An Introduction to Linear Classifier

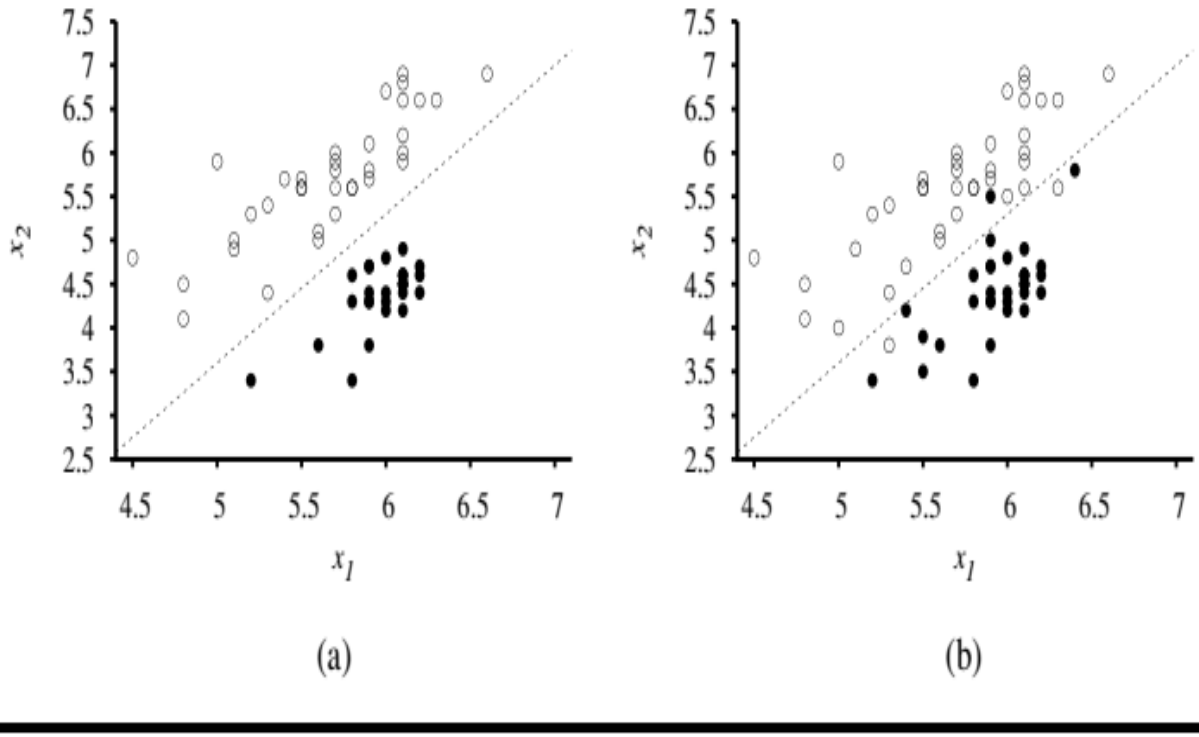

Figure 18.15 (a) Plot of two seismic data parameters, body wave magnitude x_1 and surface wave magnitude x_2, for earthquakes (white circles) and nuclear explosions (black circles) occurring between 1982 and 1990 in Asia and the Middle East (Kebeasy *et al.*, 1998). Also shown is a decision boundary between the classes. (b) The same domain with more data points. The earthquakes and explosions are no longer linearly separable.

A **decision boundary** is a line (or a surface, in higher dimensions) that separates the two classes. In Figure 18.15(a), the decision boundary is a straight line. A linear decision boundary is called a **linear separator** and data that admit such a separator are called **linearly separable**. The linear separator in this case is defined by x2 = 1.7x1 - 4.9 or x2 -1.7x1+4.9 = 0. If

after inserting a data point in x2 -1.7x1+4.9 = 0, the result is >=0, then the data point belongs to a class, otherwise it belongs to the other class. In this example we start with a linear separator w2x2+w1x1+w0 = 0. We initially randomly set the values of w2, w1 and w0. From the given data we learn the values of w2, w1 and w0 using a method called gradient descent method. In this particular example we have found w2 = 1, w1 = -1.7 and w0 =4.9 after applying gradient descent method.

Linear classifiers with a hard threshold

Let the input data consist of n numeric features x1, x2, …..xn and associated class value y =1 or 0. The data belong to two classes (y = 0 represents one class and y =1 represents another class) and can be separated by a linear separator. The linear separator can be expressed by

w1x1+w2x2+w3x3+…….+wnxn+w0 = 0

Equivalently the linear separator can be expressed by vector notation h(**w,x**) = **w.x** = 0 where

w = w1, w2, w3, ……wn, w0 are the coefficients of the linear equation and x = x1, x2, x3, …..xn, x0 are the inputs. Here x0 is always assumed to be 1.

Here we see that we can write the classification hypothesis as h**w**(**x**) = 1 if **w** · **x** ≥ 0 and 0 otherwise.

i.e., if w1x1+w2x2+w3x3+……..+wnxn+w0 >=0 the data belongs to seismic class. If w1x1+w2x2+w3x3+……..+wnxn+w0 <0 the data belongs to nuclear class

we can think of h as the result of passing the linear function **w** · **x** through a **threshold function**: h**w**(**x**) = Threshold(**w** · **x**) where Threshold(**w** · **x**) = 1 if **w** · **x** ≥ 0 and 0 otherwise. The threshold function is shown in the following figure

Learning method

Begin with random weights.

Feed each example (**x**,y) in the equation and update each weight as follows. Here y =1 represents one class and y = 0 represents another class:

$w_i \leftarrow w_i + \alpha (y - h_{\mathbf{w}}(\mathbf{x})) \times x_i$

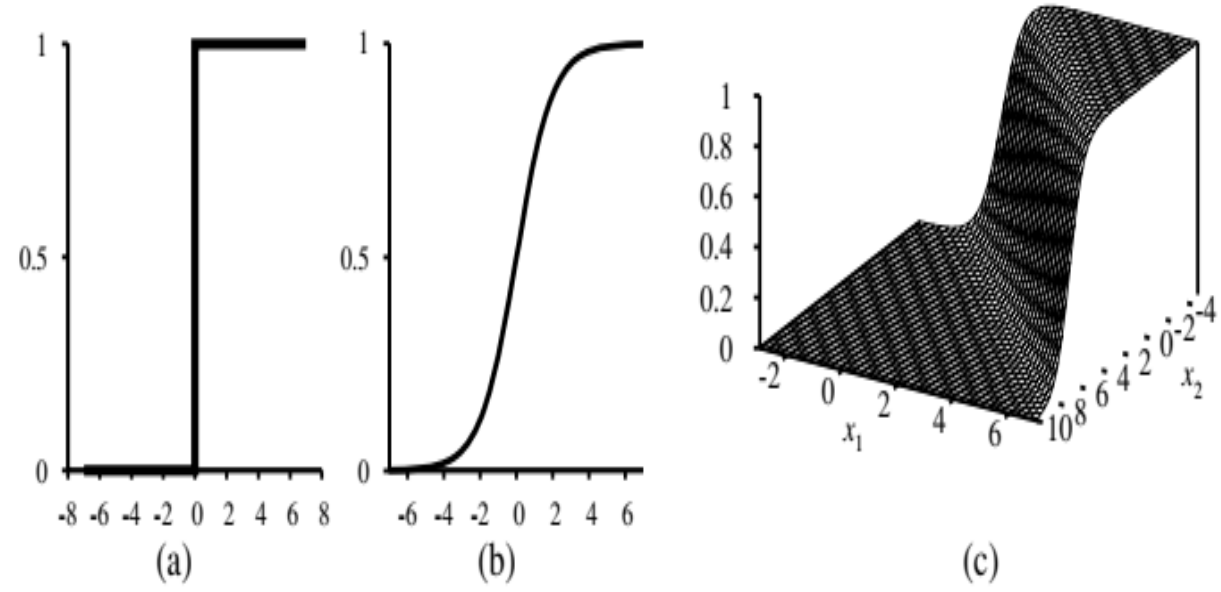

Figure 18.17 (a) The hard threshold function $Threshold(z)$ with 0/1 output. Note that the function is nondifferentiable at $z = 0$. (b) The logistic function, $Logistic(z) = \frac{1}{1+e^{-z}}$, also known as the sigmoid function. (c) Plot of a logistic regression hypothesis $h_{\mathbf{w}}(\mathbf{x}) = Logistic(\mathbf{w} \cdot \mathbf{x})$ for the data shown in Figure 18.15(b).

- If the output is correct, i.e., y = h**w**(**x**), then the weights are not changed.

- If y is 1 but h**w**(**x**) is 0, then wi is *increased* when the corresponding input xi is positive and *decreased* when xi

is negative. This makes sense, because we want to make $\mathbf{w} \cdot \mathbf{x}$ bigger so that h$\mathbf{w}(\mathbf{x})$ outputs a 1.

• If y is 0 but h$\mathbf{w}(\mathbf{x})$ is 1, then wi is *decreased* when the corresponding input xi is positive and *increased* when xi is negative. This makes sense, because we want to make $\mathbf{w} \cdot \mathbf{x}$ smaller so that h$\mathbf{w}(\mathbf{x})$ outputs a 0.

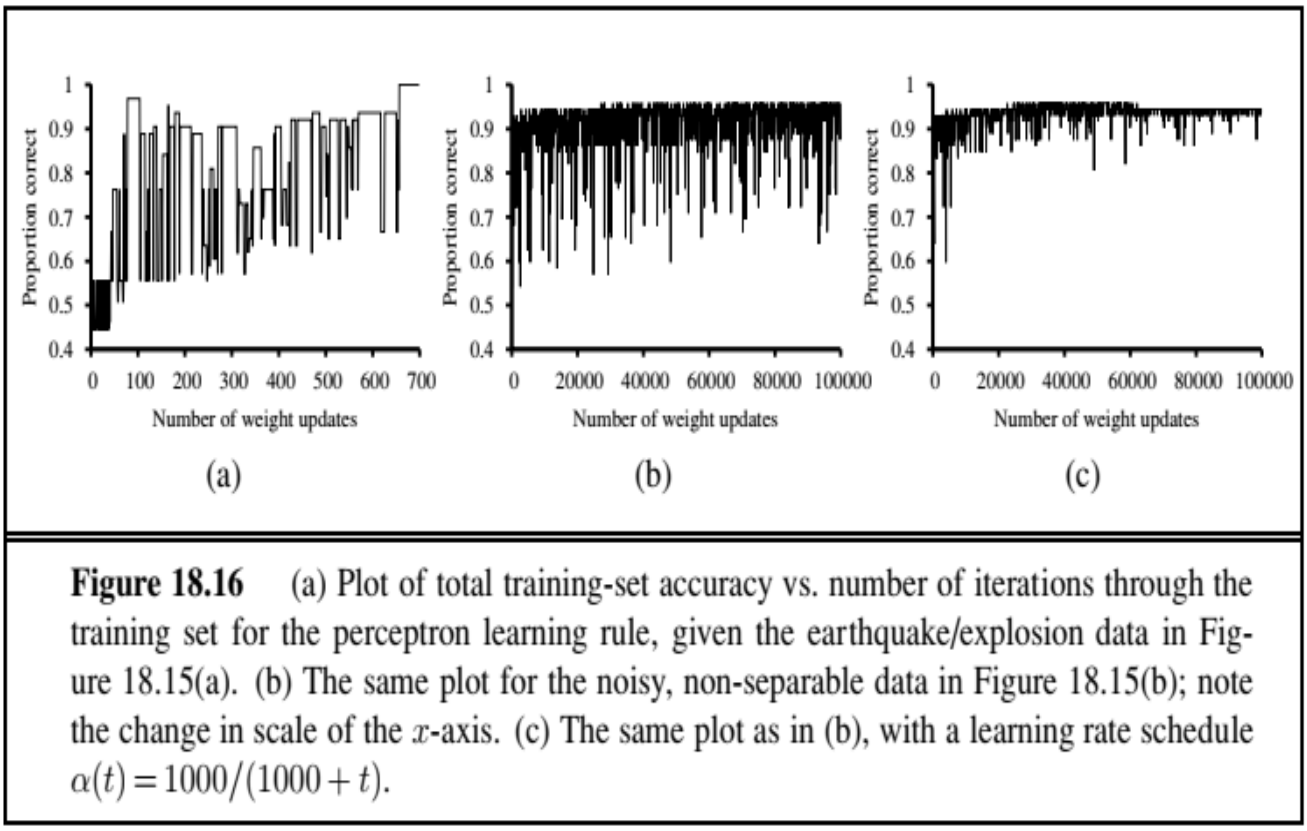

Figure 18.16 (a) Plot of total training-set accuracy vs. number of iterations through the training set for the perceptron learning rule, given the earthquake/explosion data in Figure 18.15(a). (b) The same plot for the noisy, non-separable data in Figure 18.15(b); note the change in scale of the x-axis. (c) The same plot as in (b), with a learning rate schedule $\alpha(t) = 1000/(1000 + t)$.

A complete weight update iteration using all the examples is called a cycle or one epoch.

Linear classification with logistic regression

Logistic(z) = $1/(1 + e^{-z})$ = g(z)

This is differentiable but hard threshold is not differentiable

With the logistic function replacing the threshold function, we now have

$h_{\mathbf{w}}(\mathbf{x}) = \text{Logistic}(\mathbf{w} \cdot \mathbf{x}) = 1/(1 + e^{-\mathbf{w}.\mathbf{x}}) = g(\mathbf{w} \cdot \mathbf{x})$

$h_{\mathbf{w}}(\mathbf{x})$ varies from 0 to 1. If it is >= 0.5 then **x** belongs to one class (e.g., y =1), otherwise it belongs to the other class (y =0)

For correct prediction we define a loss function. We try to minimize the loss function to get the correct prediction most of the time. The loss function depends on the weights (coefficients of the linear equation). Initially the weights are set randomly. After presenting each example, if the prediction is incorrect, the weights are updated in proportion to the derivative of

the loss function with respect to the weights. This is called gradient descent approach

Weight update rule is

wi = wi +α x derivative of loss function with respect to wi

$$\begin{aligned}\frac{\partial}{\partial w_i} Loss(\mathbf{w}) &= \frac{\partial}{\partial w_i}(y - h_{\mathbf{w}}(\mathbf{x}))^2 \\ &= 2(y - h_{\mathbf{w}}(\mathbf{x})) \times \frac{\partial}{\partial w_i}(y - h_{\mathbf{w}}(\mathbf{x})) \\ &= -2(y - h_{\mathbf{w}}(\mathbf{x})) \times g'(\mathbf{w} \cdot \mathbf{x}) \times \frac{\partial}{\partial w_i}\mathbf{w} \cdot \mathbf{x} \\ &= -2(y - h_{\mathbf{w}}(\mathbf{x})) \times g'(\mathbf{w} \cdot \mathbf{x}) \times x_i \,.\end{aligned}$$

The derivative g' of the logistic function satisfies $g'(z) = g(z)(1 - g(z))$, so we hav

$$g'(\mathbf{w} \cdot \mathbf{x}) = g(\mathbf{w} \cdot \mathbf{x})(1 - g(\mathbf{w} \cdot \mathbf{x})) = h_{\mathbf{w}}(\mathbf{x})(1 - h_{\mathbf{w}}(\mathbf{x}))$$

so the weight update for minimizing the loss is

$$w_i \leftarrow w_i + \alpha\,(y - h_{\mathbf{w}}(\mathbf{x})) \times h_{\mathbf{w}}(\mathbf{x})(1 - h_{\mathbf{w}}(\mathbf{x})) \times x_i \,.$$

Here α is called the learning rate and it must be a small quantity to help quick convergence. The weights are updated until the linear separators classifies all the

examples correctly or a desired accuracy threshold is achieved.

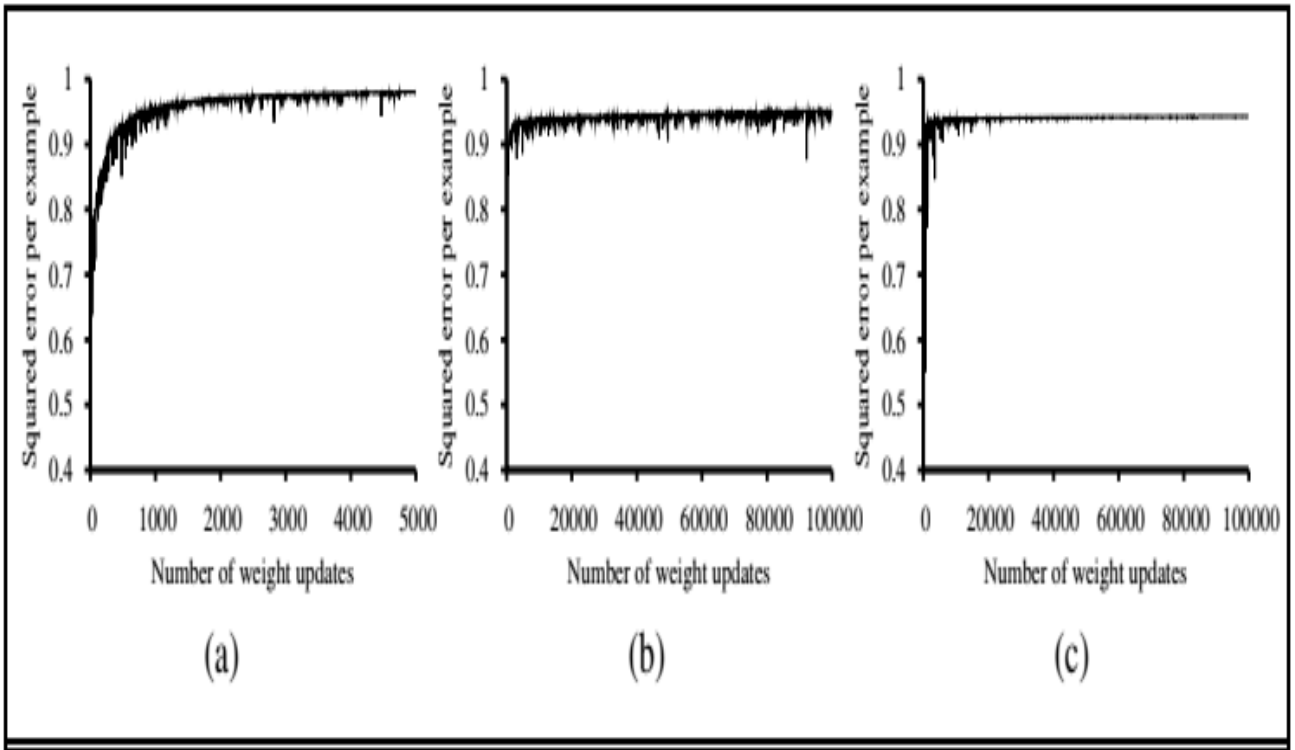

Figure 18.18 Repeat of the experiments in Figure 18.16 using logistic regression and squared error. The plot in (a) covers 5000 iterations rather than 1000, while (b) and (c) use the same scale.

Question:

The initial classifier for OR logic is randomly set to $0.5\ X_1 + 0.5\ X_2 - 1.25 = 0$. Assuming the learning rate is 0.1 and the learner is using sigmoid activation function. What will be the updated equation after all the training examples are processed once?

The initial classifier for OR logic is randomly set to $0.5\ X_1 + 0.5\ X_2 - 1.25 = 0$. Assuming the learning rate is 0.1 and the learner is using threshold activation function. What will be the updated equation after all the training examples are processed once?

The initial classifier for OR logic is randomly set to $X_1 + X_2 - 0.75 = 0$. Assuming the learning rate is 0.1 and the learner is using threshold activation function. What will be the updated equation after all the training examples are processed once?

Find the linear classifier for an OR gate using hard threshold function

The linear equation for a two input OR gate is **w.x** = w1x1+w2x2+w0 = 0. Let us assume w1=0.5, w2=0.75 and w0 = -1.25. So the initial hypothesis is

h**w**(**x**) = 0.5x1+0.75x2-1.25 = 0

1st Cycle			
x1=0, x2=0, y=0	**w.x=-1.25**	h**w**(**x**)=threshold(-1.25)=0	y ==hw(x) w1,w2,w0 remain same
x1=0, x2=1, y=1	**w.x=-0.50**	h**w**(**x**)=threshold(-0.50)=0	y /= hw(x) w1,w2,w0 need to updated
		w1 = 0.5+0.2(1-0)0 =0.5 w2=0.75+0.2(1-0)1=0.95 w0=-1.25+0.2(1-0)1=-1.05	
		hw(x)=0.5x1+0.95x2-1.05=0	
x1=1, x2=0, y=1	**w.x=-0.50**	h**w**(**x**)=threshold(-0.55)=0	y /= hw(x) w1,w2,w0 need to updated
		w1 = 0.5+0.2(1-0)1 =0.7 w2=0.95+0.2(1-0)0=0.95 w0=-1.05+0.2(1-0)1=-0.85	
		hw(x)=0.7x1+0.95x2-0.85=0	
x1=1, x2=1, y=1	**w.x=0.80**	h**w**(**x**)=threshold(0.85)=1	y == hw(x) w1,w2,w0 remain same
		w1 =0.7 w2=0.95 w0=-0.85	
		hw(x)=0.7x1+0.95x2-0.85=0	
2nd Cycle			
x1=0, x2=0, y=0	**w.x=-0.85**	h**w**(**x**)=threshold(-0.85)=0	y ==hw(x)

			w1,w2,w0 remain same
x1=0, x2=1, y=1	**w.x=0.10**	h**w**(**x**)=threshold(0.10)=1	y == hw(x) w1,w2,w0 remain same
x1=1, x2=0, y=1	**w.x=-0.15**	h**w**(**x**)=threshold(-0.15)=0	y /= hw(x) w1,w2,w0 need to updated
		w1 = 0.7+0.2(1-0)1 =0.9 w2=0.95+0.2(1-0)0=0.95 w0=-0.85+0.2(1-0)1=-0.65	
		hw(x)=0.9x1+0.95x2-0.65=0	
x1=1, x2=1, y=1	**w.x=1.2**	h**w**(**x**)=threshold(1.2)=1	y == hw(x) w1,w2,w0 remain same
		hw(x)=0.9x1+0.95x2-0.65=0	

Find the linear classifier for an OR gate using sigmoid threshold function

The linear equation for a two input OR gate is **w.x** = w1x1+w2x2+w0 = 0. Let us assume w1=0.5, w2=0.75 and w0 = -1.25. So the initial hypothesis is

h**w**(**x**) = 0.5x1+0.75x2-1.25 = 0

1st Cycle			
x1=0, x2=0, y=0	**w.x=-1.25**	g(w.x) = 0.22 h**w**(**x**)= 0	y ==hw(x) w1,w2,w0 remain same
x1=0, x2=1, y=1	**w.x=-0.50**	g(w.x) = 0.38 h**w**(**x**)=0	y /= hw(x) w1,w2,w0 need to updated

		w1 = .5+.2(1-0)(.38)(1-.38)(0)=0.5 w2=.75+.2(1-0)(.38)(1-.38)(1)=0.8 w0=-1.25+.2(1-0)(.38)(1-.38)(1)=-1.2	
		hw(x)=0.5x1+0.8x2-1.2=0	
x1=1, x2=0, y=1	**w.x=-0.70**	g(w.x) = 0.33 h**w**(**x**)=0	y /= hw(x) w1,w2,w0 need to beupdated
		w1 = .5+.2(1-0)(.33)(1-.33)(1)=0.54 w2=.8+.2(1-0)(.33)(1-.33)(0)=0.8 w0=-1.2+.2(1-0)(.33)(1-.33)(1)=-1.15	
		hw(x)=0.54x1+0.8x2-1.15=0	
x1=1, x2=1, y=1	**w.x=0.19**	g(w.x) = 0.55 h**w**(**x**)=1	y == hw(x) w1,w2,w0 remain same
		w1=0.54 w2 = 0.8 w0 = -1.15	
		hw(x)=0.54x1+0.8x2-1.15=0	
2nd Cycle			
x1=0, x2=0, y=0	**w.x=-1.15**	g(w.x) = 0.24 h**w**(**x**)=0	y ==hw(x) w1,w2,w0 remain same
x1=0, x2=1, y=1	**w.x=-0.35**	g(w.x) = 0.42 h**w**(**x**)=0	y /= hw(x) w1,w2,w0 need to be updated
		w1 = .54+.2(1-0)(.42)(1-..42)(0)=0.54 w2=.8+.2(1-0)(.42)(1-..42)(1)=0.85 w0=-1.15+.2(1-0)(.42)(1-.42)(1)=-1.1	
		hw(x)=0.54x1+0.85x2-1.1=0	
x1=1, x2=0, y=1	**w.x=-0.56**	g(w.x) = 0.36 h**w**(**x**)=0	y /= hw(x) w1,w2,w0 need to be updated
		w1 = .54+.2(1-0)(.36)(1-.36)(1)=0.59	

		w2=.85+.2(1-0)(.36)(1-.36)(0) = 0.85 w0=-1.1+.2(1-0)(.36)(1-.36)(1)=-1.05	
		hw(x)=0.59x1+0.85x2-1.05=0	
x1=1, x2=1, y=1	**w.x=0.39**	g(w.x) = 0.6 h**w**(**x**)=1	y == hw(x) w1,w2,w0 remain same
		hw(x)= 0.59x1+0.85x2-1.05 = 0	