In [34]:

```
print("\nveronica palacio villada \n")
print("codigo 1192808282 \n")
print("ingenieria de sistemas y computacion  \n")
```

```
veronica palacio villada

codigo 1192808282

ingenieria de sistemas y computacion
```

In [35]:

```
import numpy as np
from sklearn import preprocessing
input_data = np.array([[3.1, 7.9, 3.10],
[-1.2,7.8, -6.1],
[3.6, 0.4, 2.1],
[4.7, 2.6, 0.6],
[5.8, 3.6, 0.5],
[9.9, 4.4, 6.9],
[3.1, 5.5, 3.4],
[1.2, 6.2, 4.5],
[0.4, 0.8, 6.7],
[7.9, 4.7, 8.8],
[4.3,-9.9, -4.5]])
print(input_data)
```

```
[[ 3.1  7.9  3.1]
 [-1.2  7.8 -6.1]
 [ 3.6  0.4  2.1]
 [ 4.7  2.6  0.6]
 [ 5.8  3.6  0.5]
 [ 9.9  4.4  6.9]
 [ 3.1  5.5  3.4]
 [ 1.2  6.2  4.5]
 [ 0.4  0.8  6.7]
 [ 7.9  4.7  8.8]
 [ 4.3 -9.9 -4.5]]
```

In [36]:

```
data_binarized = preprocessing.Binarizer(threshold=5.1).transform(input_data)
print("\nDatos binarizados:\n", data_binarized)
data_binarized = preprocessing.Binarizer(threshold=7.7).transform(input_data)
print("\nDatos binarizados:\n", data_binarized)
```

```
Datos binarizados:
 [[0. 1. 0.]
 [0. 1. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [1. 0. 0.]
 [1. 0. 1.]
 [0. 1. 0.]
 [0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 1.]
 [0. 0. 0.]]

Datos binarizados:
 [[0. 1. 0.]
 [0. 1. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [1. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]
 [1. 0. 1.]
 [0. 0. 0.]]
```

In [37]:

```
print("\nANTES:")
print("Media =", input_data.mean(axis=0))
print("Desviación estándar =", input_data.std(axis=0))
```

```
ANTES:
Media = [3.89090909 3.09090909 2.36363636]
Desviación estándar = [3.06489049 4.73698303 4.3940606 ]
```

In [38]:

```
data_scaled = preprocessing.scale(input_data)
print("\nDESPUÉS:")
print("Media =", data_scaled.mean(axis=0))
print("Desviación estándar =", data_scaled.std(axis=0))
```

```
DESPUÉS:
Media = [ 8.57899610e-17 -1.61486985e-16  0.00000000e+00]
Desviación estándar = [1. 1. 1.]
```

In [39]:

```
data_scaler_minmax = preprocessing.MinMaxScaler(feature_range=(0,
1))
data_scaled_minmax = data_scaler_minmax.fit_transform(input_data)
print("\nMin max escalamiento de datos:\n", data_scaled_minmax)
```

```
Min max escalamiento de datos:
 [[0.38738739 1.         0.61744966]
 [0.         0.99438202 0.        ]
 [0.43243243 0.57865169 0.55033557]
 [0.53153153 0.70224719 0.44966443]
 [0.63063063 0.75842697 0.44295302]
 [1.         0.80337079 0.87248322]
 [0.38738739 0.86516854 0.63758389]
 [0.21621622 0.90449438 0.7114094 ]
 [0.14414414 0.6011236  0.8590604 ]
 [0.81981982 0.82022472 1.        ]
 [0.4954955  0.         0.10738255]]
```

In [40]:

```
data_normalized_l1 = preprocessing.normalize(input_data,
norm='l1')
data_normalized_l2 = preprocessing.normalize(input_data,
norm='l2')
print("\nL1 dato normalizado:\n", data_normalized_l1)
print("\nL2 dato normalizado:\n", data_normalized_l2)
```

```
L1 dato normalizado:
 [[ 0.21985816  0.56028369  0.21985816]
 [-0.0794702   0.51655629 -0.40397351]
 [ 0.59016393  0.06557377  0.3442623 ]
 [ 0.59493671  0.32911392  0.07594937]
 [ 0.58585859  0.36363636  0.05050505]
 [ 0.46698113  0.20754717  0.3254717 ]
 [ 0.25833333  0.45833333  0.28333333]
 [ 0.10084034  0.5210084   0.37815126]
 [ 0.05063291  0.10126582  0.84810127]
 [ 0.36915888  0.21962617  0.41121495]
 [ 0.22994652 -0.52941176 -0.24064171]]

L2 dato normalizado:
 [[ 0.3431127   0.87438398  0.3431127 ]
 [-0.12030718  0.78199664 -0.61156148]
 [ 0.85982792  0.09553644  0.50156629]
 [ 0.86962513  0.48106922  0.11101597]
 [ 0.84737022  0.52595393  0.07304916]
 [ 0.77076067  0.3425603   0.53719683]
 [ 0.43231037  0.76700228  0.47414686]
 [ 0.15475178  0.79955085  0.58031916]
 [ 0.05917652  0.11835304  0.99120669]
 [ 0.62079759  0.36933528  0.69152137]
 [ 0.36770945 -0.84658687 -0.38481221]]
```

In [41]:

```
import numpy as np
from sklearn import preprocessing
input_labels = ['red', 'black', 'red', 'green', 'black', 'yellow',
'white','pink']
encoder = preprocessing.LabelEncoder()
encoder.fit(input_labels)
print("\nMapeo de etiquetas:")
for i, item in enumerate(encoder.classes_):
 print(item, '-->', i)
test_labels = ['green', 'red', 'black']
encoded_values = encoder.transform(test_labels)
print("\nLabels =", test_labels)
print("Encoded values =",list(encoded_values))
encoded_values = [3, 0, 4,1,2]
decoded_list = encoder.inverse_transform(encoded_values)
print("\nEncoded values =", encoded_values)
print("Decoded labels =", list(decoded_list))
```

```
Mapeo de etiquetas:
black --> 0
green --> 1
pink --> 2
red --> 3
white --> 4
yellow --> 5

Labels = ['green', 'red', 'black']
Encoded values = [1, 3, 0]

Encoded values = [3, 0, 4, 1, 2]
Decoded labels = ['red', 'black', 'white', 'green', 'pink']
```

In [ ]: