Name : Harshada Mhaske Div : B

```
import numpy as np
import pandas as pd

# Load Boston housing data manually from the original source
data_url = "http://lib.stat.cmu.edu/datasets/boston"
raw_df = pd.read_csv(data_url, sep="\s+", skiprows=22, header=None)

# Process the raw data
data = np.hstack([raw_df.values[::2, :], raw_df.values[1::2, :2]])
target = raw_df.values[1::2, 2]

# Feature names from the original dataset
feature_names = [
    "CRIM", "ZN", "INDUS", "CHAS", "NOX", "RM", "AGE",
    "DIS", "RAD", "TAX", "PTRATIO", "B", "LSTAT"
]

# Create the DataFrame
boston_df = pd.DataFrame(data, columns=feature_names)
boston_df['MEDV'] = target  # Add target column

# View the first few rows
boston_df.head()
```

	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	1.0	296.0	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	2.0	242.0	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	2.0	242.0	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	3.0	222.0	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	3.0	222.0	18.7	396.90	5.33	36.2

```
data = pd.DataFrame(data, columns=[
    "CRIM", "ZN", "INDUS", "CHAS", "NOX", "RM", "AGE",
    "DIS", "RAD", "TAX", "PTRATIO", "B", "LSTAT"
])
data['PRICE'] = target
data.head(10)
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	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	2.0	242.0	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	2.0	242.0	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	3.0	222.0	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	3.0	222.0	18.7	396.90	5.33	36.2
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3.0	222.0	18.7	394.12	5.21	28.7
6	0.08829	12.5	7.87	0.0	0.524	6.012	66.6	5.5605	5.0	311.0	15.2	395.60	12.43	22.9
7	0.14455	12.5	7.87	0.0	0.524	6.172	96.1	5.9505	5.0	311.0	15.2	396.90	19.15	27.1
8	0.21124	12.5	7.87	0.0	0.524	5.631	100.0	6.0821	5.0	311.0	15.2	386.63	29.93	16.5
9	0.17004	12.5	7.87	0.0	0.524	6.004	85.9	6.5921	5.0	311.0	15.2	386.71	17.10	18.9

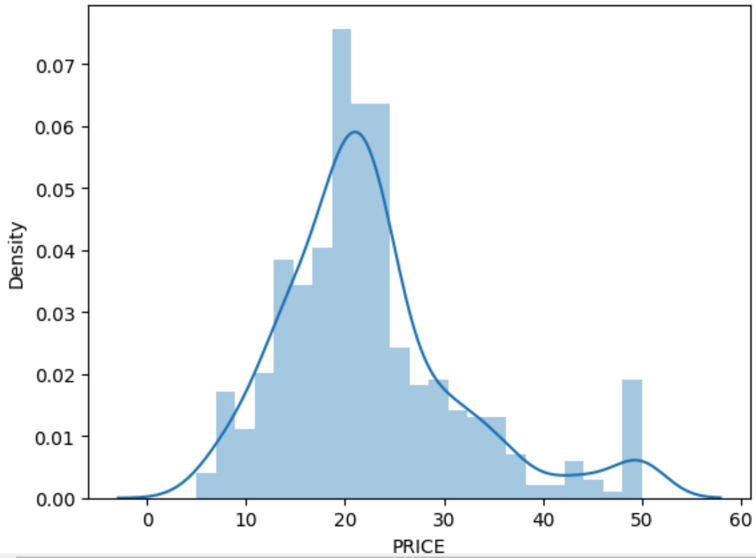
```
#Shape of the data
print(data.shape)
#Checking the null values in the dataset
data.isnull().sum()
```

(506, 14)

	0
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
PRICE	0

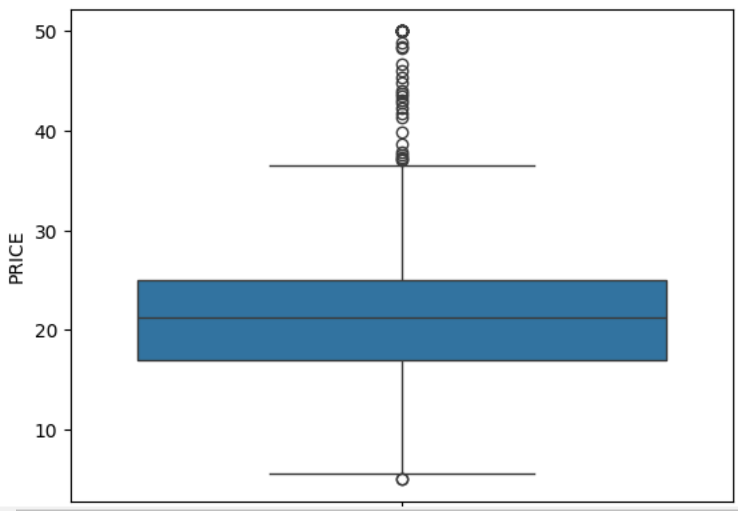
```
#checking the distribution of the target variable
import seaborn as sns
sns.distplot(data.PRICE)
#The distribution seems normal, has not be the data normal we would have perform log transformation or took to square root of the data i
# Normal distribution is need for the machine learning for better predictiblity of the model
```

```
<ipython-input-16-6e69d4d32b98>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data.PRICE)
<Axes: xlabel='PRICE', ylabel='Density'>
```

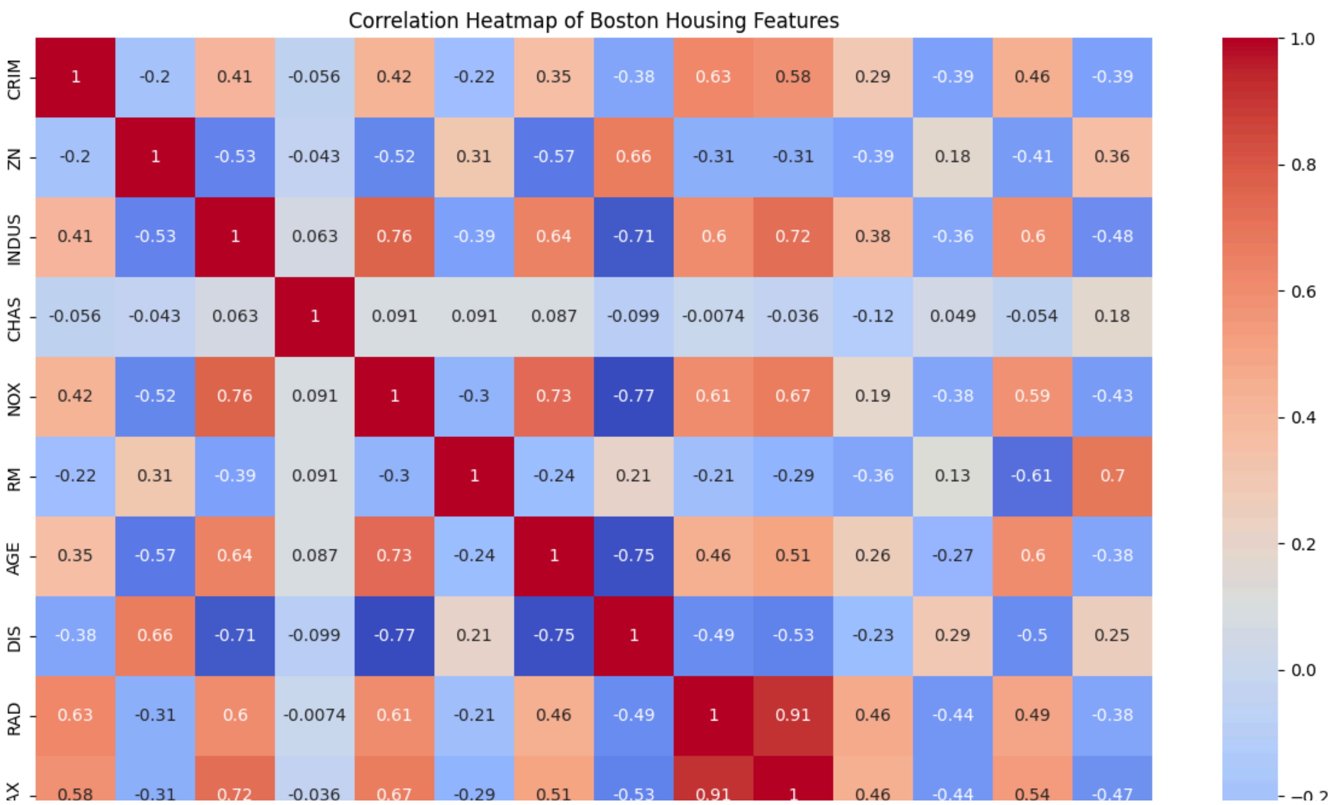



```
#Distribution using box plot
sns.boxplot(data.PRICE)
```

```
<Axes: ylabel='PRICE'>
```

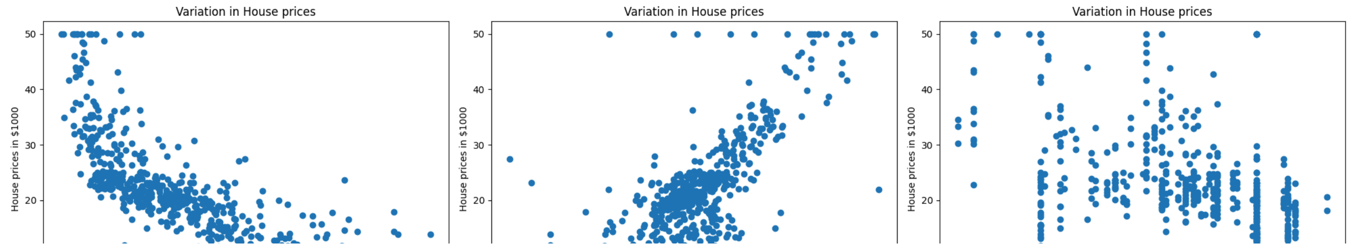



```
correlation = data.corr()
correlation.loc['PRICE']

import matplotlib.pyplot as plt
import seaborn as sns

fig, axes = plt.subplots(figsize=(15, 12))
sns.heatmap(correlation, square=True, annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap of Boston Housing Features")
plt.show()
```




```
plt.figure(figsize=(20, 5))
features = ['LSTAT', 'RM', 'PTRATIO']

for i, col in enumerate(features):
    plt.subplot(1, len(features), i + 1)
    x = data[col]
    y = data['PRICE']
    plt.scatter(x, y, marker='o')
    plt.title("Variation in House prices")
    plt.xlabel(col)
    plt.ylabel("House prices in $1000")

plt.tight_layout()
```

```
plt.show()
```

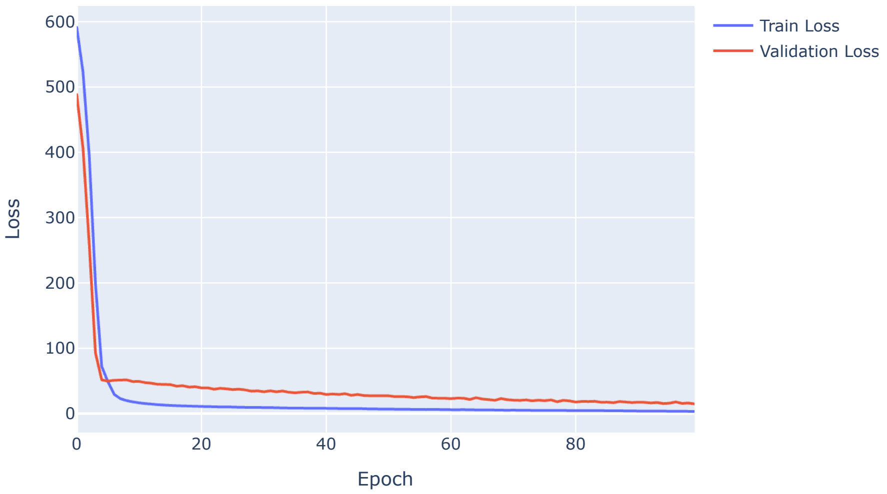



```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import plot_model
import matplotlib.pyplot as plt
import plotly.graph_objects as go

# Splitting the data into dependent and independent variables
X = data.iloc[:, :-1]  # All columns except the last one (features)
y = data['PRICE']      # Last column as the dependent variable (target)

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Normalizing the dataset (Standardization)
sc = StandardScaler()
X_train = sc.fit_transform(X_train)  # Fit and transform on the training data
X_test = sc.transform(X_test)        # Only transform on the testing data

# Linear Regression Model
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting on the test set
y_pred = regressor.predict(X_test)

# Evaluating the Linear Regression Model
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
print(f'Linear Regression RMSE: {rmse}')
print(f'Linear Regression R-squared: {r2}')

# Neural Network Model
model = Sequential()
model.add(Dense(128, activation='relu', input_dim=X_train.shape[1]))  # Input layer
model.add(Dense(64, activation='relu'))  # Hidden layer 1
model.add(Dense(32, activation='relu'))  # Hidden layer 2
model.add(Dense(1))  # Output layer

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mae'])

# Fit the model
history = model.fit(X_train, y_train, epochs=100, validation_split=0.05)

# Predicting on the test set
nn_pred = model.predict(X_test)

# Evaluating the Neural Network Model
nn_rmse = np.sqrt(mean_squared_error(y_test, nn_pred))
nn_r2 = r2_score(y_test, nn_pred)
print(f'Neural Network RMSE: {nn_rmse}')
print(f'Neural Network R-squared: {nn_r2}')

# Visualizing the Neural Network architecture
plot_model(model, to_file='model_architecture.png', show_shapes=True, show_layer_names=True)

# Plotting the loss curve
fig = go.Figure()
fig.add_trace(go.Scattergl(y=history.history['loss'], name='Train Loss'))
fig.add_trace(go.Scattergl(y=history.history['val_loss'], name='Validation Loss'))
fig.update_layout(height=500, width=700, xaxis_title='Epoch', yaxis_title='Loss')
fig.show()
```

```
Linear Regression RMSE: 4.928602182665336
Linear Regression R-squared: 0.668759493535632
Epoch 1/100
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` arg
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
12/12 ━━━━━━━━━━━━━━━━━━━━ 7s 93ms/step - loss: 603.9944 - mae: 22.5837 - val_loss: 488.4428 - val_mae: 20.6244
Epoch 2/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 17ms/step - loss: 548.6918 - mae: 21.4893 - val_loss: 406.7101 - val_mae: 18.6688
Epoch 3/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - loss: 449.3212 - mae: 18.9746 - val_loss: 260.1602 - val_mae: 14.5035
Epoch 4/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 242.8915 - mae: 13.3405 - val_loss: 93.3343 - val_mae: 7.5019
Epoch 5/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 34ms/step - loss: 81.3693 - mae: 7.2333 - val_loss: 51.9718 - val_mae: 4.7307
Epoch 6/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 35ms/step - loss: 60.6014 - mae: 6.1230 - val_loss: 50.1304 - val_mae: 4.5625
Epoch 7/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step - loss: 28.0745 - mae: 4.1224 - val_loss: 51.3699 - val_mae: 4.6706
Epoch 8/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 47ms/step - loss: 22.4576 - mae: 3.5598 - val_loss: 51.7529 - val_mae: 4.8941
Epoch 9/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - loss: 19.1814 - mae: 3.3007 - val_loss: 51.9432 - val_mae: 4.7043
Epoch 10/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 53ms/step - loss: 21.1598 - mae: 3.3461 - val_loss: 49.4561 - val_mae: 4.4757
Epoch 11/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - loss: 18.7914 - mae: 3.1490 - val_loss: 49.6433 - val_mae: 4.3318
Epoch 12/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 55ms/step - loss: 16.6121 - mae: 2.9137 - val_loss: 47.7592 - val_mae: 4.3779
Epoch 13/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 49ms/step - loss: 13.4711 - mae: 2.6594 - val_loss: 46.8746 - val_mae: 4.2010
Epoch 14/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - loss: 12.3607 - mae: 2.6485 - val_loss: 45.2455 - val_mae: 4.1862
Epoch 15/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - loss: 12.6428 - mae: 2.6405 - val_loss: 45.0997 - val_mae: 4.1130
Epoch 16/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - loss: 12.9422 - mae: 2.7110 - val_loss: 44.7501 - val_mae: 4.1013
Epoch 17/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - loss: 11.8329 - mae: 2.6098 - val_loss: 42.3827 - val_mae: 3.9511
Epoch 18/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 12.1455 - mae: 2.5067 - val_loss: 43.0707 - val_mae: 3.9745
Epoch 19/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - loss: 12.7663 - mae: 2.5725 - val_loss: 40.9813 - val_mae: 3.8973
Epoch 20/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - loss: 11.4098 - mae: 2.3695 - val_loss: 41.4893 - val_mae: 3.8812
Epoch 21/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - loss: 15.1339 - mae: 2.6775 - val_loss: 39.7430 - val_mae: 3.7834
Epoch 22/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - loss: 12.4264 - mae: 2.4596 - val_loss: 39.8033 - val_mae: 3.7997
Epoch 23/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - loss: 11.8803 - mae: 2.4557 - val_loss: 37.7045 - val_mae: 3.6961
Epoch 24/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - loss: 9.2345 - mae: 2.2909 - val_loss: 39.0899 - val_mae: 3.7450
Epoch 25/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 8.9360 - mae: 2.2720 - val_loss: 38.2963 - val_mae: 3.6250
Epoch 26/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 9.3837 - mae: 2.3483 - val_loss: 37.2310 - val_mae: 3.6174
Epoch 27/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.6653 - mae: 2.2557 - val_loss: 37.7893 - val_mae: 3.6755
Epoch 28/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 9.1939 - mae: 2.2975 - val_loss: 36.6240 - val_mae: 3.5290
Epoch 29/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 9.6379 - mae: 2.2705 - val_loss: 34.7356 - val_mae: 3.4923
Epoch 30/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 9.2719 - mae: 2.2914 - val_loss: 35.1351 - val_mae: 3.5039
Epoch 31/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.9853 - mae: 2.1681 - val_loss: 33.7866 - val_mae: 3.4141
Epoch 32/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 10.9167 - mae: 2.3418 - val_loss: 35.2153 - val_mae: 3.4593
Epoch 33/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 9.1738 - mae: 2.2676 - val_loss: 33.8066 - val_mae: 3.4229
Epoch 34/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.7277 - mae: 2.2164 - val_loss: 34.9612 - val_mae: 3.4481
Epoch 35/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.8101 - mae: 2.0479 - val_loss: 33.0818 - val_mae: 3.3164
Epoch 36/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.6795 - mae: 2.1633 - val_loss: 32.2370 - val_mae: 3.3165
Epoch 37/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 9.2147 - mae: 2.1688 - val_loss: 33.1441 - val_mae: 3.2989
Epoch 38/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.8242 - mae: 2.0542 - val_loss: 33.5240 - val_mae: 3.3972
Epoch 39/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 9.0208 - mae: 2.1955 - val_loss: 31.2624 - val_mae: 3.1904
Epoch 40/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.1439 - mae: 2.0816 - val_loss: 31.6186 - val_mae: 3.2358
Epoch 41/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 9.9076 - mae: 2.2069 - val_loss: 29.5987 - val_mae: 3.1162
Epoch 42/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.9142 - mae: 2.1475 - val_loss: 30.4064 - val_mae: 3.2016
Epoch 43/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 8.7017 - mae: 2.1710 - val_loss: 29.6996 - val_mae: 3.1366
```

```
Epoch 44/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.9511 - mae: 2.0209 - val_loss: 30.8902 - val_mae: 3.2322
Epoch 45/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 7.8022 - mae: 2.0651 - val_loss: 28.3880 - val_mae: 3.0180
Epoch 46/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.6956 - mae: 2.0404 - val_loss: 29.6285 - val_mae: 3.1206
Epoch 47/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.8852 - mae: 2.0287 - val_loss: 28.1182 - val_mae: 3.0946
Epoch 48/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 6.9091 - mae: 1.9300 - val_loss: 27.7347 - val_mae: 3.0205
Epoch 49/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.2833 - mae: 1.9952 - val_loss: 27.6650 - val_mae: 3.0414
Epoch 50/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 6.3840 - mae: 1.8686 - val_loss: 27.8050 - val_mae: 3.0843
Epoch 51/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.2245 - mae: 1.9362 - val_loss: 27.6323 - val_mae: 3.0043
Epoch 52/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 6.9733 - mae: 1.9493 - val_loss: 26.3727 - val_mae: 2.9223
Epoch 53/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 7.0205 - mae: 2.0060 - val_loss: 26.5702 - val_mae: 3.0362
Epoch 54/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.7886 - mae: 1.8704 - val_loss: 26.1444 - val_mae: 2.9820
Epoch 55/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.9101 - mae: 1.8565 - val_loss: 24.9457 - val_mae: 2.8588
Epoch 56/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 6.7292 - mae: 1.8947 - val_loss: 25.9635 - val_mae: 2.9507
Epoch 57/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.9963 - mae: 1.8797 - val_loss: 26.4472 - val_mae: 2.9919
Epoch 58/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 6.4481 - mae: 1.9301 - val_loss: 24.1222 - val_mae: 2.8818
Epoch 59/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 7.2427 - mae: 2.0125 - val_loss: 23.9870 - val_mae: 2.8921
Epoch 60/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 7.0565 - mae: 1.9580 - val_loss: 23.8393 - val_mae: 2.9029
Epoch 61/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 5.5646 - mae: 1.7248 - val_loss: 23.3274 - val_mae: 2.8144
Epoch 62/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 7.2673 - mae: 1.9726 - val_loss: 24.1395 - val_mae: 2.8715
Epoch 63/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 8.1775 - mae: 2.1230 - val_loss: 23.9549 - val_mae: 2.9810
Epoch 64/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 6.0745 - mae: 1.8644 - val_loss: 22.0309 - val_mae: 2.7619
Epoch 65/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 6.3683 - mae: 1.8647 - val_loss: 24.7513 - val_mae: 2.9534
Epoch 66/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 6.3598 - mae: 1.8968 - val_loss: 22.6886 - val_mae: 2.8588
Epoch 67/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 6.6075 - mae: 1.9511 - val_loss: 21.8509 - val_mae: 2.7915
Epoch 68/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - loss: 4.9583 - mae: 1.6935 - val_loss: 20.7487 - val_mae: 2.7271
Epoch 69/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.0884 - mae: 1.7203 - val_loss: 23.3941 - val_mae: 2.8411
Epoch 70/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.5937 - mae: 1.8019 - val_loss: 21.6840 - val_mae: 2.7330
Epoch 71/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 6.1656 - mae: 1.9096 - val_loss: 20.9544 - val_mae: 2.7440
Epoch 72/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 5.9397 - mae: 1.7989 - val_loss: 20.6319 - val_mae: 2.7998
Epoch 73/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.3385 - mae: 1.7874 - val_loss: 21.3492 - val_mae: 2.7821
Epoch 74/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.9214 - mae: 1.6673 - val_loss: 19.9503 - val_mae: 2.7041
Epoch 75/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.4067 - mae: 1.5780 - val_loss: 20.9678 - val_mae: 2.7431
Epoch 76/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.8954 - mae: 1.8476 - val_loss: 20.1565 - val_mae: 2.7152
Epoch 77/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.4482 - mae: 1.7262 - val_loss: 21.3321 - val_mae: 2.8250
Epoch 78/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.9227 - mae: 1.6877 - val_loss: 18.5426 - val_mae: 2.6551
Epoch 79/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.5702 - mae: 1.6407 - val_loss: 20.7381 - val_mae: 2.7411
Epoch 80/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.0296 - mae: 1.6979 - val_loss: 19.7712 - val_mae: 2.7379
Epoch 81/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 5.2885 - mae: 1.7014 - val_loss: 18.0544 - val_mae: 2.6222
Epoch 82/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 5.3197 - mae: 1.7590 - val_loss: 19.1309 - val_mae: 2.7153
Epoch 83/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.9480 - mae: 1.6499 - val_loss: 18.8791 - val_mae: 2.6925
Epoch 84/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 5.4591 - mae: 1.7736 - val_loss: 19.2158 - val_mae: 2.7312
Epoch 85/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 5.3782 - mae: 1.7400 - val_loss: 17.8183 - val_mae: 2.5488
Epoch 86/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.9045 - mae: 1.6490 - val_loss: 17.9375 - val_mae: 2.6576
Epoch 87/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.5922 - mae: 1.6407 - val_loss: 17.0865 - val_mae: 2.5880
Epoch 88/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 4.2839 - mae: 1.5810 - val_loss: 18.8464 - val_mae: 2.7334
Epoch 89/100
```

```
Epoch 89/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.1646 - mae: 1.5880 - val_loss: 18.0950 - val_mae: 2.6718
Epoch 90/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.0712 - mae: 1.5684 - val_loss: 17.3004 - val_mae: 2.5955
Epoch 91/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.7581 - mae: 1.4679 - val_loss: 17.8890 - val_mae: 2.6414
Epoch 92/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 4.4195 - mae: 1.5391 - val_loss: 17.5956 - val_mae: 2.6831
Epoch 93/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.0690 - mae: 1.4838 - val_loss: 16.7389 - val_mae: 2.6056
Epoch 94/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.8287 - mae: 1.6387 - val_loss: 17.3797 - val_mae: 2.6209
Epoch 95/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.0349 - mae: 1.5175 - val_loss: 15.7889 - val_mae: 2.6104
Epoch 96/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 4.3919 - mae: 1.5947 - val_loss: 16.3679 - val_mae: 2.5991
Epoch 97/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.4282 - mae: 1.5670 - val_loss: 18.2408 - val_mae: 2.7294
Epoch 98/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.5331 - mae: 1.4198 - val_loss: 15.9883 - val_mae: 2.5924
Epoch 99/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.0123 - mae: 1.5403 - val_loss: 16.5472 - val_mae: 2.5783
Epoch 100/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.4497 - mae: 1.4016 - val_loss: 15.2528 - val_mae: 2.6233
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step
Neural Network RMSE: 3.2350694015851387
Neural Network R-squared: 0.8572871385866964
```



```
# Ensure that you have compiled your model with 'mae' as a metric
model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mae'])

# Fit the model
history = model.fit(X_train, y_train, epochs=100, validation_split=0.05)

# Plotting Mean Absolute Error (MAE) during training and validation
import plotly.graph_objects as go

fig = go.Figure()

# Train MAE
fig.add_trace(go.Scattergl(y=history.history['mae'], name='Train MAE'))

# Validation MAE
fig.add_trace(go.Scattergl(y=history.history['val_mae'], name='Validation MAE'))

# Update layout
fig.update_layout(height=500, width=700,
                  xaxis_title='Epoch',
                  yaxis_title='Mean Absolute Error')

# Show the plot
fig.show()
```

```
Epoch 1/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 21ms/step - loss: 3.9493 - mae: 1.5070 - val_loss: 14.4235 - val_mae: 2.5434
Epoch 2/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.2676 - mae: 1.5876 - val_loss: 15.8536 - val_mae: 2.5451
Epoch 3/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 3.8668 - mae: 1.4775 - val_loss: 17.3891 - val_mae: 2.6847
Epoch 4/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 4.2653 - mae: 1.4918 - val_loss: 14.7345 - val_mae: 2.6053
Epoch 5/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.6917 - mae: 1.4832 - val_loss: 16.5258 - val_mae: 2.5650
Epoch 6/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.8346 - mae: 1.4787 - val_loss: 13.7171 - val_mae: 2.4558
Epoch 7/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.9862 - mae: 1.4467 - val_loss: 16.1016 - val_mae: 2.6154
Epoch 8/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.7514 - mae: 1.4498 - val_loss: 14.8855 - val_mae: 2.5055
Epoch 9/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.6616 - mae: 1.4344 - val_loss: 16.5480 - val_mae: 2.6373
Epoch 10/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 3.9233 - mae: 1.5303 - val_loss: 15.4411 - val_mae: 2.5133
Epoch 11/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 3.1790 - mae: 1.3484 - val_loss: 15.7078 - val_mae: 2.5912
Epoch 12/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.6381 - mae: 1.3885 - val_loss: 15.3047 - val_mae: 2.5802
Epoch 13/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.6460 - mae: 1.4200 - val_loss: 13.4620 - val_mae: 2.4978
Epoch 14/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.5922 - mae: 1.4402 - val_loss: 15.3447 - val_mae: 2.5411
Epoch 15/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - loss: 3.1267 - mae: 1.3094 - val_loss: 15.4153 - val_mae: 2.5992
Epoch 16/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 3.3223 - mae: 1.3751 - val_loss: 13.9686 - val_mae: 2.4597
Epoch 17/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - loss: 2.8885 - mae: 1.2636 - val_loss: 15.1747 - val_mae: 2.5471
Epoch 18/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - loss: 3.1030 - mae: 1.2636 - val_loss: 12.8697 - val_mae: 2.4535
Epoch 19/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - loss: 3.4844 - mae: 1.4085 - val_loss: 13.4380 - val_mae: 2.4567
Epoch 20/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.6361 - mae: 1.4088 - val_loss: 14.3115 - val_mae: 2.5389
Epoch 21/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 3.3096 - mae: 1.3562 - val_loss: 13.3733 - val_mae: 2.4059
Epoch 22/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.1754 - mae: 1.3300 - val_loss: 14.9122 - val_mae: 2.6030
Epoch 23/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.1542 - mae: 1.2868 - val_loss: 13.6521 - val_mae: 2.4578
Epoch 24/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.6470 - mae: 1.2152 - val_loss: 13.1754 - val_mae: 2.4216
Epoch 25/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.0267 - mae: 1.2821 - val_loss: 14.3243 - val_mae: 2.5018
Epoch 26/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.6698 - mae: 1.1728 - val_loss: 14.4516 - val_mae: 2.4896
Epoch 27/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.9081 - mae: 1.2201 - val_loss: 14.4259 - val_mae: 2.4758
Epoch 28/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 2.7179 - mae: 1.1898 - val_loss: 13.1552 - val_mae: 2.3784
Epoch 29/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 2.8308 - mae: 1.3004 - val_loss: 13.6989 - val_mae: 2.4428
Epoch 30/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.7925 - mae: 1.2216 - val_loss: 13.9928 - val_mae: 2.4597
Epoch 31/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.4946 - mae: 1.2236 - val_loss: 13.6737 - val_mae: 2.6413
Epoch 32/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 3.0332 - mae: 1.3762 - val_loss: 13.2655 - val_mae: 2.4488
Epoch 33/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - loss: 3.0812 - mae: 1.2880 - val_loss: 15.0615 - val_mae: 2.4517
Epoch 34/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 2.5863 - mae: 1.1879 - val_loss: 15.0721 - val_mae: 2.5444
Epoch 35/100
```