In [2]:

```
# Import necessary libraries
import numpy as np  # For numerical operations
import pandas as pd  # For handling datasets
from sklearn.model_selection import train_test_split  # Splitting data into train & test sets
from sklearn.linear_model import LinearRegression  # Linear Regression Model
from sklearn.preprocessing import StandardScaler  # Standardization of data
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score  # Evaluation metrics
```

```
C:\Users\OM\AppData\Roaming\Python\Python311\site-packages\pandas\core\arrays\masked.py:60: UserWarning: Pandas requires version
'1.3.6' or newer of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (
```

In [3]:

```
# Importing Keras (for Neural Network)
import keras
from keras.models import Sequential  # To define a sequential model
from keras.layers import Dense  # Fully connected layers
```

In [4]:

```
boston = pd.read_csv("boston_house_prices.csv")  # Reads CSV file into a DataFrame
# Selecting Features and Target
# Selecting 3 input features:
# 1. LSTAT (Percentage of lower status population)
# 2. RM (Average number of rooms per dwelling)
# 3. PTRATIO (Pupil-teacher ratio by town)
X = boston[['LSTAT', 'RM', 'PTRATIO']]
# Target variable: House Price
y = boston['PRICE']
# Splitting the Dataset into Training and Testing Sets
# 80% of data used for training, 20% for testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)
```

In [11]:

```
# Standardizing the Dataset (Feature Scaling)
# Standardization improves model performance by normalizing feature values
scaler = StandardScaler()  # Initializing StandardScaler
X_train_scaled = scaler.fit_transform(X_train)  # Fit and transform training data
X_test_scaled = scaler.transform(X_test)  # Transform test data using the same scaler
# Linear Regression Model
lr_model = LinearRegression()  # Initializing Linear Regression Model
lr_model.fit(X_train_scaled, y_train)  # Training the model using scaled training data
# Predicting house prices on test data
y_pred_lr = lr_model.predict(X_test_scaled)
# Evaluating Linear Regression Model
mse_lr = mean_squared_error(y_test, y_pred_lr)  # Mean Squared Error
mae_lr = mean_absolute_error(y_test, y_pred_lr)  # Mean Absolute Error
r2_lr = r2_score(y_test, y_pred_lr)  # R² Score (Model accuracy measure)
# Displaying evaluation metrics
print("Linear Regression Model Evaluation:")
print(f"Mean Squared Error: {mse_lr}")
print(f"Mean Absolute Error: {mae_lr}")
print(f"R2 Score: {r2_lr}")
# Neural Network (ANN) Model
# Creating a Deep Learning Model using Keras Sequential API
model = Sequential([
Dense(128, activation='relu', input_dim=3),  # Input layer (3 features) & first hidden layer (128 neurons)
Dense(64, activation='relu'),  # Second hidden layer with 64 neurons
Dense(32, activation='relu'),  # Third hidden layer with 32 neurons
Dense(16, activation='relu'),  # Fourth hidden layer with 16 neurons
Dense(1)  # Output layer (Predicting a single value - House Price)
])
# Compiling the model
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
# Optimizer: Adam (Adaptive Learning Rate Optimization)
# Loss function: Mean Squared Error (MSE) - Suitable for regression problems
# Metric: Mean Absolute Error (MAE) - Helps measure performance
```

```
Linear Regression Model Evaluation:
Mean Squared Error: 30.340105190234596
Mean Absolute Error: 3.5844321029226935
R2 Score: 0.6733732528519258
```

```
C:\Users\OM\AppData\Roaming\Python\Python311\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer
in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

In [12]:

```
# Training the Neural Network
history = model.fit(X_train_scaled, y_train, epochs=100, validation_split=0.05,
verbose=1)
# Training for 100 epochs
# Using 5% of training data as validation set to monitor overfitting
# `verbose=1` displays detailed training progress
#Epoch 1/100
#12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 26ms/step - loss: 547.8306 - #mae: 21.6359 - val_loss: 445.7750 - va
#Epoch 2/100
#12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 550.6208 - #mae: 21.6498 - val_loss: 403.5681 - val
#Epoch 3/100
#12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 433.7596 -
```

```
Epoch 1/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m12s [0m 86ms/step - loss: 543.6867 - mae: 21.7232 - val_loss: 426.0903 - val_mae: 1
Epoch 2/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 505.7161 - mae: 20.6915 - val_loss: 349.7185 - val_mae: 17
Epoch 3/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 393.6682 - mae: 18.1779 - val_loss: 208.3057 - val_mae: 13
Epoch 4/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 17ms/step - loss: 199.0245 - mae: 12.3191 - val_loss: 64.2445 - val_mae: 7.0
Epoch 5/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 58.2551 - mae: 6.0617 - val_loss: 48.5511 - val_mae: 4.923
Epoch 6/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 55.6602 - mae: 5.3771 - val_loss: 30.6176 - val_mae: 4.212
Epoch 7/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 41.0048 - mae: 4.6306 - val_loss: 24.5015 - val_mae: 3.751
Epoch 8/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 32.7700 - mae: 4.3268 - val_loss: 24.1418 - val_mae: 3.611
Epoch 9/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 34.6469 - mae: 4.2185 - val_loss: 20.9367 - val_mae: 3.507
Epoch 10/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 32.1898 - mae: 4.0009 - val_loss: 21.4170 - val_mae: 3.460
Epoch 11/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 30.5013 - mae: 4.0704 - val_loss: 19.4947 - val_mae: 3.328
Epoch 12/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 28.8264 - mae: 3.8318 - val_loss: 18.2770 - val_mae: 3.245
Epoch 13/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 26.6472 - mae: 3.8434 - val_loss: 18.2322 - val_mae: 3.198
Epoch 14/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 29.1395 - mae: 3.8823 - val_loss: 16.0675 - val_mae: 3.070
Epoch 15/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 24.6831 - mae: 3.5222 - val_loss: 16.0994 - val_mae: 3.018
Epoch 16/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 16ms/step - loss: 20.8788 - mae: 3.3663 - val_loss: 14.2841 - val_mae: 2.916
Epoch 17/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 22.3232 - mae: 3.4255 - val_loss: 15.3348 - val_mae: 2.957
Epoch 18/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m1s [0m 39ms/step - loss: 17.3776 - mae: 3.0663 - val_loss: 12.1229 - val_mae: 2.774
Epoch 19/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 19ms/step - loss: 24.4867 - mae: 3.4381 - val_loss: 14.1247 - val_mae: 2.850
Epoch 20/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 19ms/step - loss: 20.2957 - mae: 3.3028 - val_loss: 11.0734 - val_mae: 2.643
Epoch 21/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 24ms/step - loss: 21.5813 - mae: 3.1854 - val_loss: 12.2201 - val_mae: 2.658
Epoch 22/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 20ms/step - loss: 17.0770 - mae: 3.1312 - val_loss: 10.7931 - val_mae: 2.558
Epoch 23/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 21.4717 - mae: 3.2729 - val_loss: 11.2300 - val_mae: 2.612
Epoch 24/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m1s [0m 16ms/step - loss: 21.0031 - mae: 3.2462 - val_loss: 10.2366 - val_mae: 2.524
Epoch 25/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 19.5689 - mae: 3.0302 - val_loss: 10.3037 - val_mae: 2.546
Epoch 26/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 21.0289 - mae: 3.0690 - val_loss: 9.7249 - val_mae: 2.5349
Epoch 27/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 24.1059 - mae: 3.1886 - val_loss: 8.8359 - val_mae: 2.4088
Epoch 28/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 21.8863 - mae: 3.0241 - val_loss: 9.1360 - val_mae: 2.5164
Epoch 29/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 16.5263 - mae: 2.8124 - val_loss: 8.3687 - val_mae: 2.4316
Epoch 30/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 12.7324 - mae: 2.5649 - val_loss: 8.1404 - val_mae: 2.4198
Epoch 31/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 16.5486 - mae: 2.6935 - val_loss: 8.6430 - val_mae: 2.5278
Epoch 32/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 21.7684 - mae: 3.0349 - val_loss: 8.1820 - val_mae: 2.5039
Epoch 33/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 11.8373 - mae: 2.4046 - val_loss: 8.1994 - val_mae: 2.4883
Epoch 34/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 14.2739 - mae: 2.6427 - val_loss: 9.0508 - val_mae: 2.6023
Epoch 35/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 14.5830 - mae: 2.6371 - val_loss: 7.2982 - val_mae: 2.3821
Epoch 36/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 18ms/step - loss: 13.5659 - mae: 2.5972 - val_loss: 9.2852 - val_mae: 2.6430
Epoch 37/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 15.9864 - mae: 2.9435 - val_loss: 7.5030 - val_mae: 2.4333
Epoch 38/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 19.4981 - mae: 2.7825 - val_loss: 8.6291 - val_mae: 2.5327
Epoch 39/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 14.8476 - mae: 2.7120 - val_loss: 7.5730 - val_mae: 2.4164
Epoch 40/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 12.8046 - mae: 2.4625 - val_loss: 8.9290 - val_mae: 2.6225
Epoch 41/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 23.0360 - mae: 2.9777 - val_loss: 6.8944 - val_mae: 2.3022
Epoch 42/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 15.2815 - mae: 2.5849 - val_loss: 10.2947 - val_mae: 2.795
Epoch 43/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 19.0994 - mae: 2.7697 - val_loss: 6.7064 - val_mae: 2.2308
Epoch 44/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 13.8101 - mae: 2.6132 - val_loss: 8.6604 - val_mae: 2.5447
Epoch 45/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 23ms/step - loss: 19.0243 - mae: 2.8394 - val_loss: 7.4203 - val_mae: 2.4036
Epoch 46/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 16ms/step - loss: 15.2788 - mae: 2.6085 - val_loss: 7.8858 - val_mae: 2.4385
```

```
Epoch 47/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 13.3263 - mae: 2.4936 - val_loss: 8.7293 - val_mae: 2.4921
Epoch 48/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 17.7264 - mae: 2.8079 - val_loss: 6.5557 - val_mae: 2.1882
Epoch 49/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 21.8197 - mae: 2.8009 - val_loss: 8.0497 - val_mae: 2.4369
Epoch 50/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 11.0437 - mae: 2.4622 - val_loss: 8.2025 - val_mae: 2.4671
Epoch 51/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 14.8936 - mae: 2.6519 - val_loss: 7.4570 - val_mae: 2.3165
Epoch 52/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 14.8528 - mae: 2.5916 - val_loss: 7.9178 - val_mae: 2.4657
Epoch 53/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 13.0678 - mae: 2.4546 - val_loss: 8.1344 - val_mae: 2.4450
Epoch 54/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 11.8199 - mae: 2.4799 - val_loss: 9.3937 - val_mae: 2.5801
Epoch 55/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 19.5630 - mae: 2.8624 - val_loss: 6.6416 - val_mae: 2.1562
Epoch 56/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 15.5945 - mae: 2.5871 - val_loss: 10.6892 - val_mae: 2.733
Epoch 57/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 12.1568 - mae: 2.4836 - val_loss: 7.5099 - val_mae: 2.2729
Epoch 58/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 15.7663 - mae: 2.6998 - val_loss: 7.1794 - val_mae: 2.2729
Epoch 59/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 20.9249 - mae: 2.9092 - val_loss: 7.6805 - val_mae: 2.3824
Epoch 60/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 12.2927 - mae: 2.3870 - val_loss: 9.4301 - val_mae: 2.5326
Epoch 61/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 11.1233 - mae: 2.4303 - val_loss: 8.0139 - val_mae: 2.3748
Epoch 62/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 12.2299 - mae: 2.5824 - val_loss: 9.3103 - val_mae: 2.5341
Epoch 63/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 14.8777 - mae: 2.6149 - val_loss: 6.7354 - val_mae: 2.1907
Epoch 64/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 11.7033 - mae: 2.4568 - val_loss: 10.5510 - val_mae: 2.680
Epoch 65/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 17ms/step - loss: 13.2904 - mae: 2.6771 - val_loss: 8.8410 - val_mae: 2.5046
Epoch 66/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 13.6509 - mae: 2.6170 - val_loss: 7.2300 - val_mae: 2.2516
Epoch 67/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 15.0482 - mae: 2.6374 - val_loss: 9.3105 - val_mae: 2.5200
Epoch 68/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 12.1291 - mae: 2.4542 - val_loss: 7.6533 - val_mae: 2.3286
Epoch 69/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 16ms/step - loss: 10.9769 - mae: 2.4600 - val_loss: 7.3979 - val_mae: 2.2292
Epoch 70/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 13.5058 - mae: 2.4917 - val_loss: 7.9313 - val_mae: 2.3350
Epoch 71/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 11.7679 - mae: 2.5099 - val_loss: 9.3719 - val_mae: 2.5486
Epoch 72/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 13.6345 - mae: 2.5247 - val_loss: 7.0106 - val_mae: 2.1462
Epoch 73/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 12.6449 - mae: 2.3881 - val_loss: 9.1002 - val_mae: 2.4980
Epoch 74/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 15.9245 - mae: 2.5901 - val_loss: 8.1376 - val_mae: 2.3719
Epoch 75/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 15.8226 - mae: 2.6097 - val_loss: 7.6356 - val_mae: 2.2372
Epoch 76/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 17ms/step - loss: 12.7828 - mae: 2.3728 - val_loss: 10.1217 - val_mae: 2.638
Epoch 77/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 12.3740 - mae: 2.3993 - val_loss: 6.4866 - val_mae: 2.1118
Epoch 78/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 12.8350 - mae: 2.4980 - val_loss: 11.4893 - val_mae: 2.727
Epoch 79/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 15.1856 - mae: 2.5562 - val_loss: 6.9320 - val_mae: 2.1697
Epoch 80/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 12.0645 - mae: 2.4415 - val_loss: 8.8845 - val_mae: 2.4572
Epoch 81/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 14.3679 - mae: 2.4548 - val_loss: 7.4752 - val_mae: 2.2098
Epoch 82/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 13.6894 - mae: 2.4690 - val_loss: 7.4541 - val_mae: 2.2210
Epoch 83/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 11.5407 - mae: 2.3552 - val_loss: 9.0693 - val_mae: 2.4705
Epoch 84/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 13.6930 - mae: 2.5886 - val_loss: 8.0108 - val_mae: 2.3253
Epoch 85/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 17ms/step - loss: 14.5838 - mae: 2.5502 - val_loss: 8.0225 - val_mae: 2.2982
Epoch 86/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 22ms/step - loss: 11.0885 - mae: 2.2977 - val_loss: 9.7583 - val_mae: 2.5485
Epoch 87/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 14ms/step - loss: 15.1284 - mae: 2.6252 - val_loss: 6.8525 - val_mae: 2.1456
Epoch 88/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 15ms/step - loss: 12.8842 - mae: 2.4832 - val_loss: 8.0403 - val_mae: 2.2982
Epoch 89/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 19ms/step - loss: 12.1975 - mae: 2.3512 - val_loss: 7.7076 - val_mae: 2.2314
Epoch 90/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 14.9518 - mae: 2.5006 - val_loss: 8.7125 - val_mae: 2.4210
Epoch 91/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 12ms/step - loss: 12.2752 - mae: 2.4206 - val_loss: 7.4087 - val_mae: 2.2363
Epoch 92/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 12.2730 - mae: 2.4045 - val_loss: 8.7600 - val_mae: 2.3716
Epoch 93/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 12.9999 - mae: 2.4086 - val_loss: 7.5932 - val_mae: 2.2497
Epoch 94/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 11.4558 - mae: 2.3021 - val_loss: 10.6551 - val_mae: 2.694
Epoch 95/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 13.4010 - mae: 2.5193 - val_loss: 7.6782 - val_mae: 2.2079
Epoch 96/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 10ms/step - loss: 15.8329 - mae: 2.4969 - val_loss: 7.2144 - val_mae: 2.1533
Epoch 97/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 11ms/step - loss: 15.9514 - mae: 2.5868 - val_loss: 7.2622 - val_mae: 2.1412
Epoch 98/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 14.0789 - mae: 2.5148 - val_loss: 10.4689 - val_mae: 2.607
Epoch 99/100
 [1m12/12 [0m  [32m━━━━━━━━━━━━━━━━━━━━ [0m [37m [0m  [1m0s [0m 13ms/step - loss: 11.8978 - mae: 2.4677 - val_loss: 7.6535 - val_mae: 2.1700
```

```
Epoch 100/100
[1m12/12[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m0s[0m 11ms/step - loss: 15.0442 - mae: 2.5918 - val_loss: 9.1131 - val_mae: 2.4900
```

In [14]:

```python
# Evaluating the Neural Network Model
y_pred_nn = model.predict(X_test_scaled)  # Predicting house prices on test data
mse_nn, mae_nn = model.evaluate(X_test_scaled, y_test)  # Evaluating model performance
# Displaying Neural Network Evaluation Metrics
print("\nNeural Network Model Evaluation:")
print(f"Mean Squared Error: {mse_nn}")
print(f"Mean Absolute Error: {mae_nn}")
# House Price Prediction for New Data
new_data = np.array([[0.1, 10.0, 5.0]])
# New input values: LSTAT=0.1, RM=10.0, PTRATIO=5.0
new_data_scaled = scaler.transform(new_data)
# Applying the same standardization as training data
# Predicting price using trained neural network model
prediction = model.predict(new_data_scaled)
```

```
[1m4/4[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m0s[0m 6ms/step
[1m4/4[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m0s[0m 12ms/step - loss: 17.5633 - mae: 2.7298

Neural Network Model Evaluation:
Mean Squared Error: 21.3414363861084
Mean Absolute Error: 2.894954204559326
[1m1/1[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m0s[0m 69ms/step
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but StandardScaler
was fitted with feature names
  warnings.warn(
```

```
[1m1/1[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m0s[0m 71ms/step
```

In [15]:

```python
# Displaying the predicted house price
print("\nPredicted House Price:", prediction[0][0])
```

```

Predicted House Price: 81.10206
```

In [ ]:

```
```