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

**Reg.no:220701010**

**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 matplotlib.pyplot as plt

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

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

from sklearn.preprocessing import StandardScaler

np.random.seed(42)

X = np.linspace(0, 10, 100)

y = 2 \* X + 1 + np.random.normal(0, 1, 100)

X = X.reshape(-1, 1)

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(units=64, activation='relu', input\_dim=1))

model.add(Dense(units=32, activation='relu'))

model.add(Dense(units=1))

model.compile(optimizer='adam', loss='mean\_squared\_error')

history = model.fit(X\_train, y\_train, epochs=100, batch\_size=10, validation\_split=0.2)

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

plt.scatter(X\_test, y\_test, color='blue', label='True values')

plt.scatter(X\_test, y\_pred, color='red', label='Predictions')

plt.plot(X\_test, y\_pred, color='red', linewidth=2)

plt.title('Artificial Neural Network Regression')

plt.xlabel('X')

plt.ylabel('y')

plt.legend()

plt.show()

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('Epochs')

plt.ylabel('Loss')

plt.legend()

plt.show()

**OUTPUT**:  
