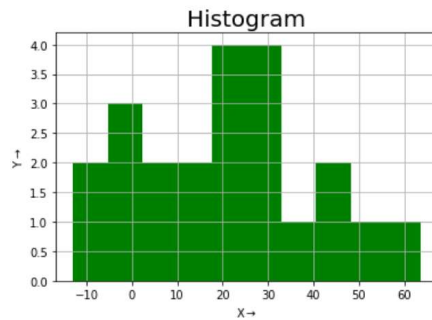


```
In [70]: plt.hist(np.random.normal(23,19,22), color='green')
plt.title("Histogram",fontsize='20')
plt.xlabel("X"+r"$\rightarrow$")
plt.ylabel("Y"+r"$\rightarrow$")
plt.grid()
```



## DISCUSSION and CONCLUSION

The various types of plots/charts like histograms, plot based on sine/cosine function based on data from a matrix, have been studied and created in the Jupyter platform. Further, labelling of different axes in a plot and data in a plot, have been performed.

<b>CRITERIA</b>	<b>TOTAL MARKS</b>	<b>MARKS OBTAINED</b>	<b>COMMENTS</b>
Concept (A)	2		
Implementation (B)	2		
Performance (C)	2		
Total	6		

# EXPERIMENT- 4

## AIM

To implement linear regression model on housing data.

## SOFTWARE USED

Jupyter Platform - Python Programming Language

## PROGRAM CODE

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

data = pd.read_csv("kc_house_data.csv")

data.head()
data.describe()

data['bedrooms'].value_counts().plot(kind='bar')
plt.title('number of Bedroom')
plt.xlabel('Bedrooms')
plt.ylabel('Count')
sns.despine

plt.figure(figsize=(10,10))
sns.jointplot(x=data.lat.values, y=data.long.values, height=10)
plt.ylabel('Longitude', fontsize=12)
plt.xlabel('Latitude', fontsize=12)
plt.show()
sns.despine

plt.scatter(data.price, data.sqft_living)
plt.title("Price vs Square Feet")

plt.scatter(data.price, data.long)
plt.title("Price vs Location of the area")

plt.scatter(data.price, data.lat)
plt.xlabel("Price")
plt.ylabel('Latitude')
plt.title("Latitude vs Price")
```

```

plt.scatter(data.bedrooms,data.price)
plt.title("Bedroom and Price ")
plt.xlabel("Bedrooms")
plt.ylabel("Price")
plt.show()
sns.despine

plt.scatter([(data['sqft_living']+data['sqft_basement']),data['price']])

plt.scatter(data.waterfront,data.price)
plt.title("Waterfront vs Price ( 0= no waterfront)")

train1 = data.drop(['id', 'price'],axis=1)

train1.head()

data.floors.value_counts().plot(kind='bar')

plt.scatter(data.floors,data.price)

plt.scatter(data.condition,data.price)

plt.scatter(data.zipcode,data.price)
plt.title("Which is the pricey location by zipcode?")


from sklearn.linear_model import LinearRegression

reg = LinearRegression()

labels = data['price']
conv_dates = [1 if values == 2014 else 0 for values in data.date ]
data['date'] = conv_dates
train1 = data.drop(['id', 'price'],axis=1)

from sklearn.model_selection import train_test_split

x_train , x_test , y_train , y_test = train_test_split(train1 , labels , test_size = 0.10,random_state =2)

reg.fit(x_train,y_train)

reg.score(x_test,y_test)

```

# OUTPUT

jupyter housesales (2) (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

```
In [35]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mpl_toolkits
%matplotlib inline
```

```
In [36]: data = pd.read_csv("kc_house_data.csv")
```

```
In [37]: data.head()
```

Out[37]:

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

5 rows × 21 columns

```
In [38]: data.describe()
```

Out[38]:

	id	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	g
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000	21613.00
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430	7.65
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743	1.17
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000	1.00
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000	7.00
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000	7.00
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000	8.00
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000	13.00

jupyter housesales (2) (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

```
In [38]: data.describe()
```

Out[38]:

	id	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	g
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000	21613.00
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430	7.65
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743	1.17
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000	1.00
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000	7.00
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000	7.00
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000	8.00
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000	13.00

```
In [39]: data['bedrooms'].value_counts().plot(kind='bar')
plt.title('number of Bedroom')
plt.xlabel('Bedrooms')
plt.ylabel('Count')
sns.despine
```

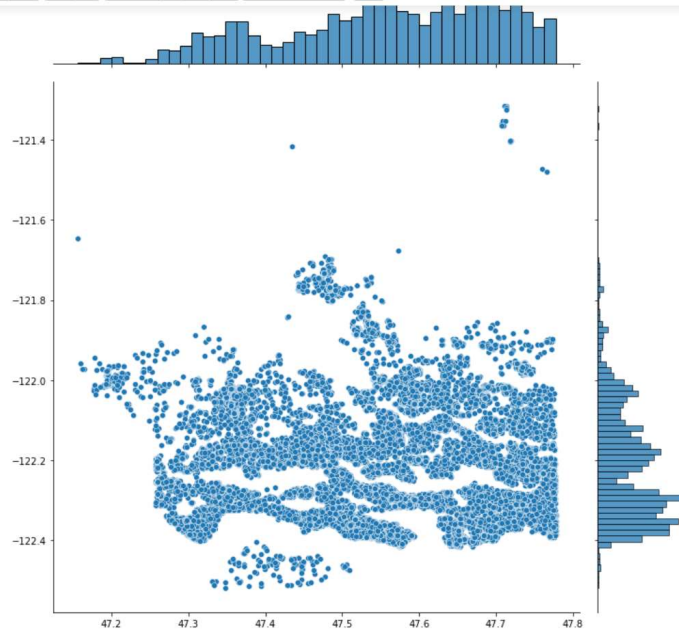
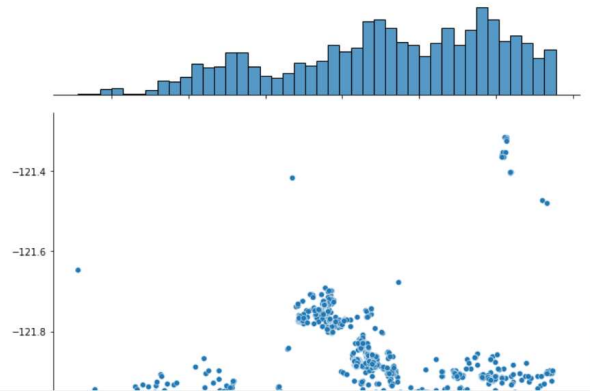
Out[39]: <function seaborn.utils.despine(fig=None, ax=None, top=True, right=True, left=False, bottom=False, offset=None, trim=False)>

Bedrooms	Count
1	10000
2	7000
3	2500
4	1500
5	1000



```
In [40]: plt.figure(figsize=(10,10))
sns.jointplot(x=data.lat.values, y=data.long.values, height=10)
plt.ylabel('Longitude', fontsize=12)
plt.xlabel('Latitude', fontsize=12)
plt.show()
sns.despine
```

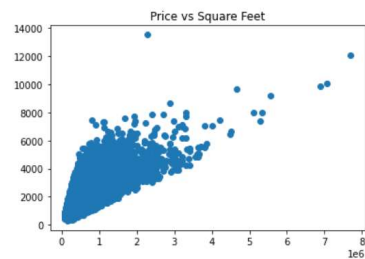
<Figure size 720x720 with 0 Axes>



```
Out[40]: <function seaborn.utils.despine(fig=None, ax=None, top=True, right=True, left=False, bottom=False, offset=None, trim=False)>
```

```
In [41]: plt.scatter(data.price, data.sqft_living)
plt.title("Price vs Square Feet")
```

```
Out[41]: Text(0.5, 1.0, 'Price vs Square Feet')
```



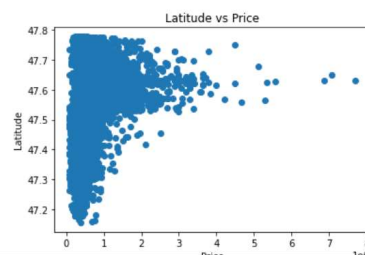
```
In [42]: plt.scatter(data.price, data.long)
plt.title("Price vs Location of the area")
```

```
Out[42]: Text(0.5, 1.0, 'Price vs Location of the area')
```



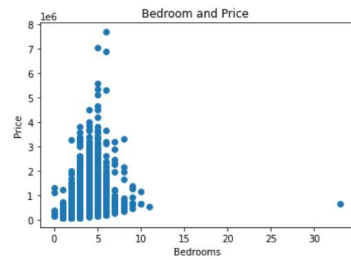
```
In [43]: plt.scatter(data.price, data.lat)
plt.xlabel("Price")
plt.ylabel("Latitude")
plt.title("Latitude vs Price")
```

```
Out[43]: Text(0.5, 1.0, 'Latitude vs Price')
```





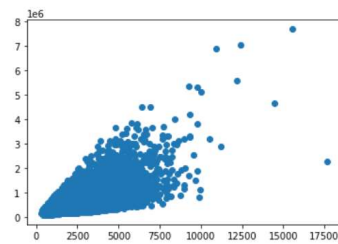
```
In [44]: plt.scatter(data.bedrooms,data.price)
plt.title("Bedroom and Price ")
plt.xlabel("Bedrooms")
plt.ylabel("Price")
plt.show()
sns.despine
```



```
Out[44]: <function seaborn.utils.despine(fig=None, ax=None, top=True, right=True, left=False, bottom=False, offset=None, trim=False)>
```

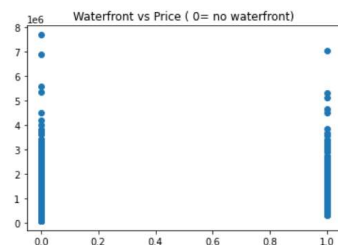
```
In [45]: plt.scatter((data['sqft_living']+data['sqft_basement']),data['price'])
```

```
Out[45]: <matplotlib.collections.PathCollection at 0x7fcb9c76e90>
```



```
In [46]: plt.scatter(data.waterfront,data.price)
plt.title("Waterfront vs Price ( 0= no waterfront)")
```

```
Out[46]: Text(0.5, 1.0, 'Waterfront vs Price ( 0= no waterfront)')
```





```
In [47]: train1 = data.drop(['id', 'price'],axis=1)
```

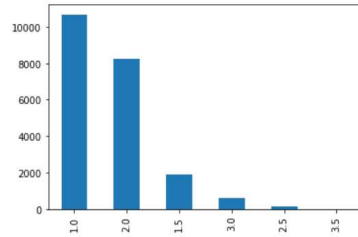
```
In [48]: train1.head()
```

```
Out[48]:
```

	date	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated
0	20141013T000000	3	1.00	1180	5650	1.0	0	0	3	7	1180	0	1955	0
1	20141209T000000	3	2.25	2570	7242	2.0	0	0	3	7	2170	400	1951	1991
2	20150225T000000	2	1.00	770	10000	1.0	0	0	3	6	770	0	1933	0
3	20141209T000000	4	3.00	1960	5000	1.0	0	0	5	7	1050	910	1965	0
4	20150218T000000	3	2.00	1680	8080	1.0	0	0	3	8	1680	0	1987	0

```
In [49]: data.floors.value_counts().plot(kind='bar')
```

```
Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x7fcb9ae6a50>
```

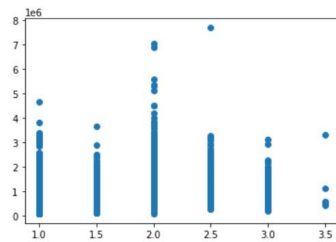


```
In [50]: plt.scatter(data.floors,data.price)
```

```
Out[50]: <matplotlib.collections.PathCollection at 0x7fcb9ae6a50>
```

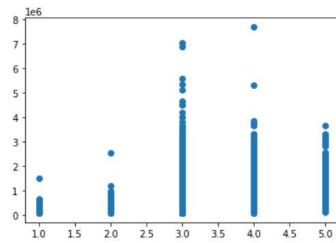
```
In [50]: plt.scatter(data.floors,data.price)
```

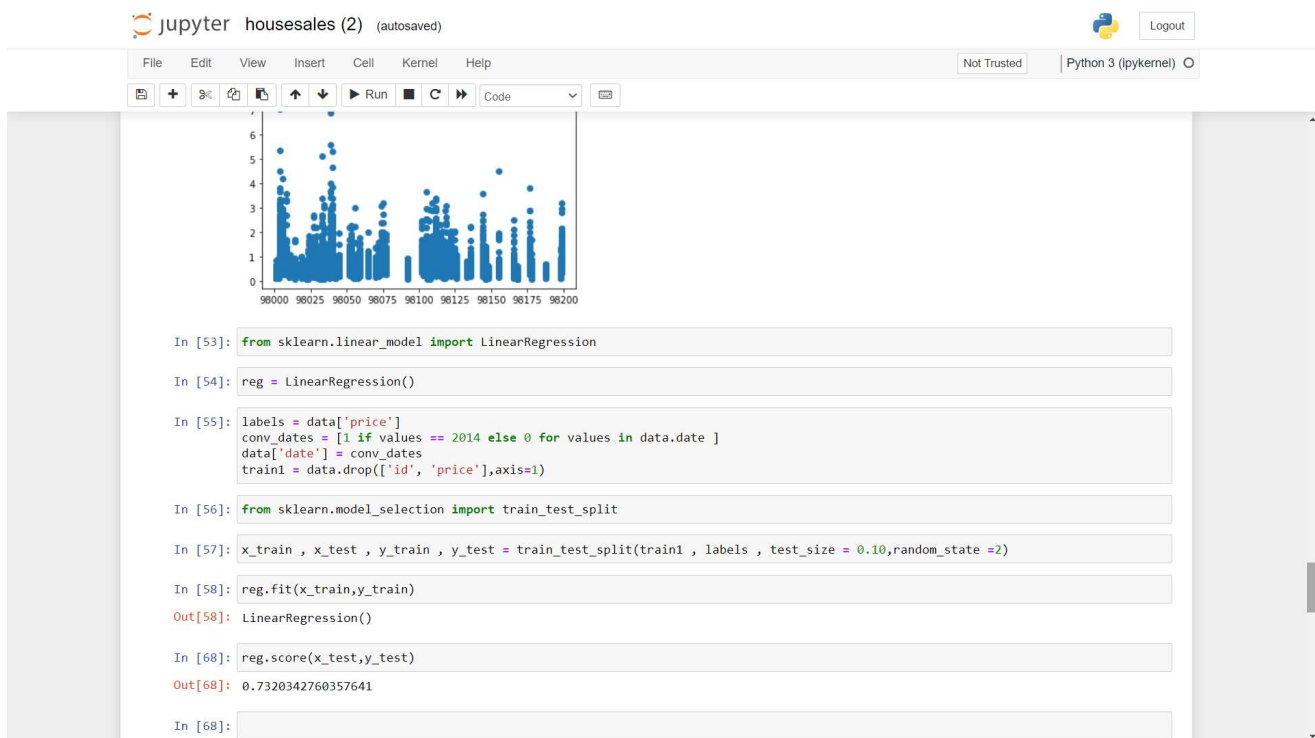
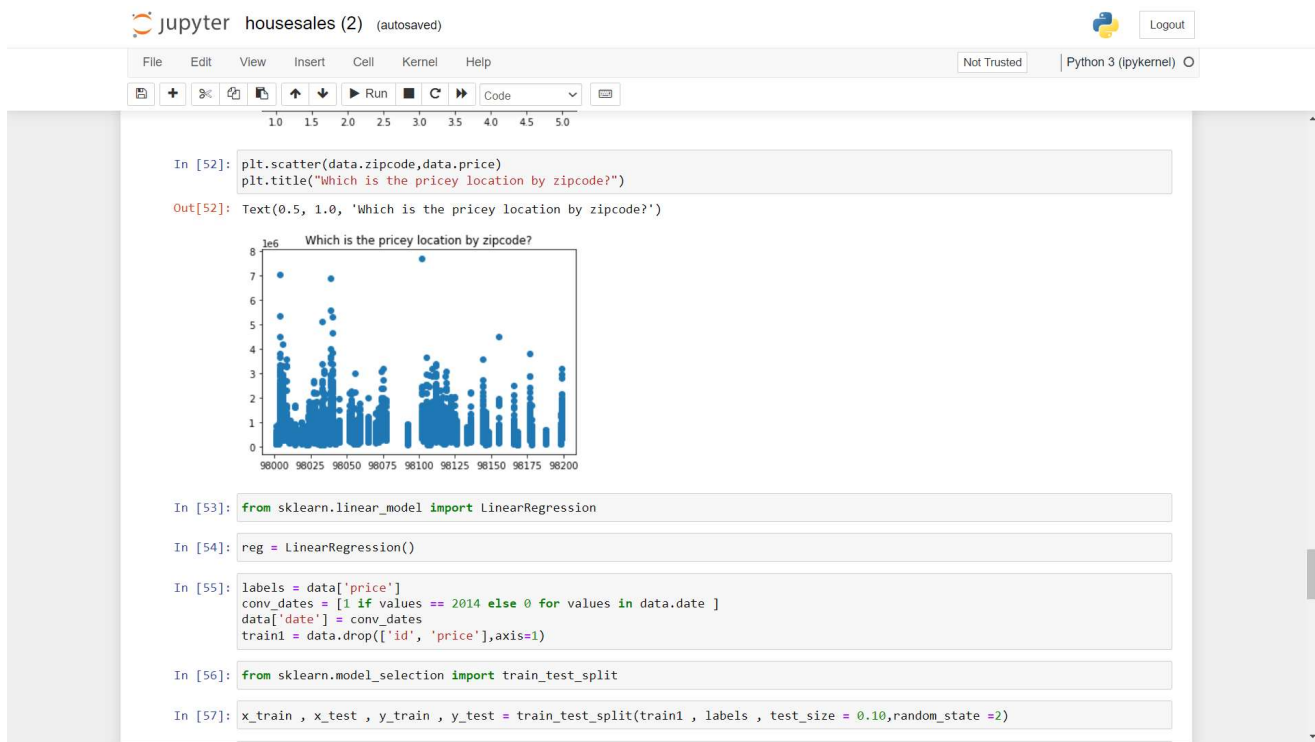
```
Out[50]: <matplotlib.collections.PathCollection at 0x7fcb9a185d0>
```



```
In [51]: plt.scatter(data.condition,data.price)
```

```
Out[51]: <matplotlib.collections.PathCollection at 0x7fcb9b76a90>
```





## DISCUSSION and CONCLUSION

The linear regression model has been applied and executed successfully on housing data.

<b>CRITERIA</b>	<b>TOTAL MARKS</b>	<b>MARKS OBTAINED</b>	<b>COMMENTS</b>
Concept (A)	2		
Implementation (B)	2		
Performance (C)	2		
Total	6		