Population Data Science with Python

Yaser Khorrami John Karuitha

September 8, 2023

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

1 Introduction to Python for Data Science

1.1 Background

In this section, we delve into the basics of Python for Data Science. Python is a simple yet powerful programming language that has utility in web development, scientific computing, data science and machine learning. For a start, there are two versions of Python; Python version 2 and Python version 3. In this course, we work exclusively with Python version 3. Moreover, our interest in this section is the use of Python for data analysis. Let us first install Python.

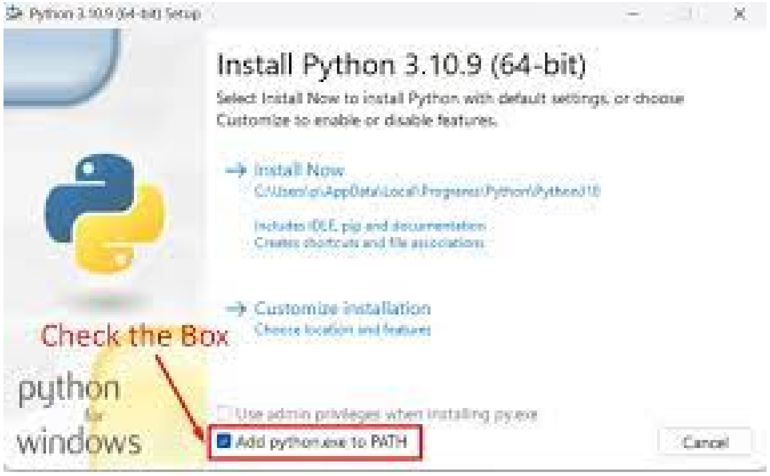
1.2 Installing Python

The installation of Python will differ slightly depending on the operating system; Windows, Mac, and Linux. The site https://www.python.org/downloads/ contains the Python executables for each operating system. ASt the time of writing this book, the Python version release is Python 3.11.5. However, installation procedures do not change much. The internet is full of tutorials on the installation of Python. In this book, we refer the reader to the available installation guidelines.

1.2.1 Installing Python on Windows

Microsoft has a comprehensive set of installation procedures for installing Python on Windows available on this website https://learn.microsoft.com/en-us/windows/python/beginners. Microsoft recommends the installation of Python from the Microsoft Store. We also recommend this approach because it will save you from the complications of setting the Python path. The link also contains information about the installation of VS Code, a popular text editor for writing Python code. We recommend that you also install VS Code.

If you choose to download and install Python directly from the Python Website, ensure that you set the path correctly. Specifically, when installing Python, ensure that you tick the choice `Add Python to Path` in the installation dialogue box (See Figure 1.1).



Python 3.10.9 (64-bit) Setup

Install Python 3.10.9 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ Install Now
C:\Users\[illegible]\AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☐ Use admin privileges when installing py.exe

☑ Add python.exe to PATH

Figure 1.1: Add Python to Path

1.2.2 Installing Python on Mac OS

We refer the reader to the following website https://www.makeuseof.com/how-to-install-python-on-mac/ for instructions on installing Python on Mac OS. We spewcifically point you to the section titled “How to Install Python With the Official Installer” as it offers a simpler and direct way to install Python on Mac OS. We also recomend that the readers install VS Code by following instructions on this site https://code.visualstudio.com/docs/setup/mac.

1.2.3 Installing Python on Linux

Most linux distributions come with linux pre-installed. For instance, Ubuntu comes with the latest Linux 3 release installed. To check the version of Python on Linux, open the terminal and run the following command.

```
python3 --version
```

To install VS Code, follow the instructions on this link https://code.visualstudio.com/docs/setup/linux.

1.3 Popular Python Text Editors and Interactive Development Environments (IDEs).

There are numerous popular IDEs and text editors for use with Python. The most popular IDE is `pycharm`. Pycharm comes in two flavors, the professional edition and the community edition.

The community edition has reduced functionality compared to the professional edition.

The most popular text editor for Python is `VS Code`. VS Code is free to download and use. This is our editor of choice iin this book. Our choice of VS Code is out of our personal preference. You can follow the contents of this book while using other platforms like Sublime text, Jupyter notebooks, among others.

1.4 Setting up VS Code for Python Programming

VS Code is a text editor. To make VS Code work with Python (and other programming languages), we need to install appropriate VS Code extensions. In our case, we install the following VS Code extensions.

- Python.
- Jupyter
- Code Runner.
- Quarto
- Prettier.

Let us illustrate how to install the Python extension.

- First, open the Extensions view (Ctrl+Shift+X).
- Filter the extension list by typing 'python'.
- Click on the Python extension (Verify that it the extensions is created by Microsoft).
- Finally, Install the extension (See Figure 2 and 3 below).

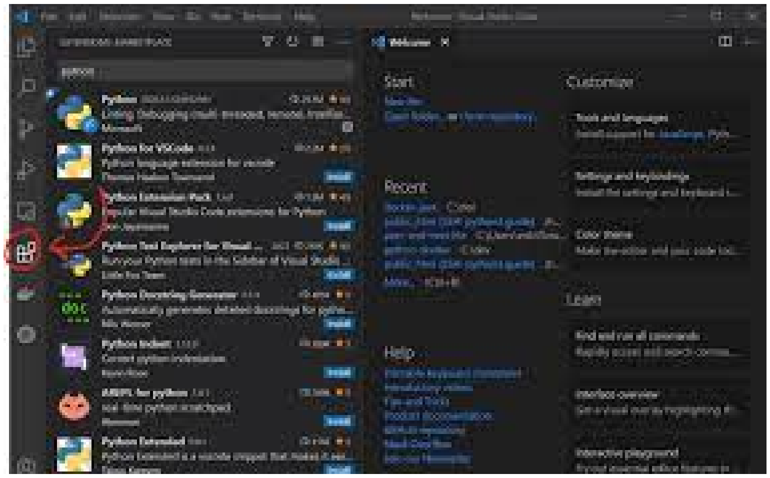


Figure 1.2: Open the extensions panel

You can follow the same procedure to install the other extensions.

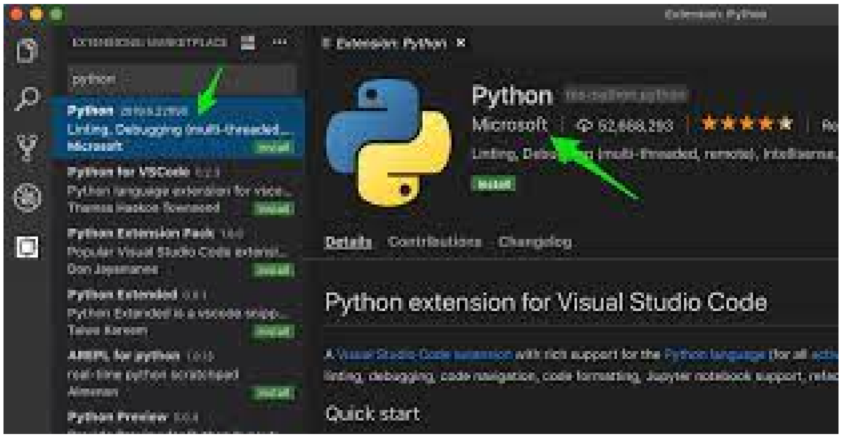

Extension: Python
EXTENSIONS: MARKETPLACE
Extension: Python
python
Python
Linting, Debugging (multi-threaded...
Microsoft
Install
Python for VSCode
Python language extension for vsco...
Install
Python Extension Pack
Popular Visual Studio Code extensi...
Don Jayamanne
Install
Python Extended
Python Extended is a vscode snipp...
Install
AREPL for python
real-time python scratchpad
Almenon
Install
Python Preview
Python
Microsoft
52,668,293
Linting, Debugging (multi-threaded, remote), Intellisense,
Install
Details
Contributions
Changelog
Python extension for Visual Studio Code
A Visual Studio Code extension with rich support for the Python language (for all
linting, debugging, code navigation, code formatting, Jupyter notebook support,
Quick start

Figure 1.3: Install the Python extension

1.5 Installing Python Packages

1.6 Loading Data into Python

We shall work with data from the United Nations Population Department (UNPD) to illustrate data analysis in Python. The data consists of population and life expectancy estimates and is available in the following website: https://population.un.org/wpp/Download/Standard/MostUsed/.

The first step in analyzing data in Python is to load the standard libraries: pandas for importing files, matplotlib and seaborn for data visualization, and numpy for mathematical operations. When importing the libraries, it is common, though not neccesary to alias the packages (like pd for pandas and plt for matplotlib.pyplot). This convention makes it easy to reference the libraries when writing code. Not that you could use any other alias. However, in the Python community, pandas is usually aliased as pd. The same is the case for the other libraries.

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

We start by importing the data using pandas. Pandas has many handy functions for importing data in variuous formats. Given that our data is in Ms Excel format, we use the `pd.read_excel()` function to import the data.

The `pd.read_excel()` webpage details the numerous arguments that we could supply to the function. To keep things simple, we will just supply the file path. The data is in the first sheet of the excel workbook and has column names as the first row. Hence, we stick with the default arguments; sheet_name=0, and header=0. Note that we could also supply a list of alterantive column names to the `names` parameter. For now, we leave the names parameter to the default of none.

```
population = pd.read_excel("data/unpd_pop.xlsx", na_values="...")
```

1.7 Selecting Rows and Columns of Data

Let us select a few variables of interest from the data. In Pandas, we can directly select columns using the following syntax:

```
data[["column1", "column2"]]
```

In the above syntax, we provide Pandas with a list of columns that we desire to pick.

A popular alternative is to use the `loc` and `iloc` functions. The `loc` function selects columns by name while the `iloc` function selects by index location. The syntax above translates to the following syntax based on `loc`:

```
data[:, ["column1", "column2"]]
```

The colon (:) tells pandas that we desire all rows. We then provide a list of columns as before. For `iloc`, we would just substitute the column names with index locations (as an integer). The advantage of the `loc` and `iloc` functions is that we can filter both rows and columns. For the rest of the chapter we shall stick to the `iloc` and `iloc` functions for filtering data.

In our case, let us filter our data so that we retain all rows and a few columns.

```
pop_sample = population.loc[:, ["region_subregion_country_area", "ISO3_code", "year", "tot
```

1.8 Exploring Data in Python

The `head` method allows us to view the first 5 rows of the data table by default. In the example below, we specify that we want to display the first 3 rows instead.

```
pop_sample.head(3)
```

	region_subregion_country_area	ISO3_code	year	total_pop_jan1_000	total_pop_july1_000
0	WORLD	NaN	1950.0	2477674.732	2499322.157
1	WORLD	NaN	1951.0	2520969.582	2543130.380
2	WORLD	NaN	1952.0	2565291.179	2590270.899

We can do the same using the `tail` method to view the last few rows of the data table.

```
pop_sample.tail()
```

	region_subregion_country_area	ISO3_code	year	total_pop_jan1_000	total_pop_july1_0
20591	Wallis and Futuna Islands	WLF	2017.0	12.002	11.9
20592	Wallis and Futuna Islands	WLF	2018.0	11.870	11.8
20593	Wallis and Futuna Islands	WLF	2019.0	11.761	11.7
20594	Wallis and Futuna Islands	WLF	2020.0	11.667	11.6
20595	Wallis and Futuna Islands	WLF	2021.0	11.642	11.6

Let us look at the number of rows and columns of the data by calling the `shape` attribute.

```
pop_sample.shape
```

```
(20596, 7)
```

The `info()` method allows us to have an overview of the data incluyding the column names and data types.

```
pop_sample.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20596 entries, 0 to 20595
Data columns (total 7 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   region_subregion_country_area  20596 non-null  object 
 1   ISO3_code                      17064 non-null  object 
 2   year                           20592 non-null  float64
 3   total_pop_jan1_000             20592 non-null  float64
 4   total_pop_july1_000            20592 non-null  float64
 5   male_pop_july1_000             20520 non-null  float64
 6   female_pop_july1_000           20520 non-null  float64
dtypes: float64(5), object(2)
```

```
memory usage: 1.1+ MB
```

We can quicly get the summary statistics using the `desribe()` method. By default, the describe method will only compute summary statistics for numeric variables. However, we can alter this behavior by supplying additional arguments. We shall revisit this method later in the book.

```
pop_sample.describe()
```

	year	total_pop_jan1_000	total_pop_july1_000	male_pop_july1_000	female_pop_j
count	20592.00000	2.059200e+04	2.059200e+04	2.052000e+04	2.05
mean	1985.50000	1.580931e+05	1.593527e+05	8.041043e+04	7.95
std	20.78311	6.155942e+05	6.202094e+05	3.139020e+05	3.07
min	1950.00000	5.150000e-01	5.110000e-01	5.670000e-01	6.5
25%	1967.75000	3.951035e+02	3.970430e+02	2.039938e+02	2.05
50%	1985.50000	4.858544e+03	4.895119e+03	2.466562e+03	2.49
75%	2003.25000	2.849011e+04	2.869103e+04	1.439467e+04	1.44
max	2021.00000	7.876932e+06	7.909295e+06	3.976648e+06	3.93

1.9 Converting Data Types in Python

In our data, we see that the columns that capture population values are of the `object` type. We shall convert the values to numeric. We can do this using two different techniques;

- The `astype` method from Python.
- The `pd.to_numeric` function from Pandas.

Let us convert the total population values using the `astype` method.

```
pop_sample[["total_pop_jan1_000", "total_pop_july1_000"]] = pop_sample[["total_pop_jan1_00
```

We could have achieved the same result using the `pd.to_numeric` function from Pandas, as follows:

```
pop_sample.describe()
```

	year	total_pop_jan1_000	total_pop_july1_000	male_pop_july1_000	female_pop_j
count	20592.00000	2.059200e+04	2.059200e+04	2.052000e+04	2.05
mean	1985.50000	1.580931e+05	1.593527e+05	8.041043e+04	7.95
std	20.78311	6.155942e+05	6.202094e+05	3.139020e+05	3.07
min	1950.00000	5.150000e-01	5.110000e-01	5.670000e-01	6.55
25%	1967.75000	3.951035e+02	3.970430e+02	2.039938e+02	2.05
50%	1985.50000	4.858544e+03	4.895119e+03	2.466562e+03	2.49
75%	2003.25000	2.849011e+04	2.869103e+04	1.439467e+04	1.44
max	2021.00000	7.876932e+06	7.909295e+06	3.976648e+06	3.93

```
pop_sample.describe(include = "object")
```

	region_subregion_country_area	ISO3_code
count	20596	17064
unique	289	237
top	Australia/New Zealand	BDI
freq	144	72

```
pop_sample.nlargest(5, "total_pop_jan1_000")
```

	region_subregion_country_area	ISO3_code	year	total_pop_jan1_000	total_pop_july1_000
71	WORLD	NaN	2021.0	7876931.987	7909295.151
70	WORLD	NaN	2020.0	7804973.773	7840952.880
69	WORLD	NaN	2019.0	7724928.292	7764951.032
68	WORLD	NaN	2018.0	7642651.364	7683789.828
67	WORLD	NaN	2017.0	7556993.443	7599822.404

```
pop_sample.nsmallest(5, "total_pop_jan1_000")
```

	region_subregion_country_area	ISO3_code	year	total_pop_jan1_000	total_pop_july1_0
12819	Holy See	VAT	2021.0	0.515	0.5
12818	Holy See	VAT	2020.0	0.525	0.5
12817	Holy See	VAT	2019.0	0.531	0.5
12816	Holy See	VAT	2018.0	0.543	0.5
12815	Holy See	VAT	2017.0	0.550	0.5

2 Summary

In summary, this book has no content whatsoever.

References