Name : Suryal D . Khirade
Roll NO: T190424399
Assignment No :04

Data Analytics I

Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing (https://www.kaggle.com/c/boston-housing) (https://www.kaggle.com/c/boston-housing (https://www.kaggle.com/c/boston-housing)) (https://www.kaggle.com/c/boston-housing (https://www.kaggle.com/c/boston-housing) (https://www.kaggle.com/c/boston-housing (https://www.kaggle.com/c/boston-housing))). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset. The objective is to predict the value of prices of the house using the given features.

STEP 1 : IMPORT LIBRARIES

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

STEP 2 : LOAD DATASET

In [4]:
```
data = pd.read_csv("C:\\Users\\alisu\\Desktop\\SIT lonvala\\TE\\6th sem\\DSBDA\\LAB\\housing.csv")
```

In [5]:
```
data
```

Out[5]:

	0.00632 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00
0	0.02731 0.00 7.070 0 0.4690 6.4210 78...
1	0.02729 0.00 7.070 0 0.4690 7.1850 61...
2	0.03237 0.00 2.180 0 0.4580 6.9980 45...
3	0.06905 0.00 2.180 0 0.4580 7.1470 54...
4	0.02985 0.00 2.180 0 0.4580 6.4300 58...
...	...
500	0.06263 0.00 11.930 0 0.5730 6.5930 69...
501	0.04527 0.00 11.930 0 0.5730 6.1200 76...
502	0.06076 0.00 11.930 0 0.5730 6.9760 91...
503	0.10959 0.00 11.930 0 0.5730 6.7940 89...
504	0.04741 0.00 11.930 0 0.5730 6.0300 80...

505 rows × 1 columns

In [6]:
```
column_names= ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM',
 'AGE', 'DIS', 'RAD', 'TAX', 'PTRATIO', 'B', 'LSTAT', 'MEDV']
```

In [8]:
```
column_names
```

Out[8]:
```
['CRIM',
 'ZN',
 'INDUS',
 'CHAS',
 'NOX',
 'RM',
 'AGE',
 'DIS',
 'RAD',
 'TAX',
 'PTRATIO',
 'B',
 'LSTAT',
 'MEDV']
```

In [13]:
```
ead_csv("C:\\Users\\alisu\\Desktop\\SIT lonvala\\TE\\6th sem\\DSBDA\\LAB\\housing.csv", delimiter='\s+', names=column_names)
```

In [14]:
```
data
```

Out[14]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9

506 rows × 14 columns

In [15]:
```
data['price'] = data.MEDV
```

In [16]:
```
data.drop(data[["MEDV"]] , axis = "columns")
```

Out[16]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	price
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9

506 rows × 14 columns

In [17]:
```
data
```

Out[17]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	price
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9	11.9

506 rows × 15 columns

In [18]:
```
data.drop(data[["MEDV"]],axis="columns",inplace=True)
```

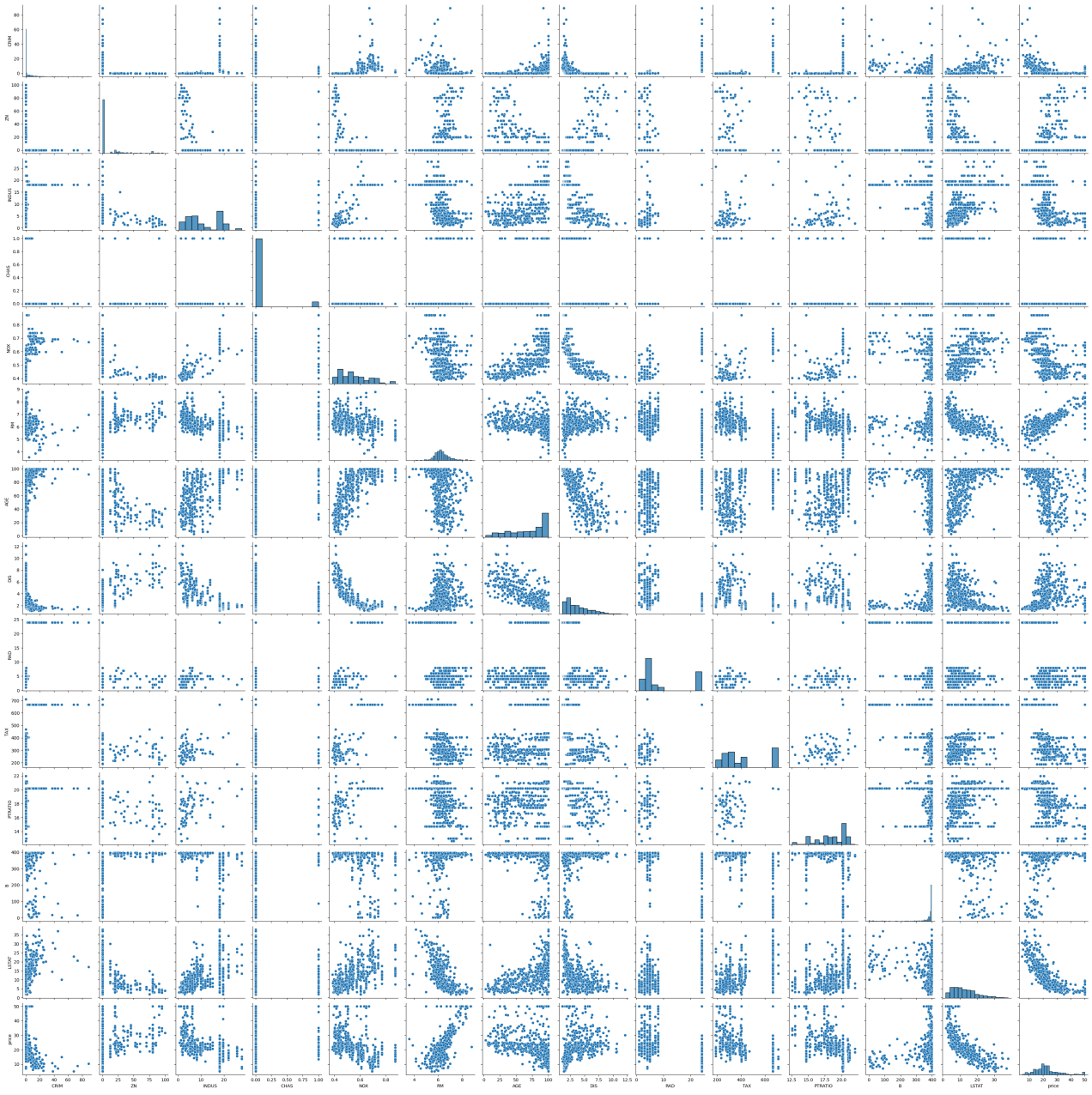
In [19]:
```
data.info()
```

```
<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   NOX      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  price    506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

In [22]:
```
sns.pairplot(data)
```

```
C:\Users\alisu\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

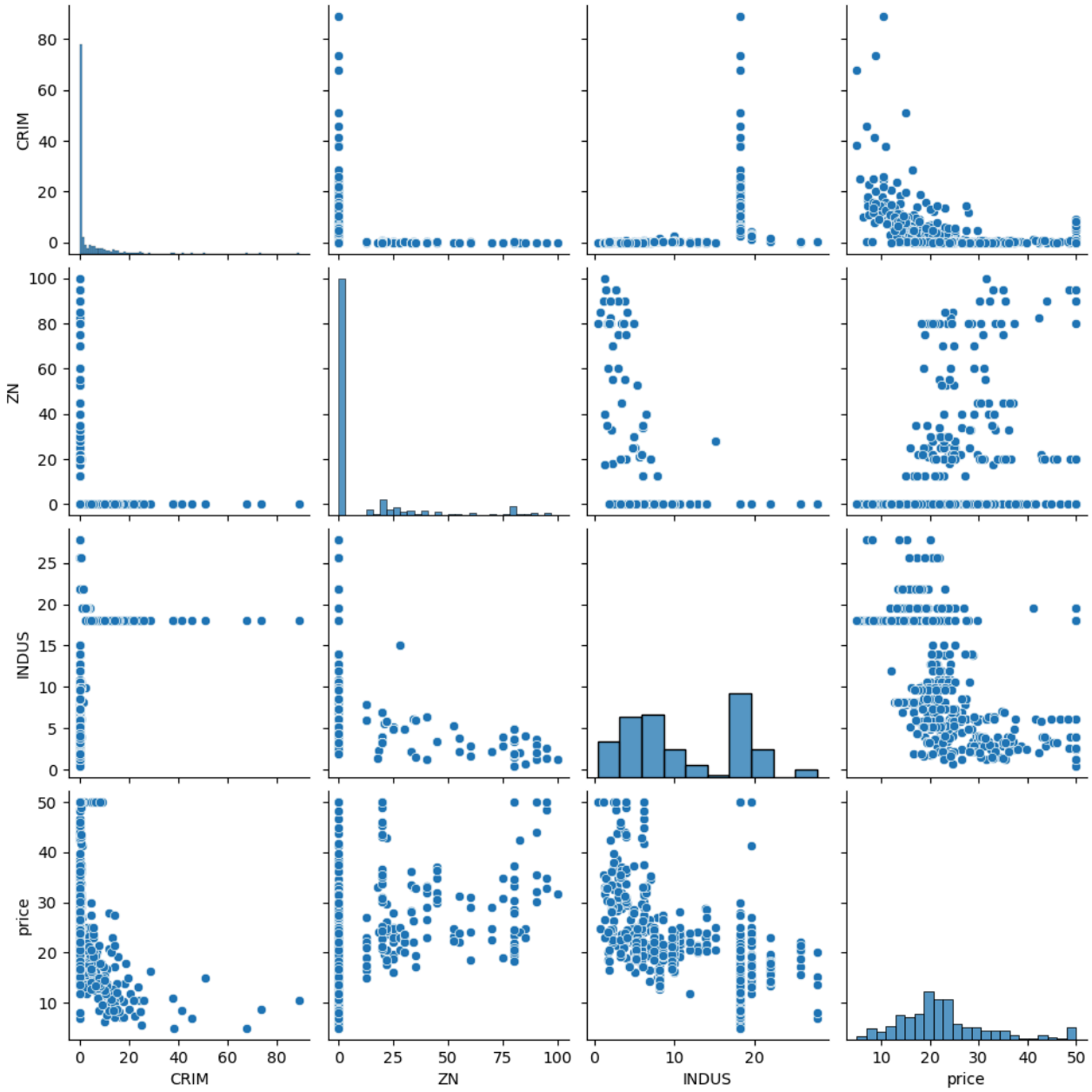
Out[22]: <seaborn.axisgrid.PairGrid at 0x117f7e12350>



In [23]:
```
sns.pairplot(data[["CRIM","ZN",'INDUS',"price"]])
```

```
C:\Users\alisu\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

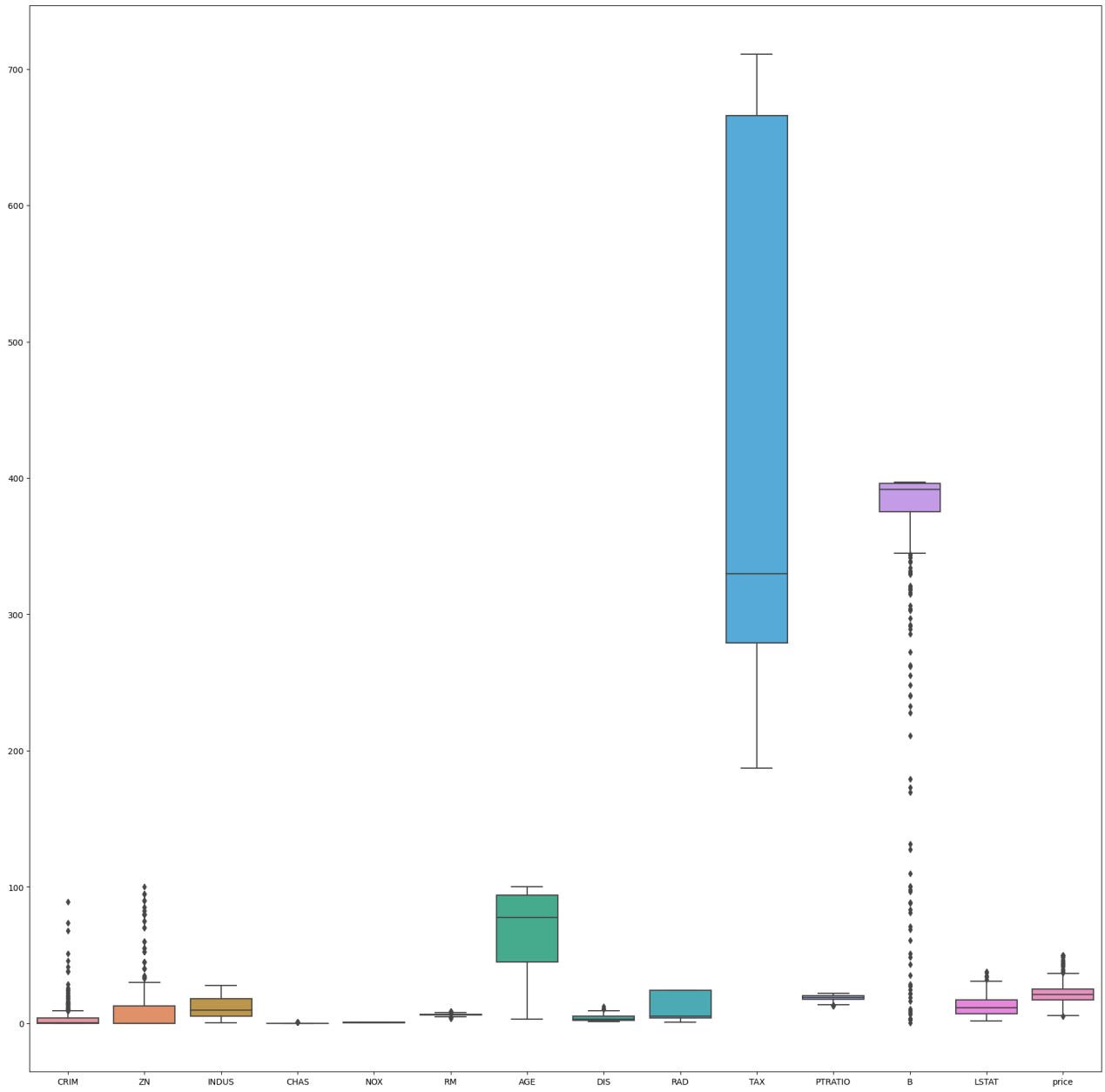
Out[23]:
```
<seaborn.axisgrid.PairGrid at 0x11785b6a350>
```



OUTLIERS FINDINGS

In [24]:
```
plt.figure(figsize=(23,23))
sns.boxplot(data)
```

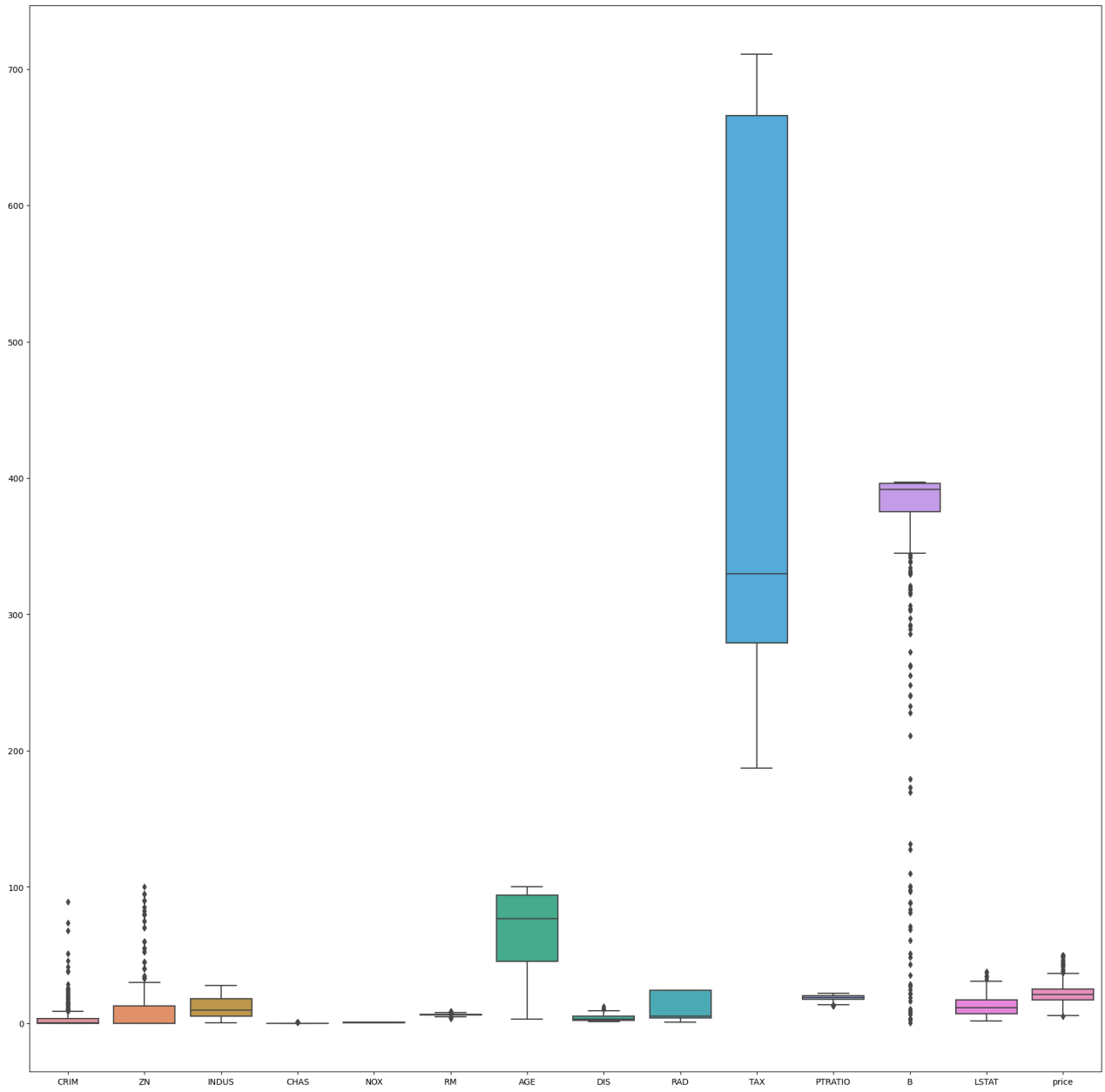
Out[24]: <Axes: >



Outliers finding

In [98]:
```
plt.figure(figsize=(23,23))
sns.boxplot(data)
```

Out[98]: <Axes: >



In [25]:
```
data.describe()
```

Out[25]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000

Use of linear regression model

Divide dataset in to input features as x and output features as y

In [26]:
```
x=data.iloc[:,:-1]
```

In [27]:
```
x
```

Out[27]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33
...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88

506 rows × 13 columns

In [28]:
```
y=data.iloc[:,-1]
```

In [29]:
```
y
```

Out[29]:
```
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
       ...
501    22.4
502    20.6
503    23.9
504    22.0
505    11.9
Name: price, Length: 506, dtype: float64
```

Split the data astraing and testing using sklearn librarys method train_test_split

In [30]:
```
from sklearn.model_selection  import  train_test_split
```

In [31]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=1)
```

In [32]:
```
xtrain
```

Out[32]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
42	0.14150	0.0	6.91	0	0.448	6.169	6.6	5.7209	3	233.0	17.9	383.37	5.81
58	0.15445	25.0	5.13	0	0.453	6.145	29.2	7.8148	8	284.0	19.7	390.68	6.86
385	16.81180	0.0	18.10	0	0.700	5.277	98.1	1.4261	24	666.0	20.2	396.90	30.81
78	0.05646	0.0	12.83	0	0.437	6.232	53.7	5.0141	5	398.0	18.7	386.40	12.34
424	8.79212	0.0	18.10	0	0.584	5.565	70.6	2.0635	24	666.0	20.2	3.65	17.16
...	...	...	...	...	...	...	...	...	...	...	...	...	...
255	0.03548	80.0	3.64	0	0.392	5.876	19.1	9.2203	1	315.0	16.4	395.18	9.25
72	0.09164	0.0	10.81	0	0.413	6.065	7.8	5.2873	4	305.0	19.2	390.91	5.52
396	5.87205	0.0	18.10	0	0.693	6.405	96.0	1.6768	24	666.0	20.2	396.90	19.37
235	0.33045	0.0	6.20	0	0.507	6.086	61.5	3.6519	8	307.0	17.4	376.75	10.88
37	0.08014	0.0	5.96	0	0.499	5.850	41.5	3.9342	5	279.0	19.2	396.90	8.77

404 rows × 13 columns

In [33]:
```
xtest
```

Out[33]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
307	0.04932	33.0	2.18	0	0.472	6.849	70.3	3.1827	7	222.0	18.4	396.90	7.53
343	0.02543	55.0	3.78	0	0.484	6.696	56.4	5.7321	5	370.0	17.6	396.90	7.18
47	0.22927	0.0	6.91	0	0.448	6.030	85.5	5.6894	3	233.0	17.9	392.74	18.80
67	0.05789	12.5	6.07	0	0.409	5.878	21.4	6.4980	4	345.0	18.9	396.21	8.10
362	3.67822	0.0	18.10	0	0.770	5.362	96.2	2.1036	24	666.0	20.2	380.79	10.19
...	...	...	...	...	...	...	...	...	...	...	...	...	...
92	0.04203	28.0	15.04	0	0.464	6.442	53.6	3.6659	4	270.0	18.2	395.01	8.16
224	0.31533	0.0	6.20	0	0.504	8.266	78.3	2.8944	8	307.0	17.4	385.05	4.14
110	0.10793	0.0	8.56	0	0.520	6.195	54.4	2.7778	5	384.0	20.9	393.49	13.00
426	12.24720	0.0	18.10	0	0.584	5.837	59.7	1.9976	24	666.0	20.2	24.65	15.69
443	9.96654	0.0	18.10	0	0.740	6.485	100.0	1.9784	24	666.0	20.2	386.73	18.85

102 rows × 13 columns

In [34]:
```
ytrain
```

Out[34]:
```
42     25.3
58     23.3
385     7.2
78     21.2
424    11.7
       ...
255    20.9
72     22.8
396    12.5
235    24.0
37     21.0
Name: price, Length: 404, dtype: float64
```

In [35]:
```
ytest
```

Out[35]:
```
307    28.2
343    23.9
47     16.6
67     22.0
362    20.8
       ...
92     22.9
224    44.8
110    21.7
426    10.2
443    15.4
Name: price, Length: 102, dtype: float64
```

In [36]:
```
len(ytest)
```

Out[36]:
```
102
```

In [37]:
```
ytest.shape
```

Out[37]:
```
(102,)
```

Model selection Linear Regression

In [38]:
```
from sklearn.linear_model import LinearRegression
```

In [40]:
```
model =LinearRegression()
```

In [41]:
```
model.fit(xtrain,ytrain)
```

Out[41]:
```
▾ LinearRegression
LinearRegression()
```

In [42]:
```
predict=model.predict(xtest)
```

In [43]:
```
predict
```

Out[43]:
```
array([32.65503184, 28.0934953 , 18.02901829, 21.47671576, 18.8254387 ,
       19.87997758, 32.42014863, 18.06597765, 24.42277848, 27.00977832,
       27.04081017, 28.75196794, 21.15677699, 26.85200196, 23.38835945,
       20.66241266, 17.33082198, 38.24813601, 30.50550873,  8.74436733,
       20.80203902, 16.26328126, 25.21805656, 24.85175752, 31.384365  ,
       10.71311063, 13.80434635, 16.65930389, 36.52625779, 14.66750528,
       21.12114902, 13.95558618, 43.16210242, 17.97539649, 21.80116017,
       20.58294808, 17.59938821, 27.2212319 ,  9.46139365, 19.82963781,
       24.30751863, 21.18528812, 29.57235682, 16.3431752 , 19.31483171,
       14.56343172, 39.20885479, 18.10887551, 25.91223267, 20.33018802,
       25.16282007, 24.42921237, 25.07123258, 26.6603279 ,  4.56151258,
       24.0818735 , 10.88682673, 26.88926656, 16.85598381, 35.88704363,
       19.55733853, 27.51928921, 16.58436103, 18.77551029, 11.13872875,
       32.36392607, 36.72833773, 21.95924582, 24.57949647, 25.14868695,
       23.42841301,  6.90732017, 16.56298149, 20.41940517, 20.80403418,
       21.54219598, 33.85383463, 27.94645899, 25.17281456, 34.65883942,
       18.62487738, 23.97375565, 34.6419296 , 13.34754896, 20.71097982,
       30.0803549 , 17.13421671, 24.30528434, 19.25576671, 16.98006722,
       27.00622638, 41.85509074, 14.11131512, 23.25736073, 14.66302672,
       21.86977175, 23.02527624, 29.0899182 , 37.11937872, 20.53271022,
       17.36840034, 17.71399314])
```

In [44]:
```
len(predict)
```

Out[44]:
```
102
```

In [45]:
```
ytest
```

Out[45]:
```
307    28.2
343    23.9
47     16.6
67     22.0
362    20.8
       ...
92     22.9
224    44.8
110    21.7
426    10.2
443    15.4
Name: price, Length: 102, dtype: float64
```

In [46]:
```
xtest
```

Out[46]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
307	0.04932	33.0	2.18	0	0.472	6.849	70.3	3.1827	7	222.0	18.4	396.90	7.53
343	0.02543	55.0	3.78	0	0.484	6.696	56.4	5.7321	5	370.0	17.6	396.90	7.18
47	0.22927	0.0	6.91	0	0.448	6.030	85.5	5.6894	3	233.0	17.9	392.74	18.80
67	0.05789	12.5	6.07	0	0.409	5.878	21.4	6.4980	4	345.0	18.9	396.21	8.10
362	3.67822	0.0	18.10	0	0.770	5.362	96.2	2.1036	24	666.0	20.2	380.79	10.19
...	...	...	...	...	...	...	...	...	...	...	...	...	...
92	0.04203	28.0	15.04	0	0.464	6.442	53.6	3.6659	4	270.0	18.2	395.01	8.16
224	0.31533	0.0	6.20	0	0.504	8.266	78.3	2.8944	8	307.0	17.4	385.05	4.14
110	0.10793	0.0	8.56	0	0.520	6.195	54.4	2.7778	5	384.0	20.9	393.49	13.00
426	12.24720	0.0	18.10	0	0.584	5.837	59.7	1.9976	24	666.0	20.2	24.65	15.69
443	9.96654	0.0	18.10	0	0.740	6.485	100.0	1.9784	24	666.0	20.2	386.73	18.85

102 rows × 13 columns

In [47]:
```
model.coef_
```

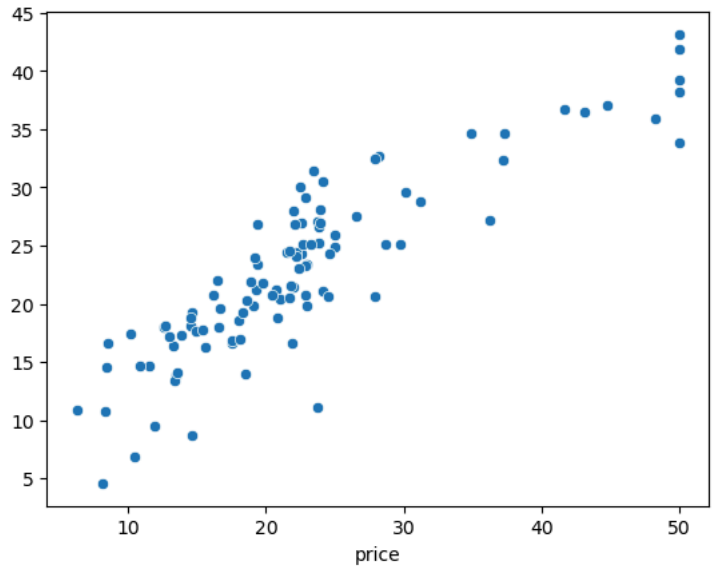
Out[47]:
```
array([-1.12386867e-01,  5.80587074e-02,  1.83593559e-02,  2.12997760e+00,
       -1.95811012e+01,  3.09546166e+00,  4.45265228e-03, -1.50047624e+00,
        3.05358969e-01, -1.11230879e-02, -9.89007562e-01,  7.32130017e-03,
       -5.44644997e-01])
```

In [48]:
```
model.intercept_
```

Out[48]:
```
42.93352585337695
```

In [49]:
```
sns.scatterplot(x=ytest,y=predict)
```

Out[49]:
```
<Axes: xlabel='price'>
```



In [50]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error
```

In [51]:
```
mean_absolute_error(ytest, predict)
```

Out[51]:
```
3.750712180838908
```

In [53]:
```
mean_squared_error(ytest, predict)
```

Out[53]:
```
23.380836480270144
```

In [ ]:
```

```