Singular Value Decomposition

November 2, 2020

0.0.1 EJEMPLO 1

La descomposición SV se puede calcular usando la función `svd()`.

[1]:
```
# Descomposición SV

from numpy import array
from scipy.linalg import svd
# Definiendo la matriz
A = array([
    [1, 2],
    [3, 4],
    [5, 6]
])
print(A)
# Factorizando
U, s, V = svd(A)
print(U)
print(s)
print(V)
```

```
[[1 2]
 [3 4]
 [5 6]]
[[-0.2298477   0.88346102  0.40824829]
 [-0.52474482  0.24078249 -0.81649658]
 [-0.81964194 -0.40189603  0.40824829]]
[9.52551809 0.51430058]
[[-0.61962948 -0.78489445]
 [-0.78489445  0.61962948]]
```

0.0.2 EJEMPLO 2

[2]:
```
# Reconstruir la matriz original

from numpy import array, diag, zeros
from scipy.linalg import svd
# Definiendo la matriz
```

```
A = array([
    [1, 2],
    [3, 4],
    [5, 6]
])
print(A)
# Factorizando
U, s, V = svd(A)
# Creando la matriz Sigma de m x n
Sigma = zeros((A.shape[0], A.shape[1]))
# Llenando Sigma con la matriz diagonal de n x n
Sigma[:A.shape[1], :A.shape[1]] = diag(s)
# Reconstruyendo la matriz
B = U.dot(Sigma.dot(V))
print(B)
```

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

0.0.3 EJEMPLO 3

Es necesario usar la función `pinv()`.

[4]:
```
# Pseudoinversa

from numpy import array
from numpy.linalg import pinv
# Definiendo la matriz
A = array([
    [.1, .2],
    [.3, .4],
    [.5, .6],
    [.7, .8]
])
print(A)
# Calculando la pseudoinversa
B = pinv(A)
print(B)
```

```
[[0.1 0.2]
 [0.3 0.4]
 [0.5 0.6]
 [0.7 0.8]]
```

```
[[-1.00000000e+01 -5.00000000e+00  9.07607323e-15  5.00000000e+00]
 [ 8.50000000e+00  4.50000000e+00  5.00000000e-01 -3.50000000e+00]]
```

Calculando la pseudoinversa manualmente:

[5]:
```python
# Pseudoinversa via SVD

from numpy import array, diag, zeros
from scipy.linalg import svd
# Definiendo la matriz
A = array([
    [.1, .2],
    [.3, .4],
    [.5, .6],
    [.7, .8]
])
print(A)
# Factorizando
U, s, V = svd(A)
# Reciproco de s
d = 1 / s
# Creando la matris D de m x n
D = zeros(A.shape)
# Llenando D con la matriz diagonal de n x n
D[:A.shape[1], :A.shape[1]] = diag(d)
# Calculando la pseudoinversa
B = V.T.dot(D.T).dot(U.T)
print(B)
```

```
[[0.1 0.2]
 [0.3 0.4]
 [0.5 0.6]
 [0.7 0.8]]
[[-1.00000000e+01 -5.00000000e+00  9.07607323e-15  5.00000000e+00]
 [ 8.50000000e+00  4.50000000e+00  5.00000000e-01 -3.50000000e+00]]
```

0.0.4 EJEMPLO 4

[8]:
```python
# Reducción de la dimensión con SVD
from numpy import array, diag, zeros
from scipy.linalg import svd

# Definiendo la matriz
A = array([
    [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    [11, 12, 13, 14, 15, 16, 17, 18, 19, 20],
    [21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
])
print(A)
```

```
# Factorizando
U, s, V = svd(A)
# Creando la matriz Sigma de m x n
Sigma = zeros((A.shape[0], A.shape[1]))
# Llenando Sigma con la matriz diagonal de n x n
Sigma[:A.shape[0], :A.shape[0]] = diag(s)
# Seleccionando
n_elementos = 2
Sigma = Sigma[:, :n_elementos]
V = V[:n_elementos, :]
# Reconstruyendo
B = U.dot(Sigma.dot(V))
print(B)
# Transformando
T = U.dot(Sigma)
print(T)
T = A.dot(V.T)
print(T)
```

```
[[ 1  2  3  4  5  6  7  8  9 10]
 [11 12 13 14 15 16 17 18 19 20]
 [21 22 23 24 25 26 27 28 29 30]]
[[ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
 [11. 12. 13. 14. 15. 16. 17. 18. 19. 20.]
 [21. 22. 23. 24. 25. 26. 27. 28. 29. 30.]]
[[-18.52157747   6.47697214]
 [-49.81310011   1.91182038]
 [-81.10462276  -2.65333138]]
[[-18.52157747   6.47697214]
 [-49.81310011   1.91182038]
 [-81.10462276  -2.65333138]]
```

Usando la clase TruncatedSVD:

[9]:
```
# Reducción de dimensión

from numpy import array
from sklearn.decomposition import TruncatedSVD
# Definiendo la matriz
A = array([
    [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    [11, 12, 13, 14, 15, 16, 17, 18, 19, 20],
    [21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
])
print(A)
# Creando la transformación
svd = TruncatedSVD(n_components = 2)
# Encajando la transformación
```

```python
svd.fit(A)
# Aplicando la transformación
resultado = svd.transform(A)
print(resultado)
```

```
[[ 1  2  3  4  5  6  7  8  9 10]
 [11 12 13 14 15 16 17 18 19 20]
 [21 22 23 24 25 26 27 28 29 30]]
[[18.52157747  6.47697214]
 [49.81310011  1.91182038]
 [81.10462276 -2.65333138]]
```