📅 DAY 12 — Data Cleaning & Preprocessing

Goal : Prepare clean, usable data for ML

1️⃣ Why Data Cleaning Matters

Real- world data is:

❌ Missing values

❌ Wrong formats

❌ Outliers

❌ Categorical (text)

ML models **hate messy data.**

> Better data > Better Model

2️⃣ Handling Missing Values

Detect missing values

```
df.isnull()
df.isnull().sum()
```

Option 1 - Drop missing values

```
df.dropna()
```

⚠️ Use only if data loss is acceptable.

Option 2 - Fill missing values (Recommended)

```
df["age"].fillna(df["age"].mean(), inplace=True)
df["city"].fillna("Unknown", inplace=True)
```

3️⃣ Removing Duplicates

```
df.duplicated()
df.drop_duplicates(inplace=True)
```

Very common in real datasets.

4️⃣ Handling Outliers (Simple Method)

Using IQR (basic idea)

```
Q1 = df["salary"].quantile(0.25)
Q3 = df["salary"].quantile(0.75)
IQR = Q3 - Q1

df = df[
    (df["salary"] >= Q1 - 1.5 * IQR) &
    (df["salary"] <= Q3 + 1.5 * IQR)
]
```

👉 Outliers can break ML models.

5️⃣ Encoding Categorical Data (VERY IMPORTANT)

ML models **cannot understand text**.

One-Hot Encoding

```
pd.get_dummies(df, columns=["city"]
```

Example:

```
city_Colombo city_Kandy
```

6️⃣ Feature Scaling (IMPORTANT

ML models work better when features are on same scale.

Min-Max Scaling(Simple)

```
df["age_scaled"] = (
    df["age"] - df["age"].min()
) / (df["age"].max() - df["age"].min()
```

Standardization (Used often)

```
df["salary_scaled"] = (
    df["salary"] - df["salary"].mean()
) / df["salary"].std()
```

7️⃣ Train-Test Split (ML CORE)

Never train and test on same data.

```
from sklearn.model_selection import train_test_split

X = df.drop("target", axis=1)
y = df["target"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

8️⃣ Real ML Pipeline Example

```
df = pd.read_csv("data.csv")

df.drop_duplicates(inplace=True)
df.fillna(df.mean(), inplace=Ture)
df = pd.get_dummies(df)

X = df.drop("target", axis=1)
y = df["target"]
```

This is **real-world preprocessing.**