Entropy, Cross-Entropy and KL-Divergence

My greatest concern was what to call it. I thought of calling it 'information,' but the word was overly used, so I decided to call it 'uncertainty.' When I discussed it with John von Neumann, he had a better idea. Von Neumann told me, 'You should call it entropy, for two reasons. In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name. In the second place, and more important, no one really knows what entropy really is, so in a debate you will always have the advantage.'

-Claude Shannon

The notions such as Entropy, Cross-Entropy and KL-Divergence are based on the Theory of Information which is found by Claude Shannon. To introduce Shannon, he is an American mathematician, electrical engineer and cryptograph. According to "A Mathematical Theory of Communication" essay which was published at 1948, he found the Theory of Information. His purpose was sending a message from sender to receiver with an efficient and secure way.

In digital world messages are consist of bits. Let's discuss about bits. A bit can be either 1 or 0. Of course all bits are not useful, some of them are un useful some of them can be just error bits. Therefore, when we send a message, we want to send the message as much as efficient. On this topic Shannon tells this in his theory.

Bit = Uncertainty divided by 2

Now let's talk about the notions of Entropy, Cross-Entropy and KL-Divergence.

Entropy is quantity of the information inside a message; thus, it tells the how much an event can a surprise.

Let's assume we had a university choice and with %65 possibility we will start to study at university A, %35 university B. When the results are announced If ÖSYM says that we won the B university our uncertainty will be reduced by 1.51 bits of information. if ÖSYM says that we won the A university our uncertainty will be reduced by 0.62 bits of information. So, how can get these numbers. Let's look at them.

Information ->> - log2(P) we will use this formula.

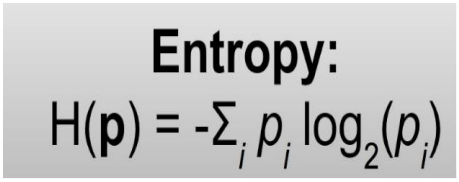
- log2(0.35) = 1,51

- log2(0.65) = 0.62

Here P is the possibility of the event can happen.

Let's look at how much average information we can get from OSYM.

(%65 x 0.62) + (%35 x 1,51) = 0.93 This calculated value shows us the Entropy. Here you can find the formula below.

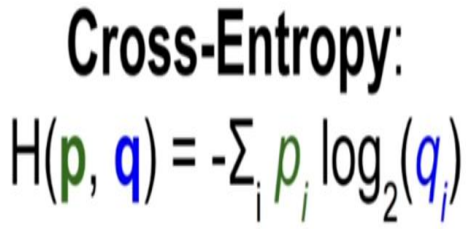


Cross-Entropy, in a nutshell average message length. In Machine learning it is used as cost while training the classificators.

Let's assume there is an election in the country and let's make an estimation of A party is going to win the election with %40 polls, party B will get %30, party C will get %20 and party D will get %10. Let's encode all batches with 4 bits. Now let's calculate the Entropy of this estimation. Since I have already explained the calculation method in details, I will give you the result which is 1.846. Our entropy is 1.846, but we encoded all batches with 4 bits. So we encode with 4 bits but the groups only get 1.846 useful bits. In order to fix this, what we should do? My guess is coding the high possibility party with small bits and low possibility parties with bigger bits. I think this will solve our problem. Let's try.

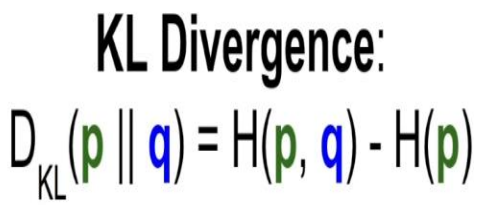
Regarding with the new assumption party A gets 2 bit, B gets 3, C gets 4 and D gets 5-bit coding.

(%40x2)+ (%30x3)+ (%20x4)+ (%10x5) = 3. As a result of this calculation, we get the result as 3. This will be our next Cross-Entropy. Now our result is better since we have calculated a smaller number as a Cross-Entropy. Here you can find the formula below.



As for KL-Divergence, you can find it by subtracting Entropy from Cross-Entropy

KL Divergence = Cross-Entropy - Entropy



In our example above, cross entropy was 3 and entropy was 1,846. So KL-Divergence here would be

3 – 1.846 = 1.154 bits

Now ı will show the application of the example we made at the top.

```python
from math import log2

# calculate the kl divergence KL(P || Q)
def kl_divergence(p, q):
    return sum(p[i] * log2(p[i]/q[i]) for i in range(len(p)))

# calculate entropy H(P)
def entropy(p):
    return -sum([p[i] * log2(p[i]) for i in range(len(p))])

# calculate cross entropy H(P, Q)
def cross_entropy(p, q):
    return entropy(p) + kl_divergence(p, q)

# define data
p = [0.4, 0.3, 0.2, 0.1]
q = [0.25, 0.125, 0.0625, 0.03125]
# calculate H(P)
en_p = entropy(p)
print('H(P): %.3f bits' % en_p)
# calculate kl divergence KL(P || Q)
kl_pq = kl_divergence(p, q)
print('KL(P || Q): %.3f bits' % kl_pq)
# calculate cross entropy H(P, Q)
ce_pq = cross_entropy(p, q)
print('H(P, Q): %.3f bits' % ce_pq)
```

```
H(P): 1.846 bits
KL(P || Q): 1.154 bits
H(P, Q): 3.000 bits
```

References :

https://www.youtube.com/watch?v=ErfnhcEV1O8&ab_channel=Aur%C3%A9lienG%C3%A9ron

https://medium.com/@emreeyukseel?p=89d26735789f

https://en.wikipedia.org/wiki/Cross_entropy