exp 5

Experiment Number: 6 (or next in your sequence)

AIM: To apply different data normalization techniques (Z-score, Min-Max, Max Abs, Robust Scaling) to a real-world dataset and compare their effects on the distribution of numerical features.

Objectives:

- Understand the concept of data normalization and its importance in machine learning.
- Learn about different normalization methods: Z-score, Min-Max, Max Abs, and Robust Scaling.
- Apply these methods to a dataset using Scikit-learn.
- Analyze and compare the effects of each technique on the data distribution.

Course Outcomes: CO3, CO5 (Related to data preprocessing, feature scaling, and model performance)

Resource/Tools: Jupyter Notebook/Python with Pandas, Scikit-learn, and Matplotlib libraries.

Theory:

Data normalization is a crucial preprocessing step that transforms numerical features to a common scale. This is important because:

- **Improved Model Performance:** Many machine learning algorithms (e.g., k-nearest neighbors, support vector machines) are sensitive to the scale of features. Normalization prevents features with larger values from dominating the learning process.
- **Faster Convergence:** Normalization can help optimization algorithms converge faster during model training.
- **Reduced Numerical Instability:** It can prevent numerical instability issues that might arise when dealing with features having vastly different scales.

Normalization Techniques:

- **Z-score Standardization:** Transforms data to have zero mean and unit variance.
  - Formula: `standardized_value = (value - mean) / std_dev`
- **Min-Max Scaling:** Scales data to a specific range, usually 0 to 1.
  - Formula: `normalized_value = (value - min) / (max - min)`
- **Max Abs Scaling:** Scales data by its maximum absolute value.

    - Formula: `scaled_value = value / max_abs`
- **Robust Scaling:** Uses the median and interquartile range (IQR) to scale features, making it less sensitive to outliers.
    - Formula: `robust_scaled_value = (value - median) / IQR`

Code:

Python

```
import pandas as pd
from sklearn.preprocessing import StandardScaler, MinMaxScaler,
MaxAbsScaler, RobustScaler
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv('titanic.csv')

# Select numerical features for scaling (example)
numerical_features = ['Age', 'Fare']

# Create a DataFrame with only the selected features
df_num = data[numerical_features].copy()

# 1. Z-score Standardization
scaler = StandardScaler()
df_num['Age_zscore'] = scaler.fit_transform(df_num[['Age']])
df_num['Fare_zscore'] = scaler.fit_transform(df_num[['Fare']])

# 2. Min-Max Scaling
scaler = MinMaxScaler()
df_num['Age_minmax'] = scaler.fit_transform(df_num[['Age']])
df_num['Fare_minmax'] = scaler.fit_transform(df_num[['Fare']])

# 3. Max Abs Scaling
scaler = MaxAbsScaler()
df_num['Age_maxabs'] = scaler.fit_transform(df_num[['Age']])
df_num['Fare_maxabs'] = scaler.fit_transform(df_num[['Fare']])

# 4. Robust Scaling
scaler = RobustScaler()
df_num['Age_robust'] = scaler.fit_transform(df_num[['Age']])
df_num['Fare_robust'] = scaler.fit_transform(df_num[['Fare']])
```

```
# Visualize the effects (example - histograms)
plt.figure(figsize=(15, 10))

plt.subplot(2, 4, 1)
df_num['Age'].hist()
plt.title('Original Age')

plt.subplot(2, 4, 2)
df_num['Age_zscore'].hist()
plt.title('Z-score')

plt.subplot(2, 4, 3)
df_num['Age_minmax'].hist()
plt.title('Min-Max')

plt.subplot(2, 4, 4)
df_num['Age_maxabs'].hist()
plt.title('Max Abs')

plt.subplot(2, 4, 5)
df_num['Fare'].hist()
plt.title('Original Fare')

plt.subplot(2, 4, 6)
df_num['Fare_zscore'].hist()
plt.title('Z-score')

plt.subplot(2, 4, 7)
df_num['Fare_minmax'].hist()
plt.title('Min-Max')

plt.subplot(2, 4, 8)
df_num['Fare_robust'].hist()
plt.title('Robust')

plt.tight_layout()
plt.show()
```

Output:

The code will generate histograms showing the original and scaled distributions of the `Age` and `Fare` features, allowing you to compare the effects of each normalization method.

Result:

The experiment demonstrates the application of different data normalization techniques. By analyzing the histograms, you can observe how each method affects the data distribution:

- **Z-score:** Centers the data around zero with unit variance.
- **Min-Max:** Scales data to a 0-1 range.
- **Max Abs:** Divides each value by the maximum absolute value.
- **Robust:** Less sensitive to outliers due to the use of median and IQR.

Learning Outcomes:

- Gained practical experience in applying various data normalization techniques.
- Understood the differences between these methods and their impact on data distribution.
- Developed skills in using Scikit-learn for data preprocessing.
- Enhanced understanding of how feature scaling can affect machine learning model performance.