In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
%matplotlib inline
```

In [2]:
```
file_name='https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0101EN/coursera/project/kc_house_data_NaN
df=pd.read_csv(file_name)
```

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	...	grade	sqft_above	sqft_basement	yr_built	yr_r
0	0	7129300520	20141013T000000	221900.0	3.0	1.00	1180	5650	1.0	0	...	7	1180	0	1955	
1	1	6414100192	20141209T000000	538000.0	3.0	2.25	2570	7242	2.0	0	...	7	2170	400	1951	
2	2	5631500400	20150225T000000	180000.0	2.0	1.00	770	10000	1.0	0	...	6	770	0	1933	
3	3	2487200875	20141209T000000	604000.0	4.0	3.00	1960	5000	1.0	0	...	7	1050	910	1965	
4	4	1954400510	20150218T000000	510000.0	3.0	2.00	1680	8080	1.0	0	...	8	1680	0	1987	

5 rows × 22 columns

In [4]:
```
print(df.dtypes)
```

```
Unnamed: 0         int64
id                 int64
date              object
price            float64
bedrooms         float64
bathrooms        float64
sqft_living        int64
sqft_lot           int64
floors           float64
waterfront         int64
view               int64
condition          int64
grade              int64
sqft_above         int64
sqft_basement      int64
yr_built           int64
yr_renovated       int64
zipcode            int64
lat              float64
long             float64
sqft_living15      int64
sqft_lot15         int64
dtype: object
```

In [5]:
```
df=pd.read_csv(file_name)

df.drop(["id", "Unnamed: 0"], axis=1, inplace = True)

df.describe()
```

Out[5]:

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_b
count	2.161300e+04	21600.000000	21603.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	2161
mean	5.400881e+05	3.372870	2.115736	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430	7.656873	1788.390691	29
std	3.671272e+05	0.926657	0.768996	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743	1.175459	828.090978	44
min	7.500000e+04	1.000000	0.500000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000	1.000000	290.000000	
25%	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000	7.000000	1190.000000	
50%	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000	7.000000	1560.000000	
75%	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000	8.000000	2210.000000	56
max	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000	13.000000	9410.000000	482

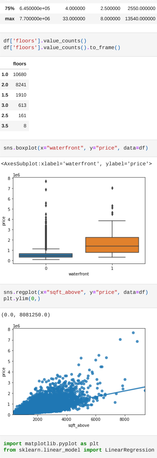
In [6]:
```
df['floors'].value_counts()
df['floors'].value_counts().to_frame()
```

Out[6]:

	floors
1.0	10680
2.0	8241
1.5	1910
3.0	613
2.5	161
3.5	8

In [7]:
```
sns.boxplot(x="waterfront", y="price", data=df)
```

Out[7]:
```
<AxesSubplot:xlabel='waterfront', ylabel='price'>
```



In [8]:
```
sns.regplot(x="sqft_above", y="price", data=df)
plt.ylim(0,)
```

Out[8]:
```
(0.0, 8081250.0)
```

In [9]:
```
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

In [10]:
```
X = df[['long']]
Y = df['price']
lm = LinearRegression()
lm
lm.fit(X,Y)
lm.score(X, Y)
```

Out[10]:
```
0.00046769430149007363
```

In [11]:
```
lm = LinearRegression()
lm

X = df[['sqft_living']]
Y = df['price']

lm.fit(X,Y)

lm.score(X,Y)
```

Out[11]:
```
0.4928532179037931
```

In [12]:
```
features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view" ,"bathrooms",
"sqft_living15","sqft_above","grade","sqft_living"]
```

In [13]:
```
lm = LinearRegression()
lm

X = df[["floors"]]
Y = df['price']

lm.fit(X,Y)
lm.score(X,Y)
```

Out[13]:
```
0.0656943100683410907
```

In [14]:
```
lm = LinearRegression()
lm

X = df[['waterfront']]
Y = df['price']

lm.fit(X,Y)
lm.score(X,Y)
```

Out[14]:
```
0.07095267538578309
```

In [15]:
```
lm = LinearRegression()
lm

X = df['lat']]
Y = df['price']

lm.fit(X,Y)
lm.score(X,Y)
```

Out[15]:
```
0.09425113672917462
```

In [16]:
```
lm = LinearRegression()
lm

X = df['sqft_basement']]
Y = df['price']

lm.fit(X,Y)
lm.score(X,Y)
```

Out[16]:
```
0.10485681526979744
```

In [17]:
```
lm = LinearRegression()
lm

X = df["view"]]
Y = df['price']

lm.fit(X,Y)

lm.score(X,Y)
```

Out[17]:
```
0.15784211584121544
```

In [18]:
```
lm = LinearRegression()
lm

X = df['sqft_living15']]
Y = df['price']

lm.fit(X,Y)

lm.score(X,Y)
```

Out[18]:
```
0.3426684607560172
```

In [19]:
```
lm = LinearRegression()
lm

X = df['sqft_above']]
Y = df['price']

lm.fit(X,Y)

lm.score(X,Y)
```

Out[19]:
```
0.3667117528382793
```

In [20]:
```
lm = LinearRegression()
lm

X = df['grade']]
Y = df['price']

lm.fit(X,Y)

lm.score(X,Y)
```

Out[20]:
```
0.44546846010928724
```

In [21]:
```
lm = LinearRegression()
lm

X = df['sqft_living']]
Y = df['price']

lm.fit(X,Y)
lm.score(X,Y)
```

Out[21]:
```
0.4928532179037931
```

In [22]:
```
Input=[('scale',StandardScaler()),('polynomial', PolynomialFeatures(include_bias=False)),
('model',LinearRegression())]
```

In [23]:
```
pipe=Pipeline(Input)
pipe
```

Out[23]:
```
Pipeline(steps=[('scale', StandardScaler()),
                ('polynomial', PolynomialFeatures(include_bias=False)),
                ('model', LinearRegression())])
```

In [24]:
```
pipe.fit(X,Y)
```

Out[24]:
```
Pipeline(steps=[('scale', StandardScaler()),
                ('polynomial', PolynomialFeatures(include_bias=False)),
                ('model', LinearRegression())])
```

In [25]:
```
pipe.score(X,Y)
```

Out[25]:
```
0.5327430940691443
```

In [26]:
```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
print("done")
```

```
done
```

In [27]:
```
features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view" ,"bathrooms","sqft_living15","sqft_above","grade","sqft_li
X = df[features ]
Y = df['price']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.15, random_state=1)


print("number of test samples :", x_test.shape[0])
print("number of training samples:",x_train.shape[0])
```

```
number of test samples : 3242
number of training samples: 18371
```

In [28]:
```
from sklearn.linear_model import Ridge
```

In [42]:
```
pr=PolynomialFeatures(degree=2)
x_train_pr=pr.fit_transform(x_train[['floors', 'waterfront','lat' ,'sqft_basement' ,'view' ,'sqft_living15','sqft_above','grade','sqft
x_test_pr=pr.fit_transform(x_test[['floors', 'waterfront','lat' ,'sqft_basement' ,'view' ,'sqft_living15','sqft_above','grade','sqft_li
```

In [43]:
```
RidgeModel=Ridge(alpha=0.1)

RidgeModel.fit(x_train_pr, y_train)
```

Out[43]:
```
Ridge(alpha=0.1)
```

In [44]:
```
RidgeModel.score(x_train_pr, y_train)
```

Out[44]:
```
0.7392211454565541
```

In [45]:
```
from sklearn.preprocessing import PolynomialFeatures
```

In [46]:
```
pr=PolynomialFeatures(degree=2)
pr
```

Out[46]:
```
PolynomialFeatures()
```

In [48]:
```
x_train_pr=pr.fit_transform(x_train[['floors', 'waterfront','lat' ,'sqft_basement' ,'view','sqft_living15','sqft_above','grade','sqft_
```

In [50]:
```
x_poly=pr.fit_transform(x_train[['floors', 'waterfront','lat' ,'sqft_basement' ,'view' ,'sqft_living15','sqft_above','grade','sqft_li
```

In [55]:
```
RidgeModel=Ridge(alpha=0.1)

RidgeModel.fit(x_train_pr, y_train)

RidgeModel.score(x_train_pr, y_train)
```

Out[55]:
```
0.7392211454565541
```

In [54]:
```
x_test_pr=pr.fit_transform(x_test[['floors', 'waterfront','lat','sqft_basement' ,'view' ,'sqft_living15','sqft_above','grade','sqft_liv

x_poly=pr.fit_transform(x_test[['floors', 'waterfront','lat' ,'sqft_basement' ,'view' ,'sqft_living15','sqft_above','grade','sqft_liv

RidgeModel=Ridge(alpha=0.1)

RidgeModel.fit(x_test_pr, y_test)

RidgeModel.score(x_test_pr, y_test)
```

Out[54]:
```
0.7549378887687774
```

In [ ]:
```

```