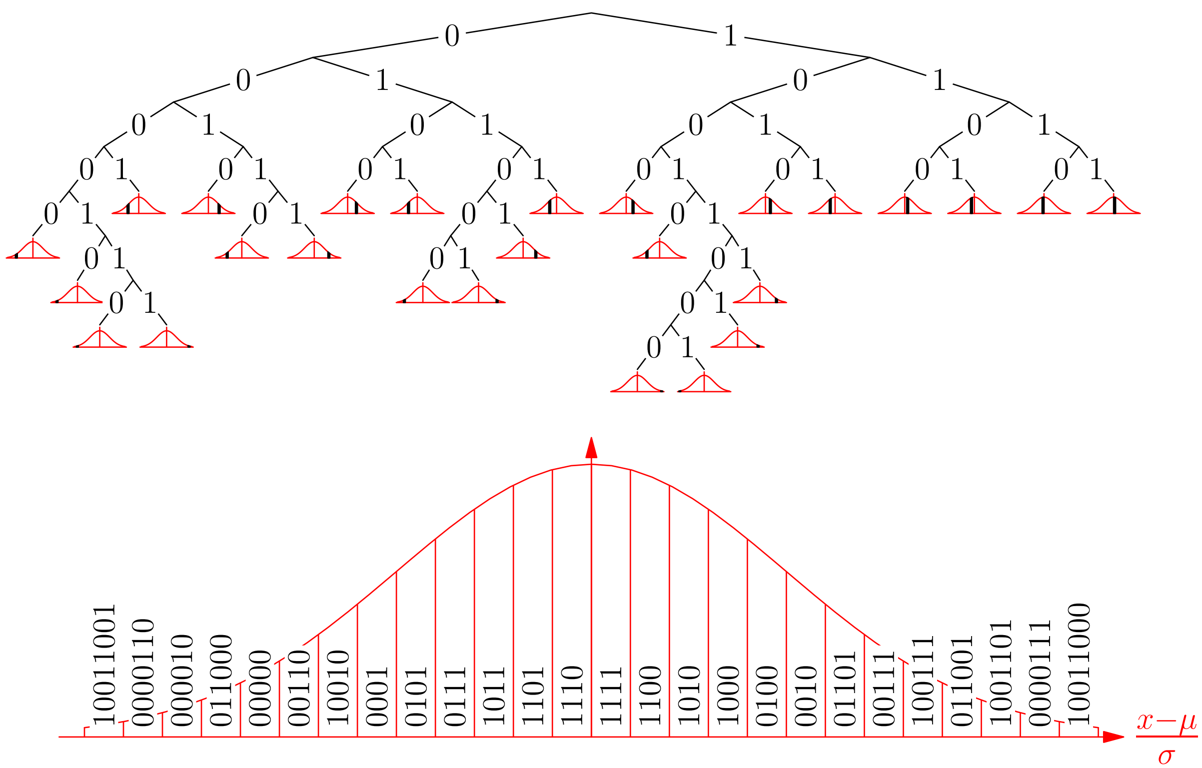
Advanced Machine Learning

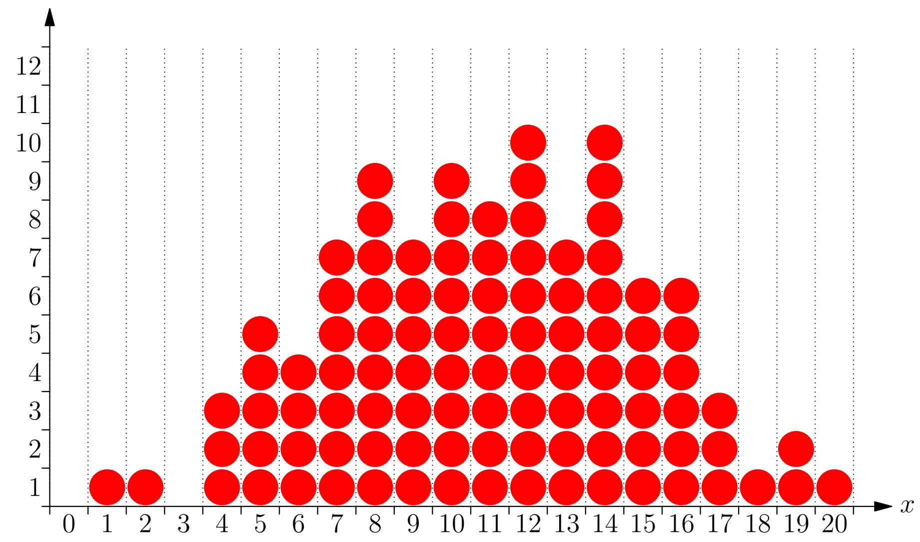
Entropy



Entropy, Coding, Maximum Entropy

Outline

1. **Measuring Uncertainty**
2. Code Length
3. Maximum Entropy



Measuring Uncertainty

- What is more uncertain tossing a coin three times or throwing a dice
- The answer depends on whether you care about the order of the coin tosses
- But, how do we answer such a question?
- Let X be a random variable denoting the possible outcomes
- Interestingly, Shannon entropy give a precise answer

$$H_X = -\sum_{x \in \mathcal{X}} \mathbb{P}(X = x) \log_2(\mathbb{P}(X = x))$$

Let's Calculate

- For an honest dice $D \in \{1,2,3,4,5,6\}$ and $\mathbb{P}(D=i) = 1/6$ so

$$H_D = -\sum_{i=1}^{6} \frac{1}{6} \log_2\left(\frac{1}{6}\right) = -\log_2\left(\frac{1}{6}\right) = \log_2(6) \approx 2.584\,\text{bits}$$

- For an honest coin where we care about the order so $C \in \{000, 001, \ldots, 111\}$ the $\mathbb{P}(C=i) = \frac{1}{8}$ and

$$H_C = -\sum_{i=0}^{7} \frac{1}{8} \log_2\left(\frac{1}{8}\right) = -\log_2\left(\frac{1}{8}\right) = \log_2(8) = 3\,\text{bits}$$

- This clearly makes sense: there are more possible outcomes; all equally likely

Unordered Coin Toss

- What if we don't care about the order of the out-come then $\mathbb{P}(HHH) = \mathbb{P}(TTT) = 1/8$, $\mathbb{P}(HHT) = \mathbb{P}(HTT) = 3/8$ so

$$H_U = -\frac{1}{4}\log_2\left(\frac{1}{8}\right) - \frac{3}{4}\log_2\left(\frac{3}{8}\right) \approx 1.811\,\text{bits}$$

- This seems reasonable, although it is not obvious how you would determine this without using entropy

- But why Shannon entropy?

Additive Entropy

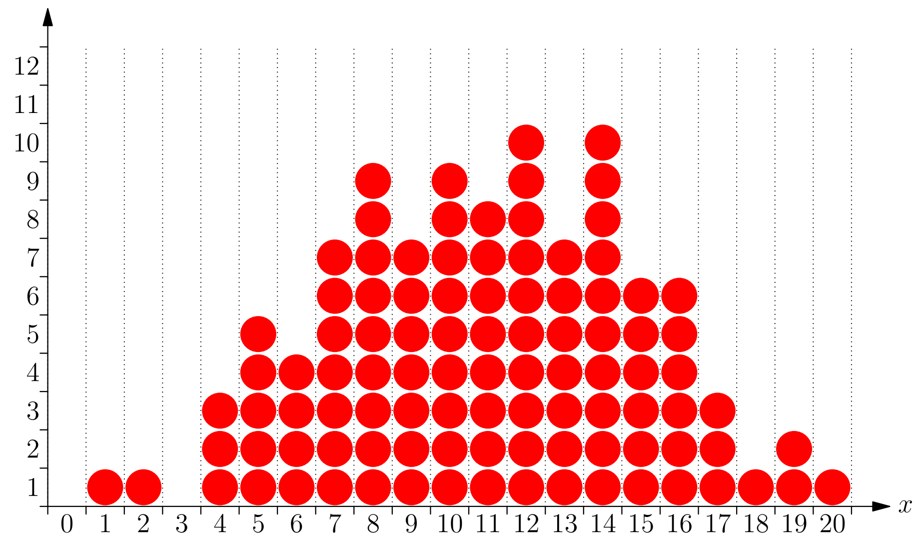
- If H_X and H_Y is the uncertainty of two independent random variable X and Y, what is the uncertainty of the combined event (X,Y)?

$$
\begin{aligned}
H_{(X,Y)} &= -\sum_{X,Y} \mathbb{P}(X,Y) \log_2(\mathbb{P}(X,Y)) \\
&= -\sum_{X,Y} \mathbb{P}(X)\mathbb{P}(Y) \log_2(\mathbb{P}(X)\mathbb{P}(Y)) \\
&= -\sum_{X,Y} \mathbb{P}(X)\mathbb{P}(Y) \left(\log_2(\mathbb{P}(X)) + \log_2(\mathbb{P}(Y))\right) \\
&= -\sum_{X} \mathbb{P}(X) \log_2(\mathbb{P}(X)) - \sum_{Y} \mathbb{P}(Y) \log_2(Y) = H_X + H_Y
\end{aligned}
$$

- Shannon's entropy is one of the few functions that satisfy this condition

Outline

1. Measuring Uncertainty
2. **Code Length**
3. Maximum Entropy



Why Measure Entropy in Bits

- Suppose we had to communicate a message with 2^n equally likely outputs (e.g. the result of n-coin tosses)
- We can do this with a binary string with n bits (011..0)
- If there were 5 possible outcomes I could do this with 3 bits, but waste 3/8 of the message
- However if we have a batch of 3 independent messages each with 5 outcomes then there are 125 possible outcomes. We could communicate this with 8 bits. This would waste 3/128 of the message
- By batching together enough messages with N outcomes then we asymptotically need just $\log_2(N)$ bits

Different Probabilities

- We "showed" that if we had N events, X_i, each with probability, $\mathbb{P}(X_i) = 1/N$, we can code the outcomes with a message of length $-\log_2(\mathbb{P}(X_i)) = \log_2(N)$
- With a shorter message we would not be able to distinguish all possible outcomes from the message
- What happens if some of outcomes occur with a different probability

X_i:	1	2	3	4	5	6
$p(X_i)$:	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{4}$
Code:	000	001	010	011	10	11
$L = -\log_2\big(p(X_i)\big)$:	3	3	3	3	2	2

Shannon's Entropy

- If the probabilities are not equal to 2^{-n} we can still find a code with a length very close to $-\log_2(\mathbb{P}(X))$ per message by transmitting a large number of messages
- The length of the message measures the amount of **surprise** on receiving the message
- Shannon's entropy is the expected length of the message to communicate a random variable X

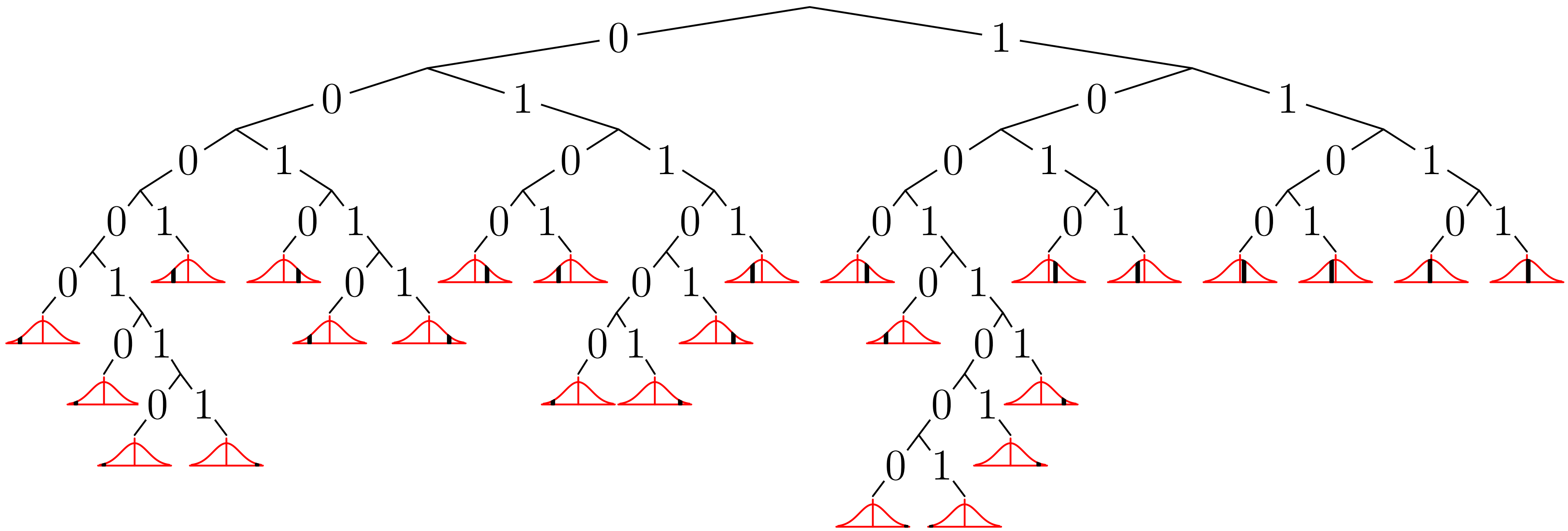
$$H_X = \mathbb{E}_X[-\log_2(\mathbb{P}(X))] = -\sum_{x\in\mathcal{X}} \mathbb{P}(X=x)\log_2(\mathbb{P}(X=x))$$

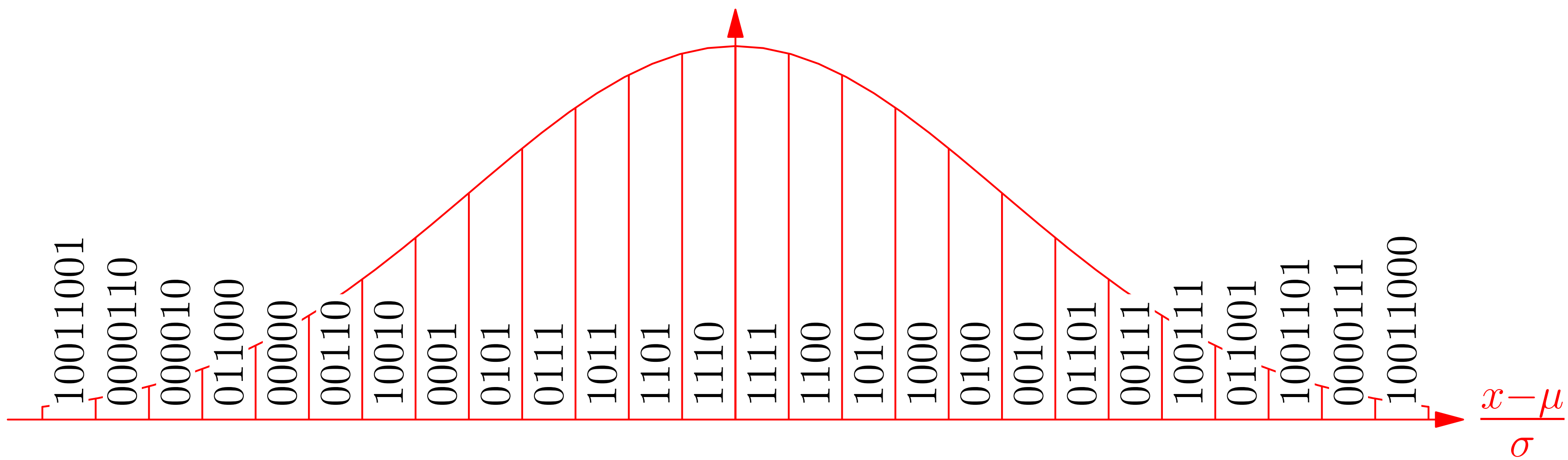
- The expected length is a measure of the uncertainty (how much information on average we need to convey the outcome)

Real Codes

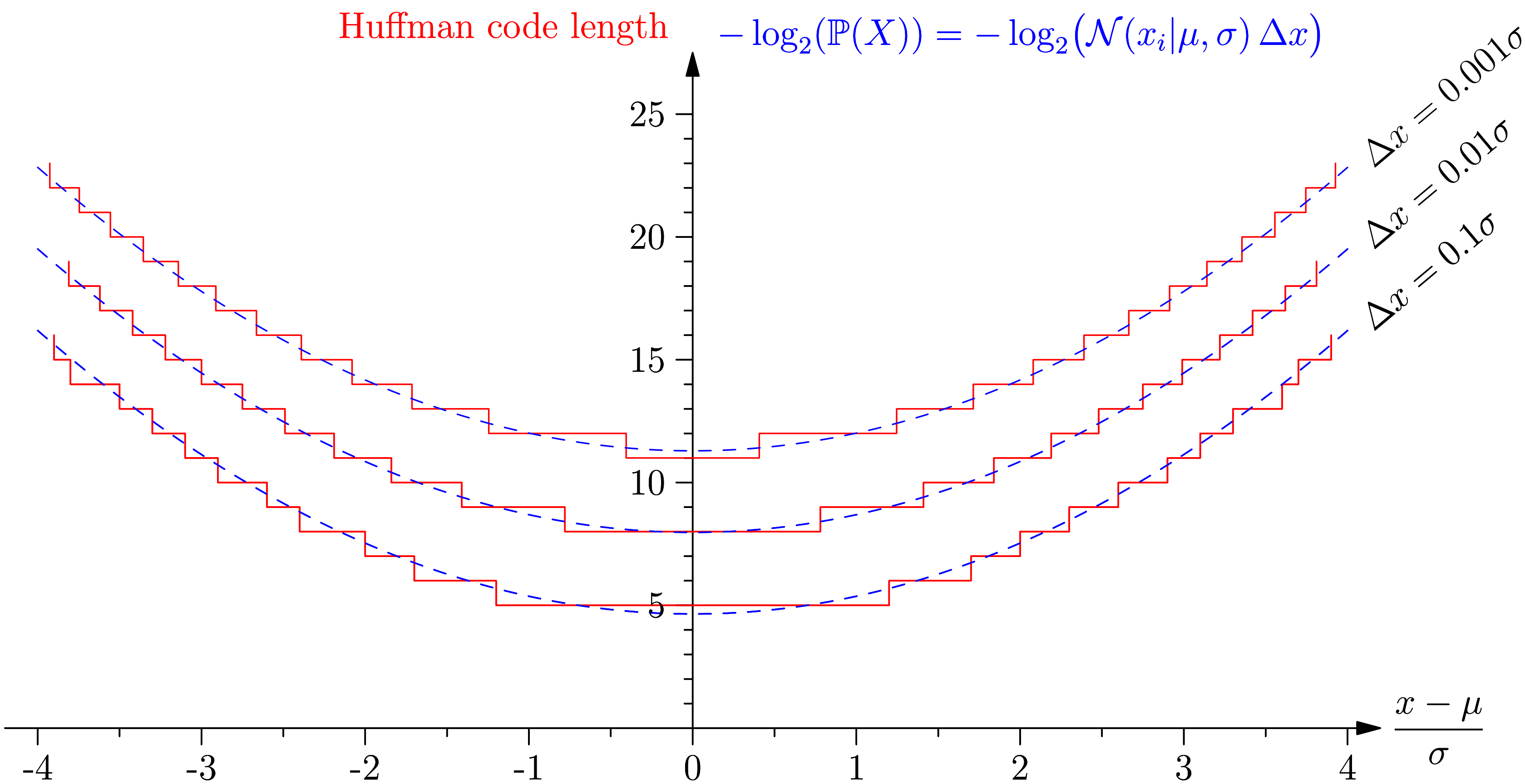
- Those of you with some computer science background will realise that we can't actually use different length strings in a code without paying some price
- We won't know where a code word ends so we can't decode the message
- An optimal solution is to use Huffman encoding where we associate the leaf of a tree with each code word
- Using the tree we can decode any message constructed using the tree
- There is a greedy algorithm for constructing the optimal tree

Coding Normals





Coding Normals to Accuracy Δx



bits and nats

- We have measured entropy in **bits** using

$$H_X = -\sum_{x \in \mathcal{X}} \mathbb{P}(X = x) \log_2\big(\mathbb{P}(X = x)\big)$$

- Sometimes it is easier to use natural logarithms

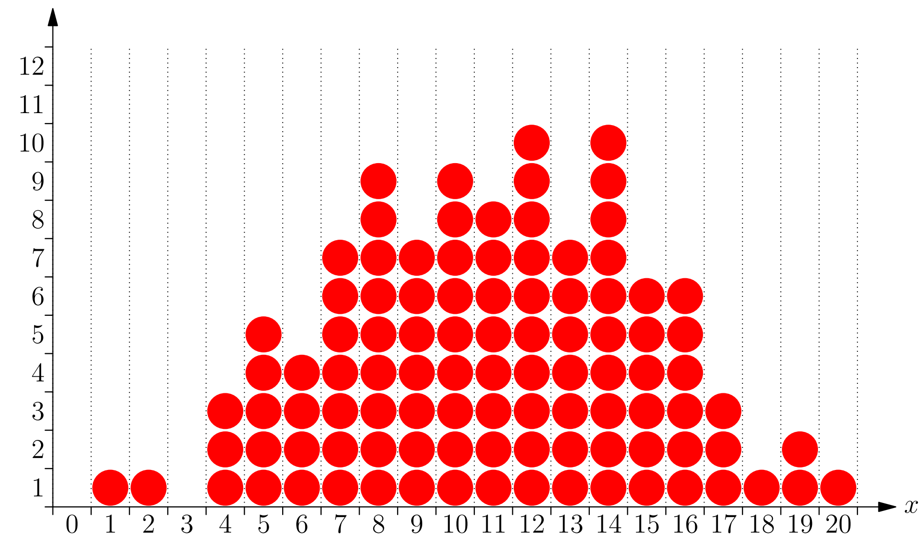
$$H_X = -\sum_{x \in \mathcal{X}} \mathbb{P}(X = x) \ln\big(\mathbb{P}(X = x)\big)$$

- In this case the entropy is measured in **nats** with 1 nat equal to $\log_2(\mathrm{e})$ bits

- This is often easier when we want to do calculus on entropy

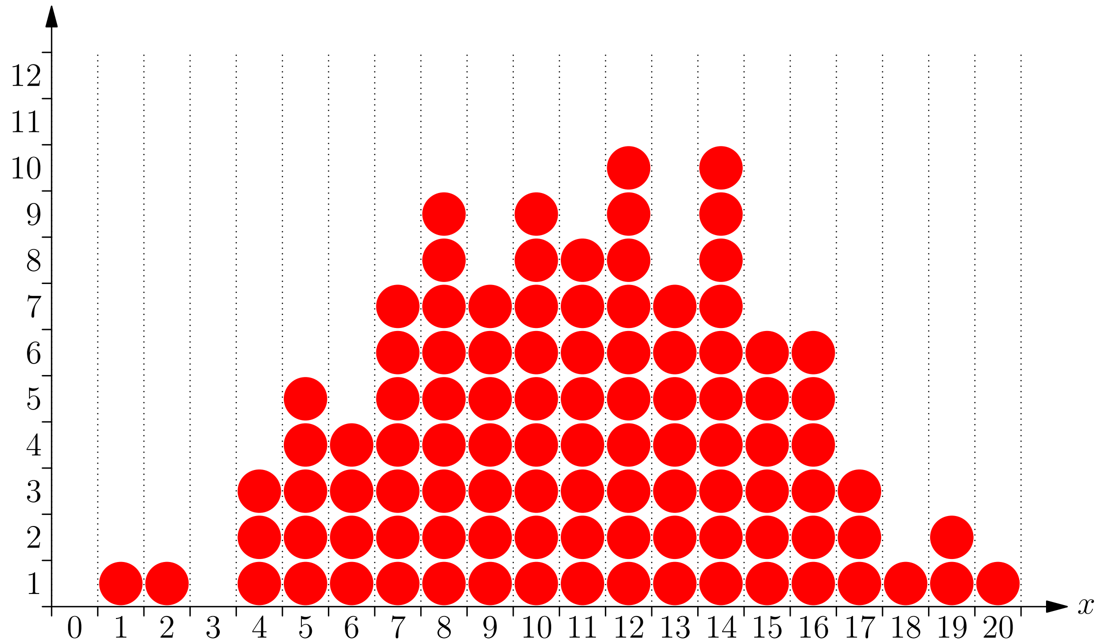
Outline

1. Measuring Uncertainty
2. Code Length
3. **Maximum Entropy**



Number of States

- Suppose I have N balls I them put in K boxes with coordinates x_i such that the mean is μ and variance is σ^2



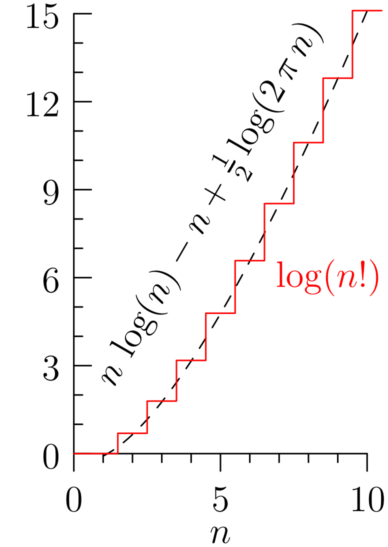
$$\mathbb{P}(\boldsymbol{n}) \propto \frac{N!}{n_1! n_2! \cdots n_K!} \left[\!\left[\sum_i \frac{n_i}{N} x_i = \mu \right]\!\right] \left[\!\left[\sum_i \frac{n_i}{N} (x_i - \mu)^2 = \sigma^2 \right]\!\right]$$

Stirling's Approximation

- We can approximate the factorial $n!$ using **Stirling's approximation**

$$n! \approx \sqrt{2\pi n}\, n^n \mathrm{e}^{-n}$$

$$\log(n!) = n \log(n) - n + \frac{1}{2}\log(2\pi n)$$



- Using this in our formula for $\mathbb{P}(\boldsymbol{n})$ we have

$$\mathbb{P}(\boldsymbol{n}) \approx C \mathrm{e}^{-N \sum_i \frac{n_i}{N} \log\left(\frac{n_i}{N}\right)} \prod_{l=1}^{3} \left[\!\left[\sum_i \frac{n_i}{N} f_l(x_i) = v_l \right]\!\right]$$

where $(f_1(x_i), v_l) = \{(1,1), (x_i, \mu), ((x_i - \mu)^2, \sigma^2)\}$

Number of States and Entropy

- Let $p(x_i) = n_i/N$ be the proportion of balls in bin i then

$$\mathbb{P}(\boldsymbol{n}) \approx C \mathrm{e}^{N H_X} \prod_{l=1}^{3} \left[\!\!\left[\sum_i \frac{n_i}{N} f_l(x_i) = v_l \right]\!\!\right]$$

  where

$$H_X = -\sum_i p(x_i) \log(p(x_i))$$

- That is, the "entropy" can be seen as a measure of the logarithm of the number of configurations

- When the number of balls, $N \to \infty$ the overwhelmingly likely configurations is the one that maximises the entropy subject to the observed mean and variance

Maximum Entropy Method

- When we are trying to infer a distribution given some observations then we can maximise the entropy subject to constraints—the entropy acts as a prior
- This is known as the **maximum entropy method**
- We can rationalise this as this is by far the most likely set of configurations consistent with the observations
- Alternatively we can see this as maximising our uncertainty given what we know—being as unbiased as possible
- It only gives a good approximation if all possibilities are equally likely

Knowing the Mean and Variance

- Consider a continuous random variable, X, with a known mean and second moment

$$\mathbb{E}[X] = \mu, \qquad \mathbb{E}\left[X^2\right] = \mu_2 = \mu^2 + \sigma^2$$

- To maximise the entropy subject to constraints consider

$$\begin{aligned}\mathcal{L}(f) = &- \int f_X(x) \log(f_X(x))\,\mathrm{d}x + \lambda_0 \left(\int f_X(x)\,\mathrm{d}x - 1 \right) \\ &+ \lambda_1 \left(\int f_X(x)\, x\,\mathrm{d}x - \mu \right) + \lambda_2 \left(\int f_X(x)\, x^2\,\mathrm{d}x - \mu_2 \right)\end{aligned}$$

- Thus

$$\frac{\delta \mathcal{L}(f)}{\delta f_X(x)} = -\log(f_X(x)) - 1 + \lambda_0 + \lambda_1 x + \lambda_2 x^2 = 0$$

- Or

$$f_X(x) = \mathrm{e}^{-1+\lambda_0+\lambda_1 x+\lambda_2 x^2}$$

Normal Distribution

- We have three constraints

$$\int \mathrm{e}^{-1+\lambda_0+\lambda_1 x+\lambda_2 x^2} \mathrm{d}x = 1$$

$$\int \mathrm{e}^{-1+\lambda_0+\lambda_1 x+\lambda_2 x^2} x \, \mathrm{d}x = \mu$$

$$\int \mathrm{e}^{-1+\lambda_0+\lambda_1 x+\lambda_2 x^2} x^2 \mathrm{d}x = \mu_2 = \mu^2 + \sigma^2$$

- Solving for λ_0, λ_1 and λ_2 then

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-(x-\mu)^2/(2\sigma^2)}$$

- That is, the normal distribution is the maximum entropy distribution given we known the mean and variance

Using Maximum Entropy

- Maximum entropy is often used to infer distributions
- It can be very effective, but it might not work well if there are other constraints that we have not included
- The place that they work superbly well is in statistical physics
- The whole of statistical physics is about inferring distributions making observations of volume, pressure, etc.
- Temperature appears rather strangely as a Lagrange multiplier

Historic Entropy

- Historically entropy was first introduced in statistical physics by Rudolf Clausius in 1865 (although Macquorn Rankine discussed it in 1850)
- Its interpretation as the number of states was introduced by Ludwig Boltzmann
- The person who got it all right was Josiah Willard Gibbs (and James Clerk Maxwell)
- Claude Shannon invented information theory base on entropy around 1948 (more on that in the next lecture)
- Ed Jaynes was the first to understand that statistical physics can be seen as an inference problem

Conclusion

- Entropy provides a measure of the disorder or uncertainty in a system
- It forms the basis of information theory which we will look at in the next lecture
- $-\log(\mathbb{P}(X = x))$ can be seen as the minimum length of a message to communication x
- This will be used as the basis of the minimum description length formalism also discussed in the next lecture
- Entropy can be used as a prior, which we often maximise subject to constraints to obtain an unbiased estimate