```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report,mean_absolute_error,mean_squared_error
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

Loading Dataset

```
df=pd.read_csv('Real_Estate_Prices_Dataset.csv')
df
```

	Location	Area (sq ft)	Bedrooms	Bathrooms	Age_of_Property (Years)	Nearby_Schools	Nearby_Hospitals	Public_Transport_Access	Recent_Renovati
0	Downtown	1500	3	2	5	3	2	Yes	
1	Suburban	2000	4	3	10	2	1	No	`
2	Countryside	1800	3	2	15	1	1	No	
3	Downtown	1000	2	1	20	4	3	Yes	
4	Suburban	1200	2	2	3	3	2	Yes	`
5	Countryside	2500	4	3	8	1	1	No	
6	Downtown	900	2	1	25	5	4	Yes	`
7	Suburban	1100	2	1	2	2	2	No	
8	Countryside	1600	3	2	12	0	0	No	`

Converting categorical to numerical values

```
df['Public_Transport_Access'] = df['Public_Transport_Access'] .map({'Yes': 1, 'No': 0})
df['Location'] = df['Location'] .map({'Downtown': 0, 'Suburban': 1, 'Countryside': 2})
df['Recent_Renovation'] = df['Recent_Renovation'] .map({'Yes': 1, 'No': 0})
df
```

	Location	Area (sq ft)	Bedrooms	Bathrooms	Age_of_Property (Years)	Nearby_Schools	Nearby_Hospi
0	0	1500	3	2	5	3	
1	1	2000	4	3	10	2	
2	2	1800	3	2	15	1	
3	0	1000	2	1	20	4	
4	1	1200	2	2	3	3	
5	2	2500	4	3	8	1	
6	0	900	2	1	25	5	
7	1	1100	2	1	2	2	
8	2	1600	3	2	12	0	

```
df.head()
```

	Location	Area (sq ft)	Bedrooms	Bathrooms	Age_of_Property (Years)	Nearby_Schools	Nearby_Hospi
0	0	1500	3	2	5	3	
1	1	2000	4	3	10	2	
2	2	1800	3	2	15	1	
3	0	1000	2	1	20	4	

```
df.columns
```

```
     Index(['Location', 'Area (sq ft)', 'Bedrooms', 'Bathrooms',
            'Age_of_Property (Years)', 'Nearby_Schools', 'Nearby_Hospitals',
            'Public_Transport_Access', 'Recent_Renovation', 'Selling_Price (USD)'],
           dtype='object')
```

Separating Target and Training Data

```
X = df[['Location', 'Area (sq ft)', 'Bedrooms', 'Bathrooms',
       'Age_of_Property (Years)', 'Nearby_Schools', 'Nearby_Hospitals',
       'Public_Transport_Access', 'Recent_Renovation']]
y = df['Selling_Price (USD)']
X.head()
```

	Location	Area (sq ft)	Bedrooms	Bathrooms	Age_of_Property (Years)	Nearby_Schools	Nearby_Hospi
0	0	1500	3	2	5	3	
1	1	2000	4	3	10	2	
2	2	1800	3	2	15	1	
3	0	1000	2	1	20	4	

Splitting Data into train and test

Double-click (or enter) to edit

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=40)
```

Training the Model i.e fitting

```
model=RandomForestRegressor(max_depth=6)
model.fit(X_train, y_train)
```

```
▾      RandomForestRegressor
RandomForestRegressor(max_depth=6)
```

hyperparameter max_depth tuning done

```
y_pred = model.predict(X_test)
```

```
     [263900. 218800.]
     4    280000
     3    220000
     Name: Selling_Price (USD), dtype: int64
```

Mean Absolute Error

Generate | Using ... | perform mean absolute error on the model | Close

< 1 of 4 > Undo Changes Use code with caution

```
mae = mean_absolute_error(y_pred, y_test)
print('The mean absolute error is:', mae)
```

```
     The mean absolute error is: 8650.0
```

Mean Squared Error

Double-click (or enter) to edit

```
mse = mean_squared_error(y_test, y_pred)

print('The mean squared error is:', mse)
```

```
The mean squared error is: 130325000.0
```

```
print("Predicted:",y_pred)
print("Actual:\n",y_test)
```

```
Predicted: [263900. 218800.]
Actual:
 4    280000
3    220000
Name: Selling_Price (USD), dtype: int64
```