Python Notes

Ralph W. Crosby, PhD.

Tue Dec 22 12:07:25 PM EST 2020

1 Documentation

1.1 Using the numpy format for docstrings

1.2 See ~/Projects/Examples for a sample

1.3 Relevant urls:

- http://docutils.sourceforge.net/docs/user/rst/quickref.html
- https://github.com/numpy/numpy/blob/master/doc/HOWTO_DOCUMENT.rst.txt
- https://github.com/numpy/numpydoc/blob/master/README.rst
- https://github.com/numpy/numpy/blob/master/doc/example.py

2 Environments

2.1 IPython

- jupyter notebook –no-browser
- <2016-01-06 Wed> ipython seems to be an issue in virtual environments. Probably best to not install it in the base or, alternatively, don't install system packages in the virtualenv.
- <2016-01-19 Tue 06:58> What seems to be working is jupyter in the base and ipython installed in the virtual environments. Looks like it's getting the correct path settings.
- <2016-01-19 Tue 07:43> OK, looks like both ipython and jupyter need to be installed in the virtual environment for the path to be correct.
- <2016-03-04 Fri 14:13> So the latest:
- Installed numpy, scipy and matplotlib outside the VE
- Created VE using the command below which installs ipython and jupyter in the VE
- <2018-07-16 Mon 15:33> At this point I'm using .envrc (layout virtualeng) with standard virtual environments. Everything is installed in the VE.

2.2 My PythonModules directory - Obsolete

- Created RWCPython.pth file in site-packages pointing to ~/Projects/PythonModules
- <2018-07-16 Mon 15:35> Moved away from this. Moved setting of PYTHONPATH to .envrc files

2.3 VirtualEnvWrapper - Obsolete

- <2016-03-04 Fri 06:42> Installed on ACNTG
- <2016-03-04 Fri 14:04> Command to create an environment:
  `mkvirtualenv -r ~/PythonVirtualEnvironments/PVERequirements.lst -p /home/crosbyr/usr/bin/python3.4 --system-site-packages`
- <2019-03-16 Sat 16:17> Moved away from this, just using virtualenv and direnv

2.4 Current setup: pyenv for python versions, building customr virtualenvs

2.4.1 <2016-03-24 Thu 09:03> More screwing with environments

- Using pyenv and –system-site-packages doesn't seem to work, site-packages not set up correctly and pip isn't installing to it.
- So it's easy enough to install all the packages into the VE.
- Remember to copy rwc.pth into the site-packages directory in the VE.

2.4.2 <2016-04-06 Wed 14:16> Even more screwing with environments

- pyenv isn't working on the gpu machine, dropping back to virtualenv-wrapper

2.4.3 <2016-06-07 Tue 07:34> flymake-mode

- For some reason elpy in python decided to use flake8 for syntax checking. Installed with pip and set up a configuration file in ~/.config/flake8.
- <2019-03-16 Sat 16:19> See flake8 configuration information above

2.4.4 <2018-05-15 Tue> Got rid of all the crap and are just using virtualenv directly.

2.4.5 <2020-01-27 Mon> Implemented pyenv on the Mac just to get python versions

- Homebrew was a problem

2.4.6 <2020-01-31 Fri> Implemented pyenv on ACNT1 and GPU, recreated all virtual envs

2.4.7 <2020-12-22 Tue 11:01> Cloned pyenv git repo replacing homebrew

- Homebrew didn't stay current enough
- 2020-12-16 08:43:53 Now installed in ~/.pyenv/pyenv via git to get more current data

3 flake8

- Symlink the flake8 file to ~/.config/flake8

4 Help

4.1 pydoc

- python -m rwcdoc -p8889 starts a browser on the port
- Needed to pull a private copy of pydoc.py (rwcdoc.py) to clear the localhost so it was accessible externally.
- Also needed to open port 8889 on the firewall (see firewall_cmd for centos 7)

4.2 ipython

```
import os
from pydoc import help
os.environ['PAGER'] = 'cat'
```

Then can use help(obj) to get better help

- Not sure this is needed, help(obj) seems to work fine without importing anything.

4.3 pydoc and tensorflow

- Because of the funky way things are imported, need to be very specific on the requests, for example:
  `I[35] ~/Projects/ACNT_2016-BAR-0006/LSTM_Example_TF: import tensorflow.models.rnn as rnn`
  `I[36] ~/Projects/ACNT_2016-BAR-0006/LSTM_Example_TF: help(rnn.rnn)`

5 iPython

5.1 Cell reference

- Use _i(n) to reference a cell
- rerun (n) to rerun the cell
- %rep (n) to pull to command line

5.2 Macros

- %macro (name) (line)
- %macro (name) *line*-(line)

5.3 Persistence

- store (name) - name is optional, will store everything
- store -r (name) - name is optional, will restore everything
- save (name) lines - Creates .py file that can be executed with
- %run (name)

5.4 Saving figures

```
plt.savefig('somefile.pdf')
```

In the same cell as the figure generation

6 jupyter

- jupyter remote configuration
  - Need to open port 8888 on firewall
  - Search jupyter remote to get cookbook
- Jupyter kernels
  - Definitions are in ~/.local/share/jupyter/kernels
- Jupyter config is in ~/.jupyter/jupyter_notebook_config.py Around line 201:
  `c.NotebookApp.token=''`
  `c.NotebookApp.password=''`

7 Scripts

7.1 ShellPrompt.py

- Build a shell prompt compressing portions of the directory path

7.2 GeneratePasswords.py

- Create potential random passwords
- Random word
- Random integer
- Random special character

8 Sphinx

8.1 Numpydoc

- Need to add
  `numpydoc_show_class_members = False`
  to the conf.py file to both eliminate warnings and get next/prev page to work

9 Updating Packages

9.1 Using `pip-review`

- To list all outdated packates
  `pip-review`
- To automatically update all packates
  `pip-review -a`

10 Win10

10.1 Note: use a common directory for all work (e.g. %HOMEPATH%\Development)

10.2 Visual Studio

- Tensorflow requires the runtime
- Install VS 2017 Community Edition
  - "Desktop Development with C++"

10.3 Python.org

- Download Windows x86-64 executable installer
- Install:
  - Default location
  - install launcher for all users
  - Add Python 3.6 to path
- Start cmd.exe and setup common directory
  ```
  cd %HOMEPATH%
  mkdir Development
  cd Development
  ```
- Update pip and setuptools to latest
  `python -m pip install -U pip setuptools`
- Create virtualenv
  ```
  pip install virtualenv
  virtualenv py36
  py36\\Scripts\\activate
  ```
- Install basic packages into VE
  ```
  pip install jupyter pyyaml
  pip install tensorflow
  ```

10.4 WinPcap

- Download
  - WinPcap from winpcap.org
  - WinPcap developers's pack
- Install winpcap
- Unzip developer's pack in place (usually %HOMEPATH%/Downloads)

- Install pcapy python package
  - Start cmd.exe and python session
    - If the environment from the preceeding step is still available, use that one and skip this.
    - Activate virtual environment
      ```
      cd %HOMEPATH%\Development
      py36\Scripts\activate
      ```
  - Set visual studio environment
    ```
    "C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\Tools\vsdevcmd"
    set INCLUDE=C:\Users\crosbyr\Downloads\WpdPack_4_1_2\WpdPack\Include
    set LIB=C:\Users\crosbyr\Downloads\WpdPack_4_1_2\WpdPack\Lib\x64
    pip install pcapy
    ```

10.5 Checkout

- Start or reuse cmd.exe and python session
- Install impacket
  ```
  pip install .\impacket
  ```
- Unpack test dataset
  - Start "git bash" session from windows start menu
  - Change into the common directory (note forward slash, not backslash) and unpack data
    ```
    cd Development\antex_data
    tar -xvf ACNT_Data_Features.tar.xz
    ```
- Start jupyter notebook session
  ```
  jupyter notebook
  ```
- Open checkout notebook \antex_code\Checkout\Windows_Checkout.ipynb
- Make sure the paths are correct (see comments in the notebook)
- Run all cells in the notebook