Due Date : 2020/01/10 23:55:00

Backpropagation – 100%

1. Description

   Write a backpropagation program to separate y = x data, which has 100 points.

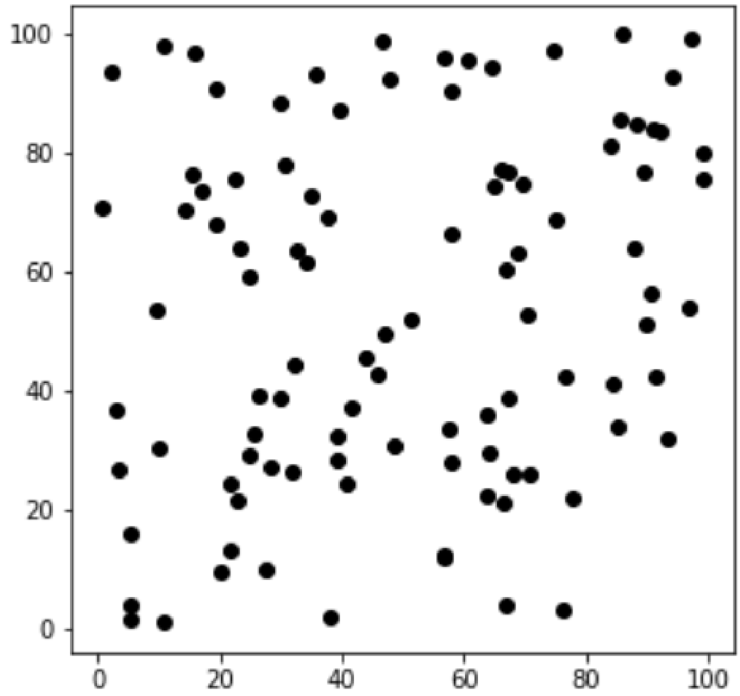


Figure 1: Data

2. Rules

   (a) Implementing a two-layer neural network.

   (b) Two inputs(x,y).

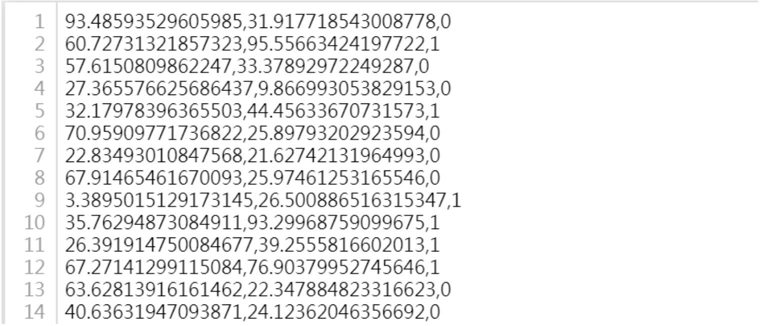
   (c) One output(label). If x>y, label = 0. If x<y, label = 1.

   (d) You must use backpropagation in this neural network.

   (e) You can only use Numpy and other python standard library, so you can't use libraries like Scikit-Learn.

3. Input Parameter

A set of data points(comma seperated : x,y,label)



```
93.48593529605985,31.917718543008778,0
60.72731321857323,95.55663424197722,1
57.6150809862247,33.37892972249287,0
27.365576625686437,9.866993053829153,0
32.17978396365503,44.45633670731573,1
70.95909771736822,25.89793202923594,0
22.83493010847568,21.62742131964993,0
67.91465461670093,25.97461253165546,0
3.3895015129173145,26.500886516315347,1
35.76294873084911,93.29968759099675,1
26.391914750084677,39.2555816602013,1
67.27141299115084,76.90379952745646,1
63.62813916161462,22.347884823316623,0
40.63631947093871,24.12362046356692,0
```

Figure 2: Inputs

4. Grading Policy

   (a) Forward propagation. 10%

   (b) Sigmoid function. 5%

   (c) Back propagation. 30%

   (d) Update weights. 10%

   (e) Plot your predictions and ground truth. 10%

   (f) Print loss(You can use Mean Absolute Error). 15%

   (g) Print accuracy. 20%

5. Output Format

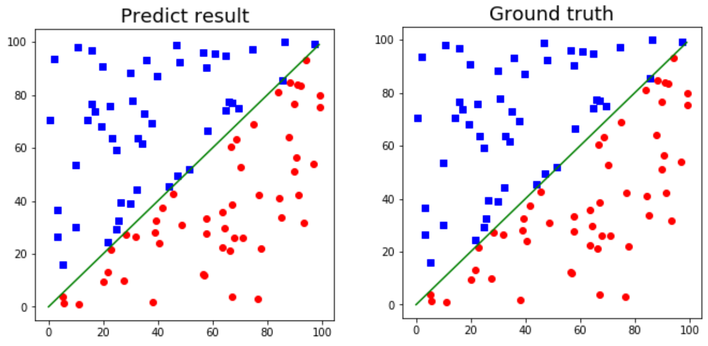
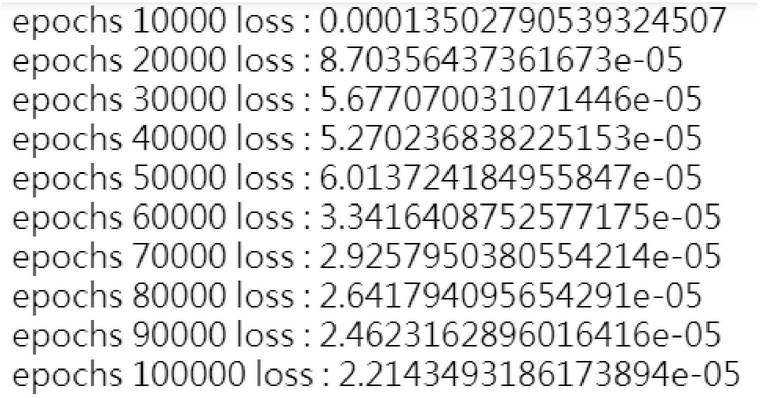
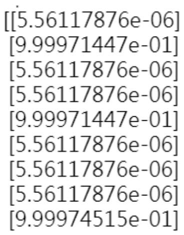


Figure 3: Result



```
epochs 10000 loss : 0.00013502790539324507
epochs 20000 loss : 8.70356437361673e-05
epochs 30000 loss : 5.677070031071446e-05
epochs 40000 loss : 5.270236838225153e-05
epochs 50000 loss : 6.013724184955847e-05
epochs 60000 loss : 3.3416408752577175e-05
epochs 70000 loss : 2.9257950380554214e-05
epochs 80000 loss : 2.641794095654291e-05
epochs 90000 loss : 2.4623162896016416e-05
epochs 100000 loss : 2.2143493186173894e-05
```

Figure 4: Loss

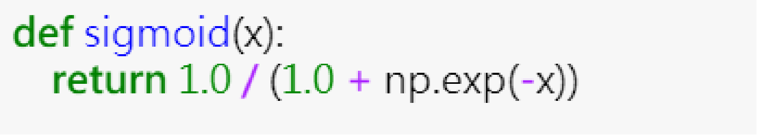


```
[[5.56117876e-06]
 [9.99971447e-01]
 [5.56117876e-06]
 [5.56117876e-06]
 [9.99971447e-01]
 [5.56117876e-06]
 [5.56117876e-06]
 [5.56117876e-06]
 [9.99974515e-01]
```

Figure 5: Accuracy

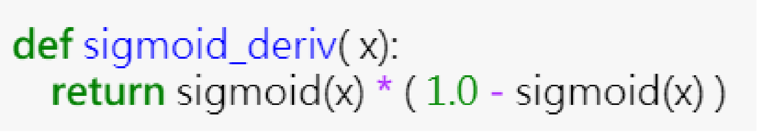
6. Reference

   (a) Sigmoid function



```
def sigmoid(x):
   return 1.0 / (1.0 + np.exp(-x))
```

   (b) Derivative of the Sigmoid function



```
def sigmoid_deriv( x):
   return sigmoid(x) * ( 1.0 - sigmoid(x) )
```

E3 Submission

Please submit a **zip** file, which contains the following, to the newE3 system.

1. Report

   (a) Explanation of how your code works.

   (b) All the content mentioned above.

   (c) Your name and student ID.

   (d) Accept formats: PDF, HTML

2. Source codes

   (a) Accept languages: C/C++, C#, R, python2/python3, MATLAB, JAVA

   (b) NO package-provided model allowed, so the algorithm must be done by you.

       However, numerical computing and visualization packages are allowed, solely for data manipulation and visualization. Take python3 for example:

       i. sklearn X
       ii. numpy O
       iii. pandas O
       iv. matplotlib O

- Your score will be determined mainly by the submitted report.
  - If there is any problem with your code, TA might ask you (through email) to demo it. Otherwise, no demo needed.
- Plagiarizing is not allowed.
  - You will get ZERO on that homework if you get caught the first time.
  - The second time, you'll FAIL this class.
- In order to meet the grading rules the school formulated, your final score of this homework will be normalized by the following formula:
  final_score = (raw_score - raw_score.mean()) * 15. / raw_score.std() + 80

My email address: kartglin.iie07g@nctu.edu.tw