★ Regression Model to Predict Median House Prices

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

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

```
# import library
import pandas as pd
```

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

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

```
# view data
house.head()
```

	CRIM	ZN	INDUS	CHAS	NX	RM	AGE	DIS	RAD	TAX	PTRATIO	B
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90

```
# info of data
house.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
 2   INDUS    506 non-null    float64
 3   CHAS     506 non-null    int64  
 4   NX       506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      506 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    float64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    506 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

```
# summary statistics
house.describe()
```

	CRIM	ZN	INDUS	CHAS	NX	RM	
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.00
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.57
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.14
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.90
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.02

```
# check for missing value
house.isna().sum()
```

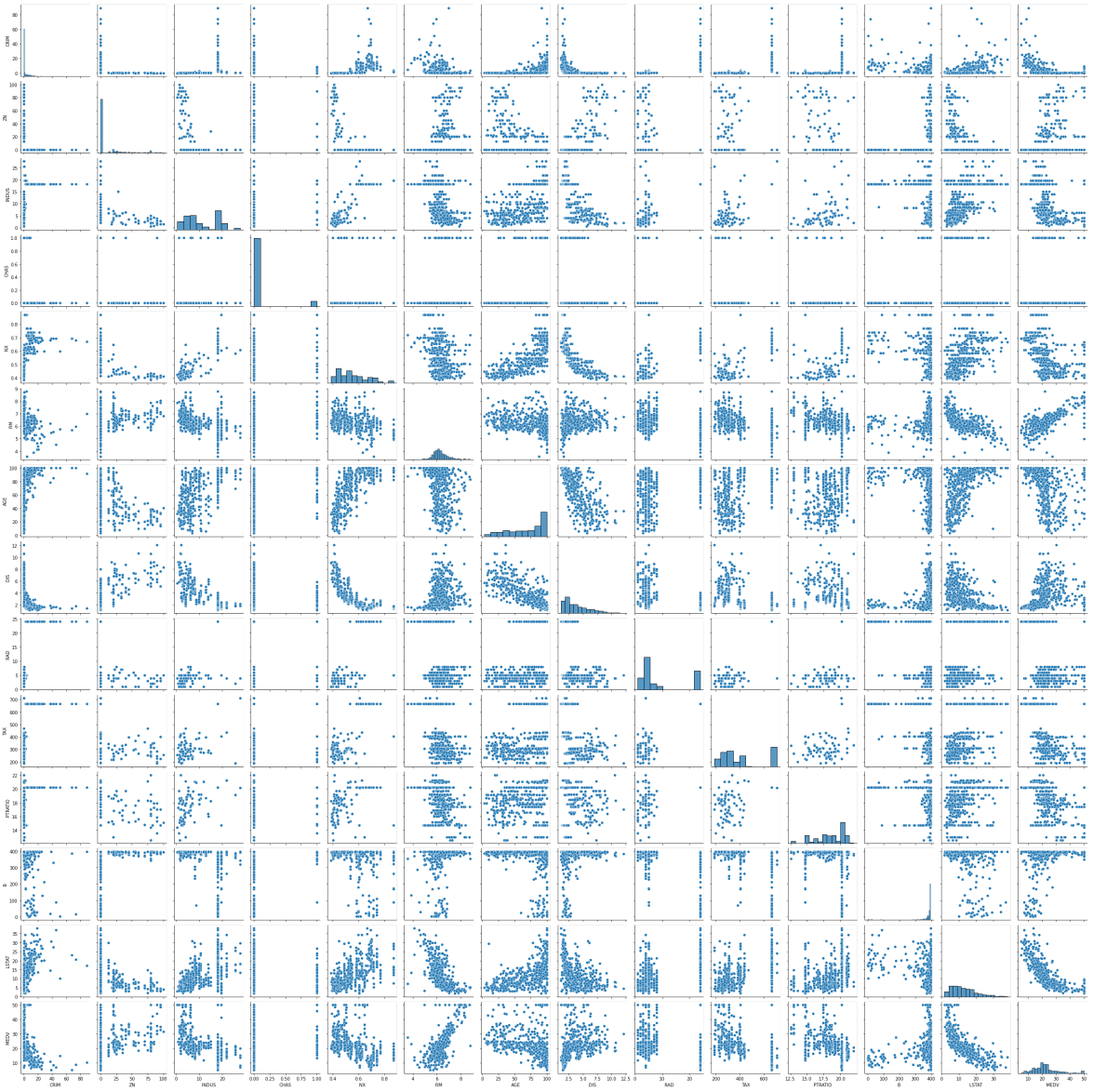
```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NX         0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
MEDV       0
dtype: int64
```

```
# check for categories
house.nunique()
```

```
CRIM       504
ZN          26
INDUS       76
CHAS         2
NX          81
RM         446
AGE        356
DIS        412
RAD          9
TAX         66
PTRATIO     46
B          357
LSTAT      455
MEDV       229
dtype: int64
```

```
# visualize pairplot
sns.pairplot(house)
```

```
<seaborn.axisgrid.PairGrid at 0x7f3c4d193690>
```



```
# columns name
house.columns
```

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

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

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

```
# split data
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=.30, random_state=2529)
```

```
# verify shape
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

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

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

```
# train model
model.fit(X_train, y_train)
```

```
LinearRegression()
```

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

```
# model evaluation
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error,mean_squared_error
```

```
# model MAE
mean_absolute_error(y_test, y_pred)
```

```
3.1550309276025073
```

```
# model MAPE
mean_absolute_percentage_error(y_test, y_pred)
```

```
0.16355935882218034
```

```
# model MSE
mean_squared_error(y_test, y_pred)
```

```
20.71801287783861
```

```
# future prediction
sample = house.sample()
sample
```

	CRIM	ZN	INDUS	CHAS	NX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEI
150	1.6566	0.0	19.58	0	0.871	6.122	97.3	1.618	5	403.0	14.7	372.8	14.1	21

```
# define X_new
X_new = sample.loc[:,X.columns]
X_new
```

	CRIM	ZN	INDUS	CHAS	NX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
150	1.6566	0.0	19.58	0	0.871	6.122	97.3	1.618	5	403.0	14.7	372.8	14.1

```
# predict for X_new
model.predict(X_new)
```

```
array([20.74780293])
```