Problem 1: Derive the weight update rule that maximize the conditional likelihood assuming that a daset $D = \{(x_i, y_i)\}_{i=1}^{n}$ is given.

the objective function is to find the optimal w^*

$$\Rightarrow w^* = \arg\max_{w} \prod_{i=1}^{N} \sigma(w^T x_i)^{y_i} (1-\sigma(w^T x_i))^{1-y_i}$$

but due to computational reason, we want to minimize the objective function rather than maximizing it.

$$\Rightarrow E(w) = \sum_{i=1}^{N} -y_i \ln(\sigma(w^T x_i)) - (1-y_i)\ln(1-\sigma(w^T x_i))$$

$$= \sum_{i=1}^{N} -y_i \ln\left(\frac{1}{1+e^{-w^T x_i}}\right) - \ln(1-\sigma(w^T x_i)) + y_i \ln(1-\sigma(w^T x_i))$$

$$= \sum_{i=1}^{N} -y_i \ln\left(\frac{1}{1+e^{-w^T x_i}}\right) - \ln\left(1-\frac{1}{1+e^{-w^T x}}\right) + y_i \ln\left(1-\frac{1}{1+e^{-w^T x}}\right)$$

$$= \sum_{i=1}^{N} -y_i \ln(1+e^{-w^T x})^{-1} - \ln\left(\frac{1+e^{-w^T x}-1}{1+e^{-w^T x}}\right) + y_i \ln\left(\frac{1+e^{-w^T x}-1}{1+e^{-w^T x}}\right)$$

$$= \sum_{i=1}^{N} y_i \ln(1+e^{-w^T x}) - \ln\left(\frac{e^{-w^T x}}{1+e^{-w^T x}}\right) + y_i \ln\left(\frac{e^{-w^T x}}{1+e^{-w^T x}}\right)$$

$$= \sum_{i=1}^{N} \cancel{y_i \ln(1+e^{-w^T x})} - \ln e^{-w^T x} + \ln(1+e^{-w^T x}) + y_i \ln(e^{-w^T x}) - \cancel{y_i \ln(1+e^{-w^T x})}$$

$$E(w) = \sum_{i=1}^{N} -\ln e^{-w^T x} + \ln(1+e^{-w^T x}) + y_i \ln(e^{-w^T x}) \qquad (a^*)$$

$$\Rightarrow \frac{\delta E(w)}{\delta w} = \left[\sum_{i=1}^{N} -\ln e^{-w^T x} + \ln(1+e^{-w^T x}) + y_i \ln(e^{-w^T x})\right]' \qquad (a)$$

$$\frac{\delta \ln e^{-w^T x}}{\delta w} = \left(\frac{1}{e^{-w^T x}}\right)(e^{-w^T x})' \quad \text{chain rule} \quad \text{" ' is the derivitive sign"}$$

$$= \left(\frac{1}{\cancel{e^{-w^T x}}}\right)(-w^T x)' \cancel{e^{-w^T x}} \quad \text{chain rule}$$

$$\frac{\delta \ln e^{-w^T x}}{\delta w} = -x \qquad (1)$$

$$\frac{\delta(\ln(1+e^{-w^T x}))}{\delta w} = \left(\frac{1}{1+e^{-w^T x}}\right)(1+e^{-w^T x})' \quad \text{chain rule}$$

$$= \left(\frac{1}{1+e^{-w^T x}}\right)(-w^T x)' e^{-w^T x}$$

$$= \frac{-x e^{-w^T x}}{1+e^{-w^T x}} \qquad (2)$$

put (1) ^ (2) into (a)

$$\Rightarrow \frac{\delta E(w)}{\delta w} = \sum_{i=1}^{N} (-)(-x) + \left(-\frac{xe^{-w^T x}}{1+e^{-w^T x}}\right) + (y_i)(-x)$$

$$= \sum_{i=1}^{N} x - x\frac{e^{-w^T x}}{1+e^{-w^T x}} - xy_i$$

$$= \sum_{i=1}^{N} x\left(1 - \frac{e^{-w^T x}}{1+e^{-w^T x}} - y_i\right)$$

but $\frac{e^{-w^T x}}{1+e^{-w^T x}} = 1 - \frac{1}{1+e^{-w^T x}}$.

$$\Rightarrow \frac{\delta(E(w))}{\delta w} = \sum_{i=1}^{N} x\left[\cancel{1} - \left(\cancel{1} - \frac{1}{1+e^{-w^T x}}\right) - y_i\right]$$

$$= \sum_{i=1}^{N} x\left(\frac{1}{1+e^{-w^T x}} - y_i\right)$$

$$\frac{\delta E(w)}{\delta w} = \sum_{i=1}^{N} x\left(\sigma(w^T x) - y_i\right) \qquad (c)$$

from a^*, w is a concave function of w; hence, there is no local minima.

On top of that concave function is easy to optimize using gradiant descent.

$\Rightarrow$ update rule w = w - (learning rate)(partial dervitrue of E(w) w.r.t w)

$$\Rightarrow \quad w_J = w_J - \alpha \nabla_w E(w)$$

from above dervitive, we found that $\nabla_w E(w) = \sum_{i=1}^{N} x\left(\sigma(w^T x) - y_i\right)$

$$\therefore \quad w_J = w_J - \alpha \sum_{i=1}^{N} x\left(\sigma(w^T x) - y_i\right)$$

Problem 2:

1. Compute $\frac{\delta \sigma(a)}{\delta w}$ when $a = w^T x$ where $x, w \in \mathbb{R}^m$

$$\sigma(a) = \frac{1}{1+e^{-a}} = (1+e^{-a})^{-1} = (1+e^{-w^T x})^{-1}$$

$$\Rightarrow \frac{\delta(1+e^{-w^T x})^{-1}}{\delta w} = \frac{(-1)(1+e^{-w^T x})'(1+e^{-w^T x})^{-2}}{\delta w}$$ chain rule

$$= (-1)(-w^T x)'(e^{-w^T x})(1+e^{-w^T x})^{-2}$$

$$= x e^{-w^T x}(1+e^{-w^T x})^{-2}$$

$$\Rightarrow \frac{\delta(1+e^{-w^T x})^{-1}}{\delta w} = \frac{x e^{-w^T x}}{(1+e^{-w^T x})^2}$$

2. For logistic regression, the probability of y, given input value x and its estimated weight vector w: can be presented as a posterior probability of y, $P(y|x,w)$ using bayes' theorem. if $y=1 \Rightarrow P(y=1|x,w) = (1+e^{-w^T x})^{-1}$ (a)

conversely, if $y=0 \Rightarrow P(y=0|x,w) = 1 - \frac{1}{1+e^{-w^T x}}$

$$= \frac{1+e^{-w^T x}-1}{1+e^{-w^T x}}$$

$$P(y=0|x) = \frac{e^{-w^T x}}{1+e^{-w^T x}} \qquad (b)$$

$P(y=1|x,w)$ means the probability of y or output label $=1$, given the input feature x and its estimated model parameter w.

$P(y=0|x,w)$ means the probability of y or output label $=0$, given the input feature x and its estimated model parameter w.

Using the Bernolli distribution, $p^k(1-p)^{1-k}$ $k \in \{0,1\}$, we could combine (a) ∧ (b) together, then we get

$$p(y|x,w) = \delta(w^T x)^y (1-\delta(w^T x))^{1-y}$$

if we generate N samples, $p(y|x,w) = \prod_{i=1}^{N} \delta(w^T x_i)^{y_i}(1-\delta(w^T x))^{1-y}$

3. Show the loss function for logistic regression and explain how we learn w

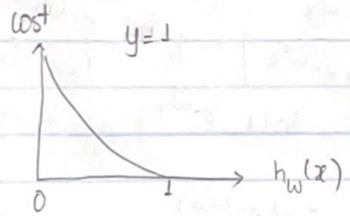
In logistic regression, we focus on doing the binary classifion; hence, $y \in \{0, 1\}$. We constraint the prediction to some value of 0, 1. Therefore, we use the sigmoid function $g(z) = \frac{1}{1+e^{(-z)}}$; hypothesis: $h_w(x) = \frac{1}{1+e^{-w^T x}}$

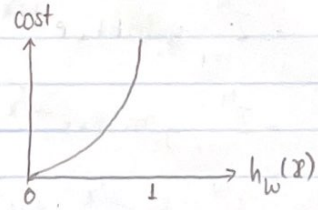
The hypothesis function returns the probability that $y=1$ given x, parameterized by w. It is written as $h(x) = P(y=1 \mid x, w)$. The decision boundary is:

$$\begin{cases} \text{if } w^T x \geq 0 \rightarrow h(x) \geq 0.5 \\ \text{elif } w^T x < 0 \rightarrow h(x) < 0.5 \end{cases}$$

The loss function is in place because we want to create a punishment when predicting 1 while it is actually 0 or 0 when it is actually 1.

$$cost(h_w(x), y) = \begin{cases} -\log(h_w(x)) & \text{if } y=1 \quad (1) \\ -\log(1-h_w(x)) & \text{if } y=0 \quad (2) \end{cases}$$





we could also add the cost function together. Now it will be written as

$$cost(h_w(x), y) = -y\log(h_w(x)) - (1-y)\log(1-h_w(x))$$

Hence, the cost function of the model is the summation of all the training data:

$$E(w) = \sum_{i=1}^{N} -y_i \log(h_w(x_i)) - (1-y_i)\log(1-h_w(x_i))$$

$$E(w) = \sum_{i=1}^{N} -y_i \log\left(\frac{1}{e^{-w^T x}}\right) - (1-y_i)\log\left(1 - \frac{1}{e^{-w^T x}}\right)$$

Problem 2.3 continue~

There is no closed-form equation to compute the value of w, that minimize the cost function. However, the cost function $E(w)$ is convex. Hence, gradient Descent is guaranteed to find global minimum.

The gradient descent update rule $w_{new} = w_{old} - \alpha \nabla_w E(w^{old}_{old})$

meaning, to compute w, we use the previous w value, substracts with α (the learning rate) time $\nabla_w E(w^{old})$ partial derivitive of $E(w)$ w.r.t w.

derive from problem (1) $w_{new} = w_j = w_j - \alpha \sum_{i=1}^{N} x_i(\sigma(w^T x) - y_i))$

For each instance, it computes the prediction error and multiplies it by the j^{th} feature value $j \in \{0, 1, 2, \ldots m\}$. Once, we get the gradient vector, containing all the partial derivitive, we can use it in Batch Gradient Descent algorithm.