Information Theory for Machine Learning

Nipun Batra

June 16, 2023

IIT Gandhinagar

The Data Compression Problem

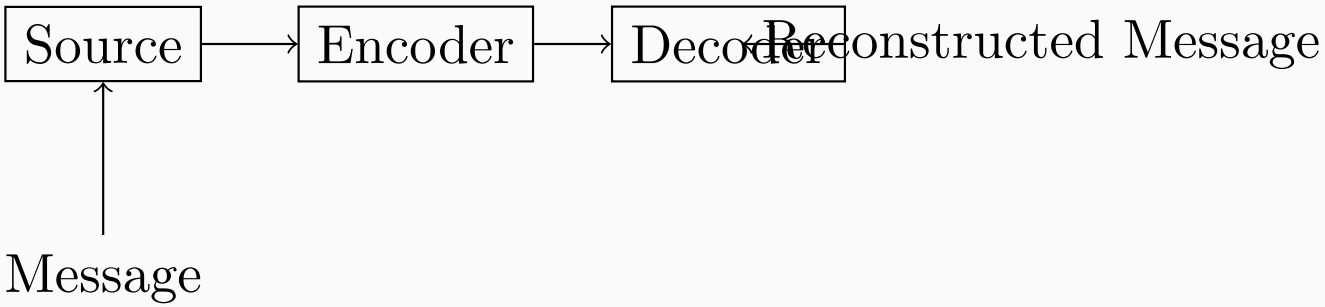


Figure 1: Data Compression Problem

Self Information

- What is more surprising: Snowing in Kashmir or Snowing in Gandhinagar?
- To formalize, let us assume that the probability of snowing in Kashmir is p_1 and that in Gandhinagar is p_2, and that $p_1 >> p_2$.
- How can we quantify the surprise?

Self Information

- Events that are less likely to occur are more surprising.
- Also, if an event is 100% likely to occur, it is not surprising at all.
- Also, if two events are independent, then the surprise of both of them occurring together is the sum of the surprise of each of them occurring individually.
- So, we need a function that maps probability to a number. Function should be: monotonic, and additive, and is 0 when the probability is 1.
- The function is $I(x) = -\log_2(x)$ also called the self information or surprisal.

Self Information

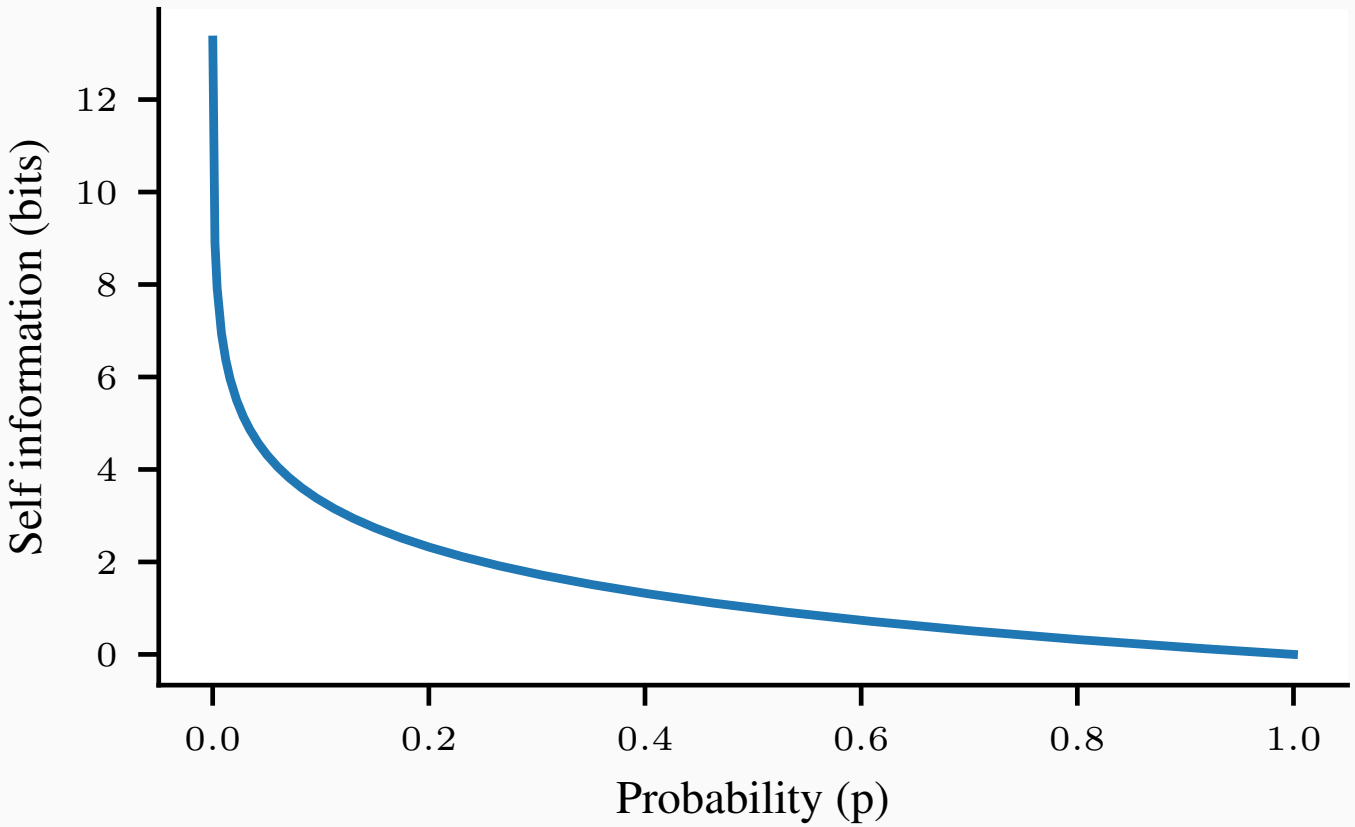
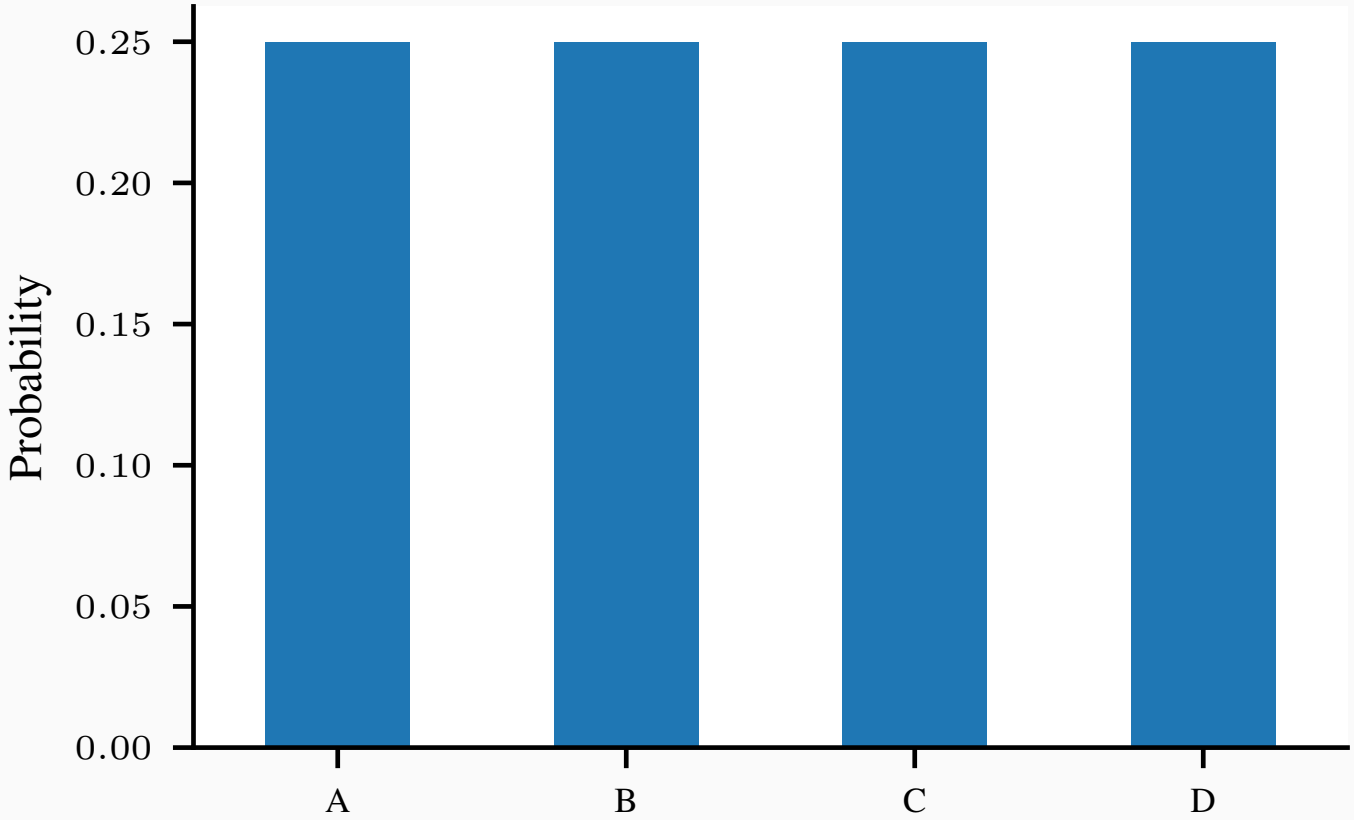


Figure 2: Self Information

Self Information

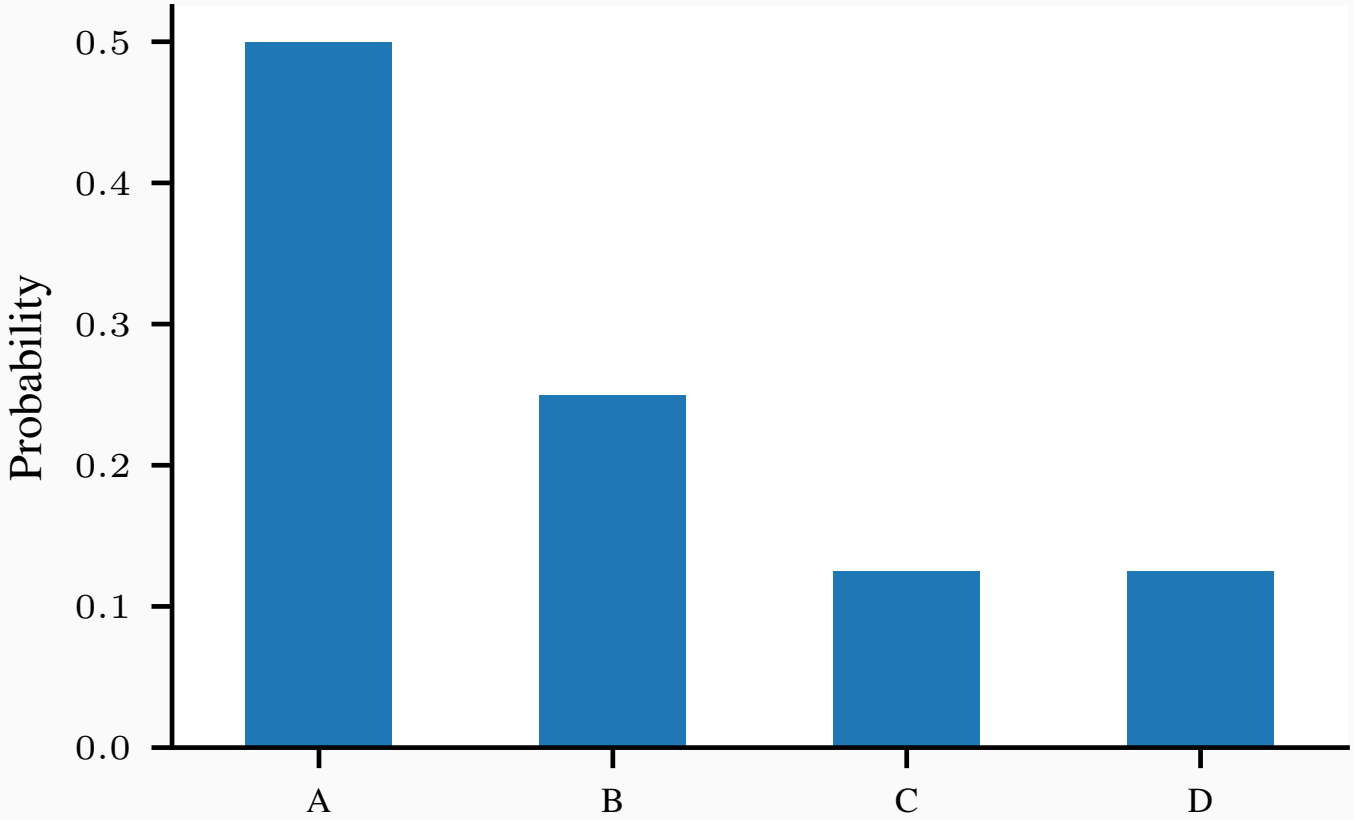
Consider a categorical random variable X with 4 possible outcomes: A, B, C, D. The probability of each of these outcomes is 0.25. What is the self information of each of these outcomes?



$I(A) = I(B) = I(C) = I(D) = 2$ bits.

Self Information

Consider a categorical random variable X with 4 possible outcomes: A, B, C, D. The probability these outcomes is 0.5, 0.25, 0.125, and 0.125. What is the self information of each of these outcomes?



$I(A) = 1$ bit, $I(B) = 2$ bits, $I(C) = I(D) = 3$ bits.

Self Information

Proof on additivity of self information: Consider two independent random variables X and Y with PMFs $p_X(x)$ and $p_Y(y)$ respectively. The joint PMF is $p_{X,Y}(x,y) = p_X(x)p_Y(y)$. The self information of the joint PMF is:

$$\begin{aligned} I(X=x, Y=y) &= -\log_2(p_X(x)p_Y(y)) \\ &= -\log_2(p_X(x)) - \log_2(p_Y(y)) \\ &= I(X=x) + I(Y=y) \end{aligned}$$

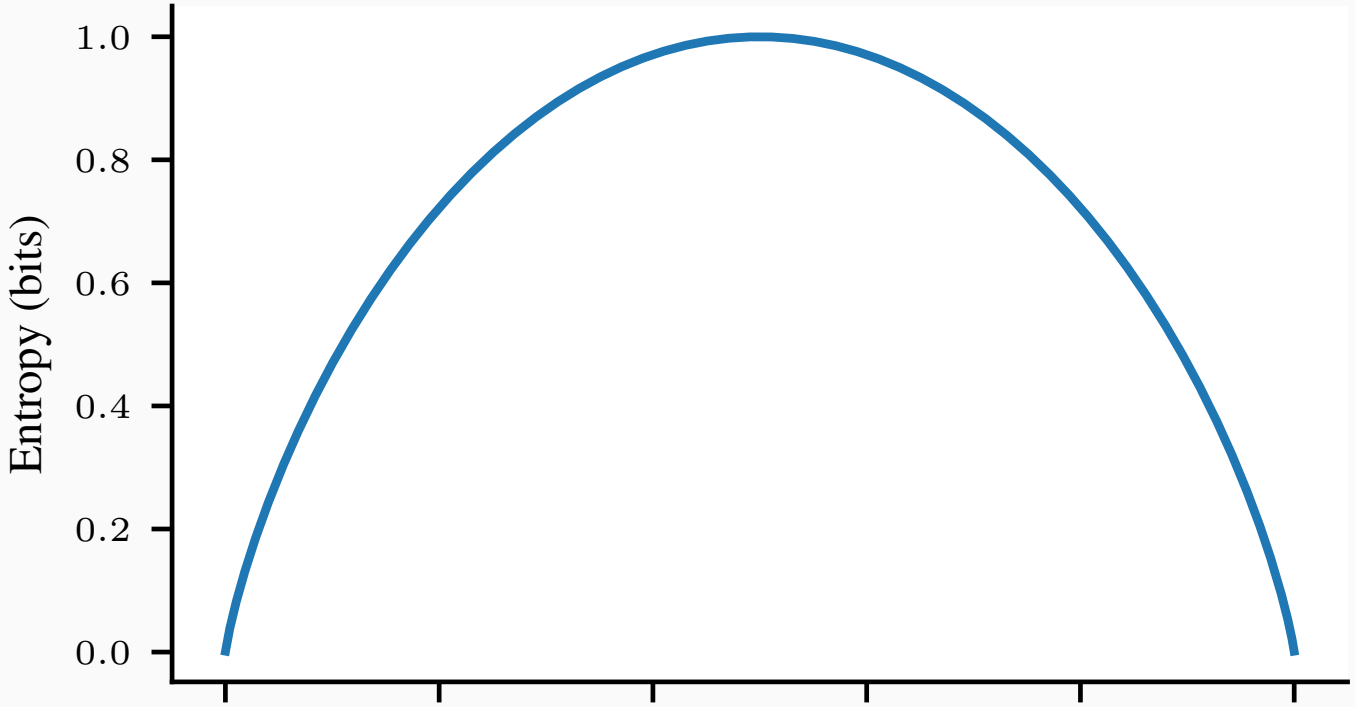
Entropy

- The entropy of a random variable is the expected value of the self information.
- $H(X) = \mathbb{E}_{X \sim p(x)}[I(X)] = \mathbb{E}_{X \sim p(x)}[-\log_2(p(x))]$
- The entropy of a random variable is the expected number of bits required to encode the random variable.
- The entropy of a random variable is the minimum number of bits required to encode the random variable.

Entropy

For a Bernoulli random variable X with probability p of success, the entropy is:

$$\begin{aligned} H(X) &= \mathbb{E}_{X \sim p(x)}[-\log_2(p(x))] \\ &= -\log_2(p) \times p - \log_2(1-p) \times (1-p) \\ &= -p \log_2(p) - (1-p) \log_2(1-p) \end{aligned}$$



Entropy

For a k class categorical random variable X with probability p_i of class i, the entropy is:

$$H(X) = \mathbb{E}_{X \sim p(x)}[-\log_2(p(x))]$$
$$= -\sum_{i=1}^{k} p_i \log_2(p_i)$$

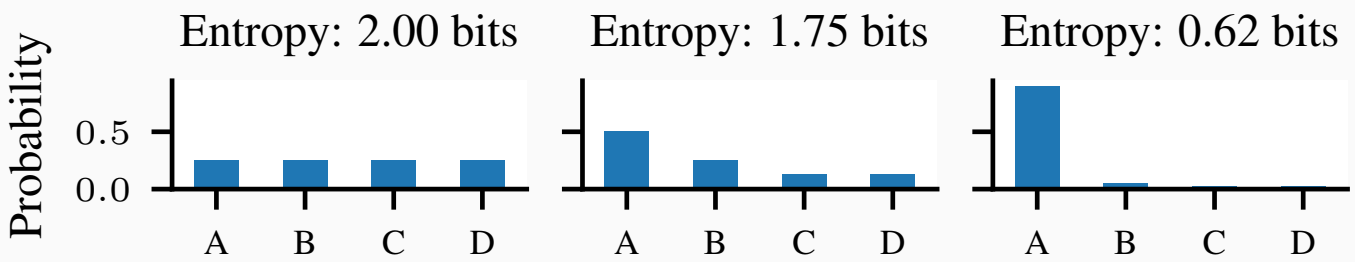


Figure 4: Entropy

Code Length

Let us assume our symbols are: A, B, C, D. Let us assume that the probability of each of these symbols is 0.25. Let us assume we use the following code to encode these symbols:

$$A \rightarrow 00$$

$$B \rightarrow 01$$

$$C \rightarrow 10$$

$$D \rightarrow 11$$

What is the expected code length?

Expected code length $= L(X) = \sum_{i=1}^{4} p_i \times l_i = 2$ bits.

Code Length

Let us assume our symbols are: A, B, C, D. Let us assume that the probability of these symbols is 0.5, 0.25, 0.125, and 0.125. Let us assume we use the following code to encode these symbols:

$$A \rightarrow 00$$

$$B \rightarrow 01$$

$$C \rightarrow 10$$

$$D \rightarrow 11$$

What is the expected code length?

Expected code length $= \sum_{i=1}^{4} p_i \times l_i = 2$ bits. But, is this the most efficient code? No! What is the entropy of this random variable? $H(X) = 1.75$ bits.

Code Length

Let us assume our symbols are: A, B, C, D. Let us assume that the probability of these symbols is 0.5, 0.25, 0.125, and 0.125. Using fixed length codes, we need 2 bits to encode each symbol.

Key idea: Use shorter codes for more frequent symbols and longer codes for less frequent symbols.

How about the following code?

$$A \rightarrow 0$$

$$B \rightarrow 10$$

$$C \rightarrow 110$$

$$D \rightarrow 111$$

Expected code length $= \sum_{i=1}^{4} p_i \times l_i = 1.75$ bits.

Optimum Code Length

By definition, we saw that the entropy of a random variable is the minimum number of bits required to encode the random variable.

This means that the expected code length of any code is always greater than or equal to the entropy of the random variable.

Relationship between entropy and expected code length

$$L(X) = \sum_{i=1}^{n} p_i \times l_i \geq H(X) \tag{1}$$

Optimum length for each symbol is given by:

$$l_i = -\log_2(p_i) \tag{2}$$

Huffman Encoding

- Huffman encoding is a method to construct a variable length code for a random variable.
- The code is constructed such that the expected code length is equal to the entropy of the random variable.
- The code is constructed such that the code is a prefix code.

Cross Entropy

Suppose we have four symbols A, B, C, D with probabilities 0.5, 0.25, 0.125, and 0.125 respectively. Let us call this distribution $p(x)$. We want to transmit some data using these symbols.

The optimum encoding scheme is: A: 0, B: 10, C: 110, D: 111.

But, for some reason, we believe that the four symbols are distributed as per $q(x)$: 0.25, 0.25, 0.25, and 0.25.

For this distribution, the optimum encoding scheme is: A: 00, B: 01, C: 10, D: 11.

Cross Entropy

Cross Entropy $H(p, q)$

The cross-entropy between two probability distributions p and q over the same underlying set of events measures the **average** number of bits needed to identify an event drawn from the set if a coding scheme used for the set is optimized for an estimated probability distribution q, rather than the true distribution p.

Cross Entropy

Cross Entropy $H(p, q)$: Optimum code length for transmitting data distributed as per p via a code as per $q = \sum_{i=1}^{4} p_i \times l_i$.
$l_q(A) = l_q(B) = l_q(C) = l_q(D) = -\log_2(q(D)) = 2$ bits.

$H(p, q) = 2$ bits.

Entropy $H(p)$: Code length for transmitting data distributed as per p via a code as per $p = \sum_{i=1}^{4} p_i \times l_i = 1.75$ bits.

KL divergence

KL Divergence $D_{KL}(p||q)$

The KL divergence between two probability distributions p and q over the same underlying set of events measures the **difference** in the **average** number of bits needed to identify an event drawn from the set if a coding scheme used for the set is optimized for an estimated probability distribution q, rather than the true distribution p.

$$
\begin{aligned}
D_{KL}(p||q) &= \sum_{i=1}^{k} p_i \log_2 \frac{p_i}{q_i} \\
&= \sum_{i=1}^{k} p_i \log_2 p_i - \sum_{i=1}^{k} p_i \log_2 q_i \\
&= H(p,q) - H(p)
\end{aligned}
$$

Example relating KL divergence and Cross Entropy

True probability distribution p: A: 0.4, B: 0.3, C: 0.2, D: 0.1
Estimated probability distribution q: A: 0.15, B: 0.55, C: 0.05, D: 0.25.
Entropy $H(p) = 1.8464$ bits.
Cross Entropy $H(p, q) = 2.4179$ bits.
Huffman code for p: A: 0, B: 10, C: 110, D: 111.
Huffman code for q: A: 10, B: 0, C: 110, D: 111.
Average code length for transmitting data distributed as per p via code as per p is: 0.4*1 + 0.3*2 + 0.2*3 + 0.1*3 = 1.9 bits.
Average code length for transmitting data distributed as per p via code as per q is: 0.4*2 + 0.3*1 + 0.2*3 + 0.1*3 = 2.2 bits.
$D_{KL}(p||q) = 2.4179 - 1.8464 = 0.5714$ bits.

Relationship between KL divergence and Maximum Likelihood Estimation

Let us assume we have a dataset $D = \{x_1, x_2, \ldots, x_n\}$ for a two class classification problem. Let us assume that the class labels are $y_1, y_2, \ldots, y_n$.