House Price Prediction

Build a machine learning model to predict the median house prices based on different independent variables.

There are 14 attributes in each case of the dataset. They are:

- CRIM - per capita crime rate by town
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.
- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling
- AGE - proportion of owner-occupied units built prior to 1940
- DIS - weighted distances to five Boston employment centres
- RAD - index of accessibility to radial highways
- TAX - full-value property-tax rate per dollar 10,000
- PTRATIO - pupil-teacher ratio by town
- B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT - % lower status of the population
- MEDV - Median value of owner-occupied homes in dollar 1000's

Dataset : https://github.com/ybifoundation/Dataset/raw/main/Boston.csv

```
# Regression Predictive Model
# Step 1 : import library
import pandas as pd
```

```
# Step 2 : import data
house =pd.read_csv('https://github.com/ybifoundation/Dataset/raw/main/Boston.csv')
```

```
# Step 3 : define y and X
house.columns
```

```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT', 'MEDV'],
      dtype='object')
```

```
y = house['MEDV']
```

```
X = house[['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT']]
```

```
# Step 4 : train test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, train_size=0.7, random_state=2529)
```

```
# check shape of train and test sample
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((354, 13), (152, 13), (354,), (152,))
```

```
# Step 5 : select model
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

```
# Step 6 : train or fit model
model.fit(X_train,y_train)
```

```
LinearRegression()
```

```
# Step 7 : predict model
y_pred=model.predict(X_test)
```

```
# Step 8 : model accuracy
from sklearn.metrics import mean_absolute_error, mean_absolute_percentage_error, mean_squared_error
```

```
mean_absolute_error(y_test,y_pred)
```

```
3.1550309276025073
```

```
mean_absolute_percentage_error(y_test,y_pred)
```

```
0.16355935882218034
```

```
mean_squared_error(y_test,y_pred)
```

```
20.71801287783861
```

```

```