REPORT AND ANALYSIS

INTRODUCTION:

The following is the report on how missing values and outliers were handled in the Gurgaon Real Estate data.

DATA EXPLORATION:

The data had features related to properties in Gurgaon real estate that help us predict the price of the properties.

From initial exploration, we found 23 columns in the dataset, out of which 16 were numerical columns and 7 were categorical using the describe function, and a total of 3803 rows.

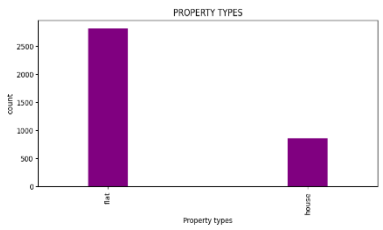
We dropped duplicated values using the drop_duplicates command, after which the number of rows dropped to 3677.

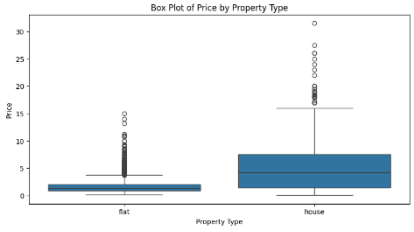
COLUMN WISE PREPROSESSING:

1. **PROPERTY TYPE COLUMN: (categorical column)**

Through exploration using ". describe () "function and "value_counts() "function, which counts the occurrence of each category, we found that property type has 2 unique categories, flats and house.

From the graph, we can see that there are approximately 75% flats and 25 % houses. We dropped the houses from the data, to avoid bias.





Finding Missing values:

```
"missing_property= dataset["property_type"].isnull(). sum ()"
```

Using this command, we discovered the property type column has no missing values.

This graph shows the number of flats and the price range, most of the flats fall in the range 0-5 crores and houses from 0-10 crores.

box plot:

```
sns.boxplot(data=dataset, x='property_type', y='price')
```

Using sns box plot command to draw a graph between price and property type, also used to detect outliers.

From the plot we can infer that,

-The median price of houses is higher than the median price of flats

-There exist outliers for both flats and houses.

Detecting Outliers:

-Outliers are data points that are outside the range of the data set.

How outliers affect the data?

- The mean and standard deviation are pushed in the direction of outliers.

- Outliers overshadow relationships within the data.

Converting to numerical:

Since Property type is a categorical feature, we converted it to numerical using one hot encoding.

```
property_encoded = pd.get_dummies(dataset['property_type'],
```

One hot encoding creates a new binary column for each unique category in the original column.

2. <u>SOCIETY COLUMN:</u> (categorial column)

The output from the below code shows that society column has 676 unique categories.

```
society_categories = dataset['society'].nunique()
```

society with the greatest number of flats is tulip violet having 75 flats followed by ss the leaf

Society with the greatest number of properties is independent with 486 properties.

Binning

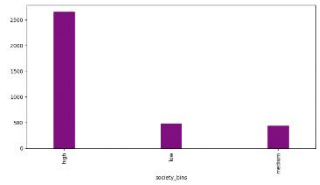
Binning is used in data pre-processing to convert continuous categorical features to categorical bins. This simplifies the model, improves its robustness.

For societies we calculate frequency of each category in society and divide frequencies into 4 bins. high with 2743 societies

low with 496 societies

medium with 437 societies

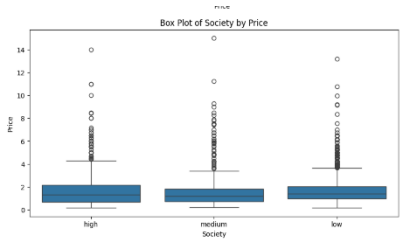
This graph shows the distribution of data among different society bins.



Top 10 societies are:

independent	435
tulip violet	75
ss the leaf	73
shapoorji pallonji joyville gurugram	42
dlf new town heights	42
signature global park	35
shree vardhman victoria	34
emaar mgf emerald floors premier	32
smart world orchard	32
dlf the ultima	31

This box plot shows that there are outliers in society column.



Handling Missing Values:

There is one missing value in society.

Since, society is a categorical feature, we can use mode to fill in the missing value.

One hot encoding for society

One hot encoding creates a new binary column for each unique bin in society in the original column.

3. <u>SECTOR COLUMN:</u> <u>(categorical)</u>

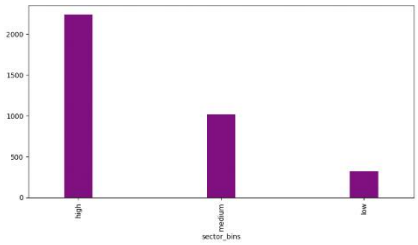
sector column has 676 unique categories, with the most frequent sector being Sohna road.

We binned sector column into 3 categories high, low and medium based on frequecy.

The output after binning tells us that the sector column contains more data with high frequency.

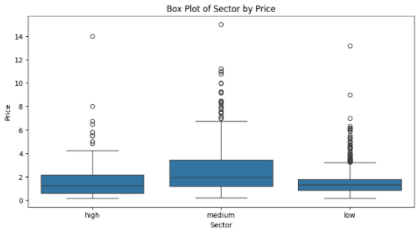
We can observe the same from the bar graph plotted with respect to the sector_bins

low bin indicates there are more sectors with less flats



Sector Vs Price

- **High Sector Bin:** This category includes prices that are considered high relative to the dataset being analysed. It represents the upper end of the price range and usually contains fewer data points, as higher prices are less common.
- **Mid Sector Bin:** The medium or mid sector bin encompasses prices that fall in the middle range. These are neither too high nor too low and represent an intermediate price level.
- **Low Sector Bin:** This bin contains the lowest prices in the dataset. It often has the highest frequency of data points, indicating that most of the prices are at the lower end of the spectrum.



- The **high sector** has a smaller IQR, suggesting less variability in prices, with no outliers.
- The **medium sector** has a larger IQR, indicating more variability, and several outliers suggest that there are prices significantly higher than the typical range.
- The **low sector** also has a larger IQR, with many outliers indicating higher prices.

The sector column has no missing values

ENCODING

We can use one hot encoding for sector bins, or we also use label encoding (which we used at end) to convert sector to a numerical column

4. <u>PRICE COLUMN:</u> (numerical)

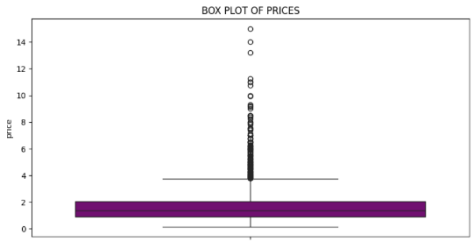
Price is the output column which we must predict based on given features.

Exploration

Description of the price column:

```
count    2817.000000
mean        1.713280
std         1.388532
min         0.160000
25%         0.900000
50%         1.380000
75%         2.050000
max        15.000000
```

This column is a numerical feature



The IQR is roughly between 0 and 2, which is the range of prices where most of the data is found. It is represented by the purple rectangle.

Median Price: The median price, which is almost 1, is shown by the black line inside the rectangle.

Outliers: The circles that are plotted above the rectangle, ranging from approximately 6 to over 14, show outlier prices that are noticeably higher than the majority.

Skewness & Kurtosis

Skewness: Quantifies a distribution's asymmetry. Negative skewness leans left, while positive skewness shows a tail that is expanding to the right. It aids in determining whether the data contains extreme values on one side or is distributed around the mean.

Kurtosis: The "peakedness" of a distribution in relation to a normal distribution is measured by kurtosis. Whereas low kurtosis denotes a flatter peak with lighter tails (fewer extreme values), high kurtosis indicates a sharper peak with heavier tails.

For continuous and large numeric datasets, we prefer using skewness and kurtosis to understand the distribution of the data and identify outliers.

Skewness= 3.3113346542178137

Kurtosis= 15.257818585808831

A skewness of 3.311 indicates a significant rightward skew, meaning the data is concentrated towards the lower values with a long tail on the higher end. A kurtosis of 15.258 suggests heavy-tailedness and a sharp peak, indicating the data has extreme values compared to a normal distribution. In summary, the distribution is highly skewed to the right with heavy tails and a sharp peak.

Missing Values

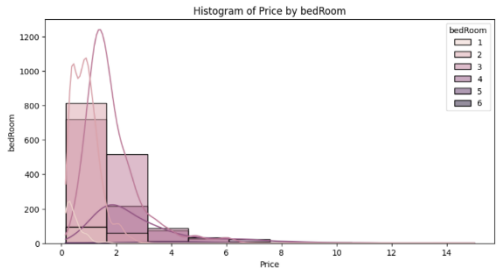
There is one missing value in this column.

Handling Missing value:

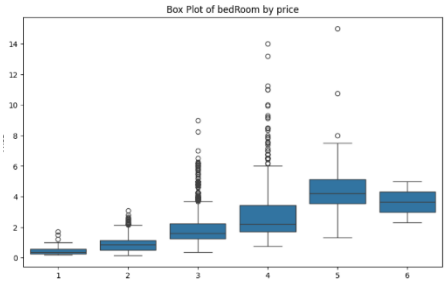
For each missing value in the "price" column, the KNN algorithm finds the 5 nearest neighbours based on the other columns in the data frame.

The missing value is then imputed with the average of the prices of the 5 nearest neighbours.

Plot a graph between price and bedrooms to compare if the price and the number of bedrooms have a linear relation.



- **One-bedroom properties**: These are most common at lower prices, with a high frequency around the price of 2.
- **Two-bedroom properties**: These are in frequency to one-bedroom properties but are less common.
- **Three or more bedrooms**: The frequency of properties with three or more bedrooms decreases and they appear mostly at higher prices.



1 Bedroom: Low pricing with few outliers, indicating that most 1-bedroom properties are affordable.

2 Bedrooms: A somewhat higher median price with more fluctuation and outliers, indicating a broader range of costs.

3 Bedrooms: A higher median price with significant fluctuation; multiple outliers indicate that some 3-bedroom residences are much more expensive.

Four bedrooms: The median price rises again, with fewer outliers than three bedrooms, indicating a more stable price range.

5 and 6 Bedrooms: Similar distributions but fewer data points, as evidenced by shorter boxes and lower variability, implying that these are less frequent on the market.

Outlier Detection

There are 188 outliers for this column with IQR score of 1.15.

Outlier Handling – Winsorization

Before winsorization there are more outliers in the range 6-30

And the outliers are denser in the price range somewhere from 6-13

We can also see that there are more outliers in the max price range which the box plot also indicates (before winsorization)

After using winserization the number of outliers decreased to 249, and the max price decreased to 5

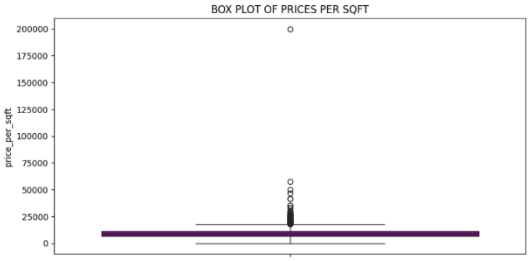
- **Interquartile Range (IQR)**: The central 50% of prices are between approximately **1.0 and 2.0**.
- **Median Price**: The median price is around **1.5**.
- **Price Variability**: The “whiskers” show that prices vary from below **1.0 to above 2.0**, excluding outliers.
- **Outliers**: Prices that are considered outliers are found after the value of **3.0** on the Price axis.

5.PRICE PER SQFT (NUMERICAL COLUMN)

EXPLORATION

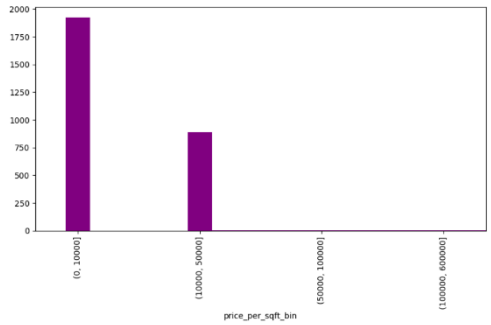
DESCRIPTION OF PRICE PER SQFT COLUMN

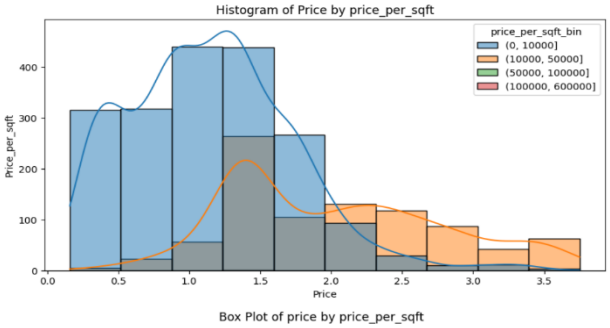
```
count      2817.000000
mean       9342.052183
std        5613.313741
min           4.000000
25%        6506.000000
50%        8347.000000
75%       11050.000000
max      200000.000000
Name: price_per_sqft, dtype: float64
```



The **(IQR)** tells that most prices per square foot are clustered near the lower end of the spectrum. The **Median Price** per square foot is within the lower range, as indicated by the line inside the box. There are several **outliers** at higher prices per square foot, shown as circles above the main plot.

On applying **BINNIG,** we found that most of the flats were in the range **`(0, 10000]`** which were **1923** in number. The same can be visualised using bar plot and histogram plot





A **SKEWNESS AND KURTOSIS** value of `476.8543922324539` indicates extreme outliers in this column.

The number of **missing values** is **one** in this column

Handling Missing value:

For each missing value in the "price_per_sqft" column, the KNN algorithm finds the 5 nearest neighbours based on the other columns in the data frame.

The missing value is then imputed with the average of the prices of the 5 nearest neighbours.

OUTLIERS

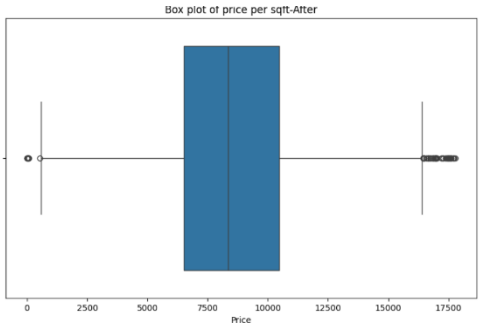
There are 100 outliers with an IQR score of 4539.25

The box plot before outlier handling shows that the **Median Price** is Around $50,000 and Most prices fall within the IQR and there are **High Price Outliers** with Several properties having a much higher price per square foot, indicated by the small circles to the right of the IQR

OUTLIERS DETECTION

We used **WINSORIZATION** for handling outliers and reduced the max price outlier from 20,000 to 17,777.

This can also be visualised in the box plot after winsorization below.



6. AREA COLUMN (NUMERICAL COLUMN)

EXPLORATION

DESCRIPTION OF PRICE PER SQFT COLUMN

```
dataset['area'].describe()
```

```
count      2817.000000
mean       2857.935747
std       25930.790790
min          72.000000
25%        1259.000000
50%        1671.000000
75%        2132.000000
max      875000.000000
Name: area, dtype: float64
```

We used binning to categorize data into 4- bins and found that most of the housing is in the area `(1000, 10000]` with a number of 2325 properties. By looking at the histogram lot by area we can also the same as properties in the area **(50, 100]** have a wide range of prices but are mostly concentrated around lower price points. We can further notice that as the area increased the price distribution widened indicating that prices increased with an increase in area (linear relationship). The same can be visualised in a scatter plot

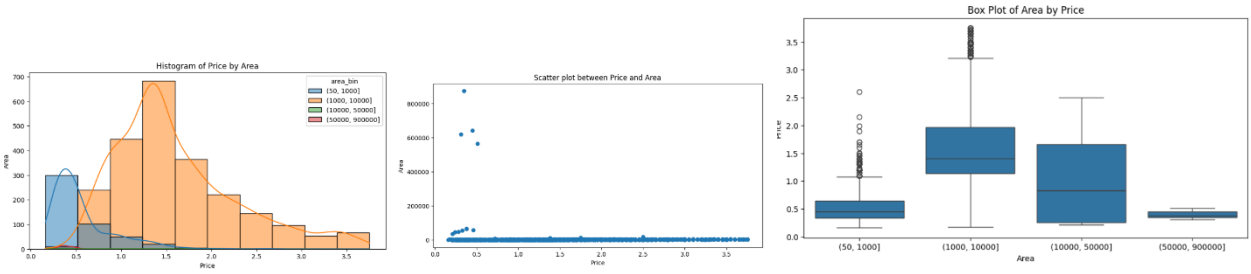
BOX-PLOT

(50, 1000]: Smaller areas with a certain range of prices and noticeable outliers.

(1000, 10000]: Mid-range areas with a wider distribution of prices and outliers.

(10000, 50000]: Larger areas with higher median prices.

(50000, 900000]: The largest areas with the highest prices and less variability.



The **SKEWNESS AND KURTOSIS** value of 778.1884813282747 indicates high deviations in the data distributions and the presence of outliers.

This column has **ONE MISSING VALUE**

Handling Missing value:

For each missing value in the "price_per_sqft" column, the KNN algorithm finds the 5 nearest neighbours based on the other columns in the data frame.

The missing value is then imputed with the average of the prices of the 5 nearest neighbours.

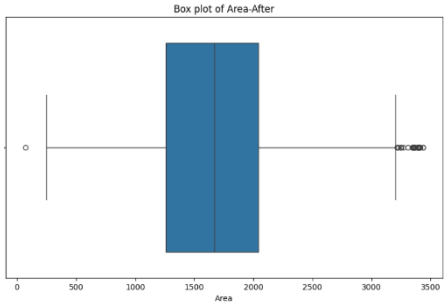
OUTLIERS

There are 104 outliers with IQR score of 872.75

The box plot before outlier handling tells that the **median area** is near the lower end as indicated by the line within the IQR. The **IQR** suggests that most data points are clustered near the end of the area scale within a small range and There are several **outliers** spread out at higher area values, indicating that there are areas significantly larger than the majority.

OUTLIER HANDLING

We used Winsorization to handle outliers and reduced the max area outlier from 87500 to 3434 which can be seen on the box plot of winsorization



7. AREA WITH TYPE COLUMN (CATEGORICAL COLUMN)

areaWithType

Super Built-up area 1950(181.16 sq.m.)Carpet area: 1161 sq.ft. (107.86 sq.m.) 17
Super Built up area 1578(146.6 sq.m.) 17
Super Built up area 1350(125.42 sq.m.) 15
Super Built up area 1650(153.29 sq.m.)Carpet area: 1022.58 sq.ft. (95 sq.m.) 14
Super Built up area 2010(186.74 sq.m.) 14
..
Carpet area: 448 (41.62 sq.m.) 1
Super Built up area 1568(145.67 sq.m.)Carpet area: 1200 sq.ft. (111.48 sq.m.) 1
Super Built up area 2132(198.07 sq.m.)Built Up area: 1800 sq.ft. (167.23 sq.m.)Carpet area: 1500 sq.ft. (139.35 sq.m.) 1
Super Built up area 5514(512.27 sq.m.) 1
Super Built up area 583(54.16 sq.m.)Carpet area: 483 sq.ft. (44.87 sq.m) 1

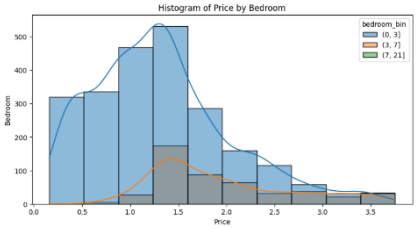
THERE ARE **NO MISSING VALUES** IN THIS COLUMN

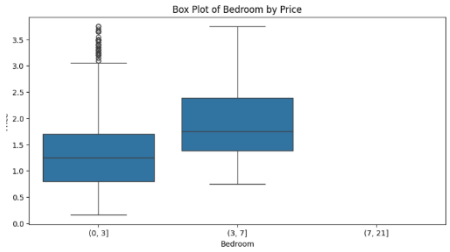
8. BEDROOM COLUMN (NUMERICAL COLUMN)

EXPLORATION

bedRoom
3 1367
2 874
4 453
1 94
5 28
6 2

On **BINNING** we found that most of the properties had bedrooms in the range (0,3] and categorised the column into 3-bins in ranges (0, 3] (3, 7] (7, 21]. **The histogram plot of price by bedroom** states that properties up to 3-bedrooms are more common at lower prices and the prices become higher for the bedrooms in the range (3,7] and the properties with bedrooms in range (7,21] are the most expensive. The box plot also gives the same insights with properties of range (0, 3] having median prices below 2 and rest of them having median prices above 2 and outliers at price above 3.5





THERE ARE **ZERO MISSING VALUES** IN THIS COLUMN.

OUTLIERS DETECTION

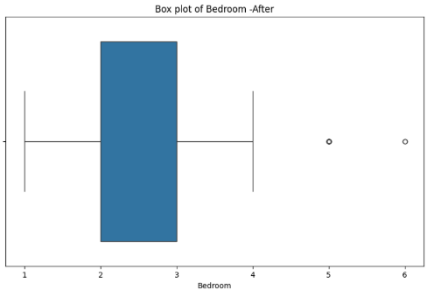
30 outliers were detected for this column with IQR score of 1.0

Before handling outliers, the **median** number of bedrooms is 4 and the data is concentrated **(IQR)** between 3&6 bedrooms and the variability seems to be about 2 to 7 bedrooms and several **outliers** are spread from 10-20 bedrooms.

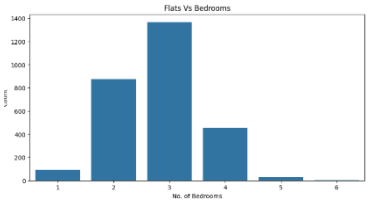
OUTLIER HANDLING

We used **TRIMMING** for handling outliers in this column since the outliers were evident from the raw data and had less outliers.

Trimming reduced the max bedroom outliers from 10-20 before handling to 6 after handling the outliers which can be seen in a box plot after TRIMMING.



The (flats vs bedroom) bar plot also indicates that majority of the flats have 3-bedrooms

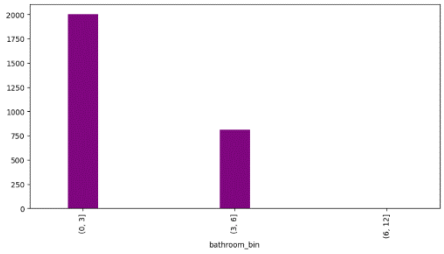


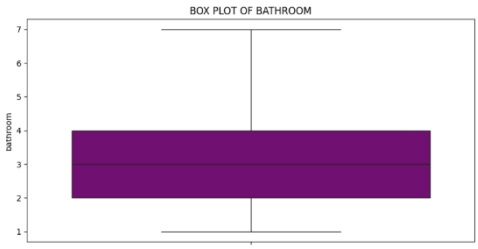
9.BATHROOM COLUMN (NUMERICAL)

Exploration

```
dataset['bathroom'].value_counts()
bathroom
2    961
3    939
4    610
5    162
1    104
6     38
7      4
```

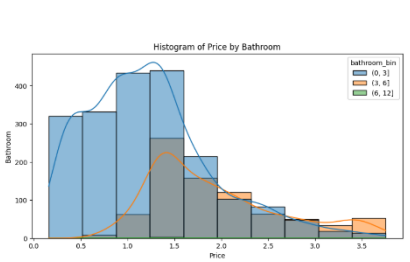
On **BINNING,** we categorised the bathroom into 3-bins where most of the properties were having bathrooms in the range (0,3] which are 2004 in number. A bar plot and a boxplot has been plotted to visualise the latter.

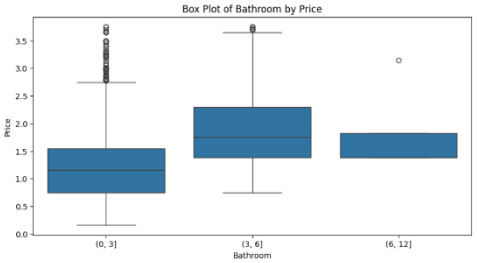




The histogram of bathrooms by price state that properties with (0,3] bathrooms are more common at lower prices for properties with (3,6] bathrooms are less common and have a wider distribution across various prices and the properties with (6,12] bathrooms are the least common and are highly priced.

The box plot of bathrooms by price state that the properties with (0,3] bathrooms have a median price around 1-1.5 and for the properties with (3,6] bathrooms have a median price around 1.75-2.25 and the highest above 2.25-3.5 for properties with (6,12] bathrooms which also has less variability. The box plot w.r.t the bathrooms says that anything priced above 3.5 is an outlier.

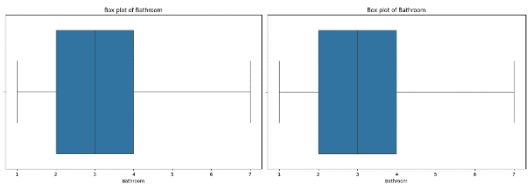




THERE ARE **NO MISSING VALUES** IN THIS COLUMN

THERE ARE **NO OUTLIERS** DETECTED FOR THIS COLUMN

The boxplot before and after handling the outliers is same as there are no outliers for this column and the max bathrooms are 7 and the median bathrooms are 3.

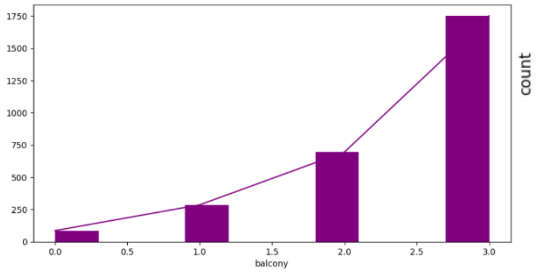


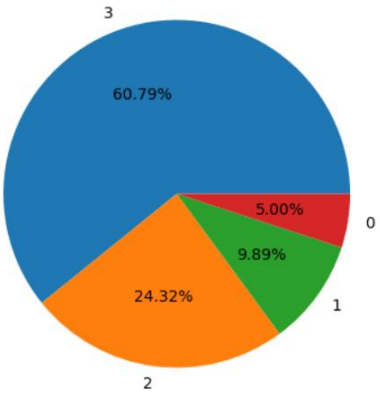
10.BALCONY COLUMN (NUMERICAL)

Exploration

```
dataset['balcony'].describe()
dataset['balcony'].value_counts()
balcony
3    1751
2     695
1     286
0      86
```

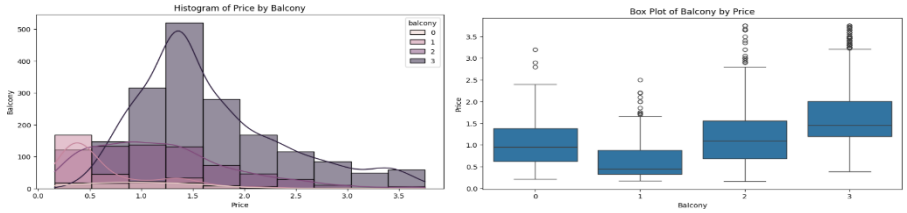
The pie chart and the bar graph indicate that most of the properties have 3-balconies and the same is stated by the describe function.





The same can be visualised by a box plot which states that the number of balconies lie between 2-3 and any property with zero balcony is considered as an outlier.

The histogram of price by balcony also states that properties with no balconies are the cheapest and the properties with 1 balcony show a slight increase in the price whereas properties with 2 balconies have higher prices and properties with 3 balconies have the highest price but not significantly different than those with 2 balconies.
The same is displayed by the box pot with some visible outliers for properties with 1 and 2 balconies which are priced above 3.
From the above we can deduce that the price is increasing with the increase in number of balconies but there is significant overlap among the different categories, indicating that the **number of balconies is not the sole determinant of property prices**.



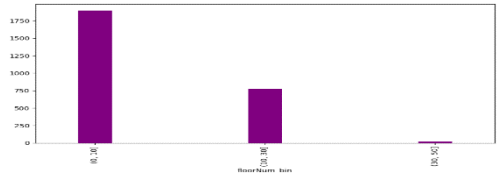
There are **NO MISSING VALUES** for this column.
We used **label encoding** to convert categorical data to numerical

11. FLOOR NUMBER COLUMN (NUMERICAL)

Exploration

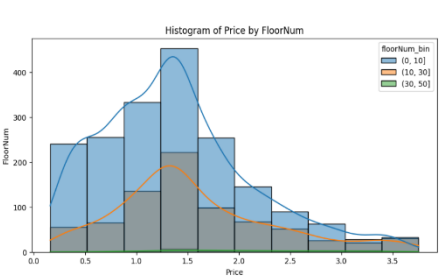
```
count    2816.000000
mean        7.997869
std         6.197977
min         0.000000
25%         3.000000
50%         7.000000
75%        11.000000
max        45.000000
```

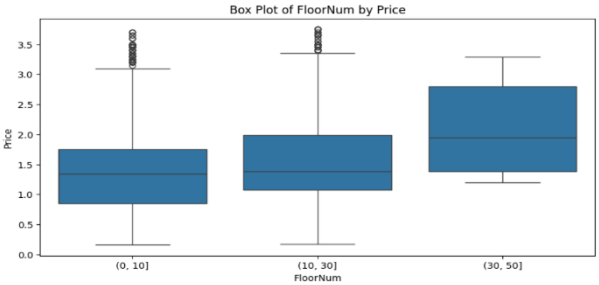
We applied **BINNING** to categorise the properties within a range of floor numbers to find out the maximum number of properties which fall under a certain range floor number and hence we found that most of the properties were falling in the range (0,10] which indicates that most of the properties were having floors in between 0-10 and they were 1896 in number. A bar plot to visualise the same.



The histogram of price by FloorNum states that **Properties on floors (0,10]** have a peak frequency at a price of approximately 1.0, suggesting this is the most common price range for these properties and **Properties on floors (10,30]** show a decrease in frequency as the price increases. **Properties on floors (30,50]** are less frequent across all price ranges, indicating they are less common in the dataset.

The box plot of price by FloorNum indicates that homes on higher floors tend to have higher median values and less price variety than those on lower levels. The lack of outliers in the highest floor range (30, 50] suggests that pricing for these properties is more stable, presumably due to a premium placed on higher-floor homes.





The **skewness and kurtosis** value of `3.1834433931327313` indicates that there is a moderate deviation in the distribution and the positive value indicates a positive skew with a sharper peak and heavier tails compared to normal distribution.

There are **TWO MISSING VALUES** in this column.

Handling Missing value:

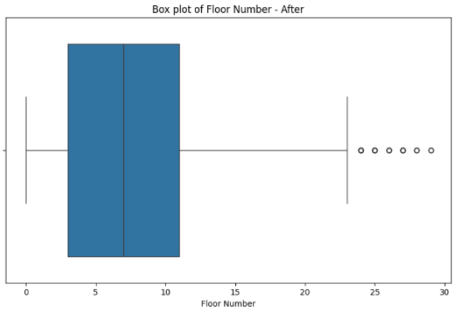
For each missing value in the "price_per_sqft" column, the KNN algorithm finds the 5 nearest neighbours based on the other columns in the data frame.

The missing value is then imputed with the average of the prices of the 5 nearest neighbours.

There are **64 OUTLIERS** in this column

We used **TRIMMING** to **handle the outliers**

The box plot **after handling the outliers** indicates that the IQR extends from approximately floor number 5 to 10. This means that the middle 50% of the data falls within this range. Whereas the median floor number is 6 or 7. The floor number 21-28 indicate outliers which are the floor numbers which are significantly higher than the rest.

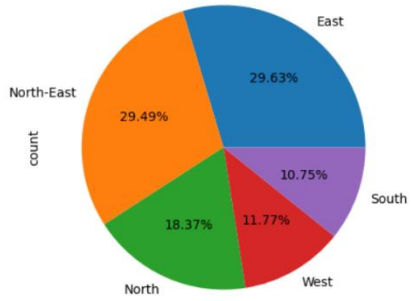


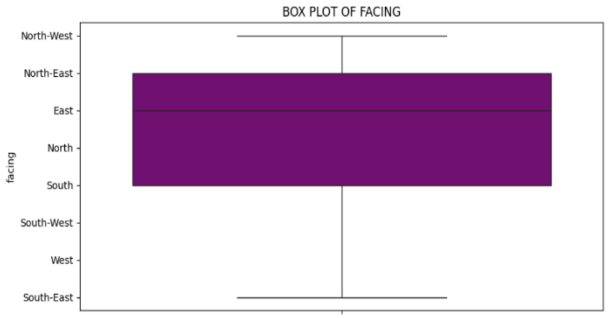
12.FACING COLUMN (CATEGORICAL FEATURE)

Exploration

```
North-East    477
East          456
North         286
South         198
West          178
North-West    153
South-East    141
South-West    130
```

The pie chart and box plot indicate that most of the properties are facing towards the EAST and many properties are facing in the directions north-east, east, north and south





The histogram and box plot of price by facing

North-West: On the off chance that the histogram shows a high recurrence of properties at lower price tags, it recommends that North-West-bound properties are more usually accessible at lower costs. On the other hand, if there are higher bars at more exorbitant cost places, it demonstrates a pattern towards more costly properties in this confronting.

North-East: A centralization of bars at specific cost ranges for North-East-bound properties would demonstrate that these costs are more common for this confronting. If the bars are fanned out, it recommends many costs for North-East-bound properties. East: The level of the bars for East-bound properties can perceive us which price tags are generally normal. If there's a top at a greater cost point, it proposes that East-bound properties will quite often be more costly.

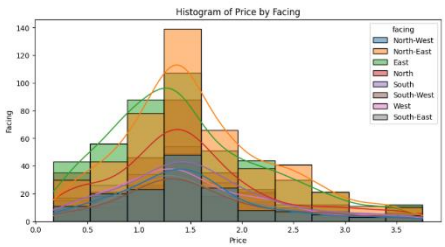
North: For North-bound properties, the dissemination of bars across the value hub will show whether there's a general pattern towards sequential costs, or on the other hand if the costs are equitably dispersed.

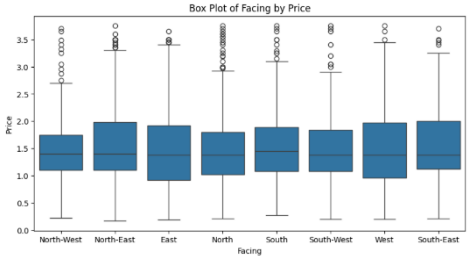
South: The histogram can uncover on the off chance that South-bound properties are by and large accessible at lower, centre, or more exorbitant cost focuses, contingent upon where the bars are concentrated.

South-West: Assuming that there are some South-West-bound properties at greater cost places, it could propose that these properties order a premium. Assuming the bars are lower, it could demonstrate that these properties are more reasonable.

West: The recurrence and level of the bars for West-bound properties will demonstrate the shared characteristic of specific price tags. A higher recurrence at a specific sticker cost proposes that it's a typical cost for West-bound properties.

South-East: The histogram will show if South-East-bound properties are even more regularly found at lower or greater cost focuses, in view of the level and spread of the bars in this class.





There are **774 MISSING VALUES** for this column

Handling Missing value:

For each missing value in the "price_per_sqft" column, the KNN algorithm finds the 5 nearest neighbours based on the other columns in the data frame.

The missing value is then imputed with the average of the prices of the 5 nearest neighbours.

We used **LABEL ENCODING** to convert the categorical data into numerical.

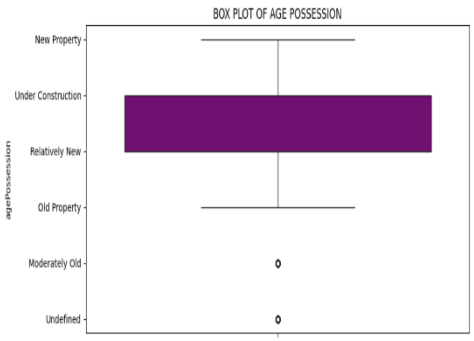
There are **178 outliers** with an IQR score of 2.0

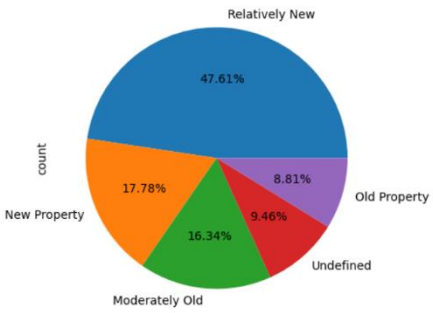
13. AGE POSSESSION COLUMN (CATEGORICAL FEATURE)

Exploration

```
Relatively New        1437
New Property           479
Moderately Old         335
Under Construction     243
Undefined              176
Old Property           123
```

The pie chart and box plot indicate that majority of the properties are **Relatively new, new property, and moderately old** we can also see this numerically through the describe function applied on the dataset on age possession column.



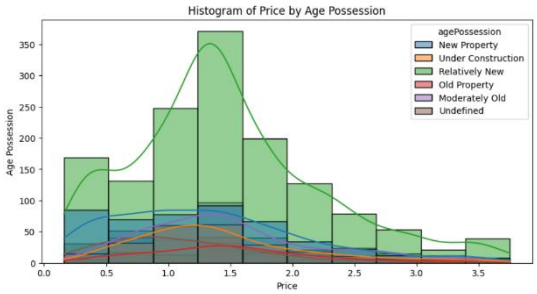


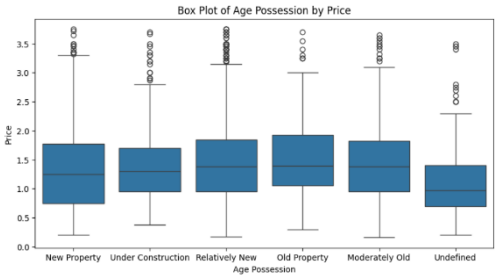
Z

THE (PRICE VS AGE) HISTOGRAM AND BOX PLOT

The **Under Construction s**hows how many properties are under construction at various prices. The **Relatively New category** has a high frequency in the mid-price range, suggesting that relatively new properties are commonly found at these prices. **Old Property Indicates** the distribution of older properties across the price spectrum and the **Moderately Old Represents** the frequency of moderately old properties, which span across various price ranges.

The overlaying green line graph peaking at a price of around 1.5 suggests that the most common price point for properties is in the mid-range, particularly for those that are relatively new.





There are **NO OUTLIERS** for this column

There are **NO MISSING VALUES** in this column

We used **LABEL ENCODING** to convert the categorical features to numerical

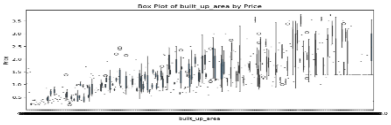
14. BUILT UP AREA COLUMN (NUMERICAL COLUMN)

Exploration

```
count       877.000000
mean       2549.531094
std       24844.683486
min          97.000000
25%        1285.000000
50%        1625.000000
75%        2000.000000
max      737147.000000
```

The describe function on the built-up area column shows that the maximum built area is 737147 which can be a potential outlier.

Through the boxplot of built-up area by price we can observe that the increase in built-up area caused an increase in the price where there are outliers at each price category

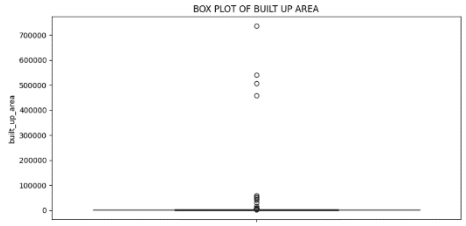


There are **1916 MISSING VALUES** in this column.

The code we used to **HANDLE MISSING VALUES** calculates median ratios between different types of property areas and uses these ratios to fill in missing values in the 'built_up_area' column of a dataset. It first updates

missing ‘built_up_area’ values using the ‘super_built_up_area’, and if any are still missing, it then uses the ‘carpet_area’ to estimate the missing values.

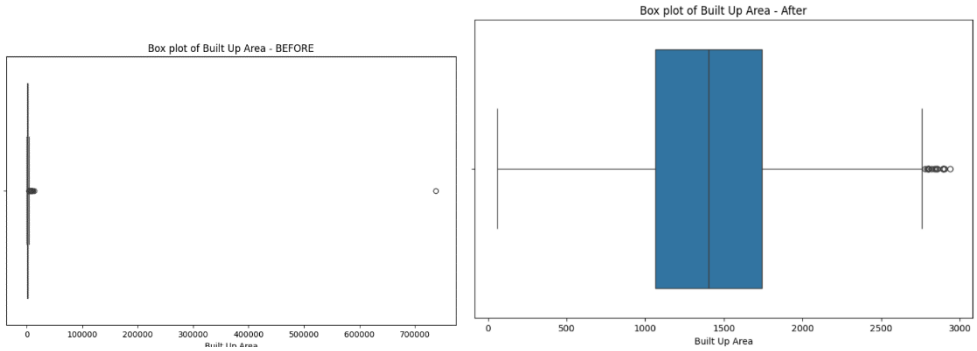
The box plot of built-up area tells that most of the built-up areas are clustered near the lower end of the scale, suggesting that many properties have smaller built-up areas, and the presence of outliers indicates that there are a few properties with significantly larger built-up spaces than the majority.



There are **109 OUTLIERS** in this column. We used **winsorization** to **HANDLE** these **OUTLIERS.**

The box plot before handling outliers had very unusual outliers which is a significantly large built-up area (7000000) compared to the rest of the data.

After handling the outliers, the box plot had an IQR of approximately 1000 to 1500 square units, represents the middle 50% of the built-up area values. The median built up area was around 1250 sq. units. The box plot suggests that most built-up areas fall between less than 500 and slightly above 2500 square units.



15. SUPER BUILT UP AREA COLUMN (NUMERICAL COLUMN)

Exploration

```
count     1861.000000
mean      1924.481725
std        764.629618
min         89.000000
25%       1480.000000
```

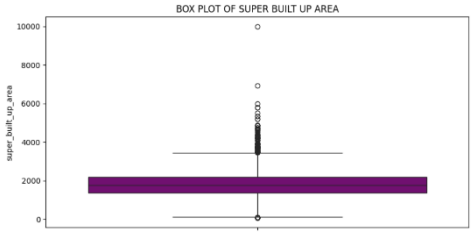
```
50%       1828.000000
75%       2215.000000
max      10000.000000
```

The describe function on the column shows that the maximum super built up area is 10,000 sq. Ft

There are **932 MISSING VALUES** in this column

The code we used to **HANDLE THE MISSING VALUES** in the 'super_built_up_area' column identifies rows where 'super_built_up_area' is missing and fills in those missing values by multiplying the 'built_up_area' by a predefined ratio from a matrix. This ratio is based on the median relationship between 'built_up_area' and 'super_built_up_area' across the dataset.

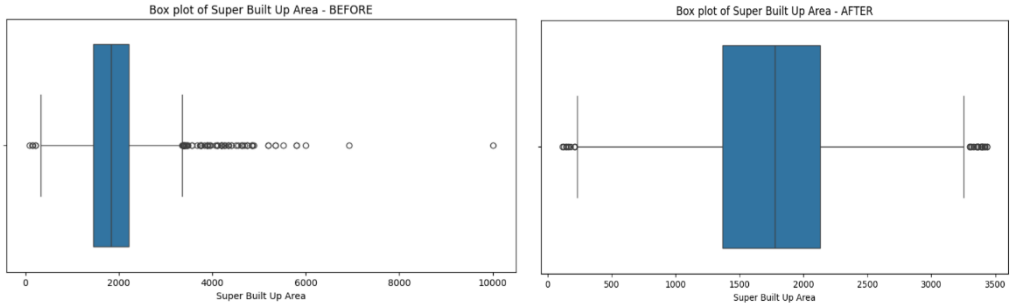
The box plot of the super built area indicates that the median value of the super built-up areas is very low. It also indicates the outliers above 4000 sq. ft which are super built-up areas that are significantly larger than many of the data points. Also indicates that while most properties have a smaller super built-up area, there are some with much larger areas.



THERE ARE **89 OUTLIERS** IN THIS COLUMN. We used **winsorisation** to **HANDLE OUTLIERS.**

The box plot before handling outliers stated that In Super Built Up Area units, the bulk of the data points are clustered between 2000 and around 4500.There are a few outliers, mostly on the higher end, which suggests that certain properties have substantially bigger Super Built Up Areas than average.

The box plot after handling outliers fixes the outlier of having very high super built-up area from **10,000 to 3000.** It also fixed the outlier having super low super built-up area.



16. CARPET AREA COLUMN (NUMERICAL COLUMN)

Exploration

```
count      1695.000000
mean       2631.402979
std       23950.488305
min          66.000000
25%         900.000000
50%        1305.000000
75%        1760.000000
max      607936.000000
```

The initial data exploration seems to have outliers. We can say that by looking at the max carpet area value.

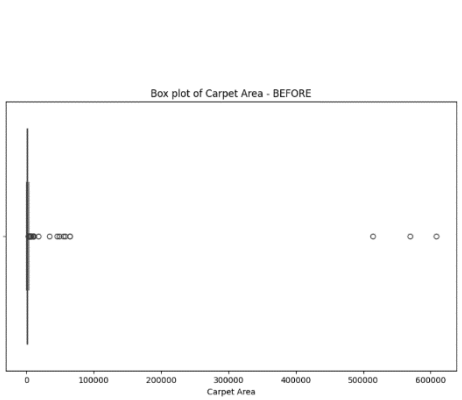
There are **1098 MISSING VALUES** in this column.

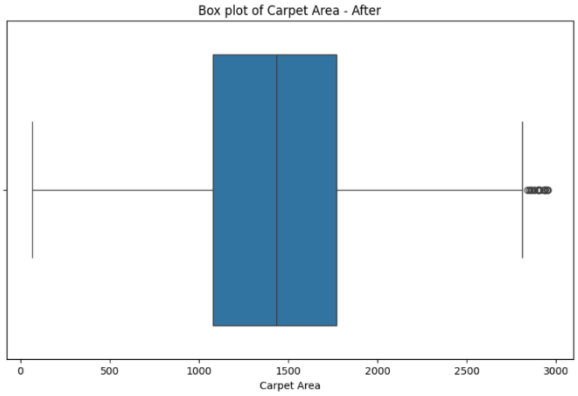
The code we used to **HANDLE THE MISSING VALUES** Identifies rows with null 'carpet_area' and multiplies it by 'built_up_area' by a conversion factor from a matrix and assigns the result back to 'carpet_area' where it was missing. It essentially, it uses a related area measurement and a predefined ratio to estimate the missing 'carpet_area' values.

There are **84 OUTLIERS** in this column. We used **winsorization** to **HANDLE THE OUTLIERS.**

The box plot before handling the outliers shows the IQR, indicating that the middle 50% of carpet areas fall between approximately 50,000 and 150,000 square units. This range is where most of the data is concentrated. The outliers are instances the IQR, indicating that the middle 50% of carpet areas fall between approximately 50,000 and 150,000 square units. This range is where most of the data is concentrated. The data distribution suggests that the carpet area data is skewed to the right, with many of the data points lying in the lower range of carpet area values.

After Handling the outliers, the IQR fell to 1000-1500 sq. Units and the max carpet area being reduced from 607936(outlier) to 2500 sq. Units.





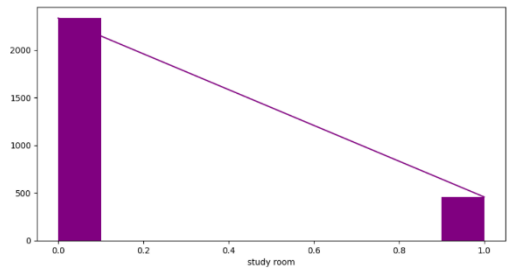
17. STUDY ROOM (NUMERICAL FEATURE)

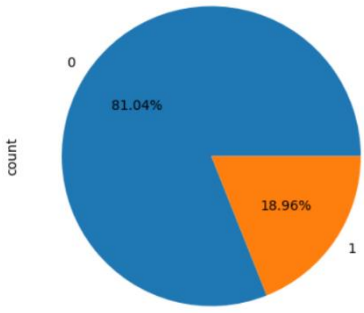
Exploration

```
study room
0     2335
1      458
```

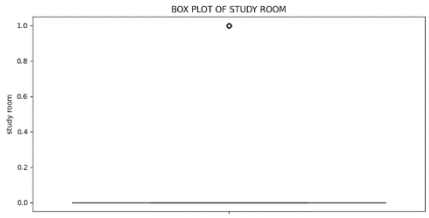
We can observe that most of the properties have no study rooms. (using describe function)

A bar plot and a pie chart have been plotted to the visualise the same.





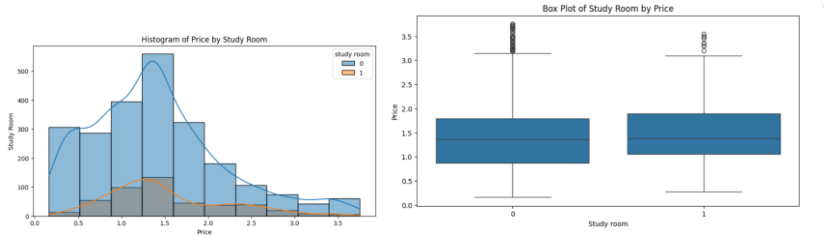
The box plot of study room has a very less variability. The absence of a visible box or whiskers in the box plot indicate that there is a little to no variability which could mean that the study of study room feature is identical across all observations. Most of the points are clustered at zero indicating that the data is highly skewed.



The histogram plot suggests that there is a noticeable peak around the 1.0 price point, suggesting that rooms with a study room are more commonly available at lower prices. It also tells that there are fewer options available without a study room, regardless of the price.

Generally, rooms with a study room are more prevalent, especially at the lower price range, while rooms without a study room are less common across all prices.

The box plot reveals that rooms lacking a study area (labelled “0”) display a broad spectrum of prices and a higher degree of fluctuation, including some notably expensive exceptions. In contrast, rooms equipped with a study area (labelled “1”) exhibit a more uniform pricing pattern with fewer deviations. Although the median price for both types of hovers around 1.5, the general pattern indicates a more stable pricing for rooms with a study space, as opposed to the more erratic pricing of rooms without it.



There are **NO MISSING VALUES** in this column

A total 458 outliers have been detected for this column

The “Box plot of Study Room - BEFORE” **BEFORE HANDLING OUTLIERS** indicates a dataset with low variability, where most values are close to zero. The middle 50% of the data is between 0.0 and 0.2, showing little spread. There’s a notable absence of high values, as seen by the lack of an upper whisker or box. An outlier at the 1.0 mark suggests an unusual data point far from the rest. This plot hints at a very consistent, low-range dataset with one exceptional value.

There's ''no outlier'' at 1.0 **AFTER HANDLING THE OUTLIERS**

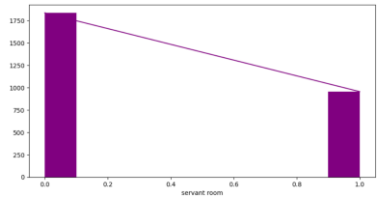
18. SERVANT ROOM (NUMERICAL FEATURE)

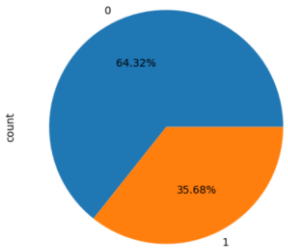
Exploration

```
servant room
0    1837
1     956
```

Using the describe function we can see that there is no servant room in majority of these properties.

A bar graph and a pie chart have been plotted to represent the same

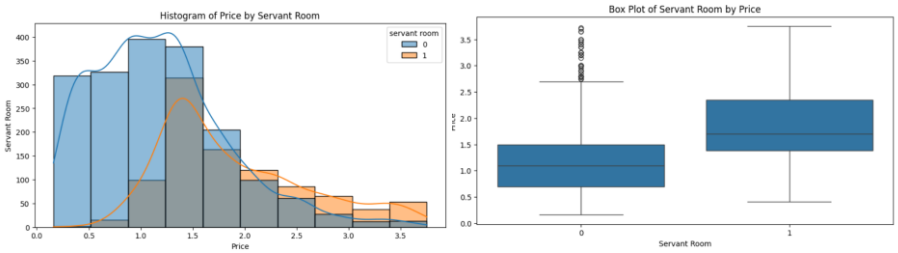




The histogram plot of price by servant room suggests that the properties without any servant rooms are generally cheaper than the ones which have a servant room

Trends: The superimposed line graphs for both categories demonstrate the distribution pattern, with rooms without a servant room peaking at lower costs and those with a servant room being more equally distributed but still skewed towards higher prices.

The box plot of price by servant room reveals that rooms without a servant room (category "0") have a broader price range and more outliers, indicating less frequently high-priced possibilities. Rooms with a servant room (category "1") have a higher median price and a narrower price distribution, indicating a premium for this amenity. Overall, the addition of a servant room tends to raise the room's price, with less variation in cost.



There are **NO MISSING VALUES** in this column.

There are **NO OUTLIERS** in this column.

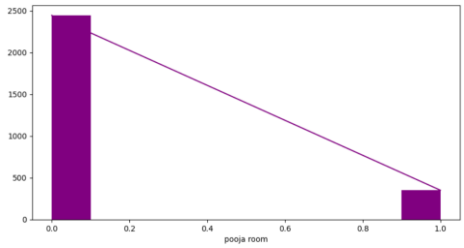
19. POOJA ROOM

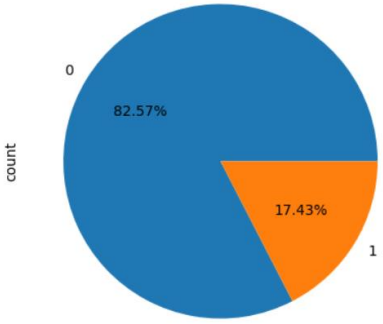
Exploration

```
pooja room
0    2443
1     350
```

We can observe that most of the properties are not offering a pooja room.

We can visualise the latter using a bar plot and a pie chart





The histogram "Histogram of Price by Pooja Room" illustrates that properties without a Pooja room are dominant in the lower price range, with a peak at the 1.0 price point. Properties with a Pooja room are less common and spread across various price points, slightly more concentrated between 1.0 and 1.5. This suggests that a Pooja room is a feature that generally increases the value of a property, making it less common in the lower price segment.

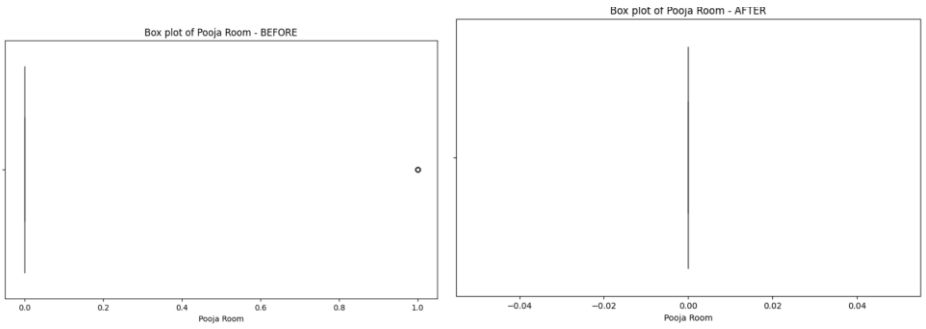
The box plot analysis for the same reveals that rooms without a Pooja room (category "0") have a higher median price and exhibit greater price variability, including some notably expensive outliers. Conversely, rooms with a Pooja room (category "1") with a lower median price and less variation in pricing, indicating a more uniform cost distribution for these rooms.

There are **NO MISSING VALUES** in this column.

There are **350 OUTLIERS** in this column.

The "Box plot of Pooja Room - BEFORE" **before handling outliers** shows a dataset with most values tightly clustered, indicating uniformity and little variation. An outlier at 0.9 stands out, suggesting a single unusual measurement or error.

The **box plot of pooja room** after handling the outliers has removed the existing outliers.



20. STORE ROOM:

```
dataset['store room'].describe()
dataset['store room'].value_counts()
```

Exploration:

From the above code, we generated descriptive statistics of the store room column.

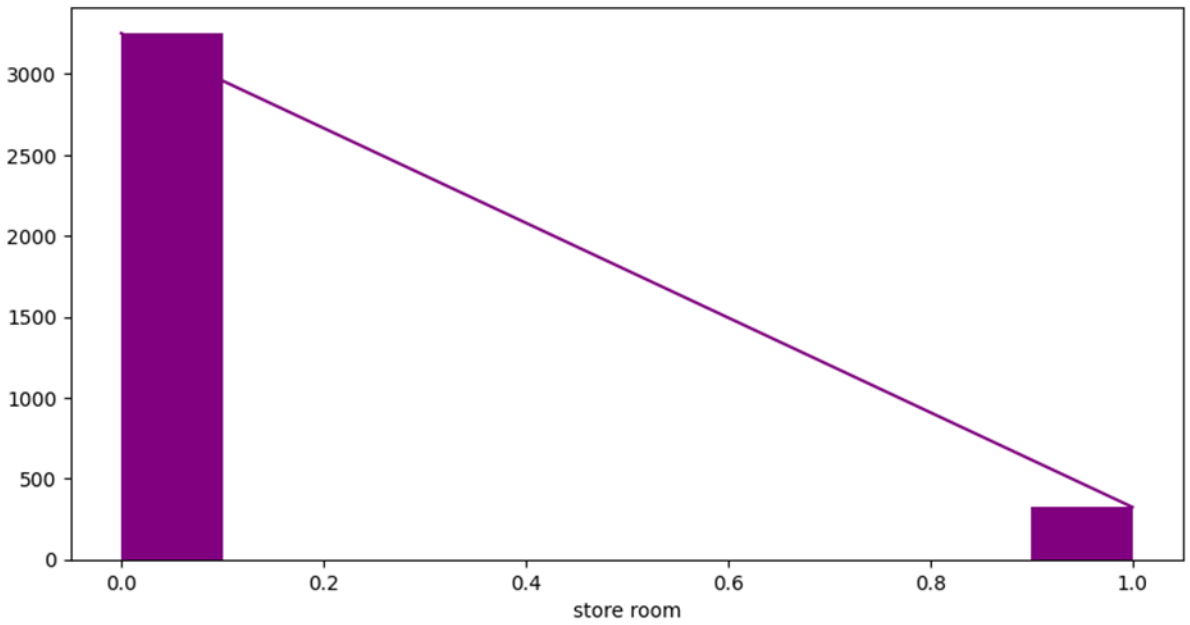
Output:

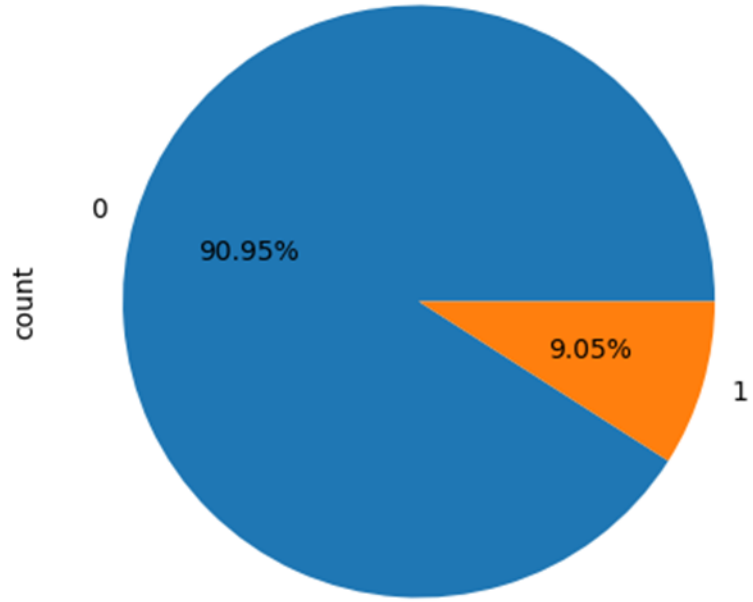
```
store room0    32511     324Name: count, dtype: int64
```

We can see that majority of the properties do not have a store room.

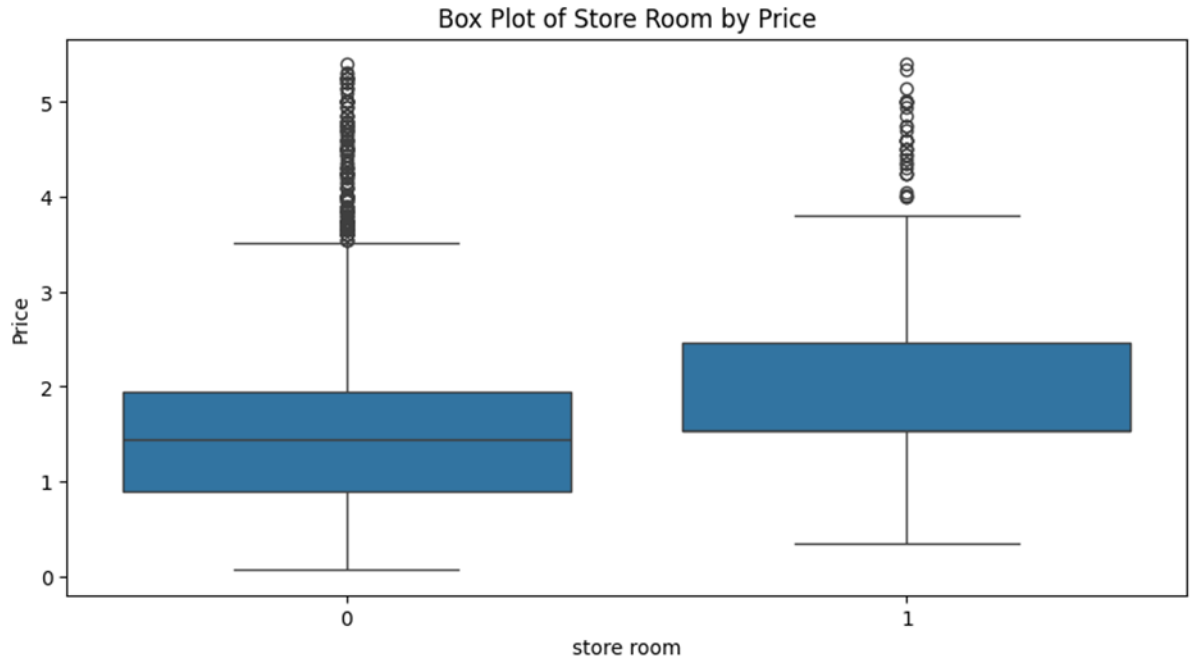
The store room column is binary, 1 indicates the presence and 0 indicates the absence of it.

The following bar graph and pie chart also show that around 90.95 % of properties don’t have a store room, and about 9.05% have them.





Plotting the box plot:



The box plot shows there are no outliers in the properties with store rooms.

Handling missing values:

```
missing_store_room= dataset["store room"].isnull().sum()
missing_store_room
```

The column has no missing values.

Outlier Detection & handling:

IQR:

It is the spread of data points, it is calculated as the difference between 75th percentile and 25th percentile (Q3-Q1)

We define the outlier boundaries as:

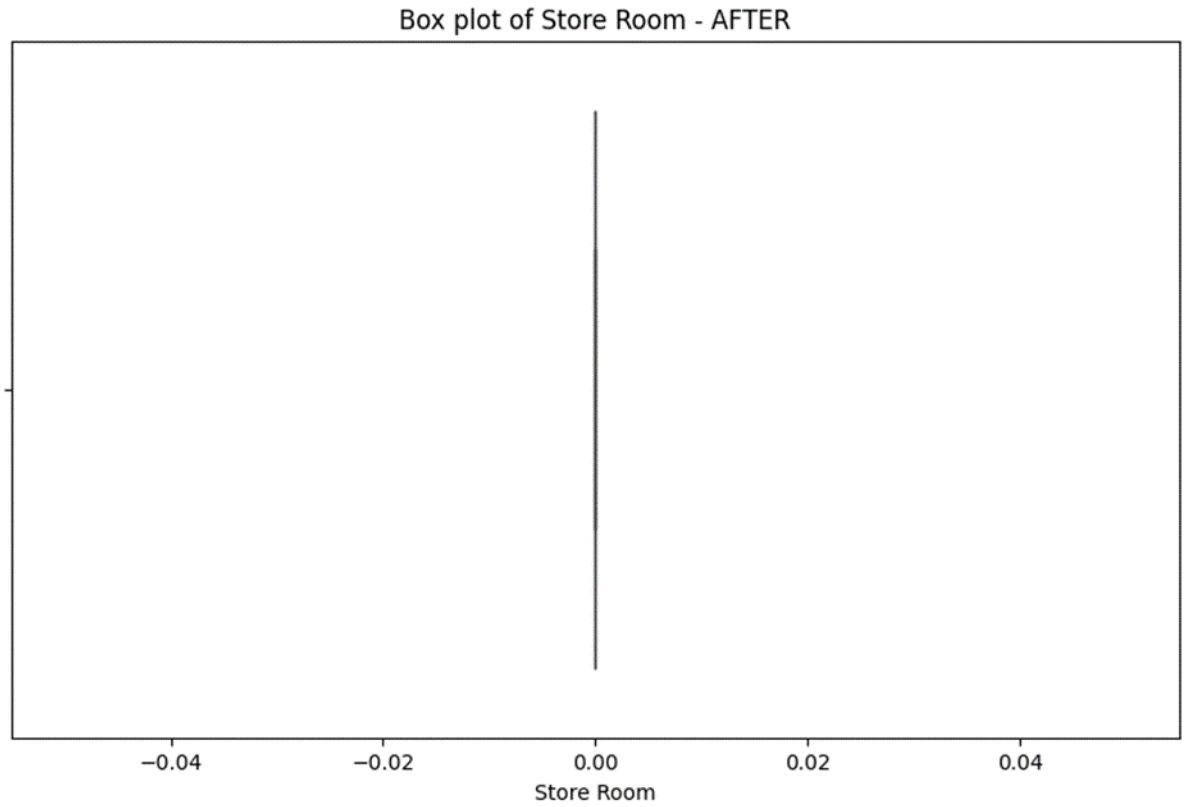
Lower Bound: Q1 – 1.5*IQR

Upper Bound: Q1 + 1.5*IQR

- ANY DATA POINTS BELOW THE LOWER BOUND OR ABOVE THE UPPER BOUND ARE CONSIDERED OUTLIERS.

WINSORIZATION:

Winsorization replaces the data points below the lower percentile with the value of the lower percentile and and data points above the upper percentile with the value of the upper percentile.

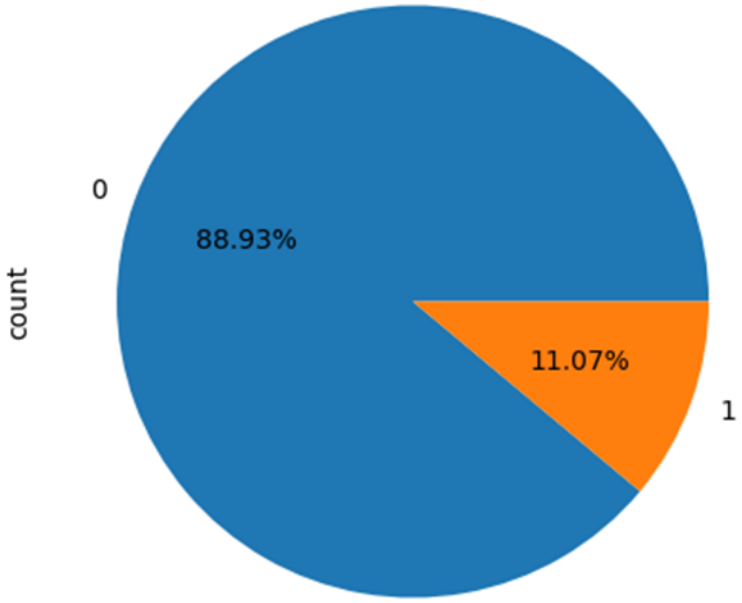


We can see from the AFTER box plot that there are no outliers after winsorization

21. Other Columns

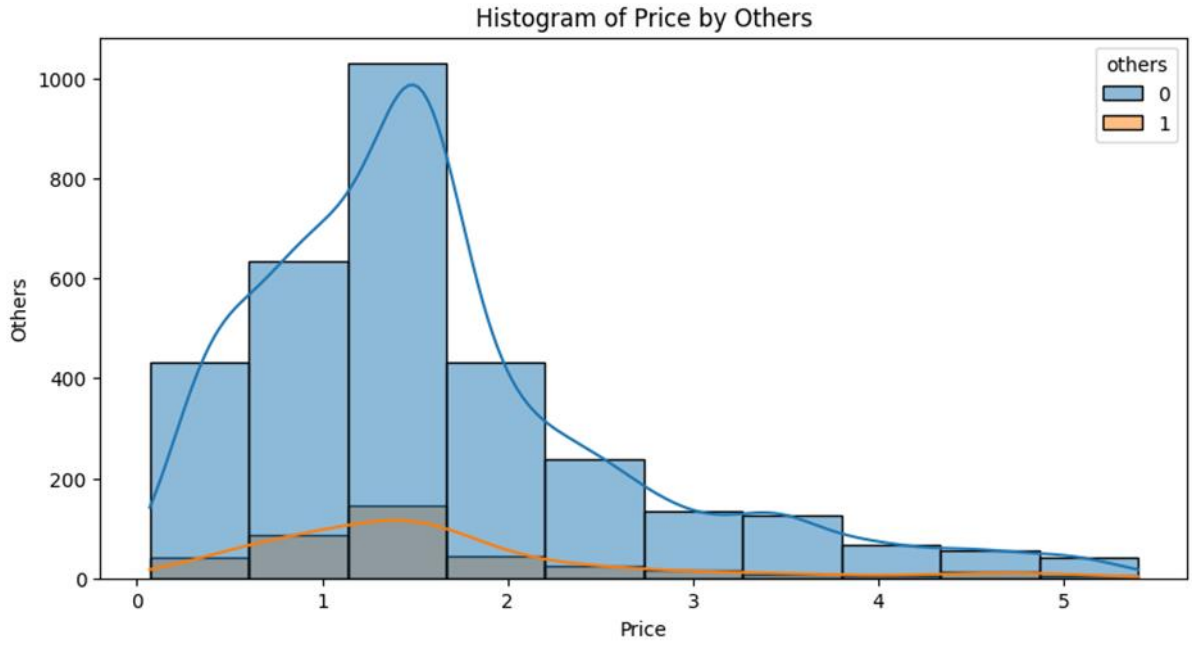
Through initial data exploration, we find that “others” column has binary values.

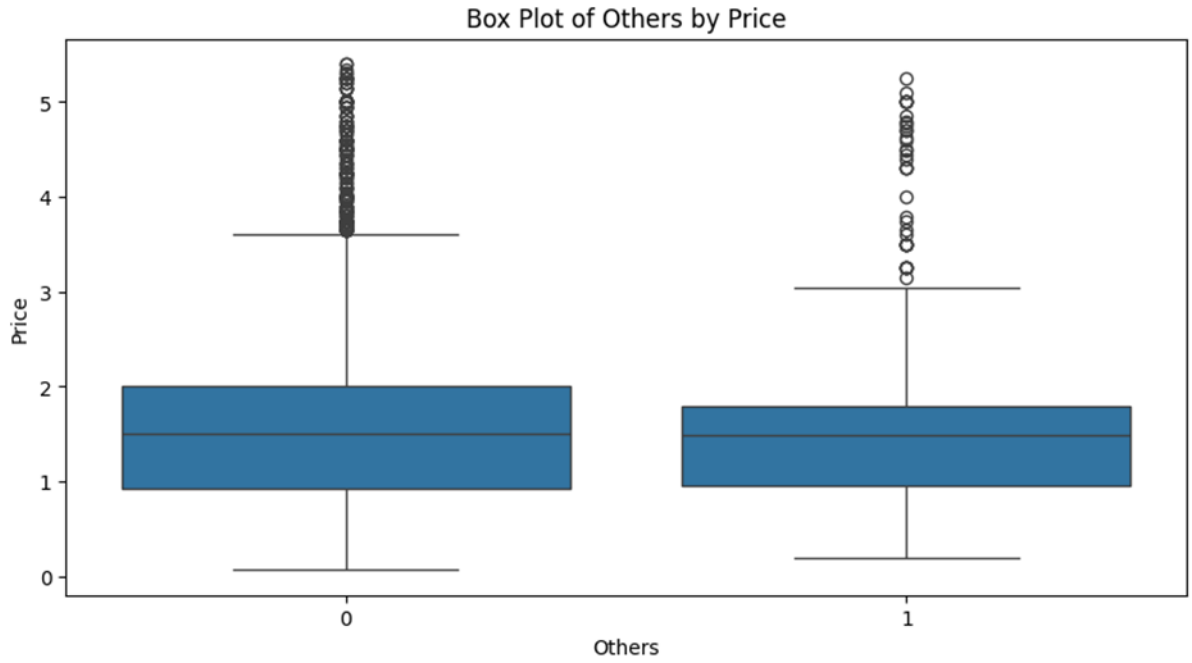
```
dataset['others'].value_counts()
others0    31901     385Name: count, dtype: int64
```



Most of the properties don’t have other rooms.

Others Vs Price graph





We can see that there are no outliers in properties where there are other rooms.

Missing Values:

We found there are no missing values in this column using the isnull().sum() function.

Outlier detection & handling:

Code for detection:

```
q1 = dataset['others'].quantile(0.25) # 25th percentile
q3 = dataset['others'].quantile(0.75) #75th percentile
IQR_others = q3-q1 #interquartile range
k = IQR_others*1.5
outliers_others= dataset[(dataset['others']< q1-k )|
(dataset['others']>q3+k)]
print("No. of outliers:",len(outliers_others))
print("IQR:",IQR_others)
```

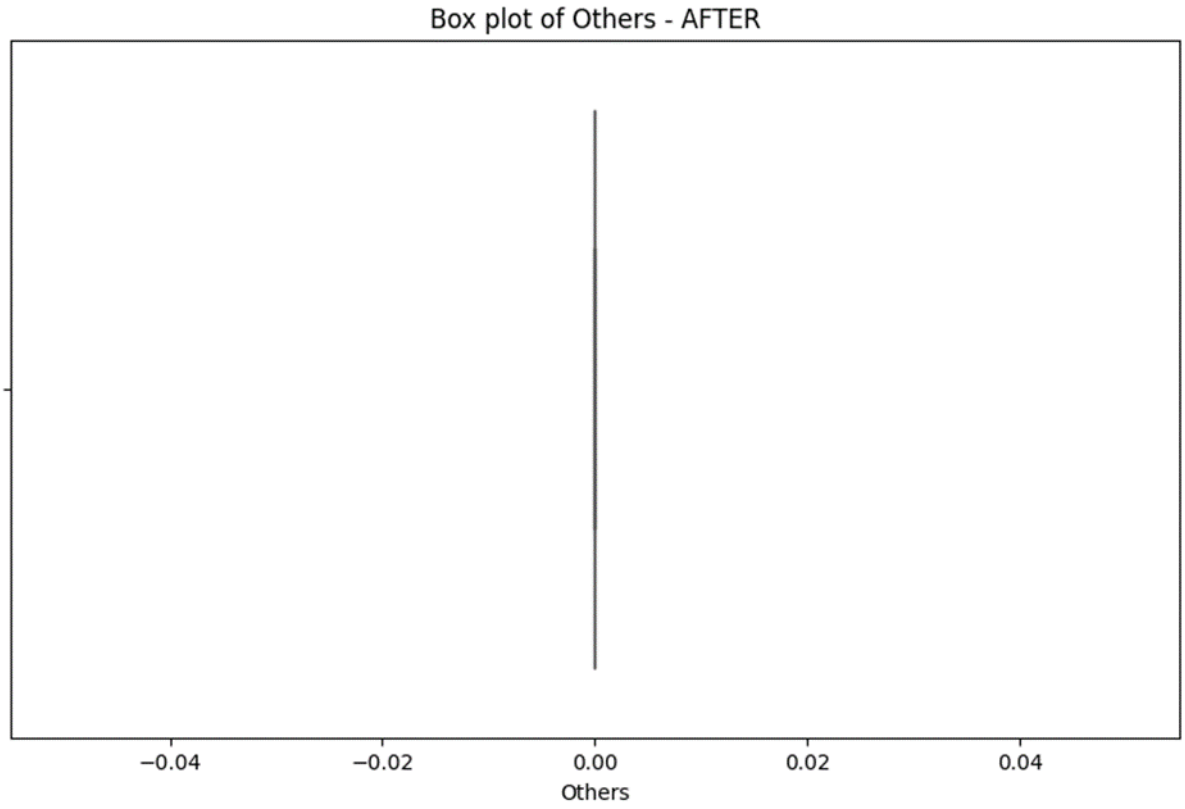
This code and the box plot shows that there are a few outliers in the data.

They have been handled using the following code:

```
Q1 = dataset['others'].quantile(0.25)
Q3 = dataset['others'].quantile(0.75)
IQR = Q3 - Q1
outlier_indices = (dataset['others'] < (Q1 - 1.5 * IQR)) |
(dataset['others'] > (Q3 + 1.5 * IQR))

IQR
dataset['others'] = np.where(outlier_indices,
dataset['others'].median(), dataset['others'])
```

This code uses the median value of the column to reduce impact of extreme values without removing any data points.



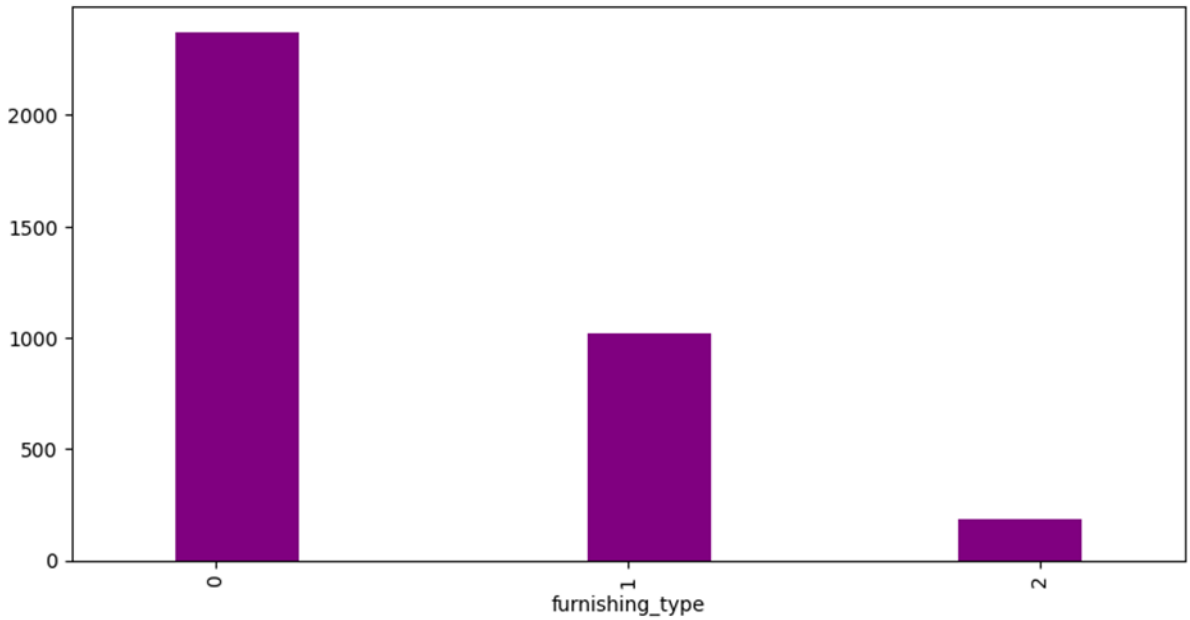
The AFTER box plot shows that the outliers have been handled.

22. FURNISHING TYPE COLUMN:

Initial exploration:

```
plt.figure(figsize=(10, 5))
dataset['furnishing_type'].hist(color='purple')
dataset['furnishing_type'].value_counts().plot(kind='bar',color='purple
',width=0.2)
```

From this code, we plot the bar graph of the column.

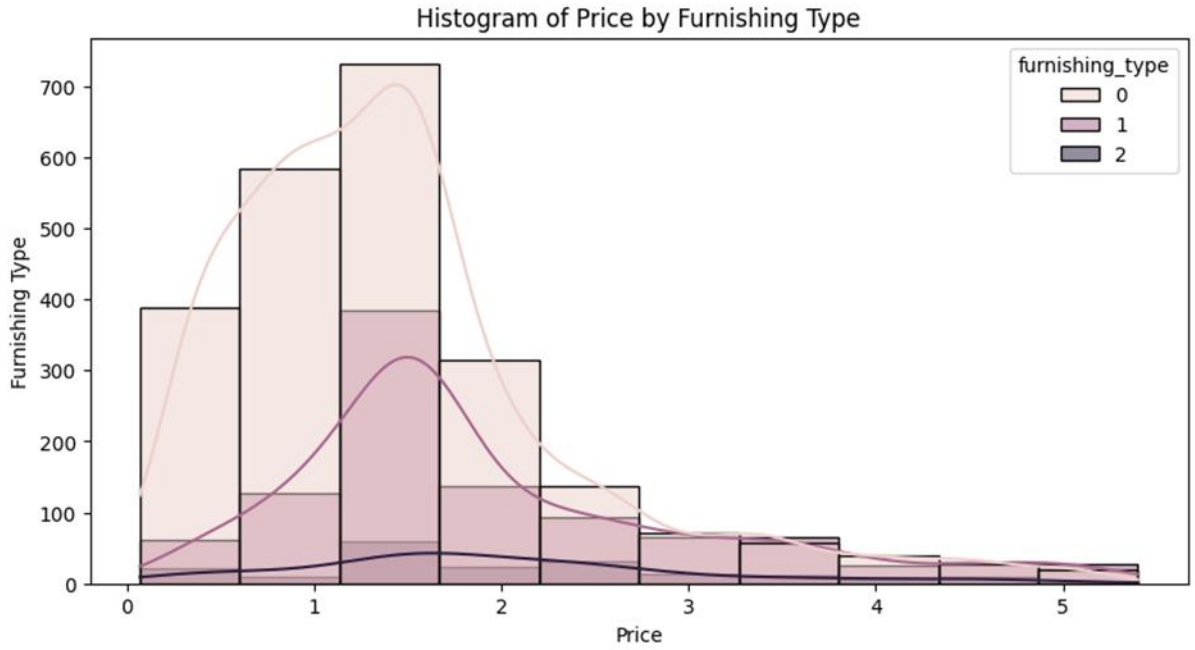


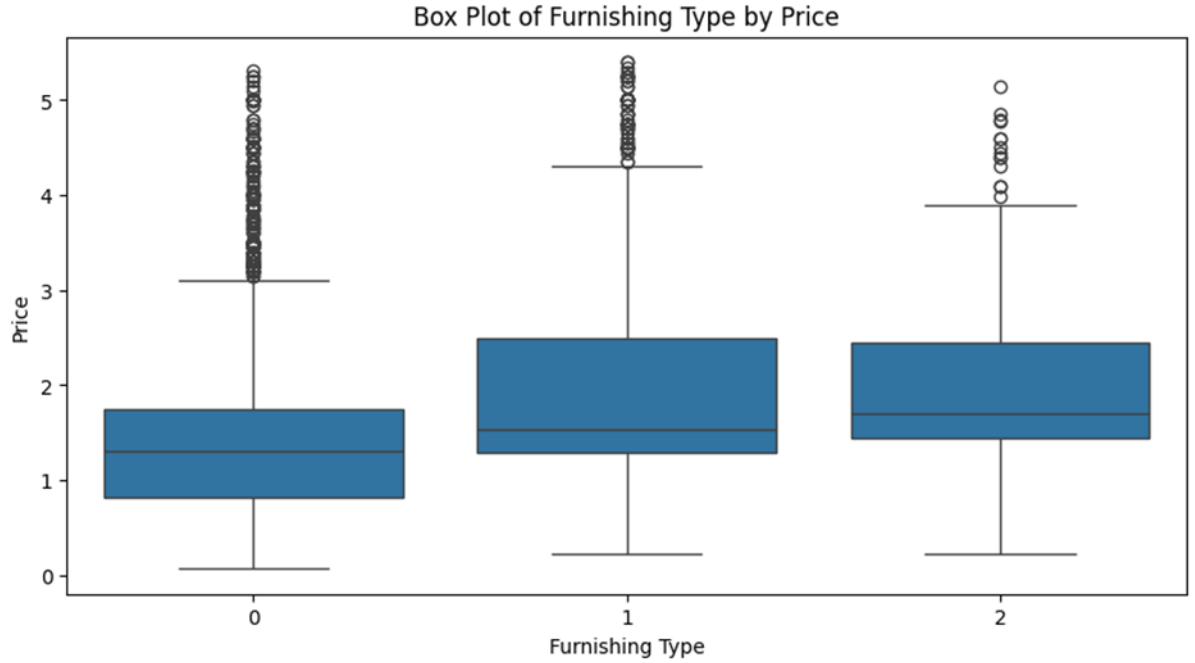
The bar graph shows that the number of properties that are unfurnished are the largest, semi-furnished are less and furnished are the least.

The outputs from the following code show how price varies depending on the furnishing type

```
plt.figure(figsize=(10, 5))
sns.histplot(data=dataset, x='price', hue='furnishing_type', kde=True,
bins=10)
plt.title('Histogram of Price by Furnishing Type')
plt.xlabel('Price')
plt.ylabel('Furnishing Type')
plt.show()

plt.figure(figsize=(10, 5))
sns.boxplot(data=dataset, x='furnishing_type', y='price')
plt.title('Box Plot of Furnishing Type by Price')
plt.xlabel('Furnishing Type')
plt.ylabel('Price')
plt.show()
```





Missing Values:

There are no missing values in the column

```
missing_furnishingtype= dataset["furnishing_type"].isnull().sum()
missing_furnishingtype
```

Outlier detection & Handling:

```
q1 = dataset['furnishing_type'].quantile(0.25) # 25th percentile
q3 = dataset['furnishing_type'].quantile(0.75) #75th percentile
IQR_furnishing_type = q3-q1 #interquartile range
```

```
k = IQR_furnishing_type*1.5
outliers_furnishing_type= dataset[(dataset['furnishing_type']< q1-k )|
(dataset['furnishing_type']>q3+k)]
print("No. of outliers:",len(outliers_furnishing_type))
print("IQR:",IQR_furnishing_type)
```

From this code, we also see that there are no outliers in the column.

23. LUXURY SCORE:

Exploration:

```
dataset['luxury_score'].value_counts()
dataset['luxury_score'].describe()
```

Through these functions we found the descriptive statistics of the column to be:

```
count    3575.000000mean       71.883636std        53.240355min
0.00000025%        31.00000050%        60.00000075%       110.000000max
174.000000Name: luxury_score, dtype: float64
```

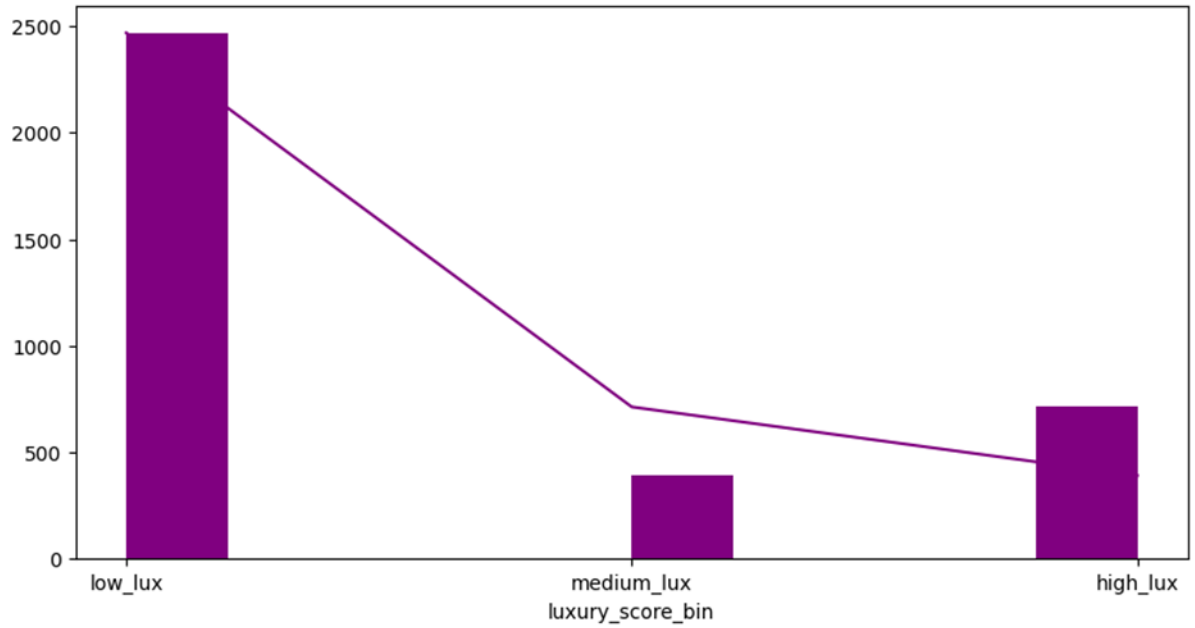
Bining:

We use binning to categorize the luxury column into different levels of luxury, such as 'low_lux', 'medium_lux', and 'high_lux'.

Bining simplies complex data, reduces noise, helps reduce effect of outliers.

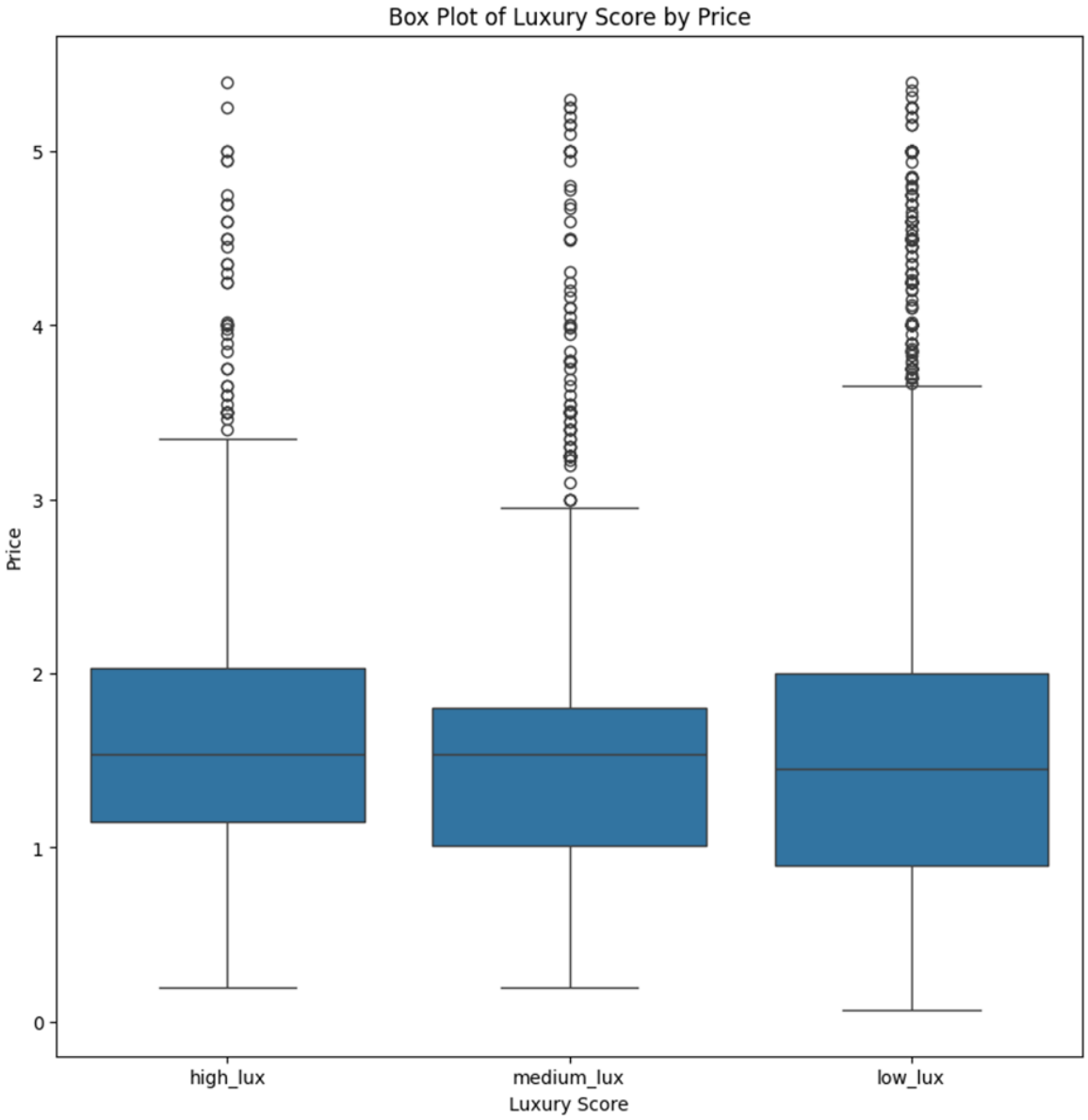
Bar plot:

```
plt.figure(figsize=(10, 5))
dataset['luxury_score_bin'].hist(color='purple')
dataset['luxury_score_bin'].value_counts().plot(color='purple')
<Axes: xlabel='luxury_score_bin'>
```



The bar graph shows that properties with low luxury score are highest.


Histogram of Price by Luxury Score
luxury_score_bin
high_lux
medium_lux
low_lux
Luxury Score
Price
0
100
200
300
400
500
600
700
0
1
2
3
4
5



These graphs show how price varies with respect to the luxury score.

Missing values:

```
_missing_luxury_score= dataset["luxury_score"].isnull().sum()

missing_luxury_score
```

There are no missing values in this column.

Outlier Detection & Handling:

```
q1 = dataset['luxury_score'].quantile(0.25) # 25th percentile
q3 = dataset['luxury_score'].quantile(0.75) #75th percentile
```

```
IQR_luxury_score = q3-q1 #interquartile range
k = IQR_luxury_score*1.5
outliers_luxury_score= dataset[(dataset['luxury_score']< q1-k )|
(dataset['luxury_score']>q3+k)]
print("No. of outliers:",len(outliers_luxury_score))
print("IQR:",IQR_luxury_score)
```

From the output of this code, we see that there are no outliers in the luxury score column.

FEATURE SELECTION:

We created a new dataset as the copy of the original dataset and dropped irrelevant columns.

```
from sklearn.preprocessing import OrdinalEncoder
columns = new_dataset.select_dtypes(exclude=['number'])
label_encoder = LabelEncoder()
for column in columns:
  label_encoder.fit(new_dataset[column])
  new_dataset[column] = label_encoder.transform(new_dataset[column])

label_encoder.fit(new_dataset['property_type'])
new_dataset['property_type'] =
label_encoder.transform(new_dataset['property_type'])
```

We used label encoding to convert categorical columns to numerical columns

We computed correlation coefficient between ‘price’ and ‘price per sqft’ to determine degree of the linear relationship between them.

We chose random forest regressor model as it handles both categorical and numerical features effectively,

Computing feature importance scores:

```
from sklearn.ensemble import RandomForestRegressor

features = new_dataset.columns.difference(['price'])

model = RandomForestRegressor()
```

```
model.fit(new_dataset[features], new_dataset['price'])

important_features = model.feature_importances_
importance = pd.DataFrame({'Feature': features, 'Importance':
important_features})
importance = importance.sort_values(by='Importance', ascending=False)
importance
```

Feature importance scores indicate the relative importance of each feature in predicting price.

Features other than price were selected as input features.

LINEAR REGRESSION MODEL:

We split dataset and price into training and testing sets.

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=42)

model = LinearRegression()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
mse = mean_squared_error(y_test, y_pred)
mse
```

The LinearRegression class from sklearn.linear_model is used to create a linear regression model.
The model is then trained on the training data (x_train, y_train) using the fit method.
Using the predict method, the trained model is used to make predictions on test data.
We calculate MEAN SQUARE ERROR (MSE) between actual and predicted values to evaluate performance.

‘0.4233426916227396’ - This value of MSE indicates the regression model is making accurate predictions.

Scatter plot to visualize relationship between Actual and predicted values.