NOTE ON ERROR FUNCTIONS

This note motivates the usage of various common **error functions** used to train **Neural Networks**.
As a general guiding principle, we keep the overarching goal in mind (we are taking a probabilistic viewpoint here). We would like to use Neural Networks in the context of learning the joint distribution of **data** and **target values**.

$$ p (x,t) = p(t|x) p(x) $$

The motivation for most error functions is to learn the **MLE**, using the model we train as an approximation to $p(t|x)$.

In other words, we use the following framework. We have some **training data** ${x^n, t^n}$, and we want to *maximize*,

$$ \mathcal{L} = \prod_n p(x^n, t^n) = \prod_n p(t^n|x^n) p(x^n) $$

Using the fact that a monotonous transformation does not change the argument of the $argmax_x(f(x))$ function, and the fact that $argmax_x(f(x)) = argmin_x(-f(x))$ we can focus our efforts on,

$$

\min - \ell = - \sum_n \ln p(t^n|x^n) - \sum_n \ln p(x^n)

$$

Where we have no impact on $p(x^n)$, hence we can reduce the problem to,

$$

\min E = \min - \sum_n \ln p(t^n|x^n)

$$

Where we simplify notation by using E as our error function.

Now, the idea is to consider the **conditional probability** $p(t^n|x^n)$ as constructed by our **Neural Network**.

Remark: The setting just described is in fact not limited to Neural Network models

SUM OF SQUARES ERROR

Take c target variables t_k where $k = 1,…,c$ and assume that $p(t_i) \bot p(t_j) \ \forall i \neq j$.

Moreover, assume that $t_k = h_k(x) + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$

Our goal is to approximate the function $h_k(x)$, with a **Neural Network**, where we express the outputs as $y_k(x;w)$, w representing the weight parameters.

The resulting likelihood function is,

$$
\mathcal{L} = \prod_k \prod_n \frac{1}{\sqrt{2\pi\sigma^2}} \exp{- \frac{(y_k^n(x;w) - t_k^n)^2}{2\sigma^2}}
$$

Taking the log and omitting terms independent of w, we end with,

$$
\begin{align}
E & = \frac{1}{2\sigma^2} \sum_n \sum_k (y_k(x^n;w) - t_k^n)^2 + Nc\ln\sigma + \frac{Nc}{2}\ln2\pi \\
& \Leftrightarrow \frac{1}{2} \sum_n \sum_k (y_k(x^n;w) - t_k^n)^2 \\
& \Leftrightarrow \frac{1}{2} \sum_n ||y(x^n;w) - t^n||^2
\end{align}
$$

DERIVATIVES

Choosing as an output function $y = a$ (**linear output activation**), we get the following,

$$ \begin{align}
\frac{\partial E^n}{\partial a} & = \frac{\partial E^n}{\partial y} \frac{\partial y}{\partial a} \\
& = (y^n - t^n)
\end{align}
$$

MINKOWSKI ERROR

Now considering the same setting as for the *sum of squares error*, but exchanging the **error distribution** to the more general,

$$
p(\epsilon) = \frac{R \beta^{\frac{1}{R}}}{2 \Gamma(\frac{1}{R})} \exp - \beta |\epsilon|^R
$$

Using the same procedure as above, we arrive at,

$$
E = \sum_n \sum_k |y_k(x^n; w) - t_k^n|^{R}
$$

Which is known as the **Minkowski R Error**.

Derivatives

It is worthwhile to note the general form of it's derivative with respect to an a weight w_{ji},

$$
\frac{\partial E}{\partial w_{ji}} = \sum_n \sum_k |y_k(x^n;w) - t_k^n|^{R-1} sgn(y_k(x^n;w) - t_k^n) \frac{\partial y_k^n}{\partial w_{ji}}
$$

CROSS ENTROPY FOR TWO CLASSES

Switching perspective from **regression problems** to **classification problems**, in other words, assuming a different kind of **data generating process**, we consider $p(t|x) = y(x;w)^t(1-y(x;w))^{1-t}$ (we model the data a deriving from a *bernoulli process*).

Now this provides a **likelihood function** of the following kind,

$$
\mathcal{L} = \prod_n (y(x^n;w))^{t^{n}} (1 - y(x^n;w))^{1-t^n}
$$

Now, denoting $y(x^n;w)$ as y^n for simplicity, this leads us to an error function of the form,

$$
E = - \sum_n t^n \ln y^n + (1 - t^n) \ln (1-y^n)
$$

which is known as the **cross entropy**.

Derivatives

We get,

$$
\frac{\partial E}{\partial y^n} = \frac{y^n - t^n}{y^n(1- t^n)}
$$

choosing as **output activation function** the **sigmoid activation** (justified below),

$$
y = g(a) = \frac{1}{1 + exp(-a)}
$$

where,

$$
g^{'}(a) = g(a)(1- g(a))
$$

we get, for the **output node** a,

$$ \begin{align}
\frac{\partial E}{\partial a^n} & = \frac{\partial E}{\partial y^n} y^n (1 - y^n) \
& = \frac{(y^n - t^n)}{y^n(1-t^n)} y^n(1 - y^n) \
& = (y^n - t^n)
\end{align}
$$

Remark of sigmoid activation functions

Consider, a classification problem, for which we have,

$$
p(z|C_k) = \exp \big(A(\theta_k) + B(z, \phi) + \theta_k^Tz \big)
$$

so in words, the distribution of the target variable z, conditioned on the *class* k, comes from the **exponential family**.

Then,

$$
p(C_1 | z) = \frac{p(z|C_1)P(C_1)}{p(z|C_1)P(C_1) + p(z|C_2)P(C_2)} = \frac{1}{1 + exp(-a)}
$$

where,

$$\begin{align}
a & = \ln \frac{p(z|C_1)P(C_1)}{p(z|C_2)P(C_2)} \
w & = \theta_1 - \theta_2 \
w_0 & = A(\theta_1) - A(\theta_2) + \ln \frac{P(C_1)}{P(C_2)}
\end{align}
$$

Remark on consistency of cross entropy

Consider the setting in which we approach infinite data, then writing

$$
\begin{align}
\lim_{n \rightarrow \inf}E & = \lim_{N \rightarrow \inf} - \frac{1}{N} \sum_n t^n \ln y^n + (1 - t^n) \ln (1-y^n) \
& = \int \int \big[t \ln y(x) + (1 - t) \ln (1-y(x)) \big] p(t|x) p(x) \,dt \,dx
\end{align}
$$

Denoting,

$$
< t|x > \,= \int t p(t|x) \,dt
$$

We have,

$$
\lim_{n \rightarrow \inf} E = - \int \bigg(\ln y(x) + (1 -) \ln(1- y(x))\bigg) p(x) \,dx
$$

Which has a *minimium* at

$$y(x) = $$

Now noting that,

$$
p(t|x) = \delta_0(t - 1)P(C_1|x) + \delta_0(t)P(C_2|x)
$$

We can conclude that,

$$
y(x) = P(C_1|x)
$$

Remark on multiple independent attributes

If we have multiple independent attributes, we can trivially generalize our approach,

Our new *likelihood* is,

$$
p(t|x) = \prod_k p(t_k|x) = \prod_k y_k^{t_k}(1- y_k)^{t_k}
$$

With corresponding *error function*,

$$
E = - \sum_n \sum_k t_k^n \ln(y_k^n) + (1 - t_k^n) \ln(1 - y_k^n)
$$

Where we will find a *minimum* corresponding to,

$$
y_k(x) = P(C_k|x)
$$

CROSS ENTROPY FOR MULTIPLE CLASSES

If we have more than two classes for the target variable, we can express our distribution of the targets given data as follows,

$$
p(t^n|x^n) = \prod_{k = 1}^c (y^n_k)^{t^n_k}
$$

which leads us to the **error function**,

$$
\min_w E = - \sum_{n = 1}^N \sum_{k = 1}^c t_k^n \ln(y_k^n)
$$

Now for the **output activation function** we pick,

$$
y_k = \frac{\exp a_k}{\sum_{k'} \exp a_{k'}}
$$

We can motivate this choice of activation function by considering the following calculation.

Assuming that $p(z | C_k)$ comes from an **exponential family**, that is

$$
p(z|C_k) = \exp \big(A(\theta_k) + B(z, \phi) + \theta_k^Tz \big)
$$

then applying Bayes theorem,

$$
p(C_k | z) = \frac{p(z|C_k)P(C_k)}{ \sum_{k'} p(z|C_{k'})} = \frac{\exp a_k}{\sum_{k'} \exp a_{k'}}
$$

where,

$$\begin{align}
a_k & = w_k^Tz + w_{k_0} \
w_k & = \theta_k\
w_{k_0} & = A(\theta_k) + \ln P(C_k)
\end{align}
$$

DERIVATIVES

We get,
$$
\begin{align}
\frac{\partial E^n}{\partial y_k} & = - \frac{t_{k'}}{y_{k'}} \
\frac{\partial y_{k'}}{\partial a_k} & = y_{k'} \delta_{kk'} - y_{k'} y_k
\end{align}

$$

Now that implies that,

$$
\begin{align}
\frac{\partial E^n}{\partial a_k} & = \sum_{k'} \frac{\partial E^n}{\partial y_{k'}} \frac{\partial y_{k'}}{\partial a_k} \
& = \sum_{k'} \frac{-t_{k'}}{y_{k'}} (y_{k'} \delta_{kk'} - y_{k'} y_k) \
& = \sum_{k'} -t_{k'} \delta_{kk'} + t_{k'} y_k \
& = -t_k + \sum_{k'} t_{k'} y_k \
& = y_k - t_k
\end{align}
$$

JUSTIFICATION VIA ENTROPY

First, consider the *defintion* of **differential entropy**,

$$
S = - \int p(x) \ln p(x) \,d x
$$

If we take a *random variable* α, then the amount of information to transmit the value of α is $-\ln p(a_k) \ \ if \ \alpha = \alpha_k$.

The *expected amount of information* needed to transmit the value of α is given by,

$$
S(x) = - \sum_k p(\alpha_k) \ln p(\alpha_k)
$$

We usually don't know $p(x)$ and need to encode our message under the assumption of some (hopefully closely) approximating distribution $q(x)$. Under $q(x)$ the information needed to encode x is then $- \ln q(x)$.

The average information needed to encode x is then the **cross-entropy**,

$$
- \int p(x) \ln q(x) \,dx
$$

Note the following connection, to the **expected loss** under a model, when we use $-\ln \mathcal{L}$ as the **loss function**.

$$ \begin{align}
\mathcal{E}[-\ln \mathcal{L}] & = - \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n= 1}^N \ln \tilde{p}(x^n) \
& = - \int p(x) \ln \tilde{p}(x) \,dx

\end{align}
$$

Now assuming our α to be a *discrete random variable*, we get,

$$

- \sum_k P(\alpha_k) \ln Q(\alpha_k)
  $$

Where we want to learn $Q(.)$ so as to minimize the expression. In our usual notation, we express $Q(\alpha_k)$, as learned by our **Neural Network**, with $y_k(x;w)$ and simplified as y_k.
Expressing $P(\alpha_k)$ as t_k, this leaves us with,

$$

- \sum_{k = 1}^c t_k \ln y_k(x)
  $$

And operationally,

$$
\min_w E = - \sum_{n = 1}^N \sum_{k = 1}^c t_k \ln y_k(x)
$$

Which is exactly the **cross entropy**.

GENERAL REMARK

In the preceding derivations, one of the discoveries, is that there exist *good* combinations of *error functions* and *output activation functions*, s.t.,

$$

$$
\frac{\partial E^n}{\partial a_k} = (y_k^n - t_k^n)
$$

The combinations that we have observed, are the following:

1. Linear output activation <-> Squared error loss (Regression)

2. Sigmoid output activation <-> Cross entropy loss (2-Class Classification)

3. Softmax output activation <-> Cross entropy loss (k-Class Classification)

One can consider these *natural pairs*. This is a remarkable finding, but not explored further here.

GENERAL CONDITIONS FOR OUTPUTS TO BE TREATED AS PROBABILITIES

Consider in general a cost function of the following kind,

$$
E^n = \sum_{k = 1}^c f(y_k^n, t_k^n) = \sum_{k = 1}^c f(|y_k^n - t_k^n|)
$$

In the limit of infinite data we get,

$$
\langle E \rangle = \sum_{k = 1}^c \iint f(|y_k - t_k|) p(t|x) p(x) \,dt \,dx
$$

where,

$$
p(t|x) = \prod_{m = 1}^c \bigg(\sum_{l = 1}^c \delta(t_m - \delta_{ml}) P((C_l | x))\bigg)
$$

So we can write,

$$

\begin{align}
\langle E \rangle & = \sum_{k = 1}^c \iint f(|y_k - t_k|) \prod_{m = 1}^c \bigg(\sum_{l = 1}^c \delta(t_m - \delta_{ml}) P((C_l | x))\bigg) p(x) \,dt \,dx \
& = \sum_{k = 1}^c \iint f(|y_k - t_k|) \prod_{m \neq k} \bigg(\sum_{l = 1}^c \delta(t_m - \delta_{ml}) P((C_l | x))\bigg) \bigg(\sum_{l = 1}^c \delta(t_k - \delta_{kl}) P((C_l | x)) \bigg) p(x) \,dt \,dx \
& = \sum_{k = 1}^c \iint f(|y_k - t_k|) \bigg(\sum_{l = 1}^c \delta(t_k - \delta_{kl}) P((C_l | x)) \bigg) p(x) \,dt \,dx \
& = \sum_{k = 1}^c \int f(|y_k - 1|) P(C_k | x) + f(|y_k|) (1 - P(C_k|x)) \ p(x) \,dx

\end{align}
$$

Note, the tricky part is to realize that (from *step 2* to *step 3*),

$$
\int \prod_{m \neq k} \bigg(\sum_{l = 1}^c \delta(t_m - \delta_{ml}) P((C_l | x))\bigg) \,dt
= 1
$$

If our goal is to *minimize the average error per pattern*, we can use the following argument to adhere to this.
Applying some **calculus of variations** (I don't have any background in that so I can only show the step), we want the *functional derivative*,
$$
\frac{\delta \langle E \rangle}{\delta y_k(x)} = - f'(1 - y_k) P(C_k|x) + f'(y_k)[1 - P(C_k|x)] = 0
$$

So,

$$
\frac{f'(1 - y_k)}{f'(y_k)} = \frac{1 - P(C_k | x)}{P(C_k | x)}
$$

Since we want $y_k(x) = P(C_k | x)$ (our functions should represent *posterior proabilities*), this implies,

$$
\frac{f'(1 - y)}{f'(y)} = \frac{1-y}{y}
$$

Solutions for which adhere to the following condition,

$$
f(y) = \int y^r (1-y)^{r-1} \,dy
$$

Hence, if we use **error functions** that satisfy $f(y) = \int y^r (1-y)^{r-1} \,dy$, we can treat the outputs y_k as *posterior probabilities over classes given the data*.

Remarks,

- For $r = 1$ we get $f(y) = \frac{y^2}{2}$ the **squared error**
- For $r = 2$ we get $f(y) = - \ln(1-y)$ which can be shown to be the **cross entropy**