Assignment-3

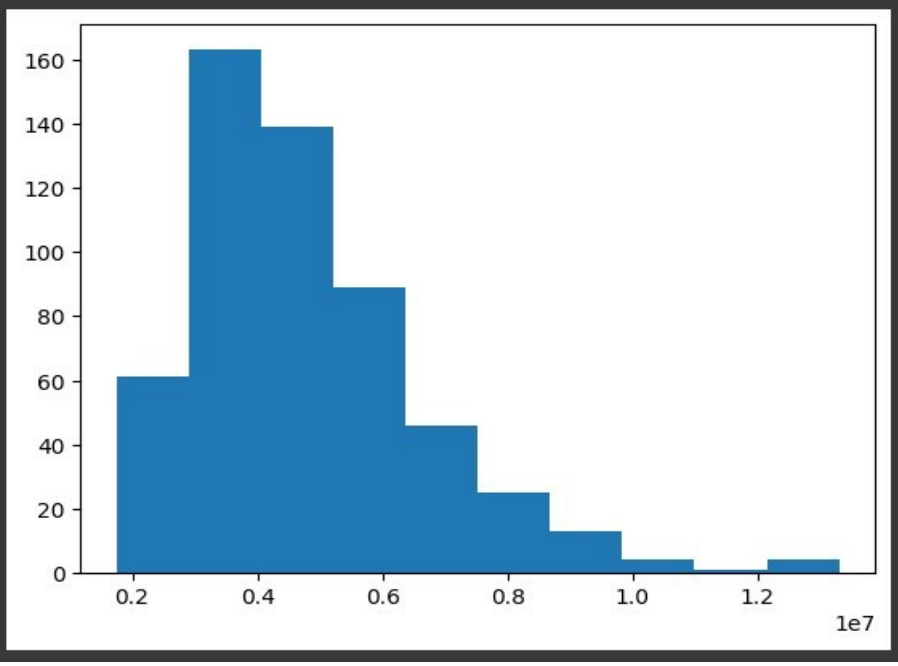
Gangisetti sathwik

20BCE7644

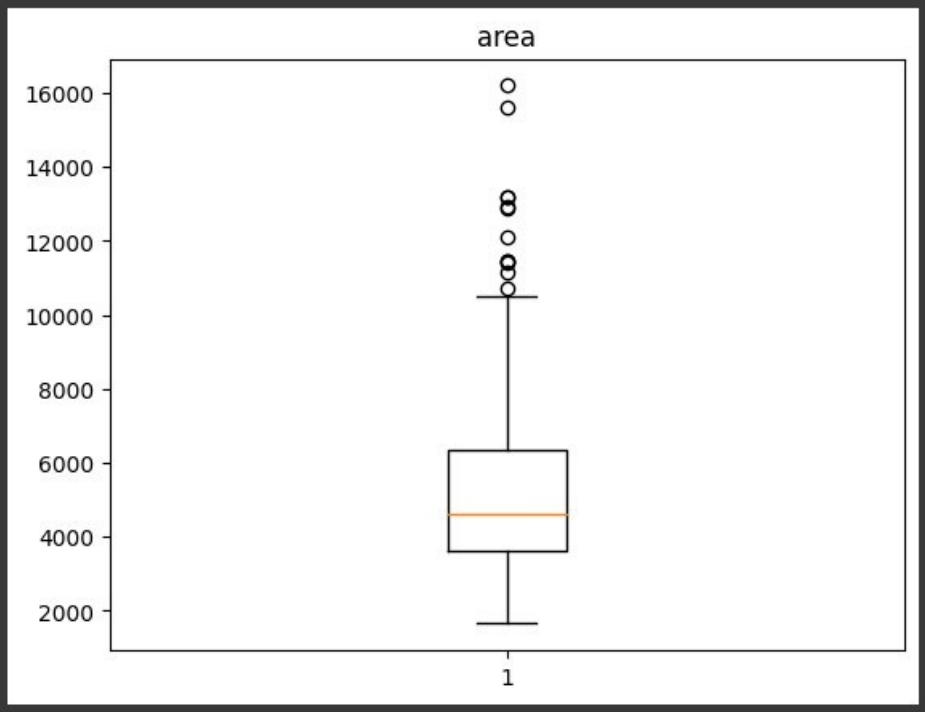
```
[ ]  import pandas as pd
     import numpy as np
     import seaborn as sns
     import matplotlib.pyplot as plt
```

```
[ ]  data = pd.read_csv('Housing.csv')
```

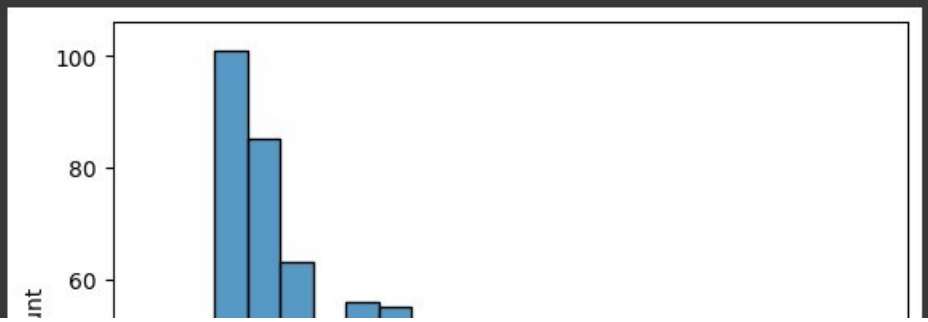
```
#Univariate Analysis
plt.hist(data.price)
plt.show()
```



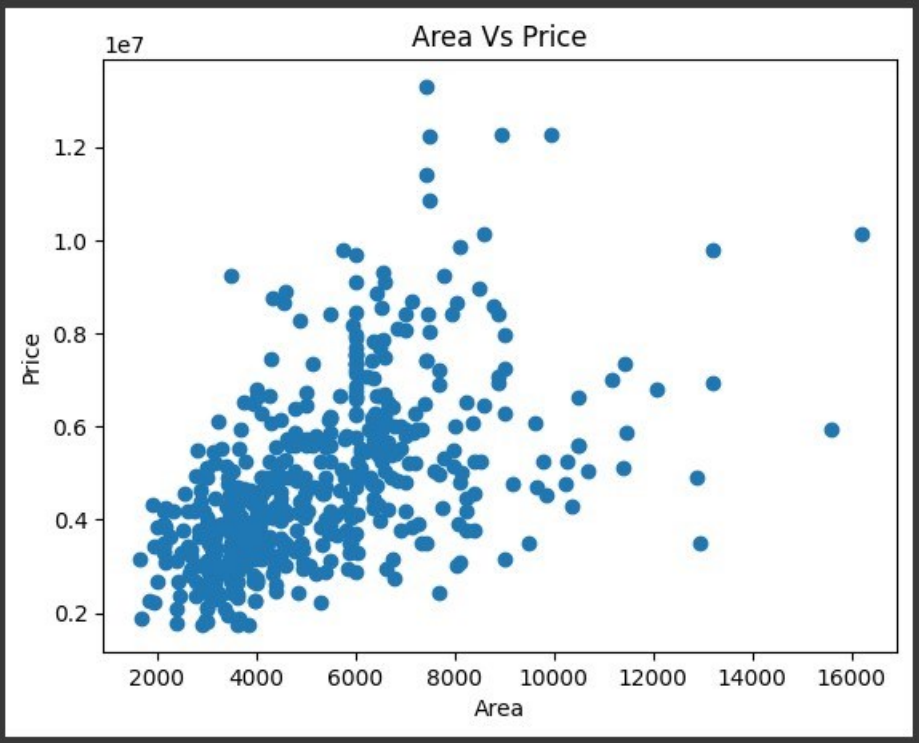
```
[ ]  plt.title("area")
     plt.boxplot(data.area)
     plt.show()
```




```
     sns.histplot(data.area)
     plt.show()
```



```
#Bivariate analysis
plt.scatter(data.area,data.price)
plt.xlabel('Area')
plt.ylabel('Price')
plt.title('Area Vs Price')
plt.show()
```




```
[ ]  #Multi Variate Analysis
     data.corr()
```

```
<ipython-input-7-384e2f4675a9>:2: FutureWarning: The default value of numeric_only
  data.corr()
```

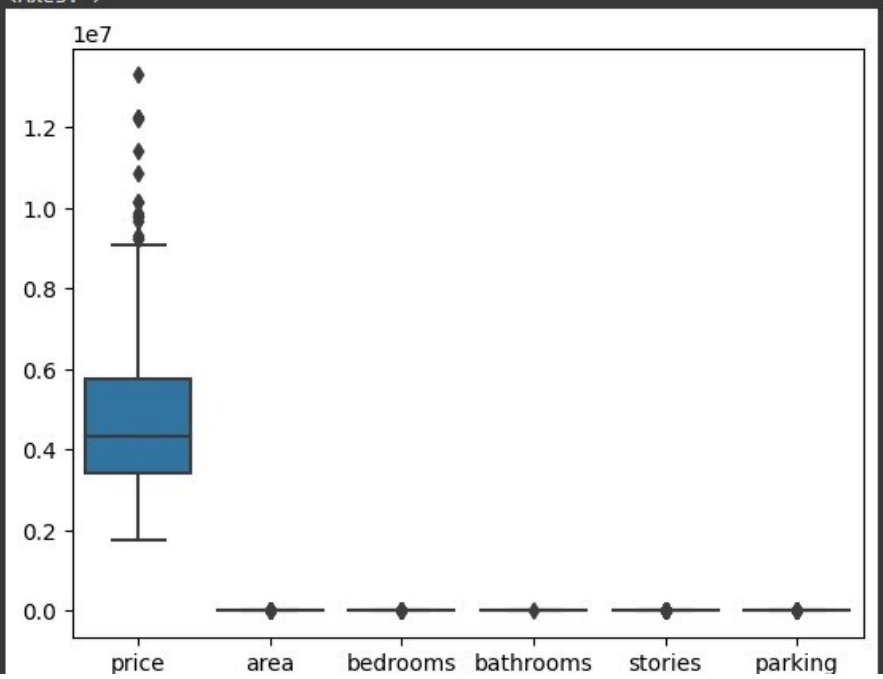
	price	area	bedrooms	bathrooms	stories	parking
price	1.000000	0.535997	0.366494	0.517545	0.420712	0.384394

```
[ ]
```

	price	area	bedrooms	bathrooms	stories	parking
price	1.000000	0.535997	0.366494	0.517545	0.420712	0.384394
area	0.535997	1.000000	0.151858	0.193820	0.083996	0.352980
bedrooms	0.366494	0.151858	1.000000	0.373930	0.408564	0.139270
bathrooms	0.517545	0.193820	0.373930	1.000000	0.326165	0.177496
stories	0.420712	0.083996	0.408564	0.326165	1.000000	0.045547
parking	0.384394	0.352980	0.139270	0.177496	0.045547	1.000000

```
sns.boxplot(data)
```

```
<Axes: >
```

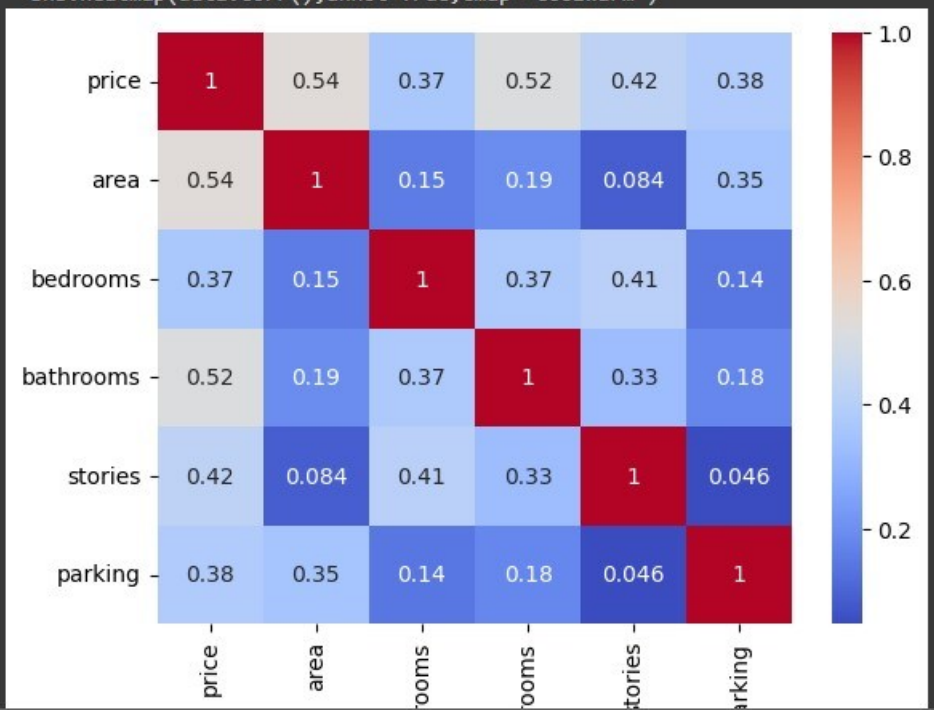


```
[ ] data.mean()
```

```
<ipython-input-9-abc01cf6c622>:1: FutureWarning: The default value of numeric
  data.mean()
price        4.766729e+06
area         5.150541e+03
bedrooms     2.965138e+00
bathrooms    1.286239e+00
stories      1.805505e+00
parking      6.935780e-01
dtype: float64
```

```
#Heat Map
sns.heatmap(data.corr(),annot=True,cmap='coolwarm')
plt.show()
```

```
<ipython-input-10-a60deadf2795>:2: FutureWarning: The default value of numeri
  sns.heatmap(data.corr(),annot=True,cmap='coolwarm')
```



```
[ ] #Descriptive Analysis
    data.head()
```

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	furnished

```
[ ] data.isna().sum()
```

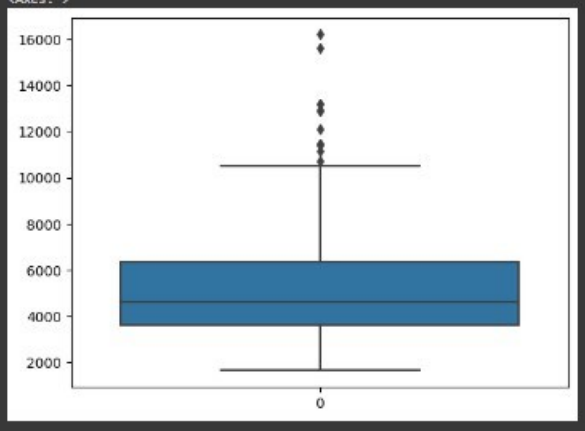
```
price             0
area              0
bedrooms          0
bathrooms         0
stories           0
mainroad          0
guestroom         0
basement          0
hotwaterheating   0
airconditioning   0
parking           0
furnishingstatus  0
dtype: int64
```

```
data.mean()
```

```
<ipython-input-13-abc01cf6c622>:1: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future version, it w
  data.mean()
price        4.766729e+06
area         5.150541e+03
bedrooms     2.965138e+00
bathrooms    1.286239e+00
stories      1.805505e+00
parking      6.935780e-01
dtype: float64
```

```
[ ] sns.boxplot(data.area)
```

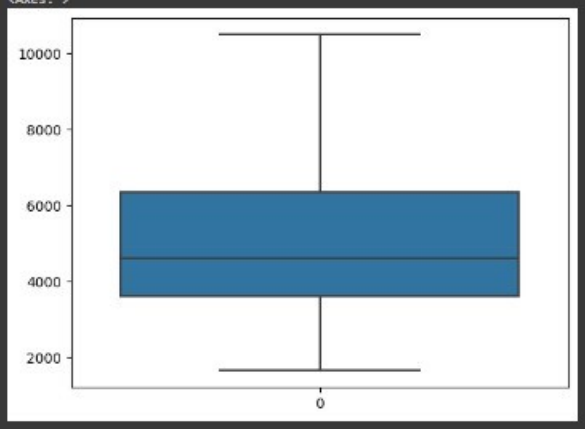
```
<Axes: >
```




```
[ ] #Removing the outliners
    q1 = data['area'].quantile(0.25)
    q3 = data['area'].quantile(0.75)

    iqr = q3 - q1

    up = q3 + (1.5 * iqr)
    low = q1 - (1.5 * iqr)

    data['area'] = np.where(data['area'] > up, up,
            np.where(data['area'] < low, low, data['area']))
```

```
sns.boxplot(data.area)
```

```
<Axes: >
```



```
[ ] data.head()
```

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus
0	13300000	7420.0	4	2	3	yes	no	no	no	yes	2	furnished
1	12250000	8960.0	4	4	4	yes	no	no	no	yes	3	furnished
2	12250000	9960.0	3	2	2	yes	no	yes	no	no	2	semi-furnished
3	12215000	7500.0	4	2	2	yes	no	yes	no	yes	3	furnished
4	11410000	7420.0	4	1	2	yes	yes	yes	no	yes	2	furnished

```
[ ] #Check for Categorical columns and perform encoding
    from sklearn.preprocessing import LabelEncoder
    le = LabelEncoder()
```

```
[ ] data.mainroad = le.fit_transform(data.mainroad)
    data.guestroom = le.fit_transform(data.guestroom)
    data.basement = le.fit_transform(data.basement)
    data.hotwaterheating = le.fit_transform(data.hotwaterheating)
    data.airconditioning = le.fit_transform(data.airconditioning)
```

```
[ ] data = pd.get_dummies(data,columns=['furnishingstatus'])
```

```
[ ] data.head()
```

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus_furnished	furnishingstatus_semi-furnished	furnishingstatus_unfurnished
0	13300000	7420.0	4	2	3	1	0	0	0	1	2	1	0	0
1	12250000	8960.0	4	4	4	1	0	0	0	1	3	1	0	0
2	12250000	9960.0	3	2	2	1	0	1	0	0	2	0	1	0
3	12215000	7500.0	4	2	2	1	0	1	0	1	3	1	0	0
4	11410000	7420.0	4	1	2	1	1	1	0	1	2	1	0	0

```
#Splitting dataset into x and y
x = data.drop('price',axis=1)
y = data['price']
```

```
x
```

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus_furnished	furnishingstatus_semi-furnished	furnishingstatus_unfurnished
0	7420.0	4	2	3	1	0	0	0	1	2	1	0	0
1	8960.0	4	4	4	1	0	0	0	1	3	1	0	0
2	9960.0	3	2	2	1	0	1	0	0	2	0	1	0
3	7500.0	4	2	2	1	0	1	0	1	3	1	0	0
4	7420.0	4	1	2	1	1	1	0	1	2	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
540	3000.0	2	1	1	1	0	1	0	0	2	0	0	1
541	2400.0	3	1	1	0	0	0	0	0	0	0	1	0
542	3620.0	2	1	1	1	0	0	0	0	0	0	0	1
543	2910.0	3	1	1	0	0	0	0	0	0	1	0	0
544	3850.0	3	1	2	1	0	0	0	0	0	0	0	1

545 rows × 13 columns

```
y
```

```
0      13300000
1      12250000
2      12250000
3      12215000
4      11410000
         ...
540     1820000
541     1767150
542     1750000
543     1750000
544     1750000
Name: price, Length: 545, dtype: int64
```

+ Code + Text

```
#Scaling the independent variables
name = x.columns
```

```
name
```

```
Index(['area', 'bedrooms', 'bathrooms', 'stories', 'mainroad', 'guestroom',
       'basement', 'hotwaterheating', 'airconditioning', 'parking',
       'furnishingstatus_furnished', 'furnishingstatus_semi-furnished',
       'furnishingstatus_unfurnished'],
      dtype='object')
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
scale = MinMaxScaler()
```

```
x = scale.fit_transform(x)
```

```
x = pd.DataFrame(x,columns=name)
```

```
x
```

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus_furnished	furnishingstatus_semi-furnished	furnishingstatus_unfurnished
0	0.651977	0.6	0.333333	0.666667	1.0	0.0	0.0	0.0	1.0	0.666667	1.0	0.0	0.0
1	0.825989	0.6	1.000000	1.000000	1.0	0.0	0.0	0.0	1.0	1.000000	1.0	0.0	0.0
2	0.938983	0.4	0.333333	0.333333	1.0	0.0	1.0	0.0	0.0	0.666667	0.0	1.0	0.0
3	0.661017	0.6	0.333333	0.333333	1.0	0.0	1.0	0.0	1.0	1.000000	1.0	0.0	0.0
4	0.651977	0.6	0.000000	0.333333	1.0	1.0	1.0	0.0	1.0	0.666667	1.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
540	0.152542	0.2	0.000000	0.000000	1.0	0.0	1.0	0.0	0.0	0.666667	0.0	0.0	1.0
541	0.084746	0.4	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	1.0	0.0
542	0.222599	0.2	0.000000	0.000000	1.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	1.0
543	0.142373	0.4	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.000000	1.0	0.0	0.0
544	0.248588	0.4	0.000000	0.333333	1.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	1.0

545 rows × 13 columns

```
#train test split
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

```
[ ] x_train.head()
```

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	furnishingstatus_furnished	furnishingstatus_semi-furnished	furnishingstatus_unfurnish
542	0.222509	0.2	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
496	0.285537	0.2	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
484	0.157062	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
507	0.220339	0.2	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
252	0.927684	0.4	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	

```
[ ] y_train.head()
```

```
542    1750000
496    2695000
484    2870000
507    2590000
252    4515000
Name: price, dtype: int64
```

```
[ ] #Building the model
    from sklearn.linear_model import LinearRegression

[ ] model = LinearRegression()

[ ] #Training the model
    model.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
    pred = model.predict(x_test)
    pred = pd.DataFrame(pred)
    pred
```

	0
0	4.046159e+06
1	6.000168e+06
2	4.565435e+06
3	7.074956e+06
4	2.860086e+06
...	...
104	6.924886e+06
105	5.407671e+06
106	4.427672e+06
107	5.068052e+06
108	5.907373e+06

109 rows × 1 columns

```
[ ] y_test = pd.DataFrame(y_test)
```