DL-1(Boston)

Step 1: Import Required Libraries

Import tensorflow as tf

From tensorflow.keras import layers, models

From sklearn.model\_selection import train\_test\_split

From sklearn.preprocessing import StandardScaler

From sklearn.datasets import load\_boston

Import numpy as np

Import matplotlib.pyplot as plt

* Note: If using TensorFlow >= 2.5, load\_boston is deprecated. You can use it from sklearn.datasets in older versions or load from online CSV.

Step 2: Load and Preprocess Dataset

# Load dataset

Boston = load\_boston()

X = boston.data

Y = boston.target

# Split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Normalize features

Scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

Step 3: Build the Deep Neural Network Model

Model = models.Sequential([

Layers.Dense(64, activation=’relu’, input\_shape=(X\_train.shape[1],)),

Layers.Dense(32, activation=’relu’),

Layers.Dense(1) # Output layer for regression

])

# Compile the model

Model.compile(optimizer=’adam’, loss=’mse’, metrics=[‘mae’])

Step 4: Train the Model

History = model.fit(X\_train, y\_train, epochs=100, batch\_size=16,

Validation\_split=0.2, verbose=0)

Step 5: Evaluate the Model

Loss, mae = model.evaluate(X\_test, y\_test, verbose=0)

Print(f”Test Mean Absolute Error: {mae:.2f}”)

Step 6: Plot Loss Curve

Plt.plot(history.history[‘loss’], label=’Train Loss’)

Plt.plot(history.history[‘val\_loss’], label=’Validation Loss’)

Plt.xlabel(‘Epoch’)

Plt.ylabel(‘MSE Loss’)

Plt.legend()

Plt.title(“Training vs Validation Loss”)

Plt.show()

Step 7: Predicting and Visualizing Results

Predictions = model.predict(X\_test).flatten()

Plt.scatter(y\_test, predictions)

Plt.xlabel(“Actual Prices”)

Plt.ylabel(“Predicted Prices”)

Plt.title(“Actual vs Predicted House Prices”)

Plt.grid(True)

Plt.plot([min(y\_test), max(y\_test)], [min(y\_test), max(y\_test)], ‘r’) # Diagonal line

Plt.show()