

- Core Library for computing in Python
- Data Science, Machine Learning ve Deep Learning te kullanılıyor.
- Yüksek performanslı çok boyutlu arraylar de çok hızlı.
- Arrayler için de birçok matematiksel operatörler bulunduruyor .
- NumPy , C ile yazılmış bir altyapıya sahip. Python fonksiyonları da aslında bu C ile sarmalananmiş bir yapıda.

NumPy Kullanım Alanları

- Array/Matrix işlemleri
- Lineer Cebir
- Dot Products
- Özdeğer Özvektör
- Determinant işlemleri
- Görüntü İşleme

NumPy 'ı koduna " pip install numpy" komutuyla ekleyebilirsin terminalden.Eğer Anaconda ile çalışıyorsan conda install numpy yazabilirsin terminale.

Numpy'ı sürekli fonksiyon kullanırken uzun ismiyle yazmak istemiyorsan import numpy as np olarak yazabilirsin.

Array oluşturma örneği:
a= np.array([1,2,3])

Array'i analiz etmek için
print(a.shape)
a arrayini analiz eder.Bunun sonucu bir tuple döndürür. Bu da (arrayin eleman sayısı,array boyutu)

yourarrayname.dtype ile array'in veri tipini öğrenebiliriz.

`yourarrayname.ndim` ile array in boyutunu öğrenebiliriz.

`yourarrayname.size` ile array in eleman sayısını verir.

`yourarrayname.itemsize` ise bit cinsinden size verir.

`yourarrayname[index]` ile array in içindeki elamana ulaşabilirsin.

Array leri birbirleriyle çarpabilirsin. Mesela :
`b = a * np.array ([2,0,2])`

Normalde listelere yeni eleman eklemek için `listeninismi.append(eleman)` ile yeni bir şey ekleyebilirsin ayrıca listeye :
`listeismi= listeismi + [eleman]` şeklinde yapabilirsin fakat ilk örnekteki `append` metodu array için çalışmıyor. İkinci örneği

array için yapınca arraydeki tüm elemanlara ekleme yapılıyor ve değerleri değişiyor tüm arrayin.

Listeyi bir sayıyla çarparsan bu listeyi o kadar kez tekrar ettirir.Fakat arraylerde çarparsan her elemanla o değer çarpılır.

Numpy akıllı bir kütüphaneye sahip mesela arrayin karekökü alınmak istenirse arrayismi= np.sqrt(arrayismi) yapılrsa tüm elemanların kareköküne ulaşılır.

Dot Product

Machine Learning ve Data Science ta kullanılıyor

```
  main.py > ...
6      a2 = np.array(l2)
7
8      # dot product
9      dot = 0
10     for i in range(len(l1)):
11         dot += l1[i] * l2[i]
12     print(dot)
13
14     dot = np.dot(a1,a2)
15     print(dot)
16
17     sum1 = a1 * a2
18     dot = np.sum(sum1)
19     print(dot)
20
21
```

Buradaki örnekte liste için değişkenine değer atanıp işlemlerin bu degiskende toplanması sağlandı. Aslında iki vektorun çarpımını gösteriyor.

dot= a1 @a2 olarak da syntax a sahip

Random sayı tutmak için
np.random.randn(hangi sayıya kadar
olabilecegi)

Numpy arrayleri listelerden daha hızlıdır.

Multidimensional Arrays (nd arrays)
Eğer bir dizinin boyutu ve eleman sayısını öğrenmek istiyorsan shape kullanıyorsun.

Eğer bir arrayin transpozesini almak istersen :
arrayismi.T yi print ederek sonuç elde

edersin.

Matrisinin tersini almak istersen:

`np.linalg.inv(arrayismi)`

Matrisin determinantini almak istersen:

`np.linalg.det(arrayismi)`

Diyagonal elemanlarını yazdırınmak istersen:

`print(np.diag(arrayismi))`

Eğer bunu `np.diag(arrayismi)` bir değişkene atayıp sonrasında ise `print(np.diag(c))` yaparsan sadece diyagonal elemanları alır ve geri kalan elemanları sıfır yapar yani birim matris gibi olur .

Slicing Arrays

Örnekleri:

`b= a[row ,column]`

`b=a[0, 1:3]` bu parça 0. satırda bulunan ve 1 den 3. indexe kadar olan elemanları elde eder.

Ayrıca negatifleri de kullanarak arrayin sonundan da ulaşabiliriz.

Ayrıca bir condition oluşturup bunu bir değerde saklayıp bu değere uygunluğu kontrol etmiş olan bir bool arrayi elde edebiliriz.

Eğer şu kodu yazarsan:

`print(a[a>2])` bu sana 2 den büyük elemanları yazdırır.

```
b = np.where(a>2, a, -1)
print(b)
```

2 den büyük eleman olmayanların olduğu yerlere -1 koyar.

```
1 import numpy as np  
2  
3 a = np.array([10,19,30,41,50,61])  
4 print(a)  
5 b = [1,3,5]  
6 print(a[b])
```

Bu kod da sadece 1,3,5inci elemanları alır ve a matrisini o şekilde yazdırır.

```
even= np.argwhere(a%2==0).flatten()  
print(a[even])
```

Mod 2 si sıfır olanları yazdırır.

Reshaping

Belli bir aralıkta array oluşturmak için yapılır.

```
a = np.arange(1,7)
```

1 den 6 ya array oluşturur tek boyutlu

Reshape yaparsan mesela:

```
b= a.reshape ((2,3)) bu elemanları 3  
boyutlu iki elemanlı olarak tekrar dizayn  
eder.
```

Eğer:`b=a[np.newaxis,:]` kullanırsan bu

listeler arrayi ve shape metodu kullanırsan bu artık (1,6) aralığını gösterir.

Newaxis aslında bölüyor,mesela listeyi newaxis in geldiği yerde yeni bir column ya da row açıyor.

Eğer iki arrayi birleştirmek istersen `c=np.concatenate((a,b))` yapabilirsin ve sonuna `(a,b, axis=none)` yaparsan bu elemanları tek bir satırılmış gibi birleştirir.

`hstack` te yeni arrayi sütun olarak ekliyor.`vstack`te arrayi yeni bir satır olarak ekler.

Broadcasting

`x=`bir array

`y=` bir array

`y=x+ a` dersen bu a yı akıllı bir sistemle tüm elemanlara ekler.

Function and Axis

`print(a.sum(axis=None))` yaparsan iki arrayde birleştirdiğin tüm elemanları toplar.

`axis=0` ise tüm elemanları satırı satırla sütunu sütunla toplar.

`axis=1` ise ilk elemanın ve ikinci elemanın kendi toplumlarını yazdırır. (Kendi içlerinde)

```
main.py > ...
1 import numpy as np
2
3 a = np.array([[7,8,9,10,11,12,13], [17,18,19,20,21,22,23]])
4 print(a)
5 print(a.mean(axis=1))
```

`mean` ortalamalarını hesaplar.

`np.std`

Data Types

`x.dtype` ile veri tipini öğrenebilirsin.

Sadece arrayin sonuna `dtype=np.int32` yaparsan direkt veri tipini dönüştürür.

Eğer `copy` fonksiyonu kullanmazsan olacak değişiklik her iki arrayi de etkiler. Fakat `arrayismi.copy()` kullanırsan

orijinal arrayi kopyalamış oluruz.

GENERATING ARRAYS

Eğer np.zeros((2,3)) istenilen boyutlarda bir 0 arrayi oluşturur.

Eğer np.ones((2,3)) kullanırsan istenilen boyutlarda float birlerden oluşan bir array oluşturur.

Eğer arrayismi= np.full((istenilen boyut),5.0) metodu kullanırsan bütün arrayi 5 elemanı olan bir şey oluşturur.

Eğer arrayismi= np.eye(3) dersen 3 e 3 lük bir birim matris oluşturur.

a=np.linspace(0,10,5) 0 dan 10 a kadar 5 eleman olacak şekilde diziyi böler ve oluşturur.her birisinin arasındaki uzaklık aynı

Random bir dizi oluşturmak için

arrayismi= np.random.random((3,2))
random sayılarından oluşan istediğimiz
boyutta bir array oluşturur.

The screenshot shows a Jupyter Notebook interface. At the top, there's a code cell with the following content:

```
5
6 a = np.random.randn(1000) # normal/Gaussian
7 print(a.mean(), a.var())
8
```

Below the code cell is the output pane, which displays the results of the print statement:

```
[0.05810728 0.29909483]
[0.1197065 0.97846745]
[0.21845932 0.35401168]
0.05456624198612931 1.014397402566839
```

Eigen Values

eigenvalues,eigenvectors=np.linalg.eig(arrayismi)

Lineer Sistem Çözümü

x=np.linalg.solve(A,B)

CSV Files

np.loadtxt and np.genfromtxt

data=np.loadtxt('dosyaismi.csv','delimtter=',
'',dtype=np.veritipi)