```
#!/usr/bin/env python
# coding: utf-8
#jonatan hernandez henao
#1053864927
# In[5]:


# PROCESAMIENTO DIGITAL

# Se importan las librería numpy y las funciones de preprocesamiento
import numpy as np
from sklearn import preprocessing
# Datos de prueba
input_data = np.array([[5.1, -2.9, 3.3],
[-1.4, 7.8, -5.1],
[3.9, 0.4, 2.1],
[7.3, -9.9, -4.5]])
print(input_data)


# In[6]:


# Binarizar los datos

data_binarized = preprocessing.Binarizer(threshold=2.1).transform(input_data)
print("\nDatos binarizados:\n", data_binarized)


# In[7]:


# Imprimir la media y la desviación estándar

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


# In[8]:


# Remover la media
```

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


# In[9]:


# Escalamiento Min Max

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)


# In[10]:


# Normalización de datos

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)


# In[ ]:


# Manejo de etiquetas
import numpy as np
from sklearn import preprocessing
# Se definen algunas etiquetas simples
input_labels = ['red', 'black', 'red', 'green', 'black', 'yellow',
'white']
# Se crea un codificador de etiquetas y se ajustan las etiquetas
encoder = preprocessing.LabelEncoder()
encoder.fit(input_labels)
```

```
# Se imprime el mapeo entre palabras y números
print("\nMapeo de etiquetas:")
for i, item in enumerate(encoder.classes_):
print(item, '-->', i)
# Codificar un conjunto de etiquetas con el codificador
test_labels = ['green', 'red', 'black']
encoded_values = encoder.transform(test_labels)
print("\nLabels =", test_labels)
print("Encoded values =", list(encoded_values))
# Decodificar un conjunto de valores usando el codificador
encoded_values = [3, 0, 4, 1]
decoded_list = encoder.inverse_transform(encoded_values)
print("\nEncoded values =", encoded_values)
print("Decoded labels =", list(decoded_list))


# In[ ]:
#jonatan hernandez henao
#1053864927
```