Aim: Demonstrate various data pre-processing techniques for a given dataset

**Theory:**

**🔍 What is Data Preprocessing?**

Data preprocessing is the **first and essential step** in any machine learning project. It involves transforming raw data into a clean and structured format that can be used by machine learning models effectively.

💡 *“Garbage in, garbage out” — if your data is messy, your model won’t work well, no matter how advanced it is.*

**🧱 Types of Data Preprocessing Techniques:**

**🔹 1. Handling Missing Values**

Real-world datasets often have missing entries due to errors, delays, or incomplete data collection.

**Techniques:**

* **Removal:** Drop rows or columns with missing values.
* **Imputation:** Replace with:
  + Mean / Median / Mode
  + Constant value
  + Forward-fill / Backward-fill

df.dropna() # Remove missing rows

df.fillna(df.mean()) # Fill with column mean

**🔹 2. Encoding Categorical Variables**

Machine learning models can't process text or labels directly. You must convert **categorical (string) data** into **numerical values**.

**Techniques:**

* **Label Encoding**: Converts categories into numbers (e.g., Male = 0, Female = 1)
* **One-Hot Encoding**: Creates separate binary columns for each category.

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

le = LabelEncoder()

df['Gender'] = le.fit\_transform(df['Gender']) # Male=1, Female=0

**🔹 3. Feature Scaling (Normalization / Standardization)**

If your dataset contains features with **different units or scales**, scaling ensures all features contribute equally.

**Techniques:**

* **Standardization**: (X - mean) / std → Produces data with mean = 0, std = 1
* **Normalization**: Scales values between 0 and 1

from sklearn.preprocessing import StandardScaler, MinMaxScaler

scaler = StandardScaler()

df\_scaled = scaler.fit\_transform(df)

**🔹 4. Removing Duplicates**

Duplicates distort statistics and model performance.

df = df.drop\_duplicates()

**🔹 5. Feature Selection**

Not all features are useful. Feature selection removes irrelevant or redundant variables, improving model performance.

**Methods:**

* Correlation Matrix
* Univariate Selection
* Recursive Feature Elimination (RFE)

**🔹 6. Outlier Detection and Treatment**

Outliers are extreme values that may mislead your model.

**Techniques:**

* Using **Z-Score** or **IQR method** to detect outliers
* Cap or remove them

from scipy import stats

df = df[(np.abs(stats.zscore(df)) < 3).all(axis=1)]

**🔹 7. Data Splitting**

Before applying ML, split data into:

* **Training set** – used to train model
* **Testing set** – used to evaluate model

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)

**📈 Why Preprocessing is Important?**

| **Without Preprocessing** | **With Preprocessing** |
| --- | --- |
| Biased models | Accurate models |
| Poor performance | Better performance |
| Slow convergence | Faster training |

**✅ Summary of Techniques:**

| **Technique** | **Purpose** |
| --- | --- |
| Missing value handling | Deal with null/NaN values |
| Encoding | Convert text labels to numbers |
| Scaling | Normalize feature ranges |
| Feature selection | Remove irrelevant inputs |
| Outlier removal | Handle extreme values |
| Train-test split | Evaluate model generalization |
|  |  |