APPLIED DATA SCIENCE

ASSIGNMENT NO : 2

TITLE : Python for Data Handling: Normalization & Standardization.

NAME: Uday Annasaheb Thete

ROLL NO : 145

CLASS : TY-B

GITHUB LINK : https://github.com/uday6725/APPLIED_DATA_SCIENCE/tree/main/ASSIGN2

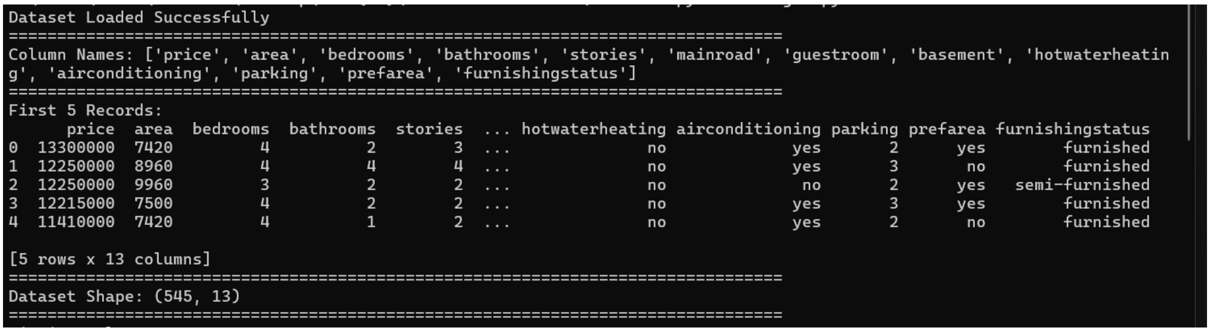
DATASET LINK: https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

CODE:

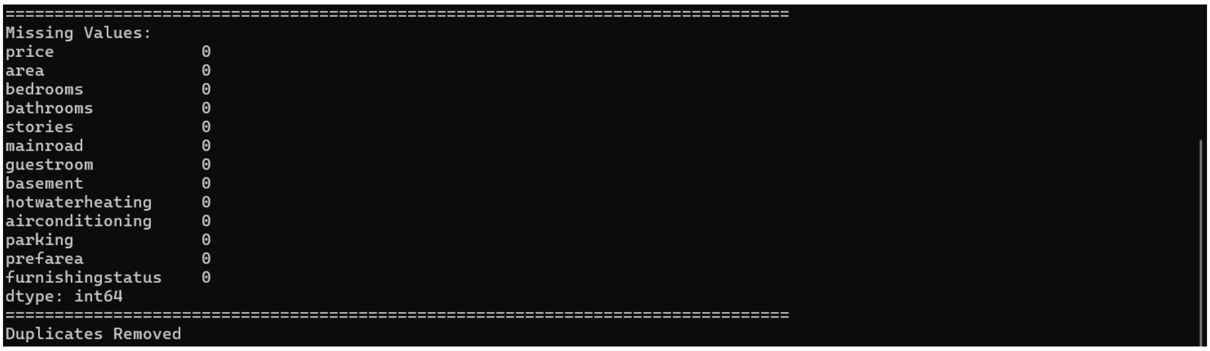
```
# ============================================================
# APPLIED DATA SCIENCE
# Assignment 2 – Normalization & Standardization
# Dataset: Housing Prices (Kaggle)
# ============================================================

# 1. Import Libraries
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler, StandardScaler

# ------------------------------------------------------------
# 2. Load Dataset
# ------------------------------------------------------------
df = pd.read_csv("Housing.csv")
print("Dataset Loaded Successfully")
print("=" * 80)

# ------------------------------------------------------------
# 3. Clean Column Names
# ------------------------------------------------------------
df.columns = df.columns.str.strip().str.lower()
```

```
print("Column Names:", df.columns.tolist())
print("=" * 80)

# --------------------------------------------------------
# 4. Basic Dataset Information
# --------------------------------------------------------
print("First 5 Records:")
print(df.head())
print("=" * 80)

print("Dataset Shape:", df.shape)
print("=" * 80)

print("Missing Values:")
print(df.isnull().sum())
print("=" * 80)

# --------------------------------------------------------
# 5. Remove Duplicates
# --------------------------------------------------------
df.drop_duplicates(inplace=True)
print("Duplicates Removed")
print("=" * 80)

# --------------------------------------------------------
# 6. Handle Missing Values
# --------------------------------------------------------
numerical_cols = df.select_dtypes(include=np.number).columns
df[numerical_cols] = df[numerical_cols].fillna(df[numerical_cols].mean())

cat_cols = df.select_dtypes(include=['object']).columns
for col in cat_cols:
    df[col] = df[col].fillna(df[col].mode()[0])

print("Missing Values Handled")
print("=" * 80)

# --------------------------------------------------------
# 7. Select Numerical Columns for Scaling
# --------------------------------------------------------
numeric_data = df.select_dtypes(include=np.number)

# --------------------------------------------------------
# 8. Normalization (Min-Max Scaling)
# --------------------------------------------------------
minmax = MinMaxScaler()
normalized = minmax.fit_transform(numeric_data)

normalized_df = pd.DataFrame(normalized, columns=numeric_data.columns)

print("Normalized Data (First 5 Rows):")
print(normalized_df.head())
print("=" * 80)

# --------------------------------------------------------
# 9. Standardization (Z-Score Scaling)
# --------------------------------------------------------
standard = StandardScaler()
standardized = standard.fit_transform(numeric_data)
standardized_df = pd.DataFrame(standardized, columns=numeric_data.columns)
print("Standardized Data (First 5 Rows):")
print(standardized_df.head())

print("Data Handling Completed Successfully")
```

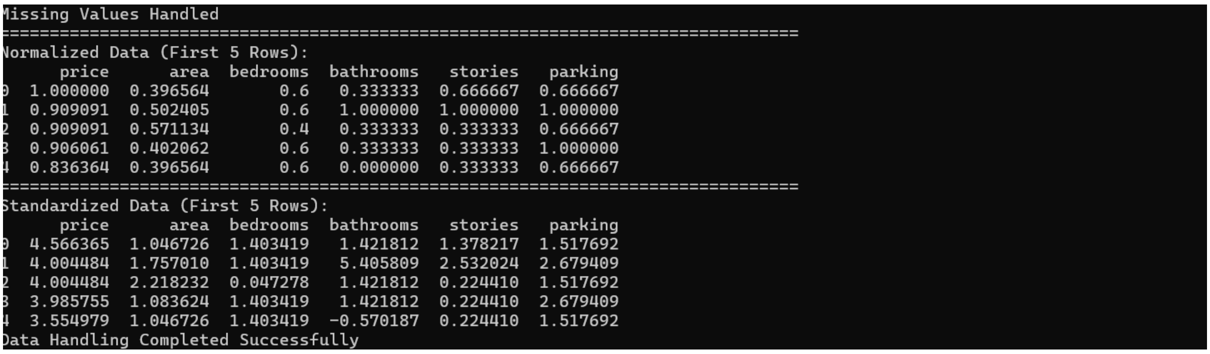
OUTPUT:



```
Dataset Loaded Successfully
==============================================================================
Column Names: ['price', 'area', 'bedrooms', 'bathrooms', 'stories', 'mainroad', 'guestroom', 'basement', 'hotwaterheatin
g', 'airconditioning', 'parking', 'prefarea', 'furnishingstatus']
==============================================================================
First 5 Records:
      price  area  bedrooms  bathrooms  stories  ... hotwaterheating airconditioning parking prefarea furnishingstatus
0  13300000  7420         4          2        3  ...              no             yes       2      yes        furnished
1  12250000  8960         4          4        4  ...              no             yes       3       no        furnished
2  12250000  9960         3          2        2  ...              no              no       2      yes   semi-furnished
3  12215000  7500         4          2        2  ...              no             yes       3      yes        furnished
4  11410000  7420         4          1        2  ...              no             yes       2       no        furnished

[5 rows x 13 columns]
==============================================================================
Dataset Shape: (545, 13)
==============================================================================
```



```
==============================================================================
Missing Values:
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
hotwaterheating     0
airconditioning     0
parking             0
prefarea            0
furnishingstatus    0
dtype: int64
==============================================================================
Duplicates Removed
```



```
Missing Values Handled
==============================================================================
Normalized Data (First 5 Rows):
      price      area  bedrooms  bathrooms   stories   parking
0  1.000000  0.396564       0.6   0.333333  0.666667  0.666667
1  0.909091  0.502405       0.6   1.000000  1.000000  1.000000
2  0.909091  0.571134       0.4   0.333333  0.333333  0.666667
3  0.906061  0.402062       0.6   0.333333  0.333333  1.000000
4  0.836364  0.396564       0.6   0.000000  0.333333  0.666667
==============================================================================
Standardized Data (First 5 Rows):
      price      area  bedrooms  bathrooms   stories   parking
0  4.566365  1.046726  1.403419   1.421812  1.378217  1.517692
1  4.004484  1.757010  1.403419   5.405809  2.532024  2.679409
2  4.004484  2.218232  0.047278   1.421812  0.224410  1.517692
3  3.985755  1.083624  1.403419   1.421812  0.224410  2.679409
4  3.554979  1.046726  1.403419  -0.570187  0.224410  1.517692
Data Handling Completed Successfully
```

INTERPRETATION:

- ## Interpretation of Normalization
- Normalization scaled housing features like price, area and bedrooms into a range between 0 and 1.
  All attributes now have equal scale regardless of original units.
  No negative values were produced and data distribution remained unchanged.

- ## Interpretation of Standardization
- Standardization transformed housing data so that mean became 0 and standard deviation became 1.
  Values above average became positive and below average became negative.
  This helps compare how far each house feature lies from the average.