ECE4870/7870 CS 4770/7770 F'17 Computer Assignment 1 Due 10/10/2017

Backpropagation Training of a MLP

Part A:

For this experiment you are to implement a multi-layer perceptron containing a single hidden layer with 10 neurons. Download the cross dataset from Canvas. The zip file contains the dataset and initial weights you should use for this experiment. The file cross_data.csv contains 314 two-dimensional samples, each with a target value of 0 or 1. The initial weights and biases for this network are listed in the tables below, and are also given in the files w1.csv, b1.csv, w2.csv, and b2.csv.

w1.csv (weights)

		From Input Node	
		x_1	x_2
	$w_1^{(1)}$	0.4033	-1.0562
	$w_2^{(1)}$	0.39	0.6544
	$w_3^{(1)}$	0.6376	-0.0601
	$w_4^{(1)}$	0.0064	-0.0462
	$w_5^{(1)}$	0.0782	0.2728
	$w_6^{(1)}$	-0.2115	1.0252
	$w_7^{(1)}$	0.7298	-0.5047
	$w_8^{(1)}$	-0.7109	0.349
	$w_9^{(1)}$	-0.9315	0.9867
	$w_{10}^{(1)}$	0.8441	0.4276

b1.csv

Node	Bias
$b_1^{(1)}$	-0.122
$b_2^{(1)}$	0.9401
$b_3^{(1)}$	0.4271
$b_4^{(1)}$	-0.1775
$b_5^{(1)}$	-0.7019
$b_6^{(1)}$	-0.3326
$b_7^{(1)}$	-0.6961
$b_8^{(1)}$	-0.9316
$b_9^{(1)}$	-0.3681
$b_{10}^{(1)}$	1.0695

b2.csv

Node	Bias
$b_1^{(2)}$	0.1131

w2.csv (weights)

		From Hidden Node									
		$w_1^{(1)}$	$w_2^{(1)}$	$w_3^{(1)}$	$w_4^{(1)}$	$w_5^{(1)}$	$w_6^{(1)}$	$w_7^{(1)}$	$w_8^{(1)}$	$w_9^{(1)}$	$w_{10}^{(1)}$
To Output Node	$w_1^{(2)}$	0.0511	0.1611	0.0238	-0.0267	0.1089	0.2381	0.0784	0.003	0.1646	-0.1779

Use the following parameters in your network:

- Sigmoid activation function for all nodes: $\phi(v) = \frac{1}{1+\exp(-v)}$
- Learning rate: $\alpha = 0.7$
- Momentum term: $\beta = 0.3$

1) Perform **one epoch** of on-line backpropagation training (update weights after each sample) on the cross dataset **in the order provided**. Do not randomize the sample presentation order for the first epoch. List all of the network weight and bias terms after the first epoch using

the same table format as used above. Limit your reported precision to 4 decimal places. Also, calculate and report the average error energy of all samples after the first epoch.

2) Continue to train the network until the change in average error energy is less than 0.001. Randomize the presentation order of the samples for each epoch after the first. Plot the average error energy per epoch and draw the final decision boundary on the cross dataset.

3) Retrain the network using new random initial weights and vary the learning rate and momentum term. What effects do the learning rate and momentum term have on the algorithm? What happens if you set all of the initial weights and bias terms to zero?

Even though the experiment in Part A deals with a specific 2:10:1 MLP, your program should be general and parameterized to be able to handle up to 20 dimensional input data, 20 hidden neurons, and 10 output neurons.

<u>Part B:</u>

Down load the data set Two_Class_FourDGaussians500.txt from Canvas.

Here is its description:

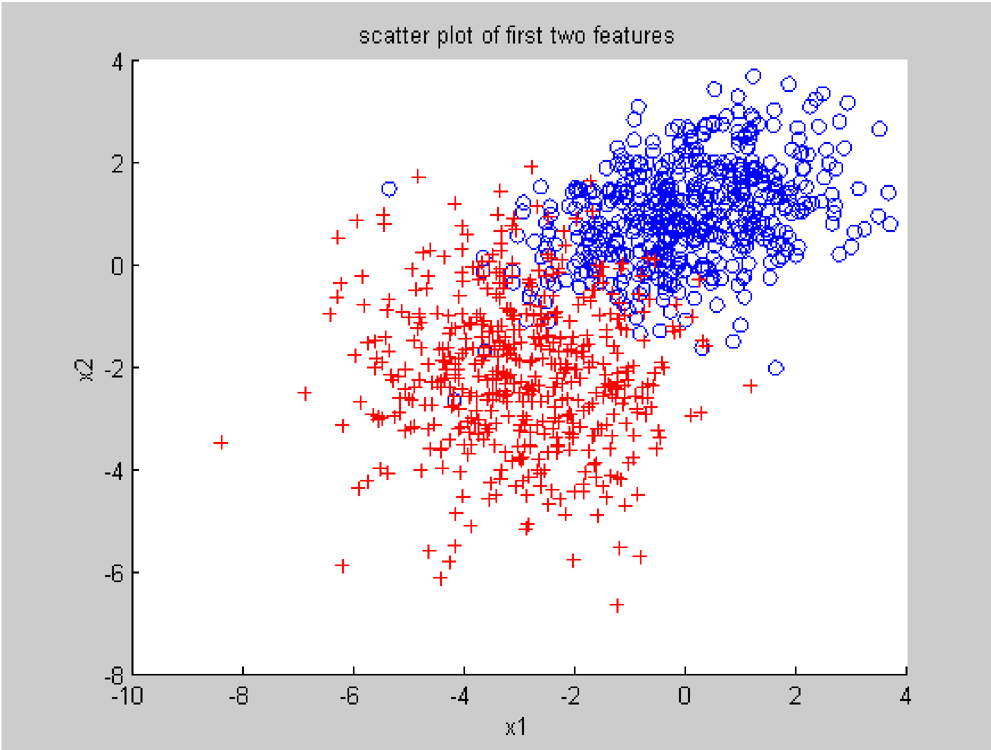
Two 4-D Gaussians with 500 points

Input parameters:

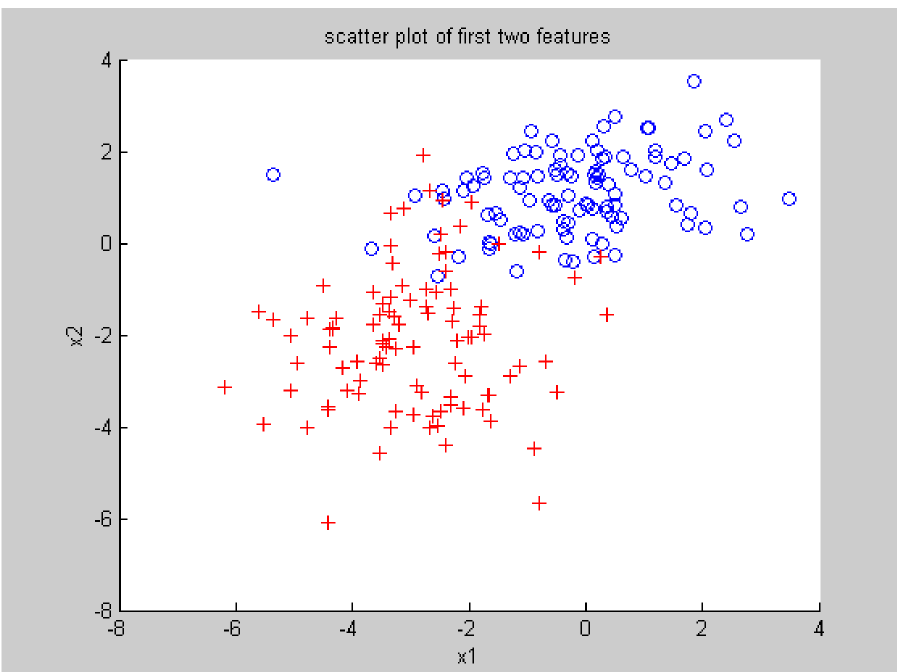
```
R1=[2 0.5 0 0; 0.5 1 0 0; 0 0 3 0; 0 0 0 1];   % covariance matrix1
mu1=[0 1 0 -1]';         % mean1
R2=[2 0 -1 1; 0 2 1 0; -1 1 4 0; 1 0 0 1];   % covariance matrix2
mu2=[-3 -2 -1 -1]';         % mean2
```

Recovered Means and Covariances

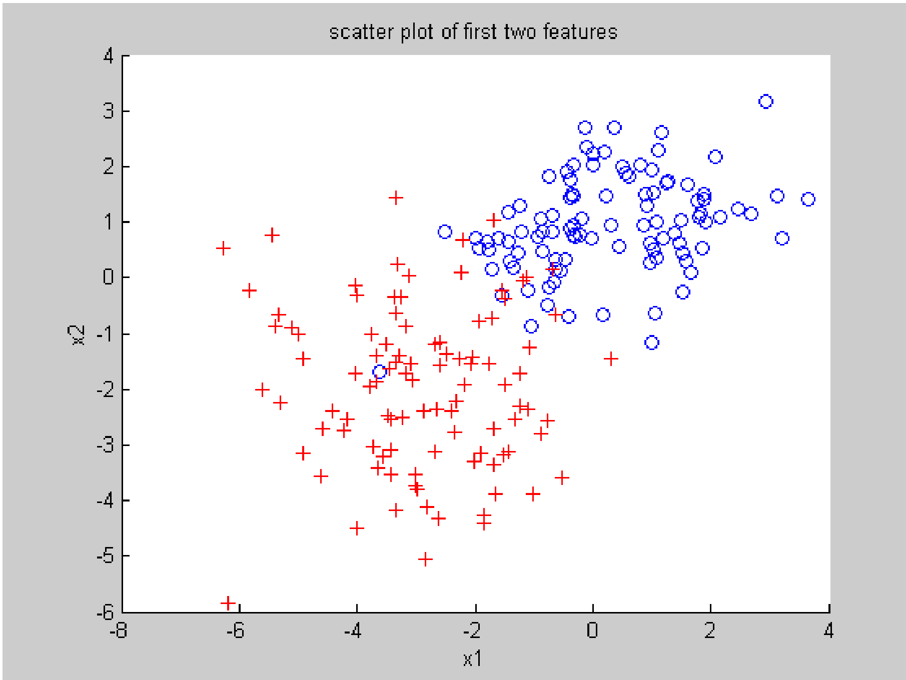
```
 Mu1:   0.0152    0.9943    0.0229   -1.0568

R1:
    1.9996    0.4948   -0.0627   -0.0424
    0.4948    0.9676    0.0861   -0.0054
   -0.0627    0.0861    3.3205    0.0259
   -0.0424   -0.0054    0.0259    1.0823

Mu2:  -2.9729   -2.0617   -0.9958   -0.9631

R2:
    2.0715   -0.1031   -1.1522    1.0122
   -0.1031    2.0908    1.0931   -0.0382
   -1.1522    1.0931    4.4312    0.0716
    1.0122   -0.0382    0.0716    1.0218
```



Data points 1 – 100 (Testing)



Data points 101-200 (Validation)



Using the **general code** developed in Part A, construct a MLP of structure 4:N:2, where N is the number of hidden units. You should leave the first 100 points of each class out of the training. You can use points 101-200 as an epoch validation set during training if you wish (or add them to the training data if you only want to use SSE for termination. Once your network has converged, test it with the first 100 points of each class.

This is the part to show your creativity.

For this assignment, print the results, and analyze how close the results come to the expected answers. **<u>This is the most important part of the assignment!</u>** Be sure to design your program in a general, well structured fashion, and document the code appropriately.

Your report should contain a sections on

1. The technical description of all techniques utilized,
2. The design of the algorithms (pseudo-code, flowcharts, or some other structured descriptive means),
3. The results of the algorithms,
4. An **analysis** of the results, i.e., did you obtain what you expected? Were there any surprises? What conclusions can you draw from the experiments? etc.
5. Well documented, structured, modular program listings.

Be sure to design your program in a general, well structured fashion, and document the code appropriately.