```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

data=pd.read_csv('/content/Housing (1).csv')

data.head()
```

```
      price  area  bedrooms  bathrooms  stories mainroad guestroom
basement  \
0  13300000  7420         4          2        3      yes        no
no
1  12250000  8960         4          4        4      yes        no
no
2  12250000  9960         3          2        2      yes        no
yes
3  12215000  7500         4          2        2      yes        no
yes
4  11410000  7420         4          1        2      yes       yes
yes

  hotwaterheating airconditioning  parking prefarea furnishingstatus
0              no             yes        2      yes        furnished
1              no             yes        3       no        furnished
2              no              no        2      yes   semi-furnished
3              no             yes        3      yes        furnished
4              no             yes        2       no        furnished
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   price             545 non-null    int64
 1   area              545 non-null    int64
 2   bedrooms          545 non-null    int64
 3   bathrooms         545 non-null    int64
 4   stories           545 non-null    int64
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64
```

```
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

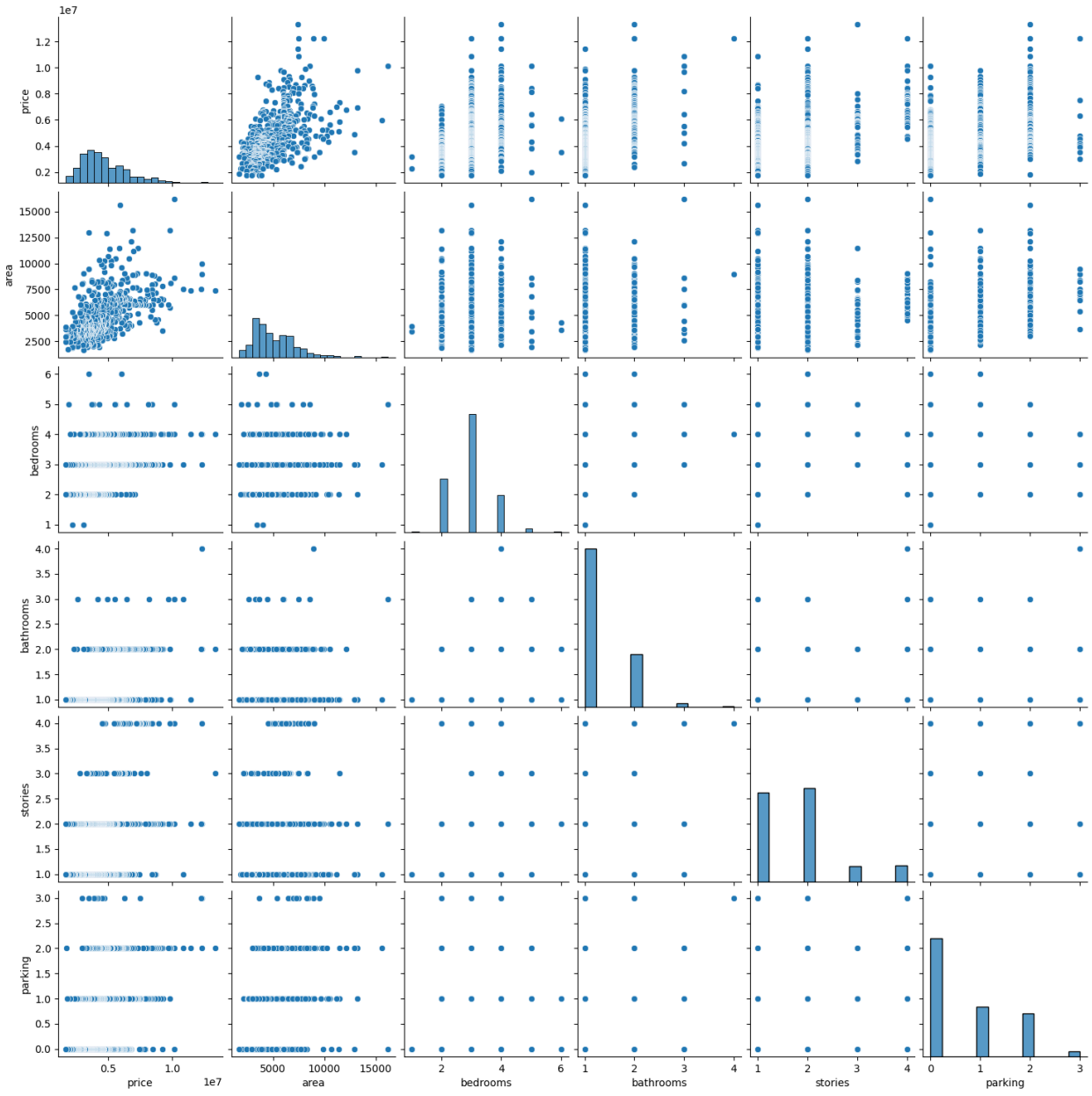
```
data.describe()
```

```
              price          area    bedrooms   bathrooms     stories
\
count  5.450000e+02    545.000000  545.000000  545.000000  545.000000

mean   4.766729e+06   5150.541284    2.965138    1.286239    1.805505

std    1.870440e+06   2170.141023    0.738064    0.502470    0.867492

min    1.750000e+06   1650.000000    1.000000    1.000000    1.000000

25%    3.430000e+06   3600.000000    2.000000    1.000000    1.000000

50%    4.340000e+06   4600.000000    3.000000    1.000000    2.000000

75%    5.740000e+06   6360.000000    3.000000    2.000000    2.000000

max    1.330000e+07  16200.000000    6.000000    4.000000    4.000000


          parking
count  545.000000
mean     0.693578
std      0.861586
min      0.000000
25%      0.000000
50%      0.000000
75%      1.000000
max      3.000000
```

```
data.isnull().sum()
```

```
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
hotwaterheating     0
airconditioning     0
parking             0
prefarea            0
```

```
furnishingstatus    0
dtype: int64

sns.pairplot(data)
plt.show()
```




```
def toNumeric(x):
    return x.map({"no":0,"yes":1})
def convert_binary():
    for column in list(data.select_dtypes(['object']).columns):
        if(column != 'furnishingstatus'):
```

```
            data[[column]] = data[[column]].apply(toNumeric)
convert_binary()

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

```
     furnished  semi-furnished  unfurnished
0            1               0            0
1            1               0            0
2            0               1            0
3            1               0            0
4            1               0            0
..         ...             ...          ...
540          0               0            1
541          0               1            0
542          0               0            1
543          1               0            0
544          0               0            1

[545 rows x 3 columns]
```

```
status = pd.get_dummies(data['furnishingstatus'], drop_first=True)

data = pd.concat([data, status], axis=1)

data.drop(columns='furnishingstatus',inplace=True)

Y = data.price
# includes the fields other than prices
X = data.iloc[:,1:]

from sklearn.preprocessing import MinMaxScaler
from statsmodels.stats.outliers_influence import
variance_inflation_factor
def preprocessing(X):
    scaler = MinMaxScaler()
    X_scaled = scaler.fit_transform(X)
    variables = X_scaled
    vif = pd.DataFrame()
    vif["VIF"] = [variance_inflation_factor(variables, i) for i in
range(variables.shape[1])]
    vif["Features"] = X.columns
    print(vif)

preprocessing(X)
```

```
         VIF         Features
0   4.642181             area
1   7.548951         bedrooms
2   1.685519        bathrooms
3   2.748302          stories
```

```
4   5.912370         mainroad
5   1.475439        guestroom
6   2.013754         basement
7   1.089327  hotwaterheating
8   1.762761  airconditioning
9   2.000022          parking
10  1.497539         prefarea
11  2.244298   semi-furnished
12  1.874527      unfurnished
```

```
X.drop(['area','bedrooms'], axis=1, inplace=True)
preprocessing(X)
```

```
         VIF         Features
0   1.591949        bathrooms
1   2.323144          stories
2   4.480333         mainroad
3   1.464301        guestroom
4   1.896633         basement
5   1.086156  hotwaterheating
6   1.720275  airconditioning
7   1.823778          parking
8   1.460957         prefarea
9   1.975297   semi-furnished
10  1.627909      unfurnished
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size =
0.25,random_state=355)
```

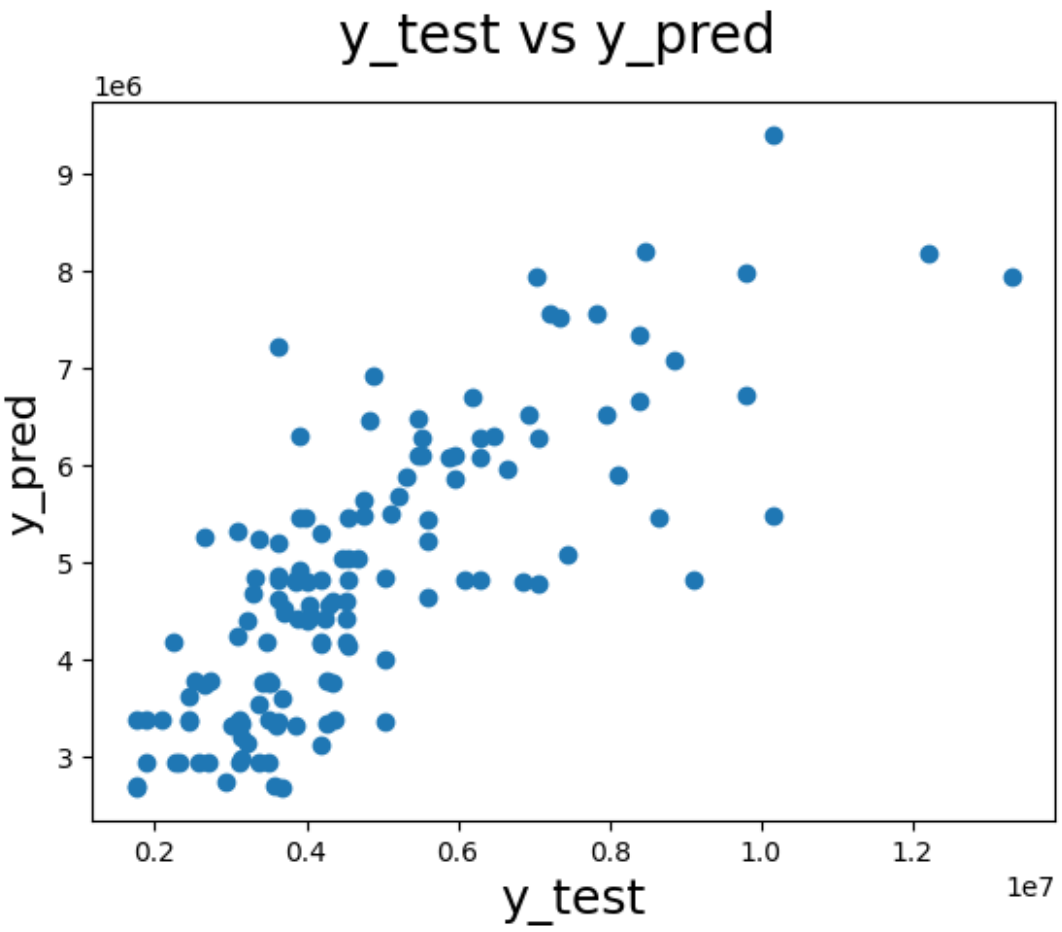
```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()
regression.fit(x_train,y_train)
```

```
LinearRegression()
```

```
y_predict = regression.predict(x_test)
```

```
plt.scatter(y_test,y_predict)
plt.suptitle('y_test vs y_pred', fontsize=20)
plt.xlabel('y_test', fontsize=18)
plt.ylabel('y_pred', fontsize=16)
```

```
Text(0, 0.5, 'y_pred')
```




```
mse=mean_squared_error(y_test,y_predict)

mse

1825146656372.6233
```