```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data = pd.read_csv('/content/Real estate.csv')
```

```
data.head()
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data.columns
```

```
Index(['No', 'X1 transaction date', 'X2 house age',
       'X3 distance to the nearest MRT station',
       'X4 number of convenience stores', 'X5 latitude', 'X6 longitude',
       'Y house price of unit area'],
      dtype='object')
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 414 entries, 0 to 413
Data columns (total 8 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   No                                      414 non-null    int64
 1   X1 transaction date                     414 non-null    float64
 2   X2 house age                            414 non-null    float64
 3   X3 distance to the nearest MRT station  414 non-null    float64
 4   X4 number of convenience stores         414 non-null    int64
 5   X5 latitude                             414 non-null    float64
 6   X6 longitude                            414 non-null    float64
 7   Y house price of unit area              414 non-null    float64
dtypes: float64(6), int64(2)
memory usage: 26.0 KB
```

```
data.describe()
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
count	414.000000	414.000000	414.000000	414.000000	414.000000	414.000000	414.000000	414.000000
mean	207.500000	2013.148971	17.712560	1083.885689	4.094203	24.969030	121.533361	37.980193
std	119.655756	0.281967	11.392485	1262.109595	2.945562	0.012410	0.015347	13.606488
min	1.000000	2012.667000	0.000000	23.382840	0.000000	24.932070	121.473530	7.600000
25%	104.250000	2012.917000	9.025000	289.324800	1.000000	24.963000	121.528085	27.700000
50%	207.500000	2013.167000	16.100000	492.231300	4.000000	24.971100	121.538630	38.450000

```
data.isnull().sum()
```

	0
No	0
X1 transaction date	0
X2 house age	0
X3 distance to the nearest MRT station	0
X4 number of convenience stores	0
X5 latitude	0
X6 longitude	0
Y house price of unit area	0

dtype: int64

```
data.columns
```

```
Index(['No', 'X1 transaction date', 'X2 house age',
       'X3 distance to the nearest MRT station',
       'X4 number of convenience stores', 'X5 latitude', 'X6 longitude',
       'Y house price of unit area'],
      dtype='object')
```

```
columns = list(data.columns)
columns
```

```
['No',
 'X1 transaction date',
 'X2 house age',
 'X3 distance to the nearest MRT station',
 'X4 number of convenience stores',
 'X5 latitude',
 'X6 longitude',
 'Y house price of unit area']
```

```
columns = [col.replace(' ','_') for col in columns]
```

```
columns
```

```
['No',
 'X1_transaction_date',
 'X2_house_age',
 'X3_distance_to_the_nearest_MRT_station',
 'X4_number_of_convenience_stores',
 'X5_latitude',
 'X6_longitude',
 'Y_house_price_of_unit_area']
```

```
data.head()
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data.columns = columns
data.head()
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_convenie
0	1	2012.917	32.0	84.87882	
1	2	2012.917	19.5	306.59470	
2	3	2013.583	13.3	561.98450	
3	4	2013.500	13.3	561.98450	
4	5	2012.833	5.0	390.56840	

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 414 entries, 0 to 413
Data columns (total 8 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   No                                      414 non-null    int64
 1   X1_transaction_date                     414 non-null    float64
 2   X2_house_age                            414 non-null    float64
 3   X3_distance_to_the_nearest_MRT_station  414 non-null    float64
 4   X4_number_of_convenience_stores         414 non-null    int64
 5   X5_latitude                             414 non-null    float64
 6   X6_longitude                            414 non-null    float64
 7   Y_house_price_of_unit_area              414 non-null    float64
dtypes: float64(6), int64(2)
memory usage: 26.0 KB
```

```
start_date = 2012.917
end_date =  2015.813
```

```
filtered_data = data[(data['X1_transaction_date'] >= start_date) & (data['X1_transaction_date'] <= end_date
filtered_data
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_conve
1	2	2012.917	19.5	306.59470	
2	3	2013.583	13.3	561.98450	
3	4	2013.500	13.3	561.98450	
7	8	2013.417	20.3	287.60250	
...	...	...	...	...	
408	409	2013.417	18.5	2175.74400	
409	410	2013.000	13.7	4082.01500	
411	412	2013.250	18.8	390.96960	
412	413	2013.000	8.1	104.81010	
413	414	2013.500	6.5	90.45606	

326 rows × 8 columns

Next steps: Generate code with filtered_data | View recommended plots | New interactive sheet

```
house_age = 20
filtered_data = data[data['X2_house_age'] > house_age]
filtered_data
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_conve
0	1	2012.917	32.0	84.87882	
6	7	2012.667	34.5	623.47310	
7	8	2013.417	20.3	287.60250	
8	9	2013.500	31.7	5512.03800	
10	11	2013.083	34.8	405.21340	
...	...	...	...	...	
395	396	2012.917	21.2	512.54870	
396	397	2012.667	37.1	918.63570	
400	401	2013.250	26.8	482.75810	
403	404	2012.667	30.9	161.94200	
405	406	2012.667	23.0	130.99450	

137 rows × 8 columns

Next steps: Generate code with filtered_data | View recommended plots | New interactive sheet

```
data
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_conve
0	1	2012.917	32.0	84.87882	
1	2	2012.917	19.5	306.59470	
2	3	2013.583	13.3	561.98450	
3	4	2013.500	13.3	561.98450	
4	5	2012.833	5.0	390.56840	
...	...	...	...	...	
409	410	2013.000	13.7	4082.01500	
410	411	2012.667	5.6	90.45606	
411	412	2013.250	18.8	390.96960	
412	413	2013.000	8.1	104.81010	
413	414	2013.500	6.5	90.45606	

414 rows × 8 columns

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
data_sorted_age = data.sort_values(by='X2_house_age')
data_sorted_age
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_conve
166	167	2013.417	0.0	292.99780	
103	104	2012.750	0.0	208.39050	
373	374	2013.083	0.0	274.01440	
105	106	2012.833	0.0	292.99780	
123	124	2013.417	0.0	185.42960	
...	...	...	...	...	
128	129	2013.083	41.3	124.99120	
173	174	2013.083	41.3	401.88070	
361	362	2013.083	41.4	281.20500	
392	393	2013.083	42.7	443.80200	
192	193	2013.167	43.8	57.58945	

414 rows × 8 columns

Next steps: Generate code with data_sorted_age | View recommended plots | New interactive sheet

```
def assign_group(row):
  if row['X2_house_age'] <= 10:
    return 'Group 1'
  elif row['X2_house_age'] <= 20:
    return 'Group 2'
  else:
    return 'Group 3'

data['group'] = data.apply(assign_group, axis=1)
```

```
data.head()
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_convenie
0	1	2012.917	32.0	84.87882	
1	2	2012.917	19.5	306.59470	
2	3	2013.583	13.3	561.98450	
3	4	2013.500	13.3	561.98450	
4	5	2012.833	5.0	390.56840	

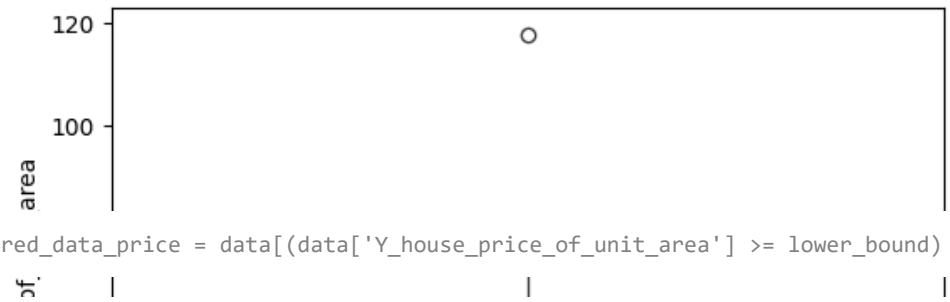
Next steps: Generate code with data | View recommended plots | New interactive sheet

```
average_price_by_group = data.groupby('group')['Y_house_price_of_unit_area'].mean()
print(average_price_by_group)
```

```
group
Group 1    46.608182
Group 2    34.138323
Group 3    35.735766
Name: Y_house_price_of_unit_area, dtype: float64
```

```
q1 = np.percentile(data['Y_house_price_of_unit_area'], 25)
q3 = np.percentile(data['Y_house_price_of_unit_area'], 75)
IQR = q3 - q1
lower_bound = q1 - 1.5 * IQR
upper_bound = q3 + 1.5 * IQR
```

```
sns.boxplot(data['Y_house_price_of_unit_area'])
plt.show()
```




```
filtered_data_price = data[(data['Y_house_price_of_unit_area'] >= lower_bound) & (data['Y_house_price_of_un
```

```
filtered_data_price
```

	No	X1_transaction_date	X2_house_age	X3_distance_to_the_nearest_MRT_station	X4_number_of_conve
0	1	2012.917	32.0	84.87882	
1	2	2012.917	19.5	306.59470	
2	3	2013.583	13.3	561.98450	
3	4	2013.500	13.3	561.98450	
4	5	2012.833	5.0	390.56840	
...	...	...	...	...	
409	410	2013.000	13.7	4082.01500	
410	411	2012.667	5.6	90.45606	
411	412	2013.250	18.8	390.96960	
412	413	2013.000	8.1	104.81010	
413	414	2013.500	6.5	90.45606	