text_mining_tutorial02

October 25, 2018

0.1 Introduction to python

0.1.1 Important Libraries:

- Any library can be downloaded by using the pip tool:

```
pip install library-name    # library-name = matplotlib, numpy, nltk ....
```

- In this lecture we will be using the following libraries:
  - numpy
  - nltk
  - matplotlib
  - pandas (propably)

Numpy (http://www.numpy.org/)

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

```
In [1]: import numpy as np

        a = np.array([1, 2, 3])   # Create a rank 1 array
        print(a.shape)            # Prints "(3,)"

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

        #Slicing the matrix
        print(b[0,0]) # prints the element in row=0, column=0
        print(b[0:2, 0:2]) # slice the matrix by taking the first two columns, and two rows
        print(b[-1, 0:2]) # retrieve the first two columns of the last row
```

```
(3,)
(2, 3)
1
[[1 2]
 [4 5]]
[4 5]
```

```
In [2]: 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.]]"


        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
```

```
[[0. 0.]
 [0. 0.]]
[[1. 1.]]
[[1. 0.]
 [0. 1.]]
[[0.66322849 0.92525543]
 [0.56123937 0.85778243]]
```

```
In [3]: x = np.array([[1,2],[3,4]], dtype=np.float64)
        y = np.array([[5,6],[7,8]], dtype=np.float64)

        print(x + y)
        print('========')
        print(x - y)
        print('========')
        print(x * y) #Element wise multiplication
        print('========')
        print(x / y)
        print('========')
        print(np.sqrt(x))
        print('========')
        print(np.dot(x, y)) #Matrix multiplication
```

```
[[ 6.  8.]
 [10. 12.]]
========
[[-4. -4.]
 [-4. -4.]]
========
[[ 5. 12.]
 [21. 32.]]
```

```
========
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
========
[[1.         1.41421356]
 [1.73205081 2.        ]]
========
[[19. 22.]
 [43. 50.]]
```

NLTK (https://www.nltk.org/)

- nltk is a leading platform for building Python programs to work with human language data.

Tokenization:

```
In [4]: import nltk

        text = """ At eight o'clock on Thursday morning Arthur didn't feel very good."""
        tokens = nltk.word_tokenize(text)
        print('Tokens:' , tokens)

        text = """ Diabetes mellitus is a group of metabolic diseases characterized by high blo
        levels that result from defects in insulin secretion, or its action, or both.
        Diabetes mellitus, commonly referred to as diabetes (as it will be in this article)
        was first identified as a disease associated with "sweet urine," and excessive muscle l
        """
        sents = nltk.sent_tokenize(text)
        print('Sentences:', sents)
```

```
Tokens: ['At', 'eight', "o'clock", 'on', 'Thursday', 'morning', 'Arthur', 'did', "n't", 'feel'
Sentences: [' Diabetes mellitus is a group of metabolic diseases characterized by high blood su
```

Text Corpora:

```
In [5]: from nltk.corpus import gutenberg
        #Downloading gutenburg dataset
        nltk.download('gutenberg')
```

```
[nltk_data] Downloading package gutenberg to
[nltk_data]     /home/miladalshomary/nltk_data...
[nltk_data]   Package gutenberg is already up-to-date!
```

```
Out[5]: True
```

```
In [6]: macbeth_sentences = gutenberg.sents('shakespeare-macbeth.txt')
        print('Sentences:', macbeth_sentences[0:10])
        print('======================')

        macbeth_words = gutenberg.words('shakespeare-macbeth.txt')
        print('Words:', macbeth_words[0:10])
        print('======================')
```

```
Sentences: [['[', 'The', 'Tragedie', 'of', 'Macbeth', 'by', 'William', 'Shakespeare', '1603',
======================
Words: ['[', 'The', 'Tragedie', 'of', 'Macbeth', 'by', 'William', 'Shakespeare', '1603', ']']
======================
```

Stemming:

```
In [7]: from nltk.stem import PorterStemmer

        ps = PorterStemmer()

        words = ["game","gaming","gamed","games"]

        print(list(map(lambda x: ps.stem(x), words)))
```

```
['game', 'game', 'game', 'game']
```

Part of Speech Tagging:

```
In [8]: nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /home/miladalshomary/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

```
Out[8]: True
```

```
In [19]: from nltk import pos_tag

         document = """Today the Netherlands celebrates King\'s Day.
         To honor this tradition, the Dutch embassy in San Francisco invited me to'
         """
         sentences = nltk.sent_tokenize(document)
         print('Sentences:')
         print(sentences)
         data = list(map(lambda x: nltk.pos_tag(nltk.word_tokenize(x)), sentences))
         print('\nPOS tags:')
         print(data[0]) #printing the tokens of the first sentence and thier POS tags
```

```
Sentences:
["Today the Netherlands celebrates King's Day.", "To honor this tradition, the Dutch embassy ir

POS tags:
[('Today', 'NN'), ('the', 'DT'), ('Netherlands', 'NNP'), ('celebrates', 'VBZ'), ('King', 'NNP'
```

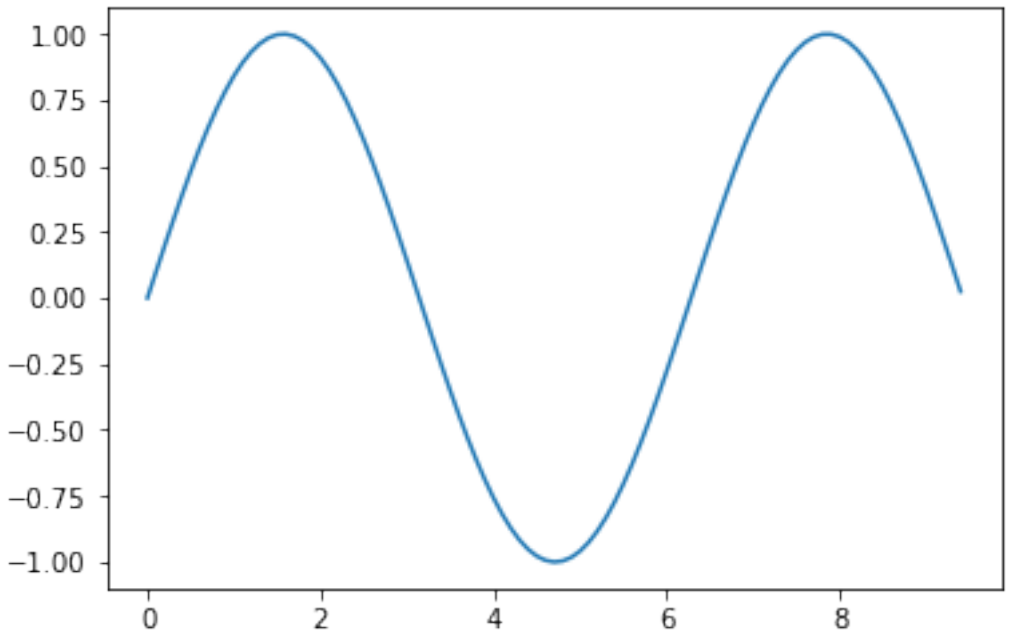
Matplotlib (https://matplotlib.org/)

- Python 2D plotting library which produces publication quality figures in a variety of hard-copy formats and interactive environments across platforms

```
In [20]: 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.show()  # You must call plt.show() to make graphics appear.
```



Pandas (https://pandas.pydata.org/)

- A Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive

- In the following we will be working with the iris dataset, provided as a csv file.

```
In [11]: import pandas as pd

         #Loading dataset from csv file into dataframe object
         df = pd.read_csv('iris.csv')
         #display first few rows
         df.head()
```

Out[11]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

```
In [12]: #Desciptive analysis on a dataframe
         df.describe()
```

Out[12]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
In [13]: #Slicing and accessing values
         df[0:3]
```

Out[13]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa

```
In [14]: df['sepal_width'][0:3]
```

```
Out[14]: 0    3.5
         1    3.0
         2    3.2
         Name: sepal_width, dtype: float64
```