Introduction to Machine Learning Homework

Machine Learning TA

National Chiao Tung University, Hsinchu

September 27, 2017

Outline

- Introduction
- Tools
  - Python
  - MATLAB
  - Other Choices
- Scoring
- Reminds

Introduction

- There will be $3 \sim 4$ homeworks in this semester
- Homework contains mathematical proof(not every time)
- You can use your familiar programming language to solve the problems in homework
- We strongly recommend choosing languages those are widely used in scientific filed, such as Python, MATLAB, and R

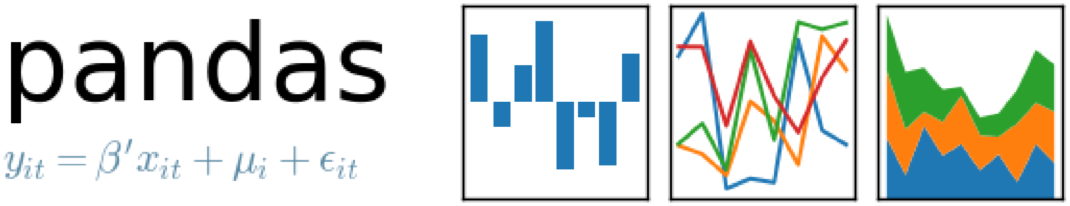
Python

- Although you can get Python from its official site, we recommend Anaconda which integrates many scientific packages for convenience
- You can use Python 2.7 or Python 3.x version

Some Useful Python Packages

- We pick up some basic part of python for beginners
- For more details, you can visit http://cs231n.github.io/python-numpy-tutorial/

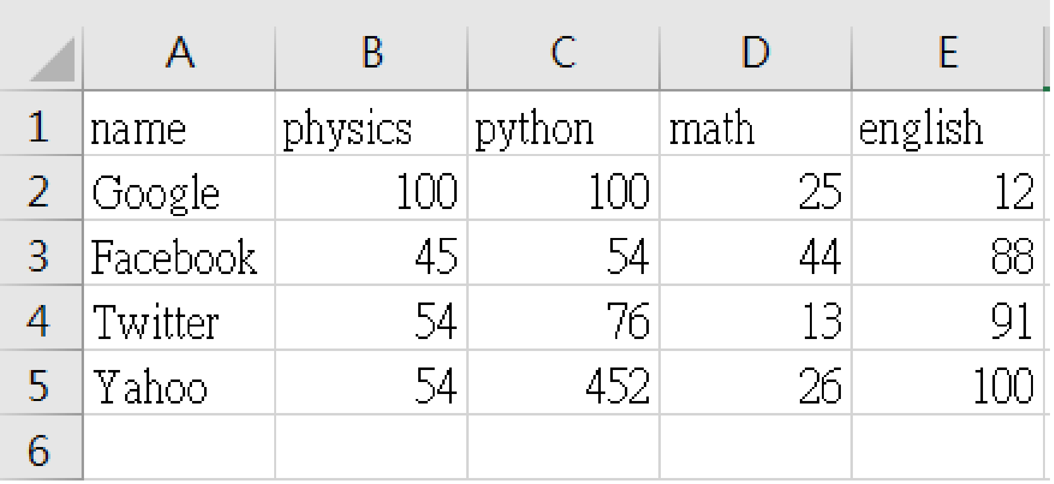
Pandas



- Load/Store formatted file, such as xlsx, txt, and csv document
- You don't need to separate/split values by yourself
- There are other packages like csv, xlrd, openpyxl can deal with formatted I/O
  You can decide what you like
- Ref: http://pandas.pydata.org/

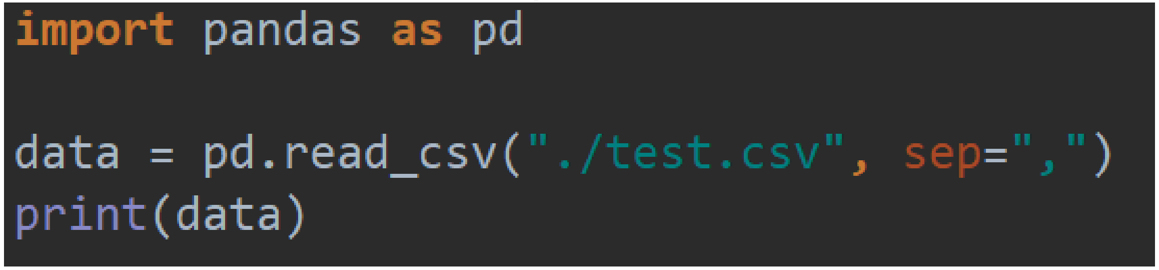
Pandas Example

input file "test.csv"



	A	B	C	D	E
1	name	physics	python	math	english
2	Google	100	100	25	12
3	Facebook	45	54	44	88
4	Twitter	54	76	13	91
5	Yahoo	54	452	26	100
6					

sample code

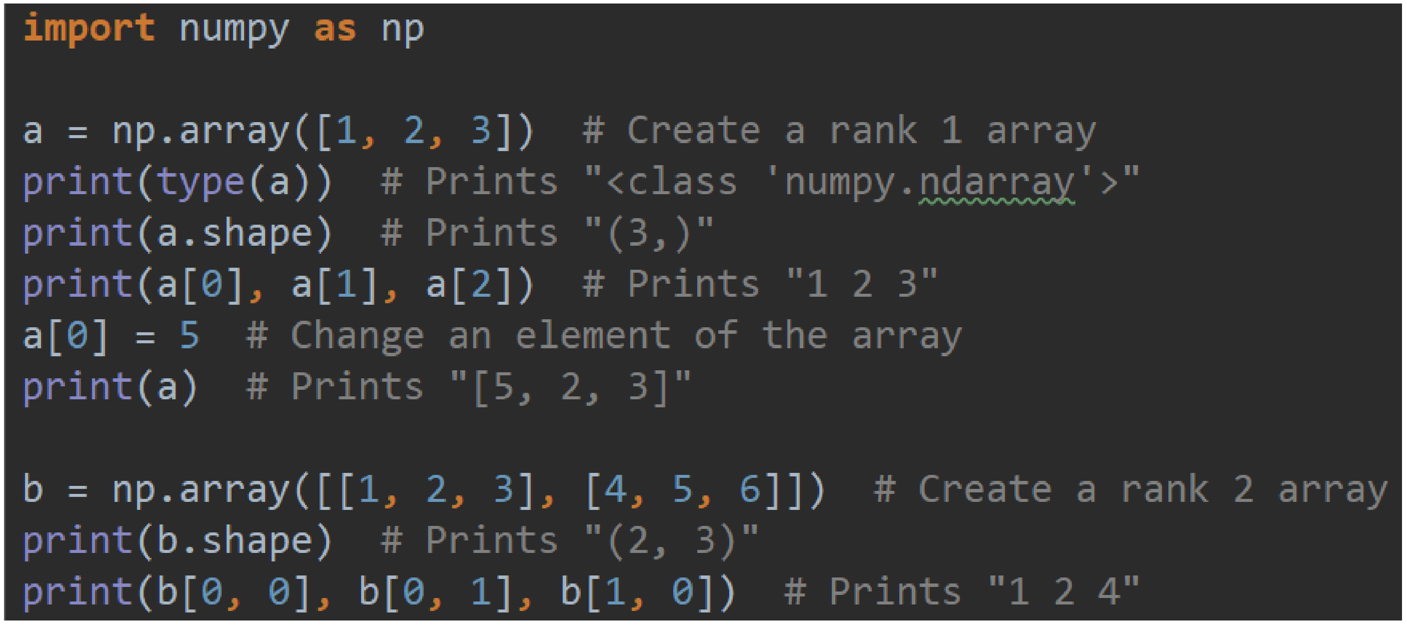


```
import pandas as pd

data = pd.read_csv("./test.csv", sep=",")
print(data)
```

Numpy

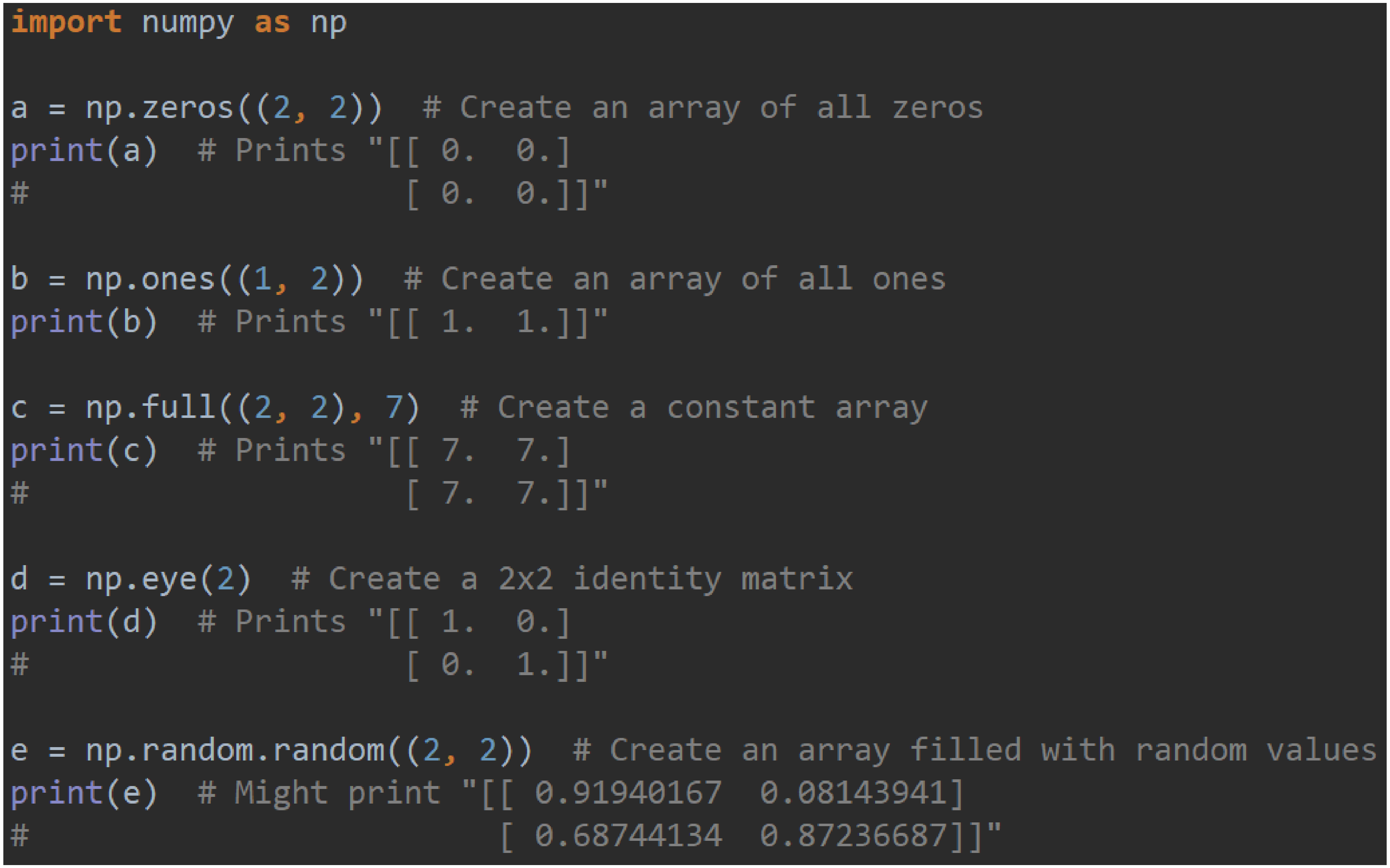
- Do vector and matrix operations
- Provide a lot of linear algebra functions
- Sometimes, training/testing data may be .npy or .npz format, you can use numpy to load
- Ref: http://www.numpy.org/

Numpy Example (1/3) Array



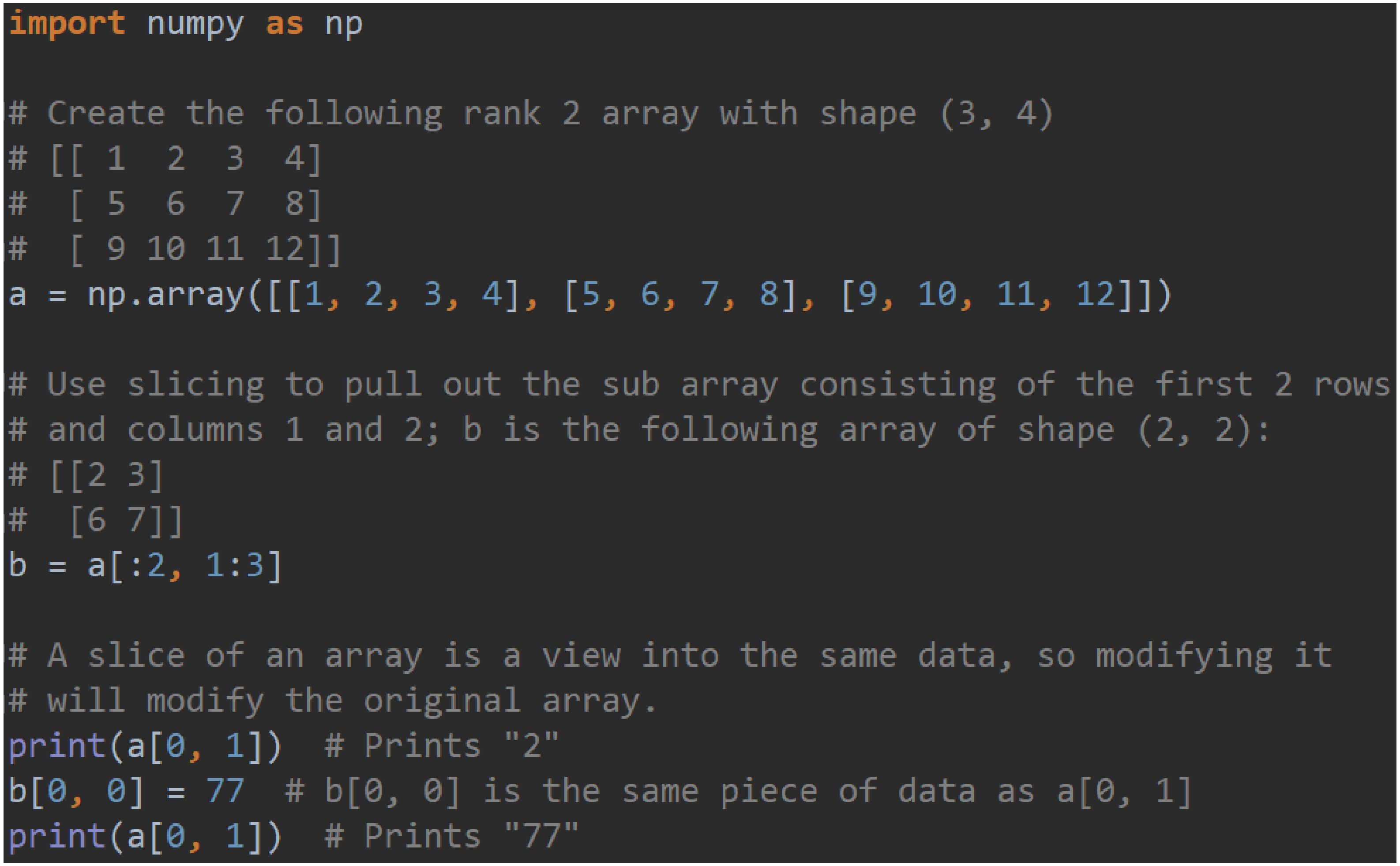
```
import numpy as np

a = np.array([1, 2, 3])  # Create a rank 1 array
print(type(a))  # Prints "<class 'numpy.ndarray'>"
print(a.shape)  # Prints "(3,)"
print(a[0], a[1], a[2])  # Prints "1 2 3"
a[0] = 5  # Change an element of the array
print(a)  # Prints "[5, 2, 3]"

b = np.array([[1, 2, 3], [4, 5, 6]])  # Create a rank 2 array
print(b.shape)  # Prints "(2, 3)"
print(b[0, 0], b[0, 1], b[1, 0])  # Prints "1 2 4"
```

Numpy Example (2/3) Array

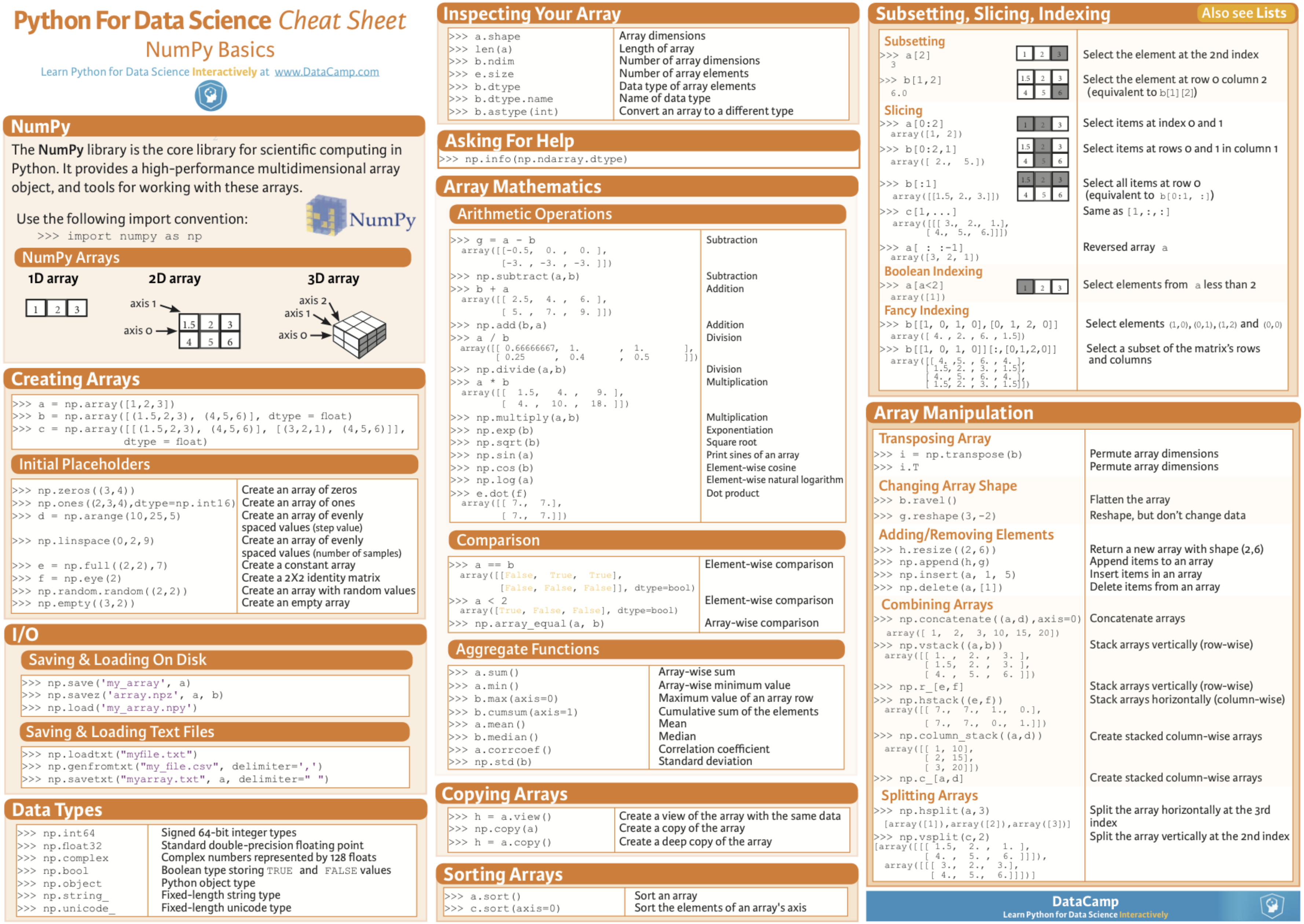


```
import numpy as np

a = np.zeros((2, 2))  # Create an array of all zeros
print(a)  # Prints "[[ 0.  0.]
#                    [ 0.  0.]]"

b = np.ones((1, 2))  # Create an array of all ones
print(b)  # Prints "[[ 1.  1.]]"

c = np.full((2, 2), 7)  # Create a constant array
print(c)  # Prints "[[ 7.  7.]
#                    [ 7.  7.]]"

d = np.eye(2)  # Create a 2x2 identity matrix
print(d)  # Prints "[[ 1.  0.]
#                    [ 0.  1.]]"

e = np.random.random((2, 2))  # Create an array filled with random values
print(e)  # Might print "[[ 0.91940167  0.08143941]
#                         [ 0.68744134  0.87236687]]"
```

Numpy Example (3/3) Array



```
import numpy as np

# Create the following rank 2 array with shape (3, 4)
# [[ 1  2  3  4]
#  [ 5  6  7  8]
#  [ 9 10 11 12]]
a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])

# Use slicing to pull out the sub array consisting of the first 2 rows
# and columns 1 and 2; b is the following array of shape (2, 2):
# [[2 3]
#  [6 7]]
b = a[:2, 1:3]

# A slice of an array is a view into the same data, so modifying it
# will modify the original array.
print(a[0, 1])  # Prints "2"
b[0, 0] = 77  # b[0, 0] is the same piece of data as a[0, 1]
print(a[0, 1])  # Prints "77"
```



Python For Data Science *Cheat Sheet*

NumPy Basics

Learn Python for Data Science **Interactively** at www.DataCamp.com

NumPy

The **NumPy** library is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Use the following import convention:

```
>>> import numpy as np
```

NumPy

NumPy Arrays

1D array

1	2	3

2D array

axis 1

axis 0

1.5	2	3
4	5	6

3D array

axis 2

axis 1

axis 0

Creating Arrays

```
>>> a = np.array([1,2,3])
>>> b = np.array([(1.5,2,3), (4,5,6)], dtype = float)
>>> c = np.array([[(1.5,2,3), (4,5,6)], [(3,2,1), (4,5,6)]],
                 dtype = float)
```

Initial Placeholders

`>>> np.zeros((3,4))`	Create an array of zeros
`>>> np.ones((2,3,4),dtype=np.int16)`	Create an array of ones
`>>> d = np.arange(10,25,5)`	Create an array of evenly spaced values (step value)
`>>> np.linspace(0,2,9)`	Create an array of evenly spaced values (number of samples)
`>>> e = np.full((2,2),7)`	Create a constant array
`>>> f = np.eye(2)`	Create a 2X2 identity matrix
`>>> np.random.random((2,2))`	Create an array with random values
`>>> np.empty((3,2))`	Create an empty array

I/O

Saving & Loading On Disk

```
>>> np.save('my_array', a)
>>> np.savez('array.npz', a, b)
>>> np.load('my_array.npy')
```

Saving & Loading Text Files

```
>>> np.loadtxt("myfile.txt")
>>> np.genfromtxt("my_file.csv", delimiter=',')
>>> np.savetxt("myarray.txt", a, delimiter=" ")
```

Data Types

`>>> np.int64`	Signed 64-bit integer types
`>>> np.float32`	Standard double-precision floating point
`>>> np.complex`	Complex numbers represented by 128 floats
`>>> np.bool`	Boolean type storing `TRUE` and `FALSE` values
`>>> np.object`	Python object type
`>>> np.string_`	Fixed-length string type
`>>> np.unicode_`	Fixed-length unicode type

Inspecting Your Array

`>>> a.shape`	Array dimensions
`>>> len(a)`	Length of array
`>>> b.ndim`	Number of array dimensions
`>>> e.size`	Number of array elements
`>>> b.dtype`	Data type of array elements
`>>> b.dtype.name`	Name of data type
`>>> b.astype(int)`	Convert an array to a different type

Asking For Help

```
>>> np.info(np.ndarray.dtype)
```

Array Mathematics

Arithmetic Operations

`>>> g = a - b`
`array([[-0.5,  0. ,  0. ],`
`       [-3. , -3. , -3. ]])`
	Subtraction

`>>> np.subtract(a,b)`	Subtraction

`>>> b + a`
`array([[ 2.5,  4. ,  6. ],`
`       [ 5. ,  7. ,  9. ]])`
	Addition

`>>> np.add(b,a)`	Addition

`>>> a / b`
`array([[ 0.66666667,  1.        ,  1.        ],`
`       [ 0.25      ,  0.4       ,  0.5       ]])`
	Division

`>>> np.divide(a,b)`	Division

`>>> a * b`
`array([[  1.5,   4. ,   9. ],`
`       [  4. ,  10. ,  18. ]])`
	Multiplication

`>>> np.multiply(a,b)`	Multiplication
`>>> np.exp(b)`	Exponentiation
`>>> np.sqrt(b)`	Square root
`>>> np.sin(a)`	Print sines of an array
`>>> np.cos(b)`	Element-wise cosine
`>>> np.log(a)`	Element-wise natural logarithm

`>>> e.dot(f)`
`array([[ 7.,  7.],`
`       [ 7.,  7.]])`
	Dot product

Comparison

`>>> a == b`
`array([[False,  True,  True],`
`       [False, False, False]], dtype=bool)`
	Element-wise comparison

`>>> a < 2`
`array([True, False, False], dtype=bool)`
	Element-wise comparison

`>>> np.array_equal(a, b)`	Array-wise comparison

Aggregate Functions

`>>> a.sum()`	Array-wise sum
`>>> a.min()`	Array-wise minimum value
`>>> b.max(axis=0)`	Maximum value of an array row
`>>> b.cumsum(axis=1)`	Cumulative sum of the elements
`>>> a.mean()`	Mean
`>>> b.median()`	Median
`>>> a.corrcoef()`	Correlation coefficient
`>>> np.std(b)`	Standard deviation

Copying Arrays

`>>> h = a.view()`	Create a view of the array with the same data
`>>> np.copy(a)`	Create a copy of the array
`>>> h = a.copy()`	Create a deep copy of the array

Sorting Arrays

`>>> a.sort()`	Sort an array
`>>> c.sort(axis=0)`	Sort the elements of an array's axis

Subsetting, Slicing, Indexing

Also see Lists

Subsetting

`>>> a[2]`
`3`
	Select the element at the 2nd index

`>>> b[1,2]`
`6.0`
	Select the element at row 0 column 2 (equivalent to `b[1][2]`)

Slicing

`>>> a[0:2]`
`array([1, 2])`
	Select items at index 0 and 1

`>>> b[0:2,1]`
`array([ 2.,  5.])`
	Select items at rows 0 and 1 in column 1

`>>> b[:1]`
`array([[1.5, 2., 3.]])`
	Select all items at row 0 (equivalent to `b[0:1, :]`)

`>>> c[1,...]`
`array([[[ 3.,  2.,  1.],`
`        [ 4.,  5.,  6.]]])`
	Same as `[1,:,:]`

`>>> a[ : :-1]`
`array([3, 2, 1])`
	Reversed array `a`

Boolean Indexing

`>>> a[a<2]`
`array([1])`
	Select elements from `a` less than 2

Fancy Indexing

`>>> b[[1, 0, 1, 0],[0, 1, 2, 0]]`
`array([ 4. , 2. , 6. , 1.5])`
	Select elements `(1,0)`,`(0,1)`,`(1,2)` and `(0,0)`

`>>> b[[1, 0, 1, 0]][:,[0,1,2,0]]`
`array([[ 4. ,5. , 6. , 4. ],`
`       [ 1.5, 2. , 3. , 1.5],`
`       [ 4. , 5. , 6. , 4. ],`
`       [ 1.5, 2. , 3. , 1.5]])`
	Select a subset of the matrix's rows and columns

Array Manipulation

Transposing Array

`>>> i = np.transpose(b)`	Permute array dimensions
`>>> i.T`	Permute array dimensions

Changing Array Shape

`>>> b.ravel()`	Flatten the array
`>>> g.reshape(3,-2)`	Reshape, but don't change data

Adding/Removing Elements

`>>> h.resize((2,6))`	Return a new array with shape (2,6)
`>>> np.append(h,g)`	Append items to an array
`>>> np.insert(a, 1, 5)`	Insert items in an array
`>>> np.delete(a,[1])`	Delete items from an array

Combining Arrays

`>>> np.concatenate((a,d),axis=0)`
`array([ 1,  2,  3, 10, 15, 20])`
	Concatenate arrays

`>>> np.vstack((a,b))`
`array([[ 1. ,  2. ,  3. ],`
`       [ 1.5,  2. ,  3. ],`
`       [ 4. ,  5. ,  6. ]])`
	Stack arrays vertically (row-wise)

`>>> np.r_[e,f]`	Stack arrays vertically (row-wise)

`>>> np.hstack((e,f))`
`array([[ 7.,  7.,  1.,  0.],`
`       [ 7.,  7.,  0.,  1.]])`
	Stack arrays horizontally (column-wise)

`>>> np.column_stack((a,d))`
`array([[ 1, 10],`
`       [ 2, 15],`
`       [ 3, 20]])`
	Create stacked column-wise arrays

`>>> np.c_[a,d]`	Create stacked column-wise arrays

Splitting Arrays

`>>> np.hsplit(a,3)`
`[array([1]),array([2]),array([3])]`
	Split the array horizontally at the 3rd index

`>>> np.vsplit(c,2)`
`[array([[[ 1.5,  2. ,  1. ],`
`        [ 4. ,  5. ,  6. ]]]),`
` array([[[ 3.,  2.,  3.],`
`        [ 4.,  5.,  6.]]])]`
	Split the array vertically at the 2nd index

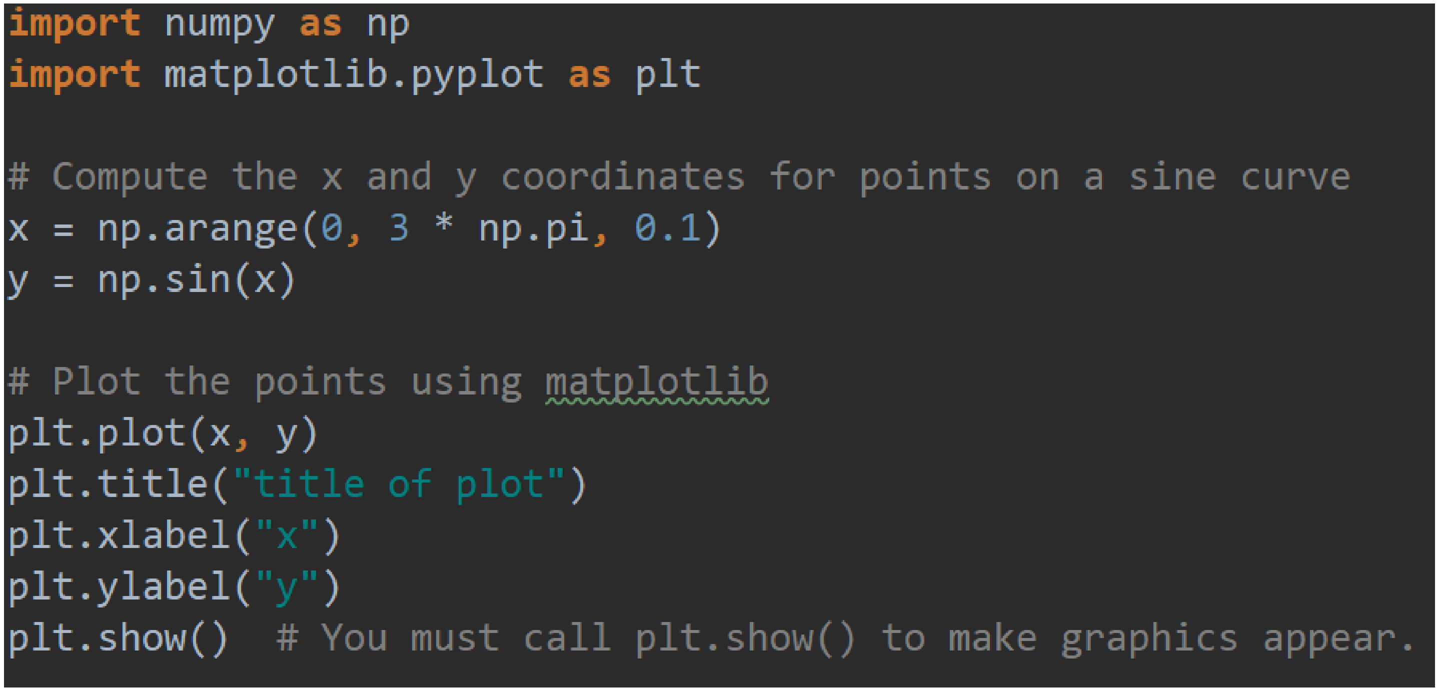
DataCamp

Learn Python for Data Science **Interactively**

Matplotlib

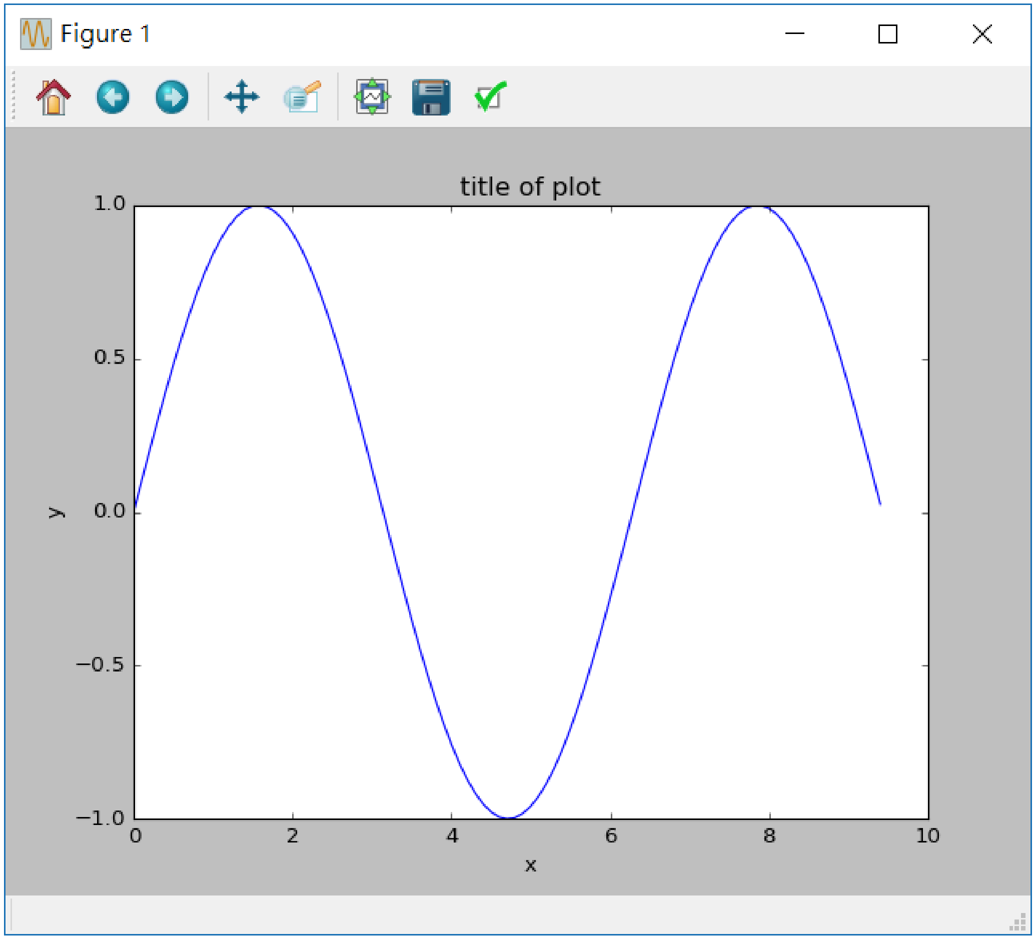
- Usually, you need to plot learning curve or other figures in the report
- Matplotlib is a powerful 2D plotting library
- Ref: https://matplotlib.org/

Matplotlib Example (1/4)



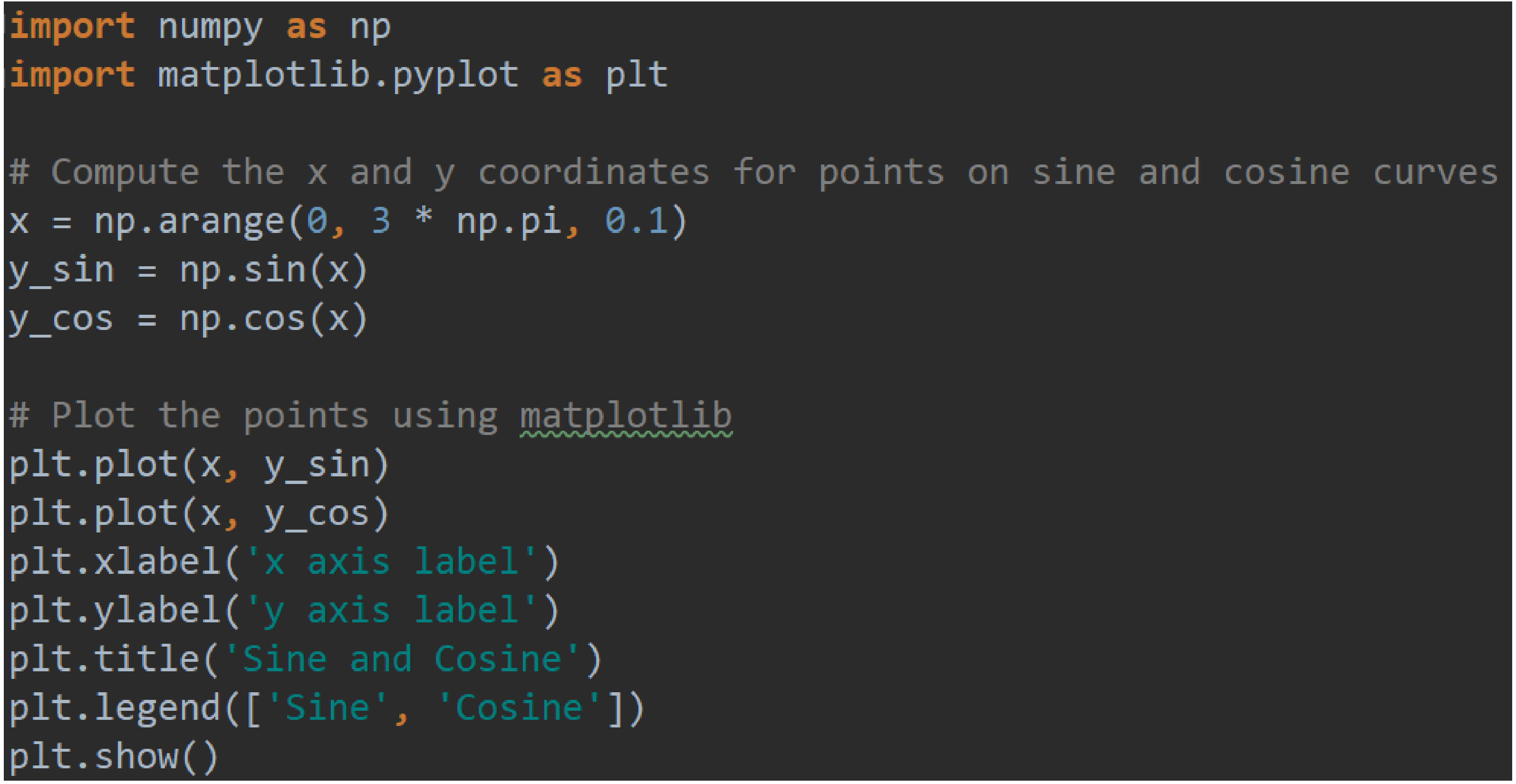
```python
import numpy as np
import matplotlib.pyplot as plt

# Compute the x and y coordinates for points on a sine curve
x = np.arange(0, 3 * np.pi, 0.1)
y = np.sin(x)

# Plot the points using matplotlib
plt.plot(x, y)
plt.title("title of plot")
plt.xlabel("x")
plt.ylabel("y")
plt.show()  # You must call plt.show() to make graphics appear.
```

Matplotlib Example (2/4)



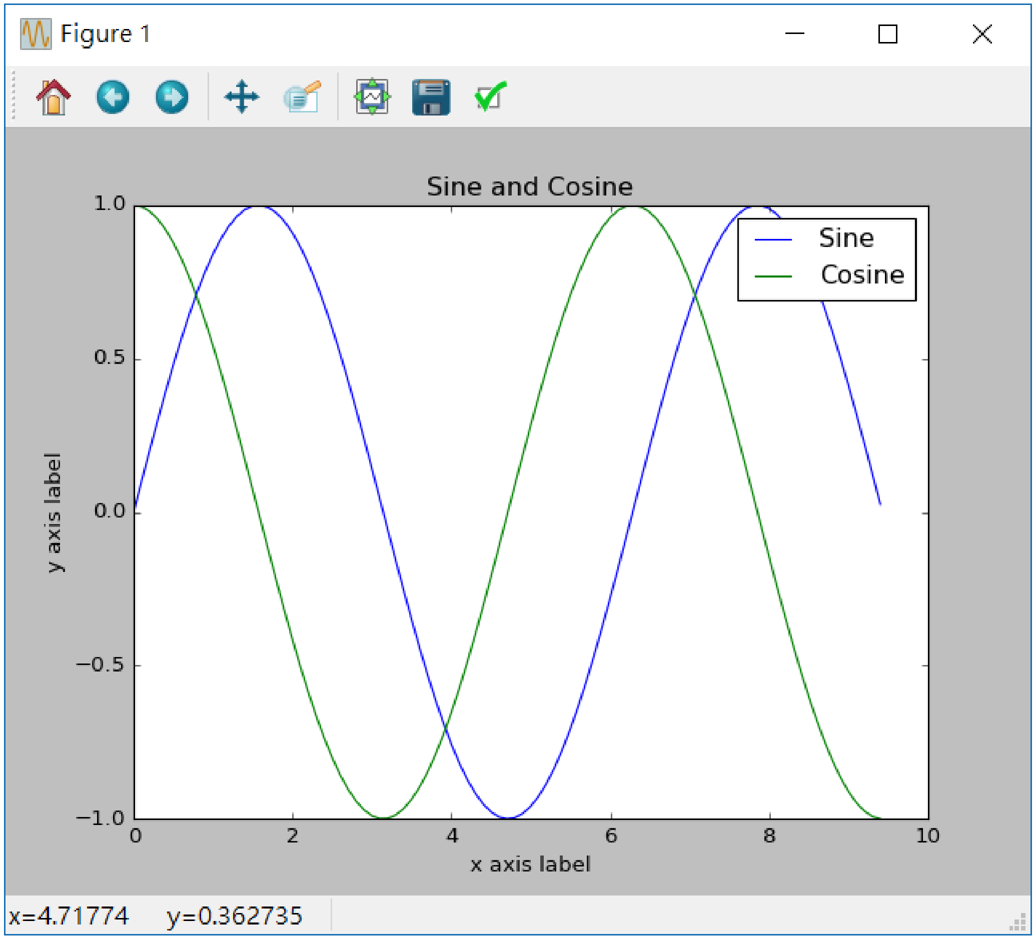
Matplotlib Example (3/4)

You can plot several curves on a figure with different color

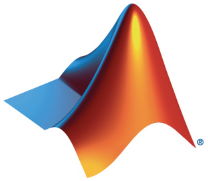


```
import numpy as np
import matplotlib.pyplot as plt

# Compute the x and y coordinates for points on sine and cosine curves
x = np.arange(0, 3 * np.pi, 0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)

# Plot the points using matplotlib
plt.plot(x, y_sin)
plt.plot(x, y_cos)
plt.xlabel('x axis label')
plt.ylabel('y axis label')
plt.title('Sine and Cosine')
plt.legend(['Sine', 'Cosine'])
plt.show()
```

Matplotlib Example (4/4)



MATLAB



NCTU has MATLAB license, you can find information from

NCTU MATLAB License

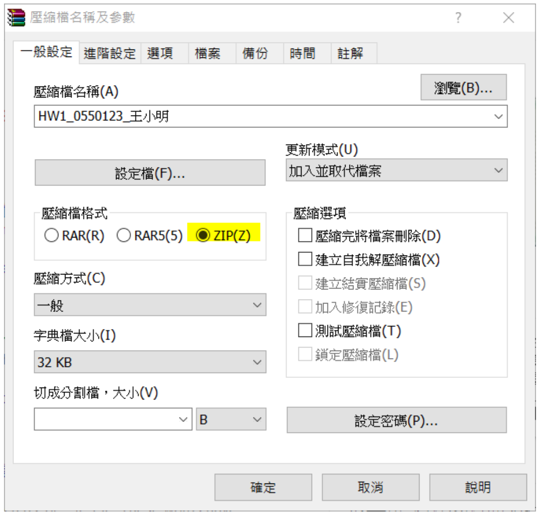
Other Choices

or other language you like...

Scoring

- Deadline: 2 weeks since the homework is announced
- If you fail to hand in homework in time, we can open another link for you to upload. But penalty/discount will reflect on the score
- Do not plagiarize other's work
- (optional) You can write down a ReadMe file to describe your develop environment(operating system, programming language) or completeness. This can help us evaluate and give a score quickly

Reminds



壓縮檔名稱(A): HW1_0550123_王小明

更新模式(U): 加入並取代檔案

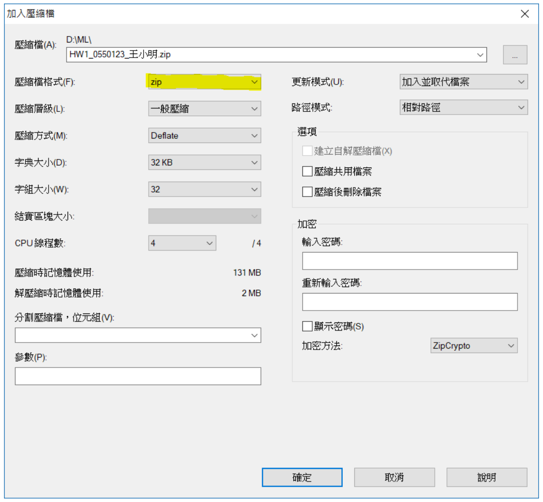
壓縮檔格式: ○ RAR(R) ○ RAR5(5) ● ZIP(Z)

壓縮選項:
- □ 壓縮完將檔案刪除(D)
- □ 建立自我解壓縮檔(X)
- □ 建立結實壓縮檔(S)
- □ 加入修復記錄(E)
- □ 測試壓縮檔(T)
- □ 鎖定壓縮檔(L)

壓縮方式(C): 一般

字典檔大小(I): 32 KB

切成分割檔，大小(V): B



壓縮檔(A): D:\ML\ HW1_0550123_王小明.zip

壓縮檔格式(F):	zip	更新模式(U):	加入並取代檔案
壓縮層級(L):	一般壓縮	路徑模式:	相對路徑
壓縮方式(M):	Deflate		
字典大小(D):	32 KB		
字組大小(W):	32		
結實區塊大小:			
CPU 線程數:	4 / 4		
壓縮時記憶體使用:	131 MB		
解壓縮時記憶體使用:	2 MB		

分割壓縮檔，位元組(V):

參數(P):

選項
- □ 建立自解壓縮檔(X)
- □ 壓縮共用檔案
- □ 壓縮後刪除檔案

加密
- 輸入密碼:
- 重新輸入密碼:
- □ 顯示密碼(S)
- 加密方法: ZipCrypto

- Compress your homework(include source code files and report) to .zip format
- The name of zip file should be HW1_StudentID_Name, like **HW2_0550123_王小明**
- Do not paste source code in your report
- No plagiarism
- No API (such as sikit-learn, LIBSVM) or toolbox is allowed