Applied Python, GLBL 6060
Jackson School for Global Affairs, Spring 2024
Syllabus[1]

Instructor: William Casey King, PhD
Email: Casey.king@yale.edu
Class Time: Tuesday, 6:00-7:50pm
Location: WTS: A-30

Teaching Assistant: TBD
Email: TBD

Office Hours: Tuesday: 3-4, Thursday 3-4, or by appointment
Office Location: Horchow Hall, 3rd floor, SW corner

Prerequisites:

1. A specific policy or research question must be submitted in advance of admission to the seminar.
2. GLBL5050 or equivalent with permission from the instructor.

Class Size: 10-person limit

Course Description:

In the rapidly evolving world of global affairs, the ability to analyze complex data and present actionable insights is not only useful, but also quickly becoming imperative. Applied Python for Global Affairs is an interdisciplinary course designed to equip students with practical Python programming skills tailored to the fields of international relations, global business, global health, economics, geopolitics, and global policy analysis. This course bridges the gap between theoretical knowledge of global affairs and the practical application of Python programming. Students hone their Python skills as the course moves from the fundamental to more advanced libraries and frameworks used in data science, machine learning and "AI," including the use of large and small language models (LLM, SLLs). The goal is for students to gain greater facility using Python for data collection, analysis, and visualization. Beyond greater technical fluency, the seminar develops students' ability to communicate sophisticated data-driven insights to a non-technical audience, including policymakers and international stakeholders.

Learning Objectives:

After completing this course, students will ...

- Can greater fluency in Python and foundational Python object types.
- Be able to design, reason about, and implement algorithms for solving computational problems.
- Be able to test and effectively debug programs.
- Know how to use Python to extract data from different types of files and other sources including the web.
- Know how to read, manipulate, describe, and visualize data using the NumPy and Pandas packages
- Be able to execute an exploratory analysis of a data set using Python

[1] The instructor reserves the right to modify the course as necessary, based on class cohort capability and pace.

- Be able to use LLM's in Python, with a focus on Anthropic's Claude
- Understand and be able to prompt engineer using Anthropic's Claude
- Understand and implement a basic deep learner on a classification task
- Understand and implement unsupervised machine learning techniques for clustering:

Instruction:

Student Commitment:

Learning to code well is easy, but requires a consistent application of effort, rather than tremendous effort at the last minute. Like fluency in any spoken language, which is best learned with daily practice, Python or any coding language is very much the same. In order for you to become Python competent, it will require considerable effort, a minimum of 8 hours per week outside of class and as much as 20 hours a week when projects are due. If you cannot allocate this amount of time, you should postpone your Python learning until you can make the necessary commitment. Obviously, the more time you are able to commit to the practice, the better your skills will be at the finish line.

Class meetings:

We meet once a week. In our meetings, I will present material via lecture. Some lectures will include in class exercises. You will be expected to share your screen and code in front of me and your fellow students. We will have quizzes. Why? Becomes fluency requires memorization. Quizzes are a helpful forcing function to make sure you're practicing coding, not just "ChatGPTing" all of your homework without really gaining proficiency.

Homework:

There are a number of weekly activities, projects, and in-class breakouts that require a computer, Python3, Jupyter, and an Anthropic subscription. Homework (weekly problem set) for that unit is assigned after class and due the following week.

Grading:

- 9 Weekly Assignments - 45% (5% each)
- Final Project - 20%
- Quizzes- 25%
- Participation and Attendance-10%

Online Tools:

- Class Github: https://github.com/Jacksonpython/spring_2024
- *Install and configure Git & Bash.*
- Anaconda with Jupyter Notebooks & Python3: https://anaconda.org/anaconda/python
- Slack:

  Use Slack to informally communicate with your classmates, ask questions, share problems and your successes (but don't share code).
- Stack Overflow: https://stackoverflow.com/
- ChatGPT
- Claude

Course Schedule:

Week1: Introduction to Course and Review

1. Syllabus review
    a. Github repo

b. Anaconda build of python
2. Review of data types
3. Review of sequence types

Week 2: Review
Homework: Unit 1: Due 11:59pm Monday
*Quiz on data and sequence types and methods
Content:
1. Review of control structures.
2. Regular expression
3. Review of functions
4. Some more advanced functions

Week 3: More Functions and Advanced Functions
Homework: Unit 2: Due 11:59 Monday
*Quiz on control structures

Week 4: Introduction to Prompt Engineering: Anthropic's Claude LLM
Homework: Unit 3: Due 11:59 Monday
*Quiz: Functions(cumulative)

Week 5: Classes in Python
Homework: Unit 4: Due 11:59 Monday
*Quiz on Anthropic Prompting

Week 6: Numpy and Pandas
Homework: Unit 5: Due 11:59 Monday

Week 7: Web scraping various methods including, Beautiful Soup, Headless Browsers and Claude LLM
Homework: Unit 6: Due 11:59 Monday

Week 8: Langchain Agent Based Model for LLM
Homework: Unit 7: Due 11:59 Monday

Week 9: Introduction to Neural Networks: Classification
Homework Unit 8: Due 11:59 Monday

Week 10: Introduction to Machine Learning: Unsupervised Learning: DBSCAN and K-Means Clustering
Homework Unit 9: Due 11:59 Monday

Week 11: Progress Presentation of Projects

Week 12: Presentations of Projects

Weekly Assignments:
The weekly assignments are designed to reinforce and extend the programming concepts covered in the lecture. A typical assignment consists of several programming exercises of varying difficulty. While some exercises can be completed in a single line of code, others may require students to combine commands in innovative ways, to design their own algorithms, and to navigate common sources of documentation. Students may consult with each other about the assignment but must write their own code and list their collaborators in their submissions. The expectation is that these assignments will take around 4 hours to complete each week. Depending on your experience with coding, this time estimate might be more or less. The weekly assignments are due at 11:59 PM the Monday before class.

Projects:
There is one large coding project. It is a group project that comes at the end of the course and involves the analysis of an actual data set using Python's system of data analysis packages. Further details about the projects will be given during the school term.

Participation:
Students are expected to participate in class activities, to contribute to discussions held in live section and on other platforms, to behave professionally towards classmates, and to help maintain a supportive atmosphere for education. Participation scores will be assigned based on these criteria.

Academic Integrity:

Avoiding Plagiarism:
Plagiarism is a serious academic offense, and students must take care not to copy code written by others. Beginning students sometimes have trouble identifying exactly when plagiarism takes place. Remember that it is generally fine to search for examples of code (e.g., on forums such as stackexchange). This is a normal part of programming and can help you learn. However, it is important that you understand the code you find and use what you learn to write your own statements. It is okay if a single line of code happens to match an example found on the internet, but you should not copy multiple lines at once. If in doubt, simply document the place you found your example code and ask your instructor for further guidance.

Chat GPT: Use it. But make sure you understand it.

Late/Extension Policy: We recognize that sometimes things happen in life outside the course, To help with these situations, we will be given 8 "late days" to use throughout the term as you see fit. Each late day gives you a 24 hour (or any part thereof) extension to any assignment in the course except the project presentations or final exam. Once you run out of late days, each 24 hour period (or any part thereof) results in a 10 percentage point deduction on the grade for that assignment. You can use a maximum of 2 late days on any single assignment. We will not be accepting any submissions more than 48 hours past the original due-date, even if you have late days. If you have **very** extenuating circumstances (for example, an emergency medical situation), please reach out to your section instructor. If you know an event is coming up that will make doing the assignment difficult, talk to your section instructor to maybe get the assignment slightly early. Plan your time accordingly!

Assignment Help:

If you are stuck on a problem:

1. Search for the error message or problem - stack overflow is a good coding help resource.
2. Keep trying to solve it on your own before you give up
3. Use ChatGPT
4. Post a message to slack.
5. Email instructor