Experiment 5

Aim: Implementation of Multiple Linear Regression and Regularization.

Objectives: To learn about Multiple Linear Regression techniques. To learn about prediction using Multiple Linear Regression To apply regularization on model.

Course Outcomes CO3, CO5

In [2]:

```
import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
```

Read the data (Excel file "Exp-5 House Price.csv").

In [3]:

```
df=pd.read_csv('Exp-5 House Price.csv')
```

In [4]:

```
df
```

Out[4]:

	Area	BHK	Bathroom	Furnishing	Locality	Parking	Price	Status	Transaction	Type	Per_Sqft
0	800.0	3	2.0	Semi-Furnished	Rohini Sector 25	1.0	6500000	Ready_to_move	New_Property	Builder_Floor	NaN
1	750.0	2	2.0	Semi-Furnished	J R Designers Floors, Rohini Sector 24	1.0	5000000	Ready_to_move	New_Property	Apartment	6667.0
2	950.0	2	2.0	Furnished	Citizen Apartment, Rohini Sector 13	1.0	15500000	Ready_to_move	Resale	Apartment	6667.0
3	600.0	2	2.0	Semi-Furnished	Rohini Sector 24	1.0	4200000	Ready_to_move	Resale	Builder_Floor	6667.0
4	650.0	2	2.0	Semi-Furnished	Rohini Sector 24 carpet area 650 sqft status R...	1.0	6200000	Ready_to_move	New_Property	Builder_Floor	6667.0
...	...	...	...	...	...	...	...	...	...	...	...
1254	4118.0	4	5.0	Unfurnished	Chittaranjan Park	3.0	55000000	Ready_to_move	New_Property	Builder_Floor	12916.0
1255	1050.0	3	2.0	Semi-Furnished	Chittaranjan Park	3.0	12500000	Ready_to_move	Resale	Builder_Floor	12916.0
1256	875.0	3	3.0	Semi-Furnished	Chittaranjan Park	3.0	17500000	Ready_to_move	New_Property	Builder_Floor	12916.0
1257	990.0	2	2.0	Unfurnished	Chittaranjan Park Block A	1.0	11500000	Ready_to_move	Resale	Builder_Floor	12916.0
1258	11050.0	3	3.0	Unfurnished	Chittaranjan Park	1.0	18500000	Ready_to_move	New_Property	Builder_Floor	12916.0

1259 rows × 11 columns

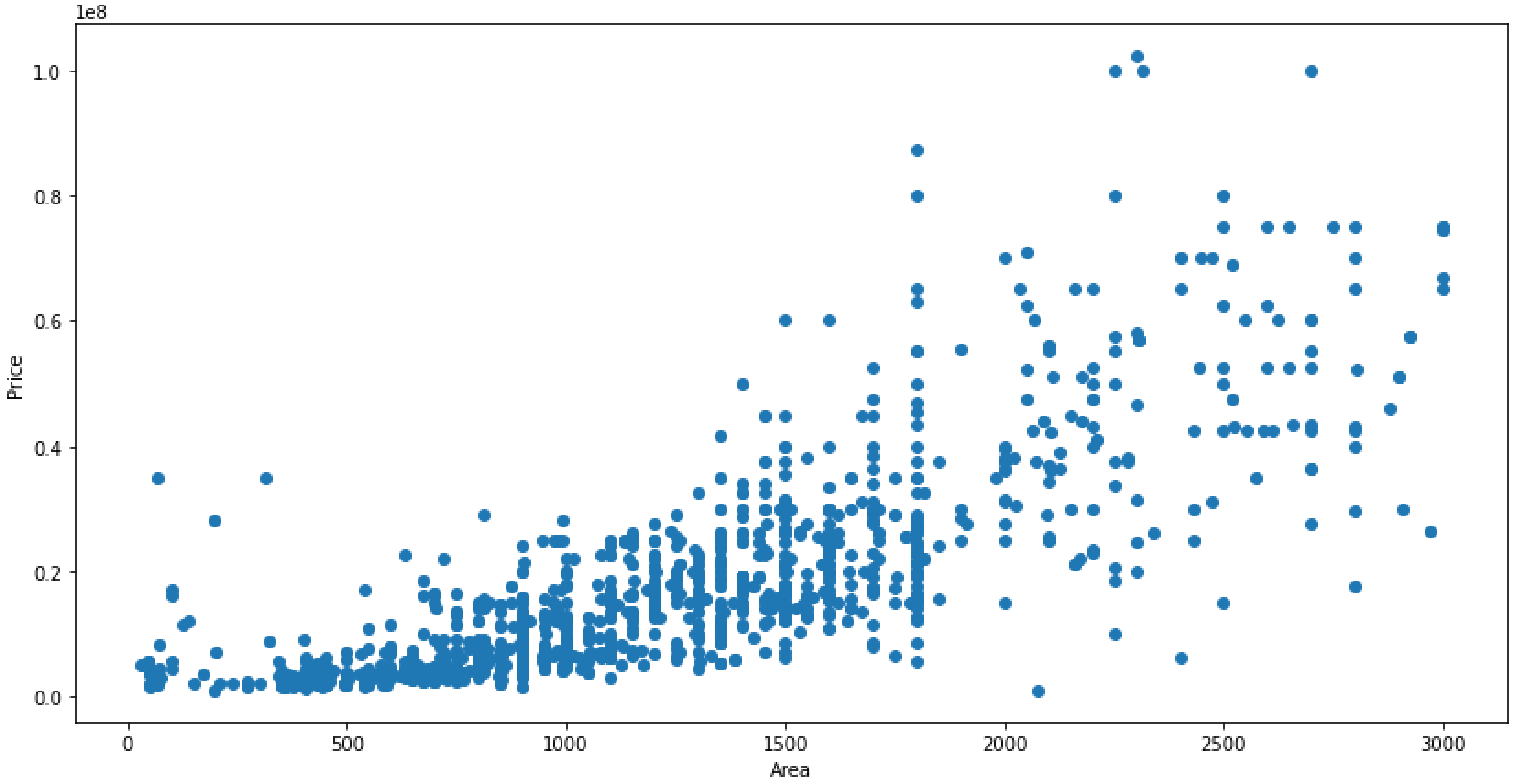
Droping the values of Area which are greater than 3000 and plotting scatter plot between Area and Price.

In [8]:

```
df.drop(df[df['Area'] > 3000].index, inplace = True)
plt.figure(figsize=(14,7))
plt.xlabel('Area')
plt.ylabel('Price')
plt.scatter(df.Area,df.Price)
```

Out[8]:

```
<matplotlib.collections.PathCollection at 0x24b0f999438>
```



In [5]:
```
df.corr()
```

Out[5]:

	Area	BHK	Bathroom	Parking	Price	Per_Sqft
Area	1.000000	0.449438	0.535104	-0.009297	0.580836	0.162832
BHK	0.449438	1.000000	0.773267	-0.070707	0.571523	0.181540
Bathroom	0.535104	0.773267	1.000000	-0.032796	0.728108	0.219169
Parking	-0.009297	-0.070707	-0.032796	1.000000	-0.000448	0.001607
Price	0.580836	0.571523	0.728108	-0.000448	1.000000	0.322859
Per_Sqft	0.162832	0.181540	0.219169	0.001607	0.322859	1.000000

Importing the encoder and encoding the required fields.

In [6]:
```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
```

In [7]:
```
df['Furnishing'] = label_encoder.fit_transform(df['Furnishing'])
df['Locality'] = label_encoder.fit_transform(df['Locality'])
df['Status'] = label_encoder.fit_transform(df['Status'])
df['Transaction'] = label_encoder.fit_transform(df['Transaction'])
df['Type'] = label_encoder.fit_transform(df['Type'])
```

In [8]:
```
df
```

Out[8]:

	Area	BHK	Bathroom	Furnishing	Locality	Parking	Price	Status	Transaction	Type	Per_Sqft
0	800.0	3	2.0	1	283	1.0	6500000	1	0	1	NaN
1	750.0	2	2.0	1	139	1.0	5000000	1	0	0	6667.0
2	950.0	2	2.0	0	49	1.0	15500000	1	1	0	6667.0
3	600.0	2	2.0	1	281	1.0	4200000	1	1	1	6667.0
4	650.0	2	2.0	1	282	1.0	6200000	1	0	1	6667.0
...	...	...	...	...	...	...	...	...	...	...	...
1254	4118.0	4	5.0	2	44	3.0	55000000	1	0	1	12916.0
1255	1050.0	3	2.0	1	44	3.0	12500000	1	1	1	12916.0
1256	875.0	3	3.0	1	44	3.0	17500000	1	0	1	12916.0
1257	990.0	2	2.0	2	45	1.0	11500000	1	1	1	12916.0
1258	11050.0	3	3.0	2	44	1.0	18500000	1	0	1	12916.0

1259 rows × 11 columns

Droping the null values.

In [9]:
```
df1 =  df.dropna()
```

In [10]:
```
df1
```

Out[10]:

	Area	BHK	Bathroom	Furnishing	Locality	Parking	Price	Status	Transaction	Type	Per_Sqft
1	750.0	2	2.0	1	139	1.0	5000000	1	0	0	6667.0
2	950.0	2	2.0	0	49	1.0	15500000	1	1	0	6667.0
3	600.0	2	2.0	1	281	1.0	4200000	1	1	1	6667.0
4	650.0	2	2.0	1	282	1.0	6200000	1	0	1	6667.0
5	1300.0	4	3.0	1	281	1.0	15500000	1	0	1	6667.0
...	...	...	...	...	...	...	...	...	...	...	...
1254	4118.0	4	5.0	2	44	3.0	55000000	1	0	1	12916.0
1255	1050.0	3	2.0	1	44	3.0	12500000	1	1	1	12916.0
1256	875.0	3	3.0	1	44	3.0	17500000	1	0	1	12916.0
1257	990.0	2	2.0	2	45	1.0	11500000	1	1	1	12916.0
1258	11050.0	3	3.0	2	44	1.0	18500000	1	0	1	12916.0

1005 rows × 11 columns

In [13]:
```
df1.describe()
```

Out[13]:

	Area	BHK	Bathroom	Furnishing	Locality	Parking	Price	Status	Transaction	Type	Per_Sq
count	1005.000000	1005.000000	1005.000000	1005.000000	1005.000000	1005.000000	1.005000e+03	1005.000000	1005.000000	1005.000000	1005.0000
mean	1504.301968	2.791045	2.575124	1.159204	190.655721	1.697512	2.224030e+07	0.935323	0.600000	0.547264	15663.6308
std	1729.104830	0.961469	1.088503	0.644102	103.791974	3.223118	2.771744e+07	0.246077	0.490142	0.498009	21170.1604
min	28.000000	1.000000	1.000000	0.000000	0.000000	1.000000	1.000000e+06	0.000000	0.000000	0.000000	1259.0000
25%	770.000000	2.000000	2.000000	1.000000	116.000000	1.000000	5.130000e+06	1.000000	0.000000	0.000000	6364.0000
50%	1150.000000	3.000000	2.000000	1.000000	179.000000	1.000000	1.400000e+07	1.000000	1.000000	1.000000	11363.0000
75%	1700.000000	3.000000	3.000000	2.000000	281.000000	2.000000	2.700000e+07	1.000000	1.000000	1.000000	18000.0000
max	24300.000000	7.000000	7.000000	2.000000	364.000000	39.000000	2.400000e+08	1.000000	1.000000	1.000000	183333.0000

In [30]:

```
df['Per_Sqft'].isnull().sum()
```

Out[30]:

```
237
```

In [24]:

```
df['Parking'].fillna(int(df['Parking'].mode()), inplace=True)
```

In [18]:

```
X= df1[['Area','BHK','Bathroom','Furnishing','Locality','Parking','Status','Transaction','Type']]
y=df1['Price']
```

Importing the model and traning and test module from sklearn library.

In [19]:

```
import numpy as np
from sklearn import model_selection
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, y, test_size=0.2)
```

In [20]:

```
linear=LinearRegression()
```

Training the Regression model and Checking the accuracy of model.

In [21]:

```
linear.fit(X_train,Y_train)
Y_pred = linear.predict(X_test)
print(f"Accuracy of Test Data is {round(linear.score(X_test, Y_test)*100,2)}%")
print(f"Accuracy of Training Data is {round(linear.score(X_train, Y_train)*100,2)}%")
```

```
Accuracy of Test Data is 61.95%
Accuracy of Training Data is 58.4%
```

```
In [36]: from sklearn.metrics import mean_squared_error, mean_absolute_error
```

Checking the Error margin between the predicted and original values.

```
In [38]: print(np.sqrt(mean_squared_error(Y_test, Y_pred)))
         print(mean_absolute_error(Y_test, Y_pred))
```

```
9156056.395526567
6438574.635686093
```

Learning Outcomes

Learned Multiple Linear Regression techniques. Learned how to make prediction using Multiple Linear Regression.

Result/ Conclusion We have successfully trained the model and predicted the results.