Not Defteri **Python Programlama**

Matrislere giriş ve Python'da matrisler

Bu yazıda matrislere ilişkin temel bilgileri aktarıyor ve konuyla ilgili bazı Python uygulamaları gerçekleştiriyorum.

Nehir Günce Daşcı • 07 NİS 2021

Matris kısaca sayıların satırlar ve sütunlar üzerinde düzenlenmiş halidir. Gözlem sayısının n, değişken sayısının p ile ifade edildiği matrisin boyutu nxp olarak belirtilir.

Python'da matrisler için yerleşik tür yoktur. Fakat iç içe liste yöntemiyle basit hesaplamalar yapılabilir ya da NumPy paketi ile matrislerle etkili bir biçimde çalışılabilir. Liste, aslında tek boyutlu bir veri yapısıdır. İki boyutlu listeler matris olarak bilinir.

Örnek 1:

```
A = [[1, 4, 5],   # iç içe liste
     [-5, 8, 9],
     [6, 8, 10],
     [0, 2, 38]]


print("A =", A)
print("A[1] =", A[1])      # 2.satır
print("A[1][2] =", A[1][2])   # 2.satırın 3.elemanı
print("A[0][-1] =", A[0][-1])  # İlk satırın sonuncu elemanı
```

```
A = [[1, 4, 5], [-5, 8, 9], [6, 8, 10], [0, 2, 38]]
A[1] = [-5, 8, 9]
A[1][2] = 9
A[0][-1] = 5
```

Örnek 2:

```
import numpy as np


a = np.array([[1, 4, 5],   # numpy array
              [-5, 8, 9],
              [6, 8, 10],
              [0, 2, 38]])


print("type:", type(a))
print("a =","\n",a, "\n")
print("a[1] =", a[1])      # 2.satır
print("a[1][2] =", a[1][2])   # 2.satırın 3.elemanı
print("a[0][-1] =", a[0][-1])  # İlk satırın sonuncu elemanı
```

```
type: <class 'numpy.ndarray'>
a =
 [[ 1  4  5]
 [-5  8  9]
 [ 6  8 10]
 [ 0  2 38]]

a[1] = [-5  8  9]
a[1][2] = 9
a[0][-1] = 5
```

```
for i in a:
    for j in i:
        print("{:>5}".format(j),end="")
    print()
```

```
    1    4    5
   -5    8    9
    6    8   10
    0    2   38
```

a matrisi

Matris Türleri

Farklı matris çeşitleri bulunmaktadır. Kare matriste satır sayısı (n) ile sütun sayısı (p) birbirine eşittir.

```
np.array(range(9)).reshape(3,3)
```

```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

3x3 boyutlu kare matris

Sıfır matrisin ise tüm elemanları 0'dır.

```
np.zeros((6,6), dtype="int")
```

```
array([[0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0]])
```

Not: Default çıktı float olarak dönüyor, integer'a çevirdim.

Sadece 1 satırı olan matrise satır matrisi, sadece 1 sütunu olan matrise sütun matrisi denir.

```
print(np.array([[1, 2, 3]]))
print("shape:", np.array([[1, 2, 3]]).shape)
```

```
[[1 2 3]]
shape: (1, 3)
```

Satır matrisi: 1x3 boyutlu, satır sayısı 1

```
print(np.array([[1],[2],[3]]))
print("shape:", np.array([[1],[2],[3]]).shape)
```

```
[[1]
 [2]
 [3]]
shape: (3, 1)
```

Sütun matrisi: 3x1 boyutlu, sütun sayısı 1

Köşegen elemanları dışındaki tüm elemanları 0 olan matris ise köşegen matris olarak bilinir. Sadece kare matrisler, köşegen matris olabilir.

```
np.diag(np.diag(np.array(range(1,26)).reshape(5,5)))
```

```
array([[ 1,  0,  0,  0,  0],
       [ 0,  7,  0,  0,  0],
       [ 0,  0, 13,  0,  0],
       [ 0,  0,  0, 19,  0],
       [ 0,  0,  0,  0, 25]])
```

Matris kare iken (yani n=p iken), köşegen elemanların 1 kalan elemanların 0 olduğu matris ise birim matristir.

```
np.identity(7, dtype = int)
```

```
array([[1, 0, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 1, 0, 0, 0],
       [0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 1]])
```

Esas köşegen üzerindeki elemanları aynı olup kalan elemanları 0 olan matris ise skaler matris olarak isimlendirilir. Birim matrisin bir skaler ile çarpılması sonucu elde edilir.

```
np.identity(7, dtype = int)*22
```

```
array([[22,  0,  0,  0,  0,  0,  0],
       [ 0, 22,  0,  0,  0,  0,  0],
       [ 0,  0, 22,  0,  0,  0,  0],
       [ 0,  0,  0, 22,  0,  0,  0],
       [ 0,  0,  0,  0, 22,  0,  0],
       [ 0,  0,  0,  0,  0, 22,  0],
       [ 0,  0,  0,  0,  0,  0, 22]])
```

Bir matrisin (örneğin A matrisi) satırları ile sütunlarının yer değiştirmesi sonucu oluşan matris ise transpose (devrik) matristir. A^T ile gösterilir.

Kolay yoldan devrik matris oluşturma:

```
X = np.array(range(12)).reshape(4,3)  # 4x3 lük matris
X
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
X.T
```

```
array([[ 0,  3,  6,  9],
       [ 1,  4,  7, 10],
       [ 2,  5,  8, 11]])
```

Döngü yardımıyla devrik matris oluşturma:

```
X = np.array(range(12)).reshape(4,3)  # 4x3 lük matris
X
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
zeros=np.zeros((3,4), dtype="int")  # 3x4'lük matris oluşturuldu, X'in transpose'u buraya eklenecek
```

```
for i in range(len(X)):
    for j in range(len(X[0])):
        zeros[j][i]=X[i][j]
print(zeros)
```

```
[[ 0  3  6  9]
 [ 1  4  7 10]
 [ 2  5  8 11]]
```

List comprehension ile devrik matris oluşturma:

```
X = np.array(range(12)).reshape(4,3)  # 4x3 lük matris
X
```

```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
zeros=np.zeros((3,4), dtype="int")  # 3x4'lük matris oluşturuldu, X'in transpose'u buraya eklenecek
```

```
for i in [ [X[i][j] for i in range(len(X))] for j in range(len(X[0]))]:
    print(i)
```

```
[0, 3, 6, 9]
[1, 4, 7, 10]
[2, 5, 8, 11]
```

Matrislerde toplama ve çıkarma işlemleri

Matrislerde toplama çıkarma işlemleri için ilgili matrislerin boyutlarının aynı olması gerekir.

```
m1=np.random.randint(20, size=(5, 3))
m1
```

```
array([[14,  9,  0],
       [ 4,  2, 16],
       [10, 15, 19],
       [ 9,  0,  5],
       [13,  6, 16]])
```

```
m2=np.random.randint(5,size=(5, 3))
m2
```

```
array([[2, 4, 0],
       [1, 3, 4],
       [2, 0, 2],
       [2, 4, 0],
       [4, 0, 3]])
```

```
m1+m2
```

```
array([[16, 13,  0],
       [ 5,  5, 20],
       [12, 15, 21],
       [11,  4,  5],
       [17,  6, 19]])
```

```
m1-m2
```

```
array([[12,  5,  0],
       [ 3, -1, 12],
       [ 8, 15, 17],
       [ 7, -4,  5],
       [ 9,  6, 13]])
```

Matrislerde çarpma işlemi

Matrisin belirli bir sayı ile çarpımında matristeki her bir eleman sabit sayı ile çarpılır.

```
m1
```

```
array([[14,  9,  0],
       [ 4,  2, 16],
       [10, 15, 19],
       [ 9,  0,  5],
       [13,  6, 16]])
```

```
m1*10
```

```
array([[140,  90,   0],
       [ 40,  20, 160],
       [100, 150, 190],
       [ 90,   0,  50],
       [130,  60, 160]])
```

Matrisin matris ile çarpımı ise klasik çarpma işleminden farklıdır ve görece daha karmaşıktır. İlk matrisin sütun sayısının ikinci matrisin satır sayısına eşit olması gerekir. Ortaya çıkan yeni matrisin satır sayısı ilk matrisin satır sayısına; sütun sayısı ikinci matrisin sütun sayısına eşittir. Python'da bu işlem döngü ile yapılır.

```
a1=np.random.randint(5, size=(5, 3))
a1
```

```
array([[0, 1, 3],
       [4, 3, 2],
       [3, 1, 3],
       [4, 1, 2],
       [3, 0, 1]])
```

```
a2=np.random.randint(2,size=(3, 4))
a2
```

```
array([[1, 0, 0, 1],
       [0, 1, 1, 1],
       [0, 0, 0, 1]])
```

```
result= np.zeros((5,4))

for i in range(a1.shape[0]):  #5'e kadar
    for j in range(a2.shape[1]):   #4'e kadar
        for k in range(a2.shape[0]):   #3'e kadar
            result[i][j] += a1[i,k] * a2[k, j]
```

```
result
```

```
array([[0., 1., 1., 4.],
       [4., 3., 3., 9.],
       [3., 1., 1., 7.],
       [4., 1., 1., 7.],
       [3., 0., 0., 4.]])
```

- 1. Adım: İlk matrisin **ilk satırı** ile ikinci matrisin **ilk sütunu** birebir eşlenerek çarpılır, bu çarpımlar toplanır, sonuç matrisinde result[0][0]'lık kısma yazılır.
- 2. Adım: İlk matrisin **ilk satırı** ile ikinci matrisin **ikinci sütunu** birebir eşlenerek çarpılır, bu çarpımlar toplanır, sonuç matrisinde result[0][1]'lık kısma yazılır.
- 3. Adım: İlk matrisin **ilk satırı** ile ikinci matrisin **üçüncü sütunu** birebir eşlenerek

çarpılır, bu çarpımlar toplanır, sonuç matrisinde result[0][2]'lık kısma yazılır.

- 4. Adım: İlk matrisin **ilk satırı** ile ikinci matrisin **dördüncü sütunu** birebir eşlenerek çarpılır, bu çarpımlar toplanır, sonuç matrisinde result[0][3]'lık kısma yazılır.
- 5. Adım: İlk arrayin ilk satırı için işlemleri tamamlandı. Bu dört adım ilk arrayin her bir satırı için yapılır.

Matrislerin bazı özellikleri

A, B,C birer matris ve r, s birer sabit sayı ve I'lar birim matris iken;

$$A+B = B+A$$
$$A+(B+C) = (A+B)+C$$
$$r(sA) = (rs)A$$
$$(r+s)A = rA+sA$$
$$r(A+B) = rA+rB$$
$$A(rB) = r(AB) = (rA)B$$
$$A(BC) = (AB)C$$
$$A(B+C) = AB+AC$$
$$(A+B)C = AC+BC$$
$$I_m A = A I_n = A$$
$$(A^T)^T = A$$
$$(A+B)^T = A^T+B^T$$
$$(AB)^T = B^T A^T$$
$$(rA)^T = rA^T$$

Matrisin tersi

A üzeri -1 ile gösterilir.

$$A \times A^{-1} = A^{-1} \times A = \mathbf{I}$$

A matrisinin tersi ile sağdan ya da soldan çarpımı birim matrise eşittir. Bir matrisin tersinin alınabilmesi için kare matris olması

şarttır.

2x2 boyutlu matrislerde hesaplanması kolaydır.

```
x1= np.random.randint(5, size=(2, 2))
x1
```

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

```
np.linalg.inv(x1)
```

```
array([[ 1.5, -0.5],
       [-0.5,  0.5]])
```

Determinant: Köşegen elemanların birbirleri ile çarpımından kalan elemanların birbirleri ile çarpımlarının çıkarılmasıdır. (1x3)-(1x1)

Tersi alınırken:

- Köşegen elemanlar yer ve işaret değiştirir.
- Her bir eleman determinanta bölünür.

Bu sebeple, matrisin determinantı 0'a eşitse, matrisin tersi yoktur. 2'den fazla boyutlar için pythonda direkt np.linalg.inv() komutunu kullanabilirsiniz.

```
x= np.random.randint(10, size=(5, 5))
x
```

```
array([[3, 0, 1, 2, 1],
       [7, 1, 1, 8, 2],
       [5, 0, 6, 9, 1],
       [9, 5, 1, 0, 3],
       [8, 0, 8, 4, 5]])
```

```
np.linalg.inv(x)
```

```
array([[ 1.14099783, -0.27657267,  0.06507592,  0.05531453, -0.1637744 ],
       [-1.17353579,  0.18655098,  0.07375271,  0.1626898 ,  0.04772234],
       [ 0.02386117, -0.20065076,  0.164859  ,  0.04013015,  0.01843818],
       [-0.48590022,  0.22234273,  0.00650759, -0.04446855,  0.03362256],
       [-1.47505423,  0.5856833 , -0.37310195, -0.11713666,  0.40563991]])
```

Python hakkında daha geniş kapsamlı bilgiye erişmek ve kariyerinizde Python

bilginizle fark yaratmak isterseniz Miuul'un sunduğu Veri Bilimi için Python Programlama eğitimine göz atabilirsiniz.

Kaynaklar

- Programiz, Python Matrices and NumPy Arrays
- Itcconline, Properties of Matrix Operations
- Reha Alpar, Çok Değişkenli İstatistiksel Yöntemler

Etiketler

matris / istatistik

Nehir Günce Daşcı

İlginizi Çekebilir

Merkezi eğilim ölçüleri nelerdir?

Çağla Öztürk Zan

Ortalama ve medyan arasında ne fark var?

Çağla Öztürk Zan