

TOOL/LANGUAGE SELECTION

TOOLS/LANGUAGES

- Significant number of people who crunch numbers for a living use Microsoft Excel or other spreadsheet programs.
- Others use proprietary statistical software like SAS, Stata, or SPSS.
- While Excel and SAS are powerful tools, they have limitations. Excel cannot handle datasets above a certain size, and does not easily allow for reproducing previously conducted analysis on new datasets.
- SAS and other programs were developed for very specific uses and do not have a large community of contributors constantly adding new tools.

LANGUAGES

- The next step beyond a tool is to learn R or Python.
- These are the two most popular programming languages used by data analysts and data scientists.
- Both are free and open source, and were developed in the early 1990's – R for statistical analysis and Python as a general-purpose programming language.

PYTHON V'S R

- R has a steep learning curve as it is a low level programming language, simple procedures can take longer codes.
- Python is known for its simplicity.
- Both languages have good data handling capabilities and options for parallel computations.
- Both have advanced graphical capabilities.
- Both get the latest features quickly as they are open source.
- Python has grown in popularity.

PYTHON

- For this module it was appropriate to choose Python as the language for data analytics.
- Easier learning curve than R was a core reason.
- Reuse of Python in other roles not just data analytics a possibility.



ANACONDA

- Anaconda is one of the most popular Python Data Science Platforms available.
- It is an open source distribution.
- Quick and easy to install, run and upgrade complex data science and machine learning environments like scikit-learn, TensorFlow, and SciPy.
- Data Science IDE we will use is spyder providing editing, testing, and debugging features.

ANACONDA DOWNLOAD

- www.anaconda.com/download
- Python 3.7 version for Windows, Mac or Linux
- Installing Python in a terminal can be problematic. Many scientific packages require a specific version of Python to run, and it is difficult to keep them from interacting with each other.
- It is also hard to keep them updated.
- Anaconda distribution makes getting and maintaining these packages quick and easy.

WHAT IS ANACONDA DISTRIBUTION

- Open source, easy to install high performance Python and R distribution, with the conda package and environment manager and collection of 1,000+ open source packages with free community support.
- Included with Anaconda:
 - NumPy – n-dimensional array for numerical computation
 - SciPy – scientific computing library for Python
 - Matplotlib – 2d Plotting library for Python
 - Pandas – powerful Python data structures and analysis toolkit
 - Seaborn – statistical graphics library for Python

PYTHON

- Python is an increasingly popular tool for data analysis.
- In recent years the number of libraries have matured.
- It is a general purpose programming language.
- Python was explicitly designed so code written in Python would be easy for humans to read and to minimize the amount of time required to write code.
- Python requires a little more training to get started but there is no ceiling to what you can do with Python.
- Python has major performance advantages over most other high level languages for analysis including Matlab and R both in terms of computation speed and memory use (R is a notorious memory hog!)

PYTHON

- Things to learn in Python immediately:
 - Data types: integers, floats, strings, Booleans, lists, dictionaries, and sets
 - functions,
 - Loops
 - Mutable (int, float, string, Boolean) vs immutable data types (list, set, dic)
 - Methods to manipulate strings
 - Importing third party modules
 - Reading and interpreting errors.

PYTHON

- Python can be used by command line executing of code, executing a saved file, or interacting with ipython
- iPython (enhanced interactive Python interpreter)
- iPython is a program that sits between the user and Python itself that adds a few bells and whistles to make it a little easier to work with.
- There are many tools/interfaces that use iPython, you tell if your tool uses iPython if you see this prompt style:

```
In [1]: print('hello')  
hello
```

PYTHON

- The other method is to execute a saved file. The file has an extension .py and is run on the command line by typing `python filename.py`
- Many tutorials available to learn Python, two recommended in the lab:
 - [Codecademy.com/learn/python](https://www.codecademy.com/learn/python)
 - [Datacamp.com/learn-python-with-anaconda](https://datacamp.com/learn-python-with-anaconda)

PANDAS

- Python itself does not include vectors, matrices, or dataframes as fundamental data types.
- As Python became more popular it was realised that this was a short-coming and new libraries were created that added these data-types to Python.
- The original library that added vectors and matrices to Python was called numpy, however it had limitations. A new library was created and built on top of numpy, that added all the nice features that we expect, it is called pandas.

SPYDER

- Editor
- Inspector/Help
- Console

The screenshot displays the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. The toolbar contains icons for file operations, running, and debugging. The Editor panel shows a Python script with the following code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Spyder Editor
4
5 This is a temporary script file.
6 """
7
8 import pandas
9 import numpy
10
11 #Load dataset from the csv file in the dataframe called nesarc_data
12 nesarc_data = pandas.read_csv('nesarc_pds.csv', low_memory=False)
13
14 #convert all columns names to uppercase
15 nesarc_data.columns = map(str.upper, nesarc_data.columns)
16 #nesarc_data.columns = map(str.lower, nesarc_data.columns)
17
18 #bug fix for display formats to avoid run time errors
19 pandas.set_option('display.float_format', lambda x: '%f'%x)
20
21 print (len(nesarc_data))
22 print (len(nesarc_data.columns))
23
24 #converting strings to numeric data for better output
25
26 nesarc_data['S3AQ3B1'] = pandas.to_numeric(nesarc_data['S3AQ3B1'], errors='coerce')
27 nesarc_data['S3AQ3C1'] = pandas.to_numeric(nesarc_data['S3AQ3C1'], errors='coerce')
28
29 #Count and percentage for each variable using value_counts function
30 print('counts for TAB12MDX - nicotine dependence in the past 12 months')
31 c1= nesarc_data["TAB12MDX"].value_counts(sort=True)
32 print (c1)
33
34 print('percentages for TAB12MDX - nicotine dependence in the past 12 months')
35 p1= nesarc_data["TAB12MDX"].value_counts(sort=True, normalize=True)
36 print (p1)
37
38 print('counts for CHECK321 - smoked in the past year, yes=1')
39 c2= nesarc_data["CHECK321"].value_counts(sort=True)
40 print (c2)
41
42 print('percentages for CHECK321 - smoked in the past year, yes=1')
43 p2= nesarc_data["CHECK321"].value_counts(sort=True, normalize=True)
44 print (p2)
```

The Help panel on the right shows the "Usage" section, which explains how to get help for any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console. It also mentions that help can be shown automatically after writing a left parenthesis next to an object, and that this behavior can be activated in *Preferences > Help*. A link to the [tutorial](#) is provided.

The IPython console at the bottom shows the output of the code:

```
Name: AGE, dtype: int64
3911
1    1706
2     219
9         2
Name: CHECK321, dtype: int64
Number of observations in subset2
5838

In [6]:
```

PYTHON SYNTAX

- Python is case sensitive.
- Editing colour coded:
 - Blue font indicates a Python keyword.
 - Dark red are for numbers.
 - Green is used for strings.
 - Purple is used for a value in an option value pair.
 - Grey font indicates comments about the program.
- Comments start with a number sign `#` or a `£` symbol and they are not analysed by Python.

RESOURCES

- www.data-analysis-in-python.org
- www.datacamp.com
- www.codecademy.com