ujkviskl6

August 8, 2024

[22]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

%matplotlib inline
```

[26]:
```
house_data = pd.read_csv('/content/data.csv')
```

[34]:
```
house_data.head()
```

[34]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot
0	2014-05-02 00:00:00	313000.0	3.0	1.50	1340	7912
1	2014-05-02 00:00:00	2384000.0	5.0	2.50	3650	9050
2	2014-05-02 00:00:00	342000.0	3.0	2.00	1930	11947
3	2014-05-02 00:00:00	420000.0	3.0	2.25	2000	8030
4	2014-05-02 00:00:00	550000.0	4.0	2.50	1940	10500

	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built
0	1.5	0	0	3	1340	0	1955
1	2.0	0	4	5	3370	280	1921
2	1.0	0	0	4	1930	0	1966
3	1.0	0	0	4	1000	1000	1963
4	1.0	0	0	4	1140	800	1976

	yr_renovated	street	city	statezip	country
0	2005	18810 Densmore Ave N	Shoreline	WA 98133	USA
1	0	709 W Blaine St	Seattle	WA 98119	USA
2	0	26206-26214 143rd Ave SE	Kent	WA 98042	USA
3	0	857 170th Pl NE	Bellevue	WA 98008	USA
4	1992	9105 170th Ave NE	Redmond	WA 98052	USA

[35]:
```
house_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           4600 non-null   object
 1   price          4600 non-null   float64
 2   bedrooms       4600 non-null   float64
 3   bathrooms      4600 non-null   float64
 4   sqft_living    4600 non-null   int64
 5   sqft_lot       4600 non-null   int64
 6   floors         4600 non-null   float64
 7   waterfront     4600 non-null   int64
 8   view           4600 non-null   int64
 9   condition      4600 non-null   int64
 10  sqft_above     4600 non-null   int64
 11  sqft_basement  4600 non-null   int64
 12  yr_built       4600 non-null   int64
 13  yr_renovated   4600 non-null   int64
 14  street         4600 non-null   object
 15  city           4600 non-null   object
 16  statezip       4600 non-null   object
 17  country        4600 non-null   object
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
```

[36]:
```
house_data.describe()
```

[36]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot
count	4.600000e+03	4600.000000	4600.000000	4600.000000	4.600000e+03
mean	5.519630e+05	3.400870	2.160815	2139.346957	1.485252e+04
std	5.638347e+05	0.908848	0.783781	963.206916	3.588444e+04
min	0.000000e+00	0.000000	0.000000	370.000000	6.380000e+02
25%	3.228750e+05	3.000000	1.750000	1460.000000	5.000750e+03
50%	4.609435e+05	3.000000	2.250000	1980.000000	7.683000e+03
75%	6.549625e+05	4.000000	2.500000	2620.000000	1.100125e+04
max	2.659000e+07	9.000000	8.000000	13540.000000	1.074218e+06

	floors	waterfront	view	condition	sqft_above
count	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000
mean	1.512065	0.007174	0.240652	3.451739	1827.265435
std	0.538288	0.084404	0.778405	0.677230	862.168977
min	1.000000	0.000000	0.000000	1.000000	370.000000
25%	1.000000	0.000000	0.000000	3.000000	1190.000000
50%	1.500000	0.000000	0.000000	3.000000	1590.000000
75%	2.000000	0.000000	0.000000	4.000000	2300.000000
max	3.500000	1.000000	4.000000	5.000000	9410.000000

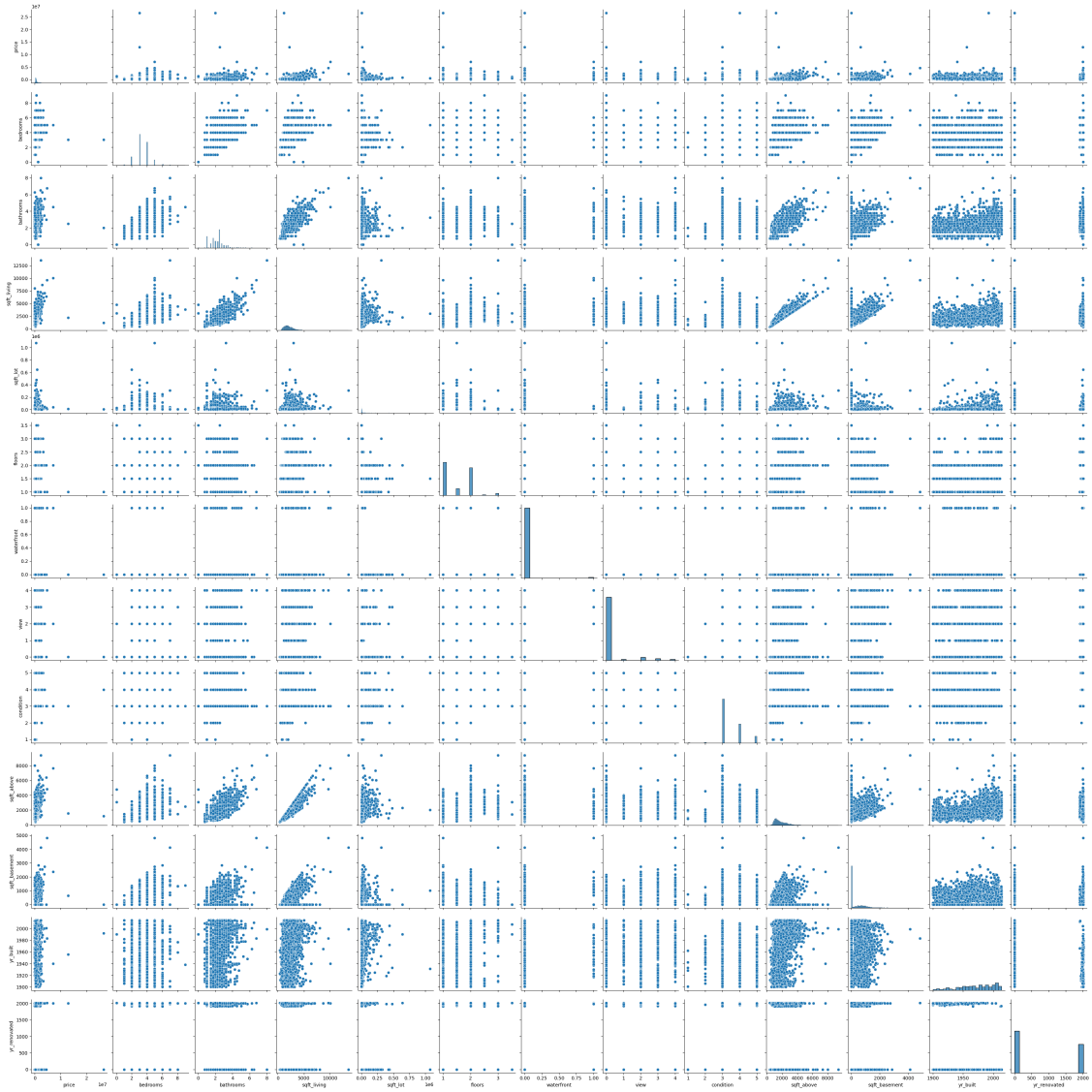
```
       sqft_basement     yr_built  yr_renovated
count    4600.000000  4600.000000   4600.000000
mean      312.081522  1970.786304    808.608261
std       464.137228    29.731848    979.414536
min         0.000000  1900.000000      0.000000
25%         0.000000  1951.000000      0.000000
50%         0.000000  1976.000000      0.000000
75%       610.000000  1997.000000   1999.000000
max      4820.000000  2014.000000   2014.000000
```

[37]:
```
house_data.columns
```

[37]:
```
Index(['date', 'price', 'bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot',
       'floors', 'waterfront', 'view', 'condition', 'sqft_above',
       'sqft_basement', 'yr_built', 'yr_renovated', 'street', 'city',
       'statezip', 'country'],
      dtype='object')
```

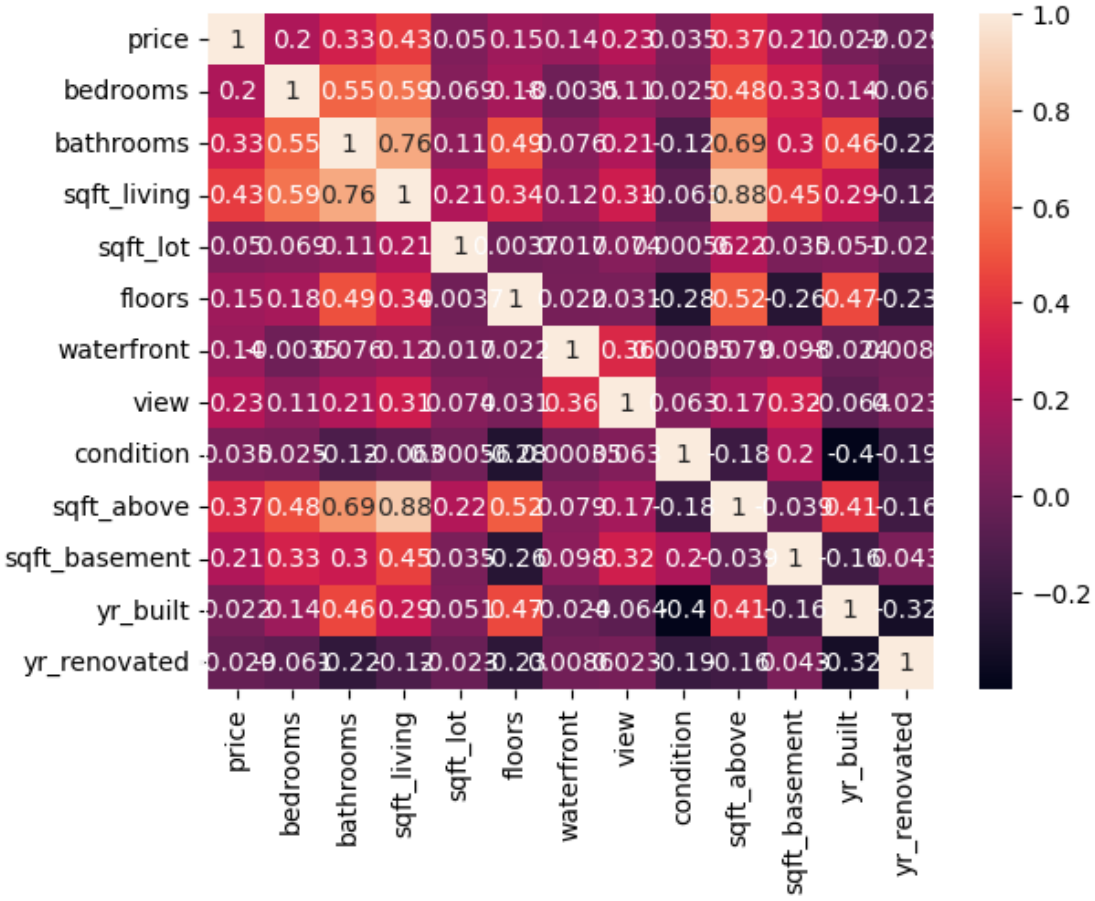
[38]:
```
sns.pairplot(house_data)
```

[38]:
```
<seaborn.axisgrid.PairGrid at 0x7f625ae7caf0>
```



[41]:
```
numeric_data = house_data.select_dtypes(include=['number'])
sns.heatmap(numeric_data.corr(), annot=True)
```

[41]: <Axes: >



[28]:
```
X = house_data[['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',␣
↪'waterfront', 'view', 'condition']]
y = house_data['price']
```

[29]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[30]:
```
feature_names = ['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors',␣
↪'waterfront', 'view', 'condition']
```

[31]:
```
X_train.columns = feature_names
X_test.columns = feature_names
```

[42]:
```
ml = LinearRegression()
```

[43]:
```
ml.fit(X_train, y_train)
```

[43]:
```
LinearRegression()
```

[45]:
```
y_pred = ml.predict(X_test)
```

Model evaluation performance

[46]:
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

[47]:
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

[51]:
```
print("Mean Squared Error:", mse)
print("R-squared:", r2)
```

```
Mean Squared Error: 986869414953.98
R-squared: 0.03233518995632512
```

Example: Predict the price of a house with 3 bedrooms, 2 bathrooms, 1500 sqft living area, 4000 sqft lot, 1 floor, not waterfront, no view, and condition 3:*italicised text*

[67]:
```
bedrooms = 2
bathrooms = 2
sqft_living_area = 2000
sqft_lot = 4000
floor = 2
waterfront = 0
view = 1
condition =
```

[70]:
```
new_data = [[bedrooms, bathrooms, sqft_living_area, sqft_lot, floor,␣
 ↪waterfront, view, condition]]
predicted_price = model.predict(new_data)
print(f'Predicted Price:, ${predicted_price[0]:,.2f}')
```

```
Predicted Price:, $264,430.15
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:465: UserWarning: X does
not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```