Everything is better with friends: Using SAS in Python applications with SASPy and open-source tools

SET UP SOFTWARE FOR REPLICATING EXAMPLES

Visit https://www.sas.com/en_us/software/university-edition/download-software.html, select your operating system, and follow the provided instructions to

(a) install VirtualBox,

(b) install SAS University Edition, and

(c) set up Shared Folders in VirtualBox.

Then start SAS University Edition and access JupyterLab within a web browser (see https://support.sas.com/software/products/university-edition/faq/jn_runvirtualbox.htm).

Note. You might also need to enable "virtualization technology" using the instructions available at http://support.sas.com/kb/46/250.html.

LOAD EXAMPLE FILES

Visit https://github.com/saspy-bffs/sgf-2020-tutorial, click the green button labelled **Clone or download**, and select **Download ZIP**. You should be prompted to download a file named *sgf-2020-tutorial-master.zip*.

Then unzip the file *sgf-2020-tutorial-master.zip*, and copy the resulting *.ipynb* files into the Shared Folders you created for VirtualBox.

Note. You can also manually load files into JupyterLab using these instructions: https://jupyterlab.readthedocs.io/en/stable/user/files.html#uploading-and-downloading.

EXECUTING EXAMPLES

Following https://jupyterlab.readthedocs.io/en/stable/user/files.html, load one of the example files, click inside a cell containing code, and use **Shift-Enter** to execute it.

We encourage you to follow along in the Notebooks files as you watch the tutorial on YouTube, or to work through them in your own time — whichever you prefer!

Appendix

PYTHON SYNTAX OVERVIEW FOR SAS PROGRAMMERS

Python is a versatile programming language with syntax resembling DATA steps in SAS, but with several important differences. The five most significant differences are as follows:

Capitalization is significant

These are **not** equivalent:

```
print('Hello, World!')
```

```
PRINT('Hello, World!')
```

Semicolons are only used to separate multiple statements on the same line

These are equivalent:

```
message = 'Hi!'
print(message)
```

```
message = 'Hi!'; print(message)
```

There are multiple, (mostly) interchangeable quoting styles

These are (mostly) equivalent:

```
'Hi!'
```

```
"Hi!"
```

```
'''Hi!'''
```

```
"""Hi!"""
```

[Strings surrounded by triple quotes can contain embedded line breaks.]

Assignment (=) and equality testing (==) use different operators

These are **not** equivalent:

```
fireworks = 'Yes!'
```

```
fireworks == 'Yes!'
```

White space is significant (and used to determine scope)

These are **not** equivalent:

```
if fireworks == 'Yes!':
    print('🎆')
```

```
if fireworks == 'Yes!':
print('🎆')
```

[Unindenting produces an error because the if-statement has no body.]