DL Assignment : 1

Name : Shrikrushna Zirape

Roll No : 41283 (BE-2)

Problem Statement : Linear Regression by using Deep Neural Network : Implement boston housing price prediction problem by Linear regression using Deep Neural network. Use Boston House price prediction dataset.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

Load Dataset

In [2]:
```
df = pd.read_csv('data/Boston.csv')
df.head(10)
```

Out[2]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	CAT. MEDV	Unnamed: 15	Unnamed: 16
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0	0	NaN	NaN
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6	0	NaN	NaN
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7	1	NaN	NaN
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4	1	NaN	NaN
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2	1	NaN	NaN
5	0.02985	0.0	2.18	0	0.458	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.7	0	NaN	NaN
6	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9	0	NaN	NaN
7	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.90	19.15	27.1	0	NaN	NaN
8	0.21124	12.5	7.87	0	0.524	5.631	100.0	6.0821	5	311	15.2	386.63	29.93	16.5	0	NaN	NaN
9	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.10	18.9	0	NaN	NaN

In [3]:
```
df.drop(columns=['Unnamed: 15','Unnamed: 16'],inplace=True)
```

In [4]:
```
df.drop(columns=['CAT. MEDV'],inplace=True)
```

Checking for null values

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
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
```

In [6]:
```
df.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    int64  
 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(11), int64(3)
memory usage: 55.5 KB
```

In [7]:
```
df.describe()
```

Out[7]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
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	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	12.653063	22.532806
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	7.141062	9.197104
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	1.730000	5.000000
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	6.950000	17.025000
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	11.360000	21.200000
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	16.955000	25.000000
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	37.970000	50.000000

Checking correlation with target variable MEDV

In [8]:
```
df.corr()['MEDV'].sort_values()
```

Out[8]:
```
LSTAT     -0.737663
PTRATIO   -0.507787
INDUS     -0.483725
TAX       -0.468536
NOX       -0.427321
CRIM      -0.388305
RAD       -0.381626
AGE       -0.376955
CHAS       0.175260
DIS        0.249929
B          0.333461
ZN         0.360445
RM         0.695360
MEDV       1.000000
Name: MEDV, dtype: float64
```

In [9]:
```
X = df.loc[:,['LSTAT','PTRATIO','RM']]
Y = df.loc[:,"MEDV"]
X.shape,Y.shape
```

Out[9]:
```
((506, 3), (506,))
```

Preparing training and testing data set

In [10]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size=0.25,random_state=10)
```

Normalizing training and testing dataset

In [11]:
```
from sklearn.preprocessing import StandardScaler
```

In [12]:
```
scaler = StandardScaler()
```

In [13]:
```
scaler.fit(x_train)
```

Out[13]:
```
StandardScaler()
```

In [14]:
```
x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)
```

Preparing model

In [15]:
```
from keras.models import Sequential
from keras.layers import Dense
```

In [16]:
```
model = Sequential()
```

In [17]:
```
model.add(Dense(128,input_shape=(3,),activation='relu',name='input'))
model.add(Dense(64,activation='relu',name='layer_1'))
model.add(Dense(1,activation='linear',name='output'))
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input (Dense)               (None, 128)               512

 layer_1 (Dense)             (None, 64)                8256

 output (Dense)              (None, 1)                 65

=================================================================
Total params: 8,833
Trainable params: 8,833
Non-trainable params: 0
_________________________________________________________________
```

In [18]:

```
model.fit(x_train,y_train,epochs=100,validation_split=0.05)
```

```
Epoch 1/100
12/12 [==============================] - 2s 53ms/step - loss: 539.2911 - mae: 21.6028 - val_loss: 708.5662 - val_mae: 23.9646
Epoch 2/100
12/12 [==============================] - 0s 13ms/step - loss: 506.3215 - mae: 20.8718 - val_loss: 671.6750 - val_mae: 23.1712
Epoch 3/100
12/12 [==============================] - 0s 11ms/step - loss: 466.7391 - mae: 19.9747 - val_loss: 622.3312 - val_mae: 22.0846
Epoch 4/100
12/12 [==============================] - 0s 9ms/step - loss: 413.6024 - mae: 18.6774 - val_loss: 553.8527 - val_mae: 20.5076
Epoch 5/100
12/12 [==============================] - 0s 11ms/step - loss: 340.0710 - mae: 16.8183 - val_loss: 467.4739 - val_mae: 18.4865
Epoch 6/100
12/12 [==============================] - 0s 12ms/step - loss: 254.0955 - mae: 14.3808 - val_loss: 367.5447 - val_mae: 16.0270
Epoch 7/100
12/12 [==============================] - 0s 9ms/step - loss: 163.4490 - mae: 11.4762 - val_loss: 269.1692 - val_mae: 13.0181
Epoch 8/100
12/12 [==============================] - 0s 11ms/step - loss: 89.2134 - mae: 8.2882 - val_loss: 189.9538 - val_mae: 10.2955
Epoch 9/100
12/12 [==============================] - 0s 10ms/step - loss: 50.0832 - mae: 5.8525 - val_loss: 143.4507 - val_mae: 8.5330
Epoch 10/100
12/12 [==============================] - 0s 10ms/step - loss: 39.4145 - mae: 4.9155 - val_loss: 120.4480 - val_mae: 7.6250
Epoch 11/100
12/12 [==============================] - 0s 10ms/step - loss: 34.2234 - mae: 4.4500 - val_loss: 109.6613 - val_mae: 7.3691
Epoch 12/100
12/12 [==============================] - 0s 11ms/step - loss: 29.7919 - mae: 4.0989 - val_loss: 104.3488 - val_mae: 7.2210
Epoch 13/100
12/12 [==============================] - 0s 10ms/step - loss: 27.2405 - mae: 3.8908 - val_loss: 99.0083 - val_mae: 7.0297
Epoch 14/100
12/12 [==============================] - 0s 13ms/step - loss: 25.6894 - mae: 3.7644 - val_loss: 94.3940 - val_mae: 6.8878
Epoch 15/100
12/12 [==============================] - 0s 9ms/step - loss: 24.4743 - mae: 3.6621 - val_loss: 91.3494 - val_mae: 6.7957
Epoch 16/100
12/12 [==============================] - 0s 13ms/step - loss: 23.6394 - mae: 3.5910 - val_loss: 90.5867 - val_mae: 6.7428
Epoch 17/100
12/12 [==============================] - 0s 11ms/step - loss: 22.8061 - mae: 3.5299 - val_loss: 89.8672 - val_mae: 6.7059
Epoch 18/100
12/12 [==============================] - 0s 11ms/step - loss: 22.1649 - mae: 3.4753 - val_loss: 88.9616 - val_mae: 6.6698
Epoch 19/100
12/12 [==============================] - 0s 11ms/step - loss: 21.4995 - mae: 3.4112 - val_loss: 88.1913 - val_mae: 6.6296
Epoch 20/100
12/12 [==============================] - 0s 9ms/step - loss: 20.9803 - mae: 3.3719 - val_loss: 87.7182 - val_mae: 6.5633
Epoch 21/100
12/12 [==============================] - 0s 10ms/step - loss: 20.4479 - mae: 3.3286 - val_loss: 86.6075 - val_mae: 6.4872
Epoch 22/100
12/12 [==============================] - 0s 9ms/step - loss: 19.9252 - mae: 3.2822 - val_loss: 85.9076 - val_mae: 6.4443
Epoch 23/100
12/12 [==============================] - 0s 10ms/step - loss: 19.5059 - mae: 3.2464 - val_loss: 85.0506 - val_mae: 6.3660
Epoch 24/100
12/12 [==============================] - 0s 11ms/step - loss: 19.1466 - mae: 3.2179 - val_loss: 84.7840 - val_mae: 6.3247
Epoch 25/100
12/12 [==============================] - 0s 9ms/step - loss: 18.7148 - mae: 3.1727 - val_loss: 84.2077 - val_mae: 6.2782
Epoch 26/100
12/12 [==============================] - 0s 9ms/step - loss: 18.3745 - mae: 3.1370 - val_loss: 84.6847 - val_mae: 6.2667
Epoch 27/100
12/12 [==============================] - 0s 11ms/step - loss: 18.0907 - mae: 3.1021 - val_loss: 85.2562 - val_mae: 6.2282
Epoch 28/100
12/12 [==============================] - 0s 12ms/step - loss: 17.7178 - mae: 3.0764 - val_loss: 84.2055 - val_mae: 6.1324
Epoch 29/100
12/12 [==============================] - 0s 10ms/step - loss: 17.4711 - mae: 3.0584 - val_loss: 83.9480 - val_mae: 6.0800
Epoch 30/100
12/12 [==============================] - 0s 10ms/step - loss: 17.1969 - mae: 3.0316 - val_loss: 81.6525 - val_mae: 5.9682
Epoch 31/100
12/12 [==============================] - 0s 9ms/step - loss: 16.9258 - mae: 3.0162 - val_loss: 81.8426 - val_mae: 5.9305
Epoch 32/100
12/12 [==============================] - 0s 9ms/step - loss: 16.7039 - mae: 2.9899 - val_loss: 82.5370 - val_mae: 5.9237
Epoch 33/100
12/12 [==============================] - 0s 11ms/step - loss: 16.4305 - mae: 2.9622 - val_loss: 82.4255 - val_mae: 5.8946
Epoch 34/100
12/12 [==============================] - 0s 9ms/step - loss: 16.2343 - mae: 2.9363 - val_loss: 82.7476 - val_mae: 5.8871
Epoch 35/100
12/12 [==============================] - 0s 9ms/step - loss: 16.0647 - mae: 2.9212 - val_loss: 82.2225 - val_mae: 5.8660
Epoch 36/100
12/12 [==============================] - 0s 10ms/step - loss: 15.8816 - mae: 2.9065 - val_loss: 81.7927 - val_mae: 5.8127
Epoch 37/100
12/12 [==============================] - 0s 9ms/step - loss: 15.6419 - mae: 2.8874 - val_loss: 81.5825 - val_mae: 5.7658
Epoch 38/100
12/12 [==============================] - 0s 9ms/step - loss: 15.5342 - mae: 2.8722 - val_loss: 81.2292 - val_mae: 5.7214
Epoch 39/100
12/12 [==============================] - 0s 11ms/step - loss: 15.2873 - mae: 2.8372 - val_loss: 83.1768 - val_mae: 5.7727
Epoch 40/100
12/12 [==============================] - 0s 9ms/step - loss: 15.2641 - mae: 2.8294 - val_loss: 83.0453 - val_mae: 5.7248
Epoch 41/100
12/12 [==============================] - 0s 9ms/step - loss: 15.1101 - mae: 2.8215 - val_loss: 82.4333 - val_mae: 5.6811
Epoch 42/100
12/12 [==============================] - 0s 10ms/step - loss: 14.9924 - mae: 2.8059 - val_loss: 82.4092 - val_mae: 5.6586
Epoch 43/100
12/12 [==============================] - 0s 9ms/step - loss: 14.8041 - mae: 2.7851 - val_loss: 81.6021 - val_mae: 5.6425
Epoch 44/100
12/12 [==============================] - 0s 11ms/step - loss: 14.6942 - mae: 2.7814 - val_loss: 80.0983 - val_mae: 5.5817
Epoch 45/100
12/12 [==============================] - 0s 9ms/step - loss: 14.6107 - mae: 2.7832 - val_loss: 80.3339 - val_mae: 5.5714
Epoch 46/100
12/12 [==============================] - 0s 9ms/step - loss: 14.4966 - mae: 2.7611 - val_loss: 83.5054 - val_mae: 5.6547
Epoch 47/100
12/12 [==============================] - 0s 9ms/step - loss: 14.4279 - mae: 2.7324 - val_loss: 80.8443 - val_mae: 5.5263
Epoch 48/100
12/12 [==============================] - 0s 9ms/step - loss: 14.3395 - mae: 2.7211 - val_loss: 79.9668 - val_mae: 5.4820
Epoch 49/100
12/12 [==============================] - 0s 9ms/step - loss: 14.3885 - mae: 2.7432 - val_loss: 80.5138 - val_mae: 5.4968
Epoch 50/100
12/12 [==============================] - 0s 10ms/step - loss: 14.2106 - mae: 2.7309 - val_loss: 83.1156 - val_mae: 5.5821
```

```
Epoch 51/100
12/12 [==============================] - 0s 10ms/step - loss: 14.1643 - mae: 2.6981 - val_loss: 79.4093 - val_mae: 5.4336
Epoch 52/100
12/12 [==============================] - 0s 9ms/step - loss: 13.8945 - mae: 2.6752 - val_loss: 80.6351 - val_mae: 5.4538
Epoch 53/100
12/12 [==============================] - 0s 11ms/step - loss: 13.7842 - mae: 2.6697 - val_loss: 80.7124 - val_mae: 5.4556
Epoch 54/100
12/12 [==============================] - 0s 11ms/step - loss: 13.6417 - mae: 2.6617 - val_loss: 79.1203 - val_mae: 5.4045
Epoch 55/100
12/12 [==============================] - 0s 14ms/step - loss: 13.5537 - mae: 2.6674 - val_loss: 78.9995 - val_mae: 5.4098
Epoch 56/100
12/12 [==============================] - 0s 10ms/step - loss: 13.5146 - mae: 2.6554 - val_loss: 79.5111 - val_mae: 5.4014
Epoch 57/100
12/12 [==============================] - 0s 10ms/step - loss: 13.3217 - mae: 2.6299 - val_loss: 80.4437 - val_mae: 5.3966
Epoch 58/100
12/12 [==============================] - 0s 14ms/step - loss: 13.3051 - mae: 2.6247 - val_loss: 80.7450 - val_mae: 5.3654
Epoch 59/100
12/12 [==============================] - 0s 11ms/step - loss: 13.2311 - mae: 2.6169 - val_loss: 78.4653 - val_mae: 5.2992
Epoch 60/100
12/12 [==============================] - 0s 11ms/step - loss: 13.1291 - mae: 2.6000 - val_loss: 80.9283 - val_mae: 5.3706
Epoch 61/100
12/12 [==============================] - 0s 10ms/step - loss: 13.0746 - mae: 2.6052 - val_loss: 80.2841 - val_mae: 5.3396
Epoch 62/100
12/12 [==============================] - 0s 9ms/step - loss: 12.9002 - mae: 2.5799 - val_loss: 79.4223 - val_mae: 5.2832
Epoch 63/100
12/12 [==============================] - 0s 10ms/step - loss: 12.8149 - mae: 2.5758 - val_loss: 79.8036 - val_mae: 5.2975
Epoch 64/100
12/12 [==============================] - 0s 9ms/step - loss: 12.6747 - mae: 2.5782 - val_loss: 77.6885 - val_mae: 5.3033
Epoch 65/100
12/12 [==============================] - 0s 9ms/step - loss: 12.7076 - mae: 2.6089 - val_loss: 79.0439 - val_mae: 5.3087
Epoch 66/100
12/12 [==============================] - 0s 9ms/step - loss: 12.5774 - mae: 2.5850 - val_loss: 80.4104 - val_mae: 5.3045
Epoch 67/100
12/12 [==============================] - 0s 9ms/step - loss: 12.4626 - mae: 2.5475 - val_loss: 80.0999 - val_mae: 5.2504
Epoch 68/100
12/12 [==============================] - 0s 10ms/step - loss: 12.3070 - mae: 2.5222 - val_loss: 80.7591 - val_mae: 5.2228
Epoch 69/100
12/12 [==============================] - 0s 11ms/step - loss: 12.2507 - mae: 2.5117 - val_loss: 79.9018 - val_mae: 5.1961
Epoch 70/100
12/12 [==============================] - 0s 9ms/step - loss: 12.1783 - mae: 2.5151 - val_loss: 81.2037 - val_mae: 5.2096
Epoch 71/100
12/12 [==============================] - 0s 11ms/step - loss: 12.1537 - mae: 2.4968 - val_loss: 78.6277 - val_mae: 5.1478
Epoch 72/100
12/12 [==============================] - 0s 11ms/step - loss: 11.9795 - mae: 2.4785 - val_loss: 81.7577 - val_mae: 5.2033
Epoch 73/100
12/12 [==============================] - 0s 9ms/step - loss: 11.9696 - mae: 2.5019 - val_loss: 81.8564 - val_mae: 5.1975
Epoch 74/100
12/12 [==============================] - 0s 9ms/step - loss: 11.8518 - mae: 2.5015 - val_loss: 80.2826 - val_mae: 5.1808
Epoch 75/100
12/12 [==============================] - 0s 10ms/step - loss: 11.7755 - mae: 2.4761 - val_loss: 80.4577 - val_mae: 5.1574
Epoch 76/100
12/12 [==============================] - 0s 9ms/step - loss: 11.7356 - mae: 2.4734 - val_loss: 81.1006 - val_mae: 5.1760
Epoch 77/100
12/12 [==============================] - 0s 9ms/step - loss: 11.7469 - mae: 2.4878 - val_loss: 79.5115 - val_mae: 5.1456
Epoch 78/100
12/12 [==============================] - 0s 10ms/step - loss: 11.5511 - mae: 2.4718 - val_loss: 82.7929 - val_mae: 5.2255
Epoch 79/100
12/12 [==============================] - 0s 10ms/step - loss: 11.6095 - mae: 2.4601 - val_loss: 81.9202 - val_mae: 5.1357
Epoch 80/100
12/12 [==============================] - 0s 10ms/step - loss: 11.4707 - mae: 2.4427 - val_loss: 80.8143 - val_mae: 5.1572
Epoch 81/100
12/12 [==============================] - 0s 10ms/step - loss: 11.3998 - mae: 2.4467 - val_loss: 78.7451 - val_mae: 5.0839
Epoch 82/100
12/12 [==============================] - 0s 9ms/step - loss: 11.3945 - mae: 2.4350 - val_loss: 81.6314 - val_mae: 5.1697
Epoch 83/100
12/12 [==============================] - 0s 9ms/step - loss: 11.2868 - mae: 2.4224 - val_loss: 80.8796 - val_mae: 5.1053
Epoch 84/100
12/12 [==============================] - 0s 9ms/step - loss: 11.2933 - mae: 2.4186 - val_loss: 80.8439 - val_mae: 5.0951
Epoch 85/100
12/12 [==============================] - 0s 9ms/step - loss: 11.3944 - mae: 2.4235 - val_loss: 81.1154 - val_mae: 5.0786
Epoch 86/100
12/12 [==============================] - 0s 9ms/step - loss: 11.3888 - mae: 2.4187 - val_loss: 83.2470 - val_mae: 5.2305
Epoch 87/100
12/12 [==============================] - 0s 10ms/step - loss: 11.0781 - mae: 2.4028 - val_loss: 80.0263 - val_mae: 5.0942
Epoch 88/100
12/12 [==============================] - 0s 9ms/step - loss: 11.1180 - mae: 2.4127 - val_loss: 80.8835 - val_mae: 5.1160
Epoch 89/100
12/12 [==============================] - 0s 9ms/step - loss: 11.0975 - mae: 2.3980 - val_loss: 83.3732 - val_mae: 5.1746
Epoch 90/100
12/12 [==============================] - 0s 9ms/step - loss: 11.0572 - mae: 2.3984 - val_loss: 81.9773 - val_mae: 5.1214
Epoch 91/100
12/12 [==============================] - 0s 9ms/step - loss: 11.0261 - mae: 2.4112 - val_loss: 81.8094 - val_mae: 5.1724
Epoch 92/100
12/12 [==============================] - 0s 9ms/step - loss: 10.9577 - mae: 2.4006 - val_loss: 82.4225 - val_mae: 5.1415
Epoch 93/100
12/12 [==============================] - 0s 9ms/step - loss: 10.9168 - mae: 2.3853 - val_loss: 83.1790 - val_mae: 5.1393
Epoch 94/100
12/12 [==============================] - 0s 9ms/step - loss: 10.9348 - mae: 2.3927 - val_loss: 83.6329 - val_mae: 5.1397
Epoch 95/100
12/12 [==============================] - 0s 9ms/step - loss: 11.0107 - mae: 2.4054 - val_loss: 81.2756 - val_mae: 5.1034
Epoch 96/100
12/12 [==============================] - 0s 9ms/step - loss: 10.9627 - mae: 2.3961 - val_loss: 84.8841 - val_mae: 5.1991
Epoch 97/100
12/12 [==============================] - 0s 9ms/step - loss: 10.7220 - mae: 2.3633 - val_loss: 82.0974 - val_mae: 5.0658
Epoch 98/100
12/12 [==============================] - 0s 9ms/step - loss: 10.8073 - mae: 2.3776 - val_loss: 81.4088 - val_mae: 5.1130
Epoch 99/100
12/12 [==============================] - 0s 10ms/step - loss: 10.7377 - mae: 2.3667 - val_loss: 83.0202 - val_mae: 5.1248
Epoch 100/100
12/12 [==============================] - 0s 10ms/step - loss: 10.6293 - mae: 2.3574 - val_loss: 83.6140 - val_mae: 5.1036
```

Out[18]: <keras.callbacks.History at 0x2d2d819cd60>

In [23]:
```
# Make predictions on the test set
y_pred = model.predict(x_test)
```

```
4/4 [==============================] - 0s 4ms/step
```

In [25]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

In [26]:
```
# Calculate metrics
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [19]:
```
output = model.evaluate(x_test,y_test)
```

```
4/4 [==============================] - 0s 4ms/step - loss: 23.1760 - mae: 3.1732
```

In [27]:
```
# Print metrics
print('Mean Squared Error:', mse)
print('Mean Absolute Error:', mae)
print('R-squared:', r2)
```

```
Mean Squared Error: 23.175983362100858
Mean Absolute Error: 3.173217832009624
R-squared: 0.7675142758666804
```

In [28]:
```
print(f"Mean Squared Error: {output[0]}"
      ,f"Mean Absolute Error: {output[1]}",sep="\n")
```

```
Mean Squared Error: 23.175983428955078
Mean Absolute Error: 3.1732177734375
```

In [29]:
```
y_pred = model.predict(x=x_test)
```

```
4/4 [==============================] - 0s 2ms/step
```

In [ ]:
```
print(*zip(y_pred,y_test))
```

In [ ]: