```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Ridge
%matplotlib inline
```

```
from google.colab import files
uploaded=files.upload()
```

Choose Files No file chosen Upload
widget is only available when the cell has been

```
df=pd.read_csv('kc_house_data.csv')
df.head()
```

Kumar Utkarsh Resolve
Dec 23, 2021
This head() will display the data from the top of the given data sets.

	id	date	price	be
0	7129300520	20141013T000000	221900.0	
1	6414100192	20141209T000000	538000.0	
2	5631500400	20150225T000000	180000.0	
3	2487200875	20141209T000000	604000.0	
4	1954400510	20150218T000000	510000.0	

```
df.dtypes
```

Kumar Utkarsh Resolve
Dec 23, 2021
dtypes will display the data types of the given dataset.

```
id                 int64
date              object
price            float64
bedrooms           int64
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
```

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

	id	price	bedroo
count	2.161300e+04	2.161300e+04	21613.0000
mean	4.580302e+09	5.400881e+05	3.3708
std	2.876566e+09	3.671272e+05	0.9300
min	1.000102e+06	7.500000e+04	0.0000
25%	2.123049e+09	3.219500e+05	3.0000
50%	3.904930e+09	4.500000e+05	3.0000
75%	7.308900e+09	6.450000e+05	4.0000
max	9.900000e+09	7.700000e+06	33.0000

Kumar Utkarsh
Resolve
Dec 23, 2021

describe function gives the statistical summary of given data sets.

```
df["floors"].value_counts().to_frame()
```

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

Kumar Utkarsh
Resolve
Dec 23, 2021

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

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

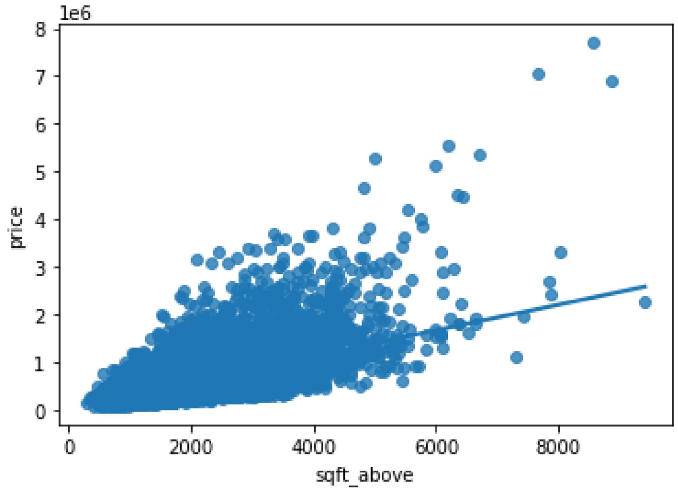
Kumar Utkarsh
Resolve
Dec 23, 2021

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

```
<matplotlib.axes._subplots.AxesSubplot at
0x7f30e07ec610>
```


```
sns.regplot(x="sqft_above", y="price", data=df, ci = None)
```

```
<matplotlib.axes._subplots.AxesSubplot at
0x7f30e06cc350>
```



Kumar Utkarsh
Resolve
Dec 23, 2021

regplot() helps to plot data and a linear regression model fit. Regplot in the seaborn library to determine if the feature sqft_above is negatively or positively correlated with price

```
X1 = df[['sqft_living']]
Y1 = df['price']
lm = LinearRegression()
lm
lm.fit(X1,Y1)
lm.score(X1, Y1)
```

```
0.4928532179037931
```

Kumar Utkarsh
Resolve
Dec 23, 2021

Fit a linear regression model to predict the price using the feature 'sqft_living'

```
features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view" ,"bathrooms",
X = df[features]
Y = df['price']
lm.fit(X,Y)
lm.score(X,Y)
```

```
0.6577027577865877
```

Kumar Utkarsh
Resolve
Dec 23, 2021

Fit a linear regression model to predict the ‘price’ using the list of features

```
Input=[('scale',StandardScaler()),('polynomial', PolynomialFeatures(include_bias=False)),(
pipe=Pipeline(Input)
pipe
pipe.fit(X,Y)
pipe.score(X,Y)
features =["floors", "waterfront","lat" ,"bedrooms"                                 ",
X = df[features ]
Y = df['price']
```

Kumar Utkarsh
Resolve
Dec 23, 2021

list to create a pipeline object, predict the ‘price’, fit the object using the features in the list features, then fit the model and calculate the R^2

```
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
```

```
RigeModel = Ridge(alpha=0.1)
RigeModel.fit(x_train, y_train)
RigeModel.score(x_test, y_test)
```

```
0.6480374087702245
```

```
pr=PolynomialFeatures(degree=2)
x_train_pr=pr.fit_transform(x_train[features])
x_test_pr=pr.fit_transform(x_test[features])

RigeModel = Ridge(alpha=0.1)
RigeModel.fit(x_train_pr, y_train)
RigeModel.score(x_test_pr, y_test)
```

```
0.7004432058878023
```

Kumar Utkarsh
Dec 23, 2021
Resolve

Ridge regression model is used to develop the best linear regression model that are fit and not causing overfitting and underfitting by estimating the value of alpha.