# 📘 DATA PREPROCESSING IN MACHINE LEARNING

*(MOST IMPORTANT PART OF ML)*

---

## 1. What is Data Preprocessing?

**Data Preprocessing is the process of cleaning and preparing raw data so that it can be used by a Machine Learning model.**

**Real-world data is usually:**

- **Incomplete**
- **Noisy**
- **Inconsistent**

**If preprocessing is not done, model accuracy becomes very poor.**

---

## 2. Why Data Preprocessing is Necessary?

**Machine Learning models:**

- **Cannot handle missing values properly**
- **Cannot understand text data**
- **Are affected by different data scales**

**So preprocessing is required to:**

- **Improve accuracy**
- **Avoid wrong predictions**
- **Make data usable**

---

3. Common Data Preprocessing Steps

Handle Missing Values
↓
Encode Categorical Data
↓

Feature Scaling
↓
Train-Test Split

---

## 4. Handling Missing Values

Missing values mean **empty or NaN values** in the dataset.

### Example:

| Age | Salary |
|---|---|
| 25 | 50000 |
| NaN | 60000 |

---

### Methods to Handle Missing Values

### Method 1: Remove Missing Data

**df.dropna()**

**Used when missing data is very small.**

---

### Method 2: Replace with Mean (Most Common)

**df['Age'].fillna(df['Age'].mean(), inplace=True**)

Used when data is numerical.

---

## 5. Encoding Categorical Data

Machine Learning models **cannot understand text**, so we convert text into numbers.

### Example:

| Gender |
|---|
| Male |
| Female |

---

### Label Encoding

**from sklearn.preprocessing import LabelEncoder**

**le = LabelEncoder()**

**df['Gender'] = le.fit_transform(df['Gender'])**

**Output:**

**Male → 1**

**Female → 0**

---

## 6. Feature Scaling

Feature scaling means **bringing all values to the same range**.

### Why needed?

Some algorithms depend on distance (KNN, SVM).

Example:

**Age → 20–60**

**Salary → 20,000–1,00,000**

Salary dominates age → wrong prediction.

---

### Standardization (Most Used)

**from sklearn.preprocessing import StandardScaler**

**scaler = StandardScaler()**

**X_scaled = scaler.fit_transform(X)**

**Now all features are on the same scale.**

---

## 7. Train-Test Split (Revision but Important)

Dataset is divided into:

- Training data (learning)
- Testing data (checking)

```
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=42

)
```

---

## 8. Model Evaluation Metrics

Accuracy alone is not enough.

---

### Confusion Matrix

**Shows correct and incorrect predictions.**

```
from sklearn.metrics import confusion_matrix


cm = confusion_matrix(y_test, y_pred)

print(cm)
```

---

### Precision, Recall, F1-score

```
from sklearn.metrics import classification_report


print(classification_report(y_test, y_pred))
```

These metrics help understand model performance deeply.