DSBDAL 03

Roll No : 13320

In [43]:
```
1 import pandas as pd
2
3 dt = pd.read_csv("C:/Users/Welcome/Downloads/housing_price_dataset.csv/
4 print(dt)
```

```
       SquareFeet  Bedrooms  Bathrooms Neighborhood  YearBuilt          Pr
ice
0            2126         4          1        Rural       1969  215355.283
618
1            2459         3          2        Rural       1980  195014.221
626
2            1860         2          1       Suburb       1970  306891.012
076
3            2294         2          1        Urban       1996  206786.787
153
4            2130         5          2       Suburb       2001  272436.239
065
...           ...       ...        ...          ...        ...
...
49995        1282         5          3        Rural       1975  100080.865
895
49996        2854         2          2       Suburb       1988  374507.656
727
49997        2979         5          3       Suburb       1962  384110.555
590
49998        2596         5          2        Rural       1984  380512.685
957
49999        1572         5          3        Rural       2011  221618.583
218

[50000 rows x 6 columns]
```

In [44]:
```
1 dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   SquareFeet    50000 non-null  int64
 1   Bedrooms      50000 non-null  int64
 2   Bathrooms     50000 non-null  int64
 3   Neighborhood  50000 non-null  object
 4   YearBuilt     50000 non-null  int64
 5   Price         50000 non-null  float64
dtypes: float64(1), int64(4), object(1)
memory usage: 2.3+ MB
```

In [45]:
```
1 dt.describe()
```

Out[45]:

	SquareFeet	Bedrooms	Bathrooms	YearBuilt	Price
count	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000
mean	2006.374680	3.498700	1.995420	1985.404420	224827.325151
std	575.513241	1.116326	0.815851	20.719377	76141.842966
min	1000.000000	2.000000	1.000000	1950.000000	-36588.165397
25%	1513.000000	3.000000	1.000000	1967.000000	169955.860225
50%	2007.000000	3.000000	2.000000	1985.000000	225052.141166
75%	2506.000000	4.000000	3.000000	2003.000000	279373.630052
max	2999.000000	5.000000	3.000000	2021.000000	492195.259972

In [46]:
```
1 dt.mean()
```

```
C:\Users\Welcome\AppData\Local\Temp\ipykernel_5652\2162581429.py:1: Future
Warning: The default value of numeric_only in DataFrame.mean is deprecate
d. In a future version, it will default to False. In addition, specifying
'numeric_only=None' is deprecated. Select only valid columns or specify th
e value of numeric_only to silence this warning.
  dt.mean()
```

Out[46]:
```
SquareFeet      2006.374680
Bedrooms           3.498700
Bathrooms          1.995420
YearBuilt       1985.404420
Price         224827.325151
dtype: float64
```

In [47]:
```
1 mean=dt.loc[:,'Bathrooms'].mean()
2 mean
```

Out[47]:
```
1.99542
```

In [48]:
```
1 dt.mean(axis=1)[0:4]
```

```
C:\Users\Welcome\AppData\Local\Temp\ipykernel_5652\2061211884.py:1: Future
Warning: Dropping of nuisance columns in DataFrame reductions (with 'numer
ic_only=None') is deprecated; in a future version this will raise TypeErro
r.  Select only valid columns before calling the reduction.
  dt.mean(axis=1)[0:4]
```

Out[48]:
```
0    43891.056724
1    39891.644325
2    62144.802415
3    42215.957431
dtype: float64
```

In [49]:
```
1 dt.mode()
```

Out[49]:

	SquareFeet	Bedrooms	Bathrooms	Neighborhood	YearBuilt	Price
0	2486.0	3.0	1.0	Suburb	1968.0	-36588.165397
1	NaN	NaN	NaN	NaN	NaN	-28774.998022
2	NaN	NaN	NaN	NaN	NaN	-24715.242482
3	NaN	NaN	NaN	NaN	NaN	-24183.000515
4	NaN	NaN	NaN	NaN	NaN	-23911.003119
...	...	...	...	...	...	...
49995	NaN	NaN	NaN	NaN	NaN	468493.877841
49996	NaN	NaN	NaN	NaN	NaN	470989.679074
49997	NaN	NaN	NaN	NaN	NaN	476671.733263
49998	NaN	NaN	NaN	NaN	NaN	482577.163405
49999	NaN	NaN	NaN	NaN	NaN	492195.259972

50000 rows × 6 columns

In [50]:
```
1 mode=dt.loc[:,'Bathrooms'].mode()
2 mode
```

Out[50]:
```
0    1
Name: Bathrooms, dtype: int64
```

In [51]:
```
1 dt.median()
```

```
C:\Users\Welcome\AppData\Local\Temp\ipykernel_5652\3024937849.py:1: Future
Warning: The default value of numeric_only in DataFrame.median is deprecat
ed. In a future version, it will default to False. In addition, specifying
'numeric_only=None' is deprecated. Select only valid columns or specify th
e value of numeric_only to silence this warning.
  dt.median()
```

Out[51]:
```
SquareFeet      2007.000000
Bedrooms           3.000000
Bathrooms          2.000000
YearBuilt       1985.000000
Price         225052.141166
dtype: float64
```

In [52]:
```
1 median = dt.loc[:,'Bathrooms'].median()
2 median
```

Out[52]:
```
2.0
```

In [53]:
```
1 dt.min()
```

Out[53]:
```
SquareFeet              1000
Bedrooms                   2
Bathrooms                  1
Neighborhood           Rural
YearBuilt               1950
Price          -36588.165397
dtype: object
```

In [54]:
```
1 dt.loc[:,'Bathrooms'].min()
```

Out[54]:
```
1
```

In [55]:
```
1 dt.max()
```

Out[55]:
```
SquareFeet               2999
Bedrooms                    5
Bathrooms                   3
Neighborhood            Urban
YearBuilt                2021
Price          492195.259972
dtype: object
```

In [56]:
```
1 dt.loc[:,'Bathrooms'].max()
```

Out[56]:
```
3
```

In [57]:
```
1 std=dt.loc[:,'Bathrooms'].std()
2 std
```

Out[57]:
```
0.8158506823229902
```

In [58]:
```
1 skewness = 3 * (mean - median)/std
2 print(skewness)
3 if (skewness<0):
4     print("Right skewed")
5 else:
6     print("Left skewed")
```

```
-0.01684131704208161
Right skewed
```

In [59]:
```
1 from sklearn import preprocessing
2 enc = preprocessing.OneHotEncoder()
3 enc_df=pd.DataFrame(enc.fit_transform(dt[['Neighborhood']]).toarray())
4 enc_df
```

Out[59]:

	0	1	2
0	1.0	0.0	0.0
1	1.0	0.0	0.0
2	0.0	1.0	0.0
3	0.0	0.0	1.0
4	0.0	1.0	0.0
...	...	...	...
49995	1.0	0.0	0.0
49996	0.0	1.0	0.0
49997	0.0	1.0	0.0
49998	1.0	0.0	0.0
49999	1.0	0.0	0.0

50000 rows × 3 columns

In [60]:
```
df_encode =dt.join(enc_df)
df_encode
```

Out[60]:

	SquareFeet	Bedrooms	Bathrooms	Neighborhood	YearBuilt	Price	0	1
0	2126	4	1	Rural	1969	215355.283618	1.0	0.0
1	2459	3	2	Rural	1980	195014.221626	1.0	0.0
2	1860	2	1	Suburb	1970	306891.012076	0.0	1.0
3	2294	2	1	Urban	1996	206786.787153	0.0	0.0
4	2130	5	2	Suburb	2001	272436.239065	0.0	1.0
...	...	...	...	...	...	...	...	...
49995	1282	5	3	Rural	1975	100080.865895	1.0	0.0
49996	2854	2	2	Suburb	1988	374507.656727	0.0	1.0
49997	2979	5	3	Suburb	1962	384110.555590	0.0	1.0
49998	2596	5	2	Rural	1984	380512.685957	1.0	0.0
49999	1572	5	3	Rural	2011	221618.583218	1.0	0.0

50000 rows × 9 columns

In [61]:
```
dt.groupby(['Neighborhood'])['Bedrooms'].mean()

```

Out[61]:
```
Neighborhood
Rural     3.506836
Suburb    3.493930
Urban     3.495332
Name: Bedrooms, dtype: float64
```

In [62]:
```
dt.groupby(['Neighborhood'])['Bedrooms'].median()
```

Out[62]:
```
Neighborhood
Rural     4.0
Suburb    3.0
Urban     3.0
Name: Bedrooms, dtype: float64
```

In [63]:
```
dt.groupby(['Neighborhood'])['Bedrooms'].std()
```

Out[63]:
```
Neighborhood
Rural     1.116168
Suburb    1.112393
Urban     1.120453
Name: Bedrooms, dtype: float64
```

In [64]:
```
dt['Neighborhood'].replace(['Rural', 'Suburb','Urban'],[1, 2,3], inplac
dt
```

Out[64]:

	SquareFeet	Bedrooms	Bathrooms	Neighborhood	YearBuilt	Price
0	2126	4	1	1	1969	215355.283618
1	2459	3	2	1	1980	195014.221626
2	1860	2	1	2	1970	306891.012076
3	2294	2	1	3	1996	206786.787153
4	2130	5	2	2	2001	272436.239065
...	...	...	...	...	...	...
49995	1282	5	3	1	1975	100080.865895
49996	2854	2	2	2	1988	374507.656727
49997	2979	5	3	2	1962	384110.555590
49998	2596	5	2	1	1984	380512.685957
49999	1572	5	3	1	2011	221618.583218

50000 rows × 6 columns

In [65]:
```
dt.groupby(['Neighborhood'])['Bedrooms'].std()
```

Out[65]:
```
Neighborhood
1    1.116168
2    1.112393
3    1.120453
Name: Bedrooms, dtype: float64
```

In [ ]:
```

```

In [ ]:
```

```