Data Science Assignment-1

Problem statement:
The goal is to understand the relationship between house features and how these variables affect the house price. Using more than one model, predict the price of the house using the given dataset. Please compare the accuracy of the models along with the drawbacks of each technique's assumptions before recommending the final prediction model.

In [1]:
```
# Price Predictor
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
housing=pd.read_excel('House_Price.xlsx')
housing.head()
```

Out[2]:

	Transaction date	House Age	Distance from nearest Metro station (km)	Number of convenience stores	latitude	longitude	Number of bedrooms	House size (sqft)	House price of unit area
0	2012.916667	32.0	84.87882	10	24.98298	121.54024	1	575	37.9
1	2012.916667	19.5	306.59470	9	24.98034	121.53951	2	1240	42.2
2	2013.583333	13.3	561.98450	5	24.98746	121.54391	3	1060	47.3
3	2013.500000	13.3	561.98450	5	24.98746	121.54391	2	875	54.8
4	2012.833333	5.0	390.56840	5	24.97937	121.54245	1	491	43.1

In [3]:
```
#train_test_split randomly split
from sklearn.model_selection import train_test_split
train_set,test_set=train_test_split(housing,test_size=0.2,random_state=42) #this gives two values
print('Rows in train set:',len(train_set))
print('Rows in test set:',len(test_set))
```

```
Rows in train set: 331
Rows in test set: 83
```

In [4]:
```
#Stratified shuffle split
from sklearn.model_selection import StratifiedShuffleSplit
split=StratifiedShuffleSplit(n_splits=1,test_size=0.2,random_state=42)
for train_index,test_index in split.split(housing,housing['Number of convenience stores']):
    strat_train_set=housing.loc[train_index]
    strat_test_set=housing.loc[test_index]
```

In [5]:
```
print('***Value count of Test set***\n',strat_test_set['Number of convenience stores'].value_counts())
print('***Value count of Train set***\n',strat_train_set['Number of convenience stores'].value_counts())
```

```
***Value count of Test set***
 0     14
5     14
1      9
3      9
6      7
4      6
7      6
8      6
2      5
9      5
10     2
Name: Number of convenience stores, dtype: int64
***Value count of Train set***
 0     53
5     53
1     37
3     37
6     30
4     25
7     25
8     24
9     20
2     19
10     8
Name: Number of convenience stores, dtype: int64
```

In [6]:
```
housing=strat_train_set.copy()
```

In [7]:
```
# Looking for correlations
corr_matrix=housing.corr()
print(corr_matrix['House price of unit area'].sort_values(ascending=False))
```

```
House price of unit area                  1.000000
Number of convenience stores              0.636209
latitude                                  0.569626
longitude                                 0.535064
Transaction date                          0.054341
Number of bedrooms                        0.016897
House size (sqft)                        -0.001269
House Age                                -0.182001
Distance from nearest Metro station (km)   -0.692165
Name: House price of unit area, dtype: float64
```

In [8]:
```
housing.isnull().sum()
```

Out[8]:
```
Transaction date                            0
House Age                                   0
Distance from nearest Metro station (km)    0
Number of convenience stores                0
latitude                                    0
longitude                                   0
Number of bedrooms                          0
House size (sqft)                           0
House price of unit area                    0
dtype: int64
```

In [9]:
```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
```

In [10]:
```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
```

Model-1

In [11]:
```
model=LinearRegression()
```

In [12]:
```
housing=strat_train_set.drop('House price of unit area',axis=1)
housing_labels=strat_train_set['House price of unit area'].copy()
model.fit(housing,housing_labels)
```

Out[12]:
```
LinearRegression()
```

In [13]:
```
some_data=housing.iloc[:5]
some_labels=housing_labels.iloc[:5]
pred=model.predict(some_data)
print('predicted values are',list(pred),'\nand actual labels are',list(some_labels))
```

```
predicted values are [27.836573416250758, 48.24702052440625, 9.881286881818596, 44.1758353887999, 45.612422611749935]
and actual labels are [22.8, 47.1, 15.5, 44.3, 41.2]
```

In [14]:
```
# Evaluating model
from sklearn.metrics import mean_squared_error
housing_predictions=model.predict(housing)
mse=mean_squared_error(housing_labels,housing_predictions)
rsme=np.sqrt(mse)
print(mse)
```

```
60.992564207350085
```

In [15]:
```
# Using better evaluation technique - Cross validation
from sklearn.model_selection import cross_val_score
scores=cross_val_score(model,housing,housing_labels,scoring="neg_mean_squared_error",cv=10)
#Cv is number of divied parts or folds.
rsme_scores=np.sqrt(-scores)
print(rsme_scores)
```

```
[ 6.39863553  8.50970388  6.24220125  9.1564894  10.91389734  6.14710518
  6.15467518  6.98682238  7.92333128 10.31401586]
```

In [16]:
```
def print_scores(scores):
    print('Scores:',scores)
    print('Mean:',scores.mean())
    print('Standard Deviation:',scores.std())
```

In [17]:
```
print_scores(rsme_scores)
```

```
Scores: [ 6.39863553  8.50970388  6.24220125  9.1564894  10.91389734  6.14710518
  6.15467518  6.98682238  7.92333128 10.31401586]
Mean: 7.874687727880141
Standard Deviation: 1.6973737272242253
```

In [18]:
```
# Testing the model on test data
X_test=strat_test_set.drop('House price of unit area',axis=1)
Y_test=strat_test_set['House price of unit area'].copy()
final_predictions=model.predict(X_test)
final_mse=mean_squared_error(Y_test,final_predictions)
final_rsme=np.sqrt(final_mse)
```

In [19]:
```
print(final_predictions,"\n\n",list(Y_test))
```

```
[46.92799712 43.96904911 15.9901229  28.09845129 45.68847365 14.34858266
 44.2302692  34.19547373 35.5376544  39.13552009 33.36858147 52.18149124
 37.91851481 30.18810431 47.19589297 41.00845666 12.826473   53.11108652
 48.24121402 40.43869339 46.49210735 32.39606974 41.20231965 35.07918803
 31.64667375 35.25238884 37.93321187 26.02637152 39.17191222 40.85840652
 30.8991204  37.73359559 43.67623775 40.82537263 11.91887837 45.24463686
 46.71834351 28.52383962 43.28937093 12.8961219  36.05050653 40.70674797
 28.57426662 44.50190757 30.27237094 51.82925752 49.69869544 37.24704106
 33.11591035 54.29178341 45.09166145 44.96316251 14.2555597  46.24275025
 48.95655511 42.47623295 42.92974125 47.27525823 50.0017231  37.99299965
 37.46147785 13.46555658 35.07462773 38.52627061 28.18225759 34.8707732
 42.61761369 50.24188009 23.42787081 40.37110314 45.31599639 13.64730391
 37.79689644 48.86665704 44.28192026 33.01842154 54.21762461 29.91043298
 47.97037097 13.6550114  38.15919255 47.34316088 54.62237778]

 [59.6, 40.3, 25.3, 24.7, 55.0, 45.1, 34.4, 23.6, 38.8, 117.5, 33.4, 56.3, 60.7, 27.7, 53.7, 30.0, 17.4, 46.6, 47.3, 42.3, 49.
3, 42.9, 22.8, 30.7, 27.0, 26.9, 37.9, 25.9, 50.2, 44.2, 29.5, 36.6, 41.0, 33.6, 11.2, 26.5, 45.1, 21.8, 40.6, 14.7, 39.0, 45.
5, 31.9, 40.2, 26.6, 58.0, 39.5, 37.4, 23.1, 54.4, 51.7, 36.3, 15.6, 51.4, 42.2, 38.4, 51.7, 49.7, 69.7, 37.8, 40.3, 11.6, 33.
4, 46.4, 25.6, 31.3, 33.1, 73.6, 19.1, 39.3, 40.1, 29.3, 32.9, 47.1, 41.9, 30.6, 58.1, 28.9, 54.8, 12.8, 55.2, 62.1, 44.7]
```

Model-2

In [20]:
```
model=DecisionTreeRegressor()
```

In [21]:
```
housing=strat_train_set.drop('House price of unit area',axis=1)
housing_labels=strat_train_set['House price of unit area'].copy()
model.fit(housing,housing_labels)
```

Out[21]:
```
DecisionTreeRegressor()
```

In [22]:
```
some_data=housing.iloc[:5]
some_labels=housing_labels.iloc[:5]
pred=model.predict(some_data)
print('predicted values are',list(pred),'\nand actual labels are',list(some_labels))
```

```
predicted values are [22.8, 47.1, 15.5, 44.3, 41.2]
and actual labels are [22.8, 47.1, 15.5, 44.3, 41.2]
```

In [23]:
```
# Evaluating model
from sklearn.metrics import mean_squared_error
housing_predictions=model.predict(housing)
mse=mean_squared_error(housing_labels,housing_predictions)
rsme=np.sqrt(mse)
print(mse)
```

```
0.0
```

In [24]:
```
# Using better evaluation technique - Cross validation
from sklearn.model_selection import cross_val_score
scores=cross_val_score(model,housing,housing_labels,scoring="neg_mean_squared_error",cv=10)
#Cv is number of divied parts or folds.
rsme_scores=np.sqrt(-scores)
print(rsme_scores)
```

```
[ 7.87413861 11.19113556  7.95609162 11.07831625 10.87918502  9.48913254
  9.52492742  9.85245701  6.22825916  9.2160762 ]
```

In [25]:
```python
def print_scores(scores):
    print('Scores:',scores)
    print('Mean:',scores.mean())
    print('Standard Deviation:',scores.std())
```

In [26]:
```python
print_scores(rsme_scores)
```

```
Scores: [ 7.87413861 11.19113556  7.95609162 11.07831625 10.87918502  9.48913254
  9.52492742  9.85245701  6.22825916  9.2160762 ]
Mean: 9.328971939409083
Standard Deviation: 1.5131956791319445
```

In [27]:
```python
# Testing the model on test data
X_test=strat_test_set.drop('House price of unit area',axis=1)
Y_test=strat_test_set['House price of unit area'].copy()
final_predictions=model.predict(X_test)
final_mse=mean_squared_error(Y_test,final_predictions)
final_rsme=np.sqrt(final_mse)
```

In [28]:
```python
print(final_predictions,"\n\n",list(Y_test))
```

```
[53.   53.9 20.7 22.3 53.9 29.3 41.1 21.4 36.8 52.2 28.4 58.8 67.7 22.8
 47.9 40.6 17.4 37.9 50.5 38.9 53.   29.3 34.3 28.6 29.4 47.4 55.3 20.7
 28.5 43.1 25.5 35.6 78.3 45.4 24.7 41.1 48.6 22.8 30.5 20.5 40.6 42.8
 30.9 42.  25.3 58.1 40.8 23.  23.6 49.8 48.  36.5 13.  53.  39.7 47.7
 45.4 47.7 63.3 38.2 42.5 20.5 28.6 35.6 22.8 28.4 40.6 63.3 16.1 40.5
 42.  20.5 29.3 44.3 38.6 28.1 59.  28.4 45.9 20.5 52.2 36.9 52.2]

 [59.6, 40.3, 25.3, 24.7, 55.0, 45.1, 34.4, 23.6, 38.8, 117.5, 33.4, 56.3, 60.7, 27.7, 53.7, 30.0, 17.4, 46.6, 47.3, 42.3, 49.
3, 42.9, 22.8, 30.7, 27.0, 26.9, 37.9, 25.9, 50.2, 44.2, 29.5, 36.6, 41.0, 33.6, 11.2, 26.5, 45.1, 21.8, 40.6, 14.7, 39.0, 45.
5, 31.9, 40.2, 26.6, 58.0, 39.5, 37.4, 23.1, 54.4, 51.7, 36.3, 15.6, 51.4, 42.2, 38.4, 51.7, 49.7, 69.7, 37.8, 40.3, 11.6, 33.
4, 46.4, 25.6, 31.3, 33.1, 73.6, 19.1, 39.3, 40.1, 29.3, 32.9, 47.1, 41.9, 30.6, 58.1, 28.9, 54.8, 12.8, 55.2, 62.1, 44.7]
```

Model-3

In [29]:
```python
model=RandomForestRegressor()
```

In [30]:
```python
housing=strat_train_set.drop('House price of unit area',axis=1)
housing_labels=strat_train_set['House price of unit area'].copy()
model.fit(housing,housing_labels)
```

Out[30]:
```
RandomForestRegressor()
```

In [31]:
```python
some_data=housing.iloc[:5]
some_labels=housing_labels.iloc[:5]
pred=model.predict(some_data)
print('predicted values are',list(pred),'\nand actual labels are',list(some_labels))
```

```
predicted values are [23.900999999999999, 44.948999999999984, 15.194000000000008, 43.77500000000006, 43.948999999999995]
and actual labels are [22.8, 47.1, 15.5, 44.3, 41.2]
```

In [32]:
```python
# Evaluating model
from sklearn.metrics import mean_squared_error
housing_predictions=model.predict(housing)
mse=mean_squared_error(housing_labels,housing_predictions)
rsme=np.sqrt(mse)
print(mse)
```

```
6.058572145015106
```

In [33]:
```python
# Using better evaluation technique - Cross validation
from sklearn.model_selection import cross_val_score
scores=cross_val_score(model,housing,housing_labels,scoring="neg_mean_squared_error",cv=10)
#Cv is number of divied parts or folds.
rsme_scores=np.sqrt(-scores)
print(rsme_scores)
```

```
[4.61710341 7.89335003 4.69523427 7.89246341 9.11095349 4.85351865
 6.41482531 5.54925493 5.08306303 8.92622794]
```

In [34]:
```python
def print_scores(scores):
    print('Scores:',scores)
    print('Mean:',scores.mean())
    print('Standard Deviation:',scores.std())
```

In [35]:
```
print_scores(rsme_scores)
```

```
Scores: [4.61710341 7.89335003 4.69523427 7.89246341 9.11095349 4.85351865
 6.41482531 5.54925493 5.08306303 8.92622794]
Mean: 6.50359944697554
Standard Deviation: 1.7034133469671018
```

In [36]:
```
# Testing the model on test data
X_test=strat_test_set.drop('House price of unit area',axis=1)
Y_test=strat_test_set['House price of unit area'].copy()
final_predictions=model.predict(X_test)
final_mse=mean_squared_error(Y_test,final_predictions)
final_rsme=np.sqrt(final_mse)
```

In [37]:
```
print(final_predictions,"\n\n",list(Y_test))
```

```
[51.438 44.434 19.744 24.827 49.561 24.427 41.198 25.706 40.107 47.025
 26.788 55.922 54.596 24.861 56.918 41.645 15.753 44.942 46.814 38.683
 47.669 38.37  33.299 28.563 25.53  44.266 50.668 23.624 34.271 41.201
 26.824 39.044 56.268 44.587 21.856 36.97  49.629 23.85  35.322 17.435
 39.766 50.208 29.683 38.457 27.057 54.735 43.726 32.905 24.907 53.268
 50.334 43.544 16.855 51.302 44.457 49.877 49.412 48.947 63.397 39.821
 37.553 16.577 28.563 41.739 24.376 28.086 39.413 63.884 15.731 39.054
 41.593 21.946 32.97  44.981 39.326 26.379 59.586 27.053 44.064 16.626
 51.432 53.108 48.869]

 [59.6, 40.3, 25.3, 24.7, 55.0, 45.1, 34.4, 23.6, 38.8, 117.5, 33.4, 56.3, 60.7, 27.7, 53.7, 30.0, 17.4, 46.6, 47.3, 42.3, 49.
3, 42.9, 22.8, 30.7, 27.0, 26.9, 37.9, 25.9, 50.2, 44.2, 29.5, 36.6, 41.0, 33.6, 11.2, 26.5, 45.1, 21.8, 40.6, 14.7, 39.0, 45.
5, 31.9, 40.2, 26.6, 58.0, 39.5, 37.4, 23.1, 54.4, 51.7, 36.3, 15.6, 51.4, 42.2, 38.4, 51.7, 49.7, 69.7, 37.8, 40.3, 11.6, 33.
4, 46.4, 25.6, 31.3, 33.1, 73.6, 19.1, 39.3, 40.1, 29.3, 32.9, 47.1, 41.9, 30.6, 58.1, 28.9, 54.8, 12.8, 55.2, 62.1, 44.7]
```