# Statistical Methods in AI

Gokul B. Nair - 201502034

## 1 Problem 1

The required implementation for each of the four sub-divisions can be found in 'q1.py'.

### 1.1 Single Sample Perceptron without margin

The following has been taken to follow by the Perceptron Algorithm:

$$Case1 : W^T(k)X(k) \leq 0, \quad y(k) = 1$$

$$Case2 : W^T(k)X(k) > 0, \quad y(k) = 0$$

The update rule followed while training the data is as follows:

$$Case1 : W(k+1) = W(k) + X(k)$$

$$Case2 : W(k+1) = W(k) - X(k)$$

For ease of programming, the following complete vector negation step is carried out for each feature vector that has label '0' so that the final update rule can be shrinked to case 1 alone in the rule shown above.
It is observed that **convergence is reached at** $13^{th}$ **epoch**. In the sample training data used, 100% Accuracy is achieved.

### 1.2 Single Sample Perceptron with margin

Here, in addition to the above code, a margin of size 'wid' is introduced. For various values of 'wid', convergence was achieved in the $n^{th}$ epoch. The observed values of 'n' are:

$$wid = 0, \quad n = 13$$

$$wid = 100, \quad n = 13$$

$$wid = 10000, \quad n = 21$$

### 1.3 Batch Perceptron

#### 1.3.1 Without margin

Here, the same implementation is used as that in subdivision 1.1 above with the only difference being that the updation is made on a bulk basis after each epoch with the appropriate set of vectors that has been accumulated in a 'temp' vector.
It is observed that **convergence is reached at** $21^{st}$ **epoch**. In the sample training data used, 100% Accuracy is achieved.

#### 1.3.2 With margin

This is similar as the previous subsection with the only difference that margin of various width, 'wid' values are3 considered. Convergence was achieved in the $n^{th}$ epoch. The observed values of 'n' are:

$$wid = 0, \quad n = 21$$
$$wid = 10, \quad n = 21$$
$$wid = 10000, \quad n = 21$$
$$wid = 1000000, \quad n = 22$$
$$wid = 10000000, \quad n = 16$$
$$wid = 100000000, \quad n = 17$$

## 2 Problem 2

The required implementation for each of the four sub-divisions can be found in 'q2.py'.

### 2.1 Relaxation Algorithm + Margin

The same algorithm is used as that under subsection 1.2 with the only defference being in the updation equation used which is the following:

$$W(k+1) = W(k) + ((wid - W^T(k)X(k)) * X(k))/||X(k)||^2$$

This was carried out for various values of margin, 'wid' and various number of epochs. The observations are notes as follows:

$$win = 10, \quad epochs = 10, \quad Reliable\ solution\ not\ reached.$$

$$win = 100, \quad epochs = 100, \quad Final\ solution\ was\ reached.$$

$$win = 1000, \quad epochs = 1000, \quad Final\ solution\ was\ reached.$$

### 2.2 Modified Perceptron Algorithm

We notice that the data set provided here is not linearly seperable. Hence, convergence is not reached. Hence, Gradient Descent like part has been implemented to obtain the best possible solution. For each epoch, the number of updations made is stored. The solution weight vector 'W' corresponding to the epoch with least number of updations is chosen and used for testing. The observations made are as follows:

$$win = 10, \quad Total\ epochs = 100, \quad Best\ Solution\ achieved\ at\ 96^{th}\ epoch.$$

$$win = 10, \quad Total\ epochs = 1000, \quad Best\ Solution\ achieved\ at\ 930^{th}\ epoch.$$

$$win = 10, \quad Total\ epochs = 10000, \quad Best\ Solution\ achieved\ at\ 1113^{th}\ epoch.$$

## 3 Problem 3

## 4 Problem 4