Pengolahan Data

Ali Akbar Septiandri

Universitas Al-Azhar Indonesia

aliakbars@live.com

April 23, 2017

Overview

1. Jupyter Notebook
2. NumPy
   - Dasar
   - Komputasi dan Agregasi
   - Boolean Logic
3. SciPy

Bahan Bacaan

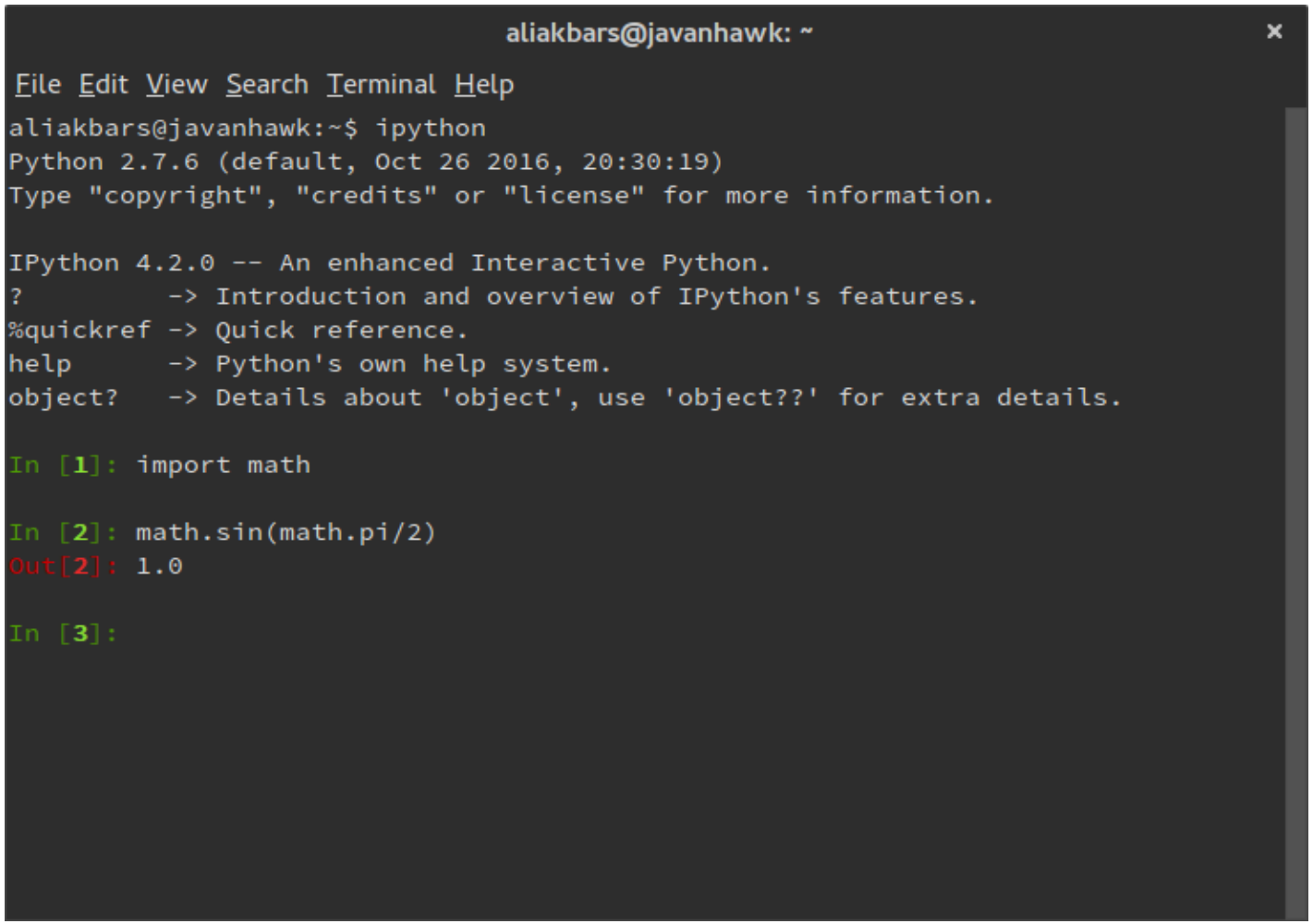
1. VanderPlas, J. (2016). Python Data Science Handbook. OReilly Media Inc.
   `https://github.com/jakevdp/PythonDataScienceHandbook`
2. Dokumentasi NumPy:
   `https://docs.scipy.org/doc/numpy/reference/`
3. Dokumentasi SciPy:
   `https://docs.scipy.org/doc/scipy/reference/`

Jupyter Notebook

Interactive Python - IPython

- Untuk kebutuhan pengolahan data, kita perlu menyimpan masukan dan keluaran dari setiap perintah yang diberikan
- Sangat membantu bila visualiasi dapat tersimpan secara "permanen"
- IPython $\rightarrow$ Jupyter Notebook

IPython



Gambar : IPython menyimpan masukan dan keluaran

Keunggulan IPython

- Profiling & timing
- Magic commands
- IPython & shell commands

Jupyter Notebook

Gambar : IPython dengan *widgets* interaktif

Jupyter Notebook

jupyter nbviewer

JUPYTER FAQ

XKCD plots in Matplotlib

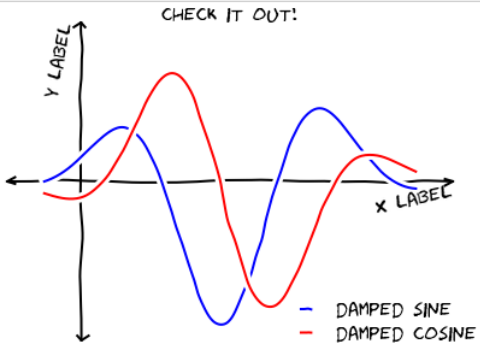
This notebook originally appeared as a blog post at Pythonic Perambulations by Jake Vanderplas.

Update: the matplotlib pull request has been merged! See This post for a description of the XKCD functionality now built-in to matplotlib!

One of the problems I've had with typical matplotlib figures is that everything in them is so precise, so perfect. For an example of what I mean, take a look at this figure:

In [1]:
```
from IPython.display import Image
Image('http://jakevdp.github.com/figures/xkcd_version.png')
```

Out[1]:



Sometimes when showing schematic plots, this is the type of figure I want to display. But drawing it by hand is a pain: I'd rather just use matplotlib. The problem is, matplotlib is a bit too precise. Attempting to duplicate this figure in matplotlib leads to something like this:

Gambar : Jupyter Notebook

NumPy

but first...

Pengolahan List

Example

```
>>> x = range(3)
>>> x * 2
?
```

Pengolahan List

Example

```
>>> x = range(3)
>>> x * 2
[0, 1, 2, 0, 1, 2]
```

Pengolahan List

Example

```
>>> x = range(3)
>>> x + 2
?
```

Pengolahan List

Example

```
>>> x = range(3)
>>> x + 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
```

Bagaimana untuk memudahkan pengolahan matriks-vektor?

Vektor dengan NumPy

Example

```
>>> import numpy as np
>>>
>>> x = np.arange(3)
>>> x
array([0, 1, 2])
>>> x * 3
array([0, 3, 6])
>>> x + 2
array([2, 3, 4])
```

Arrays

- Tidak seperti *lists*, *arrays* **harus memiliki tipe data yang sama**
- NumPy akan melakukan **upcasting**, e.g. integer $\rightarrow$ float
- Deklarasi dengan atribut `dtype`, e.g.
  `np.array([1, 2, 3, 4], dtype='float32')`

Apa yang dihasilkan saat membuat array dua dimensi?

...matrix!

Membuat Arrays

- `np.zeros()`
- `np.ones()`
- `np.full()`
- `np.arange()`
- `np.linspace()`
- `np.random.random()`
- `np.random.randint()`
- `np.eye()`

Indexing & Slicing

- Bekerja seperti pada *lists*
- Untuk multidimensi, dapat diakses dengan pemisah tanda koma, e.g. `x[:2, :3]`
- Akses baris atau kolom dengan titik-dua, e.g. `x[:, 2]`

Subarrays

- Subarrays diakses seperti membuat *view*, bukan membuat salinannya
- Gunakan metode `.copy()`

Reshaping

Example

```
grid = np.arange(1, 10).reshape((3, 3))
print(grid)
```

Concatenation

Example

```
x = np.array([1, 2, 3])
y = np.array([3, 2, 1])
np.concatenate([x, y])
```

Alternatif:

- `np.vstack()`
- `np.hstack()`
- `np.dstack()`

Splitting

Example

```
x = [1, 2, 3, 99, 99, 3, 2, 1]
x1, x2, x3 = np.split(x, [3, 5])
print(x1, x2, x3)
```

Alternatif:

- `np.vsplit()`
- `np.hsplit()`
- `np.dsplit()`

Fungsi Matematis

Beberapa fungsi NumPy juga terdapat di modul `math`, contohnya:

- `np.abs()`
- `np.sin()`, `np.cos()`, `np.tan()`
- `np.exp()`
- `np.log()`
- `np.power()`

Agregat

Beberapa fungsi yang dapat digunakan untuk melakukan agregasi data:

- `np.sum()`
- `np.prod()`
- `np.cumsum()`
- `np.cumprod()`

Fungsi Agregat

Function Name	NaN-safe Version	Description
`np.sum`	`np.nansum`	Compute sum of elements
`np.prod`	`np.nanprod`	Compute product of elements
`np.mean`	`np.nanmean`	Compute mean of elements
`np.std`	`np.nanstd`	Compute standard deviation
`np.var`	`np.nanvar`	Compute variance
`np.min`	`np.nanmin`	Find minimum value
`np.max`	`np.nanmax`	Find maximum value
`np.argmin`	`np.nanargmin`	Find index of minimum value
`np.argmax`	`np.nanargmax`	Find index of maximum value
`np.median`	`np.nanmedian`	Compute median of elements
`np.percentile`	`np.nanpercentile`	Compute rank-based statistics of elements
`np.any`	`N/A`	Evaluate whether any elements are true
`np.all`	`N/A`	Evaluate whether all elements are true

Perbandingan Nilai

- Seperti halnya operasi aritmetika, perbandingan `boolean` juga dilakukan per elemen
- Operator yang dapat digunakan mengikuti *bitwise logic operators*, i.e. &, |, ^, dan $\sim$
- e.g. `np.sum((inches > 0.5) & (inches < 1))`

Masking

Example

```
>>> x
array([[5, 0, 3, 3],
       [7, 9, 3, 5],
       [2, 4, 7, 6]])
>>> x < 5
array([[False,  True,  True,  True],
       [False, False,  True, False],
       [ True,  True, False, False]], dtype=bool)
>>> x[x < 5]
array([0, 3, 3, 3, 2, 4])
```

SciPy

Subpackages

Subpackage	Description
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions

Aljabar Linear

Beberapa hal yang dapat dilakukan dengan `scipy.linalg`:

- Mencari inverse dari matriks
- Mencari determinan
- Menghitung eigenvalues-eigenvectors
- Algoritma Dijkstra
- dll.

Statistik

Beberapa hal yang dapat dilakukan dengan `scipy.stats`:

- PDF
- CDF
- dsb.

Terima kasih