PYTHON3/ANACONDA WEEK 3

Gates

TOPICS

1. **Installing and setting up Python3/Anaconda IDE. "Hello World".**
2. Installing Python modules using the command line and conda.
3. Basic coding operations in Python: decisions, loops, functions (parameters, return values, scoping).
4. Using files in Python (csv, txt)
5. Using pandas dataframes in Python.
6. Commonly used Python data structures: lists, dictionaries, numpy arrays, sets.
7. Basic math and stats in Python and numpy.
8. Basic plotting in Python with matplotlib.pyplot

HOW TO INSTALL AND TEST ANACONDA

Please use this link tutorial:

http://drgates.georgetown.domains/ANLY500/GetPython.pdf

TOPICS

1. Installing and setting up Python3/Anaconda IDE. "Hello World".
2. **Installing Python modules using the command line and conda.**
3. Basic coding operations in Python: decisions, loops, functions (parameters, return values, scoping).
4. Using files in Python (csv, txt)
5. Using pandas dataframes in Python.
6. Commonly used Python data structures: lists, dictionaries, numpy arrays, sets.
7. Basic math and stats in Python and numpy.
8. Basic plotting in Python with matplotlib.pyplot

INSTALLING NEW PACKAGES/MODULES IN ANACONDA.

1) Sometimes this is easy. You go to your command line. You type :

conda install packagename

and you are done.

2) However, very often, it is not easy. Sometimes, you need to search for available packages…

To do this – go to your command line. In windows, choose Start and then cmd. In MAC, you will have to find it as I do not have a MAC.

On the command line, search for the package you want:

anaconda search –t conda packagename

INSTALLING NEW PACKAGES/MODULES IN ANACONDA.

See here that I have typed: **anaconda search –t conda wordcloud**

Next, it brings up the options. I need to find an option for win-64 because that is the type of OS I have.

Once I find it (if I do), I can install it with this command:

conda install –c conda-forge wordcloud=1.2.1

The format is critical!

C:\Windows\system32\cmd.exe

```
C:\Users\Ami>anaconda search -t conda wordcloud
Using Anaconda Cloud api site https://api.anaconda.org
Run 'anaconda show <USER/PACKAGE>' to get more details:
Packages:
     Name                      |  Version | Package Types   | Platforms
     ------------------------- |   ------ | --------------- | ---------------
     amueller/wordcloud        |    1.2.1 | conda           | linux-64
     conda-forge/wordcloud     |    1.2.1 | conda           | linux-64, win-32,
win-64, osx-64
     pjones/r-wordcloud        |      2.5 | conda           | linux-64
     sibirbil/wordcloud        |    1.1.3 | conda           | osx-64
                                          : A little word cloud generator
     t3kcit/wordcloud          |    1.2.1 | conda           | linux-64
Found 5 packages

C:\Users\Ami>conda install -c conda-forge wordcloud=1.2.1
Fetching package metadata .............
Solving package specifications: .

Package plan for installation in environment C:\Users\Ami\Anaconda3:

The following NEW packages will be INSTALLED:

    wordcloud: 1.2.1-py35_0  conda-forge
```

HERE IS A CLOSER LOOK – TRY THIS ON YOUR ANACONDA

C:\Windows\system32\cmd.exe

```
C:\Users\Ami>anaconda search -t conda wordcloud
Using Anaconda Cloud api site https://api.anaconda.org
Run 'anaconda show <USER/PACKAGE>' to get more details:
Packages:
     Name                      |  Version | Package Types   | Platforms
     ------------------------- |   ------ | --------------- | ---------------
     amueller/wordcloud        |    1.2.1 | conda           | linux-64
     conda-forge/wordcloud     |    1.2.1 | conda           | linux-64, win-32,
win-64, osx-64
     pjones/r-wordcloud        |      2.5 | conda           | linux-64
     sibirbil/wordcloud        |    1.1.3 | conda           | osx-64
                                          : A little word cloud generator
     t3kcit/wordcloud          |    1.2.1 | conda           | linux-64
Found 5 packages

C:\Users\Ami>conda install -c conda-forge wordcloud=1.2.1
Fetching package metadata .............
Solving package specifications: .

Package plan for installation in environment C:\Users\Ami\Anaconda3:

The following NEW packages will be INSTALLED:

    wordcloud: 1.2.1-py35_0  conda-forge
```

PRACTICE:

1) Install the package called pandas.

2) Install the package called wordcloud.

TOPICS

1. Installing and setting up Python3/Anaconda IDE. "Hello World".
2. Installing Python modules using the command line and conda.
3. **Basic coding operations in Python: decisions, loops, functions (parameters, return values, scoping).**
4. Using files in Python (csv, txt)
5. Using pandas dataframes in Python.
6. Commonly used Python data structures: lists, dictionaries, numpy arrays, sets.
7. Basic math and stats in Python and numpy.
8. Basic plotting in Python with matplotlib.pyplot

BASIC PROGRAMMING IN PYTHON3

Class – I have created several chapters that cover basic Python3.

As such, I will not duplicate them here.

Here are links to the all chapters and the topics – please review and practice with the examples to make sure you understand basic Python3:

http://drgates.georgetown.domains/ANLY500/python/

Pay close attention to Chapters 13 & 14 – for data cleaning, wrangling, and APIs, etc.

(Note: there are no chapters 10 – 12)

AFTER YOUR INITIAL REVIEW

1) Once you review the chapters from the previous slide

(http://drgates.georgetown.domains/ANLY500/python/)

you should have a basic understanding of how to use Python 3.

2) You should be comfortable with functions, parameters, scope, files and I/O, data structures such as lists and dictionaries, data wrangling, and using APIs.

TOPICS

1. Installing and setting up Python3/Anaconda IDE. "Hello World".
2. Installing Python modules using the command line and conda.
3. Basic coding operations in Python: decisions, loops, functions (parameters, return values, scoping).
4. **Using files in Python (csv, txt)**
5. **Using pandas dataframes in Python.**
6. Commonly used Python data structures: lists, dictionaries, numpy arrays, sets.
7. Basic math and stats in Python and numpy.
8. Basic plotting in Python with matplotlib.pyplot

READING DATA INTO PYTHON AND USING PANDAS

The next several slides will show examples for reading data into Python.

Examples will include using pandas and dataframes.

Note that basic file I/O including reading, writing, and appending can be located in chapter 8 of the python 3 reference book:
http://drgates.georgetown.domains/ANLY500/python/

PANDAS INTRODUCTION

Gates

REMINDER: BASIC READING IN CSV FILE

```
##Basic open for reading

FILE=open("AirNowResultsLatLong.csv", "r")

##Now, FILE points to the contents of the FILE

## Use a for loop to print the first few lines of
##the file

for line in FILE:

    print(line)
```

```
##Because the file is csv, we can split each line
and
## create a list
for line in FILE:
    LineList = line.split(sep=",")
    print(LineList[0])
FILE.close()

##Using with open
with open("AirNowResultsLatLong.csv") as FILE:
    for line in FILE:
        print(line)
```

USING PANDAS AND DATAFRAMES

```
##An easier way to read a file and use the data
## is to use a package called pandas
##Here, I am importing pandas and calling it pd
##I am also importing numpy and calling it np
import pandas as pd
import numpy as np
```

```
##Note - my dataset has no col names so I will
##create them: see example on next slide
colnames=["city", "state", "date", "type", "value"]
Mydf = pd.read_csv("AirNowResultsLatLong.csv",
names=colnames, header=None)
print(Mydf)
print(Mydf["state"])
print(np.mean(Mydf["value"]))
##To get the names of the columns
print(Mydf.columns)
```

USING PANDAS DATAFRAMES

When using dataframes and pandas, data will appear as a set of rows and columns.

It is also easier to access, manipulate, and aggregate data.

Notice here that the dataframe has the names given on the previous slide. (see code from previous slide)

	city	state	date	type	value
0	Northern Virginia	VA	2016-09-10	OZONE	51
1	Northern Virginia	VA	2016-09-10	PM2.5	55
2	Northern Virginia	VA	2016-09-10	PM10	20
3	Northern Virginia	VA	2016-09-10	OZONE	51
4	Northern Virginia	VA	2016-09-10	PM2.5	55
5	Northern Virginia	VA	2016-09-10	PM10	20
6	Chicago	IL	2016-09-10	OZONE	31
7	Chicago	IL	2016-09-10	PM2.5	33
8	Santa Cruz	CA	2016-09-10	OZONE	28
9	Santa Cruz	CA	2016-09-10	PM2.5	27
10	Fort Lee	NJ	2016-09-10	OZONE	54
11	Fort Lee	NJ	2016-09-10	PM2.5	62

INTRODUCTION TO PYTHON 3 PANDAS

Python pandas (http://pandas.pydata.org/) is an open source library that offers excellent data structures, such as the pandas **dataframe**, as well as a number of analysis tools.

The pandas library is installed with Anaconda and can be used by including the following import statement:

```
import pandas as pd
```

PANDAS: SERIES

```
import numpy as np
import pandas as pd

# Create an array from 0 to 4
myData=np.arange(5)

# Note the index (row) value names
indexValue=["C1", "C2", "C3", "C4", "C5"]
mySeries=pd.Series(myData, index=indexValue)
print(mySeries)
```

The output:

```
C1    0
C2    1
C3    2
C4    3
C5    4
```

PANDAS: SERIES AND DICTIONARIES

```
myDict={"Name":"Bob", "Age":29,
"Degree":"MS"}
```

```
print(pd.Series(myDict))
```

The Output:

Age 29

Degree MS

Name Bob

PANDAS: SERIES

```
myDict2={"Grade1":90.1, "Grade2":88.5,
"Grade3":93.6}


mySeries=pd.Series(myDict2)

print(mySeries)

print("Grade 2 is: ", mySeries[1])

print("The mean of the grades:”,
mySeries.mean())

print("Grades plus 5 points added
is:\n", mySeries+5)

print("Grade 1 is: ",
mySeries.get("Grade1"))
```

The Output:

```
Grade1    90.1
Grade2    88.5
Grade3    93.6

Grade 2 is:  88.5

The mean of the
grades:  90.73
Grades plus 5
points added is:
Grade1    95.1
Grade2    93.5
Grade3    98.6

Grade 1 is:  90.1
```

PANDAS: DATAFRAME

```
import pandas as pd

gradebook={"Student1": pd.Series([89.3, 78.7,
92.2], index=['Grade1', 'Grade2', 'Grade3']),

           "Student2": pd.Series([77.3, 83.4,
91.8], index=['Grade1', 'Grade2', 'Grade3']),

           "Student3": pd.Series([97.1, 88.6,
98.5], index=['Grade1', 'Grade2', 'Grade3'])

           }


gradeBookDF=pd.DataFrame(gradebook)

print(gradeBookDF)
```

OUTPUT: DATA FRAME

	Student1	Student2	Student3
Grade1	89.3	77.3	97.1
Grade2	78.7	83.4	88.6
Grade3	92.2	91.8	98.5

PANDAS DF: CREATE EMPTY DF AND ADD VALUE

```
#Create an empty dataframe

Gradebook2 = pd.DataFrame(Gradebook, index=['G1', 'G2', 'G3'], columns=['Bob
Smith', 'Sandy Stern'])


print(Gradebook2)

#Fill in values

Gradebook2.ix["G1","Bob Smith"]=98.1

print(Gradebook2)
```

The Output:

	Bob Smith	Sandy Stern
G1	NaN	NaN
G2	NaN	NaN
G3	NaN	NaN

	Bob Smith	Sandy Stern
G1	**98.1**	NaN
G2	NaN	NaN
G3	NaN	NaN

PANDAS DF: ADD NEW COLUMN

```
#Create an empty dataframe

Gradebook2 = pd.DataFrame(Gradebook, index=['G1', 'G2', 'G3'],
columns=['Bob Smith', 'Sandy Stern'])

print(Gradebook2)

#Create a new column

Gradebook2["NewColumn"]="NaN"

print(Gradebook2)
```

The Output

```
      Bob Smith   Sandy Stern NewColumn
G1        NaN          NaN        NaN
G2        NaN          NaN        NaN
G3        NaN          NaN        NaN
```

PANDAS DF: ADD VALUES

```
import random
for i in range(len(Gradebook2.BobSmith)):
    Gradebook2.ix[i,"BobSmith"]=random.randint(50,100)
print(Gradebook2)
```

The Output:

```
   BobSmith SandyStern NewColumn
G1       91        NaN       NaN
G2       56        NaN       NaN
G3       63        NaN       NaN
```

PANDAS DF: CONVERT DICT AND ADD

```
MyDict=[{"Name":"Bob", "Age":29, "Degree":"MS"}, {"Name":"Rob",
"Age":34, "Degree":"PhD"}]

DictDF=pd.DataFrame.from_dict(MyDict)

DictDF.insert(2, 'NewColumn', [20007, 23604])

print(DictDF)
```

The Output:

```
   Age Degree  NewColumn Name
0   29     MS      20007  Bob
1   34    PhD      23604  Rob
```

PANDAS DF: DROPPING ROWS AND COLUMNS

```
MyDict=[{"Name":"Bob", "Age":29, "Degree":"MS"}, {"Name":"Rob", "Age":34, "Degree":"PhD"}]

DictDF=pd.DataFrame.from_dict(MyDict)

DictDF.insert(2, 'NewColumn', [20007, 23604])

#REMOVE the “Degree” column


DictDF=DictDF.drop("Degree", axis=1)

#axis=1 is the column, axis=0 is the row


#Remove the first row (row 0)

DictDF=DictDF.drop(0)

print(DictDF)
```

READ CSV TO PANDAS DF

```
csvFile="MyCSVFile3.csv"

File2=open(csvFile, "w", newline='')

Header=(["FirstName", "Lastname", "Grade1", "Grade2",
"Grade3"])

Data1=(["John", "Smith", 90.3, 87.5, 77.2])

Data2=(["Bob", "Benson", 88.8, 77.7, 66.6])

Fwriter=csv.writer(File2)

Fwriter.writerow(Header)

Fwriter.writerow(Data1)

Fwriter.writerow(Data2)

File2.close()

csvDataFrame=pd.read_csv(csvFile)

print(csvDataFrame)
```

The Output:

```
 FirstName Lastname  Grade1  Grade2  Grade3
0      John    Smith    90.3    87.5    77.2
1       Bob   Benson    88.8    77.7    66.6
```

PANDAS DF: ADDING A NEW FEATURE PART 1

```
import pandas as pd
import csv
csvFile="MyCSVFile4.csv"
File2=open(csvFile, "w", newline='')
Header=(["FirstName", "Lastname", "Grade1", "Grade2", "Grade3"])
Data1=(["John", "Smith", 90.3, 97.5, 97.2])
Data2=(["Bob", "Benson", 88.8, 77.7, 66.6])
Data3=(["Sally", "Sue", 78.8, 71.7, 76.6])
Data4=(["Annie", "Apple", 58.8, 67.7, 69.6])
Fwriter=csv.writer(File2)
Fwriter.writerow(Header)
for i in [Data1, Data2, Data3, Data4]:
    Fwriter.writerow(i)
File2.close()
csvDataFrame=pd.read_csv(csvFile)
```

PANDAS DF: ADDING A NEW FEATURE PART 2

```
csvDataFrame["NewFeature"]="NaN"
for i in range(len(csvDataFrame.Grade1)):
    Avg=mean([csvDataFrame.ix[i,"Grade1"], csvDataFrame.ix[i,"Grade2"]])
    if Avg > 89.9:
        csvDataFrame.ix[i,"NewFeature"]="A"
    elif 79.9 < Avg < 90:
        csvDataFrame.ix[i,"NewFeature"]="B"
    elif 69.9 < Avg < 80:
        csvDataFrame.ix[i,"NewFeature"]="C"
    else:
        csvDataFrame.ix[i,"NewFeature"]="D"
print(csvDataFrame)
```

OUTPUT

	FirstName	Lastname	Grade1	Grade2	Grade3	**NewFeature**
0	John	Smith	90.3	97.5	97.2	A
1	Bob	Benson	88.8	77.7	66.6	C
2	Sally	Sue	78.8	81.7	86.6	B
3	Annie	Apple	58.8	67.7	69.6	D

A FULL REVIEW OF READING DATA AND USING PANDAS

Class – the following offers a very good overview of various methods for managing data in Python:

http://drgates.georgetown.domains/ANLY500/python/Chapter14DataWranglingCleaningGates.pdf

Please review this carefully as I will not duplicate the information here and will assume that you have read it.

TOPICS

1. Installing and setting up Python3/Anaconda IDE. "Hello World".
2. Installing Python modules using the command line and conda.
3. Basic coding operations in Python: decisions, loops, functions (parameters, return values, scoping).
4. Using files in Python (csv, txt)
5. Using pandas dataframes in Python.

6. Commonly used Python data structures: lists, dictionaries, numpy arrays, sets. - covered here: http://drgates.georgetown.domains/ANLY500/python/IntroPythonGatesChapter%207__DataStructures_Lists_Dict_.pdf

7. Basic math and stats in Python and numpy. - covered here: http://drgates.georgetown.domains/ANLY500/python/IntroPythonGatesChapter%206__DataTypes_Strings_BasicMath.pdf

8. Basic plotting in Python with matplotlib.pyplot - covered here: http://drgates.georgetown.domains/ANLY500/python/IntroPythonGatesChapter%209__BasicGraphics.pdf

MORE ON NUMPY

RE: http://www.numpy.org/

https://docs.scipy.org/doc/numpy-dev/user/quickstart.html

The Basics

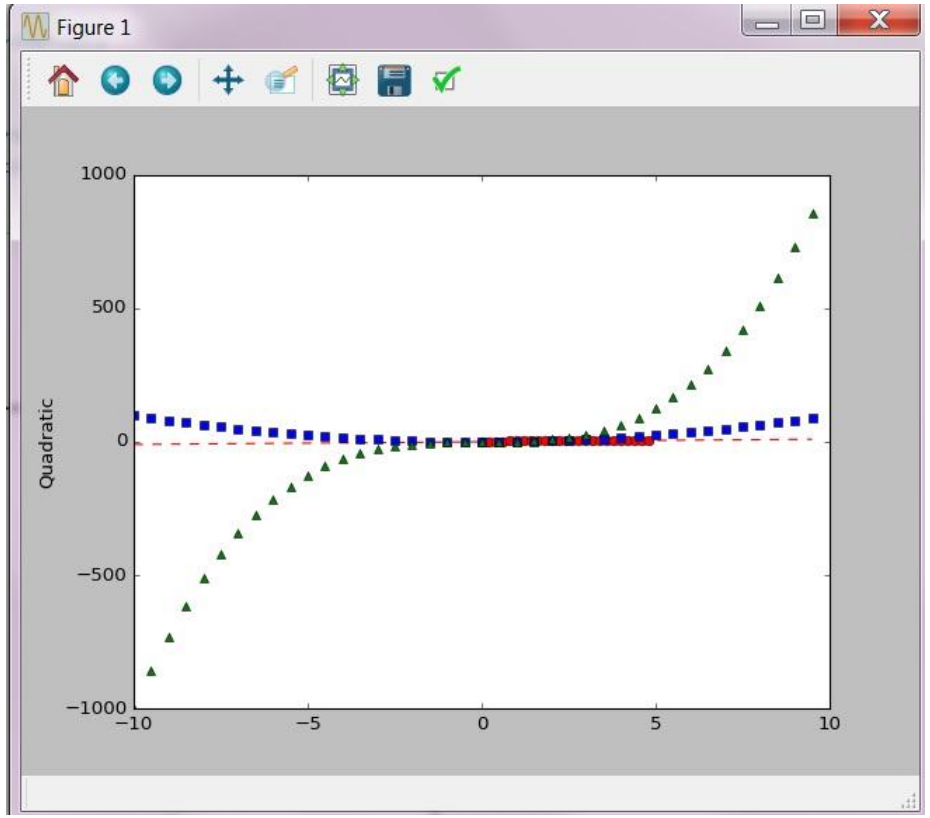
NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank*.

For example, the coordinates of a point in 3D space `[1, 2, 1]` is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[ 1., 0., 0.],
 [ 0., 1., 2.]]
```

NumPy's array class is called `ndarray`. It is also known by the alias `array`.

NUMPY AND PLOT EXAMPLES




```python
##numpy
import numpy as np
a = np.arange(15).reshape(3, 5)
print(a)
print(a.shape)
print(a.ndim)

b=np.arange( 10, 30, 5 )
print(b)
from numpy import pi
numbers=np.linspace( 0, 2, 9 )
print(numbers)
x = np.linspace( 0, 2*pi, 100 )
f = np.sin(x)
print(f)

x = np.arange(0., 5., 0.2)
y=x + 2

import matplotlib.pyplot as plt
plt.plot(x,y, "ro")
plt.ylabel('Quadratic')
plt.show()

t=np.arange(-10,10,.5)
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```

NUMPY AND MATPLOTLIB.PYPLOT

1) numpy is often used for functions, calculations, and plotting.

2) matplotlib is Python's standard plotting package and pyplot is the most common library.

3) Use the Internet and my chapter references to review these Python options.

References:

https://matplotlib.org/users/pyplot_tutorial.html

https://docs.scipy.org/doc/numpy-dev/user/quickstart.html

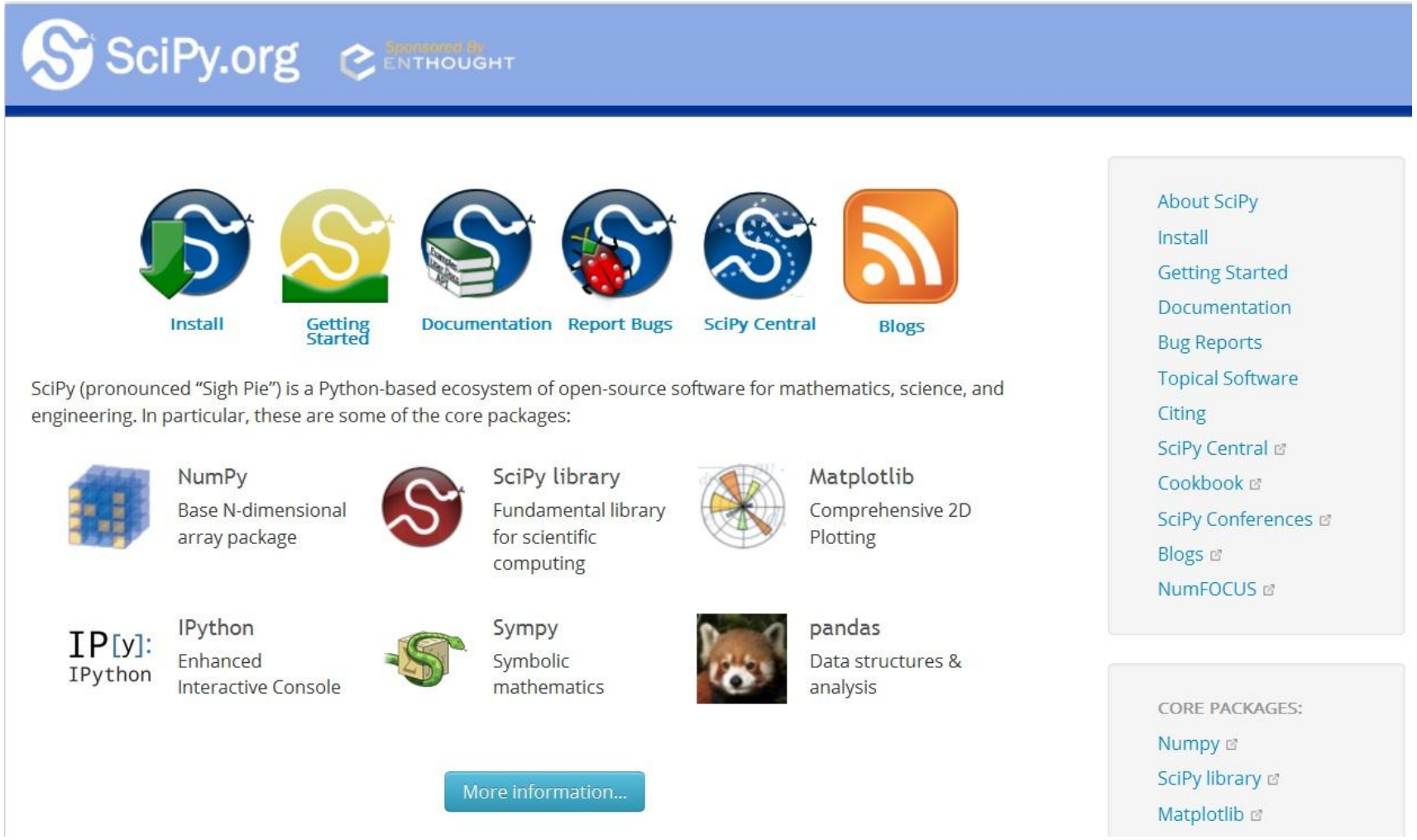
https://docs.scipy.org/doc/numpy-1.12.0/reference/routines.statistics.html

https://docs.scipy.org/doc/scipy/reference/stats.html

SCIPY: SCIPY.ORG

scipy contains core and common Python packages.

Review the site and investigate numpy, scipy, pandas, and matplotlib.



SCIKIT-LEARN

The scikit-learn package contains a lot of analysis tools.

http://scikit-learn.org/stable/index.html

1) PCA

http://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_iris.html

2) Classification – SVM – Random Forest

http://scikit-learn.org/stable/supervised_learning.html#supervised-learning

3) Clustering – kmeans

http://scikit-learn.org/stable/modules/clustering.html#clustering

It is a good idea to read these pages and to gain some practice and familiarity with these packages. I recommend kmeans, SVM, and random forest.

SCIKIT-LEARN

Machine Learning for Python 3
Intro to Clustering
Gates

SCIPY.ORG

SciKits(short for SciPyToolkits)

Getting scikit-Learn

http://scikits.appspot.com/scikit-learn

All SciKits

http://scikits.appspot.com/scikits

STEPS FOR GETTING SCIKIT-LEARN

Homepage: http://scikit-learn.org/stable/

Built on: NumPy, Matplotlib, Scipy

To Install: http://scikit-learn.org/stable/install.html

C:\Windows\system32\cmd.exe

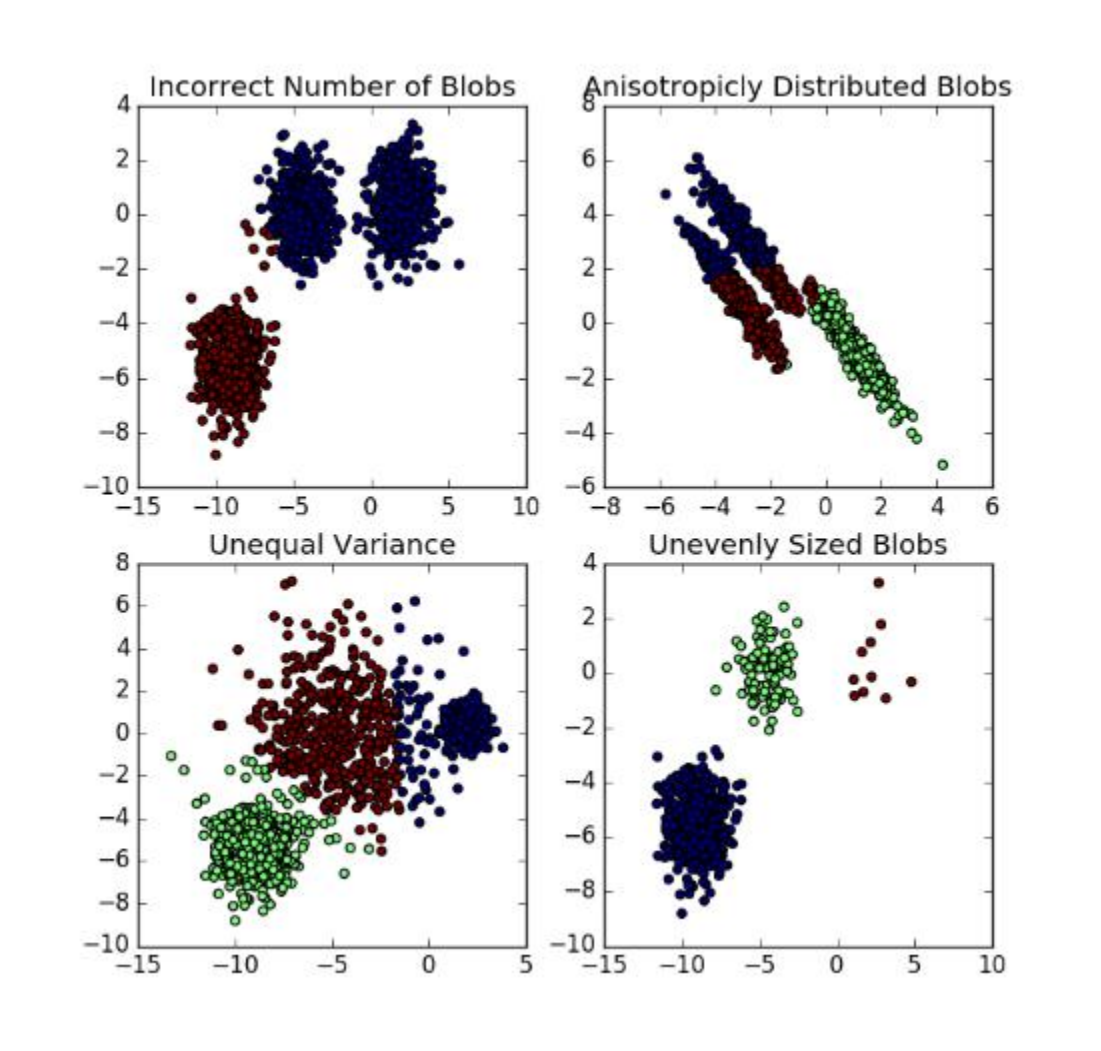
```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Ami>conda install scikit-learn_
```

C:\Windows\system32\cmd.exe

```
                                          Total:       128.6 MB

The following packages will be UPDATED:

    conda:        4.1.12-py35_0      --> 4.2.7-py35_0
    conda-env:    2.5.2-py35_0       --> 2.6.0-0
    mkl:          11.3.1-0           --> 11.3.3-1
    mkl-service:  1.1.2-py35_0       --> 1.1.2-py35_2
    numexpr:      2.5-np110py35_0    --> 2.6.1-np111py35_0
    numpy:        1.10.4-py35_0      --> 1.11.1-py35_1
    scikit-learn: 0.17.1-np110py35_0 --> 0.17.1-np111py35_1
    scipy:        0.17.0-np110py35_0 --> 0.18.1-np111py35_0

Proceed ([y]/n)? y

Fetching packages ...
conda-env-2.6. 100% |################################| Time: 0:00:00 248.99 kB/s
mkl-11.3.3-1.t 100% |################################| Time: 0:01:08   1.68 MB/s
mkl-service-1. 100% |################################| Time: 0:00:00 730.31 kB/s
numpy-1.11.1-p 100% |################################| Time: 0:00:01   1.68 MB/s
conda-4.2.7-py 100% |################################| Time: 0:00:00   1.26 MB/s
numexpr-2.6.1- 100% |################################| Time: 0:00:00   1.21 MB/s
scipy-0.18.1-n 100% |################################| Time: 0:00:48 254.79 kB/s
scikit-learn-0 100% |################################| Time: 0:00:04 801.84 kB/s
Extracting packages ...
[      COMPLETE      ]|##################################################| 100%
Unlinking packages ...
[      COMPLETE      ]|##################################################| 100%
Linking packages ...
[      COMPLETE      ]|##################################################| 100%

C:\Users\Ami>
```

ON A PC, FROM CMD

TEST YOUR INSTALL: SCIKIT-LEARN K MEANS



http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html

A K MEANS CODE EXAMPLE AND TUTORIALS

The Code: http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html

Imports

sklearn.cluster KMeans

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans

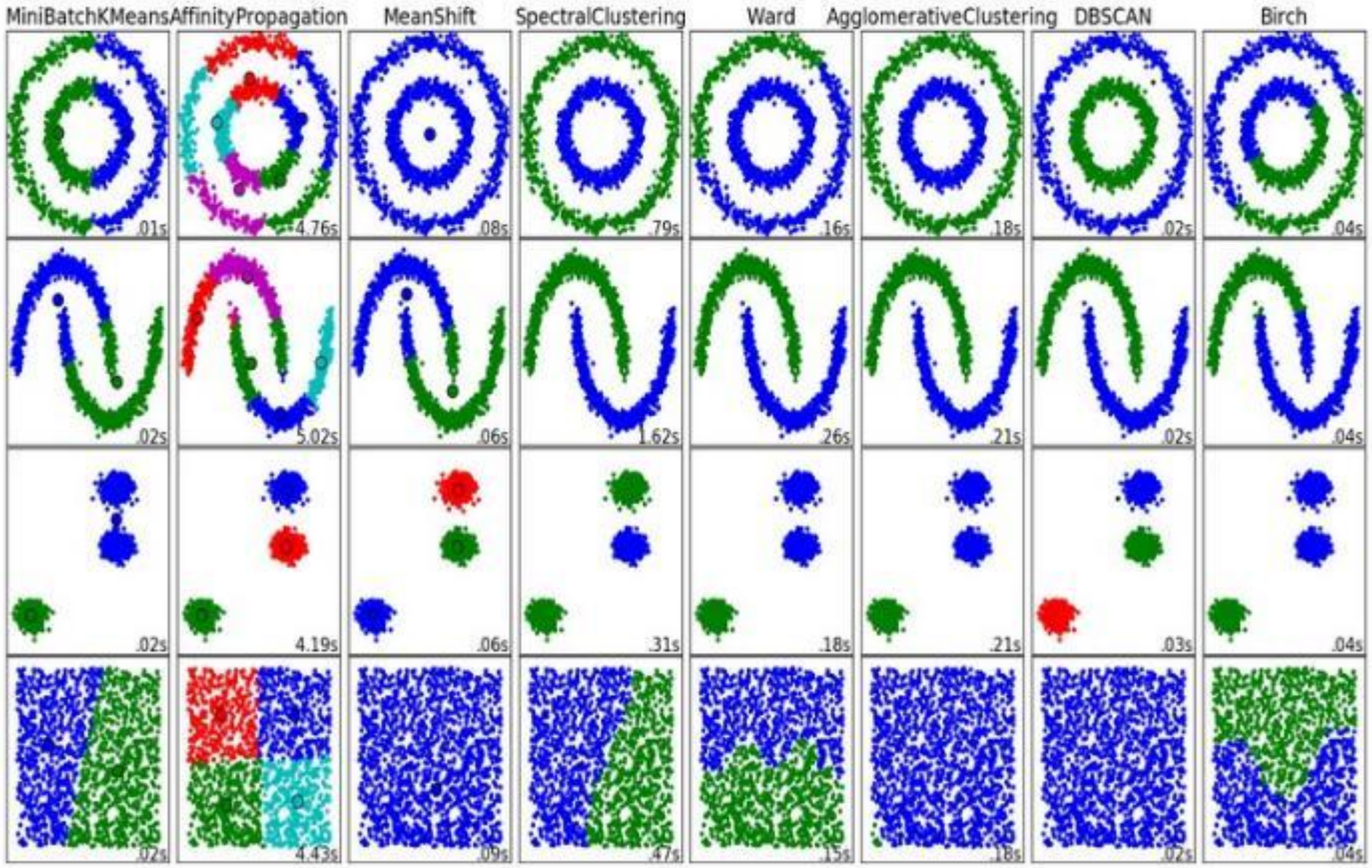
sklearn.datasets make_blobs

http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html#sklearn.datasets.make_blobs

matplotlib.pyplot (See Ami's Python Book – Chapter 9)

SCIKIT-LEARN CLUSTERING ALGORITHMS

2.3.1. Overview of clustering methods



A comparison of the clustering algorithms in scikit-learn

K MEANS

sklearn.cluster.KMeans

class `sklearn.cluster.KMeans`(*n_clusters=8, init='k-means++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='auto', verbose=0, random_state=None, copy_x=True, n_jobs=1*)

Methods

`fit` (X[, y])	Compute k-means clustering.
`fit_predict` (X[, y])	Compute cluster centers and predict cluster index for each sample.
`fit_transform` (X[, y])	Compute clustering and transform X to cluster-distance space.
`get_params` ([deep])	Get parameters for this estimator.
`predict` (X)	Predict the closest cluster each sample in X belongs to.
`score` (X[, y])	Opposite of the value of X on the K-means objective.
`set_params` (**params)	Set the parameters of this estimator.
`transform` (X[, y])	Transform X to a cluster-distance space.

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans

CODE EXAMPLE

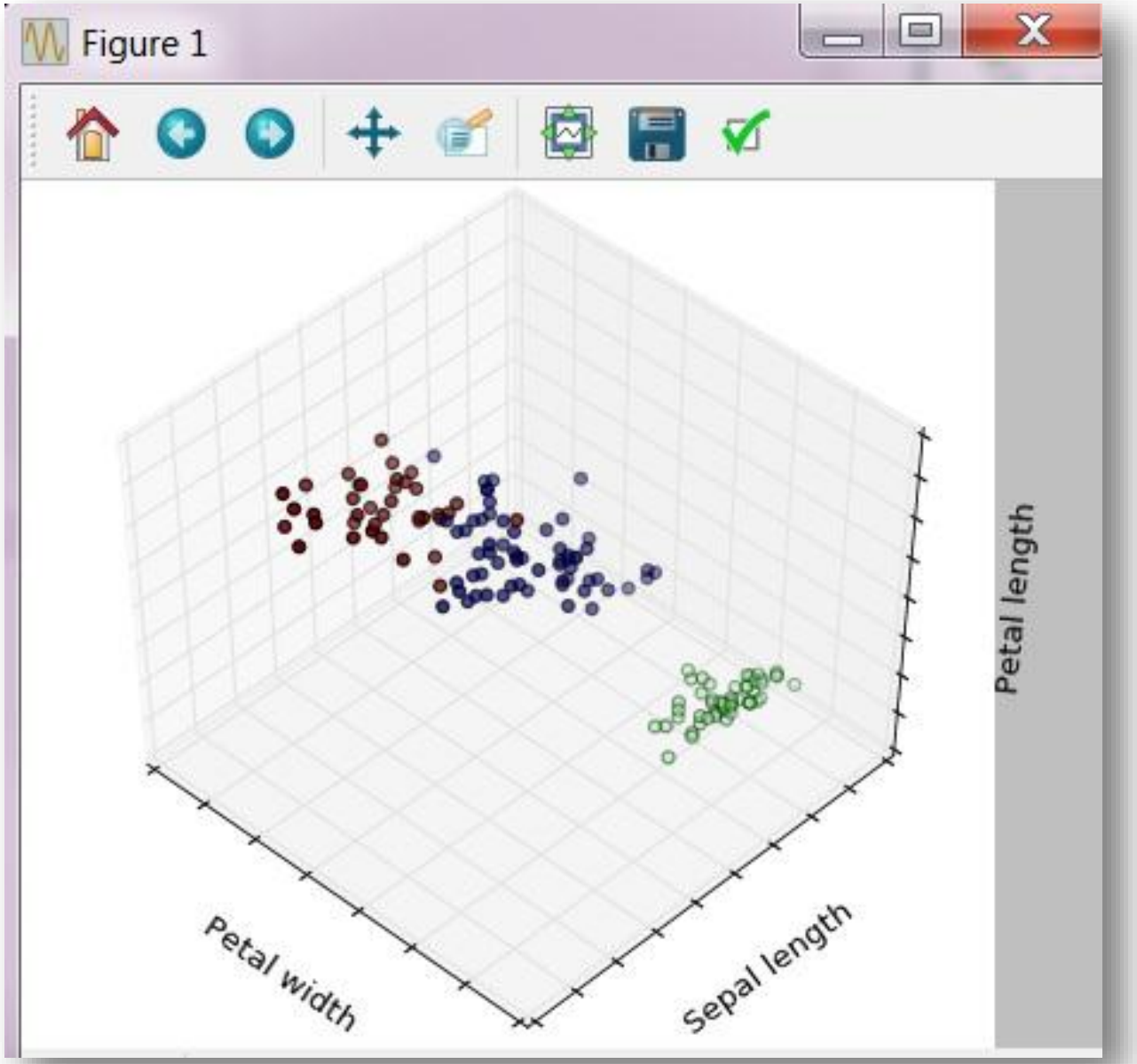
```
#kmeansSMALL.py

import numpy as np

import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import
Axes3D

from sklearn.cluster import KMeans

from sklearn import datasets

iris = datasets.load_iris()

#print(iris.data[0:5])

#Iris dataset:  http://scikit-
#learn.org/stable/auto_examples/data
sets/plot_iris_dataset.html

X = iris.data

y = iris.target
```

```
Result=KMeans(n_clusters=3,
verbose=1)
fignum = 1
fig = plt.figure(fignum, figsize=(4,
3))
#Axes3D is part of Matplot lib, rect
defines the rectangle
#elev stores the elevation angle in
the z plane
#azim stores the azimuth angle in the
x,y plane
ax = Axes3D(fig, rect=[0, 0, .90, 1],
elev=48, azim=134)
#fit(X[, y])Compute k-means
clustering
Result.fit(X)
```

CODE CONT.

```
#labels_: Labels of each point, attribute of Kmeans

labels = Result.labels_

#print(labels)

#X[:, 3] are the x's, X[:, 0] are the y's, X[:, 2] are the z's

#ref: http://matplotlib.org/mpl_toolkits/mplot3d/tutorial.html

#c can be single color format str or a seq of color specs of len
N

# The labels are an array. labels.astype(np.float) casts to
float

ax.scatter(X[:, 3], X[:, 0], X[:, 2], c=labels.astype(np.float))

ax.w_xaxis.set_ticklabels([])

ax.w_yaxis.set_ticklabels([])

ax.w_zaxis.set_ticklabels([])

ax.set_xlabel('Petal width')

ax.set_ylabel('Sepal length')

ax.set_zlabel('Petal length')

plt.show()
```

OUTPUT



DPSCAN CODE (AND VS KMEANS)

The following code is broken into many parts.

The code has several comments that offer web references and resources.

IMPORTING AND CREATING DATA

```
import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn import cluster, datasets
from sklearn.preprocessing import StandardScaler

n_samples = 500

noisy_circles = datasets.make_circles(n_samples=n_samples, factor=.5, noise=.05)
noisy_moons = datasets.make_moons(n_samples=n_samples, noise=.05)
blobs = datasets.make_blobs(n_samples=n_samples, random_state=8)

#Reference
#http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html
#no_structure = np.random.rand(n_samples, 2)
```

SETTING UP COLORS, NAMES, ETC.

```
colors = np.array([x for x in 'bgrcmykbgrcmykbgrcmykbgrcmyk'])
colors = np.hstack([colors] * 20)

clustering_names = ['KMeans','DBSCAN']

plt.figure(figsize=(len(clustering_names) * 2 + 3, 10))
plt.subplots_adjust(left=.02, right=.98, bottom=.001, top=.96, wspace=.25,hspace=.25)

#Reference:
#http://matplotlib.org/faq/howto_faq.html
```

NORMALIZE DATA: PREPROCESSING

```
plot_num = 1

datasets = [noisy_circles, noisy_moons, blobs]

for i_dataset, dataset in enumerate(datasets):
    X, y = dataset
    print("Std and mean of original data :", round(np.std(X),2), "and ",
round(np.mean(X),2))
    # normalize dataset for easier parameter selection
    #Reference
    #http://scikit-
#learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
    X = StandardScaler().fit_transform(X)
    print("Std and mean of whitened data :", round(np.std(X),2), "and ",
round(np.mean(X),2))
```

SET UP DPSCAN AND KMEANS

```
dbscan = cluster.DBSCAN(eps=.2)

#eps: The maximum distance between two samples for
them #to be considered as in the same neighborhood.
#ref:
# http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
# http://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html

KMeans=cluster.KMeans(n_clusters=3)
#n_clusters is your choice of k

clustering_algorithms = [KMeans, dbscan]
```

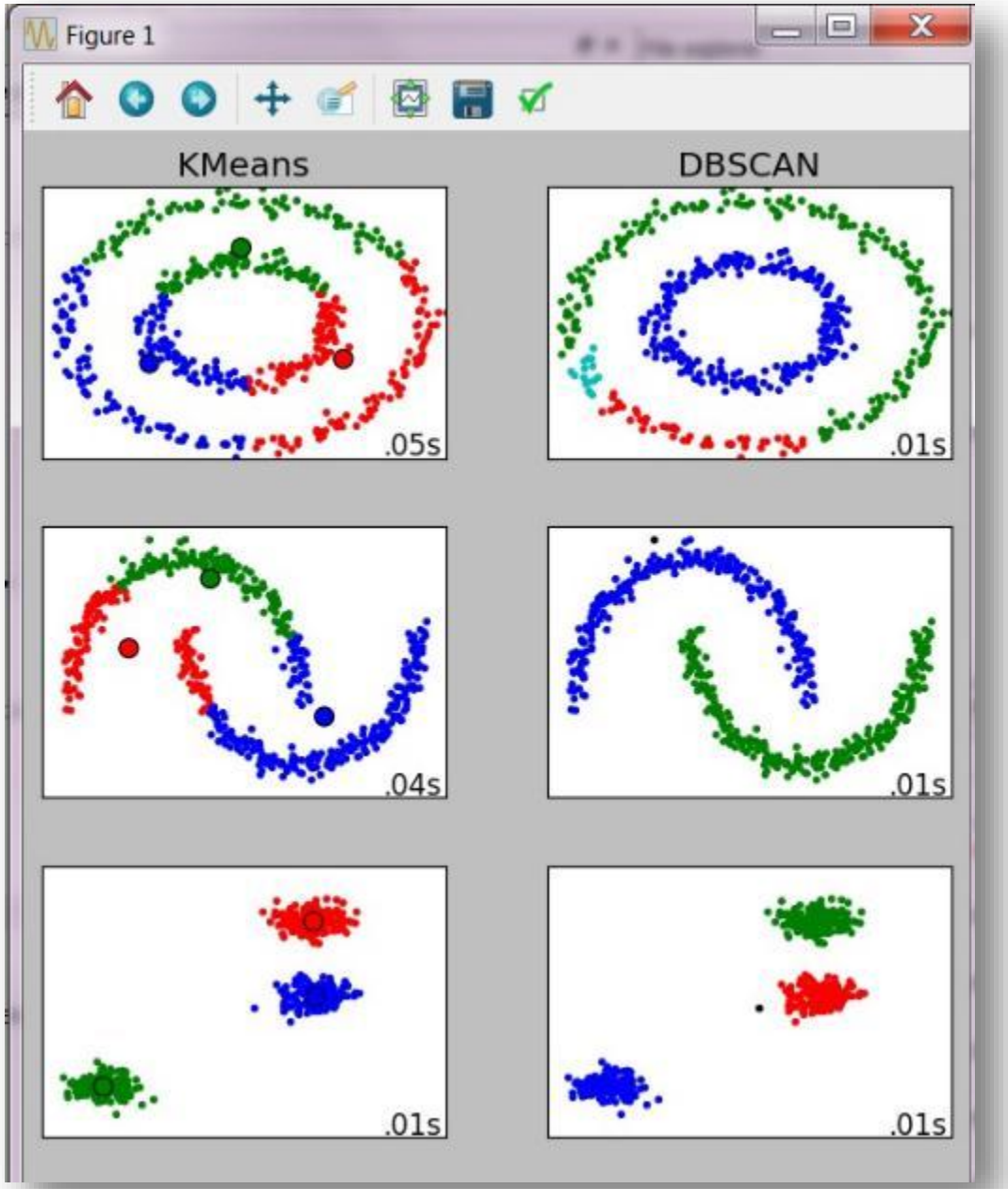
RUN THE CLUSTERING ALGORITHMS

```
for name, algorithm in zip(clustering_names, clustering_algorithms):
        #record the time
        t0 = time.time()

        #run each cluster algorithm
        algorithm.fit(X)
        #note duration
        t1 = time.time()

        if hasattr(algorithm, 'labels_'):
            #cast labels to int
            # labels_ : Labels of each point
            y_pred = algorithm.labels_.astype(np.int)
        else:
            #kmeans:Predict the closest cluster each X belongs to
            #http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
            y_pred = algorithm.predict(X)
```

ALSO IN THE **FOR LOOP** – MAKE THE PLOTS

```
        # plot – all of the following is in the for loop on the prev slide
        #reference
        #http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.subplot
        plt.subplot(4, len(clustering_algorithms), plot_num)
        if i_dataset == 0:
            plt.title(name, size=18)

        plt.scatter(X[:, 0], X[:, 1], color=colors[y_pred].tolist(), s=10)

        #https://docs.python.org/3/library/functions.html#hasattr
        if hasattr(algorithm, 'cluster_centers_'):
            centers = algorithm.cluster_centers_
            center_colors = colors[:len(centers)]
            plt.scatter(centers[:, 0], centers[:, 1], s=100, c=center_colors)
        plt.xlim(-2, 2)
        plt.ylim(-2, 2)
        plt.xticks(())
        plt.yticks(())
        plt.text(.99, .01, ('%.2fs' % (t1 - t0)).lstrip('0'), transform=plt.gca().transAxes, size=15,
             horizontalalignment='right')
        plot_num += 1

#Not in for loop – shows the plot
plt.show()
```

RESULTING CLUSTER PLOT



FURTHER RESOURCES

Kumar's Book - Chapter 8
http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

Excellent Python Clustering Notes and Examples
http://scikit-learn.org/stable/modules/clustering.html#k-means

KMeans with scikit-learn examples
http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_iris.html

MatplotLib
http://matplotlib.org/mpl_toolkits/mplot3d/tutorial.html

The Public Iris Dataset
http://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

Preproccessing Data and Normalization
http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html