CS446: Machine Learning **Spring 2017**

Problem Set 1

Zhiyu Zhu *Handed In: February 4, 2017*

1. Answer to problem 1

   a. In order to test all the hypothesis with the training sample data, we can test the algorithm with all possibility terms:
   $(x_0 = 0) \wedge (x_1 = 1) \wedge (x_1 = 0) \wedge (x_1 = 1) \wedge \ldots \wedge (x_n = 0) \wedge (x_n = 1)$
   Then we can go through the algorithm by iteration. For example, if we see from a positive algorithm, and in that specific algorithm, we have $(x_0 = 1)$, then it is the negative of $(x_0 = 0)$. The hypothesis that $(x_0 = 0)$ will then be cross out and deleted from the hypothesis. In this way, any sample spaces that $(x_i = a)$ can be used to cross out the hypothesis that $(x_i = \neg a)$. If both $(x_i = a)$ and $(x_i = \neg a)$ are cross out, then x_i is not included in the algorithm. After finishing all the iterations on training data, the algorithm is constructed by all the hypothesis left.

   b. To prove the algorithm, we need two steps to test:
   First: we do the iterations over all the training samples in order to find a hypothesis that is consistent with the whole sample spaces.
   To achieve the above, we can use the approach in part a that any sample spaces that $(x_i = a)$ can be used to cross out the hypothesis that $(x_i = \neg a)$. In this way, when iterations on all the sample spaces and training data are finished, we can get a hypothesis h_a.
   Second, we need to test this hypothesis output is correct which means that it really consistent with all the sample spaces.
   To exam the hypothesis h_a, we need to do the iteration again. Whenever, we found a term in h_a that shows $(x_i = a)$ though there is a negative training data that $x_i = \neg a$, we can immediately end all the iteration and goes to the conclusion that t NO CONSISTENT hypothesis exits. On the other hand, if all the iteration finishes and there is not term in the h_a that disagree with the sample space, then we can assume that $h = h_a$

   c. We do iteration n times on each training data spaces since there are n variables in total. Also, we test each training data twice because we need to test both about positive hypothesis terms $(x_i = a)$ and $(x_i = \neg a)$, so we need to do totally $2 \times n \times m$. So the running time is:
   $O(2 \times n \times m) = O(nm)$

   d. There are two conditions.
   First, if the hypothesis generated from above algorithm tested by the new

example gives a positive results. Then we can say that all the terms given by the new example are correctly labeled by the hypothesis.
Second, on the other hand if it gives a negative result, then there are two possible answer, maybe it is negatively labeled by the hypothesis or the training data set are not large enough to delete the variable from the hypothesis(ignore both $(x_i = a)$ and $(x_i = \neg a)$ terms). So, if we see a negative result, we need a lot more information to judge if it is negatively labeled by our hypothesis.

2. Answer to problem 2

   a. we can know that the shortest distance from a point to a plane is a vector started from that point, ended on that plane and also orthogonal to the plane. For the vector that is orthogonal to $\vec{w}^T$ is $\vec{w}$.
   Suppose there is a point $\vec{x}_m$ on the hyper-plane that satisfy$\|\vec{x}_m - \vec{x}_0\|$ is the minimum distance from $\vec{x}_0$ to the hyper-plane $\vec{w}^T\vec{x} + \theta = 0\}$. Then the vector $vecx_m - \vec{x}_0$ should be in the same direction of unit vecotr of $\vec{w}$
   In this way, we need to use the dot product:
   $\vec{x}_m - \vec{x}_0 = \pm\|\vec{x}_m - \vec{x}_0\|\frac{\vec{w}}{\|\vec{w}\|}$
   since we want to compute $\vec{x}_m$ onto the hyper-plane, we can use the dot product with $\vec{w}^T$ to both sides:
   $\vec{w}^T\vec{x}_m - \vec{w}^T\vec{x}_0 = \pm\|\vec{x}_m - \vec{x}_0\|\frac{\vec{w}^T\cdot\vec{w}}{\|\vec{w}\|}$
   also, we can know that $\vec{w}^T \cdot \vec{w} = \|\vec{w}\|^2$ and $\vec{x}_m$ on the hyper-plane so that $\vec{w}^T\vec{x} + \theta = 0$, then the formula is translated into:
   $\vec{w}^T\vec{x}_0 + \theta = \pm\|\vec{x}_m - \vec{x}_0\|\frac{\|\vec{w}\|^2}{\|\vec{w}\|}$
   then the final simplification is:
   $\|\vec{x}_m - \vec{x}_0\| = \frac{|\vec{w}^T\vec{x}_0+\theta|}{\|\vec{w}\|}$

   b. we can know from part a that shortest distance from a plane to a plane is a vector started from that plane, ended on that plane and also orthogonal to both two plane. For the vector that is orthogonal to $\vec{w}^T$ is $\vec{w}$.
   Suppose there is a point $\vec{x}_m$ on the first hyper-plane and a point $\vec{x}_n$ on the second hyper-plane that satisfy$\|\vec{x}_m - \vec{x}_0\|$ is the minimum distance between two hyper-plane. Then the vector $vecx_m - \vec{x}_n$ should be in the same direction of unit vecotr of $\vec{w}$
   In this way, we need to use the dot product:
   $\vec{x}_m - \vec{x}_n = \pm\|\vec{x}_m - \vec{x}_n\|\frac{\vec{w}}{\|\vec{w}\|}$
   with the function of the hyper-planes:
   $\vec{w}^T\vec{x}_m - \vec{w}^T\vec{x}_n = \pm\|\vec{x}_m - \vec{x}_n\|\frac{\vec{w}^T\cdot\vec{w}}{\|\vec{w}\|}$
   after the simplification of the formula above and the definition that :
   $\vec{w}^T\vec{x} = \theta_1$ and $\vec{w}^T\vec{x} = \theta_2$
   so the final answer is that:
   $\|\vec{x}_m - \vec{x}_n\| = \frac{|\theta_1-\theta_2|}{\|\vec{w}\|}$

3. Answer to problem 3.a

a.1 First question

a.1.1 first of all, assume there exits a hyper-plane that satisfies the separability and has the equation that:
$\vec{a}^T\vec{x} + b$
since it separates all the $y = 1$ and $y = -1$, there must be two point that are closest to the hyper-plane and satisfies:
a minimun of the positive data $\min_{y=1}(\vec{a}^T\vec{x} + b) \geq 0$ when $\vec{x} = \vec{x_+}$ on plane D
and a maximun if the negative data: $\max_{y=1}(\vec{a}^T\vec{x_-} + b) < 0$ when $\vec{x} = \vec{x_-}$ on plane D
since that $(\vec{a}^T\vec{x_+} + b) \geq 0 > (\vec{a}^T\vec{x_-} + b)$, there must be a value c satisfies that the hyper-plane are between $\vec{x_+}$ and $\vec{x_-}$.
In order to find the plane with the formula generate in problem 2.b to find the function of plane, we can assume a value c that give the plane $\vec{a}^T\vec{x} + b - c$ that satisfies:
$(\vec{a}^T\vec{x_+} + b) - c \geq 0 > (\vec{a}^T\vec{x_-} + b) - c$ and the plane $\vec{a}^T\vec{x} + b$ is equal distance to $\vec{x_-}$ and $\vec{x_+}$.
by plugging in the formula in problem b.2:
$\frac{(\vec{a}^T\vec{x_+}+b)-c}{\|\vec{a}\|} = -\frac{(\vec{a}^T\vec{x_-}+b)-c}{\|\vec{a}\|}$
therefore, by multiplying by $\|\vec{a}\|$ on both side:
$c = \frac{(\vec{a}^T\vec{x_+}+b)+(\vec{a}^T\vec{x_-}+b)}{2}$
since when $y = 1$, $(\vec{a}^T\vec{x_+} + b) - c \geq \frac{(\vec{a}^T\vec{x_+}+b)+(\vec{a}^T\vec{x_-}+b)}{2}$
and when $y = -1$, $-1(\vec{a}^T\vec{x_+} + b) - c \geq -1\frac{(\vec{a}^T\vec{x_+}+b)+(\vec{a}^T\vec{x_-}+b)}{2}$
we can conclude that there exist a $\vec{w}^T$ and θ that satisfies:
$y(\vec{w}^T\vec{x} + \theta) \geq 1 - \delta$ when $\delta = 0 \qquad \forall(\vec{x_i}, y_i) \in D$

a.1.2 according to the definition of the linear program, we can see that:
$y_i(\vec{w}^T\vec{x_i} + \theta) \geq 1 - \delta, \qquad \forall(\vec{x_i}, y_i) \in D$
as we set $\delta = 0$, the formula is changed to:
$y_i(\vec{w}^T\vec{x_i} + \theta) \geq 1, \qquad \forall(\vec{x_i}, y_i) \in D$
then simply plugging in the y_i value we can get¡ that:
$(\vec{w}^T\vec{x_i} + \theta) \geq 1, \qquad \forall(\vec{x_i}, y_i) \in D$ when $y_i = 1$
$(\vec{w}^T\vec{x_i} + \theta) \leq< -1, \qquad \forall(\vec{x_i}, y_i) \in D$ when $y_i = -1$
which means that D is linearly separable

a.1.3 there are two conditions:
first $1 - \delta \geq 0$ means that $0 \leq \delta \leq 1$ and there is a possible hyper plane that can prove the separability of data set D
second $1 - \delta < 0$ means that $0 \leq \delta \leq 1$ and the hyper plane cannot separability data set D.

a.2 according to the conclusion we reach in problem a.1.3, if we have this new linear program, then we have to make $-\delta \geq 0$ means that $\delta \leq 0$ to guarantee that there is a hyper plane can prove the separability of the data set D. And according to the definition of the linear program: $\delta \geq 0$. According to those two limitation, we have to make $\delta = 0$ and reaches the solution that:
$\vec{w} = 0\ \theta = 0\ \delta = 0$

a.3 if we plug in $\vec{x_+}$ for $y_i = 1$, we can get a relationship that:
$y_i(\vec{w_1}\vec{x_1} + \vec{w_2}\vec{x_2} + \vec{w_3}\vec{x_3} + \ldots + \vec{w_n}\vec{x_n} + \theta) = \vec{w_1} + \vec{w_2} + \vec{w_3} + \ldots + \vec{w_n} + \theta \geq 1$ when $\delta = 0$ and $y_i = 1$
also$-(-\vec{w_1} - \vec{w_2} - \vec{w_3} - \ldots - \vec{w_n} + \theta) = (\vec{w_1} + \vec{w_2} + \vec{w_3} + \ldots + \vec{w_n} - \theta) \geq 1$ when $\delta = 0$ and $y_i = -1$
since we do not know if θ is positive or negative, we can result in the following two equation:
$(\vec{w_1} + \vec{w_2} + \vec{w_3} + \ldots + \vec{w_n}) \geq 1 + |\theta|$
and $(\vec{w_1} + \vec{w_2} + \vec{w_3} + \ldots + \vec{w_n}) \geq 1 - |\theta|$
since $1 + |\theta| \geq 1 + |\theta|$:
the optimal solution for $\delta = 0$ is:
$(\vec{w_1} + \vec{w_2} + \vec{w_3} + \ldots + \vec{w_n}) \geq 1 + |\theta|$

b. Answer to problem 3.b

b.1 In this part of the problem, we can see that we need to first set up all the variables:
A of size (m + 1)*(n + 2) and setting all the delta to zero
$\vec{c}$ are all zero except last entries representing delta equals to 1
$\vec{b}$ all entries set to zero
then the iterations are done to find the product and the result of the all optimization variables: $\vec{w}$ θ δ
Then, the code is above:

```
function [w,theta,delta] = findLinearDiscriminant(data)
%% setup linear program
[m, np1] = size(data);
n = np1-1;
% write your code here
A = zeros(m+1,n+2);
b = zeros(m+1,1);
c = zeros(n+2,1);
% first creat all variables and set all entries to zero
c(1:n,1) = 0;
% set all w to zero
c(n+1,1) = 0;
% set threshold to zero
c(n+2,1) = 1;
% set delta to 1
y = data(1:m,n+1);
for i = 1:m
        A(i, 1:n) = y(i)*data(i,1:n);
end
% find all the products
A(1:m,n+1) = y;
% set theta
A(1:m,n+2) = 1;
%set delta
A(m+1,n+2) = 1;
%set delta

b(1:m,1) = 1;
% set the algorithm
b(m+1,1) = 0;
% set delta > 0

%% solve the linear program
%adjust for matlab input: A*x <= b
[t, z] = linprog(c, -A, -b);

%% obtain w,theta,delta from t vector
w = t(1:n);
theta = t(n+1);
delta = t(n+2);

end
```

b.2 In order for finding the monotone conjunction which means $x_1 \wedge x_2$, only last line gives a positive result:

hw1sample2d.txt

```
0 0 -1
0 1 -1
1 0 -1
1 1 1
```

Then we uses the code to plot out the negative and positive data with range from -10 to 10 and step wideth 1.
in addition the separation plane is also ploted:

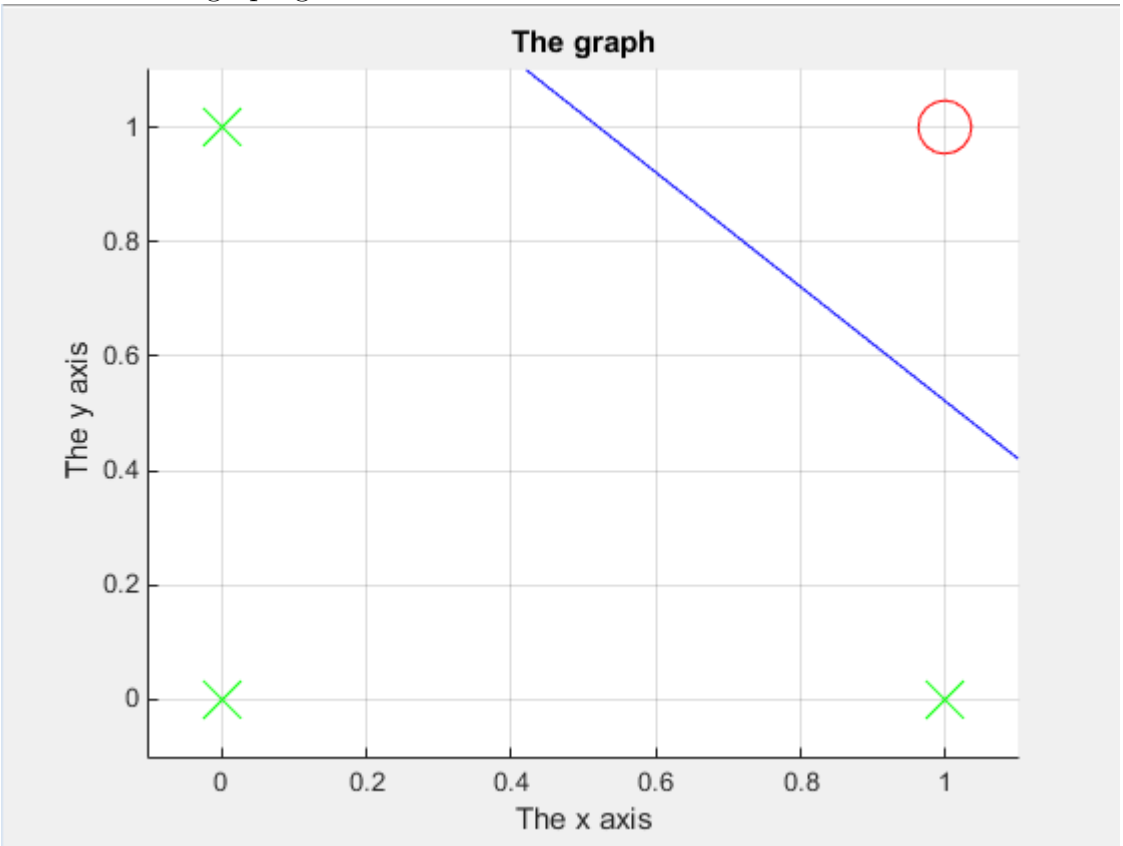
plot2dSeparator.m | hw1sample2d.txt | findLinearDiscriminant.m

```
% This function plots the linear discriminant.
% YOU NEED TO IMPLEMENT THIS FUNCTION

function plot2dSeparator(w, theta)

tempx = -10:1:10;
y = -w(1)/w(2)*tempx - theta/w(2);
plot(tempx, y, 'b');

end
```

With the result generated by the program, the optimization data is shown below:

```
w =

    2.9104
   -2.0498
    0.1775
  190.5196
    0.1399
   -3.1010
   -2.9534
 -193.2778
    1.1679
   -8.8942

theta =

  -90.2115


delta =

  -2.4158e-13
```

Then the final graph generated is:



b.3 Since we want it becomes more accurate in the result:

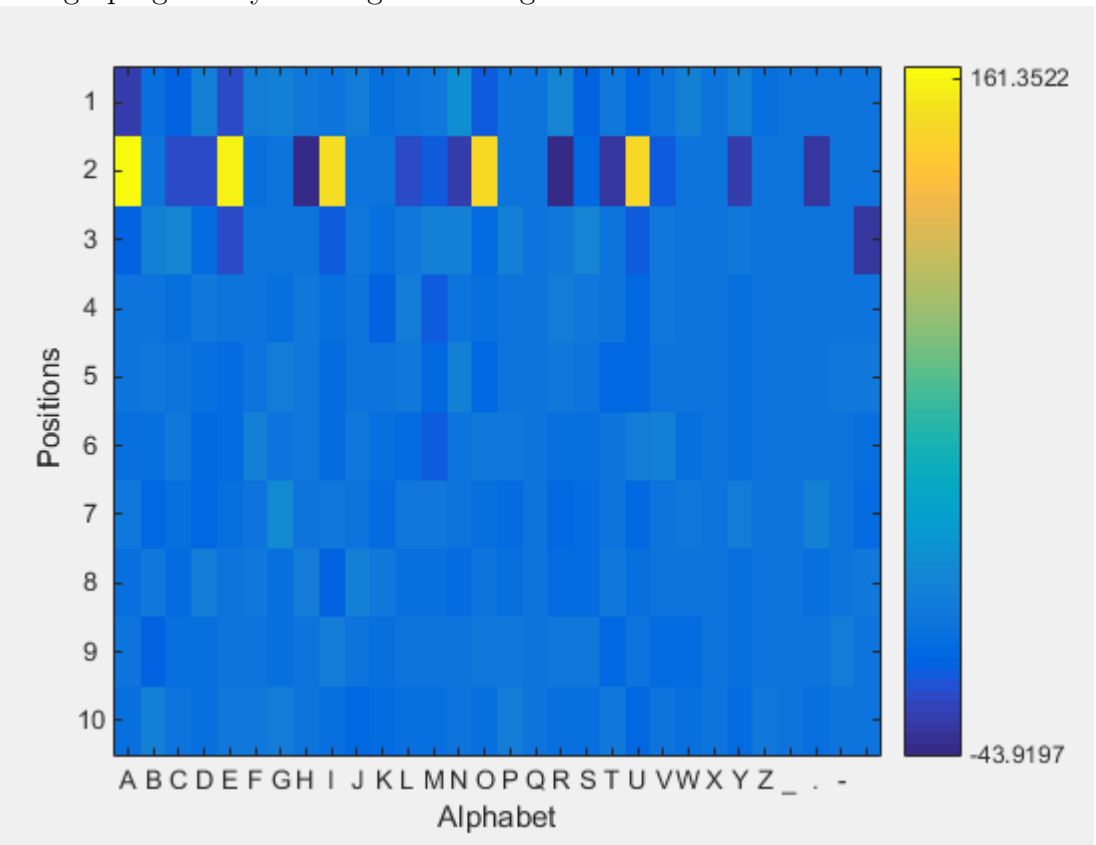
computeLabel.m | plot2dSeparator.m | hw1sample2d.txt | findLinearDiscriminan

```
% This function computes the label for the given
% feature vector x using the linear separator represented by
% the weight vector w and the threshold theta.
% YOU NEED TO WRITE THIS FUNCTION.

function y = computeLabel(x, w, theta)
if w'*x + theta >= 0
    y = 1;
else
    y = -1;
end
end
```

And the result of the original setting is :

```
Optimization terminated.

delta =

   1.2612e-09


accuracyInTrain =

     1


accuracyInTest =

     1
```

The graph given by the original setting is:



Accoring to the training data given to us, we found that if we put, for example "E" in the second place and increase first element of position, then accuracy in testing data and accuracy in training data will be improved accordingly:

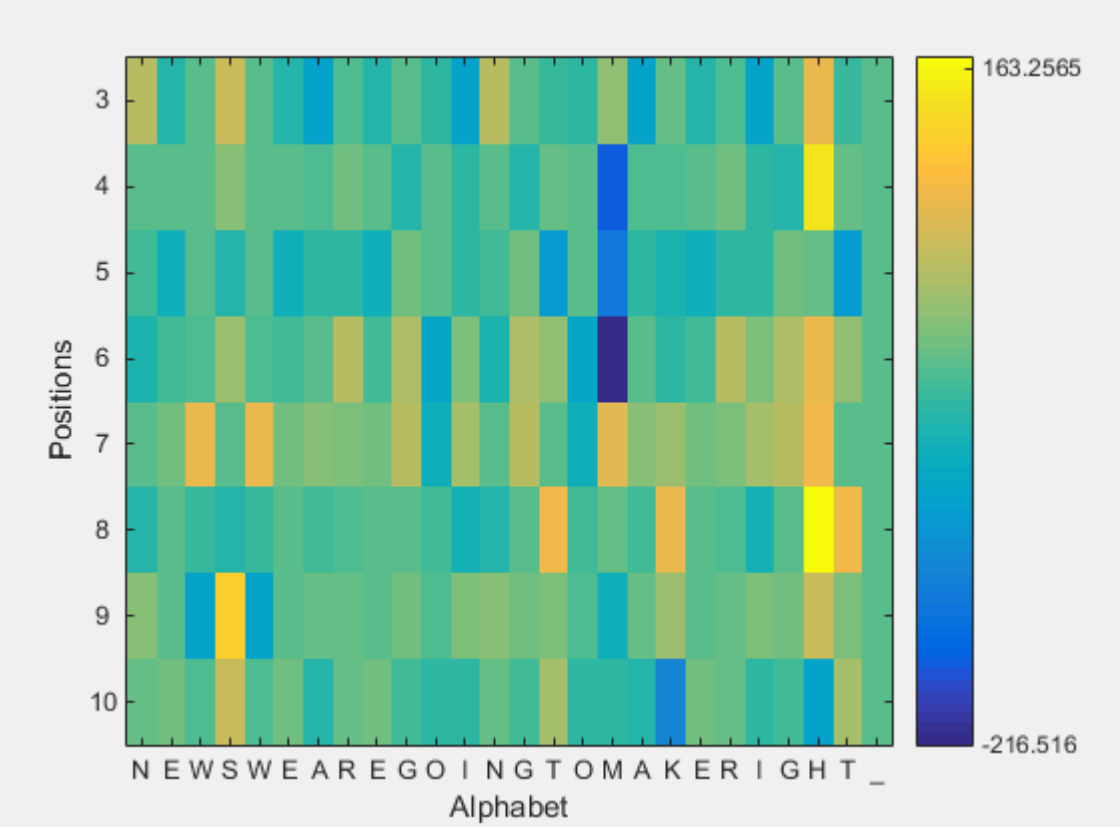
learnConjunctions.m | learnBadges.m | learnMultipleHyperplanes.m | computeAccuracy.m | featu

```
% This script is to help you solve the badge learning problem.
% You may only need to change the first two lines of this file.
% Run this script after
% 1- finishing the implementation of findLinearDiscriminant function
% 2- implementing computeLabel function

%% define the characters of interest (alphabet) and character positions of int
alphabet = 'NEWSWEAREGOINGTOMAKERIGHT_';
positions = 3:10;
```

The result of optimization is:

```
Optimization terminated.

delta =

   5.8081e-09


accuracyInTrain =

     1


accuracyInTest =

    0.7553
```

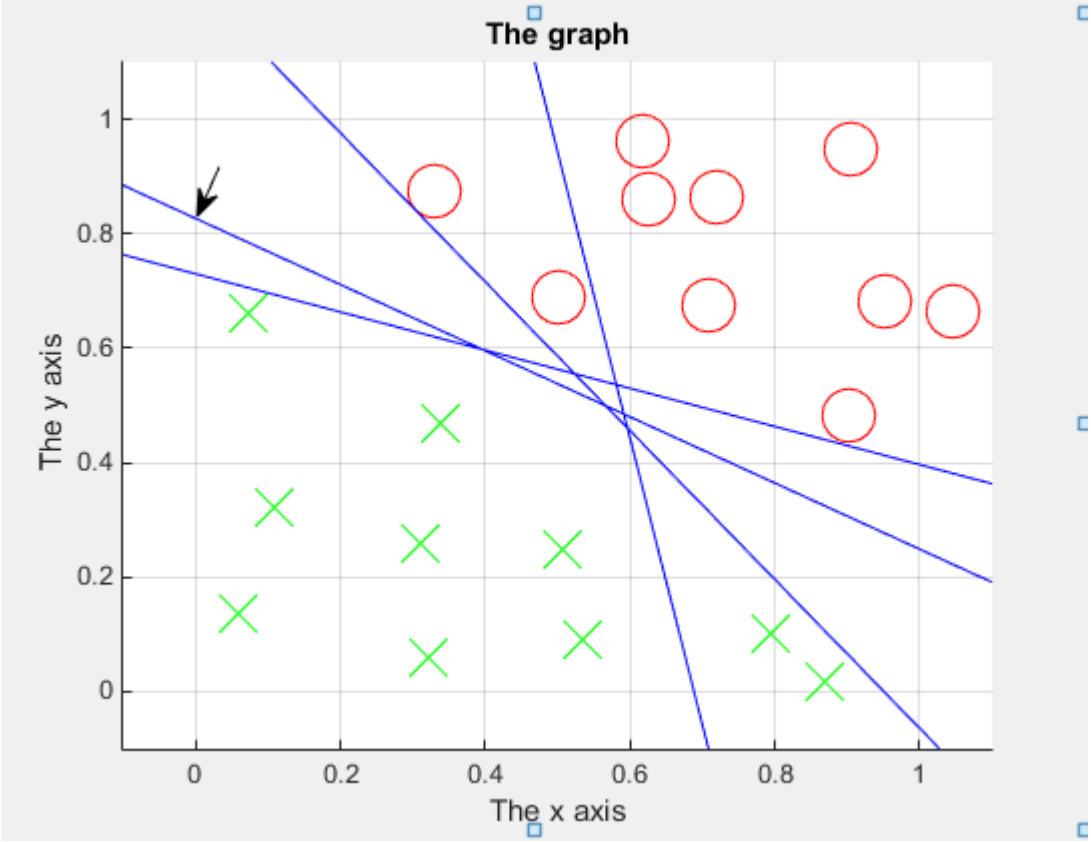
The graph after optimization is:



b.4 The code we used is similar to what we used in b.1:

```
% This function solves the LP problem for a given weight vector
% to find the threshold theta.
% YOU NEED TO FINISH IMPLEMENTATION OF THIS FUNCTION.

function [theta,delta] = findLinearThreshold(data,w)
%% setup linear program
[m, np1] = size(data);
n = np1-1;
% write your code here
A = zeros(m+1,n+2);
b = zeros(m+1,1);
c = zeros(n+2,1);
% first creat all variables and set all entries to zero
c(1:n,1) = 0;
% set all w to zero
c(n+1,1) = 0;
% set threshold to zero
c(n+2,1) = 1;
% set delta to 1
y = data(1:m,n+1);
for i = 1:m
        A(i, 1:n) = y(i)*data(i,1:n);
end
% find all the products
A(1:m,n+1) = y;
% set theta
A(1:m,n+2) = 1;
%set delta
A(m+1,n+2) = 1;
%set delta
b(1:m,1) = 1;
% set the algorithm
b(m+1,1) = 0;
% set delta > 0
%% solve the linear program
%adjust for matlab input: A*x <= b
[t, z] = linprog(c, -A, -b, [], [], [w' -inf -inf], [w' inf inf]);
%% obtain w,theta,delta from t vector
w = t(1:n);
theta = t(n+1);
delta = t(n+2);

end
```

And there are two lines that are either too closed to the positive or negative data, and one of the lines gives the wrong separation of the data that may be generated from the negative result of the $1 - \delta$ value. So only the one pointed by the arrow is prefered in finding the separability of the data set:


The graph
The y axis
The x axis
0
0.2
0.4
0.6
0.8
1