Date of Performance :

Date of Submission :

EXPERIMENT NUMBER: 3

Aim:Apply Adam Learning GD algorithms to learn the parameters of the supervised single layer feed forward neural network.

Objective:

To implement various training algorithms for feedforward neural networks.

Software Used :Python

Theory:

(a) Feedforward neural networks:

A Feedforward neural network is a type of artificial neural network in which data flows through the network in only one direction, from input layer to output layer, without looping back. It consists of an input layer, hidden layers, and an output layer. The hidden layers perform computation and feature extraction, while the output layer produces the desired outputs. The network is trained using supervised learning algorithms, where the parameters of the network are updated based on the error between the predicted outputs and the ground truth.

(b) Adam Learning GD

Adam is an optimization algorithm commonly used for training neural networks. It is an extension of the traditional Stochastic Gradient Descent (SGD) optimization method. Adam combines the benefits of SGD with the Adaptive Gradient Algorithm to provide a more robust and efficient optimization process. It uses moving averages of the parameters to provide a running estimate of the second raw moments of the gradients; the mean and variance. These moving averages are updated using exponential decay, allowing Adam to react to changes in the distribution of the gradients and adjust the learning rate accordingly. This makes the optimization process more stable and helps to avoid getting stuck in sub-optimal solutions or saddle points.

Algorithm:

Feedforward neural network:

```
Input : ProblemSize, InputPatterns iterationsmax , learnrate) Output
Network
Network ConstructNetworkLayer () ;
Networkweight InitializeWeights (Network, ProblemSize);
for i = 1 to iterations max do
        Patterni < — SelectInputPattern InputPatterns)
        Outputi < —ForwardPropagate Pattern, Network)
        BackwardPropagateError (Pattern, Output, Network)
        UpdateWeight (Pattern, Output, Network, learnrate)
end
return network;
```

Adam Learning GD

- $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \eta = 10^{-8}$ (Defaults)

$m_0 \leftarrow 0$ (Initialize 1st moment vector)

$v_0 \leftarrow 0$ (Initialize 2nd moment vector)

$i \leftarrow 0$ (Initialize step)

while Θ_i not converged **do**

$i \leftarrow i+1$

$g_i \leftarrow \nabla_\Theta f_i(\Theta_{i-1})$ (Get gradients at step i)

$m_i \leftarrow \beta_1 \cdot m_{i-1} + (1-\beta_1) \cdot g_i$ (Update biased first moment estimate)

$v_i \leftarrow \beta_2 \cdot v_{i-1} + (1-\beta_2) \cdot g_i^2$ (Update biased second raw moment estimate)

$\hat{m}_i \leftarrow m_i / (1-\beta_1^i)$ (Compute bias-corrected first moment estimate)

$\hat{v}_i \leftarrow v_i / (1-\beta_2^i)$ (Compute bias-corrected second raw moment estimate)

$\Theta_i \leftarrow \Theta_{i\ 1} - \alpha \cdot \hat{m}_i / (\sqrt{\hat{v}_t} + \eta)$ (Update parameters)

end while

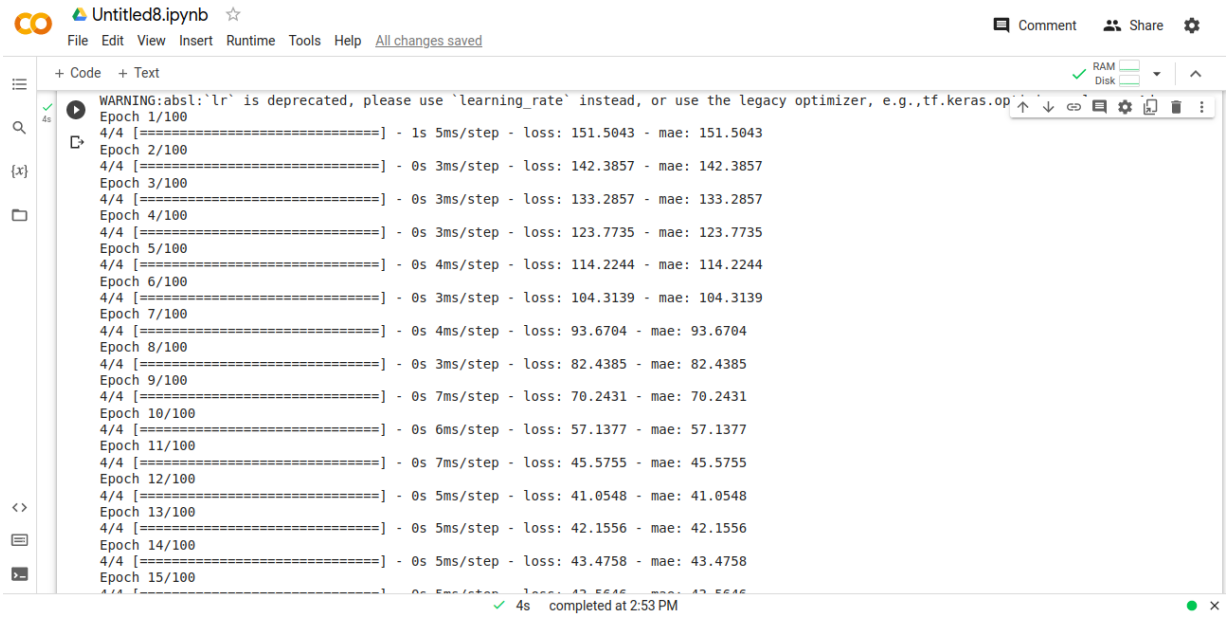
return Θ_i (resulting parameters)

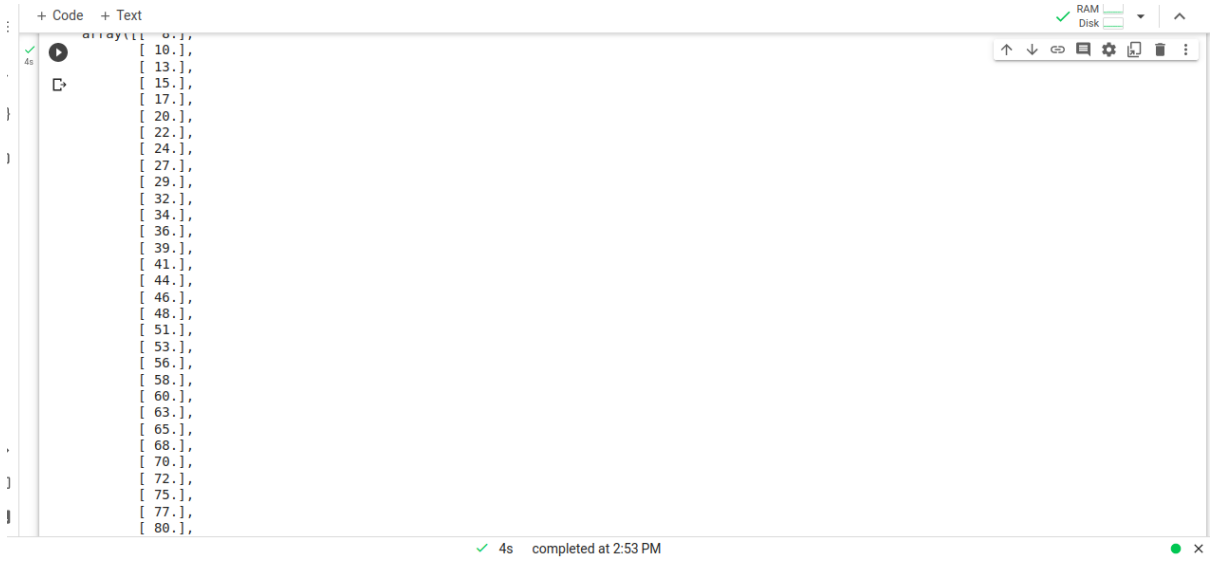
Program:

*********supervised single layer feed forward neural network using Adam Learning GD algorithm*****

```
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

X = tf.constant(tf.range(0,100,1))
Y = X + 100
X , Y
X.shape , Y.shape
tf.random.set_seed(42)
model1 = tf.keras.Sequential([
   tf.keras.layers.Dense(100),
   tf.keras.layers.Dense(10),
   tf.keras.layers.Dense(1),
])

model1.compile(loss = tf.keras.losses.MAE,
           optimizer = tf.keras.optimizers.Adam(lr = 0.01),
           metrics = ['mae'])

model1.fit(tf.expand_dims(X , axis = -1) ,Y , epochs = 100)

model1.evaluate(X , Y)

model1.predict([7])

predict = model1.predict(X)
predict = tf.round(predict)
predict
```

Output





Conclusion/Outcome:

Thus we have implemented supervised single layer feed forward neural networks using Adam Learning GD algorithm.

Marks & Signature:

R1 (4 Marks)	R2 (4 Marks)	R3 (4 Marks)	R4 (3 Mark)	Total (15 Marks)	Signature