Intro to Python

November 27, 2017

@priska

Topics

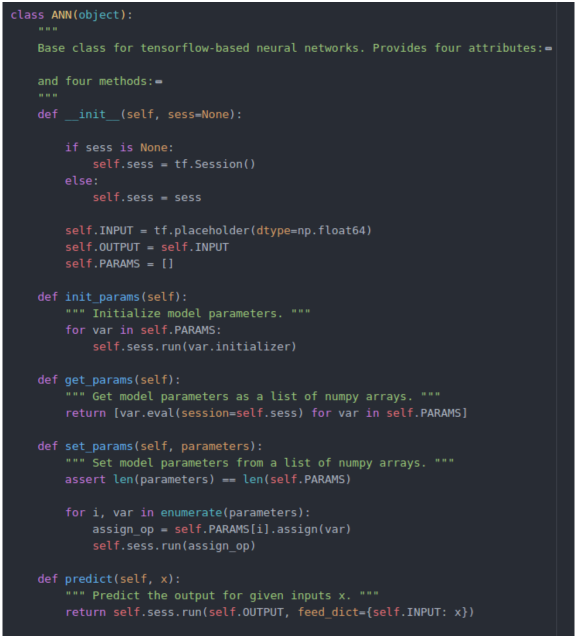
- Python in the real world
- How to get better at programming
- Classes

Python in the real world

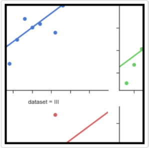
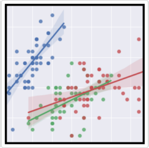
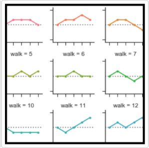
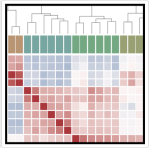
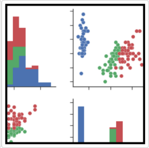
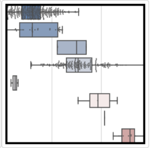
- Web development (server side) e.g., Django, Pyramid, Flask, Plone
- Data science
- Data visualization
- Desktop GUIs e.g., Tk, WxWidgets, Kivy, PyQt
- (Test) Automation
- Game development (video games and game engines)

See also famous software written in Python

```python
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response

def hello_world(request):
    return Response('Hello World!')

if __name__ == '__main__':
    with Configurator() as config:
        config.add_route('hello', '/')
        config.add_view(hello_world, route_name='hello')
        app = config.make_wsgi_app()
    server = make_server('0.0.0.0', 6543, app)
    server.serve_forever()
```



```python
class ANN(object):
    """
    Base class for tensorflow-based neural networks. Provides four attributes:

    and four methods:
    """
    def __init__(self, sess=None):

        if sess is None:
            self.sess = tf.Session()
        else:
            self.sess = sess

        self.INPUT = tf.placeholder(dtype=np.float64)
        self.OUTPUT = self.INPUT
        self.PARAMS = []

    def init_params(self):
        """ Initialize model parameters. """
        for var in self.PARAMS:
            self.sess.run(var.initializer)

    def get_params(self):
        """ Get model parameters as a list of numpy arrays. """
        return [var.eval(session=self.sess) for var in self.PARAMS]

    def set_params(self, parameters):
        """ Set model parameters from a list of numpy arrays. """
        assert len(parameters) == len(self.PARAMS)

        for i, var in enumerate(parameters):
            assign_op = self.PARAMS[i].assign(var)
            self.sess.run(assign_op)

    def predict(self, x):
        """ Predict the output for given inputs x. """
        return self.sess.run(self.OUTPUT, feed_dict={self.INPUT: x})
```


dataset = III


walk = 5
walk = 6
walk = 7
walk = 10
walk = 11
walk = 12




Python Conferences in Europe

- PyData
- EuroPython
- EuroSciPy
- PyCon DE

Watch the Python Software Foundation for more conferences and workshops.
Many conferences offer scholarships/grants you can apply for.

Meetups in Berlin

meetup.com

- Python Users Berlin
- Berlin Machine Learning group
- and many other groups

How to become a better programmer

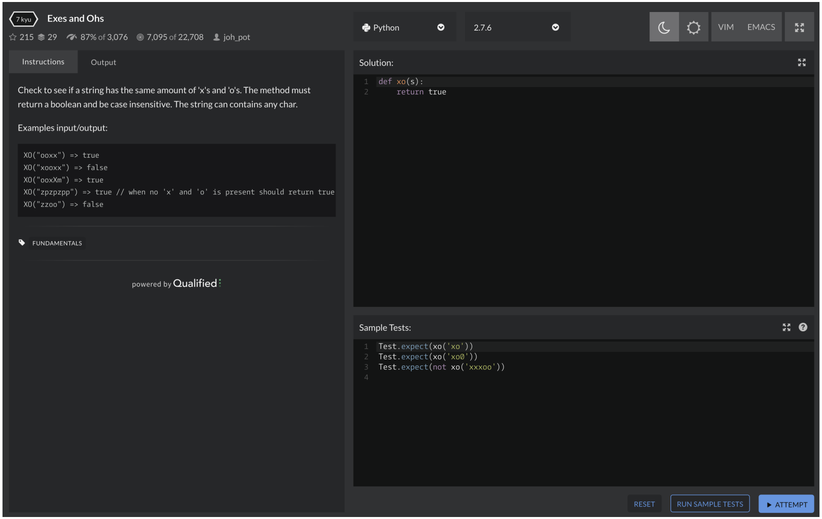
- Practice! A lot.

  Pick a task you're interested in and solve it.
  Then pick the next task/problem.
  Solve the same task more than once. Solve it in different ways.

How to become a better programmer

- Practice! A lot.
  Pick a task you're interested in and solve it.
  Then pick the next task/problem.
  Solve the same task more than once. Solve it in different ways.

- If you can't think of anything to code, try, e.g.,
  - CodeWars
  - Project Euler
  - Sphere online judge



Exes and Ohs

☆ 215 29 87% of 3,076 7,095 of 22,708 joh_pot

Check to see if a string has the same amount of 'x's and 'o's. The method must return a boolean and be case insensitive. The string can contains any char.

Examples input/output:

```
XO("ooxx") => true
XO("xooxx") => false
XO("ooxXm") => true
XO("zpzpzpp") => true // when no 'x' and 'o' is present should return true
XO("zzoo") => false
```

Solution:

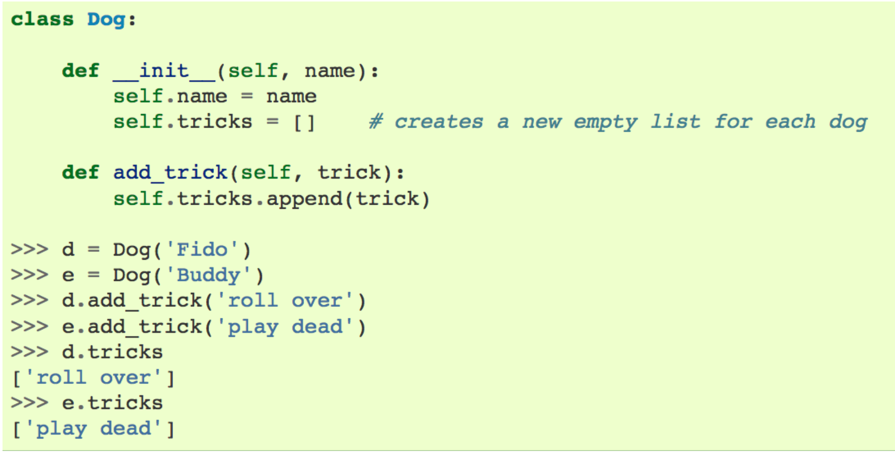
```python
def xo(s):
    return true
```

Sample Tests:

```python
Test.expect(xo('xo'))
Test.expect(xo('xo0'))
Test.expect(not xo('xxxoo'))

```

Python Classes

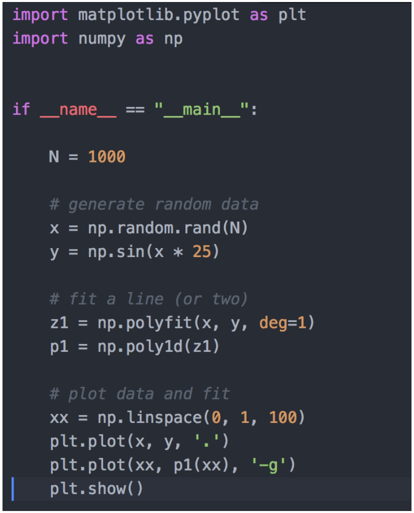
Remember?



```python
class Dog:

    def __init__(self, name):
        self.name = name
        self.tricks = []    # creates a new empty list for each dog

    def add_trick(self, trick):
        self.tricks.append(trick)

>>> d = Dog('Fido')
>>> e = Dog('Buddy')
>>> d.add_trick('roll over')
>>> e.add_trick('play dead')
>>> d.tricks
['roll over']
>>> e.tricks
['play dead']
```

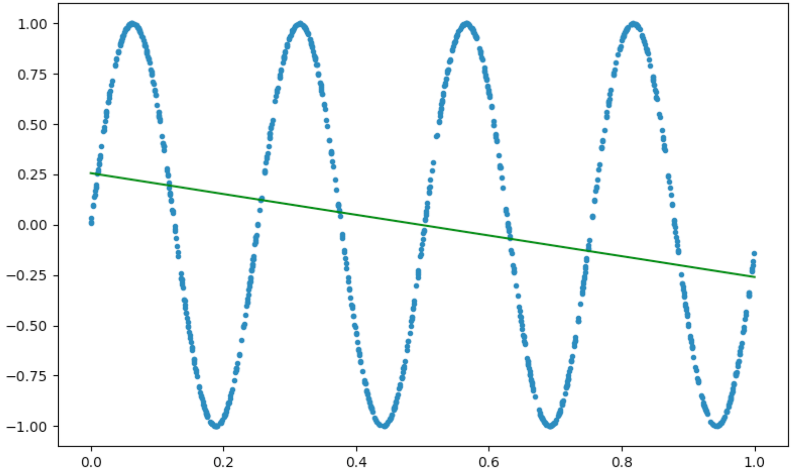
See also the Python documentation on classes

First steps with data



```
import matplotlib.pyplot as plt
import numpy as np


if __name__ == "__main__":

    N = 1000

    # generate random data
    x = np.random.rand(N)
    y = np.sin(x * 25)

    # fit a line (or two)
    z1 = np.polyfit(x, y, deg=1)
    p1 = np.poly1d(z1)

    # plot data and fit
    xx = np.linspace(0, 1, 100)
    plt.plot(x, y, '.')
    plt.plot(xx, p1(xx), '-g')
    plt.show()
```

First steps with data


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
0.2
0.4
0.6
0.8
1.0