**Lab 6**

**OBJECTIVE: Deep Learning for Housing Price Prediction**

**Lab Task**

1. **Increase or decrease the number of neurons in hidden layers**

CODE

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input

from tensorflow.keras.datasets import boston\_housing

from sklearn.preprocessing import StandardScaler

(X\_train, y\_train), (X\_test, y\_test) = boston\_housing.load\_data()

print("Training data shape:", X\_train.shape)

print("Test data shape:", X\_test.shape)



scaler = StandardScaler()

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

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

model = Sequential([

Input(shape=(X\_train\_scaled.shape[1],)), # Should be (13,)

Dense(128, activation='relu'),

Dense(64, activation='relu'),

Dense(1, activation='linear') # For regression

])

model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['mse'])

model.fit(X\_train\_scaled, y\_train, epochs=100, batch\_size=32, verbose=0)

loss, mse = model.evaluate(X\_test\_scaled, y\_test, verbose=0)

print("Mean Squared Error on Test Data:", mse)

1. **Add more hidden layers**

**CODE**

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input

from tensorflow.keras.datasets import boston\_housing

from sklearn.preprocessing import StandardScaler

(X\_train, y\_train), (X\_test, y\_test) = boston\_housing.load\_data()

print("Training data shape:", X\_train.shape)

print("Test data shape:", X\_test.shape)

scaler = StandardScaler()

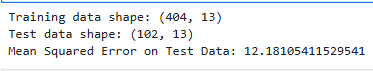
X\_train\_scaled = scaler.fit\_transform(X\_train)

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

model = Sequential([

Input(shape=(X\_train\_scaled.shape[1],)),

Dense(128, activation='relu'),



Dense(64, activation='relu'),

Dense(96, activation='relu'),

Dense(63, activation='relu'),

Dense(32, activation='relu'),

Dense(16, activation='relu'),

Dense(1, activation='linear') #

])

model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['mse'])

model.fit(X\_train\_scaled, y\_train, epochs=100, batch\_size=32, verbose=0)

loss, mse = model.evaluate(X\_test\_scaled, y\_test, verbose=0)

print("Mean Squared Error on Test Data:", mse)

1. **Increase number of epochs**

**CODE**

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input

from tensorflow.keras.datasets import boston\_housing

from sklearn.preprocessing import StandardScaler

(X\_train, y\_train), (X\_test, y\_test) = boston\_housing.load\_data()

print("Training data shape:", X\_train.shape)

print("Test data shape:", X\_test.shape)

scaler = StandardScaler()

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

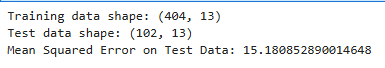
X\_test\_scaled = scaler.transform(X\_test)

model = Sequential([

Input(shape=(X\_train\_scaled.shape[1],)),

Dense(128, activation='relu'),

Dense(64, activation='relu'),



Dense(96, activation='relu'),

Dense(63, activation='relu'),

Dense(32, activation='relu'),

Dense(16, activation='relu'),

Dense(1, activation='linear') #

])

model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['mse'])

model.fit(X\_train\_scaled, y\_train, epochs=200, batch\_size=32, verbose=0)

loss, mse = model.evaluate(X\_test\_scaled, y\_test, verbose=0)

print("Mean Squared Error on Test Data:", mse)

**4. Change the dataset from `boston\_housing` to `fetch\_california\_housing` and observe the results**

**CODE**

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input

from sklearn.datasets import fetch\_california\_housing

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

from sklearn.preprocessing import StandardScaler

data = fetch\_california\_housing()

X, y = data.data, data.target

print("Dataset shape:", X.shape)

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

scaler = StandardScaler()

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

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

model = Sequential([

Input(shape=(X\_train\_scaled.shape[1],)), # 8 features for California housing

Dense(128, activation='relu'),

Dense(96, activation='relu'),

Dense(64, activation='relu'),

Dense(32, activation='relu'),

Dense(16, activation='relu'),

Dense(1, activation='linear') # Regression output

])

model.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['mse'])

model.fit(X\_train\_scaled, y\_train, epochs=100, batch\_size=32, verbose=1)

loss, mse = model.evaluate(X\_test\_scaled, y\_test, verbose=0)

print("Mean Squared Error on California Housing Test Data:", mse)

