دوره جامع پایتون:

بخش علوم داده

جلسه سیزدهم

دکتر ذبیح اله ذبیحی

ماژول های داخلی پایتون

- Math
  - شامل توابع ریاضی مختلف
- Re
  - شامل الگوهای مختلف
- Random
  - توابع مختلف تولید اعداد رندوم ،انتخاب رندوم و توزیع های احتمال مختلف
- Statistics
  - انجام آنالیز اماری بر روی داده ها شامل میانگین، میانه،مد، انحراف معیار، واریانس

ماژول های نصبی

کتابخانه numpy

- Numpyمخفف Numerical Python به معنای پایتون عددی یا پایتون محاسباتی
- علم داده، هوش مصنوعی، ماشین لزنینگ
- آرایه ها و ماتریس ها، جبر خطی و حل معادلات خطی، رگرسیون، تبدیل فوریه، اعداد تصادفی، توابع آماری (کمینه، بیشینه، میانگین، میانه، چارک،انحراف معیار، واریانس)
- پایه بسیاری از کتاب خانه های دیگر است.
- آرایه های Numpyمحاسباتی سریع تر از لیست ها دارند و برای اجرای عملیات ریاضیاتی و منطقی بسیار کارآمدتر هستند.

کتابخانه pandas

- پانداس یک کتابخانه قدرتمند برای تجزیه و تحلیل داده‌ها، پیش‌پردازش (PreProcessing) و بصری‌سازی (Visualization) داده‌ها

- پیش‌پردازش داده‌ها مانند ادغام کردن، گروه‌بندی ،الحاق، تمیز کاری
- سری‌های زمانی

- هوش مصنوعی، یادگیری ماشین، یادگیری عمیق ، علم داده

کتابخانه scipy

- انجام محاسبات علمی و مهندسی، تحلیل ها و آنالیزهای یادگیری ماشینی و هوش مصنوعی
- آرایه ها و ماتریس، خوشه بندی داده ها،تبدیل فوریه (پردازش سیگنال و نویز، پردازش تصویر، پردازش سیگنال صوتی)،جبر خطی، پردازش تصویرو سیگنال،الگوریتم های بهینه سازی، درون یابی، حل کننده های معادلات دیفرانسیل معمولی ،

کتابخانه matplotpy

- رسم نمودار و مصور سازی
- رسم خطوط، اشکال دو بعدی، نمودار نقطه ای،ترسیم با رشته ها ، نمودار میله ای،نمودار هیستوگرام،رسم پراکندگی، نمودار دایره ای، نمودار توابع، نمودارهای سه بعدی، کانتور،پوسته های سه بعدی،

کتابخانه seaborn

- رسم انواع چارت هایی چون نمودارهای ماتریسی، نمودارهای شبکه ای (Grid)، نمودارهای رگرسیونی و غیره

کتابخانه pygame

- ساخت بازی

python –V

ورژن پایتون نصبی را نشان می دهد.

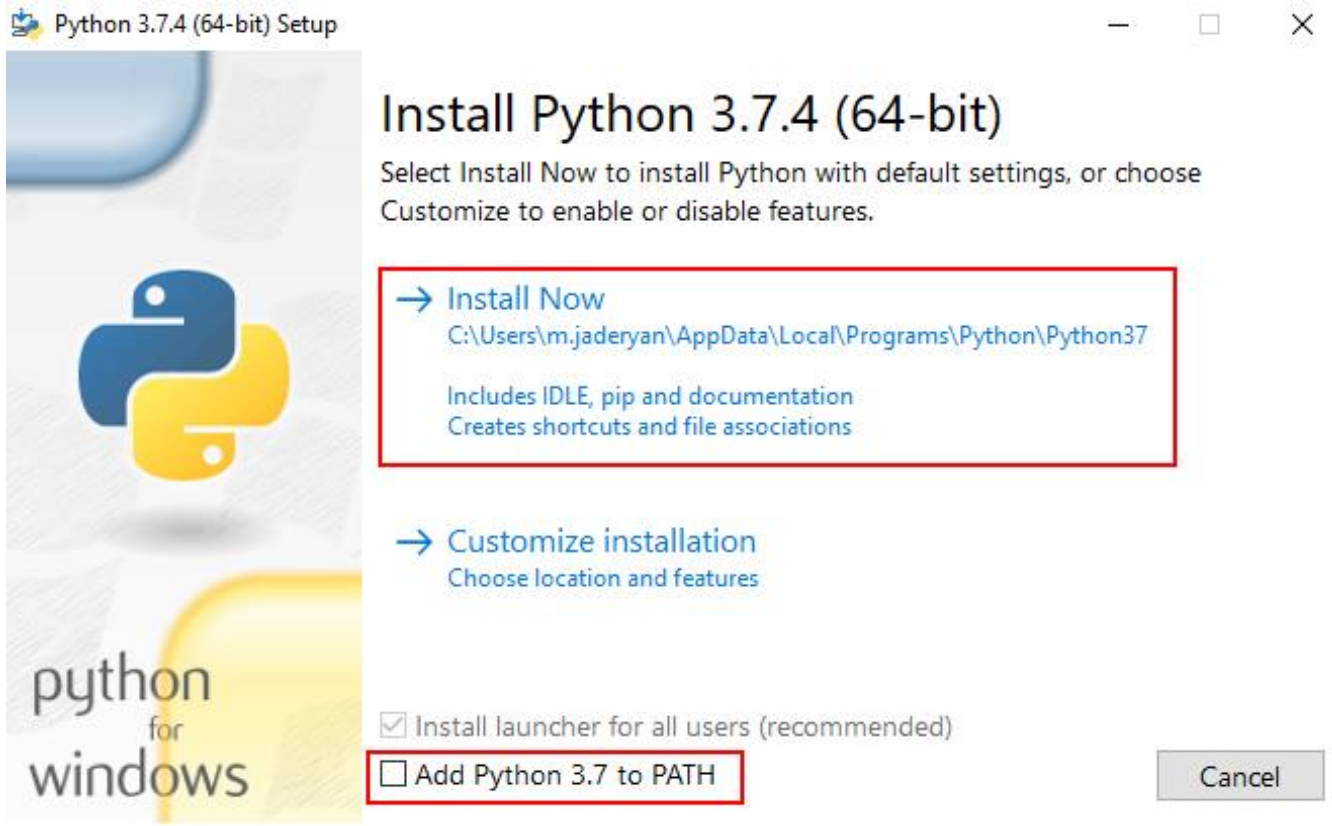
pip -V

محل نصب pip را نشان می دهد.

اگر به جای نشان دادن ورژن وادرس با ارور زیر برخوردید

'pip' is not recognized as an internal or external command,
operable program or batch file.

- اگر با ارور اسلاید قبل برخورد به این معنا هست که در مرحله نصب پایتون تیک مربوط به path را نزدید. بنابرین پایتون را لغو نصب کنید و مجدد نصب کنید و این بار تیک مربوطه را بزنید.



نصب پکیج های(ماژول/کتابخانه های پایتون)

در ویندوز

```
pip install packagename
```

در مک/لینوکس

```
sudo pip install packagename
```

چک کردن ماژول های نصب شده

```
pip freeze
```

فرخوانی اطلاعات کامل ماژول ها

```
pip show packagename
```

به روزرسانی ماژول ها

ویندوز

```
pip install --upgrade packagename
```

مک و لینوکس

```
sudo pip install --upgrade packagename
```

حذف و لغو نصب ماژول

ویندوز

```
pip uninstall packagename
```

مک و لینوکس

```
sudo pip uninstall packagename
```

نصب numpy

- در محیط خط فرمان (cmd) دستور زیر را تایپ کنید:

```
pip install numpy
```

فرخوانی ماژول

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

ورژن ماژول

```
import numpy as np
print(np.__version__)
```

ایجاد آرایه

تبدیل لیست پایتون به آرایه‌ی NumPy با دستور np.array :

```
import numpy as np
data1=[1,2,3,4]
data2=np.array(data1)
print(type(data1))
print(type(data2))
```

آرایه صفر بعدی (0-D)

```
import numpy as np
array1 = np.array(-5)
print(array1)
```

آرایه یک بعدی (1-D)

```
import numpy as np
array1 = np.array([0,12,-2,4,6,-5])
print(array1)
```

آرایه دو بعدی (2-D)

```
import numpy as np
array1 = np.array([[0, 2, -3], [-4, 0, 6]])
print(array1)
```

آرایه سه بعدی (3-D)

```
import numpy as np
array1 = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
print(array1)
```

تعیین ابعاد(رتبه) آرایه با تابع ndim

```
import numpy as np

a = np.array(-12)
b = np.array([0, -2, 3, 1, 5])
c = np.array([[1, 0, 3], [-3, 5, 4]])
d = np.array([[[1, 0, 3], [-4, 5, -6]], [[1, -3, 3], [0, 0, 6]]])

print(a.ndim)
print(b.ndim)
print(c.ndim)
print(d.ndim)
```

آرایه مراتب بالا

```
import numpy as np

array1 = np.array([1, 2, 3, 4], ndmin=7)

print(array1)
print('number of dimensions=', array1.ndim)
```

فراخوانی درایه های آرایه

```
import numpy as np
array1 = np.array([-1, 0, 5, -4,-14])
print(array1[0])
print(array1[3])
print(array1[3]/array1[0])
```

```
import numpy as np

array1 = np.array([[1,-2,0,-1,-5], [6,3,0,9,-1]])

print('2nd element on 1st dim= ', array1[0, 1])

print('5th element on 2nd dim= ', array1[1, 4])
```

```
import numpy as np

array1 = np.array([[1,0,-3,0,5], [-6,2,-8,4,1]])

print('Last element from 2nd dim= ', array1[1, -1])
```

```
import numpy as np

array1 = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])

print(array1[0, 1, 2])
```

```
import numpy as np

array1 = np.array([1, -1, 0, 12, 2, -6, 7])

print(array1[1:5])
```

```
import numpy as np

array1 = np.array([1, -1, 0, 12, 2, -6, 7])

print(array1[:3])
```

```
import numpy as np

array1 = np.array([1, -1, 0, 12, 2, -6, 7])

print(array1[-3:-1])
```

```
import numpy as np
array1 = np.array([1, -1, 0, 12, 2, -6, 7])
print(array1[0:7:2])
```

```
import numpy as np
array1 = np.array([1, -1, 0, 12, 2, -6, 7])
print(array1[: :2])
```

```
import numpy as np
array1= np.array([[0, 2, -1, 3, -5], [0, 7, 3, -9, 0]])
print(array1[1, 1:4])
```

```
import numpy as np
array1= np.array([[0, 2, -1, 3, -5], [0, 7, 3, -9, 0]])
print(array1[0:2, 2])
```

```
import numpy as np
array1= np.array([[0, 2, -1, 3, -5], [0, 7, 3, -9, 0]])
print(array1[0:2, 1:4])
```

دستور shape

- سایز(ابعاد، رتبه) ارایه را بصورت یک تاپل بر می گرداند

```
import numpy as np
array1 = np.array([[-1, 4, 0, 9], [0, 0, -1, 3]])
print(array1.shape)
```

```
import numpy as np
array1 = np.array([1, 0, 2, -4], ndmin=6)
print(array1)
print('shape of array =', array1.shape)
```

تغییر ابعاد آرایه با دستور reshape

```
import numpy as np
array1 = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newarray = array1.reshape(4, 3)
print(newarray)
```

```
import numpy as np
array1 = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newarray = array1.reshape(2, 3, 2)
print(newarray)
```

آرایه به عنوان شمارنده حلقه for

```
import numpy as np
array1 = np.array([0, 2, 4])
for i in array1:
  print(i)
```

```
import numpy as np
array1 = np.array([[3, 2, 1], [-1, -2, -3]])
for i in array1:
  print(i)
```

```
import numpy as np
array1 = np.array([[3, 2, 1], [-1, -2, -3]])
for i in array1:
  for j in i:
    print(j)
```

```
import numpy as np
array1 = np.array([[[0, 2, 4], [1, 3, 7]], [[-1, -2, -3], [-4, -5, -6]]])
for i in array1:
  print(i)
```

```
import numpy as np
array1 = np.array([[[0, 2, 4], [1, 3, 7]], [[-1, -2, -3], [-4, -5, -6]]])
for i in array1:
  for j in i:
    for k in j:
      print(k)
```

```
import numpy as np
array1 = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])
for i in np.nditer(array1):
  print(i)
```

ترکیب و الحاق دو آرایه

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array(["ali", "reza", "neda"])
array = np.concatenate((array1, array2))
print(array)
```

```
import numpy as np
array1 = np.array([[1, 2], [3, 4]])
array2 = np.array([[5, 6], [7, 8]])
array = np.concatenate((array1, array2), axis=0)
print(array)
```

```
import numpy as np
array1 = np.array([[1, 2], [3, 4]])
array2 = np.array([[5, 6], [7, 8]])
array = np.concatenate((array1, array2), axis=1)
print(array)
```

تابع concatenate دو ارایه را در راستای محور مشخص بهم متصل می کند.

اتصال آرایه ها با تابع stack()

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
array = np.stack((array1, array2), axis=0)
print(array)
```

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
array = np.stack((array1, array2), axis=1)
print(array)
```

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
array = np.hstack((array1, array2))
print(array)
```

تابع hstack آرایه ها را بصورت افقی متصل می کند.

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
array = np.vstack((array1, array2))
print(array)
```

تابع Vstack آرایه ها را بطور عمودی متصل می کند.

```
import numpy as np
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
array = np.dstack((array1, array2))
print(array)
```

- تابع column_stack دو ارایه یک بعدی را بصورت ستونی بهم متصل می کند.

جدا کردن آرایه ها

```
import numpy as np
array1 = np.array([1, 2, 3, 4, 5, 6])
newarray = np.array_split(array1, 3)
print(newarray)
print(newarray[0])
print(newarray[1])
print(newarray[2])
```

```
import numpy as np
array1 = np.array([[1, 1], [2, 2], [3, 3], [4, 4], [5, 5], [6, 6]])
newarray = np.array_split(array1, 3)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.array_split(array1, 3)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.array_split(array1, 3, axis=0)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.array_split(array1, 3, axis=1)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.hsplit(array1, 3)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.vsplit(array1, 3)
print(newarray)
```

```
import numpy as np
array1 = np.array([[1, 1, 1], [2, 2, 2], [3, 3, 3], [4, 4, 4], [5, 5, 5], [6, 6, 6]])
newarray = np.hsplit(array1, 3)
print(newarray)
```

- تابع hsplit ارایه را به شکل افقی (سطری) جدا می کند.
- تابع Vsplit ارایه را به شکل عمودی (ستونی) جدا می کند

پیدا کردن درایه ها از آرایه

```
import numpy as np
array1 = np.array([1, 2, 3, 2, 5, 2, 2])
x = np.where(array1 == 2)
print(x)
```

```
import numpy as np
array1 = np.array([1, 2, 3, 4, 6, 7, 8, 9])
x = np.where(array1%3 == 0)
print(x)
```

تابع sort()

```
import numpy as np
array1 = np.array([4, -1, 0, 6])
print(np.sort(array1))
```

```
import numpy as np
array1 = np.array(['reza', 'yasin', 'ali'])
print(np.sort(array1))
```

```
import numpy as np
array1 = np.array([[3, 2, 4], [5, 0, 1]])
print(np.sort(array1))
```

عملیات ریاضی بین آرایه با آرایه

```
import numpy as np
data1=[1,2,3,4]
data2=[2,1,-1,5]
array1=np.array(data1)
array2=np.array(data2)
s1=array1+array2
s2=array1-array2
s3=array1*array2
s4=array1/array2
s5=array1%array2
print(array1,array2)
print(s1)
print(s2)
print(s3)
print(s4)
print(s5)
```

- علائم +،-،*،/ به ترتیب جمع، تفریق، ضرب و تقسیم بین درایه های مشابه دو آرایه را را انجام می دهد.

ضرب برداری با تابع dot()

```
import numpy as np
data1=[1,2,3,4]
data2=[2,1,-1,5]
array1=np.array(data1)
array2=np.array(data2)
s=np.dot(array1,array2)
print(s)
```

ضرب در ماتریس ها

- علامت * و تابع multiply ضرب درایه با درایه را انجام میدهند
- تابع dot و matmul ضرب برداری را انجام میدهند

مثال

```
import numpy as np
a=np.array([[1,0],[2,0]])
b=np.array([[1,0],[0,1]])
print("a=",a)
print("b=",b)
s1=a*b
print("s1=",s1)
s2=np.dot(a,b)
print("s2=",s2)
s3=np.multiply(a,b)
print("s3=",s3)
s4=np.matmul(a,b)
print("s4=",s4)
```

- ترانهاده ماتریس (تعویض سطر و ستون)

```
import numpy as np
array1 = np.array([[0, 2, -3], [-4, 0, 6]])
x=np.transpose(array1)
print(array1)
print(x)
```

تعیین مرتبه ماتریس

با تابع linalg.matrix_rank() می توان مرتبه یک ماتریس را تعیین کرد

```
import numpy as np
A = np.array([[2, 1, 0],
              [5, -1, 5]])

print( np.linalg.matrix_rank(A))
```

رد ماتریس

- رد یا اثر ماتریس از طریق تابع trace() تعیین می گردد

```
import numpy as np
A = np.array([[2, 1, 0],
              [5, -1, 5],
              [0, 2, -2]])
print( np.trace(A))
```

دترمینان ماتریس

دترمینان یک ماتریس با تابع linalg.det() تعیین می گردد

```
import numpy as np
A = np.array([[2, 1, 0],
              [5, -1, 5],
              [0, 2, -2]])
print( np.linalg.det(A))
```

معکوس ماتریس

معکوس ماتریس با تابع linalg.inv() محاسبه می گردد.

```
import numpy as np
A = np.array([[2, 1, 0],
              [5, -1, 5],
              [0, 2, -2]])
print( np.linalg.inv(A))
```

به توان رساندن ماتریس

با تابع linalg.matrix_power(A, n) میتوان ماتریس A را به توان n رساند.

```
import numpy as np
A = np.array([[2, 1, 0],
              [5, -1, 5],
              [0, 2, -2]])
print( np.linalg.matrix_power(A, 3))
```

توابع سرتاسری (unfun)

توابع سرتاسری مثل Sqrt ،sin و ....بر روی هر درایه ماتریس عمل می کنند.

```
import numpy as np
a=np.array([[2,4],[9,25]])
c=np.sqrt(a)
print(c)
```

ایجاد ماتریس های خاص

- ماتریس با درایه های صفر
- zeros(shape, dtype=float, order='C')
- shape ابعاد ماتریس، dtype نوع داده ماتریس (اختیاری) و order استایل پایه سطرها(اختیاری)
- ماتریس با درایه های یک
- ones(shape, dtype=float, order='C')
- ماتریس با درایه های یکسان n
- full(shape,n)
- ماتریس همانی
- eye(shape)
- ماتریس با درایه های تصادفی
- Random.random(shape)

مثال

```
import numpy as np
x=np.zeros((2,3))
print (x)
print("---------------")
y=np.ones((2,3))
print(y)
print("---------------")
z=np.full((2,3),3)
print(z)
print("---------------")
e=np.eye(2,3)
print(e)
print("---------------")
f=np.random.random((2,3))
print(f)
```

تولید اعداد رندوم

```
from numpy import random
x = random.randint(10)
print(x)
```

```
from numpy import random
x = random.rand()
print(x)
```

```
from numpy import random
x = random.rand()
print(x)
```

```
from numpy import random
x=random.randint(100, size=(10))
print(x)
```

```
from numpy import random
x = random.randint(25, size=(3, 5))
print(x)
```

```
from numpy import random
x = random.rand(5)
print(x)
```

```
from numpy import random
x = random.rand(2, 3)
print(x)
```

```
from numpy import random
x = random.choice([1, 2, 3, 4,5,6])
print(x)
```

```
from numpy import random
x = random.choice([1, 2, 3, 4], size=(3, 5))
print(x)
```

```
from numpy import random
x = random.choice(["ali", "reza", "neda", "sasan"], p=[0.1, 0.3, 0.6, 0.0], size=(10))
print(x)
```

```
from numpy import random
x = random.choice(["ali", "reza", "neda", "sasan"], p=[0.1, 0.3, 0.6, 0.0], size=(3, 5))
print(x)
```

```
from numpy import random
import numpy as np
array1 = np.array(["ali", "reza", "neda", "sasan", "mahdi"])
random.shuffle(array1)
print(array1)
```

```
from numpy import random
import numpy as np
array1 = np.array(["ali", "reza", "neda", "sasan", "mahdi"])
print(random.permutation(array1))
```

تمرین

- با تابع stack دو ارایه دو بعدی را به هم الحاق کنید.