Encoders in Machine Learning

Definition

Encoders are used to transform categorical data into numerical data that can be used by algorithms

Types

Label Encoder

Label Encoder

Purpose: Converts categorical labels into numerical labels.

How It Works: Assigns a unique number to each category.

Example: For `["Apple", "Banana", "Cherry"]`, it might assign `Apple` = 0, `Banana` = 1, `Cherry` = 2.

```python
from sklearn.preprocessing import LabelEncoder

# Define the custom order
order = ["Low", "Medium", "High"]

# Create a LabelEncoder instance
encoder = LabelEncoder()

# Fit the encoder with the custom order
encoder.fit(order)

# Sample data
data = ["Low", "Medium", "High", "Low"]

# Encode the data
encoded_data = encoder.transform(data)
print(encoded_data)
```

One-Hot Encoder

One-Hot Encoder

Purpose: Converts categorical values into a binary matrix, avoiding any ordinal relationship.

How It Works: Creates a new binary column for each category.

Example: For `["Apple", "Banana", "Cherry"]`, it creates:

- `Apple` → `[1, 0, 0]`
- `Banana` → `[0, 1, 0]`
- `Cherry` → `[0, 0, 1]`

```python
from sklearn.preprocessing import OneHotEncoder
import numpy as np

# Sample data: let's say we have a list of fruits
data = np.array(["Apple", "Banana", "Cherry", "Apple", "Cherry"]).reshape(-1, 1)

# Initialize the OneHotEncoder
encoder = OneHotEncoder(sparse=False)

# Fit and transform the data
one_hot_encoded = encoder.fit_transform(data)

# Print the original categories
print("Original Categories:\n", encoder.categories_)

# Print the one-hot encoded data
print("\nOne-Hot Encoded Data:\n", one_hot_encoded)
```

Ordinal Encoder

Ordinal Encoder

Purpose: Encodes categorical data with a meaningful order or ranking.

How It Works: Assigns integer values based on the order of categories.

Example: For `["Low", "Medium", "High"]`, it might assign `Low` = 0, `Medium` = 1, `High` = 2.

Binary Encoder

Binary Encoder

Purpose: Reduces dimensionality compared to one-hot encoding by encoding categories into binary numbers.

How It Works: Converts categorical values into binary representation and then creates binary columns.

Example: For `["Apple", "Banana", "Cherry"]`, if encoded as binary:

- `Apple` → `00`
- `Banana` → `01`
- `Cherry` → `10`

Frequency Encoder

Frequency Encoder

Purpose: Encodes categories based on the frequency of their occurrence.

How It Works: Replaces each category with the number of times it appears in the dataset.

Example: If `Apple` appears 10 times, `Banana` 5 times, and `Cherry` 2 times, then:

- `Apple` → 10
- `Banana` → 5
- `Cherry` → 2

Target Encoder (Mean Encoding)

Target Encoder (Mean Encoding)

Purpose: Encodes categories based on the mean of the target variable for each category.

How It Works: Replaces each category with the mean of the target variable for that category.

Example: If `Apple` has an average target value of 5, `Banana` 10, and `Cherry` 7, then:

- `Apple` → 5
- `Banana` → 10
- `Cherry` → 7