```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df=pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Datasets/kc_house_data.csv')
df
```

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080
...	...	...	...	...	...	...	..
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076

21613 rows × 21 columns

```
df.shape
```

```
(21613, 21)
```

```
df.isna().sum()
```

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

```
df.isnull().any()
```

```
id               False
date             False
price            False
bedrooms         False
bathrooms        False
sqft_living      False
sqft_lot         False
floors           False
waterfront       False
view             False
condition        False
grade            False
sqft_above       False
sqft_basement    False
yr_built         False
```

```
yr_renovated     False
zipcode          False
lat              False
long             False
sqft_living15    False
sqft_lot15       False
dtype: bool
```

```
df.dtypes
```

```
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','date'],axis=1,inplace=True)
df
```

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	vi
0	221900.0	3	1.00	1180	5650	1.0	0	
1	538000.0	3	2.25	2570	7242	2.0	0	
2	180000.0	2	1.00	770	10000	1.0	0	
3	604000.0	4	3.00	1960	5000	1.0	0	
4	510000.0	3	2.00	1680	8080	1.0	0	
...	...	...	...	...	...	...	...	...
21608	360000.0	3	2.50	1530	1131	3.0	0	
21609	400000.0	4	2.50	2310	5813	2.0	0	
21610	402101.0	2	0.75	1020	1350	2.0	0	
21611	400000.0	3	2.50	1600	2388	2.0	0	
21612	325000.0	2	0.75	1020	1076	2.0	0	

21613 rows × 19 columns

```
x=df.iloc[:,1:].values
y=df["price"].values
x
```

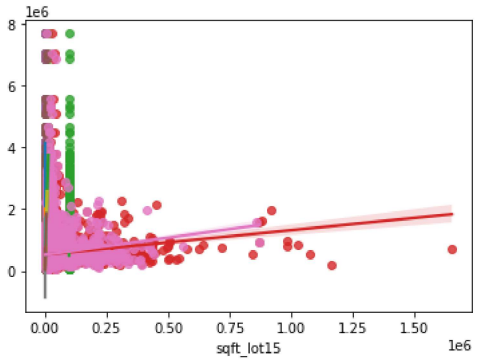
```
array([[ 3.00000e+00,  1.00000e+00,  1.18000e+03, ..., -1.22257e+02,
         1.34000e+03,  5.65000e+03],
       [ 3.00000e+00,  2.25000e+00,  2.57000e+03, ..., -1.22319e+02,
         1.69000e+03,  7.63900e+03],
       [ 2.00000e+00,  1.00000e+00,  7.70000e+02, ..., -1.22233e+02,
         2.72000e+03,  8.06200e+03],
       ...,
       [ 2.00000e+00,  7.50000e-01,  1.02000e+03, ..., -1.22299e+02,
         1.02000e+03,  2.00700e+03],
       [ 3.00000e+00,  2.50000e+00,  1.60000e+03, ..., -1.22069e+02,
         1.41000e+03,  1.28700e+03],
       [ 2.00000e+00,  7.50000e-01,  1.02000e+03, ..., -1.22299e+02,
         1.02000e+03,  1.35700e+03]])
```

```
y
```

```
array([221900., 538000., 180000., ..., 402101., 400000., 325000.])
```

```
for i in df.columns:
  if i=="price":
    continue
  else:
    print(i)
    sns.regplot(x=df'')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb9f1160610>
```




```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=1)
```

```
# from sklearn.preprocessing import StandardScaler
# scaler=StandardScaler()
# scaler.fit(x_train)
# x_train=scaler.fit_transform(x_train)
# x_test=scaler.fit_transform(x_test)
# x_train
```

```
#model creation
from sklearn.linear_model import LinearRegression
mlr=LinearRegression()
mlr.fit(x_train,y_train)
y_pred=mlr.predict(x_test)
y_pred
```

```
array([ 640058.17791108,  476677.53037962,  707658.73936309, ...,
        360521.61328911, 1387672.15189813,  367938.08050214])
```

```
print('Intersept',mlr.intercept_)
print('slope',mlr.coef_)
```

```
Intersept 3956635.475128025
slope [-3.20279315e+04  3.60409149e+04  1.06391432e+02  1.32288285e-01
  1.01348768e+04  5.38860457e+05  5.19958026e+04  2.76632242e+04
  9.55651532e+04  6.86621576e+01  3.77292747e+01 -2.51780015e+03
  2.10768444e+01 -5.49685041e+02  6.10241682e+05 -2.06289119e+05
  2.11939640e+01 -3.47658777e-01]
```

```
#Mean Absolute Error
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error
print("Mean Absolute Error is ",mean_absolute_error(y_test,y_pred))
print("Error Percentage ",mean_absolute_percentage_error(y_test,y_pred))
```

```
Mean Absolute Error is  129451.37388156046
Error Percentage  0.2530088490531335
```

```
#Mean Squared Error
from sklearn.metrics import mean_squared_error
```

```
print("MSE",mean_squared_error(y_test,y_pred))
```

```
MSE 54219291122.91789
```

```
#Root Mean Squared Error
print("Root Mean Squared Error",np.sqrt(mean_squared_error(y_test,y_pred)))
```

```
Root Mean Squared Error 232850.36208457546
```

```
# df['date']=le.fit_transform(df['date'])
```