W9 Project Briefing and Ideation

Today's agenda:

1. Discuss brief, and special instructions
2. Examples of Python art
3. Rest of class Ideation workshop / misc Python practice (as chosen by you)

1
Brief

The brief is essentially the same as for the last mathematical module. Ie the assessment criteria are the same → see Assessment Brief. [e.g. demonstrate your knowledge, practically, etc.]

What do you have to do? You have to build a project using Python that demonstrates your understanding of the math (and the Python) through a practical application.

But in terms of the specifics of what I would like, there are a number of differences compared to last term:

1. **The math**: we covered different math this term, focusing on functions, differentiation, and stats. It's not a lot, but it's higher level stuff.
2. **Python:** we learnt a lot about both Python as a language, but also about the different infrastructure that goes with it (Jupyter Notebook, command line, virtual environments)—all stuff that contributes to us being better coders.

→ The bottom line is then that I would like you to demonstrate your understanding of these two aspects. So pick a project that allows you to do that.

Just like last term, your **engagement with module concepts** (= one of the assessment criteria, see Assessment Brief) won't have to extend to all concepts, just a fair few.

A not exhaustive list of these concepts includes:

- What a function is
- Differentiating a function, and the interpretation of the first derivative
- Finding maxima/minima using differentiation

- Finding maxima/minima using gradient descent
- Linear regression
- Python: creating custom functions and classes
- Numpy
- Matplotlib (and submodules)
- Scikit-learn (and submodules)
- Pandas
- Handling data sets
- (if you like, Seaborn, or other packages that you like)
- Using virtual environments
- Understanding the difference between command line (Terminal) and coding in a file
- Basics of command line
- Understanding what packages (modules/libraries) are in Python, and how to use them
- Understanding the difference between Jupyter Notebook and .py file (= Python script)
- Understanding how Jupyter Notebook works, and why it's useful

So, your project could be, for example:

- A creative visualization of multiple datasets
- That uses Pandas and Matplotlib to handle the data and create visualizations
- And that creates a few custom classes/functions for your particular purposes
- And which includes at least some tiny evidence that you understand mathematical functions, at least in plotting

Another example:

- A project to bring out the features of a text in multiple ways
- That uses Pandas to handle text data
- And uses additional libraries to visualize that text in different ways
- And which creatively brings in some, at least tiny, demonstration of understanding of functions.

I understand that the math this term is much more abstract and less directly applicable than last term, and it may not be possible to evidence it directly. (For example, how exactly would you build a project around finding the first derivative of a function…?) **So, for this reason, all I ask is that you incorporate evidence of mathematical understanding in the tiniest way**

possible—for example by showing that you understand what it means to plot a function (or anything) on (x,y) coordinates.

2
Examples of art that relies on functions and/or Python

- ➔ Simple Art from (mostly) functions: https://mnrzrad.github.io/post/art_from_code/
- ➔ Fractals: https://towardsdatascience.com/creating-fractals-in-python-a502e5fc2094/
- ➔ [Animated Fractals, shown just for coolness] https://matplotlib.org/matplotblog/posts/animated-fractals/
- ➔ Data art – e.g. students at Bangor
- ➔ Word art – see Datacamp Wine
- ➔ + see example works by two actual artists, Matt DesLaurier and Andreas Gysin

Further resources for inspiration

- Cool tutorial of very nice Perlin noise: https://hackernoon.com/how-to-create-digital-generative-art-with-python
- A Github page with various examples of using math to create art: https://github.com/akshaykalucha/PtythonImageArt
- ➔ This page gives an overview of Python for generative art, including a bit of history, some info on useful package (+tutorials), and some artists who use it: https://visualalchemist.in/2024/09/16/python-for-generative-art/

3
Ideation workshop (or AOB)

So, as you're thinking about what you would like to build, consider the following questions:

- What are areas of Python that appeal to me? Or of the math that we have covered?
- What are areas where I'm not really certain, or I feel my understanding is shaky? Perhaps I could use this to buff it up, or to discuss with Daniella to see if it can get clarified?
- What is it that I am excited about?

- Python is especially strong for handling data, so could there be an artistic project I would like to do that would use data of some kind?
- What kind of data is of interest to me? Text, images, location data, numbers…? About animals (penguins), flowers, weather, bird migration patterns, tram traffic info, crime statistics, sea levels, Trump's tweets (…),

Given how Python's strength is in data, and because AI is all about data, I would encourage you to think about a project that involves some form of data.

Think about these questions, take notes. Also, use this time to chat about different parts of the math/Python that you would want to clarify.

We'll spend the rest of the module doing project support, so you can use the time in class to develop the project and to get feedback.

No matter what you do, you'll need to have become comfortable with the material we covered in Python in order to do it. So make sure to use this time to start practicing for real—and to ask me for help, to clarify things or to get to grips with it.