *What's the longest word Shakespeare ever used? What's his longest palindrome? Read his complete works in seconds.*

Todo: Record video showing targeted code edits

Code Generation

CHOOSE YOUR OWN ADVENTURE

Todo: Complete and submit

MODULE SEVEN

Coding ecosystem topics

"Copying other people's code from the Internet"

"Hiding from the complexity"

Todo: View in Playposit, answering embedded questions

Code lesson

DICTIONARIES AND NESTED DICTIONARIES

Objectives

- Learn what dictionaries are and how to build, access, and change them
- Learn the value of nested dictionaries, and how to work with them
- Loop through a dictionary the way you've looped through lists
- Discover how nested dictionaries are used to represent things on the Internet
- Bringing it together: What is a tweet?

Todo: Complete all micro-assessments and submit notebook

Web science demo

INSTAGRAM

Insight: *Algorithmic bias in your feed*

Todo: Record video showing targeted code edits

Hypothesis generation

ALGORITHMIC BIAS

Todo: Generate at least two hypotheses for aspects of social media that might be vulnerable to algorithmic bias. Discuss what additional data would make it possible to test these predictions.

MODULE EIGHT

Coding ecosystem topics

"Breaking problems into parts"

"Becoming a good coder"

Todo: View in Playposit, answering embedded questions

Code lesson

ADDING FUNCTIONALITY TO CODE

Objectives

- Discover how to define functions of our own
- Draw pictures with "turtle graphics" as custom functions
- Experience benefits of "refactoring" code
- Bringing it together: Drawing a circle

Todo: Complete all micro-assessments and submit notebook

Web science demo

ROBOCALLS

Insight: *Junk text yourself, and robocall yourself*

Todo: Record video showing targeted code edits

Code Generation

CREATIVE COMBINATIONS

Todo: Complete and submit

GRADING

The grading will be as follows:

(15%) Coding ecosystem topics

Eight code notebooks, first worth 1%, others worth 2%

(20%) DEMOS, PROBLEM SETS, AND REFLECTIONS :

Demos and problem sets (15%)

Demos will be spot graded for correctness of code edits

Problem sets will be autograded

Hypothesis generation and reflection (5%)

Will be spot graded by hand by rubric

(5%) TOPIC LECTURES

Lecture "attendance" questions

(60%) EXAMS

Mid-term exam on Canvas, proctored 25%

Final exam on Canvas, proctored 35%

At 60%, most of your grade will proctored

(0%) ELECTIVE

Optional and ungraded work, and opportune feedback

GENERAL POLICIES

DUE DATES AND LATE SUBMISSIONS

Assignments are due Sundays at 8:55PM PST. Late submissions are still welcome. You will be penalized 15 % each late day until you reach 50%. After that, you can maximally receive 50% for all submissions until the Sunday of the last course Module. This means that if you did perfect on all course content, but put it all off until the last week, it would not be possible to pass the course.

SPOT CHECKING

Some assignments are graded on spot checks. Under spot checking, most submissions by most students will get full credit by default, but some will be selected randomly for close grading. When you are selected, if we find that your work was not up to standard, we reserve the right to review your previous submissions and re-evaluate assignments from past weeks that received the default maximum grade. Even if low quality submissions slip through the cracks for many weeks before getting caught, you should not breath easy.

CHEATING

You are responsible for abiding by the UC Davis Code of Conduct (https://ossja.ucdavis.edu/code-academic-conduct). Any suspicion of cheating of any kind will be reported to Student Judicial Affairs. Cheating will result in an 'F' in the course. Cheating includes (but is not limited to): sharing information during exams, taking photos of exams, discussing exams with late takers, having possession of or copy of a stolen exam (includes exams posted to work-sharing sites), turning in work that is not yours, taking exams for others, etc. You are expected to turn in your own work. Plagiarism will not be tolerated. If you use the work of another, you must appropriately cite that work. For example, I drew much of this text from my colleague Jeanette Ruiz.

In general, if you feel a temptation to cheat, you should ask why you are in college, and consider taking time off.

CODE AND COLLABORATION

In the world of code, cheating is less black as white. With the obvious exception of quizzes and exams, I encourage you to discuss course activities with your friends and classmates as you are working on them. You will definitely learn more in this class if you work with others than if you do not. Ask questions, answer questions, and share ideas liberally.

Learning cooperatively is different from sharing answers. You shouldn't be showing your code to other students, except to your project partner or to someone who has already submitted the assignment and is helping you finish. You shouldn't copy code from others or let others copy from you. If you are helping another student, don't just tell them the answer; they will learn very little and run into trouble on exams. Instead, try to guide them toward discovering the solution on their own. Problem solving practice is the key to progress in code.

Of course, cooperation has a limit, and that limit is sharing code.

— Do not copy code from any student who is not your partner.

— Do not allow any student other than your partner to copy code from you. This is also cheating

— Do not copy full solutions from online sources such as Stack Overflow, Pastebin, and GitHub. Piecemeal copying from such sources is encouraged.

— Do not post your solutions publicly during or after the semester.

RELIGIOUS OBSERVANCES

If you have a religious observance that conflicts with your participation in the course, please meet with me before the end of the second week of the term to discuss appropriate accommodations.

STUDENT ACCESSIBILITY NEEDS

Access is a right that students have. The Americans with Disabilities Act (ADA) is a federal anti-discrimination statute that provides comprehensive civil rights protection for persons with disabilities. Among other things, this legislation requires that all students with disabilities be guaranteed a learning environment that provides for reasonable accommodation of their disabilities.

Students requiring disability-related academic adjustments and services may consult the UC Davis Student Disability Center. For those with documented testing accommodations, please have your accommodations letter emailed to me ASAP. Also, it is your responsibility to contact me, at least one week before each exam to arrange to take your tests at an alternate location/for longer test periods. Tests must be taken on the same day as the rest of the class.

I recognize that students may have health, mental, or physical issues and concerns beyond those recognized by university services. I encourage you to ask for what you need in order to be successful in the course.