**Name: Arshdeep Singh Mathadu**

**Roll No: 41**

**Subject: Artificial neural network**

**Class : TE**

**Branch: AI & DS**

**Practical – 1**

**Problem statement :** Write a Python program to plot a few activation functions that are being used in neural networks.

**Code :**

**Binary sigmoid function**

import numpy as np

import matplotlib.pyplot as plt

import numpy as np

plt.style.use('seaborn')

plt.figure(figsize=(8,4))

def SigmoidBinary(t):

    return 1/(1+np.exp(-t))

t = np.linspace(-5, 5)

plt.plot(t, SigmoidBinary(t))

plt.title('Binary Sigmoid Activation Function')

plt.show()

**Relu (Rectified linear unit)**

plt.style.use('seaborn')

plt.figure(figsize=(8,4))

def RectifiedLinearUnit(t):

    lst=[]

    for i in t:

        if i>=0:

            lst.append(i)

        else:

            lst.append(0)

    return lst

arr = np.linspace(-5, 5)

plt.plot(arr, RectifiedLinearUnit(arr))

plt.title('Rectified Linear Unit Activation Function')

plt.show().

**Leaky Relu**

plt.style.use('seaborn')

plt.figure(figsize=(8,4))

def LeakyRectifiedLinearUnit(arr):

    lst=[]

    for i in arr:

        if i>=0:

            lst.append(i)

        else:

            lst.append(0.01\*i)

    return lst

arr = np.linspace(-5, 5)

plt.plot(arr, LeakyRectifiedLinearUnit(arr))

plt.title('Leaky ReLU Activation Function')

plt.show()

**Softmax**

plt.style.use('seaborn')

plt.figure(figsize=(8,4))

def softmax(t):

    return np.exp(t) / np.sum(np.exp(t))

t = np.linspace(-5, 5)

plt.plot(t, softmax(t))

plt.title('Softmax Activation Function')

plt.show()

**Output :**







