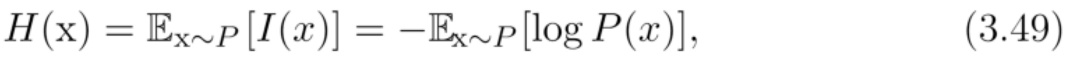
# The intuition behind Shannon's Entropy

[WARNING: TOO EASY!]

Aerin Kim Sep 30, 2018 · 6 min read ★

In Chapter 3.13 Information Theory of *The Deep Learning Book by Ian Goodfellow, it says:*

> *we define the* ***self-information of an event X = x*** *to be*
>
> ***I(x) = −log P (x)***
>
> *Our definition of I(x) is therefore written in units of nats. One nat is the amount of information gained by observing an event of probability 1/e.*
>
> *…*
>
> *We can quantify the amount of uncertainty in an entire probability distribution using the* ***Shannon entropy.***



$$H(\mathrm{x}) = \mathbb{E}_{\mathrm{x}\sim P}[I(x)] = -\mathbb{E}_{\mathrm{x}\sim P}[\log P(x)], \tag{3.49}$$

The definition of Entropy for a probability distribution (from The Deep Learning Book)

## But what does this formula mean?

For anyone who wants to be fluent in Machine Learning, understanding Shannon's entropy is crucial. Shannon's Entropy leads to a function which is the bread and butter of an ML practitioner — **the cross entropy** that is heavily used as a loss function in classification and also **the KL divergence** which is widely used in variational inference.

## To understand entropy, we need to start thinking in terms of the "bits".

Bits are either 0 or 1.

Therefore, with 1 bit, we can represent 2 different facts (aka **information**), either one or zero (or True or False). Let's say you are a commander in World War II in 1945. Your telegrapher told you that if Nazis surrender, he will send you a '1' and if they don't, he will send you a '0'.

In 2018, you can send the exact same information on a smartphone typing

"The war is over" (instead of 1 bit, we use 8 bits * 15 characters = **120 bits**)

"The war is not over" (8 bits * 19 characters = **152 bits**)

So, we're using more than 100 bits to send a message **that could be reduced to just one bit.**

Let's say there are four possible war outcomes tomorrow instead of two. 1) Germany and Japan both surrender. 2) Germany surrenders but Japan doesn't. 3) Japan surrenders but Germany doesn't. 4) Both don't surrender. Now your telegrapher would need 2 bits (00,01,10,11) to encode this message. In the same way, he would need only 8 bits even if there are 256 different scenarios.

A little more formally, **the entropy of a variable** is the "amount of information" contained in the variable. You can think of variable as **news from the telegrapher**. The news can be anything. It doesn't have to be 4 states, 256 states, etc. In real life, news can be millions of different facts.

Now, back to our formula 3.49:

$$H(\mathrm{x}) = \mathbb{E}_{\mathrm{x}\sim P}[I(x)] = -\mathbb{E}_{\mathrm{x}\sim P}[\log P(x)], \tag{3.49}$$

The definition of Entropy for a probability distribution (from The Deep Learning Book)

**I(x)** is **the information content of X**.

I(x) itself is **a random variable.** In our example, the possible outcomes of the War. Thus, **H(x)** is **the expected value of every possible information.**

Using the definition of expected value, the above equation can be re-written as

$$H(X) = -\sum_{x} P(x)\cdot \log P(x)$$

$$= \sum_{x} \underbrace{P(x)}_{\text{The prob. of event X}} \cdot \underbrace{\log\left(\frac{1}{P(x)}\right)}_{\text{WHAT IS THIS ?}}$$

Because -log P(x) = log (1/P(x))

## Wait… Why do we take the reciprocal of probability?

H(X) is the **total amount of information** in an **entire** probability distribution. This means **1/p(x)** should be **the information of each case** (winning the war, losing the war, etc).

### Then the question is…

### Why is 1/p(x) the amount of information?

Lets say there is 50 50 chance that the Nazis would surrender (p = 1/2). Then, if your telegrapher tells you that they did surrender, you can eliminate the uncertainty of total 2 events (both surrender and not surrender), which is the reciprocal of p (=1/2).

When your events are all equally likely to happen and you know that one event just happened, you can eliminate the possibility of every other event (total 1/p events) happening. For example, let's say there are 4 events and they are all equally likely to happen (p = 1/4). When one event happens, it says the other three events didn't happen. Thus, we know what happened to total 4 events.

### What about not equally likely events?

Let's say there is a 75% chance that Nazis will surrender and a 25% chance that they won't.

> *How much information does the event 'surrender' have?*

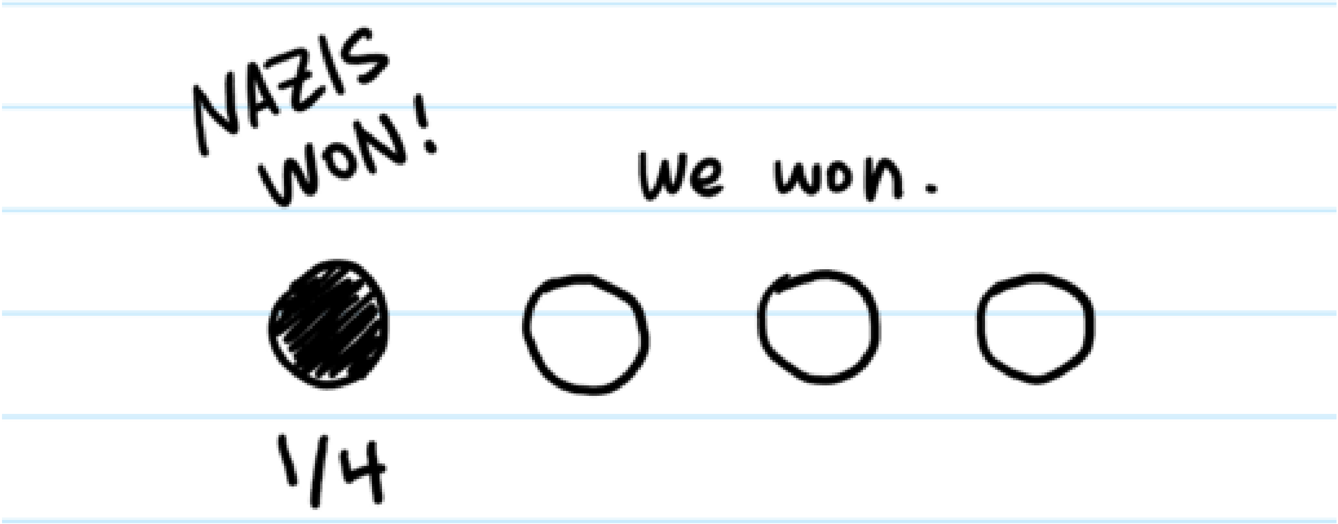
**log (1/0.75) = log(1.333) = 0.41** (log base 2 omitted going forward)

> *How much information does the event 'not surrender' have?*

**log (1/0.25) = log(4) = 2**

**As you see, the unlikely event has a higher entropy.**

Here is the intuition on **why information is the reciprocal of the probability.**

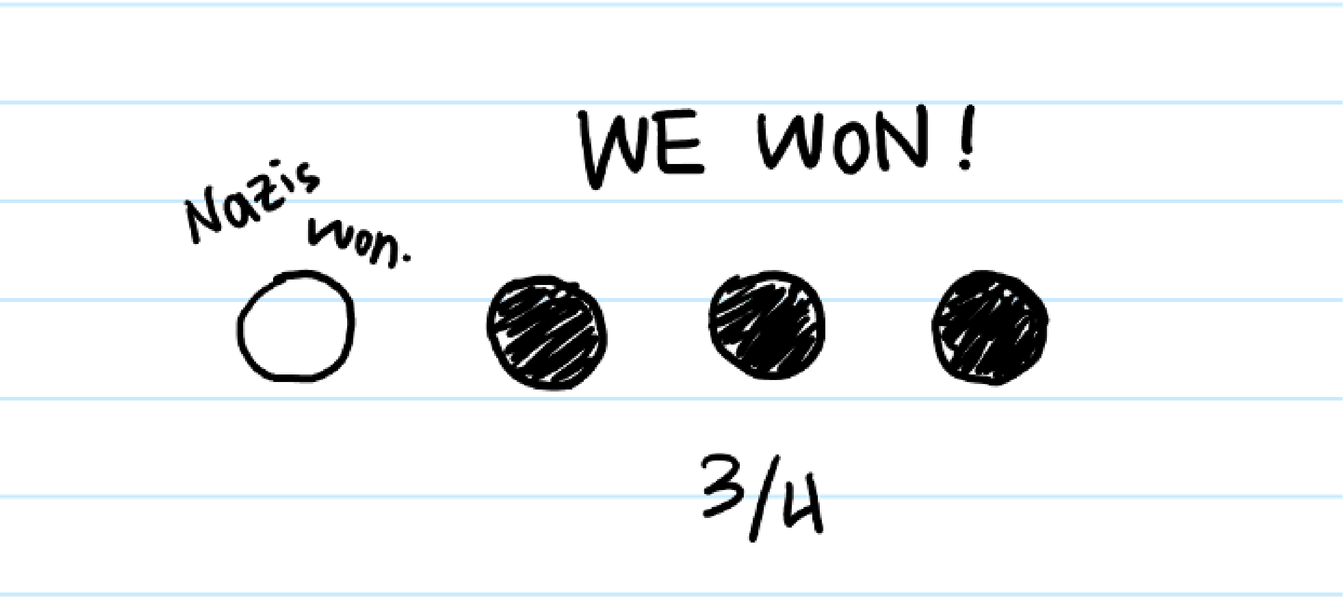


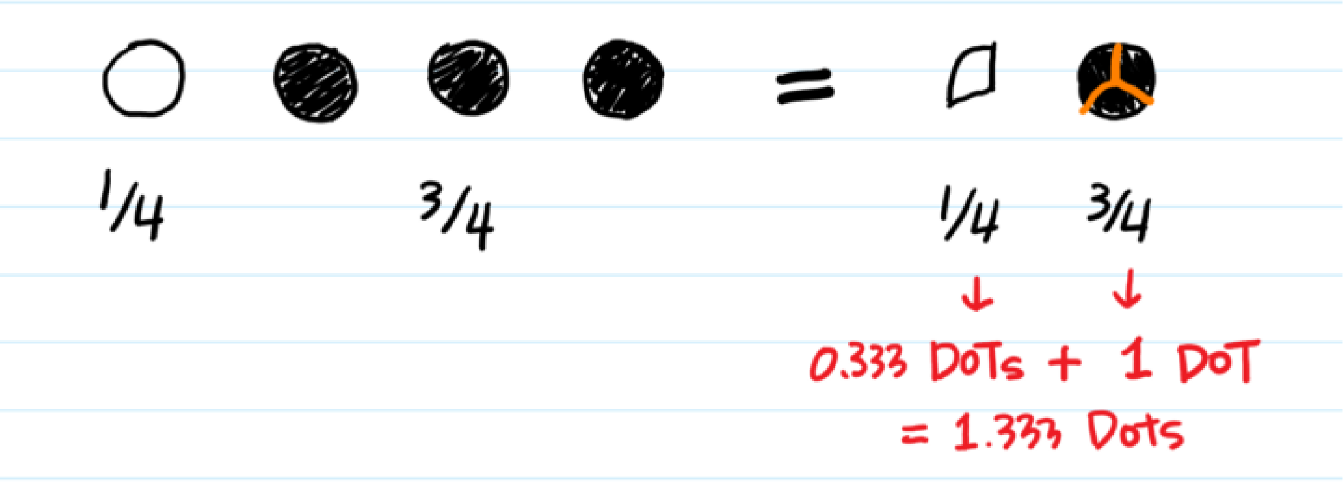
The black dot is the news.

By knowing the black dot, we can eliminate 3 other white dots at the same time.

**Total 4 dots (total information) burst**.

Now, by knowing **ONE** black dot, how many **TOTAL dots** can we burst?





We can eliminate total 1 and 1/3 = 1.333 dots, which is **the reciprocal of 3/4.**

> The total amount of dots you can burst = the information content in EACH news.

Thus, **the information in EVERY possible news** is 0.25 * log(4) + 0.75 * log(1.333)= 0.81 (Shannon's entropy formula.)

Now we know where 1/p comes from. But why the log? Shannon thought that the information content of anything can be measured in bits. To write a number *N* in bits, we need to take a log base 2 of *N*.

### Takeaway

If we have P(win) =1, the entropy is 0. It has 0 bits of uncertainty. (-log1 = 0)

Note that thermodynamic "entropy" and the "entropy" in information theory both capture increasing randomness.

Notice that in our example, with the "equally likely" messages, the entropy is higher (2 bits) than the "not equally likely" messages (0.81 bits). This is because there is less uncertainty in "not equally likely" messages. One event is more likely to come up than the other. This reduces the uncertainty.

For implementation addicts, here is the Python code.

See how the more the number of characters, the greater the uncertainty (entropy).

```python
import math
import random

def H(sentence):
    """
    Equation 3.49 (Shannon's Entropy) is implemented.
    """
    entropy = 0
    # There are 256 possible ASCII characters
    for character_i in range(256):
        Px = sentence.count(chr(character_i))/len(sentence)
        if Px > 0:
            entropy += - Px * math.log(Px, 2)
    return entropy
```

```
# The telegrapher creates the "encoded message" with length 10000.
# When he uses only 32 chars
simple_message ="".join([chr(random.randint(0,32)) for i in
range(10000)])

# When he uses all 255 chars
complex_message ="".join([chr(random.randint(0,255)) for i in
range(10000)])

# Seeing is believing.

In [20]: H(simple_message)
Out[20]: 5.0426649536728 the

In [21]: H(complex_message)
Out[21]: 7.980385887737537

# The entropy increases as the uncertainty of which character will
be sent increases.
```

In the next post, I'll explain how we extend Shannon's Entropy to Cross Entropy and KL Divergence.

If you like my post, could you please clap? It gives me motivation to write more. :)

Information Theory   Entropy   Machine Learning   Deep Learning   Python