# HW1 Individual

**Jimmy Ye**
CSE 190: Neural Networks
University of California, San Diego
jiy162@ucsd.edu

## 1 Perceptrons

### 1.1

Assuming d=2, derive the equation for the line that is the decision boundary.

$$y(x) = 0 \tag{1}$$
$$w_1 x_1 + w_2 x_2 + w_0 = 0 \tag{2}$$
$$w_2 x_2 = -w_1 x_1 - w_0 \tag{3}$$
$$x_2 = -(\frac{w_1}{w_2})x_1 - \frac{w_0}{w_2} \tag{4}$$

### 1.2

Prove that the distance from the decision boundary to the origin is given by:

$$l = \frac{w^\top x}{\|w\|}$$

Note that this is for $x$ on the decision boundary, so from $w^\top x + w_0 = 0$, we get $l = \frac{w^\top x}{\|w\|} = \frac{-w_0}{\|w\|}$.

We know that $w$ is orthogonal to the decision boundary. Then the desired distance, by definition of distance from a point to a line, is given by the scalar $c$ s.t. $c\hat{w}$ is on the line (where $\hat{w} = \frac{w}{\|w\|}$, the unit vector parallel to $w$).

So, let $x = c\hat{w}$ in equation (4). Then

$$c\frac{w_2}{\|w\|} = -\frac{cw_1 w_1}{w_2\|w\|} - \frac{w_0}{w_2} \tag{5}$$
$$c\frac{w_2}{\|w\|} + c\frac{w_1 w_1}{w_2\|w\|} = -\frac{w_0}{w_2} \tag{6}$$
$$c(w_2 + \frac{w_1 w_1}{w_2}) = -\|w\|\frac{w_0}{w_2} \tag{7}$$
$$c(w_2 w_2 + w_1 w_1) = -\|w\| w_0 \tag{8}$$
$$c = -\frac{w_0}{\|w\|} \tag{9}$$

### 1.3

Write down the perceptron learning rule as an update equation.

$w_i = w_i + \alpha(t - y)x_i$, where $\alpha$ is the learning rate, $t$ is the target output, and $y$ is the current output.

### 1.4

Initialize $w_1, w_2$ and $\theta$ to be 0 and fix the learning rate to 1, and train the perceptron to learn NAND, adding one row for each "randomly" selected pattern. Stop when the learning converges.

| $x_1$ | $x_2$ | Net | Output | Teacher | $w_1$ | $w_2$ | Threshold $\theta(=-w_0)$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | -1 | -1 | -1 |

### 1.5

Is the solution unique? Why or why not?

No, other possible weights are valid solutions, like $w_0 = 2, w_1 = -2, w_2 = -2$, which could be reached with the same training example if we instead used a training rate of 2.

## 2 Logistic Regression

Show that the gradient of the Cross Entropy cost function for the logistic activation function is:

$$-\frac{\partial E(w)}{\partial w_j} = \sum_{n=1}^{N} (t^n - y^n) x_j^n$$

First, let's calculate $\frac{\partial y^n}{\partial w_j}$:

$$\begin{aligned}
\frac{\partial y^n}{\partial w_j} &= \frac{\frac{\partial}{\partial w_j}(1 + e^{-w^\top x})}{(1 + e^{-w^\top x})^2} && \text{Quotient rule} && (10)\\
&= \frac{-x_j^n e^{-w^\top x}}{(1 + e^{-w^\top x})^2} && && (11)\\
&= x_j^n \frac{1}{1 + e^{-w^\top x}} \frac{-e^{-w^\top x}}{1 + e^{-w^\top x}} && && (12)\\
&= x_j^n y^n (1 - y^n) && && (13)
\end{aligned}$$

And calculating the desired gradient:

$$\begin{aligned}
-\frac{\partial E(w)}{\partial w_j} &= \frac{\partial}{\partial w_j} \Big[ \sum_{n=1}^{N} (t^n \ln(y^n) + (1 - t^n) \ln(1 - y^n)) \Big] && \text{Substituting } E(w) && (14)\\
&= \sum_{n=1}^{N} \Big[ \frac{\partial}{\partial w_j} (t^n \ln(y^n) + (1 - t^n) \ln(1 - y^n)) \Big] && \text{Linearity of derivative} && (15)\\
&= \sum_{n=1}^{N} \Big[ t^n \frac{\partial}{\partial w_j} \ln(y^n) + (1 - t^n) \frac{\partial}{\partial w_j} \ln(1 - y^n) \Big] && \text{Linearity of derivative} && (16)\\
&= \sum_{n=1}^{N} \Big[ \frac{t^n}{y^n} \frac{\partial y^n}{\partial w_j} + \frac{1 - t^n}{1 - y^n} \frac{\partial (1 - y^n)}{\partial w_j} \Big] && \text{Derivative of ln} && (17)\\
&= \sum_{n=1}^{N} \Big[ x_j^n t^n (1 - y^n) - x_j^n y^n (1 - t^n) \Big] && \text{Using (13)} && (18)\\
&= \sum_{n=1}^{N} \Big[ (t^n - y^n) x_j^n \Big] && && (19)
\end{aligned}$$

## 3 Softmax Regression

Show that the gradient of the Cross Entropy cost function for softmax regression is:

$$-\frac{\partial E(w)}{\partial w_{jk}} = \sum_{n=1}^{N}(t_k^n - y_k^n)x_j^n$$

Let $a_k^n = exp(w_k^\top x^n)$.

First, let's calculate $\frac{\partial}{\partial w_{jk}} \ln(y_i^n)$:

$$\begin{aligned}
\frac{\partial}{\partial w_{jk}} \ln(y_i^n) &= \frac{\partial}{\partial w_{jk}} \ln\left(\frac{a_i^n}{\sum_{h=1}^{c} a_h^n}\right) && \text{Substituting } y_k^n && (20)\\
&= \frac{\partial}{\partial w_{jk}} \left(\ln(a_i^n) - \ln\left(\sum_{h=1}^{c} a_h^n\right)\right) && \text{Logarithm property} && (21)\\
&= \frac{\partial}{\partial w_{jk}} \ln(a_i^n) - \frac{\partial}{\partial w_{jk}} \ln\left(\sum_{h=1}^{c} a_h^n\right) && \text{Linearity of derivative} && (22)
\end{aligned}$$

Note that $\frac{\partial a_i^n}{\partial w_{jk}}$ is $x_j^n a_k^n$ when $i = k$, and is 0 otherwise. Thus, when $i = k$, the first term is $x_n^n a_k^n / a_k^n = x_j^n$, otherwise it is 0, and the second term is always $x_j^n y_k^n$.

Calculating the desired gradient:

$$\begin{aligned}
-\frac{\partial E(w)}{\partial w_{jk}} &= \frac{\partial}{\partial w_{jk}} \Big[\sum_{n=1}^{N}\sum_{i=1}^{c} t_i^n \ln(y_i^n)\Big] && \text{Substituting } E(w) && (23)\\
&= \sum_{n=1}^{N}\sum_{i=1}^{c}\left[t_i^n \frac{\partial}{\partial w_{jk}} \ln(y_i^n)\right] && \text{Linearity of derivative} && (24)\\
&= \sum_{n=1}^{N}\left[t_k^n x_j^n - \sum_{i=1}^{c} t_i^n x_j^n y_k^n\right] && \text{Using (22)} && (25)\\
&= \sum_{n=1}^{N}\left[t_k^n x_j^n - x_j^n y_k^n\right] && \text{One-hot encoding: there is a unique } t_i = 1 && (26)\\
&= \sum_{n=1}^{N}\left[(t_k^n - y_k^n)x_j^n\right] && && (27)
\end{aligned}$$