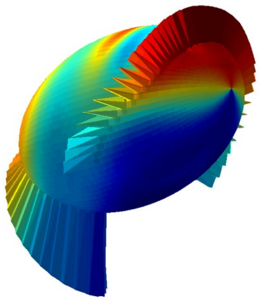
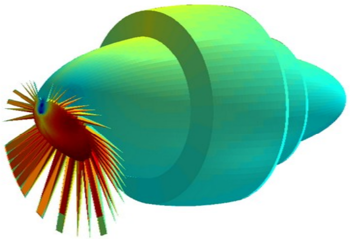
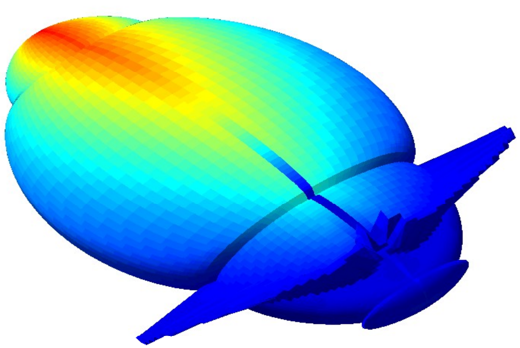
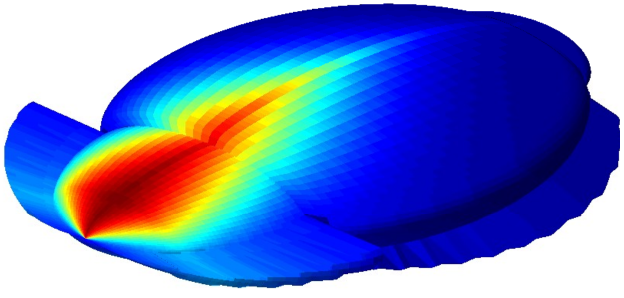




Lecture 8
Neural Classifiers





Using Backpropagation for Classification Problems

Backpropagation requires that $\frac{\partial J^q}{\partial w_{jk}}$ exists $\forall w_{ij}$

$\Rightarrow f_i'(s_i^q)$ must exist

$\Rightarrow f(u)$ (activation function) must be differentiable

$$\rightarrow \text{ use } f(u) = \frac{1}{1+e^{-u}}$$

$$\text{or } f(u) = \tanh(u)$$

instead of hard threshold.

But classification requires that output neurons have 0/1 or +1/-1 output

??????????

Possible Solutions

- Use 0/1 or +1/-1 as target values y_i^q

- Set operating parameters *H*, *L*, *L*<*H* such that:

$\hat{y}_i \geq H$ is considered a match for $y_i = +1$

$\hat{y}_i \leq L$ is considered a match for $y_i = 0$ or -1

- During training, use 0/1 or +1/-1 as targets but if $\hat{y}_i$ matches the operating targets, make no weight change

Typical values for *H*, *L* are:

H=0.75 L=0.25 for 0-1 sigmoid
H=0.75 L=-0.75 for +/- sigmoid

Multi-Class Classification

If there are more than 2 classes, how should the output be represented?

Solution:

- If there are M classes, have M neurons in the output layer.
- During training, use one-hot codes as targets for each class, e.g. if $M = 3$,
  Class 1 = [1 0 0]   Class 2 = [0 1 0]   Class 3 = [0 0 1]
- After training, for a test input x, choose the class as the output neuron with the highest output, e.g., Output = [0.1 0.6 0.2] $\rightarrow$ [0 1 0]
- If "no choice" is allowed, require that an output must be higher than some threshold θ to be considered 1.

A better solution: softmax classifiers

Logistic Regression

Consider a case where a binary outcome (yes/no, true/false, 0/1) depends on a continuous variable (feature) *x*

For example:
Is patient A sick or well given their temperature?
Will borrower B pay back a loan given their income?
Will person C live until next year given their age?

The general form is:
Is *F* true or false given the value of *x*? $F : x \to \{0,1\}$

Often, the dependence of *F* on *x* is:

- Monotonic
- Uncertain

In this situation, we can ask the question in terms of probabilities:
What is the probability that *F* = 1 given *x*?

For a more detailed intro, see http://ufldl.stanford.edu/tutorial/supervised/LogisticRegression/

Let:

$$y \equiv P(F = 1 \mid x) = p(x; \mathbf{w})$$

Where

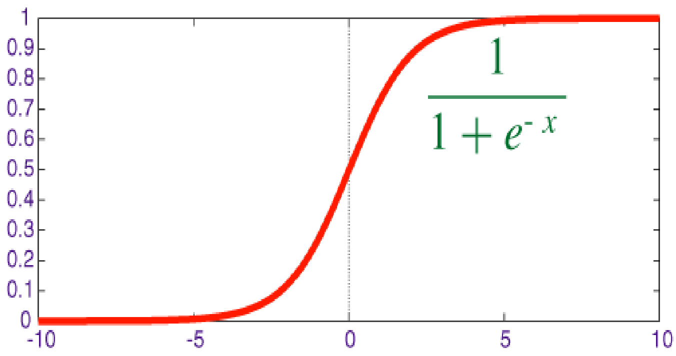
- $p\,(x;\, \mathbf{w}) \equiv$ probability estimate model
- $\mathbf{w} \equiv$ parameters of model p

We need to tune $\mathbf{w}$ based on data to obtain the optimal p.

Logistic regression fits a linear model by postulating that:

$$\underbrace{\log \frac{p(x;\mathbf{w})}{1 - p(x;\mathbf{w})}}_{\text{log odds ratio}} = w_1 x + w_0 \qquad \mathbf{w} = \{w_0, w_1\}$$

$$p(x;\mathbf{w}) = \frac{e^{w_1 x + w_0}}{1 + e^{w_1 x + w_0}} = \frac{1}{1 + e^{-(w_1 x + w_0)}}$$



https://www.quora.com/What-is-the-sigmoid-function-and-what-is-its-use-in-machine-learnings-neural-networks

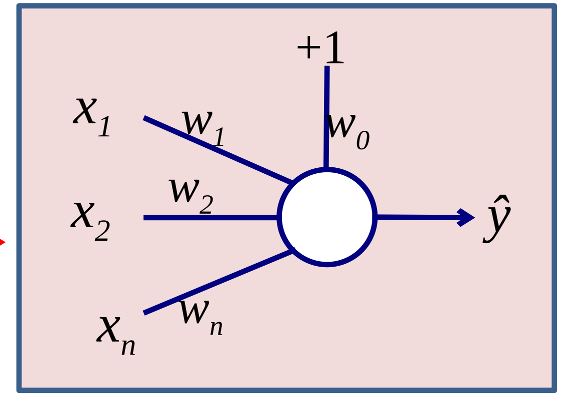
If there is more than one feature, $\{x_1, \ldots, x_n\}$, we get:

$$\log \frac{p(\boldsymbol{x};\boldsymbol{w})}{1-p(\boldsymbol{x};\boldsymbol{w})} = \boldsymbol{w}\cdot\boldsymbol{x}$$

$$\boldsymbol{x} = \{1, x_1, \ldots, x_n\}$$
$$\boldsymbol{w} = \{w_0, w_1, \ldots, w_n\}$$

$$p(\boldsymbol{x};\boldsymbol{w}) = \frac{e^{\boldsymbol{w}\cdot\boldsymbol{x}}}{1+e^{\boldsymbol{w}\cdot\boldsymbol{x}}} = \frac{1}{1+e^{-\boldsymbol{w}\cdot\boldsymbol{x}}} = \sigma(\boldsymbol{w}\cdot\boldsymbol{x}) \equiv \hat{y}$$

Probability Model



which is the activation function of a sigmoid neuron.

Thus, the output of a sigmoid output neuron in a 2-class classification problem can be seen as a logistic estimate of the probability of Class 1 given the current input: $y = P(F = 1 \mid \boldsymbol{x})$

Think of this as the probabilistic version of linear separability.

If $P(F = 1 \mid \boldsymbol{x})$ is not a monotonic (logistic) function of x, the purpose of the hidden layer is to map $\boldsymbol{x}$ to a new representation, $\boldsymbol{h}$, such that $P(F = 1 \mid \boldsymbol{h})$ is a logistic function of $\boldsymbol{h}$.

In fact, logistic regression can be seen as minimizing the ***cross-entropy loss function***:

$$J(w) = -\left[\sum_{q=1}^{M} \left(y^q \log\left(p\left(\mathbf{x}^q; \mathbf{w} \right) \right) + \left(1 - y^q\right) \log\left(1 - p\left(\mathbf{x}^q; \mathbf{w} \right)\right) \right) \right]$$

$M \equiv$ number of data points

Which gives the gradient vector:

$$\nabla J(\mathbf{w}) = \sum_{q=1}^{M} \left(p\left(\mathbf{x}^q; \mathbf{w} \right) - y^q \right) \mathbf{x}^q = \sum_{q=1}^{M} \left(\hat{y}^q - y^q \right) \mathbf{x}^q$$

Using steepest descent results in the learning rule:

$$\Delta \mathbf{w} = -\eta \nabla J(\mathbf{w}) = \eta \sum_{q=1}^{M} \left(y^q - p\left(\mathbf{x}^q; \mathbf{w} \right) \right) \mathbf{x}^q = \eta \sum_{q=1}^{M} \left(y^q - \hat{y}^q \right) \mathbf{x}^q$$

which is the perceptron learning rule, but with a sigmoid neuron.

In fact, logistic regression can be seen as minimizing the ***cross-entropy loss function***:

$$J(w) = -\left[\sum_{q=1}^{M}\left(\underbrace{\boxed{y^q}}_{=1 \text{ if class of } x^q = 1}\log\left(p\left(\mathbf{x}^q;\mathbf{w}\right)\right) + \underbrace{\boxed{\left(1 - y^q\right)}}_{=1 \text{ if class of } x^q = 0}\log\left(1 - p\left(\mathbf{x}^q;\mathbf{w}\right)\right)\right)\right]$$

Which gives the gradient vector:

$$\nabla J(\mathbf{w}) = \sum_{q=1}^{M}\left(p\left(\mathbf{x}^q;\mathbf{w}\right) - y^q\right)\mathbf{x}^q = \sum_{q=1}^{M}\left(\hat{y}^q - y^q\right)\mathbf{x}^q$$

Using steepest descent results in the learning rule:

$$\Delta\mathbf{w} = -\eta\nabla J(\mathbf{w}) = \eta\sum_{q=1}^{M}\left(y^q - p\left(\mathbf{x}^q;\mathbf{w}\right)\right)\mathbf{x}^q = \eta\sum_{q=1}^{M}\overbrace{\boxed{\left(y^q - \hat{y}^q\right)}}^{\text{Scalar}}\mathbf{x}^q$$

Batch mode

which is the perceptron learning rule, but with a sigmoid neuron.

To derive $\nabla J(\mathbf{w}) = \sum_{q=1}^{M} \left(p\left(\mathbf{x}^q;\mathbf{w}\right) - y^q \right) \mathbf{x}^q$ consider

$$J^q = -y^q \log\left(p\left(\mathbf{x}^q;\mathbf{w}\right)\right) - \left(1-y^q\right)\log\left(1-p\left(\mathbf{x}^q;\mathbf{w}\right)\right)$$

$$= -y^q \log\left(\sigma\left(\mathbf{w}\cdot\mathbf{x}^q\right)\right) - \left(1-y^q\right)\log\left(1-\sigma\left(\mathbf{w}\cdot\mathbf{x}^q\right)\right)$$

$$= -y^q \log\left(\sigma\left(\mathbf{w}\cdot\mathbf{x}^q\right)\right) - \left(1-y^q\right)\log\left(\sigma\left(-\mathbf{w}\cdot\mathbf{x}^q\right)\right)$$

$$= \begin{cases} \log\left(1+\exp(-\mathbf{w}\cdot\mathbf{x}^q)\right) & \text{if } y^q = 1 \\ \log\left(1+\exp(\mathbf{w}\cdot\mathbf{x}^q)\right) & \text{if } y^q = 0 \end{cases}$$

since

$$\log\left(\sigma\left(\mathbf{w}\cdot\mathbf{x}^q\right)\right) = \log\left(\frac{1}{1+e^{-\mathbf{w}\cdot\mathbf{x}^q}}\right) = \log(1) - \log\left(1+e^{-\mathbf{w}\cdot\mathbf{x}^q}\right) = -\log\left(1+e^{-\mathbf{w}\cdot\mathbf{x}^q}\right)$$

and

$$\log\left(\sigma\left(-\mathbf{w}\cdot\mathbf{x}^q\right)\right) = \log\left(\frac{1}{1+e^{\mathbf{w}\cdot\mathbf{x}^q}}\right) = \log(1) - \log\left(1+e^{\mathbf{w}\cdot\mathbf{x}^q}\right) = -\log\left(1+e^{\mathbf{w}\cdot\mathbf{x}^q}\right)$$

Appendix

$$J^q = \begin{cases} \log\left(1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x}^q)\right) & \text{if } y^q = 1 \\ \log\left(1 + \exp(\boldsymbol{w} \cdot \boldsymbol{x}^q)\right) & \text{if } y^q = 0 \end{cases}$$

$$\Rightarrow \frac{\partial J^q}{\partial w_j} = \begin{cases} \dfrac{\partial}{\partial w_j} \log\left(1 + \exp(-\boldsymbol{w} \cdot \boldsymbol{x}^q)\right) & \text{if } y^q = 1 \\ \dfrac{\partial}{\partial w_j} \log\left(1 + \exp(\boldsymbol{w} \cdot \boldsymbol{x}^q)\right) & \text{if } y^q = 0 \end{cases}$$

Appendix

For $y = 1$

$$\frac{\partial}{\partial w_j}\log\left(1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right) = \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\frac{\partial}{\partial w_j}\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right)$$

$$= \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\frac{\partial}{\partial w_j}(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right)$$

$$= \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)(-\boldsymbol{x}_j^q)\right) = -\left(\frac{\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\boldsymbol{x}_j^q$$

$$= -\left(1-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q = -\left(y-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q$$

Similarly, for $y = 0$

$$\frac{\partial}{\partial w_j}\log\left(1+\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)\right) = \left(\frac{\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)}{1+\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\boldsymbol{x}_j^q = \left(0+\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q$$

$$= -\left(0-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q = -\left(y-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q$$

Appendix

For $y = 1$

$$\frac{\partial}{\partial w_j}\log\left(1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right) = \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\frac{\partial}{\partial w_j}\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right)$$

$$= \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\frac{\partial}{\partial w_j}(-\boldsymbol{w}\cdot\boldsymbol{x}^q)\right)$$

$$= \left(\frac{1}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\left(\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)(-x_j^q)\right) = -\left(\frac{\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}{1+\exp(-\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\boldsymbol{x}_j^q$$

$$= -\left(1-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q = \boxed{-\left(y-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q}$$

Similarly, for $y = 0$

$$\frac{\partial}{\partial w_j}\log\left(1+\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)\right) = \left(\frac{\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)}{1+\exp(\boldsymbol{w}\cdot\boldsymbol{x}^q)}\right)\boldsymbol{x}_j^q = \left(0+\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q$$

$$= -\left(0-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q = \boxed{-\left(y-\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right)\right)\boldsymbol{x}_j^q}$$

same

$$\nabla J^q(\boldsymbol{w}) = \left[\frac{\partial J^q}{\partial w_1}\;\frac{\partial J^q}{\partial w_2}\cdots\frac{\partial J^q}{\partial w_n}\right]^T = \left(\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right) - y\right)\boldsymbol{x}^q$$

Note $\left(\sigma\left(\boldsymbol{w}\cdot\boldsymbol{x}^q\right) - y\right)$ is scalar

Appendix

$$\nabla J^q = \left[\frac{\partial J^q}{\partial w_1} \frac{\partial J^q}{\partial w_2} \cdots \frac{\partial J^q}{\partial w_n} \right]^T = \left(\sigma\left(\boldsymbol{w} \cdot \boldsymbol{x}^q \right) - y \right) \boldsymbol{x}^q$$

$$\nabla J(\boldsymbol{w}) = \sum_{q=1}^{M} \nabla J^q(\boldsymbol{w}) = \sum_{q=1}^{M} \left(\hat{y}^q - y^q \right) \boldsymbol{x}^q$$

Since $p\left(\boldsymbol{x}; \boldsymbol{w} \right) = \sigma\left(\boldsymbol{w} \cdot \boldsymbol{x} \right) \equiv \hat{y}$

Appendix

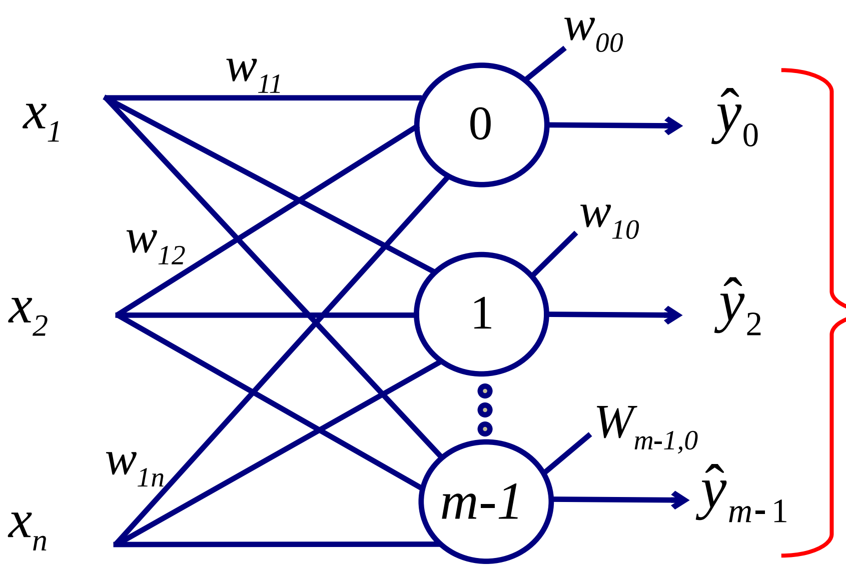
Softmax Classifier

The softmax classifier is the multi-class version of the logistic classifier.

We have $F : x \rightarrow \{0, 1, ..., m-1\}$

Since *F* can now take *m* different values (instead of two), we want to estimate:
$P(F = 0 \mid \boldsymbol{x}), P(F = 1 \mid \boldsymbol{x}), \ldots., P(F = m\text{-}1 \mid \boldsymbol{x})$

This can be done using an output layer of *m* neurons with $\hat{y}_i = P(F = i \mid \boldsymbol{x})$



These now represent a probability distribution and add up to 1

Hypothesize that:

$$P(F=i \mid \boldsymbol{x};\boldsymbol{w}) = \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{x}}}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r \cdot \boldsymbol{x}}}$$ ← Softmax function

where

- $\boldsymbol{w}_i$ is the weight vector of the output neuron for class i
- $\boldsymbol{w}$ is the weight matrix of the output layer

Then

$$p(\boldsymbol{x};\boldsymbol{w}) = \begin{bmatrix} P(F=0 \mid x;w) \\ P(F=1 \mid x;w) \\ \cdots \\ P(F=m-1 \mid x;w) \end{bmatrix} = \frac{1}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r \cdot \boldsymbol{x}}} \begin{bmatrix} e^{\boldsymbol{w}_0 \cdot \boldsymbol{x}} \\ e^{\boldsymbol{w}_1 \cdot \boldsymbol{x}} \\ \cdots \\ e^{\boldsymbol{w}_{m-1} \cdot \boldsymbol{x}} \end{bmatrix}$$

The weights can be learned by optimizing the cross-entropy loss function:

$$J(\boldsymbol{w}) = -\left[\sum_{q=1}^{M} \sum_{i=0}^{m-1} 1\{ y^q = i \} \log \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{x}^q}}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r \cdot \boldsymbol{x}^q}} \right]$$

x^q = qth input vector
y^q = class label of x^q

$1\{u\}$ = 1 if u is true, else 0

The gradient vector for the weight vector $\boldsymbol{w}_i$ is:

$$\nabla J_i^q(\boldsymbol{w}) = -\sum_{q=1}^{M} \left[\left(1\{ y^q = i \} - \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{x}^q}}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r \cdot \boldsymbol{x}^q}} \right) \boldsymbol{x}^q \right]$$

The jth component of this can be used to adjust weight w_{ij}

For more details, see: http://ufldl.stanford.edu/tutorial/supervised/SoftmaxRegression/

The weights can be learned by optimizing the cross-entropy loss function:

$$J(\boldsymbol{w}) = -\left[\sum_{q=1}^{M}\sum_{i=0}^{m-1} 1\{y^q = i\}\log\frac{e^{\boldsymbol{w}_i\cdot\boldsymbol{x}^q}}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r\cdot\boldsymbol{x}^q}}\right]$$

x^q = qth input vector
y^q = class label of x^q

$1\{u\}$ = 1 if u is true, else 0

The gradient vector for the weight vector $\boldsymbol{w}_i$ is:

$$\nabla J_i^q(\boldsymbol{w}) = -\sum_{q=1}^{M}\left[\boxed{\left(1\{y^q = i\} - \frac{e^{\boldsymbol{w}_i\cdot\boldsymbol{x}^q}}{\sum_{r=0}^{m-1} e^{\boldsymbol{w}_r\cdot\boldsymbol{x}^q}}\right)}\boldsymbol{x}^q\right]$$

scalar

The jth component of this can be used to adjust weight w_{ij}

For more details, see: http://ufldl.stanford.edu/tutorial/supervised/SoftmaxRegression/