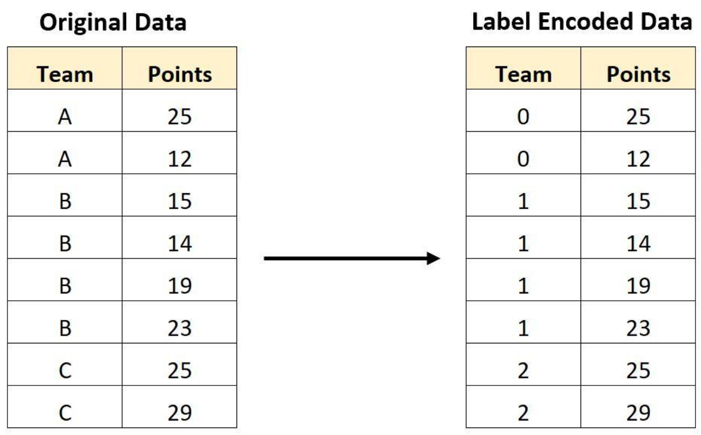
1- Label Encoding:

Label Encoding Is used to convert categorical variables into numerical format that can be readily used by algorithms.

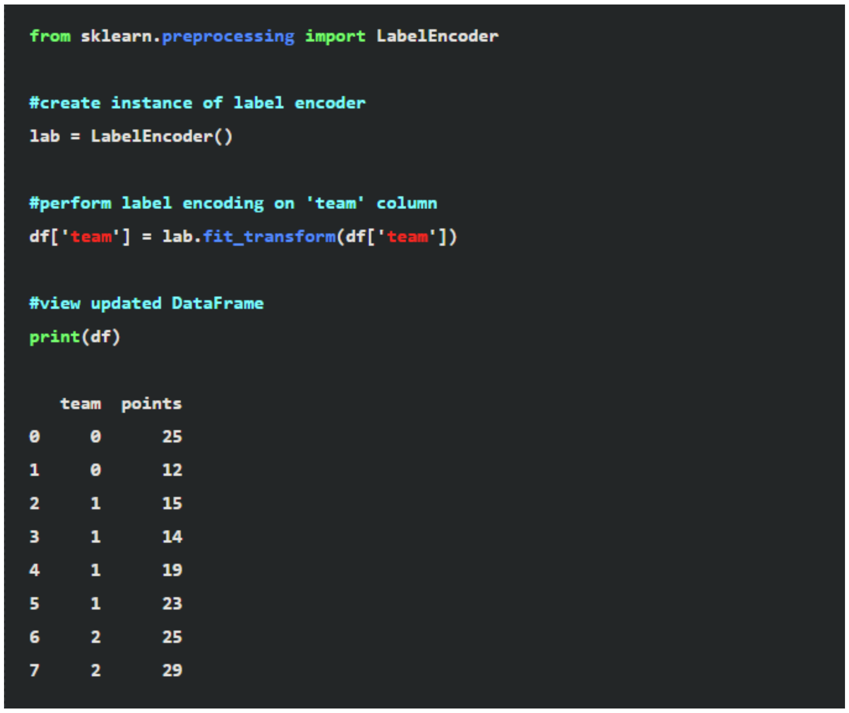


Original Data

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

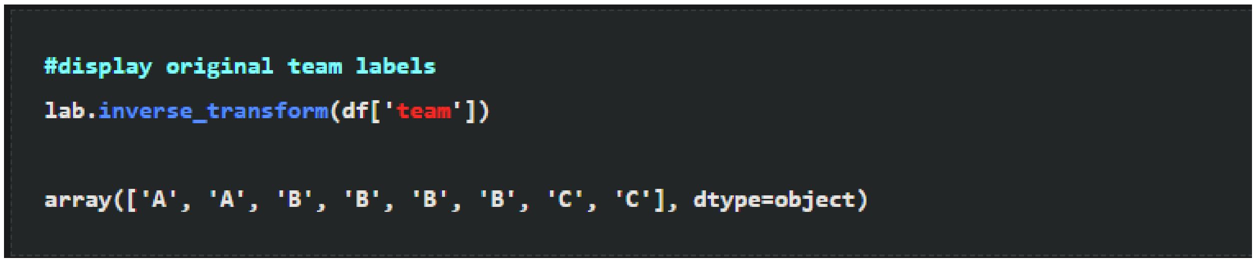
Label Encoded Data

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



```
from sklearn.preprocessing import LabelEncoder

#create instance of label encoder
lab = LabelEncoder()

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

#view updated DataFrame
print(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
```

Note that you can also use the inverse_transform() function to obtain the original values



```
#display original team labels
lab.inverse_transform(df['team'])

array(['A', 'A', 'B', 'B', 'B', 'B', 'C', 'C'], dtype=object)
```

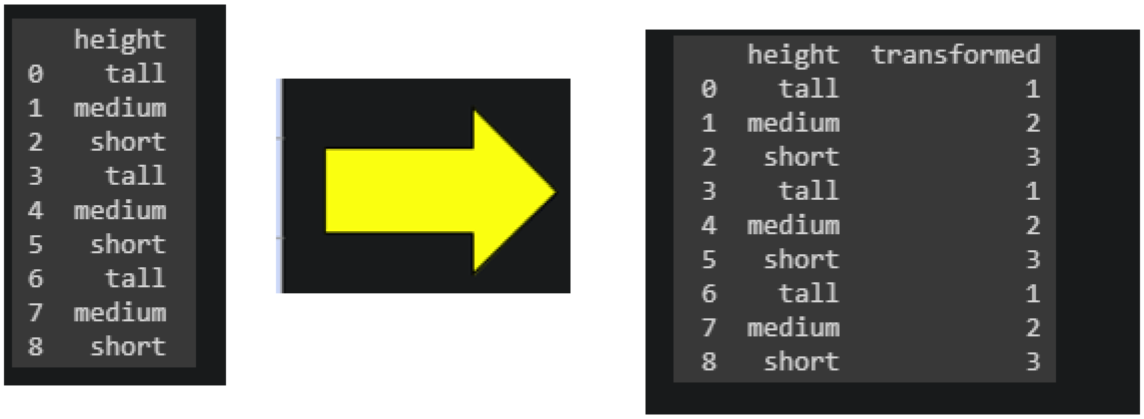
Label encoder is used when:

- The number of categories is quite large.
- When the order does not matter in categorical feature.

2- Ordinal Encoding:

It’s the same as Label Encoding but it assumes order.

hint: it used when order matters

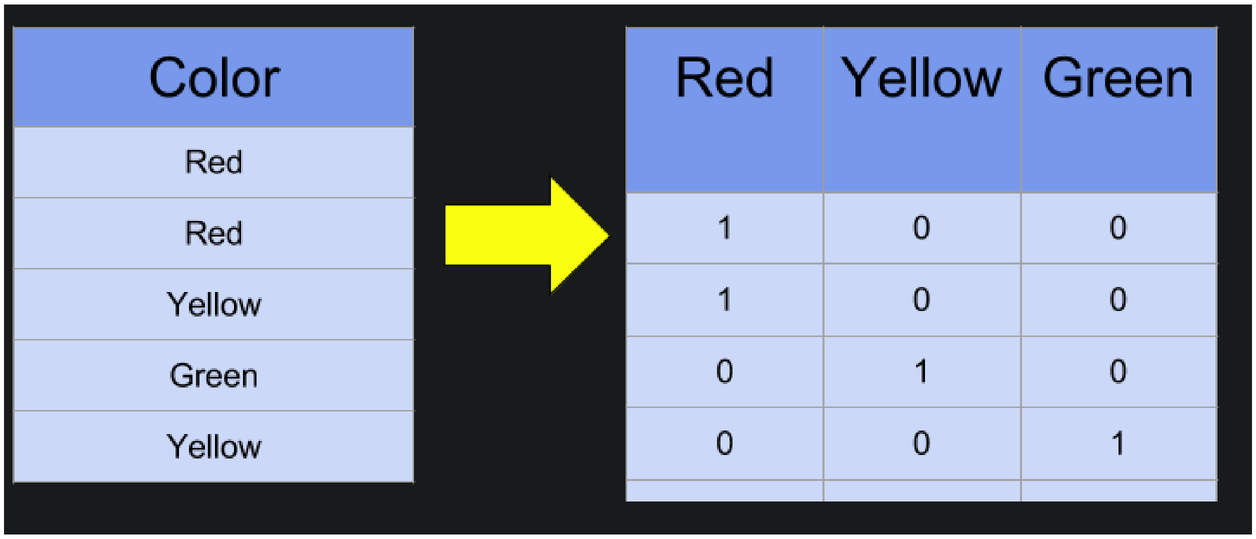


```
   height
0    tall
1  medium
2   short
3    tall
4  medium
5   short
6    tall
7  medium
8   short
```

```
   height  transformed
0    tall            1
1  medium            2
2   short            3
3    tall            1
4  medium            2
5   short            3
6    tall            1
7  medium            2
8   short            3
```

3- One-Hot Encoding:

One hot encoding creates new (binary) columns, indicating the presence of each possible value from the original data.



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

One Hot encoder is used when:

- When the order does not matter in categorical features
- Categories in a feature are fewer.

But one-hot encoding can lead to high memory consumption

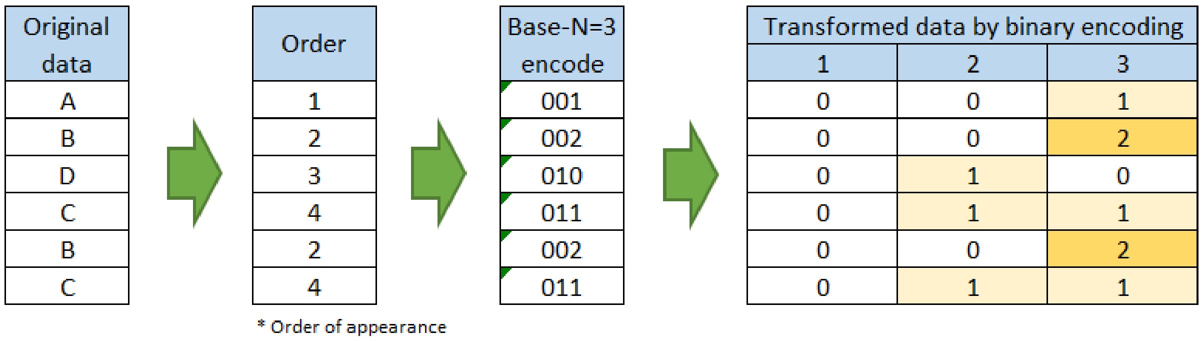
4- Binary Encoding:

Binary encoding is a combination of Hash encoding and one-hot encoding.

STEPS:

1. the categorical feature is first converted into numerical using an ordinal encoder.
2. Then the numbers are transformed in the binary number.
3. After that binary value is split into different columns.

Binary encoding works well when there are a <u>high number of categories</u>.



Original data
A
B
D
C
B
C

Order
1
2
3
4
2
4

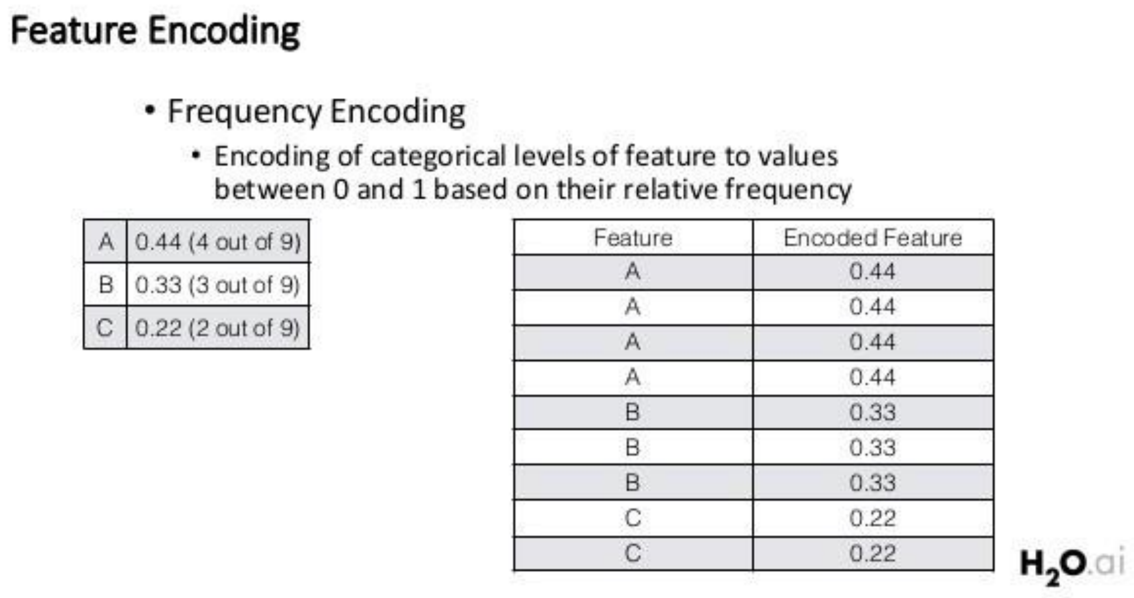
* Order of appearance

Base-N=3 encode
001
002
010
011
002
011

Transformed data by binary encoding		
1	2	3
0	0	1
0	0	2
0	1	0
0	1	1
0	0	2
0	1	1

5- Frequency Encoding:

it encodes categorical feature values to their frequencies



Feature Encoding

- Frequency Encoding
  - Encoding of categorical levels of feature to values between 0 and 1 based on their relative frequency

A	0.44 (4 out of 9)
B	0.33 (3 out of 9)
C	0.22 (2 out of 9)

Feature	Encoded Feature
A	0.44
A	0.44
A	0.44
A	0.44
B	0.33
B	0.33
B	0.33
C	0.22
C	0.22

Advantages of Count or Frequency encoding

- Straightforward to implement.
- Does not expand the feature space.
- Can work well with tree-based algorithms.

Limitations of Count or Frequency encoding

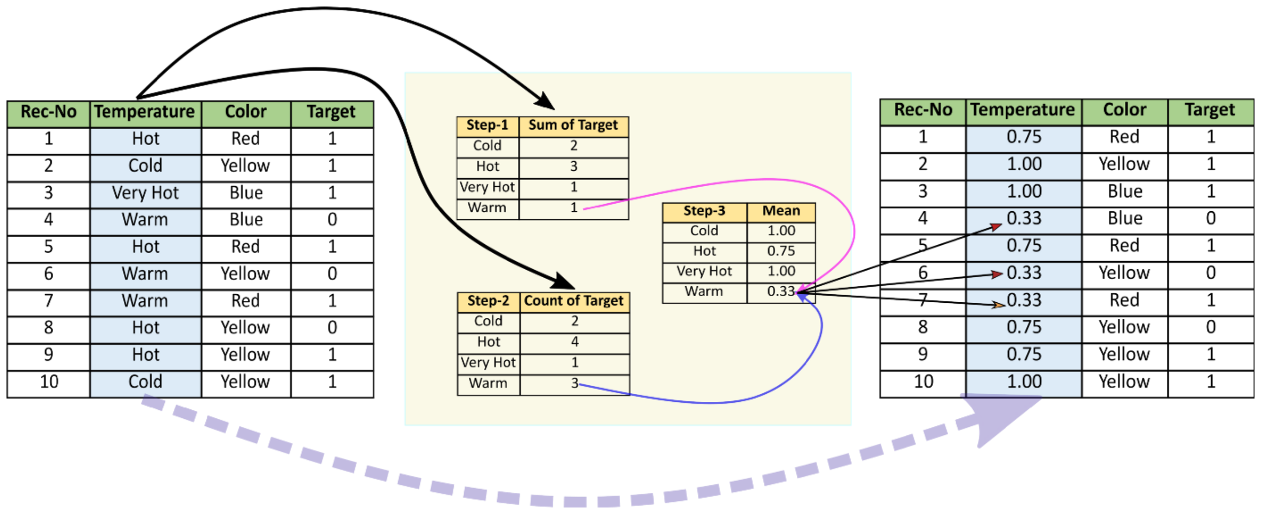
- Does not handle new categories in the test set automatically.
- We can lose valuable information if there are two different categories with the same amount of observations count—this is because we replace them with the same number.

6- Target Mean Encoding:

Target encoding is the process of replacing a <u>categorical value with the mean of the target variable</u>.

*Any non-categorical columns are automatically dropped by the target encoder model.

*You can also use target encoding to convert categorical columns to numeric.



Rec-No	Temperature	Color	Target
1	Hot	Red	1
2	Cold	Yellow	1
3	Very Hot	Blue	1
4	Warm	Blue	0
5	Hot	Red	1
6	Warm	Yellow	0
7	Warm	Red	1
8	Hot	Yellow	0
9	Hot	Yellow	1
10	Cold	Yellow	1

Step-1	Sum of Target
Cold	2
Hot	3
Very Hot	1
Warm	1

Step-2	Count of Target
Cold	2
Hot	4
Very Hot	1
Warm	3

Step-3	Mean
Cold	1.00
Hot	0.75
Very Hot	1.00
Warm	0.33

Rec-No	Temperature	Color	Target
1	0.75	Red	1
2	1.00	Yellow	1
3	1.00	Blue	1
4	0.33	Blue	0
5	0.75	Red	1
6	0.33	Yellow	0
7	0.33	Red	1
8	0.75	Yellow	0
9	0.75	Yellow	1
10	1.00	Yellow	1