The bias is outside the dot product

$g(x) = \theta_0 + \theta^T X$

Augment 1 to X inputs and θ_0 to weights vector to get

$\theta = [\theta_0 \ldots \theta_n]$

$X = [1, X_1, \ldots X_n]$

Step function: $g(x) = \theta_0 + \theta_1 X_1 + \cdots + \theta_n X_n = \theta^T X$

Logistic function: $h_\theta(x) = \frac{1}{1+exp(-\theta^T X)}$

When θ is 0, the logistic function becomes a straight line.

To model as a linear function:

$log(\frac{P(Y=1|X)}{P(Y=0|X)}) = \theta^T X$

Taking exponents on both sides

$\frac{P(Y=1|X)}{P(Y=0|X)} = exp(\theta^T X)$

Rearrange and we get

$P(Y = 1|X)(1 + exp(\theta^T X) = exp(\theta^T X)$

$P(Y = 1|X) = \frac{1}{1+exp(-\theta^T X)} = h_\theta X$

if $h_\theta X$ ¿ 0.5 then class 1

otherwise, class 2

$$\frac{d}{dx} sigmoid(X) = sigmoid(X)(1 - sigmoid(x))$$

$$\frac{d}{dx} \log\ sigmoid(X) = \frac{1}{sigmoid(X)} sigmoid(X)(1 - sigmoid(X))$$

$$\frac{d}{dx} \log\ sigmoid(X) = 1 - sigmoid(X)$$

$$\theta^{i+1} = \theta^i - \eta \nabla \mathrm{J} \theta^i$$

We compute the derivative (gradient) and take the negative of it as the direction to update parameters.

η is the learning rate. We can use it to control the size of updating the parameters.

Stop the iterative process when difference in loss compared to previous step is less than a threshold value

$(J(\theta^{i+1}) - J(\theta^{i})) < Threshold$
The condition should use the loss change as it the objective to minimize loss. A small change in parameters will not reflect a small loss change.

Small η means that loss change is slow and could end up hitting the threshold.

Large η could end up overshooting the minimum value and may never converge after many iterations.

log likelihood $= log(\prod_{X^i \in c1} P(Y = 1|X^i) \prod_{X^i \in c0} P(Y = 0|X^i))$
Combining both parts by raising to power of 1 when relevant
$log \prod_{i=1}^{m} P(Y = 1|X^i)^{y^i} P(Y = 0|X^i)^{1-y^i}$
Taking log of product to get sum
$\sum_{i=1}^{m} y^i log(P(Y = 1|X^i)) + (1 - y^i) log(1 - P(Y = 1|X^i))$
$l(\theta) = \sum_{i=1}^{m} y^i log(h_\theta(X^i)) + (1 - y^i) log(1 - h_\theta(X^i))$

Where, $y^i = 1$ when class 1
$(1 - y^i) = 1$ when class 0
$h_\theta(X^i)$ is prediction

$\frac{d}{d\theta} l(\theta) = \frac{d}{d\theta} \sum_{i=1}^{m} y^i log(h_\theta(X^i)) + (1 - y^i) log(1 - h_\theta(X^i))$
taking derivative and simplifying, we get
$\sum_{i=1}^{m} y^i (1 - h_\theta(X^i) X^i) + \sum_{i=1}^{m} (1 - y^i)(-h_\theta(X^i) X^i)$
$\sum_{i=1}^{m} (y^i - y^i h_\theta(X^i) - h_\theta(X^i) + y^i h_\theta(X^i)) X^i$
$\sum_{i=1}^{m} (y^i - h_\theta(X^i)) X^i$
$\frac{d}{d\theta}(-l(\theta)) = \sum_{i=1}^{m} (y^i - h_\theta(X^i)) X^i$

In the above equation,
$(y^i - h_\theta(X^i))$ is equal to 1 or -1 only when actuals disagree with predictions.
So, the negative log likelihood is simply the sum over all feature vectors.

One against all others:

$\hat{y} = argmax_j g_j(X)$

The predicted value depends on the maximum value of the distance between the input X and the discriminant boundary. As usual, the value on one side of boundary will be positive which classifies input X. If a value resides in the region where two or more discriminant functions show positive value, then the maximum of them would correspond to the class.

If there are k classes, then there will be k discriminant functions. One against each other:

$\hat{y} = argmax_i \sum_{j \neq i} g_{ij}(X)$

The predicted value depends on the maximum value of sum of distance between the input X and all other discriminant boundary. The maximum of this value will classify given input X.

If there are k classes, then there will be $\frac{k(k-1)}{2}$ discriminant.

When data is clustered close to one another, it is easier to use one against each other to discriminate the data. When data is far apart, one against all other does a fairly good job.

Softmax is same as sigmoid with one against all other approach. A sigmoid will classify between 2 classes. Softmax will take proportion of probability of one class over the probability of all other classes. This way, the sum of all probabilities will be equal to 1. Otherwise, the equation would not stand valid with probability ¿ 1.

$\hat{y}_j = P(y = j|X) = h_{\theta_j}(X) = \frac{exp(\theta_j^T)}{\sum_{i=1}^{k} exp(\theta_j^T)}$

Here, numerator is the probability of y belongs to j given X. Denominator is the probability of all other probabilities.

$\frac{d}{d\theta_i} softmax(\theta_j^T X) = softmax(\theta_j^T X)(\delta_{ij} - softmax(\theta_j^T X))X$

$\frac{d}{d\theta_i} \log softmax(\theta_j^T X) = \frac{1}{softmax(\theta_j^T X)} softmax(\theta_j^T X)(\delta_{ij} - softmax(\theta_j^T X))X$

$\frac{d}{d\theta_i} \log softmax(\theta_j^T X) = \delta_{ij} - softmax(\theta_j^T X)X$

Where $\delta_{ij} = 1$ when j and 0 otherwise