**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management & Engineering**

Program: B.Tech\MBA.Tech Computer Sem VI

**Course: Machine Learning**

Experiment No.07

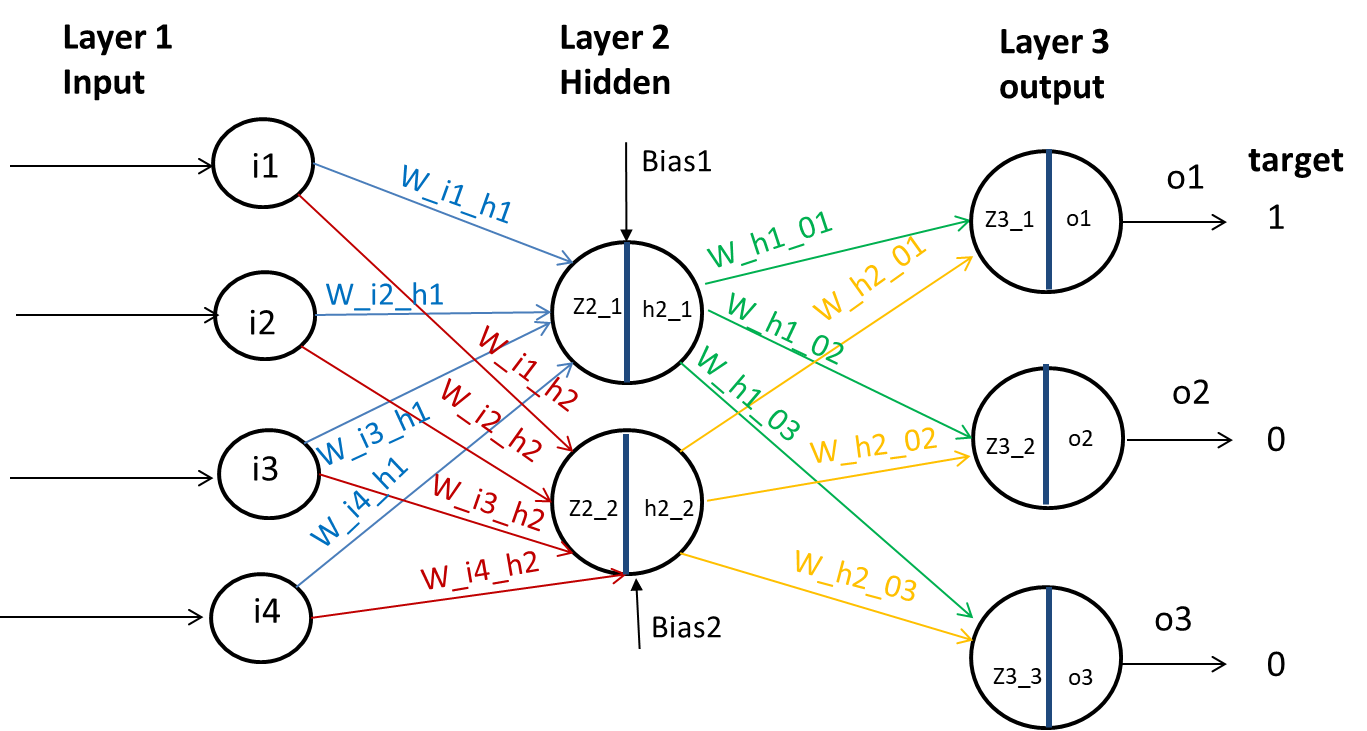
PART A

**A.1 Aim: To implement feedforwad neural network**

Implementation of feedforward neural network:

**Task1:**

**Building a three-layer feed forward neural network from scratch by initializing random weights and biases.**

****

**Task2:**

**Compute the total squared error.**

**Task 3:**

**Change the initial weights and biases and compute the error again.**

**A.2 Prerequisite:**

Python Programming, Numpy

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

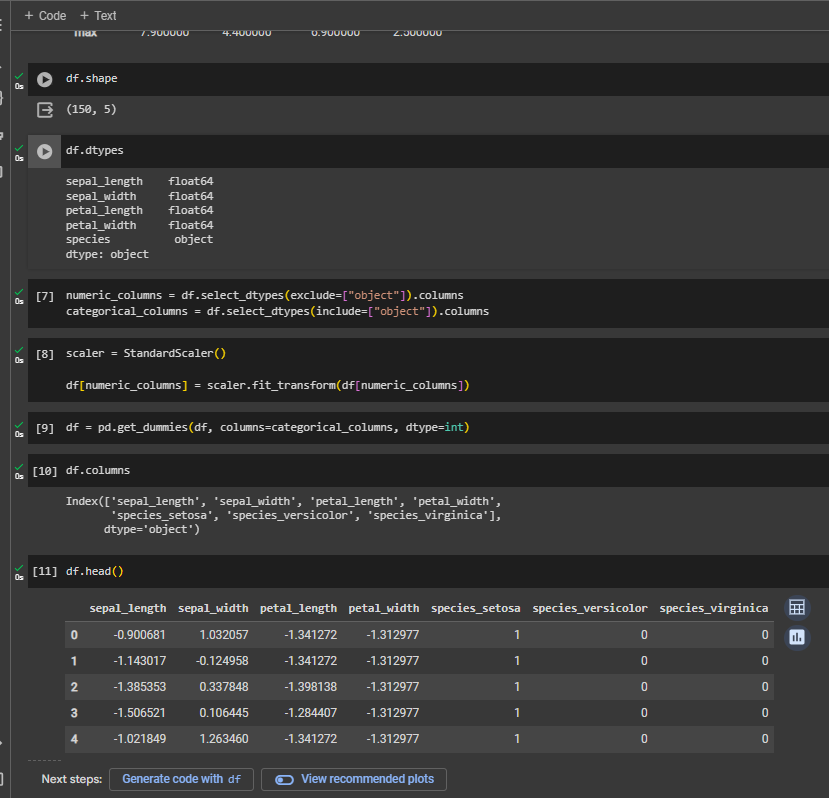
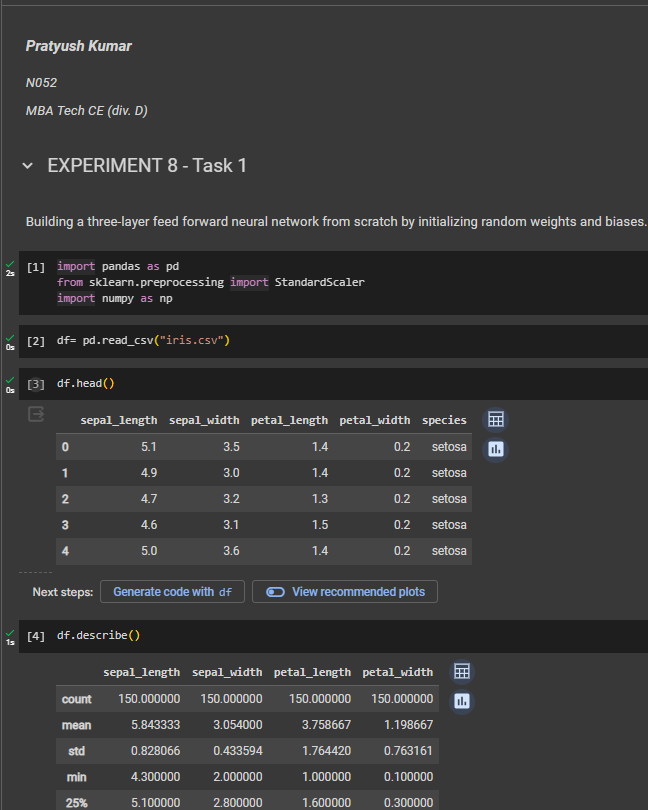
***(Students must submit the soft copy as per following segments within two hours of the practical.)***

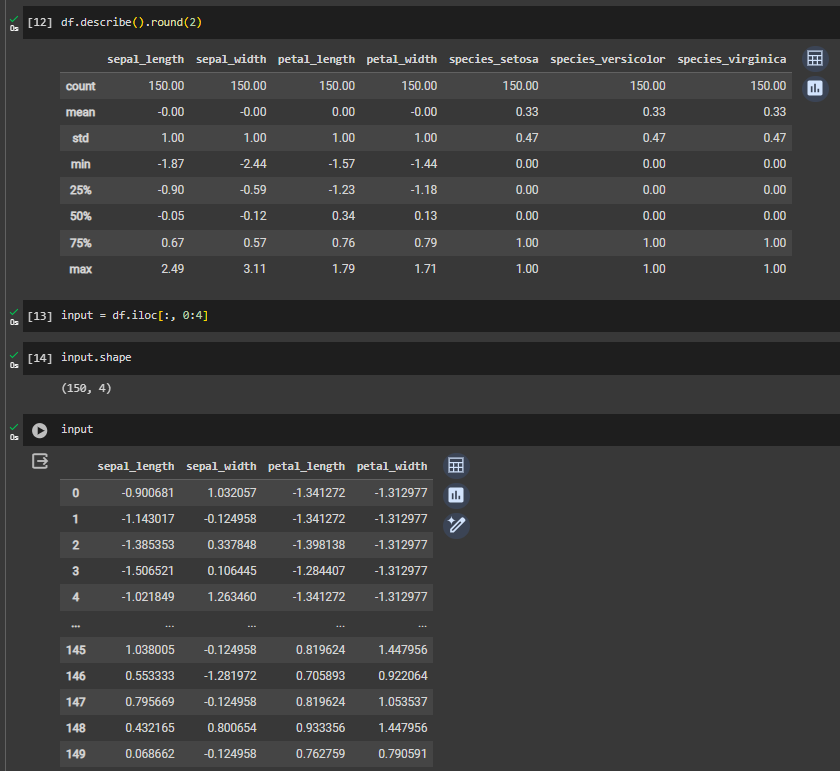
|  |  |
| --- | --- |
| Roll No. N052 | Name: Pratyush Kumar |
| Class : MBA Tech CE (div. D) | Batch : B2 |
| Date of Experiment: 02-02-2024 | Date of Submission: 03-03-2024 |
| Grade : |  |

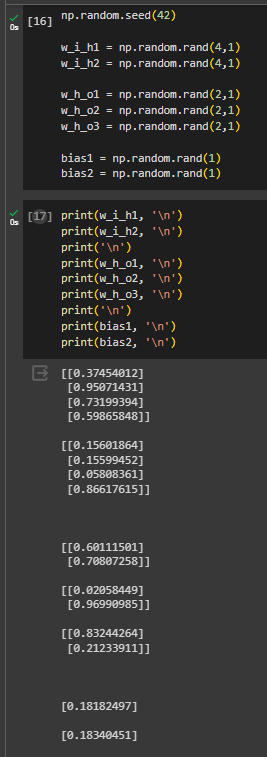
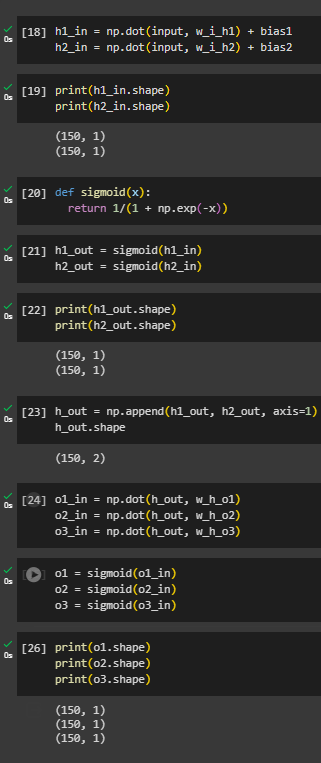
**B.1 Tasks**

**Colab link:** https://colab.research.google.com/drive/1YUOplzNPvmPIaHW1x73AwXytFdg-YwFt?usp=sharing

* **Source Code**
* *"""  
   \* This file contains code snippets to implement neural networks on iris dataset  
   \* ML-E8  
   \*  
   \* Original file is located at: https://colab.research.google.com/drive/1YUOplzNPvmPIaHW1x73AwXytFdg-YwFt  
   \* @author Pratyush Kumar (github.com/pratyushgta)  
  """*"""  
  ## EXPERIMENT 8 - Task 1  
    
  Building a three-layer feed forward neural network from scratch by initializing random weights and biases.  
  """  
    
  import pandas as pd  
  from sklearn.preprocessing import StandardScaler  
  import numpy as np  
    
  df= pd.read\_csv("iris.csv")  
    
  df.head()  
    
  df.describe()  
    
  df.shape  
    
  df.dtypes  
    
  numeric\_columns = df.select\_dtypes(exclude=["object"]).columns  
  categorical\_columns = df.select\_dtypes(include=["object"]).columns  
    
  scaler = StandardScaler()  
    
  df[numeric\_columns] = scaler.fit\_transform(df[numeric\_columns])  
    
  df = pd.get\_dummies(df, columns=categorical\_columns, dtype=int)  
    
  df.columns  
    
  df.head()  
    
  df.describe().round(2)  
    
  input = df.iloc[:, 0:4]  
    
  input.shape  
    
  input  
    
  np.random.seed(42)  
    
  w\_i\_h1 = np.random.rand(4,1)  
  w\_i\_h2 = np.random.rand(4,1)  
    
  w\_h\_o1 = np.random.rand(2,1)  
  w\_h\_o2 = np.random.rand(2,1)  
  w\_h\_o3 = np.random.rand(2,1)  
    
  bias1 = np.random.rand(1)  
  bias2 = np.random.rand(1)  
    
  print(w\_i\_h1, '\n')  
  print(w\_i\_h2, '\n')  
  print('\n')  
  print(w\_h\_o1, '\n')  
  print(w\_h\_o2, '\n')  
  print(w\_h\_o3, '\n')  
  print('\n')  
  print(bias1, '\n')  
  print(bias2, '\n')  
    
  h1\_in = np.dot(input, w\_i\_h1) + bias1  
  h2\_in = np.dot(input, w\_i\_h2) + bias2  
    
  print(h1\_in.shape)  
  print(h2\_in.shape)  
    
  def sigmoid(x):  
   return 1/(1 + np.exp(-x))  
    
  h1\_out = sigmoid(h1\_in)  
  h2\_out = sigmoid(h2\_in)  
    
  print(h1\_out.shape)  
  print(h2\_out.shape)  
    
  h\_out = np.append(h1\_out, h2\_out, axis=1)  
  h\_out.shape  
    
  o1\_in = np.dot(h\_out, w\_h\_o1)  
  o2\_in = np.dot(h\_out, w\_h\_o2)  
  o3\_in = np.dot(h\_out, w\_h\_o3)  
    
  o1 = sigmoid(o1\_in)  
  o2 = sigmoid(o2\_in)  
  o3 = sigmoid(o3\_in)  
    
  print(o1.shape)  
  print(o2.shape)  
  print(o3.shape)
* **Input/ Output**

****

****



**B.2 Conclusion:**

*(Students must write the conclusion in their own words.)*

Implemented a three-layer feedforward architecture with randomly initialized weights and biases. Analyzed the processes of forward propagation, activation functions, and the role of parameters in determining network performance.