6_Encoding

January 14, 2025

```
[ ]: '''
        Encoding Categorical Data -->

        Categorical encoding refers to the process of converting categorical data
        (data that can take on a limited number of values,typically representing␣
      ↪categories or labels)
        into a numerical format that machine learning algorithms can understand and␣
      ↪work with

        Since many machine learning models can only handle numerical data,␣
      ↪categorical features need to be
        transformed into numbers. There are several techniques to achieve this,␣
      ↪depending on the type of categorical
        data and the specific problem. Here are some common methods for categorical␣
      ↪encoding
     '''
```

```
[ ]: #   Methods -->

     #   Label Encoding
     #   One-Hot Encoding
     #   Binary Encoding
     #   Frequency Encoding
     #   Mean Encoding
     #   Ordinal Encoding
```

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

```
[ ]: '''
        Label Encoding -->

        Label encoding is another technique for converting categorical data into␣
      ↪numerical form,
        but unlike one-hot encoding, it assigns an integer to each category. Each␣
      ↪unique category
```

```
    is given a distinct integer value, which allows algorithms to process the␣
↪data in numerical form.

    Example -->

    If you have the categories (Red, Blue, Green), label encoding would assign␣
↪-->

    Red = 0
    Blue = 1
    Green = 2
'''
```

[3]:
```
#   Import Label Encoder From sklearn -->

from sklearn.preprocessing import LabelEncoder
```

[4]:
```
dataset = pd.read_csv('Data/Data.csv')
dataset
```

[4]:
```
   Country   Age   Salary Purchased
0   France  44.0  72000.0        No
1    Spain  27.0  48000.0       Yes
2  Germany  30.0  54000.0        No
3    Spain  38.0  61000.0        No
4  Germany  40.0      NaN       Yes
5   France  35.0  58000.0       Yes
6    Spain   NaN  52000.0        No
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
9   France  37.0  67000.0       Yes
```

[5]:
```
encoder = LabelEncoder()
dataset['Country'] = encoder.fit_transform(dataset['Country'])
dataset
```

[5]:
```
   Country   Age   Salary Purchased
0        0  44.0  72000.0        No
1        2  27.0  48000.0       Yes
2        1  30.0  54000.0        No
3        2  38.0  61000.0        No
4        1  40.0      NaN       Yes
5        0  35.0  58000.0       Yes
6        2   NaN  52000.0        No
7        0  48.0  79000.0       Yes
8        1  50.0  83000.0        No
9        0  37.0  67000.0       Yes
```

[ ]:

```
'''
    One Hot Encoding -->

    One-hot encoding is a technique used to represent categorical data as␣
 ↪binary vectors.
    It transforms categorical variables, which might not have a numerical␣
 ↪relationship,
    into a format that can be provided to machine learning algorithms.

    Here's how it works ->

    Each category in the data is represented by a vector of 0s and 1s.
    The length of the vector equals the number of unique categories.
    For a given category, the corresponding position in the vector is marked as␣
 ↪1, and all other positions are marked as 0.

    Example -->

    Suppose you have a dataset with three categories:
    Red, Blue, and Green. One-hot encoding would represent these categories as:

    Red: [1, 0, 0]
    Blue: [0, 1, 0]
    Green: [0, 0, 1]
'''
```

[16]:

```
data = pd.read_csv('Data/Data.csv')
data
```

[16]:

```
   Country   Age   Salary Purchased
0   France  44.0  72000.0        No
1    Spain  27.0  48000.0       Yes
2  Germany  30.0  54000.0        No
3    Spain  38.0  61000.0        No
4  Germany  40.0      NaN       Yes
5   France  35.0  58000.0       Yes
6    Spain   NaN  52000.0        No
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
9   France  37.0  67000.0       Yes
```

[17]:

```
x_data = data.iloc[:, :-1].values
y_data = data.iloc[:, -1].values
```

[12]:

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
```

[18]:

```
clt = ColumnTransformer(transformers = [('encoder', OneHotEncoder(), [0])],␣
 ↪remainder = 'passthrough')
x_data = np.array(clt.fit_transform(x_data))
x_data
```

[18]:

```
array([[1.0, 0.0, 0.0, 44.0, 72000.0],
       [0.0, 0.0, 1.0, 27.0, 48000.0],
       [0.0, 1.0, 0.0, 30.0, 54000.0],
       [0.0, 0.0, 1.0, 38.0, 61000.0],
       [0.0, 1.0, 0.0, 40.0, nan],
       [1.0, 0.0, 0.0, 35.0, 58000.0],
       [0.0, 0.0, 1.0, nan, 52000.0],
       [1.0, 0.0, 0.0, 48.0, 79000.0],
       [0.0, 1.0, 0.0, 50.0, 83000.0],
       [1.0, 0.0, 0.0, 37.0, 67000.0]], dtype=object)
```

[19]:

```
y_data = encoder.fit_transform(y_data)
y_data
```

[19]:

```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```