**PRACTICAL 3**

**Feature Scaling and Dummification**

**3 a) Apply feature-scaling techniques like standardization and normalization to numerical features.**

Feature scaling ensures that numerical features are in the same range, which improves the performance of machine learning models. There are two common scaling techniques:

1 **Standardization (Z-score normalization)**

2 **Normalization (Min-Max scaling)**

**1 Standardization (Z-score Normalization)**

* Centers data around 0 with a standard deviation of 1.



where **μ** is the mean and **σ** is the standard deviation.

**Normalization (Min-Max Scaling)**

* Scales values between **0 and 1**.
* Formula



Code:

import pandas as pd

from sklearn.preprocessing import StandardScaler, MinMaxScaler

# Sample DataFrame for Standardization and Min-Max Scaling df = pd.DataFrame({

'Years of Experience': [1, 3, 5, 7, 9],

'Salary': [20000, 40000, 60000, 80000, 100000]

})

# Applying Standardization (Z-score normalization) scaler = StandardScaler()

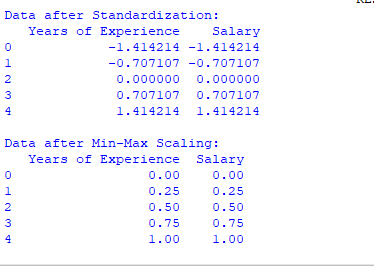
df[['Years of Experience', 'Salary']] = scaler.fit\_transform(df[['Years of Experience','Salary']]) print("Data after Standardization:")

print(df)

# Applying Min-Max Scaling to 'Years of Experience' and 'Salary' minmax\_scaler = MinMaxScaler()

df[['Years of Experience', 'Salary']] = minmax\_scaler.fit\_transform(df[['Years of Experience', 'Salary']])

print("\nData after Min-Max Scaling:") print(df)



**3 B) Perform feature Dummification to convert categorical variables into numerical representations.**

**Dummification (One-Hot Encoding for Categorical Variables)**

Dummification converts categorical variables into numerical form using **One-Hot Encoding**. Code:

import pandas as pd

# Sample DataFrame with categorical column

df = pd.DataFrame({'Job Role': ['Manager', 'Analyst', 'Engineer', 'Analyst', 'Manager'], 'Department': ['HR', 'Finance', 'Engineering', 'Finance', 'HR']})

# Perform feature dummification (One-Hot Encoding)

df\_encoded = pd.get\_dummies(df, columns=['Job Role', 'Department'], drop\_first=False) # Convert boolean values to integers explicitly (0 and 1)

df\_encoded = df\_encoded.astype(int) # Display the dummified DataFrame

print("Data after Dummification (One-Hot Encoding):") print(df\_encoded)

