Coursera Project for EDA

This is for the coursera project for explarotary data analysis

In this section, we will usa data from UCI reprisitory, where there are numerous dataset for machine learning practice and analysis. In this project we will use data Real estate valuation data set, where you may find the dataset in [https://archive.ics.uci.edu/ml/machine-learning-databases/00477/ (https://archive.ics.uci.edu/ml/machine-learning-databases/00477/)](https://archive.ics.uci.edu/ml/machine-learning-databases/00477/)

1. We will download the library

In [1]:

```
import seaborn as sns
import pandas as pd
import numpy as np
from seaborn import relplot
import matplotlib.pyplot as plt
%matplotlib inline
```

2. We downloaded the file from UCI reprisitory using pandas, the data is in xlsx format , then we will use the panda read_excel function

In [2]:

```
df =pd.read_excel('Realestate.xlsx')
```

3. By using the pandas dataframe function, we will see the details on the dataset

a. See the dataframe

In [3]:

```
df
```

Out[3]:

	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	2012.916667	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.916667	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583333	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500000	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833333	5.0	390.56840	5	24.97937	121.54245	43.1
...	...	...	...	...	...	...	...	...
409	410	2013.000000	13.7	4082.01500	0	24.94155	121.50381	15.4
410	411	2012.666667	5.6	90.45606	9	24.97433	121.54310	50.0
411	412	2013.250000	18.8	390.96960	7	24.97923	121.53986	40.6
412	413	2013.000000	8.1	104.81010	5	24.96674	121.54067	52.5
413	414	2013.500000	6.5	90.45606	9	24.97433	121.54310	63.9

414 rows × 8 columns

b. Check for null value

In [4]:

```
df.isnull().sum()
```

Out[4]:

```
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
```

c. see all information in the data set

In [5]:

```
df.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
```

In [6]:

```
df.describe()
```

Out[6]:

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitu
count	414.000000	414.000000	414.000000	414.000000	414.000000	414.000000	414.0000
mean	207.500000	2013.148953	17.712560	1083.885689	4.094203	24.969030	121.5333
std	119.655756	0.281995	11.392485	1262.109595	2.945562	0.012410	0.0153
min	1.000000	2012.666667	0.000000	23.382840	0.000000	24.932070	121.4735
25%	104.250000	2012.916667	9.025000	289.324800	1.000000	24.963000	121.5280
50%	207.500000	2013.166667	16.100000	492.231300	4.000000	24.971100	121.5386
75%	310.750000	2013.416667	28.150000	1454.279000	6.000000	24.977455	121.5433
max	414.000000	2013.583333	43.800000	6488.021000	10.000000	25.014590	121.5662

d. Deleting null value and see the information in the data set

In [7]:

```
df[df.isnull().any(axis=1)]
```

Out[7]:

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

In [8]:

```
df = df.dropna()
df = df.reset_index(drop=True)

print (df)
```

```
      No  X1 transaction date  X2 house age  \
0      1         2012.916667          32.0
1      2         2012.916667          19.5
2      3         2013.583333          13.3
3      4         2013.500000          13.3
4      5         2012.833333           5.0
..   ...                 ...           ...
409  410         2013.000000          13.7
410  411         2012.666667           5.6
411  412         2013.250000          18.8
412  413         2013.000000           8.1
413  414         2013.500000           6.5

     X3 distance to the nearest MRT station  X4 number of convenience stor
es  \
0                                  84.87882
10
1                                 306.59470
9
2                                 561.98450
5
3                                 561.98450
5
4                                 390.56840
5
..                                      ...
...
409                              4082.01500
0
410                                90.45606
9
411                               390.96960
7
412                               104.81010
5
413                                90.45606
9

     X5 latitude  X6 longitude  Y house price of unit area
0       24.98298     121.54024                        37.9
1       24.98034     121.53951                        42.2
2       24.98746     121.54391                        47.3
3       24.98746     121.54391                        54.8
4       24.97937     121.54245                        43.1
..           ...           ...                         ...
409     24.94155     121.50381                        15.4
410     24.97433     121.54310                        50.0
411     24.97923     121.53986                        40.6
412     24.96674     121.54067                        52.5
413     24.97433     121.54310                        63.9

[414 rows x 8 columns]
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
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
```

In [10]:

```
df.describe()
```

Out[10]:

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X longitu
count	414.000000	414.000000	414.000000	414.000000	414.000000	414.000000	414.0000
mean	207.500000	2013.148953	17.712560	1083.885689	4.094203	24.969030	121.5333
std	119.655756	0.281995	11.392485	1262.109595	2.945562	0.012410	0.0153
min	1.000000	2012.666667	0.000000	23.382840	0.000000	24.932070	121.4735
25%	104.250000	2012.916667	9.025000	289.324800	1.000000	24.963000	121.5280
50%	207.500000	2013.166667	16.100000	492.231300	4.000000	24.971100	121.5386
75%	310.750000	2013.416667	28.150000	1454.279000	6.000000	24.977455	121.5433
max	414.000000	2013.583333	43.800000	6488.021000	10.000000	25.014590	121.5662

In [11]:

```
df.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
```

3. Explarotary data analysis

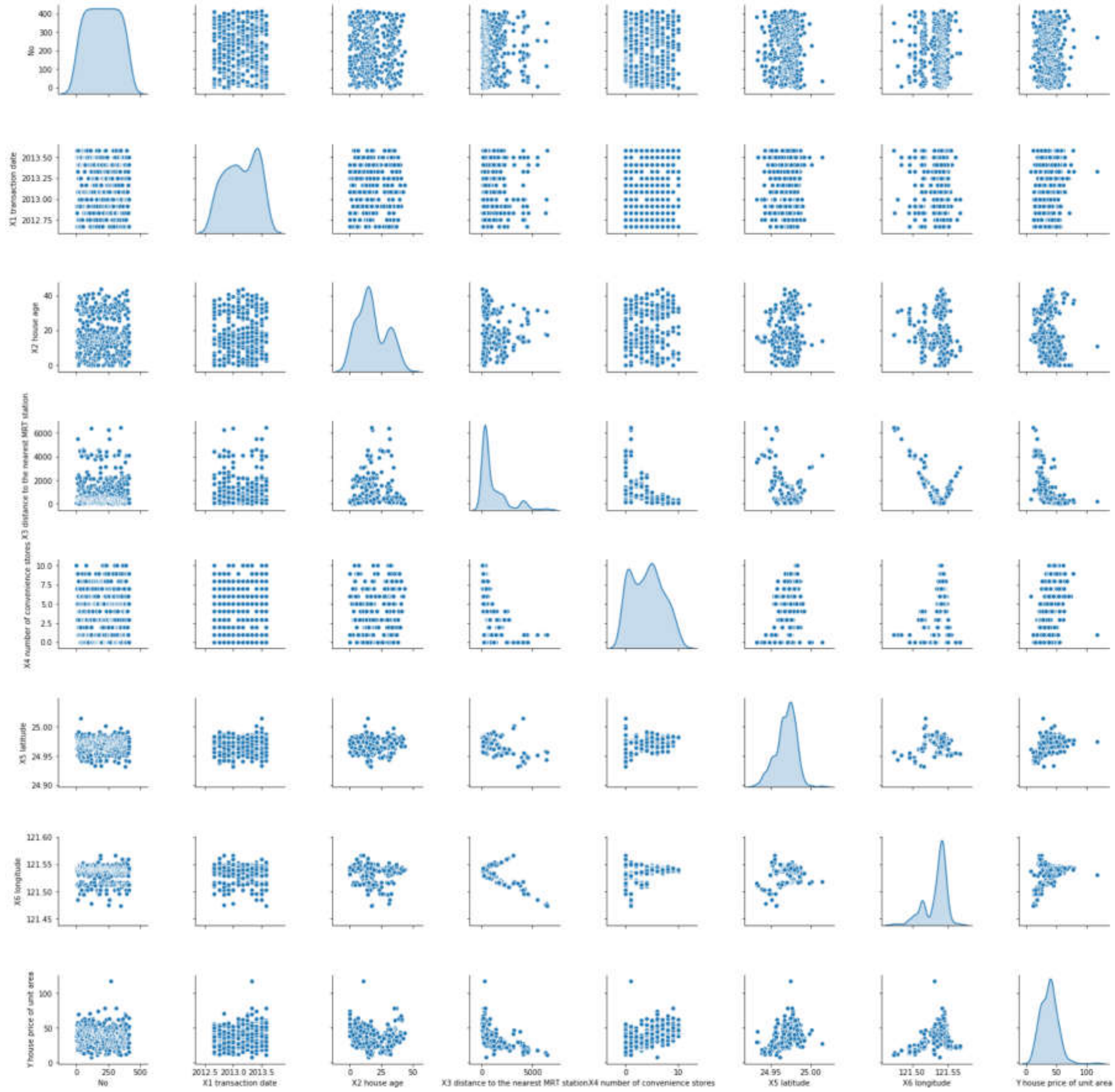
a. Pairplot the data to see what the correlation among the data

In [13]:

```
sns.pairplot(df.iloc[:, np.hstack(([0], range(1, 8)))], diag_kind='kde')
```

Out[13]:

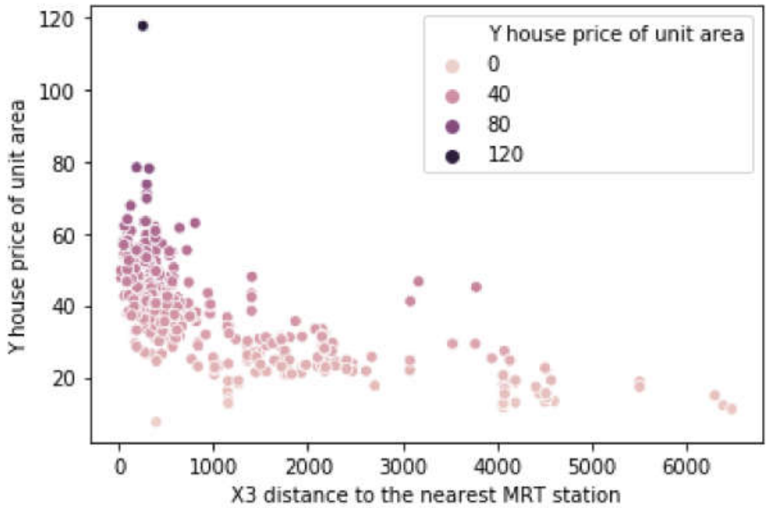
```
<seaborn.axisgrid.PairGrid at 0x1782c338e48>
```


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

b. to see the corelation we plot the Y price to the house age ,distance to the nearest MRT station , number of convenience stores

In [14]:

```
ax = sns.scatterplot(x="X3 distance to the nearest MRT station", y="Y house price of un
it area", hue="Y house price of unit area",
                     data=df)
```

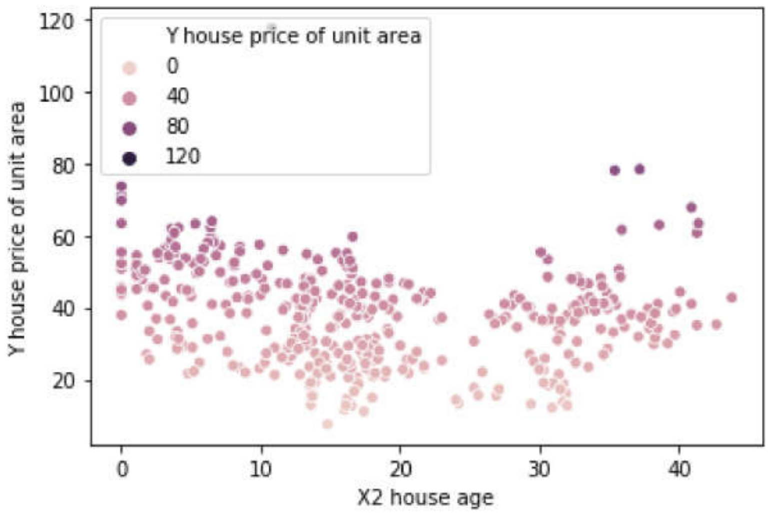


In [15]:

```
ax = sns.scatterplot(x="X4 number of convenience stores", y="Y house price of unit are
a", hue="Y house price of unit area",
                     data=df)
```

In [17]:

```
ax = sns.scatterplot(x="X2 house age", y="Y house price of unit area", hue="Y house pri
ce of unit area",
                     data=df)
```



c. Box Plot all the data to see the outliers

In [19]:

```
data = df[['Y house price of unit area','X2 house age', 'X4 number of convenience store
s', 'X3 distance to the nearest MRT station' ]]
```

In [20]:

```
data
```

Out[20]:

	Y house price of unit area	X2 house age	X4 number of convenience stores	X3 distance to the nearest MRT station
0	37.9	32.0	10	84.87882
1	42.2	19.5	9	306.59470
2	47.3	13.3	5	561.98450
3	54.8	13.3	5	561.98450
4	43.1	5.0	5	390.56840
...	...	...	...	...
409	15.4	13.7	0	4082.01500
410	50.0	5.6	9	90.45606
411	40.6	18.8	7	390.96960
412	52.5	8.1	5	104.81010
413	63.9	6.5	9	90.45606

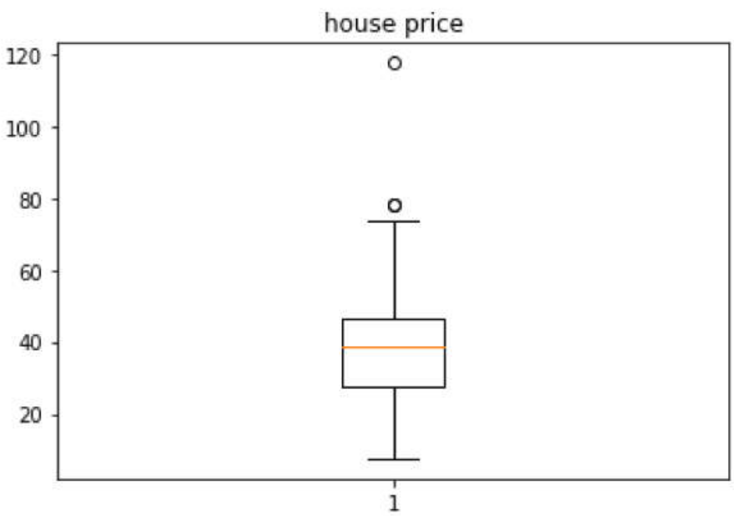
414 rows × 4 columns

In [27]:

```
fig1, ax1 = plt.subplots()
ax1.set_title('house price')
ax1.boxplot(df['Y house price of unit area'])
```

Out[27]:

```
{'whiskers': [<matplotlib.lines.Line2D at 0x1783feaf6c8>,
  <matplotlib.lines.Line2D at 0x1783feaff48>],
 'caps': [<matplotlib.lines.Line2D at 0x1783feafec8>,
  <matplotlib.lines.Line2D at 0x1783fea8f08>],
 'boxes': [<matplotlib.lines.Line2D at 0x1783fe82b88>],
 'medians': [<matplotlib.lines.Line2D at 0x1783fea8fc8>],
 'fliers': [<matplotlib.lines.Line2D at 0x1783feaba48>],
 'means': []}
```



from the boxplot there are no significant outliers in the Y price

In [31]:

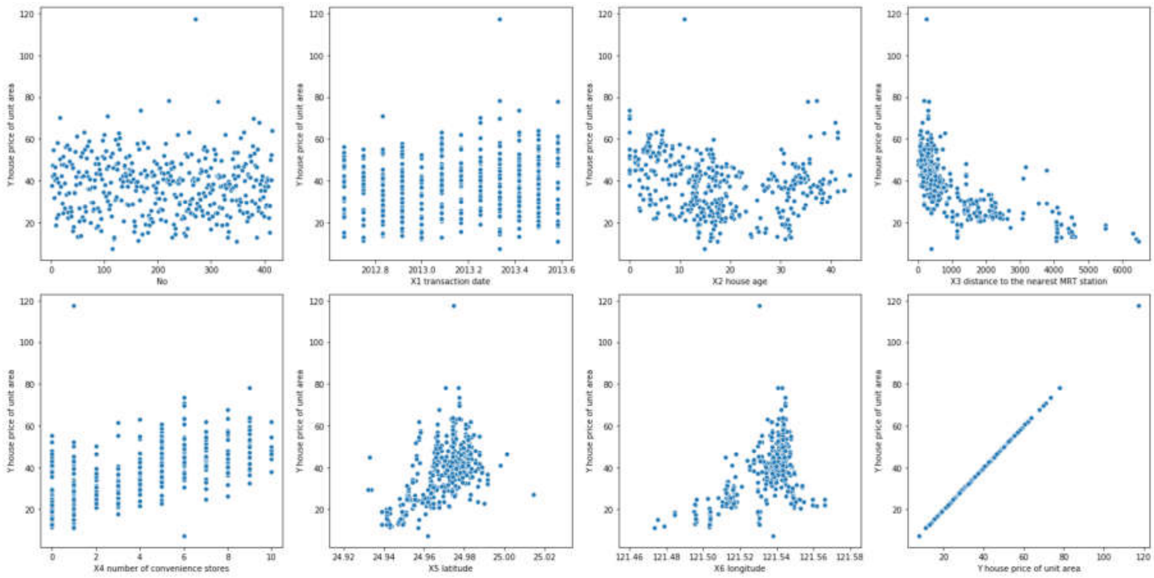
```
rows = 2
cols = 4

fig, ax = plt.subplots(rows, cols, figsize = (20, 10))

col = df.columns
index = 0

for i in range(rows):
    for j in range(cols):
        sns.scatterplot(x = col[index], y = 'Y house price of unit area', data = df, ax = ax[i][j])
        index = index + 1

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



from the pairplot above there are corelation between the number of convenience store, distance to MRT and house age

d. Create a corelation heatmap

In [35]:

```
cormat = df.corr()
cormat
```

Out[35]:

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude
No	1.000000	-0.048634	-0.032808	-0.013573	-0.012699	-0.010110	-0.011059
X1 transaction date	-0.048634	1.000000	0.017542	0.060880	0.009544	0.035016	-0.041065
X2 house age	-0.032808	0.017542	1.000000	0.025622	0.049593	0.054420	-0.048520
X3 distance to the nearest MRT station	-0.013573	0.060880	0.025622	1.000000	-0.602519	-0.591067	-0.806317
X4 number of convenience stores	-0.012699	0.009544	0.049593	-0.602519	1.000000	0.444143	0.449099
X5 latitude	-0.010110	0.035016	0.054420	-0.591067	0.444143	1.000000	0.412924
X6 longitude	-0.011059	-0.041065	-0.048520	-0.806317	0.449099	0.412924	1.000000
Y house price of unit area	-0.028587	0.087529	-0.210567	-0.673613	0.571005	0.546307	0.523287

In [37]:

```
cormat.shape
```

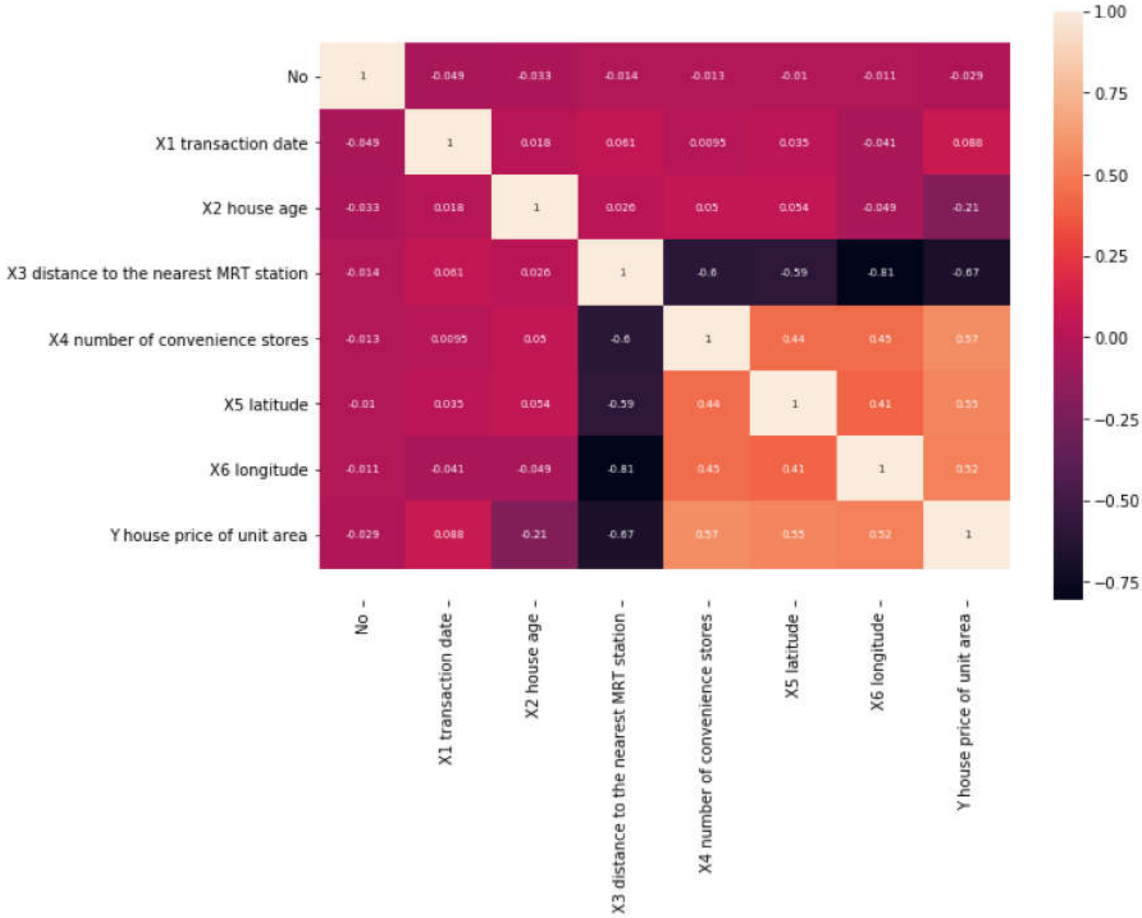
Out[37]:

```
(8, 8)
```

In [38]:

```
fig, ax = plt.subplots(figsize = (10, 7))
sns.heatmap(corrmat, annot = True, annot_kws = {'size': 7})

bottom, top = ax.get_ylim()
ax.set_ylim(bottom+0.5, top-.5)
plt.show()
```



In [39]:

```
def getCorrelatedFeature(corrdata, threshold):
    feature = []
    value = []

    for i, index in enumerate(corrdata.index):
        if abs(corrdata[index]) > threshold:
            feature.append(index)
            value.append(corrdata[index])

    df = pd.DataFrame(data = value, index=feature, columns=['corr value'])

    return df
```

In [40]:

```
threshold = 0.5
corr_df = getCorrelatedFeature(corrmat['Y house price of unit area'], threshold)
corr_df
```

Out[40]:

	corr value
X3 distance to the nearest MRT station	-0.673613
X4 number of convenience stores	0.571005
X5 latitude	0.546307
X6 longitude	0.523287
Y house price of unit area	1.000000

The hypotesis for this data are :

1. Ho : there is strong corelation between Y House price and distance the nearest MRT station, H1 : there is no corelation between Y House price and distance the nearest MRT station

2. Ho : there is strong corelation between Y House price and distance the number of convinenet store, H1 : there is no corelation between Y House price and distance the number of convinenet store

3. Ho : there is strong corelation between Y House price and house age H1 : there is no corelation between Y House price and house age

Testing the hypotesis using pearson corelation test

In [49]:

```
###### the Pearson's Correlation test
from scipy import stats
```

In [56]:

```
X4 = np.array(df['X4 number of convenience stores'])
Y = np.array(df['Y house price of unit area'])
X3 = np.array(df['X3 distance to the nearest MRT station'])
X2 = np.array(df['X2 house age'])
```

In [58]:

```
stats.pearsonr(Y,X4)
```

Out[58]:

```
(0.5710049111111494, 3.4134833404947028e-37)
```

Conduct the pearson corelation test from Y price to all variable

In [61]:

```
# Pearson's Correlation test Y to X4 ( convienience store)
from scipy.stats import pearsonr
stat, p = pearsonr(Y, X4)
print('stat=%.3f, p=%.3f' % (stat, p))
if p > 0.05:
        print('Probably independent')
else:
        print('Probably dependent')
```

```
stat=0.571, p=0.000
Probably dependent
```

In [62]:

```
# Pearson's Correlation test Y to X3 ( distance to MRT station)
from scipy.stats import pearsonr
stat, p = pearsonr(Y, X3)
print('stat=%.3f, p=%.3f' % (stat, p))
if p > 0.05:
        print('Probably independent')
else:
        print('Probably dependent')
```

```
stat=-0.674, p=0.000
Probably dependent
```

In [63]:

```
# Pearson's Correlation test Y to X2 ( House Age)
from scipy.stats import pearsonr
stat, p = pearsonr(Y, X4)
print('stat=%.3f, p=%.3f' % (stat, p))
if p > 0.05:
        print('Probably independent')
else:
        print('Probably dependent')
```

```
stat=0.571, p=0.000
Probably dependent
```

From the above test using pearson corelation it was found that there are corelations between House price and the House Age, distance to MRT Station and numbers of convinient store, so all of the Ho is accepted and H1 is rejected

Next steps are

1. Condutcing a machine learning analysis using supervised learning (regression) using a multivariable

Summary :

1. The data not need to be clean up since it was in a great structure

2. There are no missing data. even there is, the necessary action has been done

3. There are no significant outliers was found in the data

4. There are significant corelation based on the pearson corelation test between the data from price and house age, distance to MRT station and numbers of convenient store

5. Another data that can be add is the criminal data in the neighbourhodd , is this has effect to the

In [ ]:

```

```