

Week 5 – Python Basics

This week we will focus on the following:

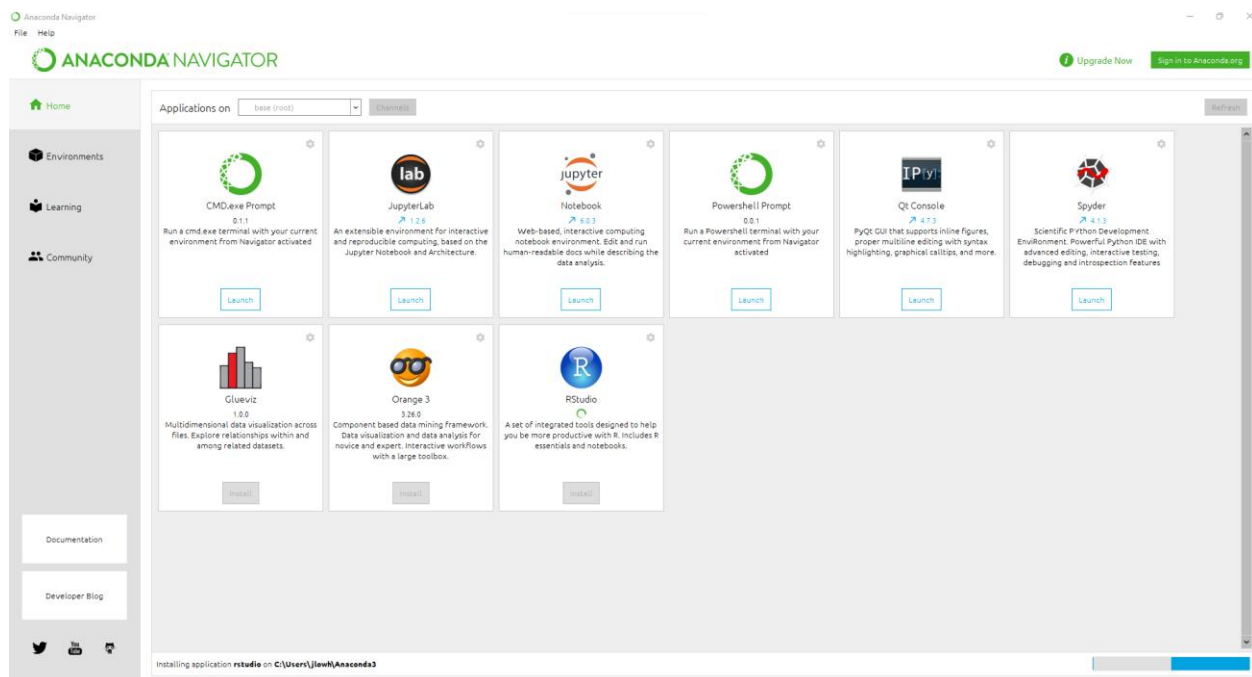
1. Installing Anaconda
2. Installing and managing libraries
3. Importing libraries, classes, and functions
4. Jupyter & Spyder
5. Python Basics

Installing Anaconda

If you installed Anaconda in week 1, congratulations, you may skip this part.

Anaconda can be installed from here: [Anaconda](#). They provide installers for Windows, MacOS, and Linux based systems so this should work for everyone.

Anaconda will ship with Jupyter and Spyder already configured. If they are not, you will need to install both from Anaconda Navigator.

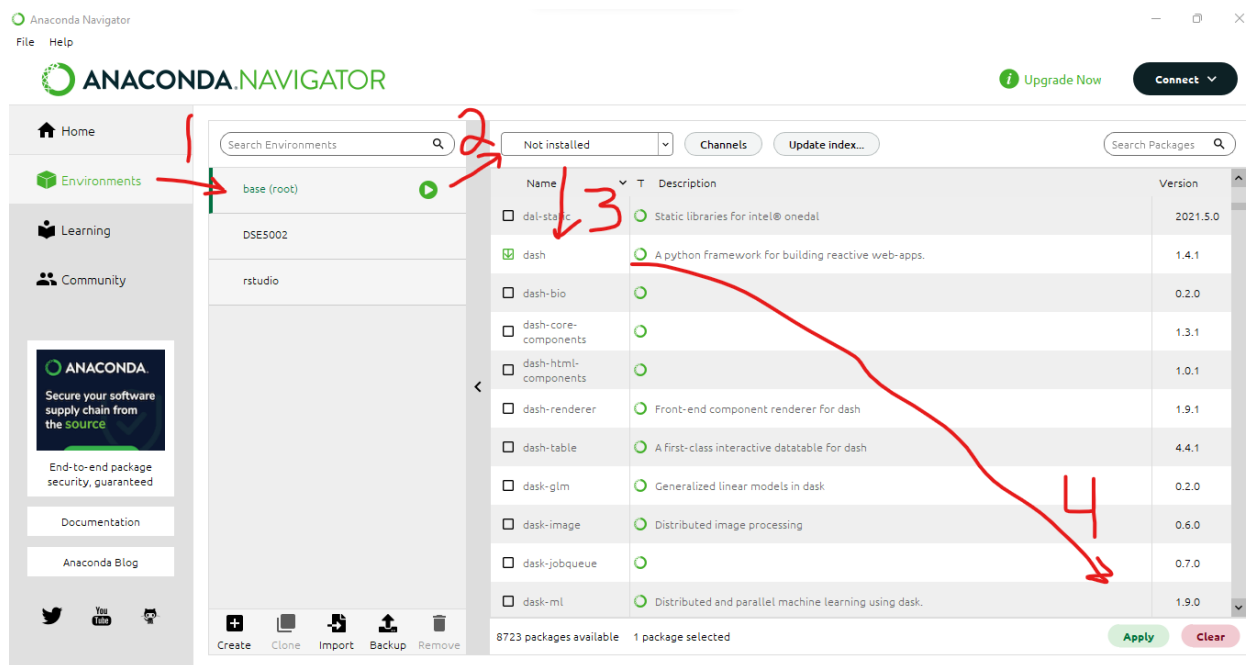


Installing and Managing Libraries

Anaconda Navigator makes it easy to install and sometimes uninstall libraries from the GUI. You may use this feature for common libraries, but if they are not available within Anaconda Navigator you will have to open the Anaconda Prompt and pip install them.

From the GUI:

1. Click on the environments tab
2. Select the environment you wish to install the library
3. Change the drop-down menu to 'Not Installed'
4. Select the library and click the 'Apply' button to install it



From Anaconda Prompt

1. Determine which environment you wish to install the library on.

- a. `conda info --envs`

```
(base) C:\Users\jlowh>conda info --envs
# conda environments:
#
base                * C:\Users\jlowh\anaconda3
DSE5002              C:\Users\jlowh\anaconda3\envs\DSE5002
rstudio              C:\Users\jlowh\anaconda3\envs\rstudio
```

2. Activate the environment if it is not already selected

- a. `conda activate <ENV>`

```
(base) C:\Users\jlowh>conda activate DSE5002  
(DSE5002) C:\Users\jlowh>
```

3. Pip install the library

- a. pip install <library>

```
(DSE5002) C:\Users\jlowh>pip install dash  
Collecting dash  
  Downloading dash-2.5.1-py3-none-any.whl (9.8 MB)  
    | 9.8 MB 939 kB/s  
Collecting dash-core-components==2.0.0  
  Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)  
Collecting Flask>=1.0.4  
  Downloading Flask-2.1.2-py3-none-any.whl (95 kB)  
    | 95 kB 1.3 MB/s  
Installing collected packages: zipp, Werkzeug, itsdangerous, importlib-metadata, click,  
s, dash-core-components, dash  
Successfully installed Flask-2.1.2 Werkzeug-2.1.2 brotli-1.0.9 click-8.1.3 dash-2.5.1 da  
ess-1.12 importlib-metadata-4.11.4 itsdangerous-2.1.2 plotly-5.8.2 tenacity-8.0.1 zipp-3  
(DSE5002) C:\Users\jlowh>
```

Where do your libraries get installed and why is that important?

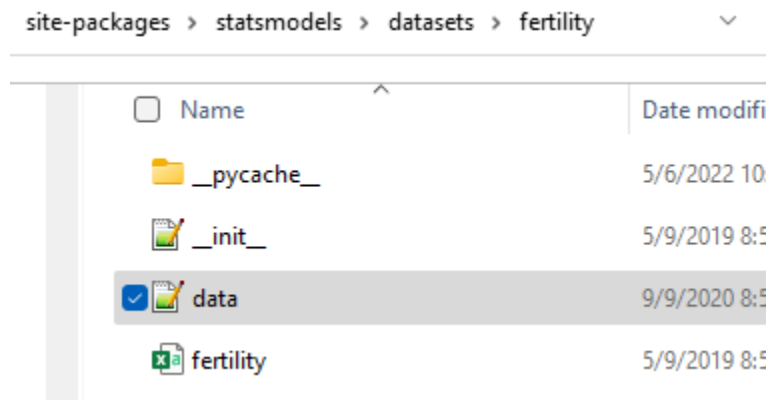
1. In Anaconda prompt type: where anaconda

This will point you to the anaconda executable and give you an idea on where the libraries are installed. You are looking for 'site-packages'

```
(base) C:\Users\jlowh\anaconda3\pkgs>where anaconda  
C:\Users\jlowh\anaconda3\Scripts\anaconda.exe
```

2. Mine happened to be in: C:\Users\jlowh\anaconda3\Lib\site-packages
3. Navigate to Statsmodels, datasets, then to the fertility folder

This folder contains an Excel CSV file with a data.py file that contains methods to read this data into python.



4. Open the `data.py` file with notepad or notepad++. I use notepad++ which is free to download. Notice that the `data.py` file contains several functions, one of which is `load_pandas`. We want to use this function to load the fertility data into python. To do so we will need to import it. Python allows you to import an entire library or specific functions within files. For this class we will always be importing **ONLY THE FUNCTIONS WE USE**.
5. To import functions, we need to use the **'from'** function to specify the path to the file containing our function and the **'import'** function to import the specific function we want to use. Python already knows the path to the library so we will simply need to specify the path to the file containing our function, remember the path is: `/statsmodels/datasets/fertility/data.py` so our import becomes **from** `statsmodels.datasets.fertility.data` **import** `load_pandas`. Once the function is loaded, we can use it to bring the data into python.

```
In [16]: from statsmodels.datasets.fertility.data import load_pandas
...:
...: load_pandas().data
Out[16]:
```

	Country Name	Country Code	...	2012	2013
0	Aruba	ABW	...	NaN	NaN
1	Andorra	AND	...	NaN	NaN
2	Afghanistan	AFG	...	NaN	NaN
3	Angola	AGO	...	NaN	NaN
4	Albania	ALB	...	NaN	NaN
..
214	Yemen, Rep.	YEM	...	NaN	NaN
215	South Africa	ZAF	...	NaN	NaN
216	Congo, Dem. Rep.	COD	...	NaN	NaN

Importing Libraries Classes and Functions

We have already seen how to import functions by navigating to Python's site-packages folder. But what about entire libraries and classes? Classes are imported exactly like

functions and as I mentioned I would like you to import the specific classes/functions you are using. Libraries are a bit different so we will look on how to do that.

As mentioned in the previous example, you can navigate to site-packages to view all the available libraries currently on your system. In the next example we will be importing the numpy library which is one of the most important and commonly used libraries in python. If you want to look at the complete functionality of numpy you can read the documentation from their website: [NumPy](#).

- To import a library simply use **import** <library> **as** <alias>
- Numpy is typically aliased as np
- To use a function simply type np.<function> as shown in the example below

```
In [17]: from statsmodels.datasets.fertility.data import load_pandas
...: import numpy as np
...:
...: df = load_pandas().data

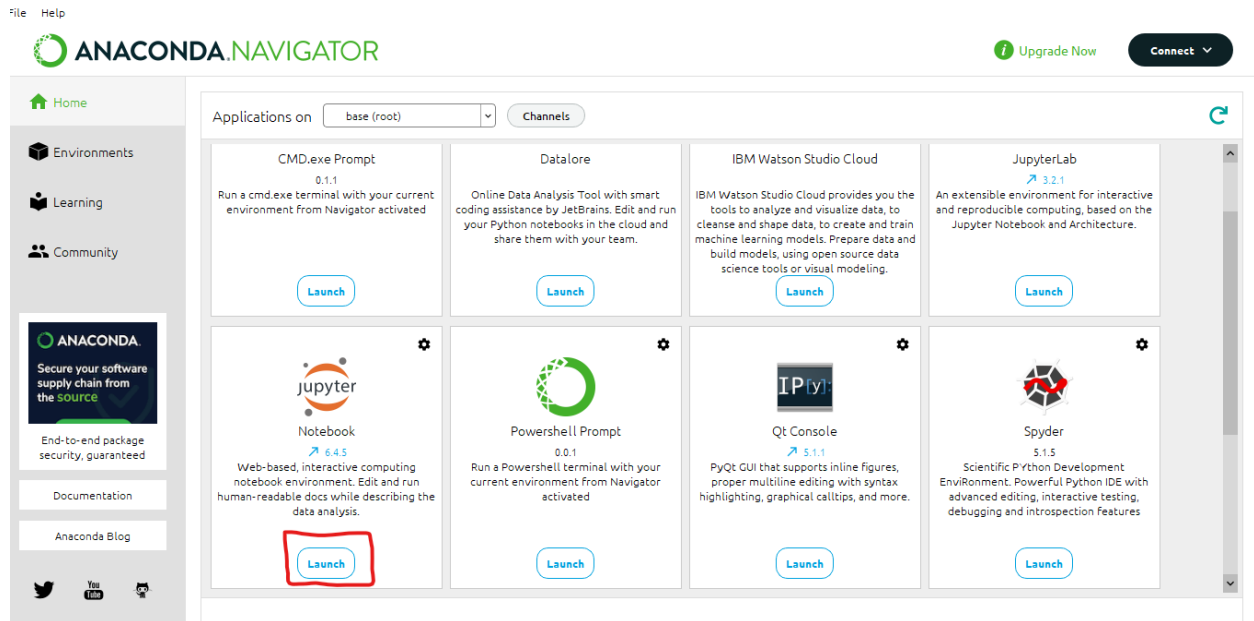
In [18]: np.mean(df[ '1963' ])
Out[18]: 5.487974093264248
```

Jupyter & Spyder

Jupyter and Spyder are the two IDEs that we will use for this class. By default, they are both installed in your base environment of Anaconda. Jupyter can be thought of as the Markdown of Python. You can write code in cells designated for code or can write markdown text. We will turn in our assignments as PDF exports of Jupyter files. Spyder is an IDE that looks and feels almost identical to RStudio. I have found it easier to transition from one language to the other using Spyder and RStudio since they are so similar.

Jupyter

To get started with Jupyter simply open Anaconda Navigator, select your environment, and launch Jupyter.

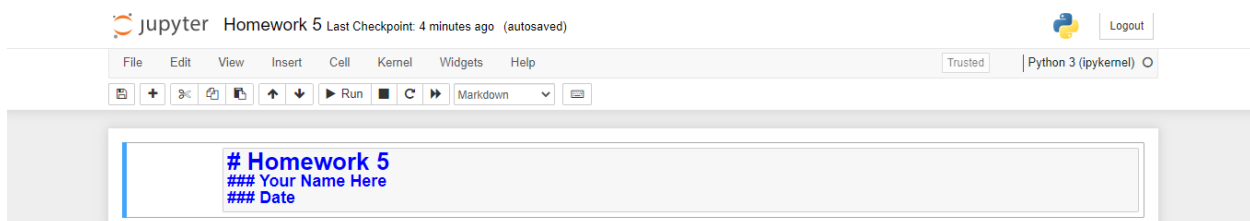


This will open a web interface in your browser of choice. To create a new notebook first navigate to your desired working directory, in this case Week 5, then select new Python3.

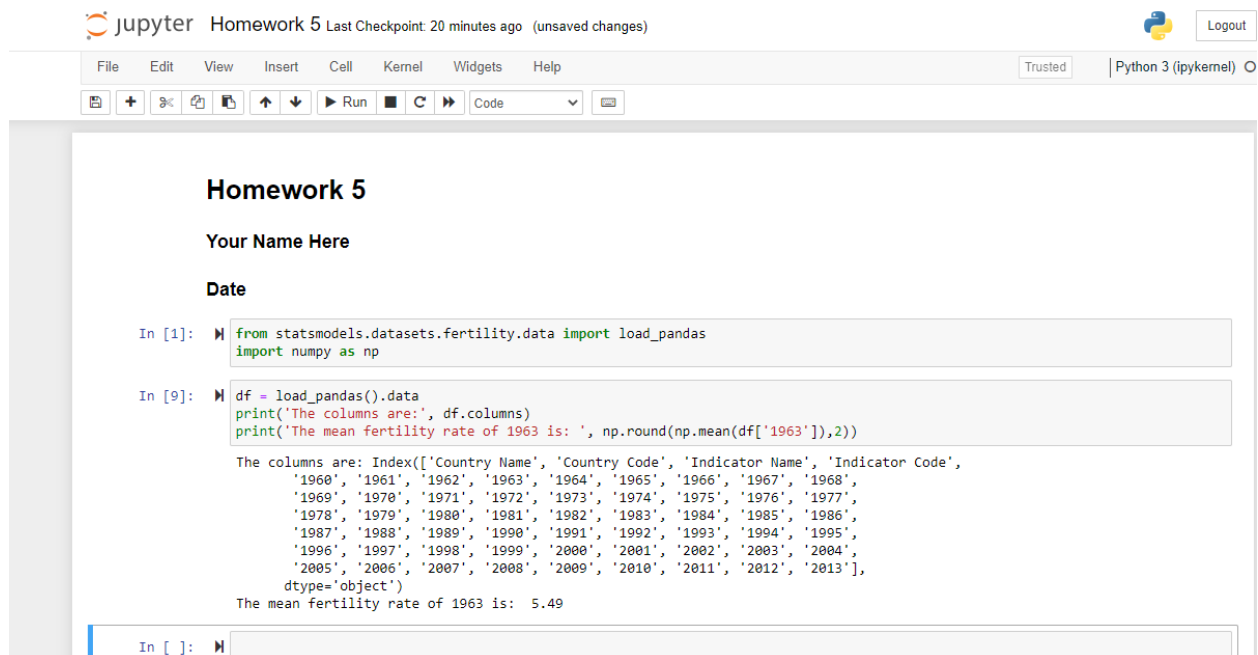


Using Jupyter

1. The first thing you should do in a new notebook is update the file name. The easiest way is to simply edit it at the top of the window. Notice I have changed mine to Homework 5.
2. After you have renamed your file, you should edit your first cell. Typically, this is a markdown cell describing the project.



3. After you have a heading configured you can hit the Run button in the middle toolbar to execute the code/markdown code within the cell.
4. The first Python code cell should contain all your imports for the notebook.
5. After that you are free to code whatever you like in any of the cells. Keep in mind a few basic rules:
 - a. Code is executed sequentially (top – down)
 - b. If you are printing multiple lines within a cell you will need to include a print statement for each line of code. Otherwise only the LAST line will be printed.
 - c. Save and save often!



The screenshot shows a Jupyter Notebook titled "Homework 5" with a last checkpoint 20 minutes ago. The notebook has a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and saving. The notebook content includes a heading "Homework 5", a placeholder "Your Name Here", and a "Date" field. Two code cells are visible. The first cell imports 'load_pandas' from 'statsmodels.datasets.fertility.data' and 'numpy' as 'np'. The second cell loads the data, prints the columns, and prints the mean fertility rate for 1963. The output of the second cell shows the columns and the mean fertility rate of 5.49.

```
In [1]: from statsmodels.datasets.fertility.data import load_pandas
import numpy as np

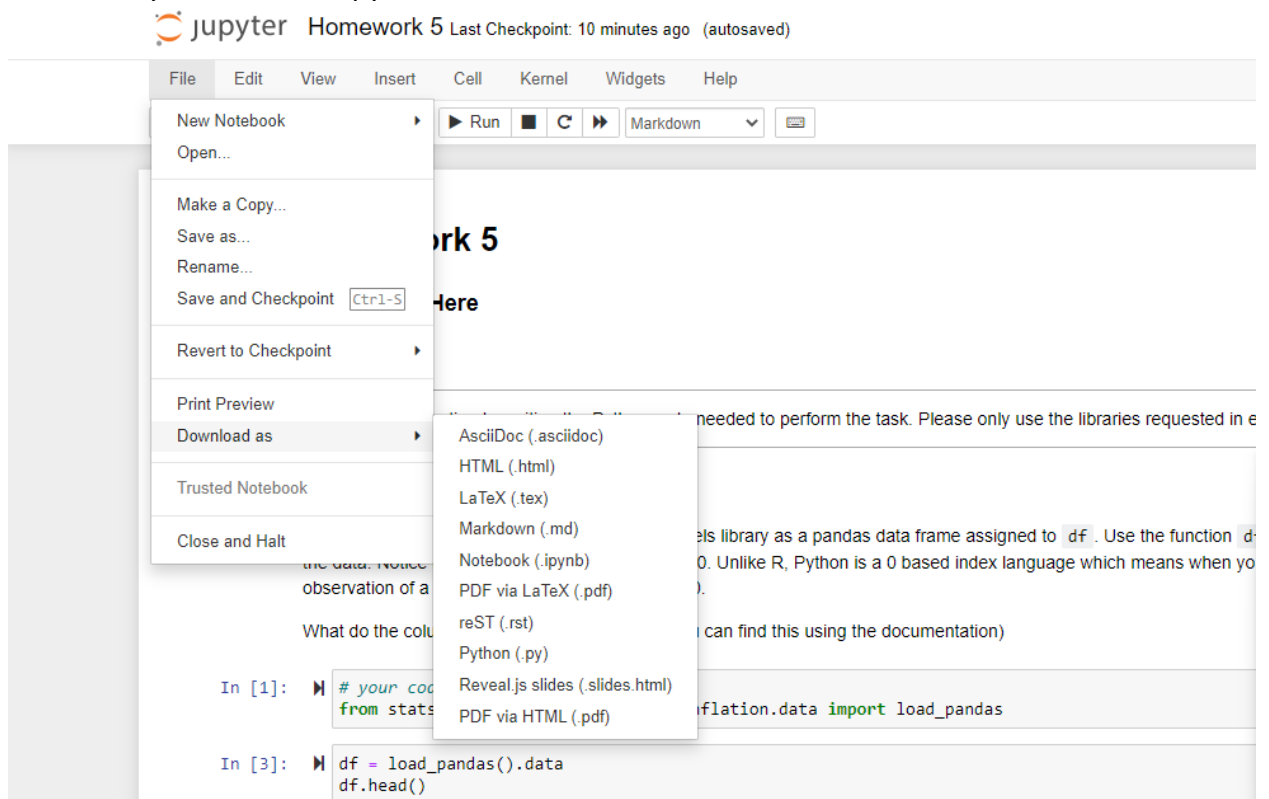
In [9]: df = load_pandas().data
print('The columns are:', df.columns)
print('The mean fertility rate of 1963 is: ', np.round(np.mean(df['1963']),2))

The columns are: Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
                        '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
                        '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
                        '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
                        '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
                        '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
                        '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013'],
                        dtype='object')
The mean fertility rate of 1963 is: 5.49
```

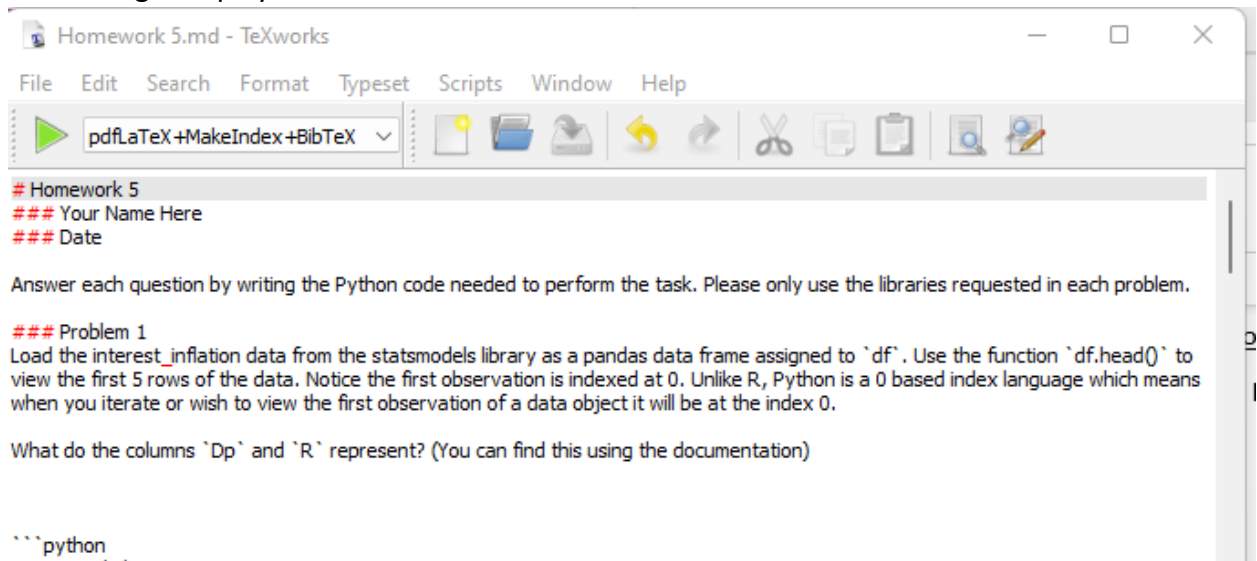
Knitting Jupyter Notebooks to PDF Documents

In week 1 we installed a complete MikTeX install which is required to perform this step.

1. Download your finished Jupyter Notebook document as TEX. File > Download As > LaTeX



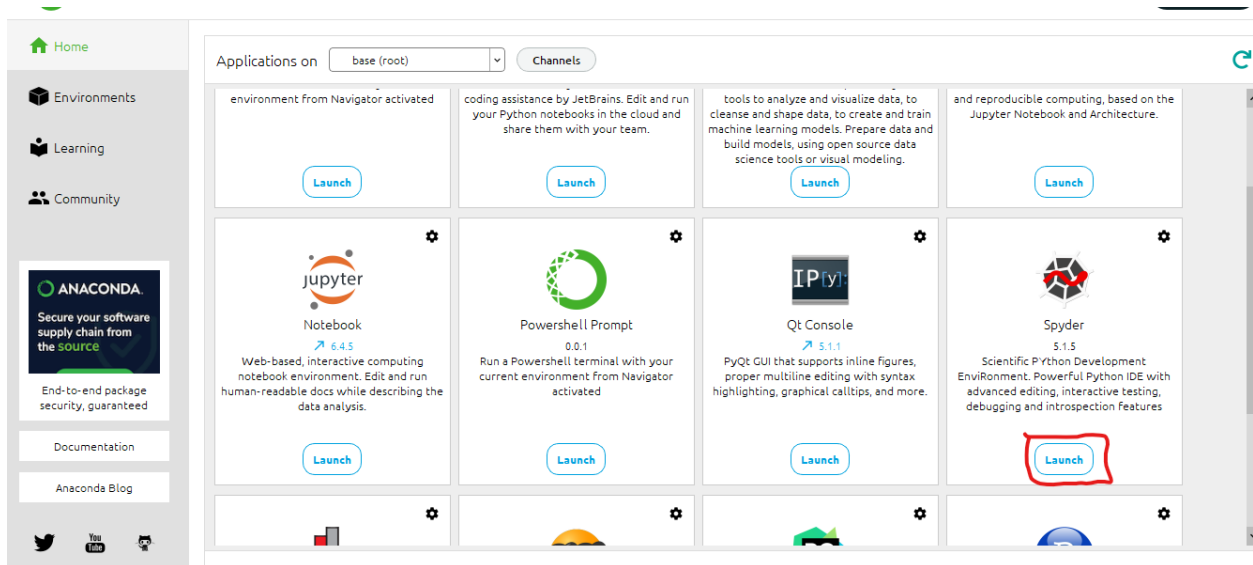
2. Open TexWorks from your computer
3. Open the TEX document that contains the code you wrote in your Jupyter Notebook.
4. Press the green play button to execute the TEX document to PDF.



Spyder

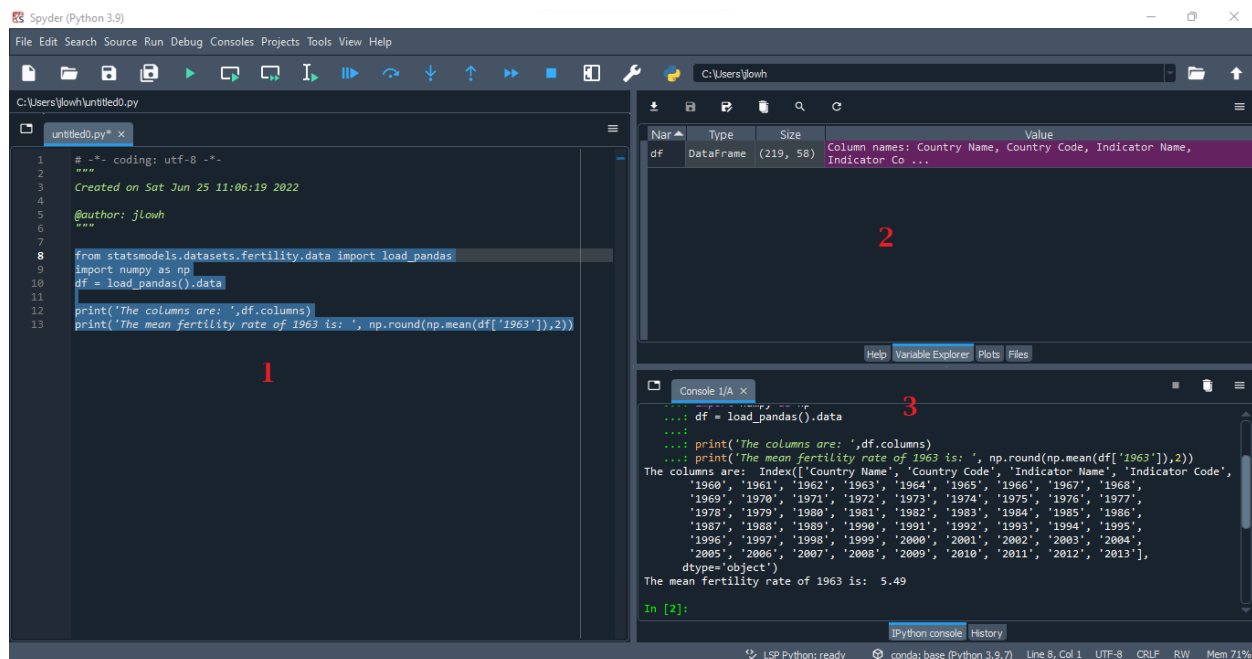
Spyder is an IDE that looks and feels much like RStudio. I have found it easier to learn and adapt to for folks transitioning to Python from R. We will not be using it directly in this class, but you may use it to aid in your Jupyter projects.

To start Spyder simply click launch from Anaconda Navigator.



Using Spyder

1. Spyder, like RStudio is setup into panels with Spyder having 3 primary panels. The left most panel is reserved for scripting while the right two panels are used to explore the file, find help with functions, explore environmental variables, and the Python console itself.



- Panel 1 – Script editor. You can create as many scripts as you want within the file menu. Be sure to save and save often!
 - Panel 2 – Help, variable explorer, plots, and file browser. You can search for help on specific functions, view the variables within your current Python environment, view any plots you have produced, or browse your file system which starts in your current working directory.
 - Panel 3 – Console. The console is where your code is executed. You can write code here directly or execute from a script you have written as shown in the example.
- To run code its best to highlight what you want from your script and use the button that looks like an I with a small play button as a subscript.

Python Basics

Operators

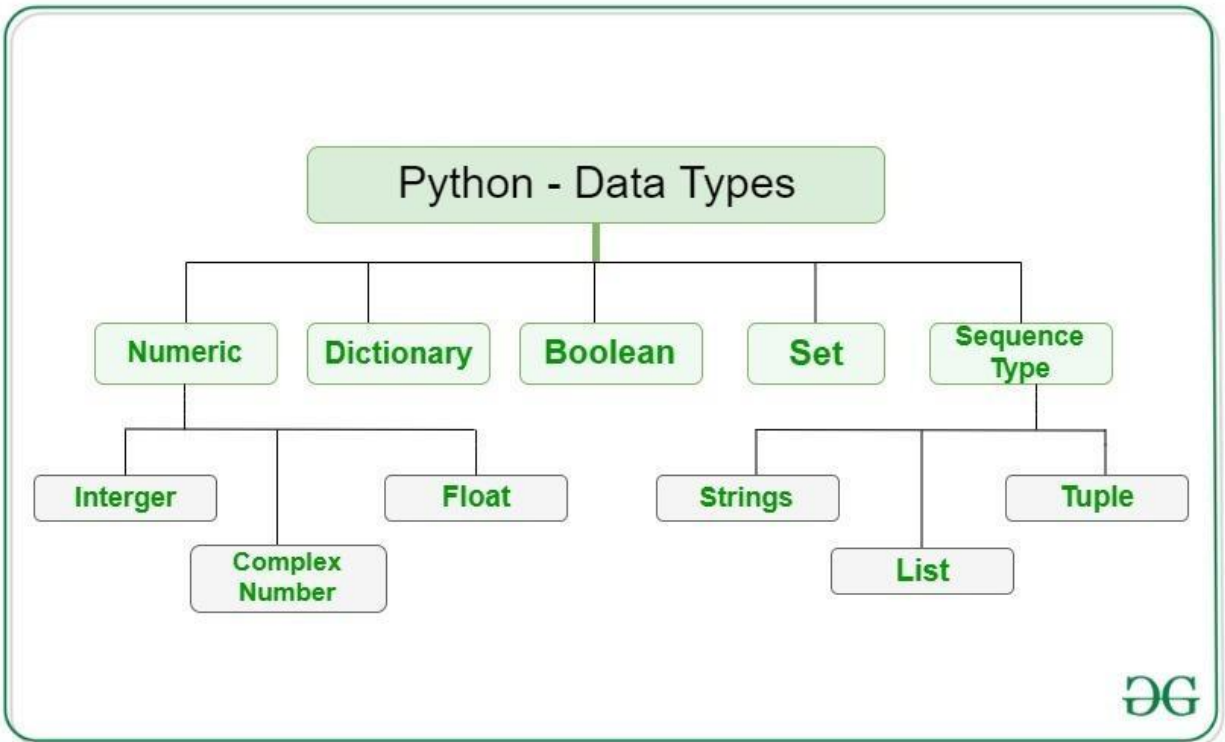
Operator	Description	Syntax
+	Addition: adds two operands	$x + y$
-	Subtraction: subtracts two operands	$x - y$

Operator	Description	Syntax
*	Multiplication: multiplies two operands	$x * y$
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	$x // y$
%	Modulus: returns the remainder when the first operand is divided by the second	$x \% y$
**	Power: Returns first raised to power second	$x ** y$
>	Greater than: True if the left operand is greater than the right	$x > y$
<	Less than: True if the left operand is less than the right	$x < y$
==	Equal to: True if both operands are equal	$x == y$
!=	Not equal to – True if operands are not equal	$x != y$
>=	Greater than or equal to True if the left operand is greater than or equal to the right	$x >= y$
<=	Less than or equal to True if the left operand is less than or equal to the right	$x <= y$
is	x is the same as y	$x \text{ is } y$
is not	x is not the same as y	$x \text{ is not } y$
and	Logical AND: True if both the operands are true	$x \text{ and } y$

Operator	Description	Syntax
or	Logical OR: True if either of the operands is true	x or y
not	Logical NOT: True if the operand is false	not x
&	Bitwise AND	x & y
	Bitwise OR	x y
~	Bitwise NOT	~x
^	Bitwise XOR	x ^ y
>>	Bitwise right shift	x>>
<<	Bitwise left shift	x<<
=	Assign value of right side of expression to left side operand	x = y + z
+=	Add AND: Add right-side operand with left side operand and then assign to left operand	a+=b a=a+b
-=	Subtract AND: Subtract right operand from left operand and then assign to left operand	a-=b a=a-b
=	Multiply AND: Multiply right operand with left operand and then assign to left operand	a=b a=a*b
/=	Divide AND: Divide left operand with right operand and then assign to left operand	a/=b a=a/b
%=	Modulus AND: Takes modulus using left and right operands and assign the result to left operand	a%=b a=a%b

Operator	Description	Syntax
//=	Divide(floor) AND: Divide left operand with right operand and then assign the value(floor) to left operand	a//=b a=a//b
=	Exponent AND: Calculate exponent(raise power) value using operands and assign value to left operand	a=b a=a**b
&=	Performs Bitwise AND on operands and assign value to left operand	a&=b a=a&b
=	Performs Bitwise OR on operands and assign value to left operand	a =b a=a b
^=	Performs Bitwise xOR on operands and assign value to left operand	a^=b a=a^b
>>=	Performs Bitwise right shift on operands and assign value to left operand	a>>=b a=a>>b
<<=	Performs Bitwise left shift on operands and assign value to left operand	a <<= b a= a << b

Python Data Types



Finding Help

It's easy to run into problems learning Python initially and can be frustrating at times. There are several ways in which you can find answers to your questions as you're learning the language. If you encounter an error, you can copy the error message, paste it in Google, and typically find answers very quickly from Stack Overflow or other websites. If you need to find documentation for basic Operations [GeeksforGeeks](https://www.geeksforgeeks.org/) is an excellent source to help you in your journey.