# NumPy - Funções Estatísticas

NumPy tem algumas funções estatísticas úteis para encontrar mínimo, máximo, desvio padrão percentil e variância, etc., dos elementos fornecidos na matriz. As funções são explicadas a seguir -

## numpy.amin() e numpy.amax()

Essas funções retornam o mínimo e o máximo dos elementos de uma determinada matriz ao longo do eixo especificado.

### Exemplo

Demonstração ao vivo

```
import numpy as np
a = np.array([[3,7,5],[8,4,3],[2,4,9]])

print 'Our array is:'
print a
print '\n'

print 'Applying amin() function:'
print np.amin(a,1)
print '\n'

print 'Applying amin() function again:'
print np.amin(a,0)
print '\n'

print 'Applying amax() function:'
print np.amax(a)
print '\n'

print 'Applying amax() function again:'
print np.amax(a, axis = 0)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[3 7 5]
[8 4 3]
```

```
[2 4 9]]

Applying amin() function:
[3 3 2]

Applying amin() function again:
[2 4 3]

Applying amax() function:
9

Applying amax() function again:
[8 7 9]
```

## numpy.ptp()

A função **numpy.ptp()** retorna o intervalo (máximo-mínimo) de valores ao longo de um eixo.

Demonstração ao vivo

```
import numpy as np
a = np.array([[3,7,5],[8,4,3],[2,4,9]])

print 'Our array is:'
print a
print '\n'

print 'Applying ptp() function:'
print np.ptp(a)
print '\n'

print 'Applying ptp() function along axis 1:'
print np.ptp(a, axis = 1)
print '\n'

print 'Applying ptp() function along axis 0:'
print np.ptp(a, axis = 0)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[3 7 5]
[8 4 3]
[2 4 9]]
```

```
Applying ptp() function:
7

Applying ptp() function along axis 1:
[4 5 7]

Applying ptp() function along axis 0:
[6 3 6]
```

## numpy.percentil()

Percentil (ou percentil) é uma medida usada em estatísticas que indica o valor abaixo do qual cai uma determinada porcentagem de observações em um grupo de observações. A função **numpy.percentile()** recebe os seguintes argumentos.

```
numpy.percentile(a, q, axis)
```

Onde,

| Sr. Não. | Argumento e Descrição |
|---|---|
| 1 | **a**<br>Matriz de entrada |
| 2 | **q**<br>O percentil a ser calculado deve estar entre 0 e 100 |
| 3 | **eixo**<br>O eixo ao longo do qual o percentil deve ser calculado |

## Exemplo

Demonstração ao vivo

```
import numpy as np
a = np.array([[30,40,70],[80,20,10],[50,90,60]])

print 'Our array is:'
print a
print '\n'

print 'Applying percentile() function:'
```

```
print np.percentile(a,50)
print '\n'

print 'Applying percentile() function along axis 1:'
print np.percentile(a,50, axis = 1)
print '\n'

print 'Applying percentile() function along axis 0:'
print np.percentile(a,50, axis = 0)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[30 40 70]
 [80 20 10]
 [50 90 60]]

Applying percentile() function:
50.0

Applying percentile() function along axis 1:
[ 40. 20. 60.]

Applying percentile() function along axis 0:
[ 50. 40. 60.]
```

## numpy.median()

**A mediana** é definida como o valor que separa a metade superior de uma amostra de dados da metade inferior. A função **numpy.median()** é usada conforme mostrado no programa a seguir.

### Exemplo

Demonstração ao vivo

```
import numpy as np
a = np.array([[30,65,70],[80,95,10],[50,90,60]])

print 'Our array is:'
print a
print '\n'

print 'Applying median() function:'
```

```
print np.median(a)
print '\n'

print 'Applying median() function along axis 0:'
print np.median(a, axis = 0)
print '\n'

print 'Applying median() function along axis 1:'
print np.median(a, axis = 1)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[30 65 70]
 [80 95 10]
 [50 90 60]]

Applying median() function:
65.0

Applying median() function along axis 0:
[ 50. 90. 60.]

Applying median() function along axis 1:
[ 65. 80. 60.]
```



## numpy.mean()

A média aritmética é a soma dos elementos ao longo de um eixo dividida pelo número de elementos. A função **numpy.mean()** retorna a média aritmética dos elementos do array. Se o eixo for mencionado, ele será calculado ao longo dele.

## Exemplo

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

```
print 'Our array is:'
print a
print '\n'

print 'Applying mean() function:'
print np.mean(a)
print '\n'

print 'Applying mean() function along axis 0:'
print np.mean(a, axis = 0)
print '\n'

print 'Applying mean() function along axis 1:'
print np.mean(a, axis = 1)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[1 2 3]
 [3 4 5]
 [4 5 6]]

Applying mean() function:
3.66666666667

Applying mean() function along axis 0:
[ 2.66666667 3.66666667 4.66666667]

Applying mean() function along axis 1:
[ 2. 4. 5.]
```



## numpy.average()

A média ponderada é uma média resultante da multiplicação de cada componente por um fator que reflete a sua importância. A função **numpy.average()** calcula a média ponderada dos elementos em um array de acordo com seus respectivos pesos dados em outro array. A função pode ter um parâmetro de eixo. Se o eixo não for especificado, a matriz será nivelada.

Considerando uma matriz [1,2,3,4] e pesos correspondentes [4,3,2,1], a média ponderada é calculada somando o produto dos elementos correspondentes e dividindo a soma pela soma dos pesos.

Média ponderada = (1*4+2*3+3*2+4*1)/(4+3+2+1)

## Exemplo

Demonstração ao vivo

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

print 'Our array is:'
print a
print '\n'

print 'Applying average() function:'
print np.average(a)
print '\n'

# this is same as mean when weight is not specified
wts = np.array([4,3,2,1])

print 'Applying average() function again:'
print np.average(a,weights = wts)
print '\n'

# Returns the sum of weights, if the returned parameter is set to True.
print 'Sum of weights'
print np.average([1,2,3, 4],weights = [4,3,2,1], returned = True)
```

Ele produzirá a seguinte saída -

```
Our array is:
[1 2 3 4]

Applying average() function:
2.5

Applying average() function again:
2.0
```

```
Sum of weights
(2.0, 10.0)
```

Em uma matriz multidimensional, o eixo para cálculo pode ser especificado.

## Exemplo

Demonstração ao vivo

```
import numpy as np
a = np.arange(6).reshape(3,2)

print 'Our array is:'
print a
print '\n'

print 'Modified array:'
wt = np.array([3,5])
print np.average(a, axis = 1, weights = wt)
print '\n'

print 'Modified array:'
print np.average(a, axis = 1, weights = wt, returned = True)
```

Ele produzirá a seguinte saída -

```
Our array is:
[[0 1]
 [2 3]
 [4 5]]

Modified array:
[ 0.625 2.625 4.625]

Modified array:
(array([ 0.625, 2.625, 4.625]), array([ 8., 8., 8.]))
```



## Desvio padrão

O desvio padrão é a raiz quadrada da média dos desvios quadrados da média. A fórmula para o desvio padrão é a seguinte -

```
std = sqrt(mean(abs(x - x.mean())**2))
```

Se a matriz for [1, 2, 3, 4], então sua média é 2,5. Portanto, os desvios quadrados são [2,25, 0,25, 0,25, 2,25] e a raiz quadrada de sua média dividida por 4, ou seja, sqrt (5/4) é 1,1180339887498949.

## Exemplo

```
import numpy as np
print np.std([1,2,3,4])
```

Demonstração ao vivo

Ele produzirá a seguinte saída -

```
1.1180339887498949
```

## Variância

A variância é a média dos desvios quadrados, ou seja, **mean(abs(x - x.mean())**2)** . Em outras palavras, o desvio padrão é a raiz quadrada da variância.

## Exemplo

```
import numpy as np
print np.var([1,2,3,4])
```

Demonstração ao vivo

Ele produzirá a seguinte saída -

```
1.25
```