How do you perform label encoding in Python?

Authored by

stats writer

June 27, 2024

Label encoding is a process of converting categorical data into numerical data to make it easier for machine learning algorithms to process. In Python, this can be achieved by using the LabelEncoder function from the scikit-learn library. First, the categorical data is imported and then the LabelEncoder is instantiated. Next, the fit_transform method is used to transform the categorical data into numerical labels. Finally, the numerical labels can be used in the machine learning algorithm. This process allows for efficient handling of categorical data in Python.

Perform Label Encoding in Python (With Example)

Often in machine learning, we want to convert into some type of numeric format that can be readily used by algorithms.

One way to do this is through label encoding, which assigns each categorical value an integer value based on alphabetical order.

For example, the following screenshot shows how to convert each unique value in a categorical variable called Team into an integer value based on alphabetical order:

Original Data

Team	Points
A	25
A	12
B	15
B	14
B	19
B	23
C	25
C	29

Label Encoded Data

Team	Points
0	25
0	12
1	15
1	14
1	19
1	23
2	25
2	29

You can use the following syntax to perform label encoding in Python:

```
from sklearn.preprocessingimport LabelEncoder

#create instance of label encoder
lab = LabelEncoder()

#perform label encoding on 'team' column
df = lab.fit_transform(df)
```

The following example shows how to use this syntax in practice.

Example: Label Encoding in Python

Suppose we have the following pandas DataFrame:

```
import pandas as pd

#create DataFrame
df = pd.DataFrame({'team': ,
'points': })

#view DataFrame
print(df)
```

```
team points
0 A 25
1 A 12
2 B 15
3 B 14
4 B 19
5 B 23
6 C 25
7 C 29
```


We can use the following code to perform label encoding to convert each categorical value in the team column into an integer value:

```
from sklearn.preprocessingimport LabelEncoder

#create instance of label encoder
lab = LabelEncoder()

#perform label encoding on 'team' column
df = lab.fit_transform(df)

#view updated DataFrameprint(df)

team points
0 0 25
1 0 12
2 1 15
3 1 14
4 1 19
5 1 23
6 2 25
7 2 29
```


From the output we can see:

Each "A" value has been converted to 0.Each "B" value has been converted to 1.Each C" value has been converted to 2.

Note that you can also use the inverse_transform() function to obtain the original values from the team column:

#display original team labels
lab.inverse_transform(df)

array(, dtype=object)