Day38_Introduction_to_Data_Preprocessing_in_ML

July 7, 2025

Introduction to Data Preprocessing in Machine Learning

Today we are learning about Data Preprocessing in Machine Learning

We will learn how to:

1. Import data
2. Handle missing values using SimpleImputer
3. Encode categorical data using LabelEncoder
4. Split the dataset into training and testing sets

1 Importing Important Libraries

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

2 Loading the Dataset

- We use pandas to read the CSV file into a DataFrame
- Replace the file path with your actual file location

[2]:
```
df = pd.read_csv(r'C:\Users\Lenovo\Downloads\Data.csv')
```

3 Separating Features and Target Variable

3.1 X contains input features (all columns except the last one)

[3]:
```
X = df.iloc[:, :-1].values
```

3.2 y contains the target/output variable (column index 3 here, adjust if needed)

[4]:
```
y = df.iloc[:, 3].values
```

4 Handling Missing Data using SimpleImputer

We can use SimpleImputer to fill the missing value with the median of the column

[6]:
```
from sklearn.impute import SimpleImputer
```

[7]:
```
# Create an imputer object with strategy = 'median'
imputer = SimpleImputer(strategy='median')
```

[8]:
```
# Fit the imputer on columns 1 and 2 of X (index 1 and 2)
imputer = imputer.fit(X[:, 1:3])
```

[9]:
```
# Replace missing values in X with the median values
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

5 Encoding Categorical Data using LabelEncoder

Machine learning models do not understand text, so we need to convert text labels into numbers

[10]:
```
from sklearn.preprocessing import LabelEncoder
```

[11]:
```
# Encoding the first column of X (e.g., country names or categories)
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])
```

[12]:
```
# Encoding the target column y if it is categorical
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
```

6 Splitting the Dataset into Training and Testing Sets

We split the data so that the model can learn on training data and be tested on unseen test data

[13]:
```
from sklearn.model_selection import train_test_split

# Splitting: 70% for training and 30% for testing
# random_state = 0 ensures the same result each time you run
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=0)

# Now, the data is ready for building a machine learning model!
```

7 Checking the Sizes of the Training and Testing Sets

[23]:
```
# Number of rows (samples) in training set
print("Number of training samples (X_train):", len(X_train))
print("Number of training labels (y_train):", len(y_train))
```

```
Number of training samples (X_train): 7
Number of training labels (y_train): 7
```

[24]:
```
# Number of rows (samples) in testing set
print("Number of test samples (X_test):", len(X_test))
print("Number of test labels (y_test):", len(y_test))
```

```
Number of test samples (X_test): 3
Number of test labels (y_test): 3
```

8 Conclusion

- Today we learned how to prepare data for machine learning.
- We handled missing values using SimpleImputer,
- converted categorical values into numbers using LabelEncoder,
- and split the dataset into training and test sets using train_test_split.
- This is the most important step before training any machine learning model.
- I understood that without clean and processed data,
- the model may give wrong predictions or not work at all.
- I will now practice with more datasets to become better at data preprocessing.

[ ]:
```

```