House Price Prediction

About

This project attempts to analyse the correlation between variables to determine the most important factors that affect house prices. The accuracy of the prediction is evaluated by checking the root square and root mean square error scores of the training model. The test is performed after applying the required pre-processing methods and splitting the data into two parts.
The model uses the data from Housing.csv and the machine learning model is developed using Python Programming Language and linear regression machine learning algorithm.

Steps:

1. Data Cleaning
2. Feature Engineering
3. Linear Regression
4. Correlation
5. Accuracy

Data Mining:

Data is read from the csv file using Pandas Library and we check for null values. We start off with the data cleaning process and the main objective is to remove the null values and replace them with either average values or remove those rows if not required. Data Mining also includes removing the data which is not required for making the prediction. Here as we can observe, the data does not contain any null values, hence we can proceed with the next step.

```
In [104]: df =  pd.read_csv('Housing.csv')
          df.head()
```

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

```
In [105]: df.shape

(545, 13)

In [106]: df.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
```

Feature Engineering:

As we can observe, most of the variables have object data type, machine learning models require integers or floats. We use feature engineering methods to convert them into numerical values. Since the values of these attributes are in binary that is in "YES" or "NO", we use Label Encoding Method. Sklearn library contains encoding function and once the conversion is done, all the "YES" become 1 and all the "NO" become zeros. Now we can proceed with the next step.

```
In [114]: from sklearn.preprocessing import LabelEncoder

In [115]: lb = LabelEncoder()

In [116]: df['mainroad'] = lb.fit_transform(df['mainroad'])
          df['mainroad'].value_counts()

1    468
0     77
Name: mainroad, dtype: int64

In [117]: df['guestroom'] = lb.fit_transform(df['guestroom'])
          df['guestroom'].value_counts()

0    448
1     97
Name: guestroom, dtype: int64

In [118]: df['hotwaterheating'] = lb.fit_transform(df['hotwaterheating'])
          df['hotwaterheating'].value_counts()

0    520
1     25
Name: hotwaterheating, dtype: int64
```

Linear Regression:

Multiple Linear Regression (MLR) is a supervised technique used to estimate the relationship between one dependent variable and more than one independent variables. Identifying the correlation and its cause-effect helps to make predictions by using these relations. To estimate these relationships, the prediction accuracy of the model is essential; the complexity of the model is of more interest.

```
In [141]: m2 = LinearRegression()
          m2.fit(x_tr,y_tr)

LinearRegression()

In [142]: # R2 score
          print('Training score',m2.score(x_tr,y_tr))
          print('Testing score',m2.score(x_te,y_te))

Training score 0.6297594795010557
Testing score 0.6604055285948669

In [143]: from sklearn.metrics import confusion_matrix,classification_report

In [144]: ypred_m2 = m2.predict(x_te)
          print(ypred_m2)

[3125055.54514033 6075936.49474194 3451230.99986177 7821353.59382406
 3632379.63007162 4339490.62421742 6753531.76372861 3520984.15158488
 7389332.28031017 7028419.62033754 3331078.48957497 7071204.52534607
 6442585.35188309 6358882.86176688 4617767.44587996 6101918.48519953
 2904764.97720884 7389332.28031017 3955537.80218495 3677972.83217974
 3513912.93769487 5685352.95699471 2885895.48666065 3995840.20876345
 5306686.59506882 4769673.61264754 4171370.50189697 7407018.04235965
 8509867.58832243 6814338.1654869  7960286.13058762 4966059.71254294
 5877672.1384599  2935149.88313042 3837796.29173881 4185956.67209031
 5174824.05444237 8199123.08203213 5757095.84904955 5222688.24678383
 3574157.73694765 3369059.62197695 5110348.03151279 5110358.84217145
 3966110.02525407 2881976.29776765 4804389.37526334 3098589.82811175
 3899650.51351077 3767845.66897203 3577961.21494616 6460049.50195696
 7225892.33532884 6345182.7546327  5338156.25404417 3546304.90651953
```

Correlation:

As we can see the furnishing status is negatively correlated to price of the house, but all the other attributes are positively correlated. It means that as if furnishing status is increased then the price decreases.

```
In [136]: df.corr()
```

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	aircond
price	1.000000	0.535997	0.366494	0.517545	0.420712	0.296898	0.255517	0.187057	0.093073	0.45295
area	0.535997	1.000000	0.151858	0.193820	0.083996	0.288874	0.140297	0.047417	-0.009229	0.22239
bedrooms	0.366494	0.151858	1.000000	0.373930	0.408564	-0.012033	0.080549	0.097312	0.046049	0.16060
bathrooms	0.517545	0.193820	0.373930	1.000000	0.326165	0.042398	0.126469	0.102106	0.067159	0.18691
stories	0.420712	0.083996	0.408564	0.326165	1.000000	0.121706	0.043538	-0.172394	0.018847	0.29360
mainroad	0.296898	0.288874	-0.012033	0.042398	0.121706	1.000000	0.092337	0.044002	-0.011781	0.10542
guestroom	0.255517	0.140297	0.080549	0.126469	0.043538	0.092337	1.000000	0.372066	-0.010308	0.13817
basement	0.187057	0.047417	0.097312	0.102106	-0.172394	0.044002	0.372066	1.000000	0.004385	0.04734
hotwaterheating	0.093073	-0.009229	0.046049	0.067159	0.018847	-0.011781	-0.010308	0.004385	1.000000	-0.13002
airconditioning	0.452954	0.222393	0.160603	0.186915	0.293602	0.105423	0.138179	0.047341	-0.130023	1.00000
parking	0.384394	0.352980	0.139270	0.177496	0.045547	0.204433	0.037466	0.051497	0.067864	0.15917
prefarea	0.329777	0.234779	0.079023	0.063472	0.044425	0.199876	0.160897	0.228083	-0.059411	0.11738
furnishingstatus	-0.304721	-0.171445	-0.123244	-0.143559	-0.104672	-0.156726	-0.118328	-0.112831	-0.031628	-0.15047

```
In [137]: df.corr()['price']

price              1.000000
area               0.535997
bedrooms           0.366494
bathrooms          0.517545
stories            0.420712
mainroad           0.296898
guestroom          0.255517
basement           0.187057
hotwaterheating    0.093073
airconditioning    0.452954
parking            0.384394
prefarea           0.329777
furnishingstatus  -0.304721
Name: price, dtype: float64
```

Accuracy:

MSE is the average of the squared error that is used as the loss function for least squares regression: It is the sum, over all the data points, of the square of the difference between the predicted and actual target variables, divided by the number of data points. RMSE is the square root of MSE.

```
In [145]: mse_m2 = mean_squared_error(y_te,ypred_m2)
          rmse_m2 = np.sqrt(mean_squared_error(y_te,ypred_m2))
          mae_m2 = mean_absolute_error(y_te,ypred_m2)
          print('MSE m2',mse_m2)
          print('RMSE m2',rmse_m2)
          print('MAE m2',mae_m2)

MSE m2 1264343237370.1475
RMSE m2 1124430.1834129798
MAE m2 847964.4816048648
```