Д. Серікбаев атындағы ШҚТУ

Алгоритмдеу және бағдарламалау тілдері (Python)

PYTHON ЖӘНЕ ДЕРЕКТЕР ҒЫЛЫМЫ

Дәріс 10

Оқытушы:
Жомартқызы Г.

Өскемен
2024

NumPy — Python кітапханасы

NumPy дегеніміз не?
NumPy - массивтермен жұмыс істеу үшін қолданылатын Python кітапханасы.
NumPy ("Numerical Python") «Сандық Python» сөзінің қысқартылған нұсқасы.

Сондай-ақ оның сызықтық алгебра, Фурье түрлендіру және матрицаларда жұмыс істеу функциялары бар.
NumPy 2005 жылы Травис Олифант құрған.

NumPy мақсаты - дәстүрлі Python тізімдерінен 50 есе жылдамырақ массив объектісін қамтамасыз ету.

Массивтер жылдамдық пен ресурстар өте маңызды болатын деректер ғылымында жиі қолданылады.

NumPy кітапханасын қалай орнатуға болады?
Қысқаша PIP (пакеттерді басқару жүйесі) арқылы

Pip — Python бағдарламалық пакеттерін басқару үшін орнатылуы керек пакеттерді басқару жүйесі.

How to Install Numpy on Python 3.11.2 on Windows (Complete Guide)
https://m.youtube.com/watch?v=K87M0sMVXZE&pp=ygUdd2luZG93cyBweXRob24gIGluc3RhbGwgbnVtcHk%3D

cmd ортасында келесі команданы теремiз:

py -m pip install numpy

Орнату нәтижесі:

```
  Consider adding this directory to PATH
Successfully installed numpy-1.26.4
```

NumPy кітапханасын қолдану

Импорттау кезінде as кілттік сөзімен қысқаша жазу:

```
import numpy as np
```

NumPy пакетіне енді numpy орнына np атауын қолдануға болады.

Массивтерді құру (Бір өлшемді массивтер)

NumPy-дегі массив объектісі ndarray деп аталады.
array() функциясын пайдаланып, NumPy массив объектісі ndarray құрай аламыз.

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

```
[1 2 3 4 5]
```

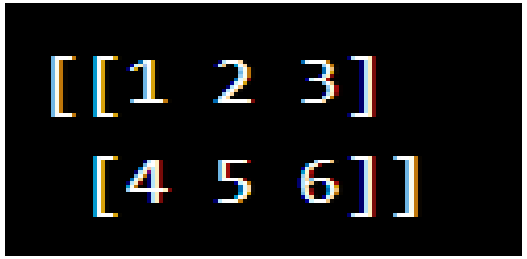
NumPy кітапханасын қолдану

2-D массивы

Элементтері бір өлшемді массивтер болатын массив екі өлшемді массив деп аталады.

Мысал.
1,2,3 және 4,5,6 мәндері бар екі массивтен тұратын екі өлшемді массив жасаңыз:

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



NumPy кітапханасын қолдану

Массив элементтеріне қол жеткізу

Массив элементіне оның реттік нөмірі арқылы қол жеткізуге болады.
NumPy массивтеріндегі индекстер 0-ден басталады, яғни бірінші элементте - 0 индекс, екіншісінде - 1 индекс және т.б.

Мысал. Келесі массивтен бірінші элементті алыңыз:

```
import numpy as np
arr = np.array([100, 200, 300, 400])
print(arr[0])
```

Нәтижесі:

```
100
```

Мысал. Келесі массивтен үшінші және төртінші элементтерді алыңыз және оларды қосыңыз есептеңіз.

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr[2] + arr[3])
```

Нәтижесі:

```
7
```

NumPy кітапханасын қолдану

Массивтерді кесу. Python тіліндегі кесу - элементтерді бір индекстен екінші берілген индексте алуды білдіреді.

кесіндіні келесідей анықтаймыз: `[start:end]`

Мысал. Келесі массивтен 1 индексінен 5 индексіне дейінгі элементтерді кесіңіз:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[1:5])
```

Нәтижесі:

```
[2 3 4 5]
```

Мысал. Элементтерді 4 индексінен массивтің соңына дейін кесіңіз:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[4:])
```

Нәтижесі:

```
[5 6 7]
```

Мысал. Элементтерді басынан бастап 4 индексіне дейін кесіңіз (қосылмаған):

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[:4])
```

Нәтижесі:

```
[1 2 3 4]
```

NumPy кітапханасын қолдану

Массивтерді біріктіру.

Біріктіру екі немесе одан да көп массивтердің мазмұнын бір массивке қоюды білдіреді. `concatenate()` функциясы қолданылады.

Мысал. Келесі 2 массивтті қосу:

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.concatenate((arr1, arr2))
print(arr)
```

Нәтижесі:

```
[1 2 3 4 5 6]
```

Массивтерге бөлу.

array_split() әдісінің қайтару мәні - әрбір бөлу массив ретінде қамтитын массив болып табылады.

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

Нәтижесі:

```
[1 2]
[3 4]
[5 6]
```

NumPy кітапханасын қолдану

Массивте іздеу

Массивте белгілі бір мәнді іздеуге және нәтижесінде сәйкестікті табатын индекстерді қайтаруға болады. Массивте іздеу үшін where() әдісін пайдаланыңыз.

Мысал. Мәні 4 болатын индекстерді табыңыз:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
```

Нәтижесі:

```
(array([3, 5, 6]),)
```

Бұл 4 мәні *3, 5* және *6* индекстерінде бар екенін білдіреді.

Мысал. Мәндері жұп болатын индекстерді табыңыз:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])
x = np.where(arr%2 == 0)
print(x)
```

Нәтижесі:

```
(array([1, 3, 5, 7]),)
```

NumPy кітапханасын қолдану

Массивте іздеу

Мысал. Мәндері тақ болатын индекстерді табыңыз:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])
x = np.where(arr%2 == 1)
print(x)
```

Нәтижесі:

```
(array([0, 2, 4, 6]),)
```

NumPy кітапханасын қолдану

NumPy массивтерін сұрыптау

Сұрыптау элементтерді реттелген реттілікпен орналастыруды білдіреді.

Мысал. Массивті сұрыптау:

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

Нәтижесі:

```
[0 1 2 3]
```

NumPy кітапханасын қолдану

Кездейсоқ массив құру

randint() әдісі - массивтің формасын көрсетуге болатын size параметрін қабылдайды.

Мысал. 0-ден 100-ге дейінгі 5 кездейсоқ бүтін саннан тұратын бір өлшемді массив жасаңыз:

```
from numpy import random
x=random.randint(100, size=(5))
print(x)
```

Нәтижесі:

```
[47 4 98 69 61]
```

Мысал. Әр жолда 0-ден 100-ге дейінгі 5 кездейсоқ бүтін сандардан тұратын үш жолдан тұратын екі өлшемді массив жасаңыз:

```
from numpy import random
x = random.randint(100, size=(3, 5))
print(x)
```

Нәтижесі:

```
[[80 54 19 74 65]
 [26 60 69 34 25]
 [50 16 53 84 90]]
```

NumPy кітапханасын қолдану

Кездейсоқ массив құру

Мысал. Кездейсоқ 5 нақты сандардан тұратын бар бір өлшемді массивты құрыңыз:

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

Нәтижесі:

```
[0.5898409 0.7300066 0.3573777 0.1211301 0.5760398]
```

NumPy кітапханасын қолдану

Қалыпты таратылу (Нормальное распределение)

Қалыпты таратылу ең маңызды таратудың бірі болып табылады.
Оны неміс математигі Карл Фридрих Гаусстың атымен Гаусс үлестірімі деп те атайды.

Ол көптеген оқиғалардың ықтималдық таралуын сипаттайдя, мысалы. IQ ұпайлары, жүрек соғу жиілігі, т.б.

Деректердің қалыпты таралуын алу үшін осы random.normal() әдісі қолданылады.

Оның үш параметрі бар:
loc- (мағынасы) график төбесі орналасқан жер.
scale- (стандартное отклонение, стандартты ауытқу) тарату графигі қаншалықты тегіс болуын сипааттайды.
size- қайтарылған массивтің формасы.

NumPy кітапханасын қолдану

Мысал. Кездейсоқ қалыпты таралуды құру керек: массив формасы (өлшемі)-2x3, loc (орташа мәні) – 1, стандартты ауытқу -2

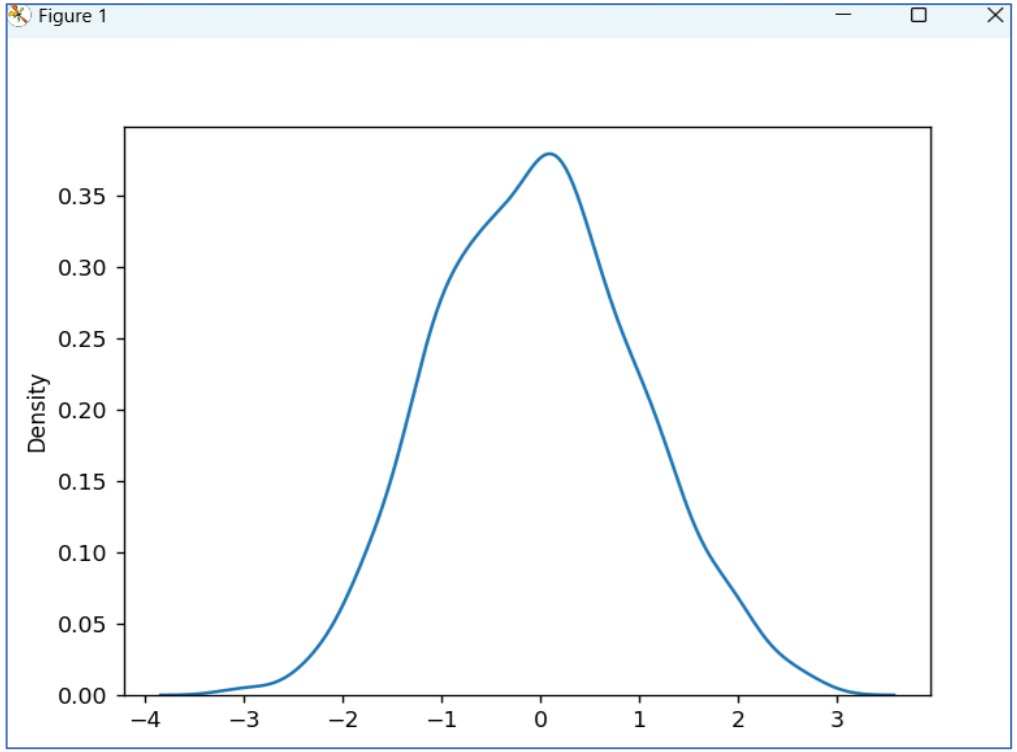
```
from numpy import random
x = random.normal(loc=1, scale=2, size=(2, 3))
print(x)
```

Нәтижесі:

```
[[1.42584767 5.10437117 1.80184243]
 [0.21754134 1.35559186 2.79806085]]
```

Қалыпты таратылу визуализациясы

```
from numpy import random
import matplotlib.pyplot as plt
import seaborn as sns
sns.distplot(random.normal(size=1000), hist=False)
plt.show()
```



Matplotlib кітапханасы

Matplotlib - бұл визуализация утилитасы ретінде қызмет ететін төмен деңгейлі Python графикалық кітапханасы.

Seaborn - бұл графикаларды салу үшін мастында Matplotlib пайдаланатын кітапхана. Ол кездейсоқ үлестірімдерді визуализациялау үшін пайдаланылады.

py -m pip install seaborn
py -m pip install matplotlib