Data Analysis with Python

House Sales in King County, USA

This dataset contains house sale prices for King County, which includes Seattle. It includes homes sold between May 2014 and May 2015.

id :a notation for a house

date: Date house was sold

price: Price is prediction target

bedrooms: Number of Bedrooms/House

bathrooms: Number of bathrooms/bedrooms

sqft_living: square footage of the home

sqft_lot: square footage of the lot

floors :Total floors (levels) in house

waterfront :House which has a view to a waterfront

view: Has been viewed

condition :How good the condition is Overall

grade: overall grade given to the housing unit, based on King County grading system

sqft_above :square footage of house apart from basement

sqft_basement: square footage of the basement

yr_built :Built Year

yr_renovated :Year when house was renovated

zipcode:zip code

lat: Latitude coordinate

long: Longitude coordinate

sqft_living15 :Living room area in 2015(implies-- some renovations) This might or might not have affected the lotsize area

sqft_lot15 :lotSize area in 2015(implies-- some renovations)

You will require the following libraries

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
```

1.0 Importing the Data

Load the csv:

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

we use the method head to display the first 5 columns of the dataframe.

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

Question 1

Display the data types of each column using the attribute dtype, then take a screenshot and submit it, include your code in the image.

In [4]:
```
df.dtypes
```

Out[4]:
```
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
```

We use the method describe to obtain a statistical summary of the dataframe.

In [5]:

```
df.describe()
```

2.0 Data Wrangling

Question 2

Drop the columns "id" and "Unnamed: 0" from axis 1 using the method drop(), then use the method describe() to obtain a statistical summary of the data. Take a screenshot and submit it, make sure the inplace parameter is set to True

In [6]:

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

In [7]:

```
print("number of NaN values for the column bedrooms :",
df['bedrooms'].isnull().sum())
print("number of NaN values for the column bathrooms :",
df['bathrooms'].isnull().sum())
```

```
number of NaN values for the column bedrooms : 13
number of NaN values for the column bathrooms : 10
```

We can replace the missing values of the column 'bedrooms' with the mean of the column 'bedrooms' using the method replace. Don't forget to set the inplace parameter top True

In [8]:

```
mean=df['bedrooms'].mean()
df['bedrooms'].replace(np.nan,mean, inplace=True)
```

We also replace the missing values of the column 'bathrooms' with the mean of the column 'bedrooms' using the method replace.Don't forget to set the inplace parameter top Ture

In [9]:

```
mean=df['bathrooms'].mean()
df['bathrooms'].replace(np.nan,mean, inplace=True)
```

In [10]:

```
print("number of NaN values for the column bedrooms :",
df['bedrooms'].isnull().sum())
print("number of NaN values for the column bathrooms :",
df['bathrooms'].isnull().sum())
```

```
number of NaN values for the column bedrooms : 0
number of NaN values for the column bathrooms : 0
```

3.0 Exploratory data analysis

Question 3

Use the method value_counts to count the number of houses with unique floor values, use the method .to_frame() to convert it to a dataframe.

In [11]:

```
fl=df['floors'].value_counts()
fl.to_frame()
```

Out[11]:

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

Question 4

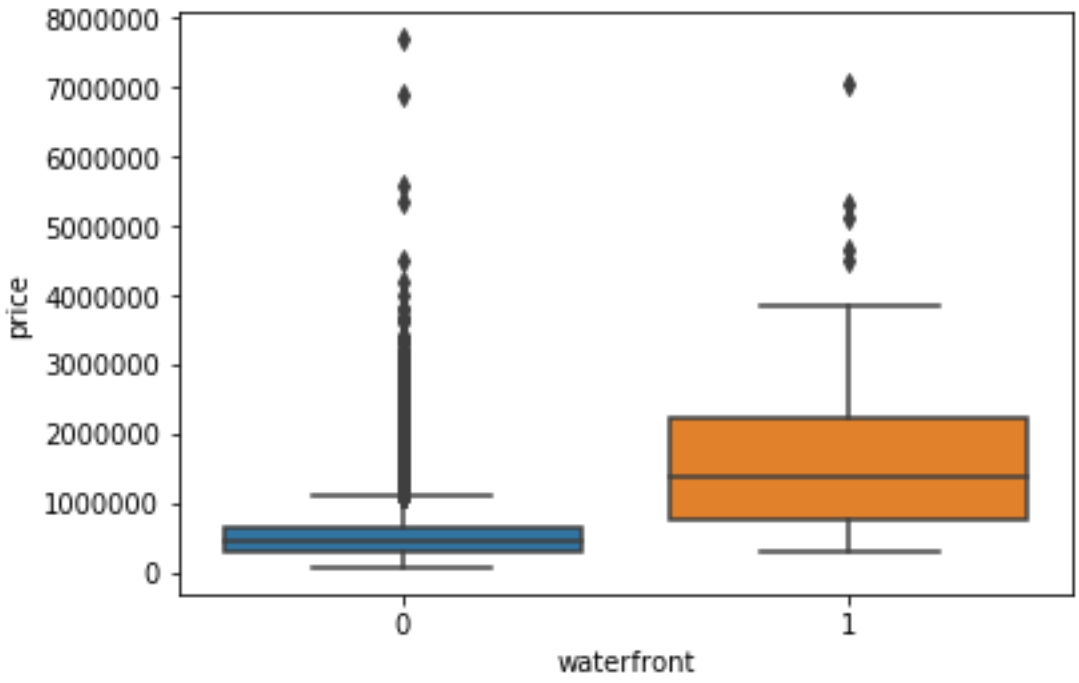
Use the function boxplot in the seaborn library to determine whether houses with a waterfront view or without a waterfront view have more price outliers .

In [12]:

```
sns.boxplot(x="waterfront", y="price", data=df)
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x13a3f39b3c8>
```



Question 5

Use the function regplot in the seaborn library to determine if the feature sqft_above is negatively or positively correlated with price.

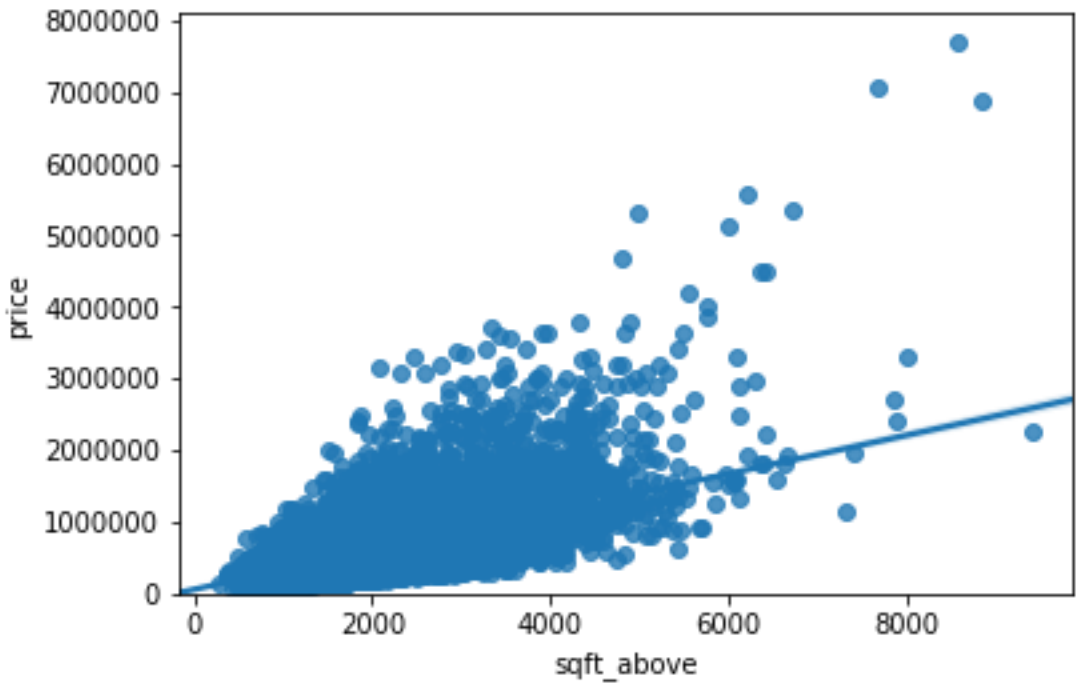
In [13]:

```
sns.regplot(x="sqft_above",y="price",data=df)
plt.ylim(0,)
```

```
C:\Users\doscsy12\Anaconda2\envs\py36\lib\site-packages\scipy\stats\stats.py:
1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing
is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future thi
s will be interpreted as an array index, `arr[np.array(seq)]`, which will res
ult either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

Out[13]:

```
(0, 8086186.822543947)
```



We can use the Pandas method corr() to find the feature other than price that is most correlated with price.

In [14]:

```
df.corr()['price'].sort_values()
```

Out[14]:

```
zipcode         -0.053203
long             0.021626
condition        0.036362
yr_built         0.054012
sqft_lot15       0.082447
sqft_lot         0.089661
yr_renovated     0.126434
floors           0.256794
waterfront       0.266369
lat              0.307003
bedrooms         0.308797
sqft_basement    0.323816
view             0.397293
bathrooms        0.525738
sqft_living15    0.585379
sqft_above       0.605567
grade            0.667434
sqft_living      0.702035
price            1.000000
Name: price, dtype: float64
```