How can I use label encoding across multiple columns in Scikit-Learn?

Authored by

stats writer

June 27, 2024

Label encoding is a technique used to convert categorical data into numerical data in order to make it usable for machine learning algorithms. In Scikit-Learn, label encoding can be applied to multiple columns simultaneously by using the "LabelEncoder" function. This function can be used to encode multiple columns containing categorical data by passing them as a list. This allows for efficient and consistent encoding of multiple columns in a dataset. Additionally, Scikit-Learn provides the "OrdinalEncoder" function, which can handle both categorical and numerical data, allowing for even more flexibility in encoding multiple columns. Overall, the use of label encoding in Scikit-Learn allows for easier and more efficient handling of categorical data in machine learning models.

Scikit-Learn: Use Label Encoding Across Multiple Columns

In machine learning, label encoding is the process of converting the values of a into integer values.

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 across multiple columns in Python:

```
from sklearn.preprocessingimport LabelEncoder

#perform label encoding on col1, col2 columns
df] = df].apply(LabelEncoder().fit_transform)
```

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

Example: Label Encoding in Python

Suppose we have the following pandas DataFrame that contains information about various basketball players:

```
import pandas as pd

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

#view DataFrame
print(df)

team position all_star points
0 A G Y 11
1 A F N 8
2 B G Y 10
3 B F Y 6
4 B F Y 6
5 C G N 5
6 C G Y 9
7 D F N 12
```

We can use the following code to perform label encoding to convert each categorical value in the team, position, and all_star columns into integer values:

```
from sklearn.preprocessingimport LabelEncoder

#perform label encoding across team, position, and all_star columns
df] = df].apply(LabelEncoder().fit_transform)

#view udpated DataFrame
print(df)
```

team position all_star points
0 0 1 1 11
1 0 0 0 8
2 1 1 1 10
3 1 0 1 6
4 1 0 1 6
5 2 1 0 5
6 2 1 1 9
7 3 0 0 12

From the output we can see that each value in the team, position, and all_star columns have been converted into integer values.

For example, in the team column 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.Each "D" value has been converted to 3.

Note that in this example we performed label encoding on three columns in the DataFrame, but we can use similar syntax to perform label encoding on as many categorical columns as we'd like.

The following tutorials explain how to perform other common tasks in Python: