**EX.NO:8 DATE:4/9/2024**

**Reg.no:220701016**

**IMPLEMENTING ARTIFICIAL NEURAL NETWORKS FOR AN**

**APPLICATION USING PYTHON – REGRESSION**

**AIM :**

To implementing artificial neural networks for an application in Regression using python.

**CODE**:

import numpy as np

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from keras.models import Sequential

from keras.layers import Dense

from keras.optimizers import Adam

import matplotlib.pyplot as plt

np.random.seed(42)

X = np.random.rand(1000, 3) # 1000 samples, 3 features

y = 3 \* X[:, 0] + 2 \* X[:, 1] \*\* 2 + 1.5 \* np.sin(X[:, 2] \* np.pi) + np.random.normal(0, 0.1, 1000) # Non-linear relationship

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

scaler = StandardScaler()

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

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

model = Sequential()

model.add(Dense(10, input\_dim=X\_train.shape[1], activation='relu'))

model.add(Dense(10, activation='relu'))

model.add(Dense(1, activation='linear'))

model.compile(optimizer=Adam(learning\_rate=0.01), loss='mean\_squared\_error')

history = model.fit(X\_train, y\_train, epochs=100, batch\_size=32, validation\_split=0.2, verbose=1)

y\_pred = model.predict(X\_test)

mse = np.mean((y\_test - y\_pred.flatten()) \*\* 2)

print(f'Mean Squared Error: {mse:.4f}')

plt.figure(figsize=(12, 6))

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val\_loss'], label='Validation Loss')

plt.title('Training and Validation Loss')

plt.xlabel('Epoch')

plt.ylabel('Loss')

plt.legend()

plt.show()

**OUTPUT**:  
