SMAI ASSIGNMENT 2

KARAN MANGLA
201301205

Q1. Convergence Proof for Single Sample Perceptron

SMAI Assignment 2

Karan Mangla
201301205

Q1. Proof of convergence for single sample perception

Thm: if the given samples are linearly separable, the algorithm will converge and return a solution vector.

Proof: we know, $a(k+1) = a(k) + y^k$ — (1)

where y^k is the misclassified sample being considered

and let a_2 be the solution vector after convergence

$$\Rightarrow a_2^T y_i > 0 \quad \forall i.$$

take (1) and a scale factor α

$$a(k+1) - \alpha a_2 = (a(k) - \alpha a_2) + y^k \quad \text{— (2)}$$

square (2)

$$(a(k+1) - \alpha a_2)^2 = ((a(k) - \alpha a_2) + y^k)^2$$

$$= (a(k) - \alpha a_2)^2 + \|y^k\|^2 + 2(a(k) - \alpha a_2)^t y^k$$

since y^k is missclassified

$$a^t(k) \cdot y^k <= 0.$$

$$\Rightarrow a(k+1) - \alpha a_2)^2 \leq (a(k) - \alpha a_2)^2 + 2\alpha a_2^t y^k + \|y^k\|^2.$$

Now, let us take β as max pattern vector,

$$\beta = \max \|y_i\|^2$$

and $\gamma = \min(a_2^t y_i)$

So, our eq^n becomes $\|a(k+1) - \alpha a_2\|^2 \leq \|a(k) - \alpha a_2\|^2 - \beta$.

in each iteration, value of LHS is reduced by k atleast.

$\Rightarrow \|a(k+1) - \alpha a_2\|^2 \leq \|a(1) - \alpha a_2\|^2 - k\beta$

since LHS (norm) can't be < 0.

so there must be atmost k_0 corrections,

$$0 = \|a(1) - \alpha a_2\|^2 / \beta$$

since k_0 is finite (depends on $a(1)$, α, a_2 (which exists).

so the algorithm will converge.

Hence proved.

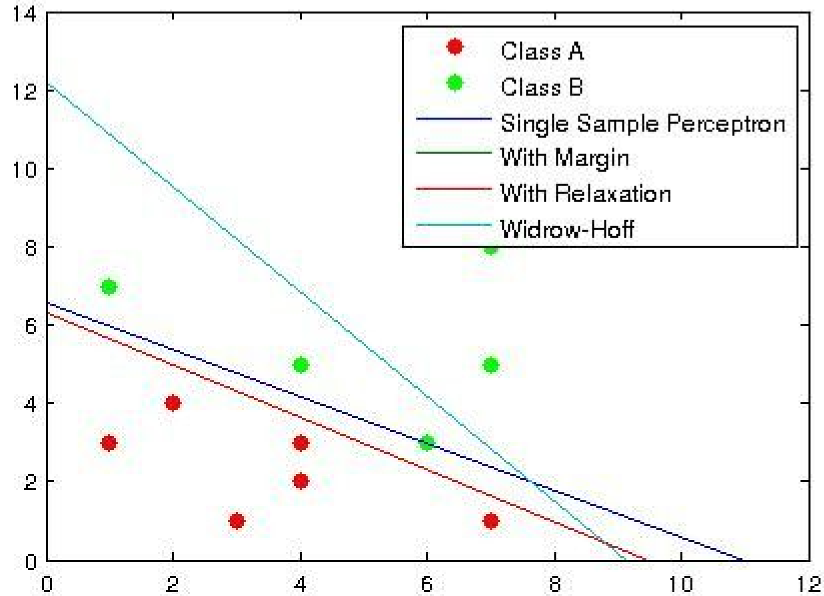
Q2.

a1 = [2;1;1]
a2 = [-1001;1;-2]
a3 = [5;5;5]
a4 = [-1;-1;-1]

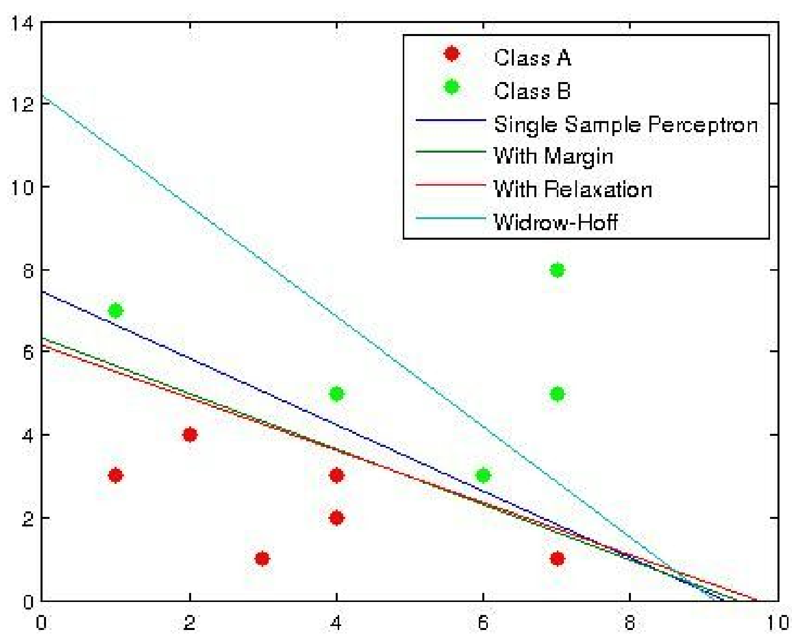
Number of iterations taken to converge for different initial weight vectors at same b value(100) :

SR NO.	INITIAL WEIGHT	SINGLE SAMPLE	WITH MARGIN	WITH RELAXATION	WIDROW-HOFF
A	[2;1;1]	193	7124	43548	1492730
B	[-1001;1;-2]	23	44	1139	15348926
C	[5;5;5]	208	7129	43556	948998
D	[-1;-1;-1]	184	7117	43547	1781006

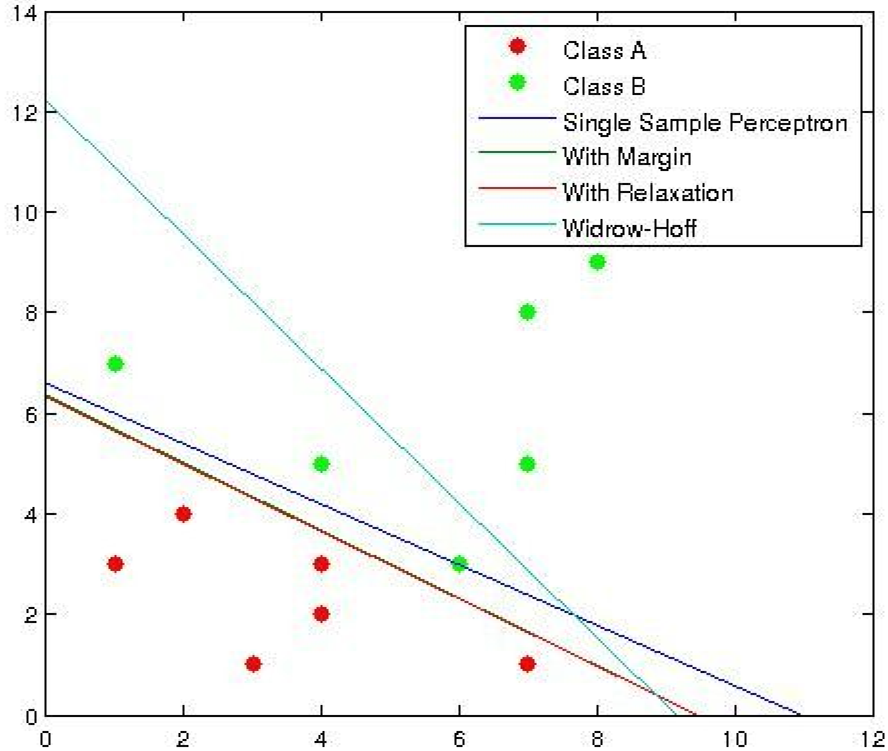
A	[2;1;1]	193	7124	43548	1492730



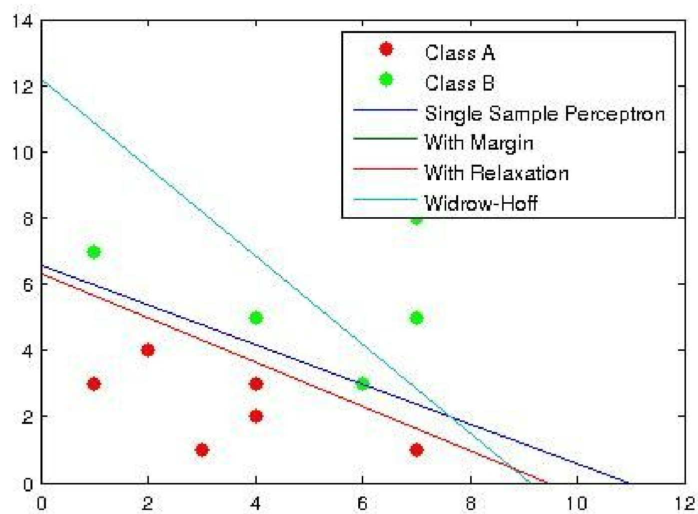
B	[-1001;1;-2]	23	44	1139	15348926



C	[5;5;5]	208	7129	43556	948998



D	[-1;-1;-1]	184	7117	43547	1781006



Number of iterations taken to converge for different initial weight vectors with different margins (b) -

- Single sample perceptron with margin

Initial weight Vector	b = 0.1	b = 1	b = 5	b = 10	b = 100	b = 1000
[2;1;1]	235	235	330	801	7124	70292
[-1001;1;-2]	23	23	23	23	44	66791
[5;5;5]	216	216	422	802	7129	70307
[-1;-1;-1]	226	226	412	790	7117	70207

- Relaxation algorithm with margin

Initial weight Vector	b = 0.1	b = 1	b = 5	b = 10	b = 100	b = 1000
[2;1;1]	29701	34225	37458	38869	43548	48243
[-1001;1;-2]	2	2	2	2	1139	48179
[5;5;5]	29756	34248	37472	38873	43556	48243
[-1;-1;-1]	25423	34116	37437	38857	43547	48243

Q3 .

Processing(downscaling) : First the 32X32 samples were downscaled to 8X8 . For this the representative of each 4X4 matrix is taken as the sum of all the elements . Then as we had to give only 0 and 7 as input to our NN and recognize them so we filtered these out from our training
data .

Features = 64
Classes = 2(0 and 7)

Training :

There are 2 output units of each sample data unit . If the digit is ‘7’, then output units would be [0 1], and if the digit is ‘0’, then output units would be [1 0].
We could have normalized the data , but added the bias unit X_0=1 so the need to normalize is now away .

- 1st layer (dimensions of input) - 65X1
- 2nd layer (or the hidden layer) has 73 units keeping in mind m/10 where m is the number of training samples . I tried with 16 and 32 number of hidden layers but the accuracy came best at m/10 number of samples and thus the number .
- 3rd layer consists of 2 output units which is either [0 1] or [1 0] according to the actual input .

The weights between layers I and II are called wij which is between ith feature in first layer and jth hidden unit in second layer. So the size is 65X73.

The weights between layers II and III are called wjk which is between jth hidden unit in second layer and the kth output unit in third layer. So the size is 73X2.

In our case, first layer has 65 entities, second layer 73, and third layer 2.

Initializations -

It was essential to initialize the weights randomly within a specified range, otherwise the activation function (logsig) would saturate.

-1/root(features) < wij < -1/root(features) and -1/root(hidden_units) < wij < -1/root(hidden_units)

eta = 1

Sigmoid function was chosen as the activation function because it is :

nonlinear , saturates, continuous and smooth, defined, monotonic.

Verdict

The accuracy came to about 100% on most runs.