Name: Tarun Tanmay

Class: MBATech Computers

Roll No: N049

Sem: 6  
  
#Experiment 1

#Feedforward propagation for IRIS Data Set

#importing libraries

import numpy as np #used for mathematical calculations

import pandas as pd #used for reading the csv file

from sklearn.preprocessing import LabelEncoder #encoding values between 0 to n-1 (here n is different labels)

from sklearn.preprocessing import StandardScaler  #making the mean of the data 0 (standardising)  
  
df = pd.read\_csv('iris\_data.csv') #reaidng the csv file using pandas

df.head() #displaying the data in teh csv file

#df.shape

|  | **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** |
| --- | --- | --- | --- | --- | --- |
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | Iris-setosa |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | Iris-setosa |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | Iris-setosa |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | Iris-setosa |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | Iris-setosa |

#converting category value of column. 'species' to numeric

label = LabelEncoder() #initialising a variable to convert data into numeric

label=label.fit\_transform(df.iloc[:,-1]) #converting 'species' column data to numeric. (all rows and last column in iloc)

#label.shape

label=label.reshape(-1,1) #reshaping the dataset by taking all rows and assigning one column to them, done since numpy can't understand it for labeling before

label.shape #now previous 150 rows are in one column

df.iloc[:,-1]=label #original string values are replaced in the species column

df #150 rows and 5 columns, the species column is replaced with numerals 0 and 2

| **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** |
| --- | --- | --- | --- | --- |
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | 0 |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | 0 |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | 0 |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | 0 |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | 0 |
| **...** | ... | ... | ... | ... | ... |
| **145** | 6.7 | 3.0 | 5.2 | 2.3 | 2 |
| **146** | 6.3 | 2.5 | 5.0 | 1.9 | 2 |
| **147** | 6.5 | 3.0 | 5.2 | 2.0 | 2 |
| **148** | 6.2 | 3.4 | 5.4 | 2.3 | 2 |
| **149** | 5.9 | 3.0 | 5.1 | 1.8 | 2 |

150 rows × 5 columns

DATASET HAS BEEN CLEANSED TILL NOW IN THE PREPROCESSING STEPS

#Scaling of attributes

#make mean 0 and variance 1

#there are 4 attributes (sepal,petal) \* (length,width)

scaler=StandardScaler() #initialising the Scaler function with variable named scaler

input\_float=df.iloc[:,0:4] #assigning of those four attribute columns

scaler.fit(input\_float) #fitting the 4 attributes

input\_float=scaler.transform(input\_float) #making the mean 0 and transforming all the entries of the column by deducting original mean

df.iloc[:,0:4]=input\_float #new transformed data columns overwrites previous data

df.head() #printing the new data

|  | **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** |
| --- | --- | --- | --- | --- | --- |
| **0** | -0.900681 | 1.032057 | -1.341272 | -1.312977 | 0 |
| **1** | -1.143017 | -0.124958 | -1.341272 | -1.312977 | 0 |
| **2** | -1.385353 | 0.337848 | -1.398138 | -1.312977 | 0 |
| **3** | -1.506521 | 0.106445 | -1.284407 | -1.312977 | 0 |
| **4** | -1.021849 | 1.263460 | -1.341272 | -1.312977 | 0 |

#insering new columns in the dataset with different names and smae values

df.insert(5,'flower1',0)

df.insert(6,'flower2',0)

df.insert(7,'flower3',0)

df #we got total 8 columns now (3 new columns we have made)

| **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** | **flower1** | **flower2** | **flower3** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | -0.900681 | 1.032057 | -1.341272 | -1.312977 | 0 | 0 | 0 | 0 |
| **1** | -1.143017 | -0.124958 | -1.341272 | -1.312977 | 0 | 0 | 0 | 0 |
| **2** | -1.385353 | 0.337848 | -1.398138 | -1.312977 | 0 | 0 | 0 | 0 |
| **3** | -1.506521 | 0.106445 | -1.284407 | -1.312977 | 0 | 0 | 0 | 0 |
| **4** | -1.021849 | 1.263460 | -1.341272 | -1.312977 | 0 | 0 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **145** | 1.038005 | -0.124958 | 0.819624 | 1.447956 | 2 | 0 | 0 | 0 |
| **146** | 0.553333 | -1.281972 | 0.705893 | 0.922064 | 2 | 0 | 0 | 0 |
| **147** | 0.795669 | -0.124958 | 0.819624 | 1.053537 | 2 | 0 | 0 | 0 |
| **148** | 0.432165 | 0.800654 | 0.933356 | 1.447956 | 2 | 0 | 0 | 0 |
| **149** | 0.068662 | -0.124958 | 0.762759 | 0.790591 | 2 | 0 | 0 | 0 |

150 rows × 8 columns

[rows,columns]=df.shape #storing the data values in rows and columns

for i in range(rows): #using i iterator to traverse the rows

  if df.iloc[i,4]==0:#if species data value is 0

    df.iloc[i,5]=1 ##condition satisfies then data for flower1 changes to 1

  elif df.iloc[i,4]==1: #if species data value is 1

    df.iloc[i,6]=1 #condition satisfies then data for flower2 changes to 1

  else:

    df.iloc[i,7]=1 #since species had data values as 2, then data of flower3 changes to 1

df

| **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** | **flower1** | **flower2** | **flower3** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | -0.900681 | 1.032057 | -1.341272 | -1.312977 | 0 | 1 | 0 | 0 |
| **1** | -1.143017 | -0.124958 | -1.341272 | -1.312977 | 0 | 1 | 0 | 0 |
| **2** | -1.385353 | 0.337848 | -1.398138 | -1.312977 | 0 | 1 | 0 | 0 |
| **3** | -1.506521 | 0.106445 | -1.284407 | -1.312977 | 0 | 1 | 0 | 0 |
| **4** | -1.021849 | 1.263460 | -1.341272 | -1.312977 | 0 | 1 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **145** | 1.038005 | -0.124958 | 0.819624 | 1.447956 | 2 | 0 | 0 | 1 |
| **146** | 0.553333 | -1.281972 | 0.705893 | 0.922064 | 2 | 0 | 0 | 1 |
| **147** | 0.795669 | -0.124958 | 0.819624 | 1.053537 | 2 | 0 | 0 | 1 |
| **148** | 0.432165 | 0.800654 | 0.933356 | 1.447956 | 2 | 0 | 0 | 1 |
| **149** | 0.068662 | -0.124958 | 0.762759 | 0.790591 | 2 | 0 | 0 | 1 |

150 rows × 8 columns

np.random.seed(12) #using seed so random number generated is same for everyone

w\_i\_h1=np.random.rand(4,1) #generating four weights for first layer neuron of hidden layer

w\_i\_h2=np.random.rand(4,1) #generating four weights for second layer neuron of hidden layer

w\_h\_o1=np.random.rand(1,1) #initialising weight for third layer of neural network (output)

w\_h\_o2=np.random.rand(1,1)  #initialising weight for third layer of neural network (output)

w\_h\_o3=np.random.rand(1,1)  #initialising weight for third layer of neural network (output)

w\_h\_o4=np.random.rand(1,1)  #initialising weight for third layer of neural network (output)

w\_h\_o5=np.random.rand(1,1)  #initialising weight for third layer of neural network (output)

w\_h\_o6=np.random.rand(1,1)  #initialising weight for third layer of neural network (output)

bias1=np.random.rand(1,1)

bias2=np.random.rand(1,1)

def sigmoid(x):#creating the sigmoid function

  y=1/(1+np.exp(-x))

  return y

inputs = np.array(df) #new data frame is fed as array and assigned to variable inputs

features=inputs[:,:4] #separating the attributes (features) from the data to be used for ai Neural network

features.shape #now we have new dataset named features with four attributes

(150, 4)

# input to hidden layer

# input features have 150x4 size

Z2\_1 = np.dot(features, w\_i\_h1) + bias1

Z2\_2 = np.dot(features, w\_i\_h2) + bias2

# apply acticvation function

h2\_1 = sigmoid(Z2\_1)

h2\_2 = sigmoid(Z2\_2)

# hidden to output layer

# hidden layer has 150x2 size

h2 =np.append(h2\_1,h2\_2,axis=1)

Z3\_1 = np.dot(h2,w\_h\_o1)

Z3\_2 = np.dot(h2,w\_h\_o2)

Z3\_3 = np.dot(h2,w\_h\_o3)

o1 = sigmoid(Z3\_1)

o2 = sigmoid(Z3\_2)

o3 = sigmoid(Z3\_3)

# error calculation

fl1= np.array(df.iloc[:,5])

fl1= fl1.reshape((-1,1))

fl2= np.array(df.iloc[:,6])

fl2= fl2.reshape((-1,1))

fl3= np.array(df.iloc[:,7])

fl3= fl3.reshape((-1,1))

error1 = (o1 - fl1)

error2 = (o2 - fl2)

error3 = (o3 - fl3)

error1\_square = np.multiply(error1, error1)

error2\_square = np.multiply(error2, error2)

error3\_square = np.multiply(error3, error3)

error = error1\_square + error2\_square + error3\_square

print(np.sum(error))

148.78292124055253