NAME: MOHAMMAD HUSSAM(2033.KHI.DEG.020)

PAIRING WITH : MAVIA ALAM KHAN (2303.KHI.DEG.017)

&

AQSA TAUHEED(2303.KHI.DEG.011)

ASSIGNMENT_3.1

Implement a label encoder for categorical data using pure Python, Pandas and NumPy.

SOLUTION:

STEP:1

```
[1]: import pandas as pd
```

```
[2]: data = pd.read_csv('./Iris.csv')
     data
```

[2]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

We have to first import the library pandas as pd .

After that we loaded iris.csv file and assign it to the variable data as shown in image , then we printed its output .

STEP:2

```
6]: def label_encoder(data):
        for col in data.columns:
            if data[col].dtype == 'object':
                data[col] = pd.factorize(data[col])[0]
        return data
```

A function called **label_encoder** is defined, which takes a single argument **data** that is expected to be a Pandas DataFrame.

for col in data.columns

The code starts a loop over each column in the DataFrame data. The loop iterates over the column names, which are returned by the columns attribute of the DataFrame.

if data[col].dtype == 'object'

Inside the loop, an if statement checks whether the data type of the column is object, which is the Pandas data type used to represent strings.

data[col] = pd.factorize(data[col])[0]

If the column data type is object, the factorize function of Pandas is applied to the column. This function returns a tuple consisting of two elements: an array of unique integers that represent the values in the column, and an array that maps the original values to their corresponding integers. We only need the first array, so we access it with the [0] index. By default, the factorize function sorts the unique integers in ascending order.

```
    return data
```

After the loop is finished, the modified DataFrame is returned by the function.

OUTPUT:

Printed output

```
[17]: label_encoder(data).head(100)
```

[17]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	0
1	2	4.9	3.0	1.4	0.2	0
2	3	4.7	3.2	1.3	0.2	0
3	4	4.6	3.1	1.5	0.2	0
4	5	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...	...
95	96	5.7	3.0	4.2	1.2	1
96	97	5.7	2.9	4.2	1.3	1
97	98	6.2	2.9	4.3	1.3	1
98	99	5.1	2.5	3.0	1.1	1
99	100	5.7	2.8	4.1	1.3	1

100 rows × 6 columns

```
[18]: label_encoder(data).tail(50)
```

[18]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
100	101	6.3	3.3	6.0	2.5	2
101	102	5.8	2.7	5.1	1.9	2
102	103	7.1	3.0	5.9	2.1	2
103	104	6.3	2.9	5.6	1.8	2
104	105	6.5	3.0	5.8	2.2	2
105	106	7.6	3.0	6.6	2.1	2
106	107	4.9	2.5	4.5	1.7	2
107	108	7.3	2.9	6.3	1.8	2
108	109	6.7	2.5	5.8	1.8	2
109	110	7.2	3.6	6.1	2.5	2
110	111	6.5	3.2	5.1	2.0	2
111	112	6.4	2.7	5.3	1.9	2
112	113	6.8	3.0	5.5	2.1	2
113	114	5.7	2.5	5.0	2.0	2
114	115	5.8	2.8	5.1	2.4	2
115	116	6.4	3.2	5.3	2.3	2

As show in output iris-sentosa = 0 , iris-versicolor = 1 and 2 = iris virginica , hence the data we pass through the function is encoded as you can see in the outputs .