ccna.ro

Python Course

Course 4

Modules

- de ce avem nevoie de modularitate?
- cum putem obtine modularitate?
- Work smart, not hard!

Python Modules

- au cautat cel mai simplu mod de a lucra intre script-uri/biblioteci.
- Python-ul poate fi trecut in C ca limbaj intermediar (nu mereu) -> ofera viteza
- Sistem de versionare usor de folosit: pip

Python Modules

- Cum ne face propriul modul?
- Demo

CSV

- comma separated file format
- datele sunt tinute separate de virgula
- primul rand dintr-un csv

CSV

- demo la tabla despre cum arata un csv.
- analizam impreuna csv-ul folosit in cadrul laboratorului cu formatia beatles.
- chiar si in cadrul unei probleme relativ simple, este mai bine sa folosim un modul deja implementat!

Modulul requests

Este un modul python care permite trimiterea de cereri HTTP (GET, POST, PUT etc) si accesarea raspunsurilor acestora intr-o maniera user-friendly.

Modulul requests

O aplicatie foarte utila este descarcarea paginilor HTML de la anumite adrese in vederea prelucrarii si extragerii de date.

GET

GET este o cerere HTTP prin care putem obtine date de la un server.

Un request de tip GET nu ar trebui sa prelucrezele datele in vreun fel.

POST

POST este o cerere HTTP prin care putem trimite date catre un server.

Un request de tip POST nu ar trebui sa prelucrezele datele in vreun fel.

Beautiful Soup 4

Este o biblioteca care permite extragerea de de date din interiorul fisierelor de tip HTML si XML.

In combinatie cu modulul requests permite obtinerea de informatii, link-uri, imagini etc. din orice pagina web.

Beautiful Soup 4

De multe ori, pentru a gasi stringuri mai complexe (adrese de e-mail, numere de telefon etc.) sau anumite tag-uri HTML este nevoie sa folosim un regex.

```
for script2 in soup.findAll("script", 'id="[0-9]+_wrapper"'):
    newFile.append(str(script2) + "\n")
```

```
<script type="text/javascript" id=" 1728189_wrapper"
src="http://domain.com/wrapper.js"></script>
```

os

- modul pentru a interactiona cu sistemul de operare.
- ajuta in cazul scripting-ului.
- in sfarsit putem sa nu mai scriem bash.

SYS

- interactionam cu env-ul script-ului
- putem citi argumente din linia de comanda cu ajutorul lui.
- putem citi/scrie direct la stdin/stdout sau putem crea pipe-uri.

Why Python?

- suport pentru aproape orice.
- prototipare rapida.
- suport matematic puternic. (numpy)

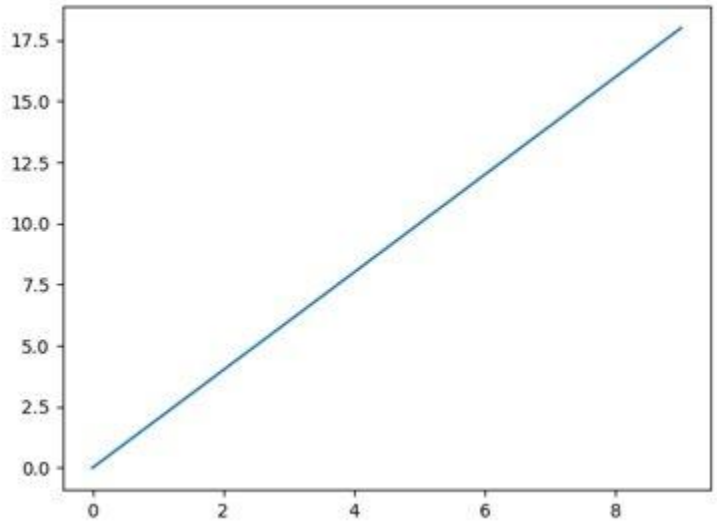
Python - Machine Learning

- In ultimii ani python a devenit tot mai popular in comunitatea de machine learning. De ce?

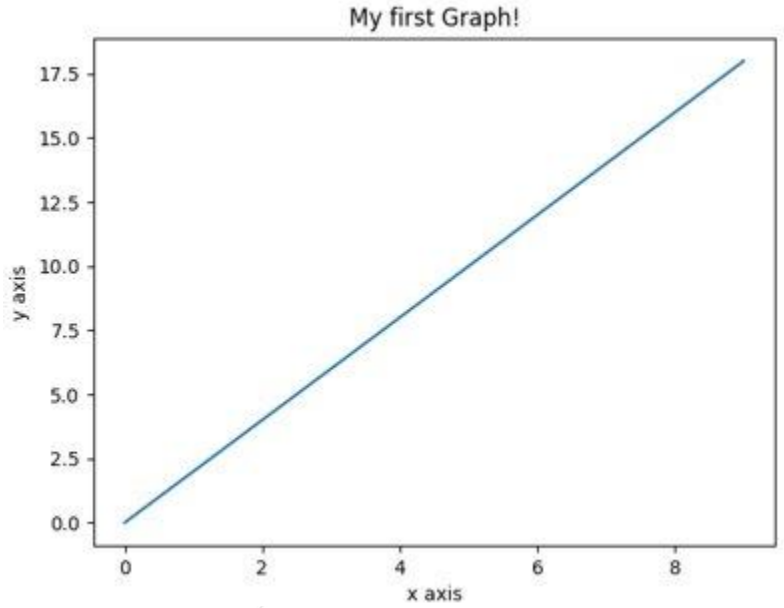
Exemplu practic: Matplotlib

- o parte importanta din machine learning este vizualizarea datelor.
- majoritatea limbajelor de programare au un mod greoi de a plota grafice.

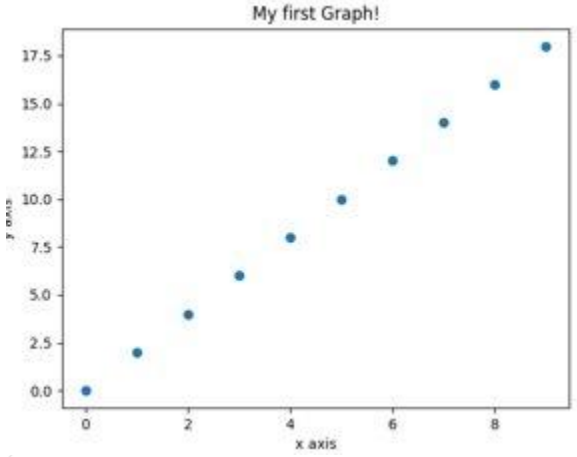
```python
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 10)
y = x * 2
plt.plot(x, y)
plt.show()
```



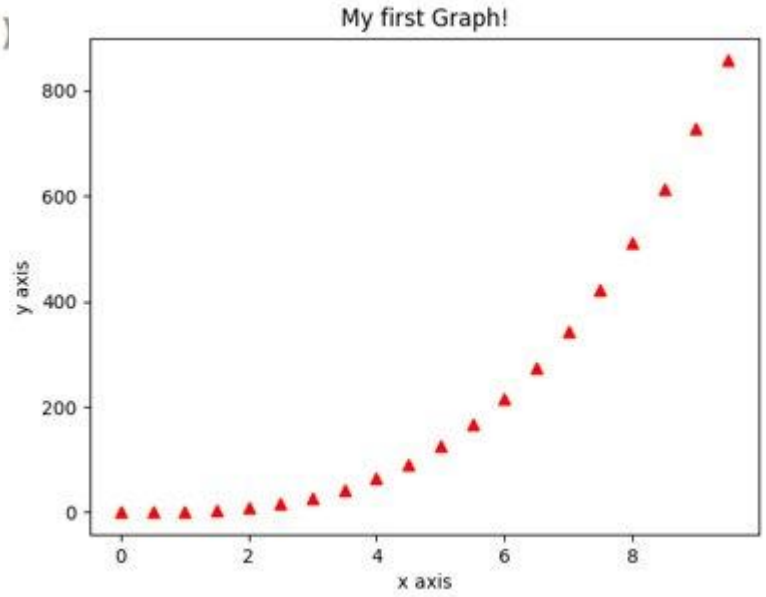
```python
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 10)
y = x * 2
plt.title("My first Graph!")
plt.xlabel("x axis")
plt.ylabel("y axis")
plt.plot(x, y)
plt.show()
```



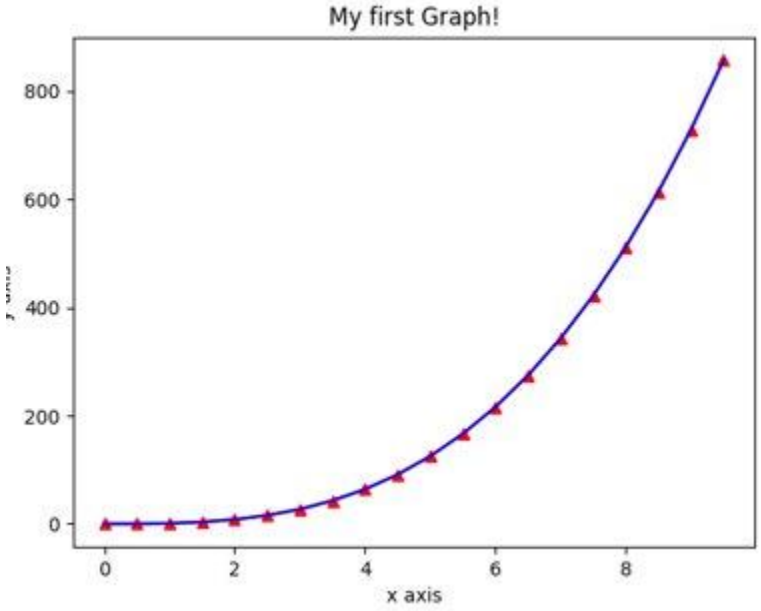
```python
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 10)
y = x * 2
plt.title("My first Graph!")
plt.xlabel("x axis")
plt.ylabel("y axis")
plt.scatter(x, y)
plt.show()
```



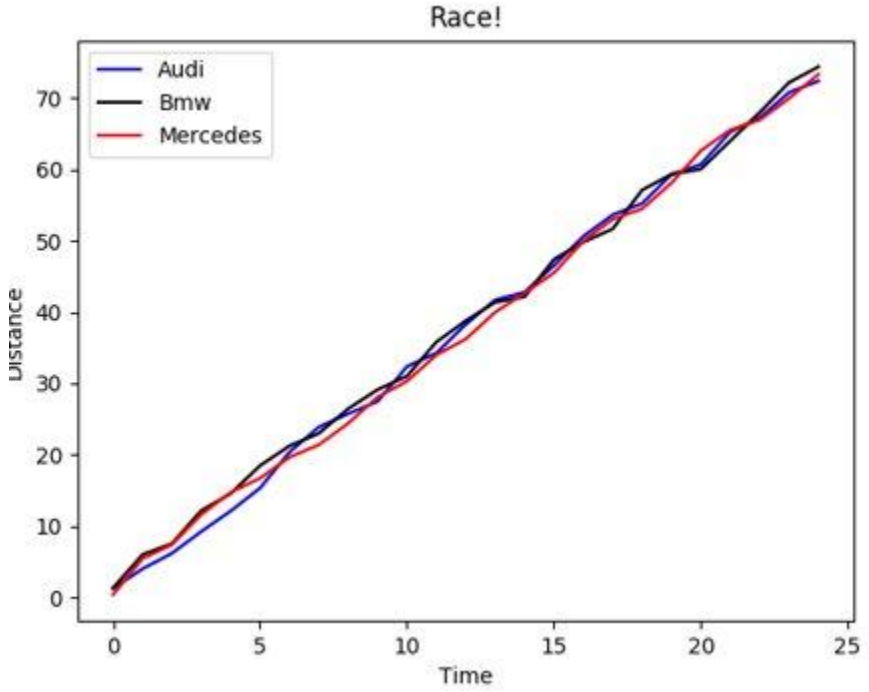
```
1   import numpy as np
2   import matplotlib.pyplot as plt
3
4   x = np.arange(0, 10, 0.5)
5   y = (x ** 3)
6   plt.title("My first Graph!")
7   plt.xlabel("x axis")
8   plt.ylabel("y axis")
9   plt.scatter(x, y, c='r', marker='^')
10  plt.show()
```



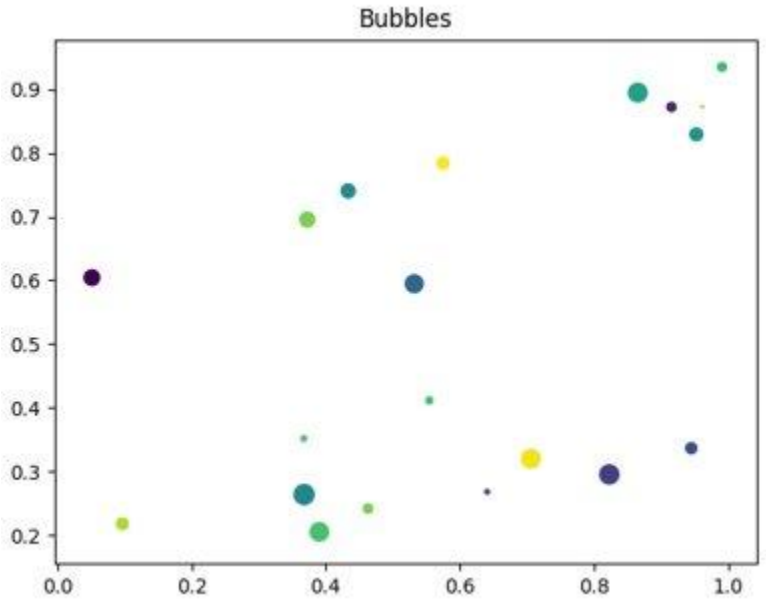
```python
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 10, 0.5)
y = (x ** 3)
plt.title("My first Graph!")
plt.xlabel("x axis")
plt.ylabel("y axis")
plt.plot(x, y, c='r', marker='^')
plt.plot(x, y, c='b')
plt.show()
```



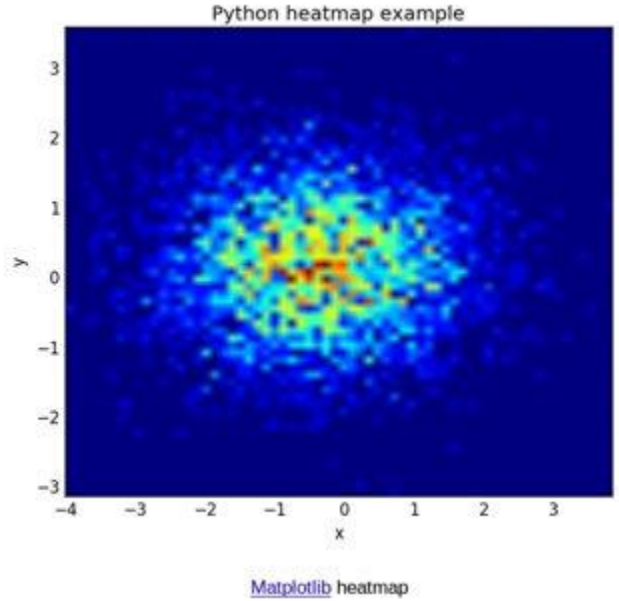
```python
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0, 25, 1)
audi = np.random.randint(low=1, high=5)*x + 3 *np.random.rand(len(x))
bmw = np.random.randint(low=1, high=5)*x + 3.5 *np.random.rand(len(x))
mercedes = np.random.randint(low=1, high=5)*x + 2.8 *np.random.rand(len(x))

plt.title("Race!")
plt.xlabel("Time")
plt.ylabel("Distance")
plt.plot(x, audi, c='blue')
plt.plot(x, bmw, c = 'black')
plt.plot(x, mercedes, c = 'red')
plt.legend(['Audi', 'Bmw', 'Mercedes'])
plt.show()
```



```
1   import matplotlib.pyplot as plt
2   import numpy as np
3
4   # create data
5   x = np.random.rand(20)
6   y = np.random.rand(20)
7   z = np.random.rand(20)
8   colors = np.random.rand(20)
9   # use the scatter function
10  plt.scatter(x, y, s=z*100,c=colors)
11  plt.title('Bubbles')
12  plt.show()
```



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

a = np.random.random((40, 40))
plt.imshow(a, cmap='hot', interpolation='nearest')
plt.show()
```



Matplotlib heatmap