untitled14

May 3, 2024

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

[2]:
```
data=pd.read_csv("C:\\Users\\nayan\\Downloads\\HousingData.csv")
data
```

[2]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	NaN	36.2
..	…	…	…	…	…	…	…	…	…	…	…	…	…	…
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	NaN	22.4
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.90	7.88	11.9

[506 rows x 14 columns]

[3]:
```
data.dropna(how="any", inplace=True)
```

[4]:
```
data.isnull().sum()
```

[4]:
```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NOX        0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
MEDV       0
dtype: int64
```

[5]:
```
data.columns
```

[5]:
```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT', 'MEDV'],
      dtype='object')
```

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

[7]:
```
x
```

[7]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	\
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3	222	
..	…	…	…	…	…	…	…	…	…	…	
499	0.17783	0.0	9.69	0.0	0.585	5.569	73.5	2.3999	6	391	
500	0.22438	0.0	9.69	0.0	0.585	6.027	79.7	2.4982	6	391	
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	

	PTRATIO	B	LSTAT
0	15.3	396.90	4.98
1	17.8	396.90	9.14

```
2       17.8  392.83   4.03
3       18.7  394.63   2.94
5       18.7  394.12   5.21
..       …     …     …
499     19.2  395.77  15.10
500     19.2  396.90  14.33
502     21.0  396.90   9.08
503     21.0  396.90   5.64
504     21.0  393.45   6.48

[394 rows x 13 columns]
```

[8]:
```
y=data['MEDV']
```

[9]:
```
y
```

[9]:
```
0      24.0
1      21.6
2      34.7
3      33.4
5      28.7
       …
499    17.5
500    16.8
502    20.6
503    23.9
504    22.0
Name: MEDV, Length: 394, dtype: float64
```

[10]:
```
## Spliting train and test data using scalar
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.25,␣
 ↪random_state=42)
```

[11]:
```
model = LinearRegression()
model.fit(x_train, y_train)
```

[11]:
```
LinearRegression()
```

[12]:
```
#predict
y_pred=model.predict(x_test)
```

[13]:
```
model.score(x_train, y_train)
```

[13]:
```
0.7894375551691273
```

[14]:
```
model.score(x_test, y_test)
```

[14]: 0.6830391070219286

[15]:
```
np.sqrt(mean_squared_error(y_test, y_pred))
```

[15]: 5.4572694398436115

[ ]:
```

```