SoftMax and Cross Entropy

Both those two methods serve the classification purpose

SoftMax

Why we need it

The softmax function, also known as the normalized exponential function. There are three purpose to apply softmax function

(1) restrict all outputs **within the interval [0,1]**

(2) all outputs will **add up to 1**

(3) new results remain in the **same order as the original (monotonous)**

Definition

So, it is defended as

$$\sigma(z) = \frac{e^{z_j}}{\sum_{j=1}^{K} e^{z_j}}$$

Cross Entropy

Why we need it

When we evaluate the results for classification, we **find the squared error is too strict**. Usually, we only require the highest value is the right label. For instance, we have a, b, c three possible outputs, if b is the ground truth, we only care about whether the value of b is the largest. If the output of b is 0.6, it does not matter whether the value of a and c is (0.2,0.2) or (0,0.4). But from squared error estimation, the first situation is preferable.

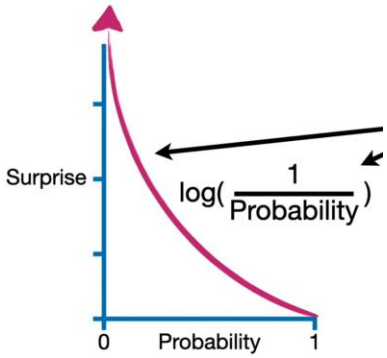
To overcome the bias from l_2 norm, we apply Cross Entropy

Definition

What is the surprise

The surprise is the metric we use to against probability. For instance, $P(A) = 0.9$. Then if A was happened, there is not too much surprise for us, since the probability of A is so high. So, in the very beginning, statisticians define surprise as the invert of probability, note as $Surprise(A) = \frac{1}{P(A)}$.

However, when $P(A) = 1$, which means the occurrence of A is absolute, if we got A, we would not feel surprised at all. But from the equation defined above, we would get $Surprise(A) = \frac{1}{P(A)} = \frac{1}{1} = 1$. Therefore, we take the log transformation on it, the equation becomes $Surprise(A) = \log\left(\frac{1}{P(A)}\right)$. When $P(A) = 1$, we get $Surprise(A) = \log(1) = 0$. When $P(A) = 0$, we get $Surprise(A) = \log\left(\frac{1}{0}\right) = undefined$. The graph of it looks like



In the convention, if we calculate the surprise of two possible outcomes, we use the log base 2, which is $Surprise = \log_2(\frac{1}{Probability})$

Property of the surprise

If we calculate the surprise of two possible outcomes, $P(A) = 0.9, P(B) = 0.1$, then we would get $Surprise(A) = \log_2\left(\frac{1}{0.9}\right) = 0.15, Surprise(B) = \log_2\left(\frac{1}{0.1}\right) = 3.32.$

But what is the surprise of combination of AAB? Since the probability of AAB is $P(A) * P(A) * P(B)$, So $Surprise(AAB) = \log_2\left(\frac{1}{P(A)*P(A)*P(B)}\right) = \log_2\left(\frac{1}{P(A)}\right) + \log_2\left(\frac{1}{P(A)}\right) + \log_2\left(\frac{1}{P(B)}\right)$

So, we get the first property: the surprise of a combination equals the sum of surprise of each element, which note as

$$Surprise(AAB) = Surprise(A) + Surprise(A) + Surprise(B)$$

And the expectation of the surprise equals to

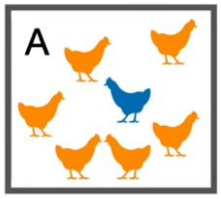
$$E(surprise) = \frac{P(A) * n * Surprise(A) + P(B) * n * Surprise(B)}{n} = \sum xP(X = x)$$

With x is a specific value for Surprise

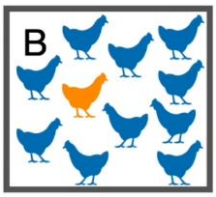
Now, plug the equation for surprise for x, we get the definition of the entropy

$$Entropy = \sum \log\left(\frac{1}{P(x)}\right)P(x) = -\sum P(x)\log\left(P(x)\right)$$

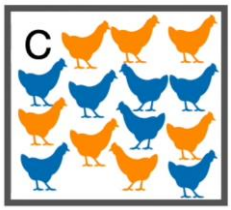
Examples



$$Entropy = \frac{6}{7}\log_2\left(\frac{1}{6/7}\right) + \frac{1}{7}\log_2\left(\frac{1}{1/7}\right) = 0.59$$



$$Entropy = \frac{10}{11}\log_2\left(\frac{1}{10/11}\right) + \frac{1}{11}\log_2\left(\frac{1}{1/11}\right) = 0.44$$



$$Entropy = \frac{1}{2}\log_2\left(\frac{1}{1/2}\right) + \frac{1}{2}\log_2\left(\frac{1}{1/2}\right) = 1$$

Therefore, we could use entropy to **measures the difference / similarity in the number** of orange and blue chickens in each area. If we get Entropy = 1, it means we have balanced number of two types of chicken. If we **increase the difference between number of different chickens, we lower the entropy**.

Cross entropy in Deep learning

Cross entropy is a little different from entropy, it is the expected entropy under the true(observed) distribution P when we use a coding scheme optimized for a predicted distribution Q. It defines as

$$Cross\ Entropy = -\sum_{c=1}^{M} Observed_c * \log\left(Predicted_c\right)$$

M is the number of output classes

Examples

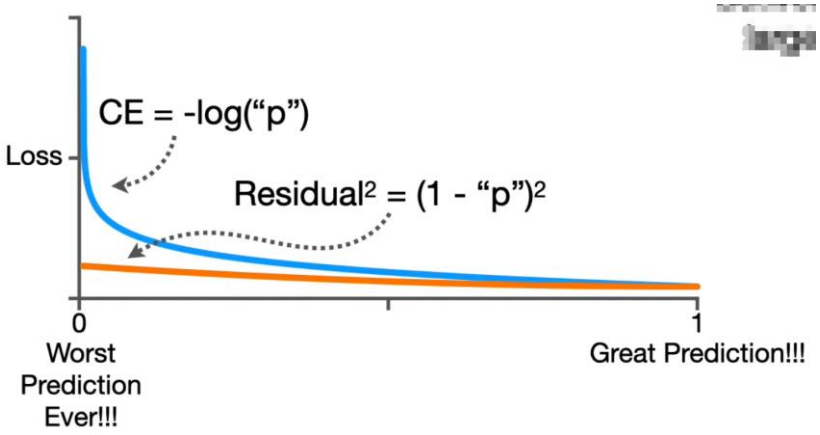
If we have three outputs from a neural network as follows, how we calculate the cross-entropy value?

Let's take Setosa as the example

$$\begin{aligned} Cross\ Entropy &= -\sum_{c=1}^{M} Observed_c * \log(Predicted_c) \\ &= -(1 * \log(0.57) + 0 * \log(0.58) + 0 * \log(0.52)) \\ &= -\log(0.57) = 0.56 \end{aligned}$$

Species	Output from Neural network	Cross entropy
Setosa	0.57	=-1*log (0.57) =0.56
Virginica	0.58	=-1*log (0.58) =0.54
Versicolor	0.52	=-1*log (0.52) =0.65

The total error of the model is simply summing all cross entropy together, $0.56 + 0.54 + 0.65 = 1.75$



As show above, the cross entropy take punishment when model get worst predictions