1 Notes on Lecture Material

1.1 Binary Logistic Regression

In class we used $y_n$ for the true label for the $n$-th example. Below, we will be using the notation $t_n$ for this true, target label. Also, in class we used $\hat{y} = \sigma(w^T x)$ to denote the probability of the positive label. Below, we will use the notation $h = \sigma(w^T x)$ for the same thing.

Therefore, if $C_1$ is the name of the positive class, $h = p(C_1|x) = \sigma(w^T x) = \frac{1}{1 + e^{-w^T x}}$.

At test time, we predict positive class $C_1$ if and only if $p(C_1|x) > P(C_2|x)$. We know that $p(C_1|x) + P(C_2|x) = 1$, which means that $p(C_1|x) > P(C_2|x) \Rightarrow P(C_1|x) > 0.5$.

This means that $\frac{1}{1 + e^{-w^T x}} > 0.5 = \frac{1}{2}$, which means that $e^{-w^T x} < 1$, therefore $-w^T x < 0$, which means $z = w^T x > 0$.

Training Logistic Regression parameters $w$ will be done by using the Maximum Likelihood Estimation principle, which says that we want to select the parameters that maximize the probability of the true labels:

$$\hat{w} = \arg \max_w P(t_1, t_2, ..., t_N|w)$$ (1)

The probability $P(t_n|w)$ is equal with:

- $P(t_n|w) = \sigma(w^T x_n) = h_n$, if $t_n = 1$.
- $P(t_n|w) = 1 - \sigma(w^T x_n) = 1 - h_n$, if $t_n = 0$.

Can we show that $P(t_n|w) = h_n^{t_n}(1 - h_n)^{(1-t_n)}$?

- If $t_n = 1$, we get $P(t_n|w) = h_n^{t_n}(1 - h_n)^{(1-t_n)} = h_n^1(1 - h_n)^0 = h_n$. Verified!
- If $t_n = 0$, we get $P(t_n|w) = h_n^{t_n}(1 - h_n)^{(1-t_n)} = h_n^0(1 - h_n)^1 = 1 - h_n$. Verified!
Assume the training examples are **independent identically distributed (i.i.d)**. The likelihood function:

\[
P(t_1, t_2, ..., t_N|w) = \prod_{n=1}^{N} P(t_n) \]

Mathematically, the weight vector \( w \) that maximizes the likelihood is going to be the same as the weight vector that maximizes the **log likelihood**. But this is equivalent with finding the weight vector \( w \) that minimizes the **negative log-likelihood**:

\[
-\ln P(t_1, t_2, ..., t_N|w) = -\sum_{n=1}^{N} t_n \ln h_n + (1 - t_n) \ln (1 - h_n)
\]

\[
-\ln P(t_1, t_2, ..., t_N|w) = -\sum_{n=1}^{N} t_n \ln (w^T x_n) + (1 - t_n) \ln (1 - \sigma(w^T x_n))
\]

### 1.2 Gradient descent

Let \( J(w) = \frac{1}{2}(w - 4)^2 + 1 \) and the initial guess \( w_0 = 0 \), for which \( J(w_0) = 9 \). Let the learning rate be \( \eta = 0.5 \). The gradient of \( J \) is \( \nabla J(w) = w - 4 \).

1. The gradient at \( w_0 \) is \( \nabla J(w_0) = w_0 - 4 = -4 \). The gradient update step is \( w_1 = w_0 - \eta \nabla J(w_0) = 0 - 0.5 \times -4 = 2 \).
   So, \( w_1 = 2 \) for which \( J(w_1) = 3 \).

2. The gradient at \( w_1 \) is \( \nabla J(w_1) = w_1 - 4 = -2 \). The gradient update step is \( w_2 = w_1 - \eta \nabla J(w_1) = 2 - 0.5 \times -2 = 3 \).
   So, \( w_2 = 3 \) for which \( J(w_2) = 1.5 \).

3. The gradient at \( w_2 \) is \( \nabla J(w_2) = w_2 - 4 = -1 \). The gradient update step is \( w_3 = w_2 - \eta \nabla J(w_2) = 3 - 0.5 \times -1 = 3.5 \).
   So, \( w_3 = 3.5 \) for which \( J(w_3) = 1.125 \).

4. and so on ... for ever?

**Bonus points**: For different values of \( \eta \), plot on the same graph \( J(w) \) and the points (in blue) corresponding to the gradient steps. Try \( \eta = 0.1, 0.5, 1, 2, \ldots \).

"Until \( J(w) \) does not improve": how do we quantify this? One method is to look at the **relative change**.

\[
\Delta J = \left| \frac{J(w_t) - J(w_{t-1})}{J(w_{t-1})} \right|
\]

If \( \Delta J \) is too small (0.0001) for a number of epochs (5 or 10), then you may consider stopping.