Data Preprocessing

It is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. It is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

1. Get the Dataset

Definition: A dataset is a collection of data, typically formatted in CSV, HTML, or XLSX files.

2. Importing Libraries

Numpy: For mathematical operations.

Matplotlib: For plotting charts.

Pandas: For data manipulation.

```
import numpy as nm

import matplotlib.pyplot as mpt

import pandas as pd
```

3. Importing the Dataset

Setting Working Directory: Ensure the directory contains your dataset.

```
data_set = pd.read_csv('Dataset.csv')
```

4. Handling Missing Data

Using Scikit-learn's Imputer:

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values=np.nan, strategy='mean')

imputer = imputer.fit(x[:, 1:3])

x[:, 1:3] = imputer.transform(x[:, 1:3])
```

5. Encoding Categorical Data

Label Encoding:

```
from sklearn.preprocessing import LabelEncoder

label_encoder_x = LabelEncoder()
```

```
x[:, 0] = label_encoder_x.fit_transform(x[:, 0])
```

One-Hot Encoding:

```
from sklearn.preprocessing import OneHotEncoder

onehot_encoder = OneHotEncoder()

x = onehot_encoder.fit_transform(x).toarray()
```

Encoding Target Variable:

```
labelencoder_y = LabelEncoder()

y = labelencoder_y.fit_transform(y)
```

6. Splitting the Dataset

Train-Test Split:

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

7. Feature Scaling

Standardization:

```
from sklearn.preprocessing import StandardScaler

st_x = StandardScaler()

x_train = st_x.fit_transform(x_train)

x_test = st_x.transform(x_test)
```