Introduction to Python for Data Science

DACSS 611 (3 Cred.)

University of Massachusetts Amherst

Fall 2025

Instructor

Dr. Omer Yalcin
518 Thompson Hall
oyalcin@umass.edu

Office Hours

You can join office hours either in-person or on Zoom. Please book an appointment at https://omerfyalcin.youcanbook.me/. If the time slots listed there do not work for you, then please email for an appointment.

Course Description & Objectives

Python has gained immense popularity as a programming language due to its ability to handle diverse types of data, powerful libraries for data analysis, robust support for tasks such as web scraping and data extraction from online sources, and its widespread use in machine learning and deep learning communities. Python is known for its readability and ease of use, making it a favorite among beginners and seasoned programmers alike. This introductory course on Python for data science will focus on the essential tools that are particularly beneficial for social data scientists and data professionals. This course will provide you with a solid foundation in Python, and equip you with the necessary skills to effectively work with data using Python. The course sessions will feature a blend of theory and practical application, with the instructor delivering lectures and guiding students through hands-on coding exercises during class sessions. The course will start off with a general introduction to programming in Python, going through basic variables and data structures, functions and modules, and object oriented programming. It will then advance into topics more specific to data science: data manipulation, data frames, collecting data from APIs, static and dynamic web pages, and working with databases. At the end of the course, students will be able to use Python for writing programs, data collection, visualization, and management.

Prerequisites

Familiarity with basic programming and data science concepts is a plus, but not required.

Textbook

All required course material is either freely available on the internet or freely available to read for you through the UMass Library. We will use portions of the following books:

- **[PDA]** McKinney, Wes. 2022. *Python for Data Analysis.* O'Reilly Media, Inc. https://wesmckinney.com/book/
- **[PDSH]** VanderPlas, Jake. 2016. *Python Data Science Handbook: Essential Tools for Working with Data.* O'Reilly Media, Inc. https://jakevdp.github.io/PythonDataScienceHandbook/

. The syllabus is subject to change with reasonable advance notice.

- **[AtBS]** Sweigart, Al. 2019. *Automate the Boring Stuff with Python: Practical Programming for Total Beginners.* No Starch Press. https://automatetheboringstuff.com/

The Course Schedule section lists what the material is for every week. Those include links to other online material listed there.

Below are some resources that we will not directly use, but that may be useful as further reference both during and beyond this course:

- MIT OCW Introduction To Computer Science And Programming In Python
- MIT OCW Introduction To Computational Thinking And Data Science
- Python Programming MOOC 2024 (University of Helsinki, Department of Computer Science)
- Python for Data Science. 2022.
- Al Sweigert's many other Python books can be read online here: https://inventwithpython.com/
- Müller, Andreas C, and Sarah Guido. 2016. *Introduction to Machine Learning with Python: A Guide for Data Scientists.* O'Reilly Media, Inc. (Read online through UMass Library)
- Grus, Joel. 2019. *Data Science from Scratch: First Principles with Python.* O'Reilly Media (Read online through Umass Library)
- James, Gareth, et al. 2023. *An Introduction to Statistical Learning: with Applications in Python.* Springer Nature (Download at https://www.statlearning.com/)
- Tunstall, Lewis, Leandro Von Werra, and Thomas Wolf. 2022. *Natural Language Processing with Transformers.* O'Reilly Media, Inc. (Read online through Umass Library)
- Hapke, Hannes, Cole Howard, and Hobson Lane. 2019. *Natural Language Processing in Action: Understanding, Analyzing, and Generating Text with Python.* Simon / Schuster (Read online through Umass Library)
- Raschka, Sebastian, Yuxi Hayden Liu, and Vahid Mirjalili. 2022. *Machine Learning with PyTorch and Scikit-Learn: Develop Machine Learning and Deep Learning Models with Python.* Packt Publishing Ltd (Read online through Umass Library)

Learning Management System

Classroom material, including slides, resources, assignments, announcements, deadlines, and more will be posted in our learning management system, Canvas. Canvas can be accessed at https://www.umass.edu/it/canvas. We will also use Canvas for asynchronous discussion. More on this in the participation section.

Lectures

Some weeks, there will be pre-recorded short lecture videos posted on Canvas. Students should watch those lectures **before** attending class alongside doing all the required readings and working on other material (e.g. YouTube videos) listed in the Course Schedule. When doing the readings and watching lectures, it

is important that you have your coding environment open and implement what you see. Google Colab is the recommended environment and it is what will be used during Classroom demonstrations. Some of the resources that we will use, such as [**PDA**] and [**PDSH**] already provide ipynb files and, in the case of the latter, Google Colab launch links.

We will use synchronous class time to do demos and work on coding problems. It is essential that students come to classes having studied the week's material and ready to face live coding challenges. Please bring your laptop to class if possible, but if you cannot for some reason, do not that prevent you from attending (we have a few computers in the classroom that you may be able to use, alongside other solutions we can find. Just let me know!). All class sessions will be recorded via Zoom and posted on Canvas afterwards. Students who cannot attend class synchronously should watch them later.

Grading

Grades are calculated as follows:

- Participation (10%)
- Homeworks (50%)
- Final Project (40%)

Participation: It is imperative that students actively and regularly participate in class discussion. Canvas's discussion section will be the primary way of participating in class discussion asynchronously. Students are expected to regularly ask and/or answer questions about python / coding topics, assignments and so on. You can also participate by sharing insights, material, and other resources on Canvas. Participation does not need to reflect expertise. Synchronous participation during class is another additional opportunity to participate, but online students who may be unable to join synchronously will not be disadvantaged in grading.

Homework Assignments: There will be a homework assignment most weeks, a total of eight. The assignments will include exercises to help you better understand concepts and methods covered during class. Collaboration is acceptable, but please write up your own answers; do **not** hand in identical written responses. Two of the ten assignments will check your progress on the final project and provide you with timely feedback. So please get started on the final project early and start making progress. See which weeks we have assignments and which of them are related to the final in Course Schedule

Deadlines and Late Work: Most assignments will be due on a Sunday at 11:59 pm. (See exceptions at Course Schedule) Each assignment—but not the final project—will have one full day of a grace period following the deadline. That means you can return the assignment penalty-free until one day after the deadline, for example, at Monday at 11:59 pm for a deadline on Sunday, 11:59 pm. However, if you do make use of the grace period, then you may also receive feedback later than those that return on time. If, without a valid excuse, an assignment is returned even later than the end of the grace period, then it will be penalized by a 5% deduction of the full grade for every day that it is late. ("Day" here means anything that runs into the day. If the grace period ended on Monday 11:59 PM, then 20 minutes later, Tuesday at 12:20 am counts as 1 day late). If you think you will not be able to meet a deadline for a good reason and you contact me by email at least 24 hours in advance of the deadline, we can work out a new deadline.

Final Project: The final project will be in the form of a data science project that investigates a social scientific question or phenomenon using methods covered in the course. It should identify an online data source, collect data using techniques we cover (from APIs or directly from web pages), clean and organize the data for further processing, and apply analysis techniques that we cover in statistical modeling, image-as-data, and text-as-data and other weeks. The project should be written as one or multiple "Interactive Python Notebook" (.ipynb) files that combine text/narration, code, and code output. Details on the final project can be found on Canvas.

It is important that students make progress on the final project throughout the semester. To ensure that this is the case, two assignments out of the ten will check in on your final project progress.

Grading Rubric for Final Course Grade:

Final letter grades are assigned using the University's Plus-Minus Grading Scale according to following rubric:

- A (94-100%)
- A- (90-93%)
- B+ (86-89%)
- B (81-85%)
- B- (77-80%)
- C+ (74-76%)
- C (70-73%)
- F (Below 70%)

Email Policy

I expect you to have lots of questions throughout the course. Please ask them! When you have a question about course material, it is likely that others in the class have the same question. It is also likely that someone else in the class can answer the question faster or better. Therefore, students are encouraged to ask their questions on Canvas's discussion section, publicly. For questions that are relevant for you but not for other students or for other types of concerns, please feel free to email me. In most cases, you can expect a response within one business day.

Incomplete Grade Policy

For your reference, a copy of the essential sections of the UMass policy regarding Incomplete grades has been provided below. (More information can be found on page 28 of the following document: https://www.umass.edu/registrar/sites/default/files/2025-04/Academic%20Regulations_2024_2025%20Final%20%287%29.pdf?1755529372):

"Students who are unable to complete course requirements within the allotted time because of severe medical or personal problems may request a grade of Incomplete from the instructor of the course. Normally, incomplete grades are warranted only if a student is passing the course at the time of the request and if the course requirements can be completed by the end of the following semester. Instructors who turn in a grade of "INC" are required to leave a written record of the following information with the departmental office of the academic department under which the course is offered: (1) the percentage of work completed, (2) the grade earned by the student on the completed work, (3) a description of the work that remains to be completed, (4) a description of the method by which the student is to complete the unfinished work, and (5) the date by which the work is to be completed. In the case of an independent study where the entire grade is determined by one paper or project, the instructor should leave with the department information pertaining to the paper or project, which will complete the course. To avoid subsequent misunderstanding it is recommended that the student also be provided with a copy of this information.

Grades of Incomplete will be counted as F's until resolved. If not resolved by the end of the following semester, they will automatically be converted to an F if taken before Fall 2004, to an IF if taken thereafter. Faculty wishing to extend this deadline must write to the Registrar's Office stipulating a specific date by which the incomplete will be completed."

AI Policy

The use of AI (ChatGPT, Claude, Gemini, etc.) in this course is **not** banned, but neither is its excessive use encouraged. AI is best used as an assistant, **not** as an autonomous agent that will do your work for you. AI is more successful for small, concrete, narrowly-defined tasks. It is not very good at larger projects that require depth of understanding and creativity, but it can generate language that gives the impression of doing well. Remember that AI can "hallucinate" or make things up, and you are ultimately responsible for the work you submit—whether AI was used or not. If I do mention AI use as a problem in feedback on your work, it will not be because of AI use per se, but because the output is shallow, or it does not properly answer the question, or it ignores something we discussed in class or from the reading material, or otherwise does not meet expectations.

University Policies

University policies regarding Accommodations, Academic Honesty, and Title IX, apply to all courses. You can find those policies at https://www.umass.edu/senate/book/non-responsible-employee-required-syllabus-statements. They are also copied verbatim below.

Academic Integrity Statement

UMass Amherst is strongly committed to academic integrity, which is defined as completing all academic work without cheating, lying, stealing, or receiving unauthorized assistance from any other person, or us-

ing any source of information not appropriately authorized or attributed. As a community, we hold each other accountable and support each other's knowledge and understanding of academic integrity. Academic dishonesty is prohibited in all programs of the University and includes but is not limited to: Cheating, fabrication, plagiarism, lying, and facilitating dishonesty, via analogue and digital means. Sanctions may be imposed on any student who has committed or participated in an academic integrity infraction. Any person who has reason to believe that a student has committed an academic integrity infraction should bring such information to the attention of the appropriate course instructor as soon as possible. All students at the University of Massachusetts Amherst have read and acknowledged the Commitment to Academic Integrity and are knowingly responsible for completing all work with integrity and in accordance with the policy: (https://www.umass.edu/senate/book/academic-regulations-academic-integrity-policy)

Accommodation Statement

The University of Massachusetts Amherst is committed to providing an equal educational opportunity for all students. If you have a documented physical, psychological, or learning disability on file with Disability Services (DS), you may be eligible for reasonable academic accommodations to help you succeed in this course. If you have a documented disability that requires an accommodation, please notify me within the first two weeks of the semester so that we may make appropriate arrangements. For further information, please visit Disability Services (https://www.umass.edu/disability/)

Title IX Statement

In accordance with Title IX of the Education Amendments of 1972 that prohibits gender-based discrimination in educational settings that receive federal funds, the University of Massachusetts Amherst is committed to providing a safe learning environment for all students, free from all forms of discrimination, including sexual assault, sexual harassment, domestic violence, dating violence, stalking, and retaliation. This includes interactions in person or online through digital platforms and social media. Title IX also protects against discrimination on the basis of pregnancy, childbirth, false pregnancy, miscarriage, abortion, or related conditions, including recovery. There are resources here on campus to support you. A summary of the available Title IX resources (confidential and non-confidential) can be found at the following link: https://www.umass.edu/titleix/resources. You do not need to make a formal report to access them. If you need immediate support, you are not alone. Free and confidential support is available 24 hours a day / 7 days a week / 365 days a year at the SASA Hotline 413-545-0800.

Course Schedule

Week 1 (Sept 2 and 4): Python Basics

- **Summary:** This week we learn the basics of Python, including basic scalar (integer, float, boolean, None) and non-scalar (string) data types, how to write Python code using Google Colab / Jupyter Notebooks (our preferred coding environment for the purpose of this course), mathematical operations and variables.
- **Required**:
  - [**PDA**] Ch. 1 Preliminaries, Ch. 2. Python Langauge Basics (until "Control Flow")

- [**AtBS**] Ch. 1. Python Basics

Week 2 (Sept 9 and 11): Control Flow and Containers

- **Summary:** This week we continue the basics with understanding control flow (if, else, elif), iteration (for and while loops), and important container data types (i.e. data types that can act as collections of other data, such as lists, tuples, dictionaries and sets).
- **Required:**
  - [**PDA**] Ch.2.3 ("Control Flow" section), Ch. 3.1. Data Structures and Sequences
  - [**AtBS**] Ch. 2. Flow Control, Ch. 4. Lists, Ch. 5. Dictionaries and Structuring Data
- **Recommended:**
  - Official Python Tutorial Ch. 5. Data Structures
  - Watch: Conditionals and Booleans - If, Else, and Elif Statements
  - Watch: Loops and Iterations - For/While Loops
  - Watch: Lists, Tuples, and Sets
  - Watch: Dictionaries - Working with Key-Value Pairs
  - Watch: Comprehensions

★ **Assignment 1** due on Sunday, September 14.

Week 3 (Sept 16 and 18): Functions, Modules, OOP

- **Summary:** For maintainable, bug-free code, it is important that we write (or "define") our own functions. Modules are files that contain function definitions. Object Oriented Programming (OOP) allows us to define our own classes that bundle data and functionality. OOP is very useful for creating deep learning architectures, among many other areas.
- **Required:**
  - [**PDA**] 3.2. Functions
  - [**AtBS**] Ch. 3 Functions
  - Official Python Tutorial Ch. 6. Modules
  - Official Python Tutorial Ch. 9. Classes
  - Watch: Functions
  - Watch: Import Modules and Exploring the Standard Library
  - Watch: Python OOP Tutorial (complete all six videos in the playlist)
- **Recommended:**
  - Official Python Tutorial: Ch. 4.7. Defining Functions
  - Official Python Tutorial: Ch. 4.8. More on Defining Functions

★ **Assignment 2** due on Sunday, September 21.

Week 4 (Sept 23 and 25): Interacting with APIs

- **Summary:** The internet is a huge source of data for research. Many platforms provide APIs (Application Programming Interface) for software developers. APIs are also used by researchers, but more for data collection purposes. The `requests` library allows for sending HTTP requests, which we use when interacting with APIs. APIs usually provide data in the form of JSON or XML files, which we can convert to Python's built-in data structures.
- **Required:**
  - requests quickstart
  - Python API tutorial
  - JSON description (just read homepage)
  - XML files
  - JSON vs XML
  - Watch: Python Requests Tutorial

Week 5 (Sept 30 and Oct 2): Scraping Static Web Pages

- **Summary:** Sometimes we want to scrape a web page directly. Instead of the more machine-oriented JSON format, web pages are typically HTML documents. `BeautifulSoup` allows us to parse HTML documents so we can extract the part of the data we need and use in our workflows.
- **Required:**
  - BeautifulSoup Documentation (go through the "Quick Start" tutorial on the first page)
  - **[AtBS]** Web Scraping
  - Watch: Web Scraping with BeautifulSoup and Requests

★ **Assignment 3** due on Sunday, October 5.

Week 6 (Oct 7 and 9): Arrays

- **Summary:** Arrays from the `numpy` library are like vectors or matrices, but they can have any number of dimensions. They are very important to the Python data science ecosystem and act as building blocks for many other packages, including `pandas`. We will also begin to learn about tensors from PyTorch, a popular deep learning library. Tensors are like arrays, but they can keep track of their gradients.
- **Required:**
  - **[PDA]** Ch. 4 Numpy Basics
  - **[PDSH]** Ch. 2. Introduction to Numpy
- **Recommended:**
  - Watch: Python NumPy Tutorial for Beginners
  - Watch: PyTorch Tutorial 02 - Tensor Basics

Week 7 (Oct 14 and 16): Dataframes

- **Summary:** The `pandas` library provides dataframe functionality to Python. It is not the only one, but it is the most popular one—at least for now!
- **Required:**
  - [**PDA**] Ch. 5 pandas Basics
  - [**PDA**] Ch. 6 Data Loading, Storage, and File Formats
- **Recommended:**
  - Watch: Pandas Tutorial (this playlist is long and does not exactly correspond one-to-one to this week or next, so watch half this week and half next if you want)

★ **Assignment 4** due on Sunday, October 19.

Week 8 (Oct 21 and 23): Data Wrangling

- **Summary:** We continue with data wrangling on dataframes. We will also introduce `polars`, a newer, faster, growing dataframe library that is useful especially for larger data.
- **Required:**
  - [**PDA**] Ch. 7 Data Cleaning and Preparation
  - [**PDA**] Ch. 8 Data Wrangling: Join, Combine, and Reshape
  - [**PDSH**] Ch. 3 Data Manipulation with Pandas
  - Polars User Guide: Getting Started
- **Recommended:**
  - Modern Polars

Last day to Drop with "DR" is October 28, Tuesday.

Week 9 (Oct 28 and 30): Data Visualization

- **Summary:** Matplotlib is Python's most popular visualization library. Seaborn is built on top of Matplotlib and makes it easy to visualize data organized into pandas dataframes.
- **Required:**
  - [**PDA**] Ch. 9 Plotting and Visualization
  - [**PDSH**] Ch. 4. Visualization with Matplotlib
  - seaborn tutorial
- **Recommended:**
  - Watch: Matplotlib Tutorial (playlist)
  - Watch: Seaborn tutorial

★ **Assignment 5** due on Sunday, November 2.

Week 10 (Nov 6): Statistical Modeling

—No class on Tuesday (Election Day)—

- **Summary:** Machine learning is one of Python's strong suits. It is a very big topic, so we will only be able to get started. This week, we will introduce the popular scikit-learn library, which is primarily used for "traditional" machine learning on datasets best represented in a tabular format using dataframes.
- **Required:**
  - [**PDA**] Introduction to Statsmodels
  - [**PDA**] Introduction to scikit-learn
  - [**PDSH**] Ch. 5. Machine Learning (until "Feature Engineering") Linear Regression in Sklearn
  - Statsmodels, Getting Started
  - Simple Linear Regression
  - Multiple Linear Regression
  - Logistic Regression in Sklearn
  - Logistic Regression in Statsmodels
  - watch pre-recorded lecture videos on Canvas → Modules
- **Recommended:**
  - Scikit-learn Crash Course
  - if you were to want to do inferential statistics with Python: Statsmodels, Getting Started

Week 11 (Nov 13): Statistical Modeling (cont'd)

—No class on Tuesday (Veterans' Day)—

- continue last week's material

★ **Assignment 6** due on Sunday, Nov 16.

Week 12 (Nov 18 and 20): Text as Data

- **Summary:** Text analysis is another area where Python has a lot of capabilities. Spacy is an opiniated, production-ready Python library that can do many text analysis tasks efficiently. The transformers library from Hugging Face provides an interface to transformers based deep learning models, many of which are for natural language processing tasks. While fitting deep neural networks is beyond the scope of is course, using pre-trained models is fairly straightforward and that is exactly what we will do!
- **Required:**
  - Spacy 101
  - Hugging Face NLP Course (Chapter 0, 1, 2)
- **Recommended:**
  - Ch. 6. Manipulating Strings, Ch. 7. Pattern Matching with Regular Expressions

Week 13 (Nov 25): Image as Data

—No class on Thursday (Thanksgiving)—

- **Summary:** We will take a similar approach here and use a few pre-trained models for image processing tasks (recognize items in an image, create a text description of an image, etc.) The pillow library is an image processing library and is useful for opening an image in Python before forwarding it to a pretrained network.
- **Required:**
  - Pillow Tutorial

★ **Assignment 7** due on Sunday, November 30.

Week 14 (December 2 and 4): Databases

- **Summary:** Using databases is useful especially when dealing with larger-than-memory datasets and has benefits like scalability and data integrity protections. SQLite is a lightweight database system built into the Python standard library. DuckDB is similar to SQLite, but it is optimized for analytical workflows and it is popular in the data science community. We will learn some basic SQL, too!
- **Required:**
  - sqlite3 module Python documentation (go through the mini tutorial at the beginning of the page)
  - Duckdb Python overview
  - Duckdb tutorial for beginners
- **Recommended:**
  - Watch: SQLite Tutorial
  - Watch: In-Process Analytical Data Management with DuckDB

★ **Assignment 8** due on Sunday, December 7.

Week 15 (December 9): Review & Catch Up

—Tuesday is the last day of classes.—

- **Summary:** This is a free (half)week for us to catch up on anything we missed and answer questions about the final project.

★ **Final Project** due on Monday, December 15. ★