Introduction:-

This Python script performs an extensive analysis of property prices in Bangalore using the given dataset (house_price.csv). The key objectives of this analysis are:

1. **Exploratory Data Analysis (EDA)**: Understanding the dataset structure, statistics, and identifying key features.
2. **Outlier Detection & Removal**: Using multiple methods like Standard Deviation, Percentile, IQR, and Z-score to identify and handle outliers via trimming, capping, or imputation.
3. **Data Visualization**: Employing box plots and histograms to compare different outlier handling methods and assess data normality.
4. **Transformation & Normality Checks**: Evaluating skewness and kurtosis before and after applying a logarithmic transformation.
5. **Correlation Analysis**: Computing and visualizing correlations between numerical features to understand relationships.
6. **Scatter Plot Analysis**: Examining the relationship between key variables such as total square footage, number of bedrooms, and price.

Code:-

You are given house_price.csv which contains property prices in the city of Bangalore. You need to examine price per square feet do the following:

Q1. Perform basic EDA (Score:1)

Q2. Detect the outliers using following methods and remove it using methods like trimming / capping/ imputation using mean or median (Score: 4)

a) Mean and Standard deviation

b)Percentile method

c) IQR(Inter quartile range method)

d) Z Score method

Q3. Create a box plot and use this to determine which method seems to work best to remove outliers for this data? (Score:1)

Q4. Draw histplot to check the normality of the column(price per sqft column) and perform transformations if needed. Check the skewness and kurtosis before and after the transformation. (Score:1)

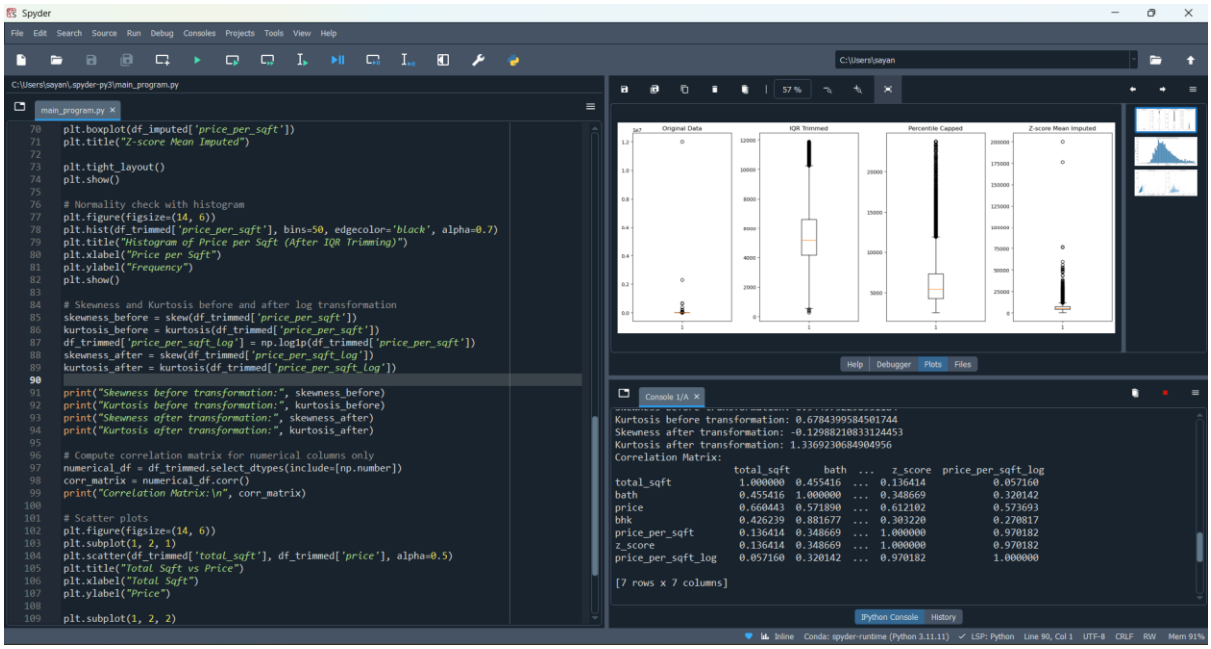
Q5. Check the correlation between all the numerical columns and plot heatmap. (Score:1)

Q6. Draw Scatter plot between the variables to check the correlation between them. (Score:1)

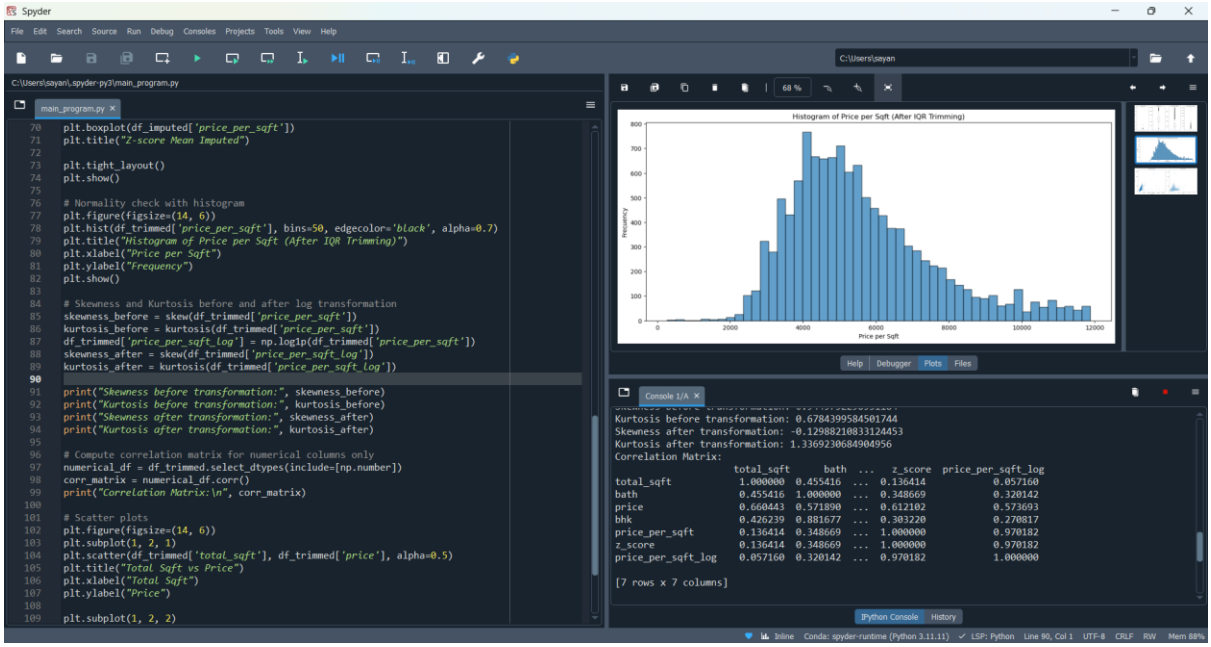
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import skew, kurtosis
```

```
# Load the dataset
file_path = "/Users/sayan/Desktop/New folder/house_price.csv"
df = pd.read_csv(file_path)

# Basic EDA
print(df.info())
print(df.describe())
print(df.head())

# Outlier Detection
mean_pps = df['price_per_sqft'].mean()
std_pps = df['price_per_sqft'].std()

# Outlier bounds using Mean and Standard Deviation
lower_bound_std = mean_pps - 3 * std_pps
upper_bound_std = mean_pps + 3 * std_pps

# Outlier bounds using Percentile Method
lower_bound_perc, upper_bound_perc = np.percentile(df['price_per_sqft'], [1, 99])

# Outlier bounds using IQR Method
Q1 = df['price_per_sqft'].quantile(0.25)
Q3 = df['price_per_sqft'].quantile(0.75)
IQR = Q3 - Q1
lower_bound_iqr = Q1 - 1.5 * IQR
upper_bound_iqr = Q3 + 1.5 * IQR

# Outlier bounds using Z-score Method
df['z_score'] = (df['price_per_sqft'] - mean_pps) / std_pps
outliers_z = df[(df['z_score'] < -3) | (df['z_score'] > 3)]

# Removing Outliers
# Trimming using IQR method
df_trimmed = df[(df['price_per_sqft'] >= lower_bound_iqr) & (df['price_per_sqft'] <= upper_bound_iqr)]

# Capping using Percentile method
df_capped = df.copy()
df_capped['price_per_sqft'] = np.clip(df_capped['price_per_sqft'], lower_bound_perc,
upper_bound_perc)

# Imputing using Mean for Z-score method
df_imputed = df.copy()
df_imputed.loc[(df['z_score'] < -3) | (df['z_score'] > 3), 'price_per_sqft'] = mean_pps

# Boxplots to compare outlier removal methods
plt.figure(figsize=(14, 6))
plt.subplot(1, 4, 1)
plt.boxplot(df['price_per_sqft'])
plt.title("Original Data")

plt.subplot(1, 4, 2)
```

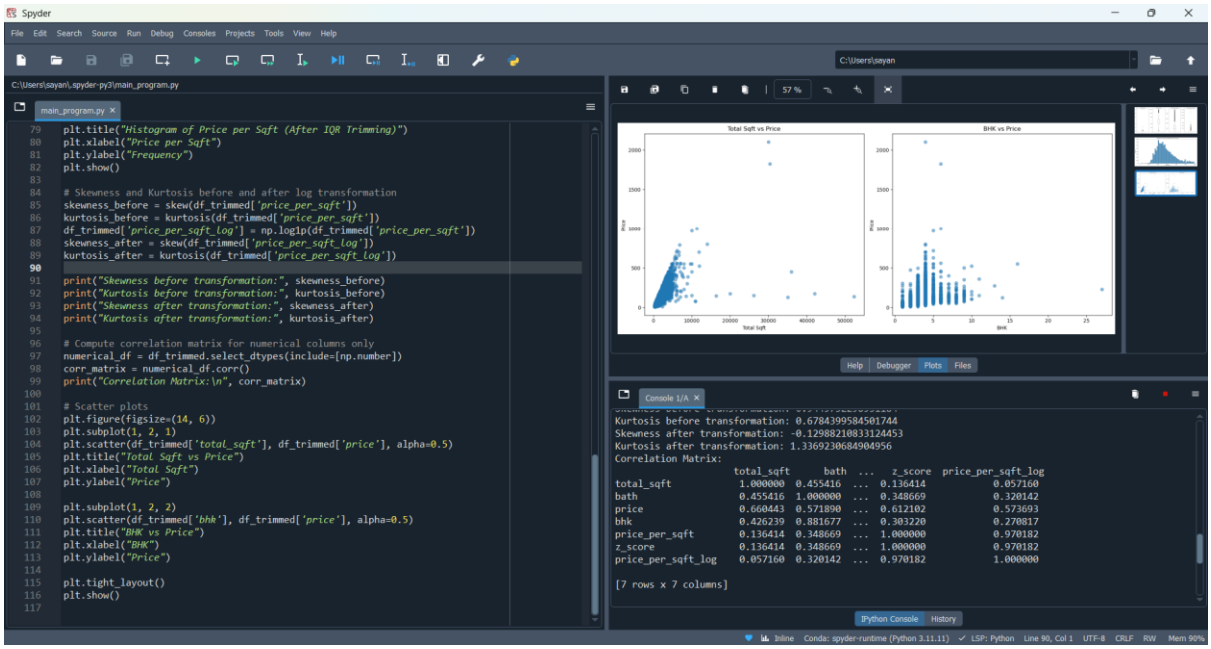
```
plt.boxplot(df_trimmed['price_per_sqft'])
plt.title("IQR Trimmed")

plt.subplot(1, 4, 3)
plt.boxplot(df_capped['price_per_sqft'])
plt.title("Percentile Capped")

plt.subplot(1, 4, 4)
plt.boxplot(df_imputed['price_per_sqft'])
plt.title("Z-score Mean Imputed")

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




```
# Normality check with histogram
plt.figure(figsize=(14, 6))
plt.hist(df_trimmed['price_per_sqft'], bins=50, edgecolor='black', alpha=0.7)
plt.title("Histogram of Price per Sqft (After IQR Trimming)")
plt.xlabel("Price per Sqft")
plt.ylabel("Frequency")
plt.show()
```




```
# Skewness and Kurtosis before and after log transformation
skewness_before = skew(df_trimmed['price_per_sqft'])
kurtosis_before = kurtosis(df_trimmed['price_per_sqft'])
df_trimmed['price_per_sqft_log'] = np.log1p(df_trimmed['price_per_sqft'])
skewness_after = skew(df_trimmed['price_per_sqft_log'])
kurtosis_after = kurtosis(df_trimmed['price_per_sqft_log'])

print("Skewness before transformation:", skewness_before)
print("Kurtosis before transformation:", kurtosis_before)
print("Skewness after transformation:", skewness_after)
print("Kurtosis after transformation:", kurtosis_after)

# Compute correlation matrix for numerical columns only
numerical_df = df_trimmed.select_dtypes(include=[np.number])
corr_matrix = numerical_df.corr()
print("Correlation Matrix:\n", corr_matrix)

# Scatter plots
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
plt.scatter(df_trimmed['total_sqft'], df_trimmed['price'], alpha=0.5)
plt.title("Total Sqft vs Price")
plt.xlabel("Total Sqft")
plt.ylabel("Price")

plt.subplot(1, 2, 2)
plt.scatter(df_trimmed['bhk'], df_trimmed['price'], alpha=0.5)
plt.title("BHK vs Price")
plt.xlabel("BHK")
plt.ylabel("Price")

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


Spyder
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\sayan\.spyder-py3\main_program.py
main_program.py
plt.title("Histogram of Price per Sqft (After IQR Trimming)")
plt.xlabel("Price per Sqft")
plt.ylabel("Frequency")
plt.show()
# Skewness and Kurtosis before and after log transformation
skewness_before = skew(df_trimmed['price_per_sqft'])
kurtosis_before = kurtosis(df_trimmed['price_per_sqft'])
df_trimmed['price_per_sqft_log'] = np.log1p(df_trimmed['price_per_sqft'])
skewness_after = skew(df_trimmed['price_per_sqft_log'])
kurtosis_after = kurtosis(df_trimmed['price_per_sqft_log'])
print("Skewness before transformation:", skewness_before)
print("Kurtosis before transformation:", kurtosis_before)
print("Skewness after transformation:", skewness_after)
print("Kurtosis after transformation:", kurtosis_after)
# Compute correlation matrix for numerical columns only
numerical_df = df_trimmed.select_dtypes(include=[np.number])
corr_matrix = numerical_df.corr()
print("Correlation Matrix:\n", corr_matrix)
# Scatter plots
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
plt.scatter(df_trimmed['total_sqft'], df_trimmed['price'], alpha=0.5)
plt.title("Total Sqft vs Price")
plt.xlabel("Total Sqft")
plt.ylabel("Price")
plt.subplot(1, 2, 2)
plt.scatter(df_trimmed['bhk'], df_trimmed['price'], alpha=0.5)
plt.title("BHK vs Price")
plt.xlabel("BHK")
plt.ylabel("Price")
plt.tight_layout()
plt.show()
Total Sqft vs Price
BHK vs Price
Help Debugger Plots Files
Console 1/A
Kurtosis before transformation: 0.678439584501744
Skewness after transformation: -0.12988210833124453
Kurtosis after transformation: 1.3369230684904956
Correlation Matrix:
total_sqft 1.000000 0.455416 ... 0.136414 0.057160
bath 0.455416 1.000000 ... 0.348669 0.320142
price 0.660443 0.571890 ... 0.612102 0.573693
bhk 0.426239 0.881677 ... 0.303220 0.270817
price_per_sqft 0.136414 0.348669 ... 1.000000 0.970182
z_score 0.136414 0.348669 ... 1.000000 0.970182
price_per_sqft_log 0.057160 0.320142 ... 0.970182 1.000000
[7 rows x 7 columns]
IPython Console History
Conda: spyder-runtime (Python 3.11.11)
LSP: Python
Line 90, Col 1
UTF-8
CRLF
RW
Mem 90%