00_hello_python

November 23, 2022

1 Hello, Python

1.1 Introduction to Python

Data Sciences Institute, University of Toronto

Instructor: Kaylie Lau | TA: Salaar Liaqat

November - December 2022

2 Contents:

1. Welcome!
2. Course Logistics
3. Key Texts
4. Setup
5. Course Tour

3 Welcome!

The course introduces the Python programming language starting from its basics, working up to loading, manipulating, and visualizing real datasets with popular data science libraries. It was designed by the University of Toronto's Data Science Institute for those who have a degree in something other than Computer Science/Statistics who are looking to enhance their data science skills for their career.

All course materials can be found on GitHub: https://github.com/kaylielau/dsi-python.git

3.1 Land Acknowledgement:

We wish to acknowledge this land on which the University of Toronto operates. For thousands of years it has been the traditional land of the Huron-Wendat, the Seneca, and the Mississaugas of the Credit River. Today, this meeting place is still the home to many Indigenous people from across Turtle Island and we are grateful to have the opportunity to work on this land.

4 Course Logistics

4.1 Course contacts

Instructor: Kaylie Lau (she/her)
Email: kaylie.lau@mail.utoronto.ca

TA: Salaar Liaqat
Email: s.liaqat@mail.utoronto.ca

4.2 Pre-Course Work

Prior to the first class please: 1. Create a Google account that can use Google Colab: 1. Go to https://colab.research.google.com/. In the upper left corner, click File, then New Notebook 2. Enter `!python --version` in the code cell, then hit ctrl+enter to run the cell and confirm that your Python version is 3.6 or above. * If you are having issues with the set-up, the TA will be available to help with this **Monday 28 November from 5pm-6pm**. 2. Complete the pre-course survey: https://forms.gle/rcVCTfZasarXAGQg9

4.3 Design

The course runs synchronously over Zoom. It consists of three classes a week for three weeks, or nine classes total. Classes are 6 PM - 8 PM EDT on Mondays and Thursdays, and 9 AM - 12 PM EDT on Saturdays. Being mindful of online fatigue, there will be one or two breaks during each class where students are encouraged to stretch, grab a drink and snacks, or ask any additional questions.

Tutorial sessions with a TA will also be offered over Zoom. These will take place from 5 PM - 6 PM EDT on Mondays and Thursdays, and 8:30 AM - 9 AM EDT and 12 PM - 12:30 PM EDT on Saturdays.

4.4 Schedule

Schedule is tentative and may be modified as needed. Learners will be notified of schedule changes. * **Day 1 (Monday 28 November, 6pm-8pm):** Getting Started I (Introduction; Python fundamentals) * **Day 2 (Thursday 1 December, 6pm-8pm):** Getting Started II (Python fundamentals) * **Day 3 (Saturday 3 December, 9am-noon):** Dealing with Reality (Control flow using conditionals and loops; Lists, tuples, sets, and dictionaries) * **Day 4 (Monday 5 December, 6pm-8pm):** In/Out (Modules; Working with files; Object-oriented programming) * **Day 5 (Thursday 8 December, 6pm-8pm):** Doing More with Data I (`numpy`) * **Day 6 (Saturday 10 December, 9am-noon):** Doing More with Data II (`pandas`) * **Day 7 (Monday 12 December, 6pm-8pm):** Visualizing Data (`matplotlib`; `seaborn`; `plotly`) * **Day 8 (Thursday 15 December, 6pm-8pm):** Professional skills: Industry case study - Hareem Naveed * **Day 9 (Saturday 17 December, 9am-noon):** Review and Ethics

4.5 Prerequisites

Learners are expected to know how to operate a computer and are also expected to be familiar with the parts of a data table or spreadsheet, summary statistics, and basic data visualizations. No prior programming knowledge is required.

4.6 Expectations

The course is a live-coding class. Learners are expected to follow along with the coding in their own Python notebooks. Learners should be active participants while coding and are encouraged to ask questions throughout. Although slides will be available, they should be referenced before or after class, as class will be dedicated to coding with the instructor. ### Technology requirements * Learners must have an internet connection and a computer to participate in online activities * Learners must have a Google account that can use Google Colab

4.7 Policies

- **Accessiblity:** We want to provide an accessible learning environment for all. If there is something we can do to make this course more accessible to you, please let us know.
- **Course communications:** Communications take place over email. Please include ”DSI-Python” or similar in the subject line, e.g. ”DSI-Python: pandas question”
- **Camera:** Keeping your camera on is optional.
- **Microphone:** Please keep microphones muted unless you need to speak. Please indicate your name before speaking as some Zoom configurations make it hard to tell who is talking!
- **Assessment:** There will be homework which is **not** graded, but highly reccomended, and there will be three assignments which **are** graded.

4.8 Key Texts

- Gries, P., Campbell, J., Montojo, J., & Coron, T. (2017). *Practical programming: An introduction to computer science using python 3.6.*
- Adhikari, DeNero, and Wagner, (2021). *Computational and Inferential Thinking: The Foundations of Data Science.*
- Thomas, R. (2021). *Avoiding Data Disasters.* https://www.fast.ai/2021/11/04/data-disasters/
- Boykis, V. (2019). *Neural nets are just people all the way down.* https://vicki.substack.com/p/neural-nets-are-just-people-all-the

5 Google Colab

Jupyter Notebooks are web applications that lets us combine pieces of executable code, text, images, and visualizations in a single document, or notebook. *Google Colab* is a Jupyter Notebook environment hosted on the cloud -- that is, on Google’s servers. Google Colab provides a standardized notebook environment, with common data science libraries already installed.

Sections of a notebook are called cells. Cells can be run independently of each other, and in any order.

5.1 Check your Python version

Try adding and running your first code cells.

1. Add a cell by clicking the **+ Code**.
2. Enter the following: `!python --version`.
3. Press `ctrl` + `enter` to run the cell.

You should see Python 3.6 or higher.

[ ]:

```
!python --version
```

```
Python 3.7.15
```

5.2 Adding text

Notebooks use markdown, a markup language, for formatting text. IBM publishes a Jupyter-specific markdown cheat sheet.

To add a text cell: 1. Click **+ Text** 2. Add some text. When you are done editing, press `ctrl` + `enter` to run the cell and format your text.

6 Course Tour

6.1 Getting Started

What are the core features of Python?

6.1.1 Topics

- Data types and variables
- Expressions and statements
- Functions
- Dealing with errors
- Comments and readability

6.1.2 What is Python?

Python is a general-purpose programming language first released in 1991. It has since become a popular language for data science, thanks to an enthusiastic community and a large ecosystem of code libraries and tools that make it easier to perform common tasks throughout the data science life cycle.

```
# create a variable
degrees_celsius = 25

# convert to Fahrenheit
(9 / 5) * degrees_celsius + 32

# make it a function
def c_to_f(degrees_c):
    return (9 / 5) * degrees_c + 32
```

6.2 Dealing with Reality

What tools does Python give us for working with collections of values and for writing programs that change their behaviour based on data?

6.2.1 Topics

- Logic
- Conditionals
- Iteration
- Sequences and collections of values

```
numbers = [2, 52, 18, 27, 5, 1937]

for number in numbers:
    if number % 2 == 0:
        print('Even')
    else:
        print('Odd')
```

6.3 In/Out

How can we bring in outside code and data to accomplish tasks? How can we get data out of our programs?

6.3.1 Topics

- Modules
- Working with files
- Object-oriented programming

```
import os

for f in os.listdir('folder'):
    if f.endswith('.csv'):
        # process file
```

6.4 Doing More with Data

How can we work with real data for analysis?

6.4.1 Topics

- `numpy` and `pandas`
- Loading tabular data
- Exploring data
- Wrangling data
- Reshaping and combining datasets

```
[ ]: import pandas as pd

     neighbourhoods = pd.read_csv('/content/data/
      ↪neighbourhood-profiles-2016-140-model.csv')
     neighbourhoods[['Characteristic', 'City of Toronto']].head()
```

```
[ ]:               Characteristic City of Toronto
     0          Neighbourhood Number             NaN
     1          TSNS2020 Designation             NaN
     2              Population, 2016       2,731,571
     3              Population, 2011       2,615,060
     4  Population Change 2011-2016           4.50%
```

6.5 Visualizing Data

How do we create visualizations in Python?

6.5.1 Topics

- `matplotlib`, `seaborn` and `plotly`
- Bar charts, histograms, scatterplots, line charts
- Faceting and layering

```
[ ]: neighbourhoods = (neighbourhoods
                       .query('Characteristic == "Population, 2016"')
                       .drop(columns=['_id',
                                      'Category',
                                      'Topic',
                                      'Data Source',
                                      'City of Toronto'])
                       .melt(id_vars=['Characteristic'],
                             var_name='Neighbourhood',
                             value_name='Population, 2016'))
```

```python
neighbourhoods['Population, 2016'] = neighbourhoods['Population, 2016'].str.replace(',', '')
neighbourhoods['Population, 2016'] = pd.to_numeric(neighbourhoods['Population, 2016'])
neighbourhoods = neighbourhoods.sort_values(by='Population, 2016', ascending=False)
```

[ ]:

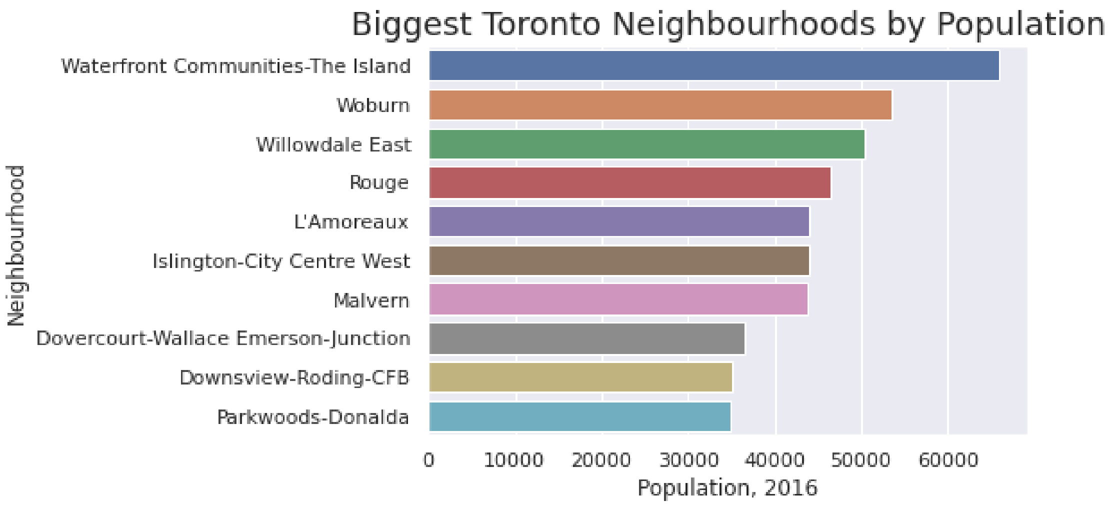
```python
import seaborn as sns

sns.set_theme()

(sns.barplot(data=neighbourhoods.head(10),
            y='Neighbourhood',
            x='Population, 2016')
    .set_title('Biggest Toronto Neighbourhoods by Population',
              fontdict={'fontsize': 18}))
```

[ ]:

```
Text(0.5, 1.0, 'Biggest Toronto Neighbourhoods by Population')
```



6.6 Do It Again

How can we make our work easier to reproduce and/or maintain?

6.6.1 Topics

- Environments
- Testing and debugging
- Moving beyond notebooks

[ ]:

```
import doctest

def c_to_f(degrees_c):
    '''
    Convert degrees Celsius to Fahrenheit
    >>> c_to_f(0)
    32.0
    >>> c_to_f(10)
    50.0
    '''
    return (9 / 5) * degrees_c + 32

doctest.testmod()
```

```
PYDEV DEBUGGER WARNING:
sys.settrace() should not be used when the debugger is being used.
This may cause the debugger to stop working correctly.
If this is needed, please check:
http://pydev.blogspot.com/2007/06/why-cant-pydev-debugger-work-with.html
to see how to restore the debug tracing back correctly.
Call Location:
  File "/usr/lib/python3.7/doctest.py", line 1487, in run
    sys.settrace(save_trace)
```

[ ]:

```
TestResults(failed=0, attempted=2)
```

6.7 Ethics

What do we even mean when we talk about ethics in tech?

6.7.1 Topics

- What does ”tech ethics” usually entail?
- Where do our data and models come from?
- What responsibility do we have for the data we collect and use?

6.8 Professional Skills

How can we prepare for a data science job interview?

6.8.1 Topics

- Technical resources

- Common questions

7 Let's get started!