Information & Entropy[1]. Daniel Boley

1 Information

We have 2 kinds of vehicles, cars and trucks, and 2 countries of origin: foreign and domestic. Denote the basic probabalitities $p_c = pr(car)$, $p_{\bar{c}} = pr(truck)$, $p_f = pr(foreign)$, $p_{\bar{f}} = pr(domestic)$, the joint probabilities, $p_{fc} = pr(foreign\&car)$, etc. such that (for example) $p_{fc} + p_{\bar{f}c} = p_c$, etc., the conditional probabilities, $q = p_{f|c} = pr(foreign|car)$, $r = p_{f|\bar{c}} = pr(foreign|truck)$, etc. We have the relations $p_c + p_{\bar{c}} = 1$, $p_f + p_{\bar{f}} = 1$, $p_{f|c} + p_{\bar{f}|c} = 1$, etc., and $p_{fc} = p_{f|c}p_c = qp_c$, etc.

Let $I(p)$ be the information obtained when learning of an event which has probability p. For example, if we have an object that we know is a vehicle, and we discover that it is a car, then the information we obtain from this discovery is $I(p_c)$. We make the following assumptions:

1. $I(p) > 0$ for all p: $0 < p < 1$.
2. If $p = 1$, the event is absolutely certain, so $I(1) = 0$.
3. $I(p)$ is a smooth function of only the numerical probability p.
4. If two events occur in sequence, then the amount of information is additive. If the first event has probability p, and the second event has probability q assuming the first event has occured, then the total information is the sum of the information learned from the first event, plus the additional information learned from the second event once the first event is known.

The last assumption can be applied to the car example. Upon learning that a vehicle is a car, we obtain $I(p_c)$ amount of information. Subsequently we learn that the car is foreign, obtaining an additional $I(q)$ amount of the information. This should equal the total amount of information we would obtain if we learned all at once that the vehicle was a foreign car, namely $I(p_{fc}) = I(p_c p_{f|c}) = I(p_c q)$. This relation should hold regardless of the specific numerical values of p_c and q, but should depend only on the numerical values of p_c and q. So we have the relation

$$I(pq) = I(p) + I(q) \text{ for any probability values } p, q. \tag{1}$$

What kind of function obeys a relation such as (1)? The answer is a logarithmic function. Since the p's are less than 1, and $I(p) \geq 0$, it must be that $I(p) = -\log p$, where log is the usual logarithm in some arbitrary base. The proof that $I(p)$ must be a logarithm function depends on assuming (1) holds for any positive numbers, and then first plugging in integer values, then rationals, and then invoking continuity.

[1] 2019-10-8-130

Here are a sample of the steps in this proof. Assume (1) holds for all positive numbers. First $I(p \cdot 1) = I(p) + I(1) \forall p$ implies that $I(1) = 0$. Then (1) implies relations like $I(b^2) = I(b) + I(b)$, $I(b^k) = kI(b)$, $kI(b^l) = I(b^{kl}) = lI(b^k)$, which implies $I(b^{k/l}) = (k/l)I(b)$, for arbitrary positive integers k, l and any positive b. Then we invoke continuity to get $I(b^s) = sI(b)$, for any real $s \geq 0$. There is one degree of freedom left, namely the scaling of I, which is set by fixing the value of $I(0.5)$, or equivalently choosing the base of the logarithm. The usual custom is to use logarithms base 2, yielding $I(.5) = 1$.

2 Entropy

While sitting by the side of a freeway, you watch $n = 10$ zillion objects pass by during the day. You know that every object passing by is a vehicle, that np_c of those vehicles are cars, that np_{fc} vehicles are foreign cars, etc. We group all the vehicles other than cars in the complementary category $\bar{c}$ and call them trucks (even if they are really buses). What is the average amount of information you learn by recognizing whether each object going by is a car or a truck, is foreign or domestic? The key word here is "average". We need an expectation. If all we can distinguish is cars vs trucks, then the total amount of information we learn during the day is equal to the number of cars times the information learned per car plus the number of trucks times the amount of information learned per truck:

$$\text{total information} = np_c I(p_c) + np_{\bar{c}} I(p_{\bar{c}}).$$

Divide by n to obtain the average information learned per vehicle:

$$\text{average information} \equiv H_c = p_c I(p_c) + p_t I(p_{\bar{c}}) = -p_c \log p_c - p_{\bar{c}} \log p_{\bar{c}}. \tag{2}$$

This quantity is defined as the *entropy*. It is seen to be a statistical average over many instances. It is also symmetric: $H_c = H_{\bar{c}}$.

3 Mutual Information

We actually have three entropies in the current situation: (A) the entropy of cars versus trucks ignoring the country of origin, (B) the entropy of country of origin ignoring the type of vehicle, and (C) the entropy of the 4 subcategories, foreign cars, domestic cars, foreign trucks, domestic trucks. These entropies are, respectively,

$$\begin{array}{lrcl}
\text{cars vs trucks} & H_c & = & -p_c \log p_c - p_{\bar{c}} \log p_{\bar{c}} \\
\text{foreign vs domestic} & H_f & = & -p_f \log p_f - p_{\bar{f}} \log p_{\bar{f}} \\
\text{4 subcategories} & H_{fc} & = & -p_{fc} \log p_{fc} - p_{\bar{f}c} \log p_{\bar{f}c} - p_{f\bar{c}} \log p_{f\bar{c}} - p_{\bar{f}\bar{c}} \log p_{\bar{f}\bar{c}},
\end{array}$$

where H_{fc} is called the *joint* entropy.

While watching all the vehicle pass by, we learn H_c amount of information *on average* by distinguishing cars from trucks. We slowly learn to distinguish foreign from domestic, learning on the average $H_{fc} - H_c$ additional information. But this could be very little if, for example, almost all the cars were foreign and almost all the trucks were domestic. In this case, we could almost just guess the country of origin just knowing the type of vehicle. The quantity $H_{fc} - H_c$ is called the *conditional entropy* and has the following expression:

$$\begin{aligned}
H_{f|c} &\stackrel{\text{def}}{=} \text{conditional entropy} \stackrel{\text{def}}{=} H_{fc} - H_c \\
&= [-p_{fc}\log p_{fc} - p_{\bar{f}c}\log p_{\bar{f}c} - p_{f\bar{c}}\log p_{f\bar{c}} - p_{\bar{f}\bar{c}}\log p_{\bar{f}\bar{c}}] \\
&\quad - [-p_c \log p_c - p_{\bar{c}}\log p_{\bar{c}}] \\
&= [-p_{fc}\log p_{fc} - p_{\bar{f}c}\log p_{\bar{f}c} - p_{f\bar{c}}\log p_{f\bar{c}} - p_{\bar{f}\bar{c}}\log p_{\bar{f}\bar{c}}] \\
&\quad - [-p_{fc}\log p_c - p_{\bar{f}c}\log p_c - p_{f\bar{c}}\log p_{\bar{c}} - p_{\bar{f}\bar{c}}\log p_{\bar{c}}] \\
&= [-p_{fc}\log \frac{p_{fc}}{p_c} - p_{\bar{f}c}\log \frac{p_{\bar{f}c}}{p_c} - p_{f\bar{c}}\log \frac{p_{f\bar{c}}}{p_{\bar{c}}} - p_{\bar{f}\bar{c}}\log \frac{p_{\bar{f}\bar{c}}}{p_{\bar{c}}}] \\
&= p_c[-p_{f|c}\log p_{f|c} - p_{\bar{f}|c}\log p_{\bar{f}|c}] + p_{\bar{c}}[-p_{f|\bar{c}}\log p_{f|\bar{c}} - p_{\bar{f}|\bar{c}}\log p_{\bar{f}|\bar{c}}] \\
&= p_c H_{f/c} + p_{\bar{c}} H_{f/\bar{c}},
\end{aligned}$$

where $H_{f/c}$, $H_{f/\bar{c}}$ denote the *local entropies* of "foreign-ness" local to the class of cars and to the class of trucks, respectively. Please note that the conditional entropy, $H_{f|c}$, is the weighted sum of these local entropies, each weighted by the prior probability that one is in the class where that local entropy applies.

If the two attributes, car vs truck and foreign vs domestic, were independent, then we would expect that learning whether a vehicle is foreign would yield the same amount of information whether or not we knew that the vehicle was a car. That is, we would expect $H_{f|c} = H_f$. On the other hand, if we had the situation where almost all the cars were foreign and almost all the trucks domestic, then we would expect that $H_{f|c}$ would be very small, especially when compared to H_f. We can measure this with the difference $H_{f|c} - H_f$, which takes on the following expression:

$$\begin{aligned}
M_{fc} &\stackrel{\text{def}}{=} \text{mutual information} \stackrel{\text{def}}{=} H_f - H_{f|c} \\
&= [-p_f \log p_f - p_{\bar{f}}\log p_{\bar{f}}] \\
&\quad - [-p_{fc}\log \frac{p_{fc}}{p_c} - p_{\bar{f}c}\log \frac{p_{\bar{f}c}}{p_c} - p_{f\bar{c}}\log \frac{p_{f\bar{c}}}{p_{\bar{c}}} - p_{\bar{f}\bar{c}}\log \frac{p_{\bar{f}\bar{c}}}{p_{\bar{c}}}] \\
&= p_{fc}\log \frac{p_{fc}}{p_c p_f} + p_{\bar{f}c}\log \frac{p_{\bar{f}c}}{p_c p_{\bar{f}}} + p_{f\bar{c}}\log \frac{p_{f\bar{c}}}{p_{\bar{c}} p_f} + p_{\bar{f}\bar{c}}\log \frac{p_{\bar{f}\bar{c}}}{p_{\bar{c}} p_{\bar{f}}}.
\end{aligned}$$

This quantity is called the *mutual information* . Notice that it equals

$$M_{fc} = H_f + H_c - H_{fc}$$

and is symmetric in the probabilities of f, c. When f and c are independent, the terms under the log's are all equal to 1, hence $M_f = 0$. When f is completely determined by c and viceversa, then the conditional entropies take on the value $H_{c|f} = H_{f|c} = 0$ and the mutual information takes on the value $M_{fc} = H_f = H_c$. In general, the form of the expressions for the various quantities implies naturally that all of the quantities M_f, H_c, H_f, H_{fc}, $H_{f|c}$, $H_{c|f}$, are all non-negative, and the pure entropies for f and c are bounded by $-\log {}^1\!/_2$ (occuring when $p = {}^1\!/_2$). This implies that the these values must lie in the ranges (some of these inequalities result from the fact that we have binary categories):

$$\begin{array}{ccccc} 0 & < & H_c & < & \log 2 \\ 0 & < & H_f & < & \log 2 \\ 0 & < & H_{fc} & < & H_c + H_f \\ 0 & < & H_{f|c} & < & H_f \\ 0 & < & H_{c|f} & < & H_c \\ 0 & < & M_{fc} & < & \min\{H_c, H_f\}. \end{array}$$

4 Relative Entropy

Suppose we spent yesterday recording vehicles and using the numbers to estimate the various probabilities of cars vs trucks, foreign vs domestic. Today, we watch the vehicles go by. What is the change in the average information we get from classifying the individual vehicles, due to a change in their overall probabilities? If we see a vehicle and identify it as a car, then we believe we have gotten $I(p_c) = -\log p_c$ amount of information. We attach this amount of pseudo-information to each individual car. Likewise, we attach $I(p_{\bar{c}})$ amount of pseudo-information to each individual truck. However, today is Sunday and there are fewer trucks on the road. So the actual probability of a car is q_c. Therefore, after m vehicles have passed by, we have attached a total amount of pseudo-information of

$$m\mathcal{E}_q(-\log p_c) = -mq_c \log p_c - mq_{\bar{c}} \log p_{\bar{c}}, \tag{3}$$

that is, m times the average amount of pseudo-information attached per vehicle. Here $\mathcal{E}_q$ denotes the expectation under the probability distribution q. The difference between the pseudo-information attached under the naive assumption that the probability was p_c and the actual amount of information per vehicle under the new probability is

$$\begin{aligned} m\mathcal{E}_q(-\log p) - mH(q) &= mq_c \log q_c + mq_{\bar{c}} \log q_{\bar{c}} - mq_c \log p_c - mq_{\bar{c}} \log p_{\bar{c}} \\ &= mq_c \log \frac{q_c}{p_c} + mq_{\bar{c}} \log \frac{q_{\bar{c}}}{p_{\bar{c}}} \end{aligned} \tag{4}$$

The quantity $\mathrm{KL}(q||p) \overset{\text{def}}{=} \mathcal{E}_q(-\log p) - H(q)$ is called the *relative entropy* or the Kullback-Leibler "distance" between two probability distributions q and p. This quantity is zero exactly when $q = p$, and is positive when $q \neq p$, but it might not be finite, and it is not symmetric in q and p. To see this, let

$$f(p_c) \overset{\text{def}}{=} KL(q||p) = q_c \log q_c + (1 - q_c)\log(1 - q_c) - q_c \log p_c - (1 - q_c)\log(1 - p_c)$$

be defined by the formula (4). Then a short calculation yields the derivative:

$$f'(p_c) = \frac{\mathrm{d}}{\mathrm{d}p_c} f(p_c) = \frac{p_c - q_c}{p_c(1 - p_c)}.$$

It is easily seen that

$$\begin{array}{ll} f'(p_c) > 0 & \text{if } p_c > q_c \\ f'(p_c) = 0 & \text{if } p_c = q_c \\ f'(p_c) < 0 & \text{if } p_c < q_c \end{array}$$

so the global minimum on $[0, 1]$ is achieved at $p_c = q_c$. Furthermore, if $p_c = q_c$ then $f(p_c) = 0$, but if $p_c = 0$ or $p_c = 1$ (and $p_c \neq q_c$), then $f(p_c) = \infty$.

5 Twenty Questions

We are given N possible items x_k each with its own probabality p_k of occurence, for $k = 1, \dots, N$. We observe a sequence of such items. This could represent strings drawn from an alphabet of N characters. In the following we relate the entropy of the probability distribution $\{p_k\}_{k=1}^N$ to the minimum description length needed to encode this sequence when subjected to compression.

To identify each individual item in a sequence, we ask a series of yes/no questions. The result is a binary tree $\mathbf{T}$ of outcomes. The root corresponds to the first question, the left child corresponds to a NO answer, the right child corresponds to a YES answer. Each child node corresponds to another yes/no question which leads to the left if the answer is no and to the right if the answer is yes. So after two questions, we can end up at one of four nodes. Obtaining an answer to each question corresponds to descending one level in the tree. This continues until we reach a leaf of the tree, which corresponds to identifying the specific item. The tree has N leaves, each leaf corresponding to a specific item.

The cost associated with each item is the number of questions needed to identify it, i.e. the number of links that must be traversed within the tree from the root to the corresponding leaf. The expected total cost for all the items is the sum of all the costs, each weighted by the corresponding probablity:

$$E(\text{cost}(\mathbf{T})) = \sum_{k=1}^{N} p_k \text{cost}(x_k). \tag{5}$$

To prove some properties of this cost function, we need an intermediate result. Define the functional

$$q(x_k) = \left(1/2\right)^{\text{cost}(x_k)}$$

and define the corresponding functional over the entire tree to be the total of all the q-values summed over all the leaves: $q(\mathbf{T}) = \sum_1^N q(x_k)$. We will show that that $q(T) = 1$ is a constant for all binary trees, by induction.

A tree $\mathbf{T}_1$ consisting of just a single node has an q-value of $q(\mathbf{T}_1) = q((root)) = (1/2)^0 = 1$. Suppose $\mathbf{T}$'s root has a left subtree $\mathbf{L}$ and a right subtree $\mathbf{R}$. Then each path leading from $\mathbf{T}$'s root to a leaf of $\mathbf{L}$ is one more than the length of the path from $\mathbf{L}$'s root to that same leaf, so that we have one more power of $1/2$ in the q-value from $\mathbf{T}$'s root, compared to the q-value from $\mathbf{L}$'s root. The analogous property holds for the leaves of $\mathbf{R}$. Hence

$$q(\mathbf{T}) = \sum_{x_k \in L} q_{\mathbf{T}}(x_k) + \sum_{x_k \in R} q_{\mathbf{T}}(x_k) = 1/2 \sum_{x_k \in L} q_{\mathbf{L}}(x_k) + 1/2 \sum_{x_k \in R} q_{\mathbf{R}}(x_k) = 1/2(q(\mathbf{L}) + q(\mathbf{R})), \tag{6}$$

where we use the notation $x_k \in L$ to mean x_k is a leaf of tree $\mathbf{L}$, $q_{\mathbf{L}}(x_k)$ to denote the q-value of x_k as a leaf of the tree $\mathbf{L}$, etc. We can now apply the induction argument to fix the q-value for all binary trees. The property $q(\mathbf{T}) = 1$ holds for trees with just 1 level of nodes (i.e. a single node). If $q(\mathbf{T}) = 1$ holds for all trees up to l levels, then formula (6) implies it also holds for trees with $l + 1$ levels.

So, given a binary tree $\mathbf{T}$ with leaves x_k, $k = 1, \ldots, N$, we have established that corresponding q-values satisfy

$$q(x_k) > 0 \text{ and } \sum_k q(x_k) = 1. \tag{7}$$

We use the shorthand $q_k = q(x_k)$, so $\log_2 q_k$ is the length [cost] of the path from the root of the tree to the k-th leaf. The expected cost for all the items [leaves] can now be written

$$E(\text{cost}(\mathbf{T})) = -\sum_{k=1}^{N} p_k \log_2(q_k) \tag{8}$$

The probabilities p_k are given to us, but we have the choice to design the tree to minimize this average cost. That is, we can choose the q_k's to minimize the cost functional (8) subject to conditions (7). What is the optimal choice? In the previous section, we showed that $q_k = p_k$ is the optimal choice when $N = 2$. A simple (but cluttered) "calculus of variations" argument shows the same holds in the general case[2]. Of course, we are restricted to q-values such that $\log_2 q$ is an integer, because $-\log_2 q$ is a path length in the tree. But if N is very large, we can set the q-values to be very close to the theoretical optimum.

6 Relation to Binary decisions: Cars vs Trucks

For each vehicle, we get to ask one question: Car, yes or no? That would seem to require one question per vehicle. However, we can combine n consecutive vehicles into a single sequence $x = \{v_1, v_2, \ldots, v_n\}$, so we will have 2^n possible sequences, $x_1, \ldots, x_{2^n}$. If the probability of a car is p_c, then the probability of any given sequence of n vehicles of which i are cars is $p_c^i(1-p_c)^{n-i}$. Hence we can construct a binary tree to represent the collection of possible n-vehicle sequences $x_1, \ldots, x_{2^n}$, each with a probability p_k, and path length $-\log_2 q_k$, with $q_k \approx p_k$, $k = 1, \ldots, 2^n$. We can completely identify each n-vehicle sequence by the path from the root to the corresonding leaf, and each path is identified by a sequence of binary decisions (e.g. LRRL$\cdots$) corresponding to Yes-No-No-Yes answers, and these sequences of decisions can be encoded as bit strings: 1001$\cdots$. The length of the k-th bit string is $-\log_2 q_k$. Suppose we collect M sequences of n vehicles each, each sequence being encoded by the bitstring corresponding to its path through the tree, where M is a *very* large number. The total length of all the bitstrings concatenated together will be $-M\sum_k p_k \log_2 q_k \approx -M\sum_k p_k \log_2 p_k$. There are $\binom{n}{i}$ sequences with exactly i cars, each having a probability of $p_c^i(1-p_c)^{n-i}$, so

[2]Main idea: choose two q's such that $q_i < p_i$, $q_j > p_j$. Then $p_i \log_e(q_i + \Delta q) + p_j \log_e(q_j - \Delta q) \approx p_i \log_e q_1 + p_j \log_e q_j + \left(\frac{p_i}{q_i} - \frac{p_j}{q_j}\right)\Delta q$. But $\left(\frac{p_i}{q_i} - \frac{p_j}{q_j}\right) > 0$, hence this choice of q's cannot be the optimal choice.

the total length of the bitstrings will be[3]

$$-M\sum_{i=0}^{n}\binom{n}{i}p_c^i(1-p_c)^{n-i}\log_2 p_c^i(1-p_c)^{n-i} = -M[p_c n\log_2 p_c + (1-p_c)n\log_2(1-p_c)]. \quad (9)$$

Hence the average length per sequence will be $-[p_c n\log_2 p_c + (1-p_c)n\log_2(1-p_c)]$, and the average per vehicle will be $-[p_c\log_2 p_c + (1-p_c)\log_2(1-p_c)]$. Interpret (9) as follows. After M sequences (each consisting of n vehicles) have been recorded, we have seen an expected total of Mnp_c cars and $Mnp_{\bar{c}}$ trucks. We can allocate $\log_2 p_c$ bits for each car and $\log_2 p_{\bar{c}}$ bits for each truck. The total number of bits allocated over all M sequences of vehicles will exactly match (9) and also the intuition in section 1. But, to be precise, this is an average, since the amount of information attached to cars depends also on the frequency of trucks as well as cars.

7 Encoding Symbols that are Equally Likely

We consider encoding all sequences of n symbols, each taken from an alphabet A with a symbols, each equally likely. There are $N = a^n$ such sequences $S_0, \dots, S_{N-1}$, each sequence being equally likely. So we can model the average encoding length by simply adding up the lengths of all the codes for every sequence. Each sequence is encoded by a bit string, which if we ignore leading zeros (except for plain 0), corresponds to an natural number between 0 and some E_{N-1}. Sort the sequences by the numerical value of this encoding. Let E_k be the numerical value represented by encoding for the k-th sequence.

Example 1. Example using $a = 3, n = 4, N = 81$

oooooo0	AAAA
oooooo1	AAAB
ooooo10	AAAC
ooooo11	AABA
oooo100	AABB
⋮	⋮
oo11111	BABB
o100000	BABC
⋮	⋮
o111111	CBAA
1000000	CBAB
⋮	⋮
1010000	CCCC
bit encoding	original sequence

[3]Details of derivation: $\sum_{i=0}^{n}\binom{n}{i}p^i(1-p)^{n-i}\log_2 p^i(1-p)^{n-i} = \sum_{i=0}^{n}\binom{n}{i}p^i(1-p)^{n-i}i\log_2 p + \sum_{i=0}^{n}\binom{n}{i}p^i(1-p)^{n-i}(n-i)\log_2(1-p)$. Then $\sum_{i=0}^{n}\binom{n}{i}p^i(1-p)^{n-i}i\log_2 p = pn\log_2 p\sum_{i=1}^{n}\binom{n-1}{i-1}p^{i-1}(1-p)^{n-i}$ $= pn\log_2 p\cdot 1$.

Since all encodings are distinct, and in order $0 \leq E_0 < E_1 < \cdots$, we must have that $E_0 \geq 0$, $E_1 \geq 1$, $E_2 \geq 2$, …, $E_k \geq k$, etc. Let $|k|$ denote the number of bits needed to represent the natural number k. It is well known that $j < k$ implies $|j| \leq |k|$. So we have that the sum of all the lengths of all the encodings for all the sequences is at least the total lengths of all the bit representations of natural numbers up to N:

$$\begin{array}{rcl} \sum_{i=1}^{N-1} |E_N| & \geq & \sum_{i=0}^{N-1} |i| \\ & = & mN - 2^{m-1} - 2^{m-2} - \cdots - 2^2 - 2^1 \\ & = & mN - 2^m + 2 \end{array}$$

where 2^m is the smallest power of 2 greater than or equal to N. In the example above with $a = 3, n = 4, N = 81$, we have $m = 7$. We count all the bits listed in the first column, then subtract the leading zeroes. There are a total of $7 \cdot 81$ bits, 64 "leading zeroes" in the left-most position of each bit string, 32 "leading zeroes" in the second position of each bit string, etc. So the total number of bits, not counting leading zeroes is $7 \cdot 81 - 64 - 32 - 16 - 8 - 4 - 2 = 441$. If all the symbols were equally likely, so all sequences were equally likely, then this means that we would require on average $m - 2^m/N + 2/N > m - 2 = \lceil n \log_2 a \rceil - 2$ bits per sequence, and hence $O(\log_2 a)$ bits per symbol within each sequence.

If we were to pad every encoding to m bits, then we would require exactly m bits per sequence, which is within a constant of the minimum required using the variable length encoding just discussed. So one encoding that achieves the lower bound (within a constant) is to use the same number $O(\log_2 a)$ of bits for every symbol.

8 Encoding two symbols which are not equally likely

We again consider encoding all sequences of n symbols, each taken from an alphabet A with a symbols, each equally likely. There are $N = a^n$ such sequences $S_0, \ldots, S_{N-1}$, each sequence being equally likely. We have just shown that the most efficient encoding must take at least $O(\log_2 a)$ bits per symbol, and by assigning a consecutive natural number to each sequence, we can achieve this bound.

Choose the most efficient encoding for a sequence S of length N. Split the alphabet into two parts A_1 and A_2. Then the sequence S can be encoded by combining two encodings:

line 1.	s_1	s_2	s_3	s_4	$\cdots$	s_N	original sequence
line 2.	1	2	2	1	$\cdots$	?	from A_1 or A_2
line 3.	$\tilde{s}$	$\tilde{s}$	$\tilde{s}$	$\tilde{s}$	$\cdots$	?	encoding within A_1, A_2

The most efficient encoding in Line 3 would be to use $b_1 \approx \log |A_1| \overset{\text{def}}{=} \log a_1$ bits per symbol if the symbol were in A_1 and choose $b_2 \approx \log |A_2| \overset{\text{def}}{=} \log a_2$ bits if the symbol were in A_2, where $a_1 + a_2 = a$. The total number of bits used to encode the sequence in Line 3 would be $N\tilde{b} = Np_1 b_1 + Np_2 b_2$, where p_1, p_2 are the fraction of symbols from the respective subalphabet A_1, A_2, and $p_1 + p_2 = 1$. We have that

$$\tilde{b} = p_1 \log a_1 + p_2 \log a_2 = p_1 \log ap_1 + p_2 \log ap_2 = \log a + p_1 \log p_1 + p_2 \log p_2.$$

Since $\log a$ is the optimal number of bits needed to encode Line 1, we would need at least $-(p_1 \log p_1 + p_2 \log p_2)$ bits to encode Line 2, otherwise we would have found a more efficient way to encode Line 1. This is exactly the entropy (average information) associated with a 2-symbol alphabet with probability p_1, p_2=$1-p_1$. To complete the picture, we would have to produce an encoding that can achieve this limit. One such encoding is a Huffman code. Another naive approach is to use the same construction as in Example 1, but with the sequences sorted by probabilility of occurence, with the most likely occuring first (and hence getting the shortest encoding).

9 Approximation of factorial and binomial coefficient.

Let $I(n) = \int_1^n \ln t dt = n \ln n - n + 1$. Let $T(n) = \sum_1^n \ln k - {}^1\!/_2 \ln n$ be the trapezoidal rule approximation using a step size of 1. The two are related by $I(n) = T(n) + \mathrm{Err}(n)$ with $\mathrm{Err}(n) = \frac{1}{12} \sum_1^n \frac{d^2}{dx^2} \ln \xi_k$ where $k-1 \le \xi_k \le k$. Using $\frac{d^2}{dx^2} \ln x = -\frac{1}{x^2}$, we can bound the error by $|\mathrm{Err}(n)| < \frac{1}{12} \sum_1^\infty \frac{1}{k^2} = \frac{\pi^2}{72}$. Putting all this together, we arrive at the bounds for $\ln n!$:

$$n \ln n - n + {}^1\!/_2 \ln n + 1 - \frac{\pi^2}{72} \le \sum_1^n \ln k = \ln n! \le n \ln n - n + {}^1\!/_2 \ln n + 1.$$

Hence we have the expression for $n!$ within a close constant:

$$n! = \left(\frac{n}{e}\right)^n \sqrt{n} \cdot \gamma, \text{ where } 2.3700 \cdots = e^{1-\frac{\pi^2}{72}} \le \gamma \le e = 2.7183 \cdots.$$

We can then write the bounds for the binomial coefficient, where $0 \le x \le 1$, and using the notation $\bar{x} \stackrel{\text{def}}{=} 1-x$:

$$\binom{n}{nx} \stackrel{\text{def}}{=} \frac{n!}{(nx)!(n\bar{x})!} = \frac{\eta}{x^{nx}\bar{x}^{n\bar{x}}\sqrt{nx\bar{x}}},, \text{ where } .3207 \cdots = e^{1-\frac{\pi^2}{72}-2} \le \eta \le e^{1-2*(1-\frac{\pi^2}{72})} = .4839 \cdots$$

Taking natural logarithms, we can write this in terms of the entropy (based on the natural logarithm):

$$\ln \binom{n}{nx} = -\ln \sqrt{nx\bar{x}} + nH_e(x) + \ln \eta, \text{ where } -1.1370 \cdots \le \ln \eta \le -0.7258 \cdots.$$

or based on the logarithm base 2:

$$\log_2 \binom{n}{nx} = -\log_2 \sqrt{nx\bar{x}} + nH_2(x) + \log_2 \eta, \text{ where } -1.6403 \cdots \le \log_2 \eta \le -1.0471 \cdots.$$

We note that the Stirling Approximation to $n!$ is $n! \to n^n e^{-n} \sqrt{n}\sqrt{2\pi}$, as $n \to \infty$. Observing that $n!$ approaches this limit monoticically from above, we can tighten the bounds on γ, η, E:

$$\begin{array}{rcl}
2.5066\ldots = \sqrt{2\pi} & \leq \ \gamma \ \leq e & = \ 2.7183\ldots \\
0.9189\ldots = \ln\sqrt{2\pi} & \leq \ln\gamma \leq 1 & = \\
0.3392\ldots = \frac{\sqrt{2\pi}}{e^2} & \leq \ \eta \ \leq \frac{e}{2\pi} & = \ 0.4326\ldots \\
-1.0811\ldots = \ln\sqrt{2\pi} - 2 & \leq \ln\eta \leq 1 - \ln(2\pi) & = -0.8378\ldots \\
-1.5596\ldots = \ln\sqrt{2\pi} - 2 & \leq \ln\eta \leq 1 - \ln(2\pi) & = -1.2087\ldots
\end{array}$$

If x is bounded away from 0 and 1 (i.e., there is a constant c such that $0 < c < x < 1 - c$), then Stirling's approximation implies $\eta \to 1/\sqrt{2\pi} = .3989\ldots$ as $n \to \infty$.

10 Typical set

Let S_n denote a string of 0's and 1's of length n. The individual entries are independent, and the probability of a 1 is p. We define $\mathrm{Sum}(S_n)$ denote the sum of all the entries (or equivalently, the number of 1's in the string), and $\mathrm{Aver}(S_n) = \frac{1}{n}\mathrm{Sum}(S_n)$ be the "average" entry. Then

$$\Pr\{\mathrm{Aver}(S_n) = x\} = \Pr\{\mathrm{Sum}(S_n) = nx\} = \binom{n}{nx} p^{nx}\bar{p}^{n\bar{x}} = \frac{\eta}{\sqrt{nx\bar{x}}}\left(\frac{p}{x}\right)^{nx}\left(\frac{\bar{p}}{\bar{x}}\right)^{n\bar{x}}. \qquad (10)$$

As a side remark, we note that the log of this quantity can be written in terms of the relative entropy:

$$\log\Pr\{\mathrm{Aver}(S_n) = x\} = n\mathrm{KL}(x||p) - {}^1\!/_2\log(nx\bar{x}) + \log\eta.$$

This can be interpreted as follows: if the probability of a 1 bit were x, then the most common value for $\mathrm{Aver}(S_n)$ would be x. But the actual probability of a 1 bit is p, and the KL quantity measures the discrepancy between the two probability distributions on the bit strings induced by the probability for each individual bit.

Returning to the discussion in this section, the most common value for $\mathrm{Aver}(S_n)$ is p, and this is also the expected value for $\mathrm{Aver}(S_n)$. The probability of this value occuring is

$$\Pr\{\mathrm{Aver}(S_n) = p\} = \Pr\{\mathrm{Sum}(S_n) = np\} = \binom{n}{np} p^{nx}\bar{p}^{n\bar{x}} = \frac{\eta}{\sqrt{np\bar{p}}}. \qquad (11)$$

The number of strings with exactly np 1's (assuming np is an integer) is[4]

$$\#\{S_n : \mathrm{Aver}(S_n) = p\} = \#\{S_n : \mathrm{Sum}(S_n) = np\} = \binom{n}{np} = \eta\frac{1}{p^{np}\bar{p}^{n\bar{p}}\sqrt{np\bar{p}}} = \eta\frac{2^{nH(p)}}{\sqrt{np\bar{p}}} \qquad (12)$$

As $n \to \infty$, the probability mass for this set of strings shrinks, yet there are on the order of $2^{nH(p)}$ such strings. So we would need at least nH bits to encode them, and we would still miss a lot of strings.

[4]From here on, all the logs and H's use base 2 logarithms.

In contrast, we consider the set $\mathcal{S}_\delta$ all strings whose average value is within δ of the expected value p, where δ is a constant or a slowly shrinking function of n. The number of such strings is approximately

$$\begin{aligned} \#\{S_n : \mathrm{Aver}(S_n) \in [p-\delta, p+\delta]\} &= \#\{S_n : \mathrm{Sum}(S_n) \in [np-n\delta, np+n\delta]\} \\ &\approx 2n\delta \cdot \#\{S_n : \mathrm{Aver}(S_n) = p\} \\ &= \eta \cdot 2n\delta \cdot \frac{1}{p^{np}\bar{p}^{n\bar{p}}\sqrt{np\bar{p}}} = \eta \cdot 2n\delta \cdot \frac{2^{nH(p)}}{\sqrt{np\bar{p}}}. \end{aligned} \tag{13}$$

If δ is constant or a shrinking function of n, we would need on the order of $nH(p) + \Box \log n\delta$ bits to encode all these strings, taking an average of $H(p) + \epsilon$ bits per entry in the string, where $\Box, \epsilon$ are small quantities depending on p. As n grows, $\epsilon = O((\log n\delta)/n)$ shrinks toward zero, and also the probability distribution for $\Pr\{\mathrm{Aver}(S_n) = x\}$, converted to a density in $x \in [0,1]$ approaches a normal distribution with mean p and standard deviation $\sqrt{p\bar{p}/n}$. So if δ were set to a constant times the standard deviation, then probability mass of $\mathcal{S}_\delta$ would stay constant as n grows. But if δ were held constant (or shrinks at a slower rate than $1/\sqrt{n}$), then the probability mass for $\mathcal{S}_\delta$ would approach 1 as n grows. So, using just a little more than nH bits per string entry would suffice to encode almost all the strings, missing only the strings that will almost never occur. The number of extra bits becomes arbitrarily small as n grows.

To summarize this section, the set described in equation (13) is called a *typical set* modulo δ. For a fixed δ, the probability mass of this set approaches 1, while the number of entries in this set grows as $O(2^{nH(p)}\delta\sqrt{n})$. The fraction of the total number of bit strings of length n that lie in the typical set *shrinks* as $O(2^{n(H(p)-1)}\delta\sqrt{n})$, where $-1 \le H(p) - 1 \le 0$. Hence, while the size of the typical set grows as n grows, the size of the untypical set (consisting of all the bit strings outside the typical set) grows even faster, even though the total probability mass of the untypical set shrinks to zero. This statement is true even for shrinking δ such that $\delta > O(1/\sqrt{n})$.

We end this with a remark that we can encode *all* strings of length n using only $\log p + \epsilon$ bits per entry on average, One such encoding is the ad hoc encoding described in sec. 7, but a more standard encoding accomplishing the same purpose is the Huffman encoding. Here ϵ is an extra amount to account for the fact that we need an integer number of bits to encode each string S_n. The size of this extra amount, when amortized over each entry in the string, becomes arbitrarily small as n grows. Hence $\log p + \epsilon$ bits per entry is not only necessary, but also sufficient.

11 Comments to be Filled In

```
relative entropy (KL distance).  A distribution p requires a code of aver
len H(p).  If we use a code based on distribution q instead, then the
average of a code word is H(p)+ KL(p,q) = H(p)+ sum p(x) log (p(x)/q(x)).
```

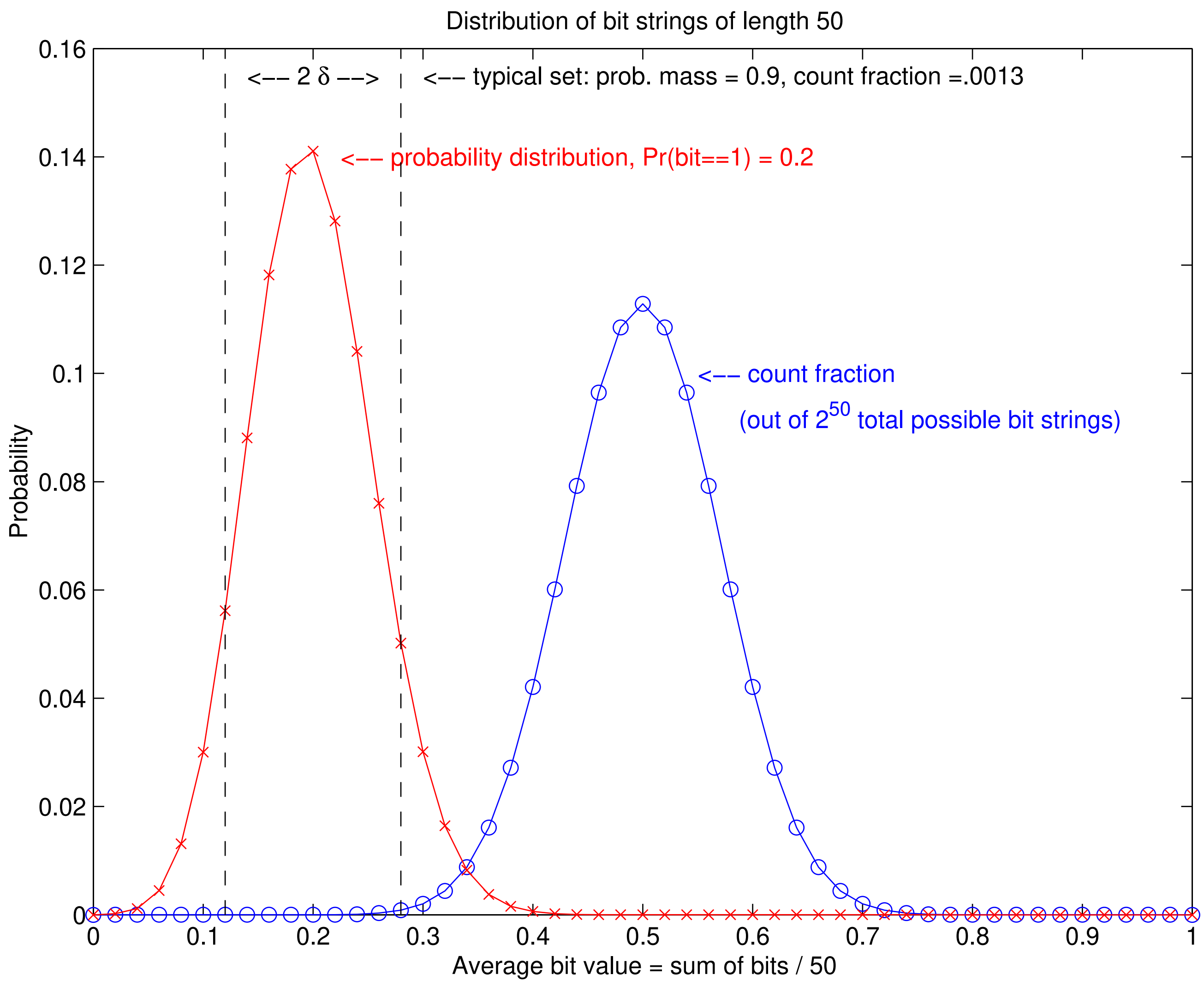

Figure 1: Illustration of typical set. For bit strings of length 50, with a probability of 0.2 that a bit equals 1, there are 2^{50} possible bit strings. Only $.0013 = 0.13\%$ of those bit strings have a sum between 6 and 14, corresponding to an average bit value between 0.12 and 0.28. But the probability of a bit string average being in that range is approximately $0.9 = 90\%$. For bigger n, both bell curves become narrower (as $1/\sqrt{n}$), hence the set of strings whose average is in $[.12, .28]$ would become a much smaller fraction of the total number of bit strings, but the probability mass of this set would increase.

```
jensen's inequality: E f(X) >= f(E X) if f is convex (convex <-> f' >=0).
if f strictly convex and E f(X) = f(E X) , then x is a constant.

markov property: I(X1;X2) >= I(X1;X3). -> given two different starting
distributions, relative entropy between distributions at time t
given by same markov chain decreases as time progresses.

fano's ineq: \hat X = g(Y) is an estimator of X. P_e = Pr{\hat X \neq X}, then
   H(P_e)+P_e \log ( |{\cal X}| - 1 ) \ge H(X|Y)


gamble = expected doubling rate of wealth depending on how a gambler
distrbibutes  his $1 of wealth among the horses.
Two horses, winning prob p1 p2.  gambler bets 2^s split b1 b2 (fractions)
among horses. Then expected value of the log of the winnings after one round
is
p1 log s b1 o1+ p2 log s b2 o2 =
    log s + p1 log b1 + p2 log b2
          + p1 log o1 + p2 log o2.
Here o1, o2 are the payoff amounts on a $1 bet.  Using log base 2, this
gives the doubling rate.  Doubling rate is maximized if b1=p1, b2=p2.
```

Contents