Practical – 3

Aim: To apply the feature scaling techniques for normalizing the data.

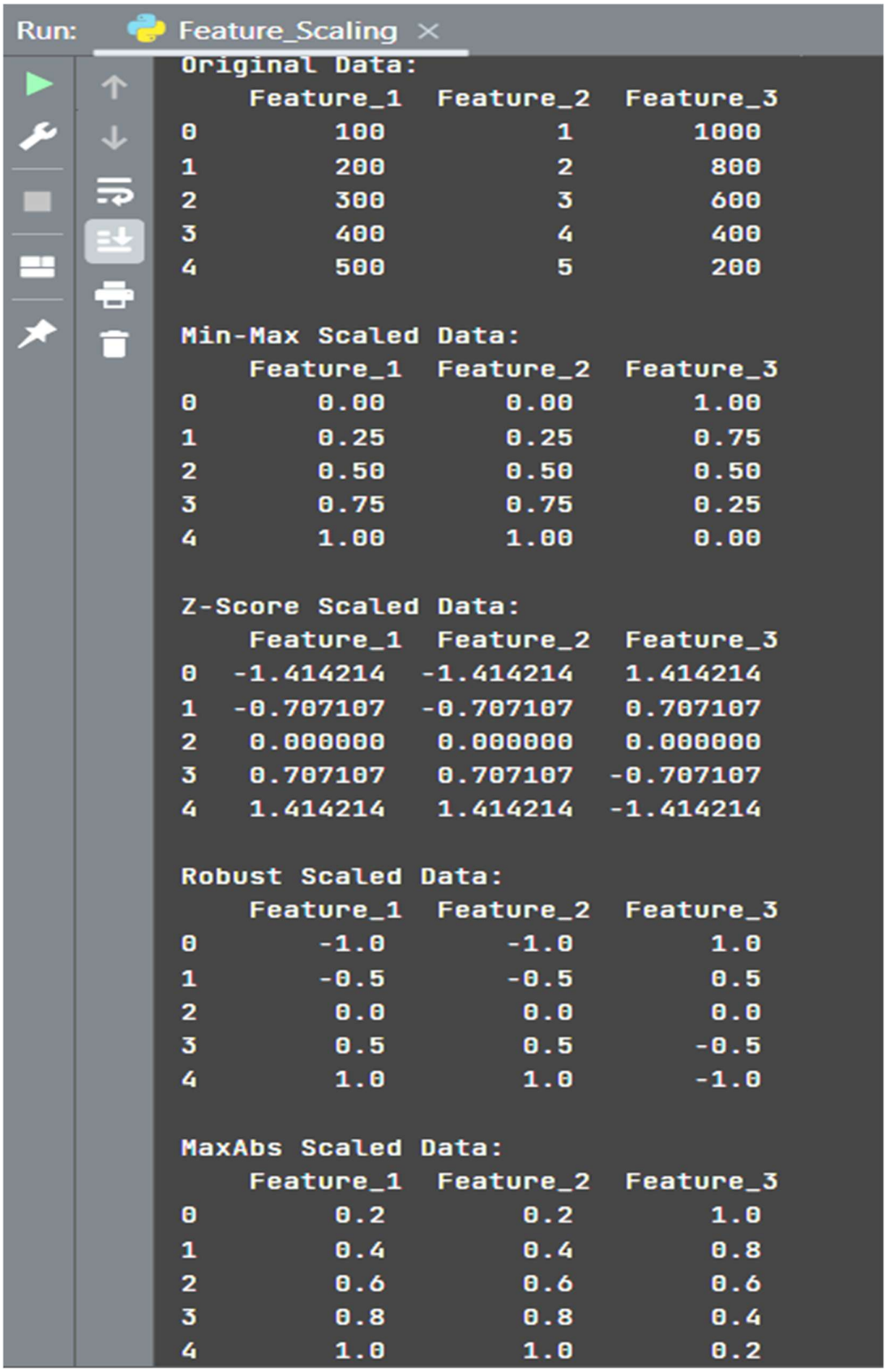
- **Solution:**

Feature_Scaling.py

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler, MaxAbsScaler
# Sample dataset
data = {
  'Feature_1': [100, 200, 300, 400, 500],# Higher range
  'Feature_2': [1, 2, 3, 4, 5],# Smaller range
  'Feature_3': [1000, 800, 600, 400, 200]# Reverse range
}
df = pd.DataFrame(data)
print("Original Data:\n", df)
# 1. Min-Max Scaling
min_max_scaler = MinMaxScaler()
df_minmax = pd.DataFrame(min_max_scaler.fit_transform(df), columns=df.columns)
print("\nMin-Max Scaled Data:\n", df_minmax)
# 2. Z-Score Scaling (Standardization)
standard_scaler = StandardScaler()
df_standardized = pd.DataFrame(standard_scaler.fit_transform(df), columns=df.columns)
print("\nZ-Score Scaled Data:\n", df_standardized)
# 3. Robust Scaling
robust_scaler = RobustScaler()
df_robust = pd.DataFrame(robust_scaler.fit_transform(df), columns=df.columns)
print("\nRobust Scaled Data:\n", df_robust)
# 4. MaxAbs Scaling
maxabs_scaler = MaxAbsScaler()
df_maxabs = pd.DataFrame(maxabs_scaler.fit_transform(df), columns=df.columns)
print("\nMaxAbs Scaled Data:\n", df_maxabs)

```

- **Output:**


Run:
Feature_Scaling
Original Data:
Feature_1 Feature_2 Feature_3
0 100 1 1000
1 200 2 800
2 300 3 600
3 400 4 400
4 500 5 200
Min-Max Scaled Data:
Feature_1 Feature_2 Feature_3
0 0.00 0.00 1.00
1 0.25 0.25 0.75
2 0.50 0.50 0.50
3 0.75 0.75 0.25
4 1.00 1.00 0.00
Z-Score Scaled Data:
Feature_1 Feature_2 Feature_3
0 -1.414214 -1.414214 1.414214
1 -0.707107 -0.707107 0.707107
2 0.000000 0.000000 0.000000
3 0.707107 0.707107 -0.707107
4 1.414214 1.414214 -1.414214
Robust Scaled Data:
Feature_1 Feature_2 Feature_3
0 -1.0 -1.0 1.0
1 -0.5 -0.5 0.5
2 0.0 0.0 0.0
3 0.5 0.5 -0.5
4 1.0 1.0 -1.0
MaxAbs Scaled Data:
Feature_1 Feature_2 Feature_3
0 0.2 0.2 1.0
1 0.4 0.4 0.8
2 0.6 0.6 0.6
3 0.8 0.8 0.4
4 1.0 1.0 0.2

Practical – 4

Aim: To apply the feature scaling techniques for normalizing the data.

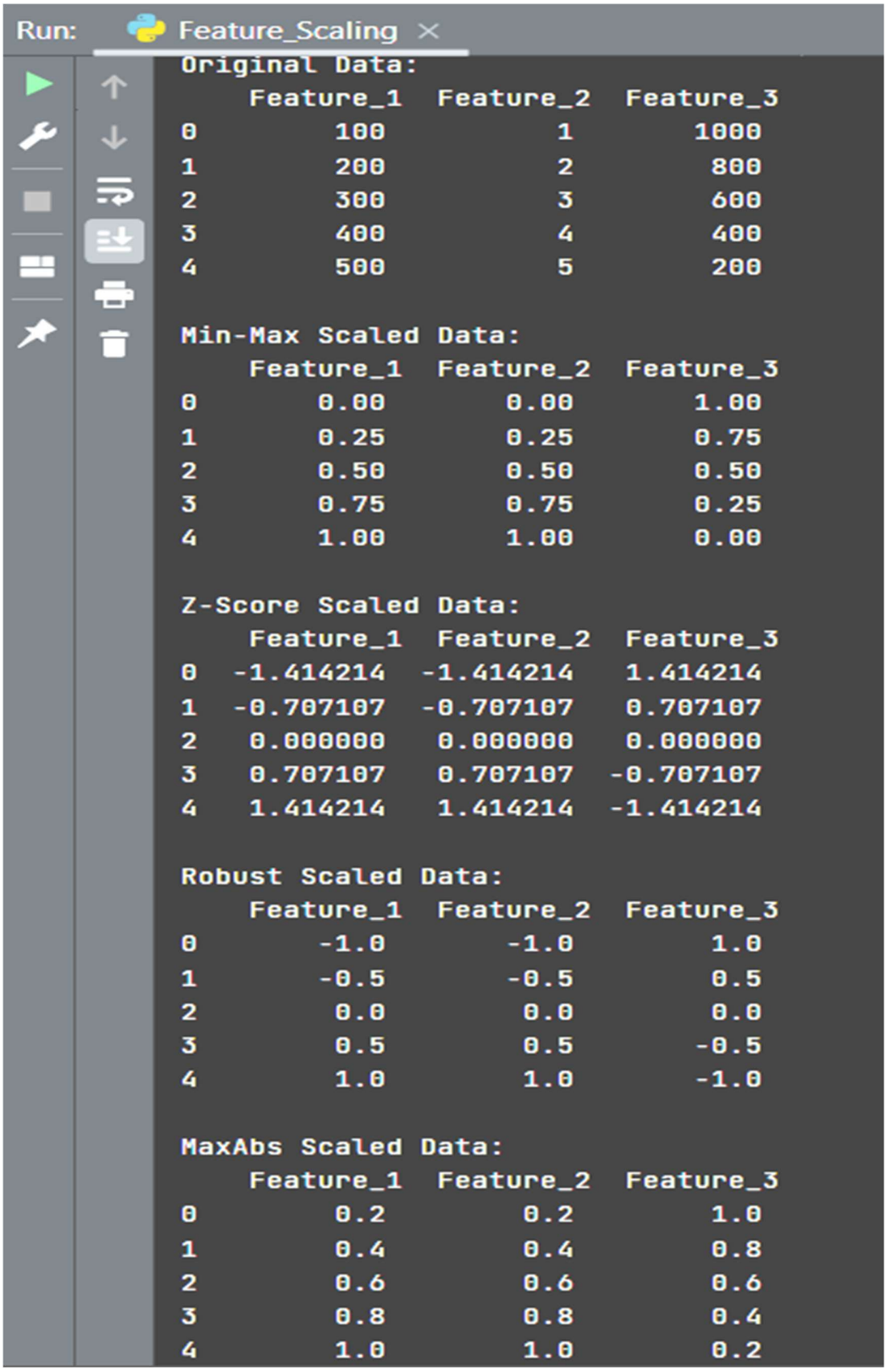
- **Solution:**

Feature_Scaling.py

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler, MaxAbsScaler
# Sample dataset
data = {
  'Feature_1': [100, 200, 300, 400, 500],# Higher range
  'Feature_2': [1, 2, 3, 4, 5],# Smaller range
  'Feature_3': [1000, 800, 600, 400, 200]# Reverse range
}
df = pd.DataFrame(data)
print("Original Data:\n", df)
# 1. Min-Max Scaling
min_max_scaler = MinMaxScaler()
df_minmax = pd.DataFrame(min_max_scaler.fit_transform(df), columns=df.columns)
print("\nMin-Max Scaled Data:\n", df_minmax)
# 2. Z-Score Scaling (Standardization)
standard_scaler = StandardScaler()
df_standardized = pd.DataFrame(standard_scaler.fit_transform(df), columns=df.columns)
print("\nZ-Score Scaled Data:\n", df_standardized)
# 3. Robust Scaling
robust_scaler = RobustScaler()
df_robust = pd.DataFrame(robust_scaler.fit_transform(df), columns=df.columns)
print("\nRobust Scaled Data:\n", df_robust)
# 4. MaxAbs Scaling
maxabs_scaler = MaxAbsScaler()
df_maxabs = pd.DataFrame(maxabs_scaler.fit_transform(df), columns=df.columns)
print("\nMaxAbs Scaled Data:\n", df_maxabs)

```

- **Output:**


Run:
Feature_Scaling
Original Data:
Feature_1 Feature_2 Feature_3
0 100 1 1000
1 200 2 800
2 300 3 600
3 400 4 400
4 500 5 200
Min-Max Scaled Data:
Feature_1 Feature_2 Feature_3
0 0.00 0.00 1.00
1 0.25 0.25 0.75
2 0.50 0.50 0.50
3 0.75 0.75 0.25
4 1.00 1.00 0.00
Z-Score Scaled Data:
Feature_1 Feature_2 Feature_3
0 -1.414214 -1.414214 1.414214
1 -0.707107 -0.707107 0.707107
2 0.000000 0.000000 0.000000
3 0.707107 0.707107 -0.707107
4 1.414214 1.414214 -1.414214
Robust Scaled Data:
Feature_1 Feature_2 Feature_3
0 -1.0 -1.0 1.0
1 -0.5 -0.5 0.5
2 0.0 0.0 0.0
3 0.5 0.5 -0.5
4 1.0 1.0 -1.0
MaxAbs Scaled Data:
Feature_1 Feature_2 Feature_3
0 0.2 0.2 1.0
1 0.4 0.4 0.8
2 0.6 0.6 0.6
3 0.8 0.8 0.4
4 1.0 1.0 0.2