Information theory and decision tree

Jianxin Wu

LAMDA Group
National Key Lab for Novel Software Technology
Nanjing University, China
wujx2001@gmail.com

March 2, 2020

Contents

In this chapter, we introduce two topics: a few basic concepts and results from information theory and a very simple decision tree model. We have put these two things together because the specific decision tree model we introduce here is based on entropy, the core concept in information theory. We will also mention in passing applications of information theory in pattern recognition and machine learning—e.g., in feature selection and neural network learning.

1 Prefix code and Huffman tree

We will use the Huffman code as an example to motivate information theory. Suppose we want to count the number of different trees in a garden. You are given a hand held smart device, which can automatically transmit the GPS coordinates (i.e., the location) to a server. The device automatically recognizes the species of the tree in front of it using some pattern recognition techniques, and also transmits this to the same server. There are only five different types of trees in the garden, denoted by the symbols `a`, `b`, `c`, `d`, and `e`, respectively. 50% (or, $\frac{1}{2}$) of the trees in the garden are of type `a`, and the other four types each represent 12.5% (or, $\frac{1}{8}$) of the trees.

A simple idea is to represent these symbols as five binary codes:

$$\mathtt{000}, \mathtt{001}, \mathtt{010}, \mathtt{011}, \mathtt{100}\,.$$

Hence, we need to transmit three bits for each tree's identity. For some reason, the hand held device has extremely limited transmission bandwidth, and we want to use the *minimum number of bits* to code these tree types.

The Huffman code is such a code. It uses different numbers of bits to code the symbols:

$$\mathtt{a}\!:\!\mathtt{0}, \mathtt{b}\!:\!\mathtt{100}, \mathtt{c}\!:\!\mathtt{101}, \mathtt{d}\!:\!\mathtt{110}, \mathtt{e}\!:\!\mathtt{111}\,.$$

Hence, the average number of bits per tree is

$$\frac{1}{2} \times 1 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = 2\,,$$

which is smaller than 3. If there are 1000 trees in the garden, 1000 bits of bandwidth has been saved by using the Huffman code.

The Huffman code is a *prefix* code, meaning that the code for any symbol will not be a prefix of any other code. For example, {`0,100,010`} is not a prefix code because `0` is a prefix of `010`. The five symbols in our garden example also form a prefix code.

For a prefix code, we do not need any marker to separate two symbols. For example, `11111001000101` is decoded as `111/110/0/100/0/101`, or `edabac`, without using additional information.

The Huffman code is an optimal prefix code, and it is built using the Huffman tree. Given a set of m discrete symbols $s_1, s_2, \ldots, s_m$, we assume each symbol s_i is associated with an occurrance probability a_i ($a_i \geq 0$ and $\sum_{i=1}^{m} a_i = 1$). The Huffman tree is built using the following steps.

- Build a priority queue with n nodes, each node s_i has a weight a_i.

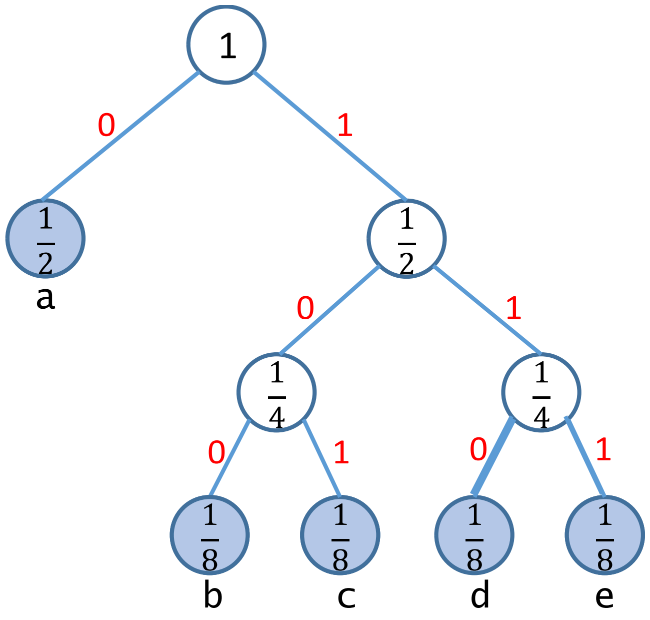


Figure 1: An example Huffman tree. Filled nodes are the symbols, and the others are internal nodes.

- Remove the two nodes with the smallest weights from the priority queue.
- Create a new node, whose weight is the sum of the weights of the two removed nodes, and place these two nodes as its children. Put the new node into the priority queue.
- Repeat the previous two steps until the priority queue is empty.

The Huffman tree is a binary tree. Figure 1 is the Huffman tree for our garden example. We label the edge connecting a node to its left and right children with a `0` and `1`, respectively. The Huffman code for a symbol is the concatenation of the labels of all edges in the path from the root to that symbol.

From the tree building process, we observe symbols appearing frequently (e.g., `a`) have short codes, and rare symbols have longer codes.

2 Basics of information theory

The average length of the binary string (or average number of bits) per symbol is closely related to the uncertainty in the symbols. To transmit a symbol over a communication channel, the binary codes must fully specify or describe the symbol. Let X be a discrete random variable, and $p(X = s_i) = a_i$ $(1 \le i \le m)$ be the probability mass function. Now let us consider an extreme case, in which

$a_1 = 1$ and $a_i = 0$ for $i > 1$ (i.e., only s_1 will occur). There is no uncertainty at all in this case. In order to transmit n symbols, we just need to transmit the integer n once, and the receiver understands this means a sequence with n symbols (which are all s_1). In other words, the number of bits per symbol is $\frac{1}{n}$, or 0 when $n \to \infty$. The least uncertainty leads to the shortest possible codes.

In another extreme, we consider the most uncertain case. Intuitively, the uniform distribution (i.e., $a_i = \frac{1}{m}$) is the most uncertain case, in which we cannot favor any particular symbol. It is obvious that the Huffman code requires $\lceil \log_2 m \rceil$ bits for every symbol, in which $\lceil \cdot \rceil$ is the ceiling function. Because for any p.m.f., we can encode all symbols by using $\lceil \log_2 m \rceil$ bits, the most uncertain p.m.f. leads to the longest possible codes.

Entropy is the core concept in *information theory*.[1] Entropy is a measure of uncertainty (or unpredictability) of information content, and technically, it provides a theoretically unbreakable limit on the shortest possible binary codes, if the symbols to be transmitted are not approximated (i.e., it is a lossless code) and is not compressed. Although this book is not on computer communication, Shannon entropy is now widely used in machine learning and pattern recognition because it can measure the *information content* of a distribution, which is very useful in our subject.

2.1 Entropy and uncertainty

Formally, the entropy of a discrete random variable X with p.m.f. $p_i = p(X = s_i) = a_i$ $(1 \le i \le m)$ is

$$H(X) = -\sum_{i=1}^{m} p_i \log_2 p_i \,, \tag{1}$$

and the unit for entropy is the *bit*. It is obvious that $H(X) = 0$ when $p_i = 1$ for some i, and $H(X) = \log_2 m$ when X is a discrete uniform distribution, which meets our expectations that $H(X)$ is the shortest possible average (or expected) code length.[2] We can also write the entropy as an expectation:

$$H(X) = -\mathbb{E}_X[\log_2 p(X)] \,, \tag{2}$$

where p is the probability mass function of X.

2.2 Joint and conditional entropy

Given two discrete random variables X and Y, their joint entropy is

$$H(X,Y) = -\sum_x \sum_y p(x,y) \log_2 p(x,y) = -\mathbb{E}_{(X,Y)}[\log_2 p(X,Y)] \,, \tag{3}$$

[1]Entropy also refers to an important concept in thermodynamics. We can say Shannon entropy to specifically refer to entropy in information theory. Claude Elwood Shannon is an American mathematician, electrical engineer, and cryptographer, in particular, "the father of information theory."

[2]Note that the length must be an integer in real-world communication channels, hence $\lceil \log_2 m \rceil$. Other bases (such as e and 10) are also used in entropy calculation.

in which we have omitted the domain for $X(x)$ and $Y(y)$ and the subscript of the p.m.f. (which is clear from the variables). The joint entropy can be extended to more than two variables.

When $X = x$, the notation $Y|X = x$ denotes another random variable (Y conditioned on a specific value of $X = x$), and it has an entropy

$$H(Y|X = x) = -\sum_{y} p_{Y|X=x}(y|x) \log_2 p_{Y|X=x}(y|x)\,, \tag{4}$$

or $H(Y|X = x) = -\sum_y p(y|x) \log_2 p(y|x)$. The conditional entropy $H(Y|X)$ is defined as

$$H(Y|X) = \sum_{x} p(x) H(Y|X = x) = -\sum_{x,y} p(x, y) \log_2 p(y|x)\,, \tag{5}$$

which is the weighted average of $H(Y|X = x)$ for all x. The conditional entropy can also be written as

$$-\mathbb{E}_{(X,Y)}[\log_2 p(Y|X)]$$

because of the last equality.

With some simple manipulations, we have

$$H(X, Y) = H(X) + H(Y|X)\,. \tag{6}$$

That is, the conditional entropy $H(Y|X)$ is the difference between the information contained in the joint random vector (X, Y) and the random variable X. We can roughly interpret this equality as: the information content in (X, Y) is the sum of the information in X and the part of Y that is not dependent on X (i.e., the information in Y with the effect of X removed).

2.3 Mutual information and relative entropy

The conditional distribution is not symmetric; hence in general

$$H(Y|X) \neq H(X|Y)\,.$$

However, since $H(X, Y) = H(X)+H(Y|X) = H(Y)+H(X|Y)$, we always have

$$H(X) - H(X|Y) = H(Y) - H(Y|X)\,. \tag{7}$$

We can roughly interpret this equality as: the difference in the amount of information between X and $X|Y$ (distribution of X if we know Y) equals the difference in the amount of information between Y and $Y|X$ (distribution of Y if we know X). Thus, this difference can be naturally treated as the information content that is common to both X and Y. The *mutual information* between X and Y is defined as

$$I(X; Y) = \sum_{x,y} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)} \tag{8}$$

$$= \mathbb{E}_{(X,Y)}\left[\log_2 \frac{p(X,Y)}{p(X)p(Y)}\right] \tag{9}$$

$$= I(Y;X)\,. \tag{10}$$

Please note there is *not* a negative sign before the expectation. Mutual information is symmetric, and it is easy to verify that

$$I(X;Y) = H(X) - H(X|Y) \tag{11}$$

$$= H(Y) - H(Y|X) \tag{12}$$

$$= H(X) + H(Y) - H(X,Y)\,. \tag{13}$$

The equality

$$I(X;X) = H(X)$$

holds, because $I(X;X) = H(X) - H(X|X)$ and $H(X|X) = 0$. Hence, $H(X)$ is the mutual information between X and itself, and sometimes the entropy $H(X)$ is called the *self-information*.

The definition of mutual information shows it is measuring the difference in the amount of information in the joint $p(X,Y)$ and the two marginals $p(X)$ and $p(Y)$. The difference (or "distance") between two distributions is measured by the Kullback–Leibler divergence (or KL distance). For two p.m.f.s $p(x)$ and $q(x)$ (defined on the same domain), the KL divergence is

$$\mathrm{KL}(p\|q) = \sum_x p(x)\log_2 \frac{p(x)}{q(x)}\,. \tag{14}$$

Note that there is not a negative sign before the summation. Hence,

$$I(X;Y) = \mathrm{KL}\left(p(x,y)\|p(x)p(y)\right)\,. \tag{15}$$

The KL divergence is also called the *relative entropy*. When $p(x) = 0$ or $q(x) = 0$ for some x, we assume $0\log_2 \frac{0}{q(x)} = 0$, $p(x)\log_2 \frac{p(x)}{0} = \infty$, and $0\log_2 \frac{0}{0} = 0$.

The KL "distance" is not symmetric. It does not satisfy the triangle inequality either. Hence, it is *not* a distance metric. However, it is indeed non-negative, and $\mathrm{KL}(p(x)\|q(x)) = 0$ implies $p(x) = q(x)$ for any x.

2.4 Some inequalities

The non-negativity of the KL distance is proved using Jensen's inequality. We have

$$-\mathrm{KL}(p\|q) = \sum_{x:p(x)>0} p(x)\log_2 \frac{q(x)}{p(x)} \le \log_2\left(\sum_{x:p(x)>0} p(x)\frac{q(x)}{p(x)}\right)$$

because $\log_2(x)$ is a concave function and $\sum_x p(x) = 1$, $p(x) \ge 0$. Then,

$$-\mathrm{KL}(p\|q) \le \log_2 \sum_{x:p(x)>0} q(x) \le \log_2 \sum_x q(x) = \log_2 1 = 0\,.$$

Hence, the KL distance is always non-negative.

The KL distance reaches 0 if and only if a) for any x with $p(x) > 0$, $\frac{p(x)}{q(x)} = c$ is a constant (equality condition of Jensen's inequality), and b) $\sum_{x:p(x)>0} q(x) = \sum_x q(x) = 1$. The condition b) means that $q(x) = 0$ when $p(x) = 0$. Hence, $p(x) = cq(x)$ holds for any x. Because $\sum_x p(x) = \sum_x q(x) = 1$, we have $c = 1$—that is, *the KL distance is 0 if and only if for any x, $p(x) = q(x)$.*

The non-negativity of the KL distance has many implications and corollaries:

- The mutual information $I(X;Y)$ is non-negative. It is 0 if and only if X and Y are independent.
- Let U be the discrete uniform distribution with m events—i.e., $p_U(u) = \frac{1}{m}$ for any u. Then, $\mathrm{KL}(X\|U) = \sum_x p(x) \log_2 \frac{p(x)}{1/m} = \log_2 m - H(X)$. Hence, for any X, we have

$$H(X) = \log_2 m - \mathrm{KL}(X\|U)\,, \tag{16}$$

$$0 \le H(X) \le \log_2 m\,. \tag{17}$$

  That is, $\log_2 m$ is indeed the upper bound of the uncertainty, and $\lceil \log_2 m \rceil$ the longest average length of binary codes. The equality holds only if the distribution is a uniform one.
- $H(X) \ge H(X|Y)$. In other words, knowing additional information (Y) to a random variable (X) will not increase our uncertainty. The proof is simple: $H(X) - H(X|Y) = I(X;Y) \ge 0$. Hence, the equality holds if and only if X and Y are independent. That is, knowing Y will reduce the uncertainty about X, unless they are independent.
- $H(X) + H(Y) \ge H(X,Y)$, and the equality holds if and only if they are independent. The proof is trivial, because $I(X;Y) = H(X) + H(Y) - H(X,Y) \ge 0$.

Figure 2 summarizes these relationships. The small red circle is $H(X)$, which is decomposed into two parts: $I(X;Y)$ which is the common information between X and Y (the purple region), and $H(X|Y)$ which is the part of X's information independent of Y. The large blue circle is $H(Y)$, composed of two parts $I(X;Y)$ and $H(Y|X)$. The union of the two circles is the entropy of the joint $H(X,Y)$.

The purple area is non-empty if X and Y are dependent—that is, $I(X;Y) > 0$. If X and Y are independent, the two circles do not overlap with each other. In this figure, the region corresponding to $H(X|Y)$ is a part inside that corresponding to $H(X)$, hence $H(X|Y) \le H(X)$. Similarly, $H(Y|X) \le H(Y)$.

The purple area also indicates $I(X;Y) = I(Y;X)$. However, the two circles are of different sizes, indicating that $H(X|Y) \ne H(Y|X)$ in general.

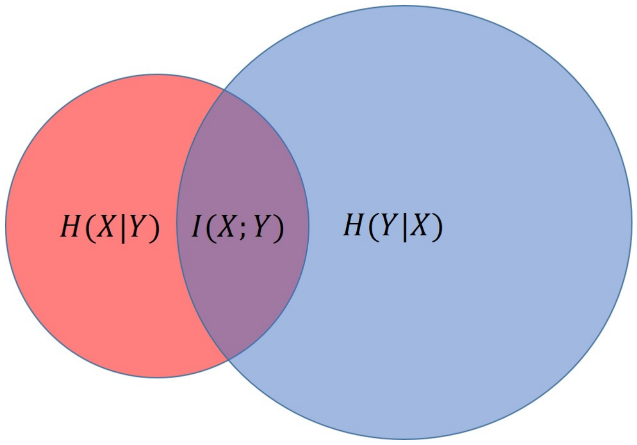


Figure 2: Relationships between entropy, conditional entropy, and mutual information. ()

2.5 Entropy of discrete distributions

To end this section, we introduce the entropy of some commonly used discrete distributions.

The entropy of an m-event discrete uniform random variable is $\log_2 m$.

The entropy of a Bernoulli distribution with success rate $0 < p < 1$ ($p_1 = p$ and $p_2 = 1 - p$) is $-p \log_2 p - (1-p)\log_2(1-p)$.

The geometric distribution with success rate $0 < p \leq 1$ has a p.m.f. $p_i = (1-p)^{i-1}p$, and its entropy is

$$\frac{-(1-p)\log_2(1-p) - p\log_2 p}{p}\,,$$

which is $\frac{1}{p}$ times the entropy of a Bernoulli distribution with the same parameter.

3 Information theory for continuous distributions

Entropy can also be computed for continuous random variables, by replacing the summations with integrals.

3.1 Differential entropy

Given a continuous random variable X with p.d.f. $p(x)$, its differential entropy (or simply entropy) is defined as

$$h(X) = -\int p(x) \ln p(x)\, \mathrm{d}x\,. \tag{18}$$

Note that we use the natural logarithm and use h instead of H (and the unit of differential entropy is the *nat*). The integration is supposed to happen only for those x with $p(x) > 0$. In the continuous case, the integration can be infinite, though. Since the definition only depends on the p.d.f., we can also write the differential entropy as $h(p)$.

Let us take the normal distribution as an example. Let $p(x) = N(x; \mu, \sigma^2)$ be a normal p.d.f.; its differential entropy is

$$\begin{aligned} h(X) &= -\int p(x) \left(-\frac{1}{2} \ln(2\pi\sigma^2) - \frac{(x-\mu)^2}{2\sigma^2} \right) \mathrm{d}x && (19) \\ &= \frac{1}{2} \ln(2\pi\sigma^2) + \frac{\mathrm{Var}(X)}{2\sigma^2} && (20) \\ &= \frac{1}{2} \ln(2\pi e\sigma^2)\,. && (21) \end{aligned}$$

In the above calculation, we used the fact that $\int p(x)\,\mathrm{d}x = 1$ and $\mathrm{Var}(x) = \int p(x)(x-\mu)^2\,\mathrm{d}x = \sigma^2$. The entropy of a standard normal distribution $N(0,1)$ is $\frac{1}{2}\ln(2\pi e)$.

Note that the differential entropy can be zero or negative, which is different from the entropy for discrete random variables. If $\sigma \ll 1$, then the entropy of $N(0,\sigma^2)$ is negative. More precisely, the entropy is negative if $\sigma^2 < \frac{1}{2\pi e} = 0.0585$, or $\sigma < 0.2420$.

The joint entropy of (X, Y) is defined as

$$h(X,Y) = -\int p(x,y) \ln p(x,y)\,\mathrm{d}x\,\mathrm{d}y\,, \tag{22}$$

and can be extended to more random variables. Similarly, the conditional differential entropy is defined as

$$h(X|Y) = -\int p(x,y) \ln p(x|y)\,\mathrm{d}x\,\mathrm{d}y\,. \tag{23}$$

The following equation also holds for conditional entropies:

$$h(X|Y) = h(X,Y) - h(Y) \quad \text{and} \quad h(Y|X) = h(X,Y) - h(X)\,. \tag{24}$$

The mutual information between X and Y is

$$I(X;Y) = \int p(x,y) \ln \frac{p(x,y)}{p(x)p(y)}\,\mathrm{d}x\,\mathrm{d}y\,, \tag{25}$$

and we still have

$$\begin{aligned} I(X;Y) &= h(X) - h(X|Y) && (26) \\ &= h(Y) - h(Y|X) && (27) \\ &= h(X) + h(Y) - h(X,Y)\,. && (28) \end{aligned}$$

Figure 2 is still valid if we replace all H with h.

The KL distance (or KL divergence, or relative entropy) for two p.d.f.s $f(x)$ and $g(x)$ defined on the same domain is defined as

$$\mathrm{KL}(f\|g) = \int f(x) \ln \frac{f(x)}{g(x)} \, \mathrm{d}x \,, \tag{29}$$

and we still have $\mathrm{KL}(f\|g) \geq 0$, $I(X;Y) \geq 0$, and $h(X|Y) \leq h(X)$. The equality in the last two inequalities holds when X and Y are independent. The equality condition for $\mathrm{KL}(f\|g) \geq 0$ is a little bit more complex, and requires $f = g$ *almost everywhere*.[3]

If $p(x)$ is the p.d.f. of a uniform distribution on the range $[a, b]$, its entropy is $\ln(b-a)$.

If $p(x)$ is the p.d.f. of a normal distribution $N(\mu, \sigma^2)$, then its entropy is $\frac{1}{2}\ln(2\pi e \sigma^2)$.

If $p(x)$ is the p.d.f. of an exponential distribution—i.e., $p(x) = \lambda e^{-\lambda x}$ for $x \geq 0$ ($\lambda > 0$) and $p(x) = 0$ for $x < 0$—its entropy is $1 - \ln(\lambda)$.

If $p(x)$ is the p.d.f of a Laplace distribution, i.e.,

$$p(x) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right), \quad (b > 0)\,,$$

its entropy is $1 + \ln(2b)$.

If $X > 0$ is a random variable and $\ln(X)$ follows a normal distribution $N(\mu, \sigma^2)$, X is called a log-normal distribution. Its entropy is $\frac{1}{2}\ln(2\pi e\sigma^2) + \mu$—that is, the entropy of a log-normal random variable X is μ plus the entropy of $\ln(X)$.

3.2 Entropy of a multivariate Gaussian

In this section, we compute the entropy of the d-dimensional multivariate normal distribution

$$p(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma) = (2\pi)^{-\frac{d}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})\right)$$

using a series of simplifications.

First, let the transformation $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{\mu}$ define another random vector Y. The Jacobian of this transformation is $\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = I_d$, and its determinant is 1. So, $p_Y(\boldsymbol{y}) = p_X(\boldsymbol{x} - \boldsymbol{\mu})$, and $\mathrm{d}\boldsymbol{y} = \mathrm{d}\boldsymbol{x}$. We have

$$\int p(\boldsymbol{x}) \ln p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \int p(\boldsymbol{y}) \ln p(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \,. \tag{30}$$

[3]To fully specify the precise meaning of *almost everywhere* requires knowledge from measure theory, which is beyond the scope of this book. $f = g$ almost everywhere means that the set of elements satisfying $f \neq g$ is of measure zero. We can approximately treat "$f = g$ almost everywhere" as meaning the number of elements satisfying $f \neq g$ is at most countably infinite for the continuous distributions we encounter in this book.

In other words, *translating a distribution does not change the differential entropy.* Hence, we just need to compute the entropy of the centered normal $p(\boldsymbol{y}) = N(\boldsymbol{y}; \boldsymbol{0}, \Sigma)$, or $h(Y)$.

Second, we apply the whitening transform to $\boldsymbol{y}$—i.e., $\boldsymbol{z} = \Lambda^{-\frac{1}{2}} U \boldsymbol{y}$ in which $\Sigma = U^T \Lambda U$, U is an orthogonal matrix and Λ is a (positive definite) diagonal matrix, and the new random vector

$$Z \sim N(\boldsymbol{0}, I_d)\,.$$

The Jacobian of this transformation is

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{y}} = \Lambda^{-\frac{1}{2}} U\,,$$

and the determinant of it is $|\Lambda^{-\frac{1}{2}} U| = |\Lambda|^{-\frac{1}{2}} |U| = |\Lambda|^{-\frac{1}{2}} = |\Sigma|^{-\frac{1}{2}}$ (because $|U| = 1$ and $|\Sigma| = |\Lambda|$). How will this transformation affect $h(Y)$?

Let $Z = AY$, where A is a fixed square positive definite matrix and Y is a random vector. Hence, $Y = A^{-1} Z$. The Jacobian of this transformation is A. Since A is positive definite, its determinant $|A|$ is positive. In other words, $|A|$ and $|\det(A)| = \det(A)$ mean the same thing. So, $p_Z(\boldsymbol{z}) = \frac{1}{|A|} p_Y(A^{-1}\boldsymbol{z})$, and $\mathrm{d}\boldsymbol{z} = |A|\,\mathrm{d}\boldsymbol{y}$. Thus, we have

$$\begin{aligned}
h(Z) &= -\int p_Z(\boldsymbol{z}) \ln p_Z(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z} && (31)\\
&= -\int \frac{1}{|A|} p_Y(A^{-1}\boldsymbol{z}) \ln\left(\frac{1}{|A|} p_Y(A^{-1}\boldsymbol{z})\right)\mathrm{d}\boldsymbol{z} && (32)\\
&= -\int p_Y(\boldsymbol{y}) \left(\ln \frac{1}{|A|} + \ln p_Y(\boldsymbol{y})\right)\mathrm{d}\boldsymbol{y} && (33)\\
&= -\int p_Y(\boldsymbol{y}) \ln p_Y(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} + \ln|A| \int p_Y(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} && (34)\\
&= h(Y) + \ln|A|\,. && (35)
\end{aligned}$$

Hence, for a linear transformation $Z = AY$ $(A \succ 0)$, we always have

$$h(Z) = h(Y) + \ln|A|\,. \qquad (36)$$

Applying this rule to the transformation $\boldsymbol{z} = \Lambda^{-\frac{1}{2}} U \boldsymbol{y}$ (and note that $|\Lambda^{-\frac{1}{2}} U| = |\Sigma|^{-\frac{1}{2}}$), we get

$$h(Z) = h(Y) + \ln|\Sigma|^{-\frac{1}{2}}\,.$$

The final step is to compute the entropy $h(Z)$, where Z is a centered spherical normal distribution $N(\boldsymbol{0}, I_d)$. For independent random variables, the entropy of the joint is the sum of the entropies of the individual random variables. We have computed the entropy of a standard normal distribution as $\frac{1}{2}\ln(2\pi e)$. Thus,

$$h(Z) = \frac{d}{2}\ln(2\pi e)\,.$$

Finally, we have

$$h(X) = h(Y) \tag{37}$$

$$= h(Z) - \ln |\Sigma|^{-\frac{1}{2}} \tag{38}$$

$$= h(Z) + \ln |\Sigma|^{\frac{1}{2}} \tag{39}$$

$$= \frac{d}{2}\ln(2\pi e) + \frac{1}{2}\ln|\Sigma| \tag{40}$$

$$= \frac{1}{2}\ln\left((2\pi e)^d|\Sigma|\right)\,. \tag{41}$$

Through a series of simplifications, we have arrived at a neat conclusion: the entropy of a multivariate normal distribution $X \sim N(\boldsymbol{\mu}, \Sigma)$ is

$$h(X) = \frac{1}{2}\ln\left((2\pi e)^d|\Sigma|\right)\,. \tag{42}$$

3.3 The Gaussian as the maximum entropy distribution

Now we are ready to show that the multivariate Gaussian distribution $N(\boldsymbol{\mu}, \Sigma)$ has the largest entropy among distributions whose *mean and entropy exist*, and whose *covariance matrix is* Σ.

Because a translation does not change the entropy, we can assume the multivariate Gaussian X has a p.d.f.

$$p(\boldsymbol{x}) = (2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{x}^T\Sigma^{-1}\boldsymbol{x}\right)\,.$$

Let q be the p.d.f. of another d-dimensional random vector Y; we can assume $\mathbb{E}[\boldsymbol{y}] = \boldsymbol{0}$, and $\mathrm{Var}(Y) = \mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T] = \Sigma (= U^T\Lambda U)$.

One further simplification (which we are familiar with) is to remove the effect of the covariance matrix. Let us define two new random vectors X' and Y' with

$$\boldsymbol{x}' = \Lambda^{-\frac{1}{2}}U\boldsymbol{x}, \quad \boldsymbol{y}' = \Lambda^{-\frac{1}{2}}U\boldsymbol{y}\,.$$

Then, $X' \sim N(\boldsymbol{0}, I_d)$, $\mathbb{E}[\boldsymbol{y}'] = \boldsymbol{0}$, and $\mathrm{Var}(Y') = I_d$. We use p' and q' to denote their p.d.f.s, respectively. The entropy of X' is $\frac{1}{2}\ln\left((2\pi e)^d\right)$.

In the above, we have proved that $h(X') = h(X) + \ln|\Sigma|^{-\frac{1}{2}}$ and $h(Y') = h(Y) + \ln|\Sigma|^{-\frac{1}{2}}$. Hence, to prove X is the maximum entropy distribution, we just need to show $h(X') \geq h(Y')$.

We start from $\mathrm{KL}(q'\|p') \geq 0$, because that is the inequality we are familiar with.

$$0 \leq \mathrm{KL}(q'\|p') \tag{43}$$

$$= \int q'(\boldsymbol{x})\ln\frac{q'(\boldsymbol{x})}{p'(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x} \tag{44}$$

$$= \int q'(\boldsymbol{x})\ln q'(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - \int q'(\boldsymbol{x})\ln p'(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{45}$$

$$= -h(Y') - \int q'(\boldsymbol{x}) \ln p'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \,. \tag{46}$$

We can pause here and take a closer look at the term

$$-\int q'(\boldsymbol{x}) \ln p'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \,.$$

This term is called the *cross entropy* between q' and p'. For two p.d.f.s p and q defined on the same domain, the cross entropy is defined for continuous and discrete distributions as[4]

$$\mathrm{CE}(p,q) = -\int p(\boldsymbol{x}) \ln q(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \,, \tag{47}$$

$$\mathrm{CE}(p,q) = -\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log_2 q(\boldsymbol{x}) \,, \tag{48}$$

respectively. Note that cross entropy is not symmetric. And, we have

$$\mathrm{CE}(q,p) = h(q) + \mathrm{KL}(q\|p) \quad \text{and} \quad \mathrm{CE}(p,q) = h(p) + \mathrm{KL}(p\|q) \tag{49}$$

for any two p.d.f.s p and q. These equalities also hold for discrete random vectors by changing h to H.

Later we will show that cross entropy is very useful for machine learning and pattern recognition. However, for now, if we can show $\mathrm{CE}(q',p') = h(p')$ (when p' is $N(\mathbf{0}, I_d)$ and q' has equal covariance matrix as p'), and the proof is finished. Under these assumptions, we expand the cross entropy:

$$-\int q'(\boldsymbol{x}) \ln p'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = -\int q'(\boldsymbol{x}) \left(\ln((2\pi)^{-\frac{d}{2}}) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} \right) \mathrm{d}\boldsymbol{x} \tag{50}$$

$$= \frac{1}{2} \ln\left((2\pi)^d\right) \int q'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + \frac{1}{2} \int q'(\boldsymbol{x}) \boldsymbol{x}^T\boldsymbol{x} \, \mathrm{d}\boldsymbol{x} \,. \tag{51}$$

Note that $\int q'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 1$ (since q' is a p.d.f.), and

$$\int q'(\boldsymbol{x}) \boldsymbol{x}^T\boldsymbol{x} \, \mathrm{d}\boldsymbol{x} = \sum_{i=1}^{d} \int q'(\boldsymbol{x}) x_i^2 \, \mathrm{d}\boldsymbol{x}$$

in which $\boldsymbol{x} = (x_1, x_2, \dots, x_d)^T$, and $\int q'(\boldsymbol{x}) x_i^2 \, \mathrm{d}\boldsymbol{x}$ is the (i,i)-th entry in the covariance matrix of Y', i.e.,

$$\int q'(\boldsymbol{x}) x_i^2 \, \mathrm{d}\boldsymbol{x} = 1 \,.$$

Hence, the cross entropy equals

$$-\int q'(\boldsymbol{x}) \ln p'(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \frac{1}{2} \ln\left((2\pi)^d\right) + \frac{d}{2} = \frac{1}{2} \ln\left((2\pi e)^d\right) \,, \tag{52}$$

[4]There does not seem to be a widely accepted notation for cross entropy. We use CE in this book.

which is exactly $h(X')$. In other words, if a distribution has bounded mean value and the same covariance as a Gaussian, then the cross entropy between this distribution and the multivariate normal distribution equals the entropy of the Gaussian.

Putting all these equations together, we have $0 \leq -h(Y') + h(X')$. Hence, $h(Y') \leq h(X')$ and consequently $h(Y) \leq h(X)$, which finishes the proof of the Gaussian's maximum entropy property.

4 Information theory in ML and PR

After introducing some basic concepts and facts in information theory, in this section we describe some applications of information theory in machine learning and pattern recognition, but will refrain from diving into their technical details.

4.1 Maximum entropy

We know the maximum entropy distribution is the distribution with the highest uncertainty under certain assumptions, or, without commitment to or favoring any particular point in the space of all distributions that follow these assumptions. For example, the continuous uniform distribution in the range $[a, b]$ treats all points in this range equally.

Sometimes our prior knowledge or training data specifies some constraints. The principle of maximum entropy states that we should seek a distribution that

- satisfies these constraints, and
- has maximum entropy

among all distributions that satisfy the constraints. The principle of maximum entropy is widely used in natural language processing.

Let us assume we need to build a probability mass function p that will translate a Chinese word into English with some randomness. We have a large corpus and know there are four possible candidates `a,b,c,d`. Hence, we have a constraint $p(\mathtt{a}) + p(\mathtt{b}) + p(\mathtt{c}) + p(\mathtt{d}) = 1$. We also require $p(\mathtt{a}) \geq 0$ (and similarly for `b`, `c` and `d`). From the corpus we observe the translations `a` or `c` appear roughly once every three times, which leads to another constraint $p(\mathtt{a}) + p(\mathtt{c}) = \frac{1}{3}$. Then, the translation probability p can be any p.m.f. that satisfy these six constraints. However, the maximum entropy solution will find a p.m.f. that has the largest entropy. Maximum entropy leads to translations that vary and may be more attractive to the readers.

A similar situation occurs in specifying prior distributions. In MAP or Bayesian learning, the prior distribution should encode our knowledge about the parameters (e.g., range of values, approximate mean values, etc.) However, we do not want to introduce any bias apart from the prior knowledge. Hence, the one with maximum entropy is often used—for example, the uninformative prior (uniform distribution in a range) or a normal prior (which is the maximum entropy prior when the variance is fixed.)

4.2 Minimum cross entropy

In a multiclass classification problem, if the logistic regression model is used, the classification model is called the multinomial logistic regression (which is an extension of the logistic regression method) or softmax regression. In an m-class problem, m linear directions $\boldsymbol{\beta}_j$ ($1 \leq j \leq m$) are learned, each corresponding to one class. For any example $\boldsymbol{x}$, the values $\boldsymbol{x}^T\boldsymbol{\beta}_j$ are processed by the softmax transformation and form a probability estimate, i.e.,

$$\Pr(y = j|\boldsymbol{x}) \approx f^j(\boldsymbol{x}) = \frac{\exp(\boldsymbol{x}^T\boldsymbol{\beta}_j)}{\sum_{j'=1}^{m}\exp(\boldsymbol{x}^T\boldsymbol{\beta}_{j'})}\,, \tag{53}$$

in which $f^j(\boldsymbol{x})$ is the estimated probability for $\boldsymbol{x}$ belonging to the j-th class. In a problem with n training examples $(\boldsymbol{x}_i, y_i)$ ($1 \leq i \leq n$, $y_i \in \mathcal{Y} = \{1, 2, \ldots, m\}$), softmax regression maximizes the following objective:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}[\![y_i = j]\!]\log_2 f^j(\boldsymbol{x}_i)\,, \tag{54}$$

in which $[\![\cdot]\!]$ is the indicator function. Similarly to what we have analyzed for logistic regression, softmax regression aims at making the estimated probabilities $f^j(\boldsymbol{x}_i)$ compatible with the training set. In logistic regression, the objective is to maximize $\prod_{i=1}^{n} f(\boldsymbol{x}_i)^{y_i}\,(1 - f(\boldsymbol{x}_i))^{1-y_i}$ ($y_i \in \{0, 1\}$), which is a special case of Equation 54 if we take a base-2 logarithm.

Now consider two distributions p and q, where p is the probability observed from the training set $p_{ij} = [\![y_i = j]\!]$, and q is the probability estimated by our model $q_{ij} = f^j(\boldsymbol{x}_i)$. Then, maximizing the above objective is equivalent to minimizing

$$\mathrm{CE}(p, q) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}\log_2 q_{ij}\,. \tag{55}$$

Hence, multinomial logistic regression seeks to minimize the cross entropy.

In general, if we have a target distribution p (e.g., based on the training set) and a learned estimate q, minimizing the cross-entropy will force the estimate q to simulate the target p. Hence, the cross entropy between target and estimated distributions is called the *cross entropy loss*.

The cross entropy loss is popular in neural network learning. For example, a deep learning classification model can have m nodes as the output, estimating the probability of an example belonging to the m classes, respectively. When an example $\boldsymbol{x}_i$ belongs to the j-th class (i.e., $y_i = j$), the target distribution is a length m vector whose entries are all 0 except the j-th entry (whose value is 1). The loss incurred by $(\boldsymbol{x}_i, y_i)$ is the cross entropy between this target distribution and the output of the network.

Since $\mathrm{CE}(p,q) = H(p) + \mathrm{KL}(p\|q)$ and $H(p)$ (entropy of the target distribution) is constant, to minimize the cross entropy is equivalent to minimizing the Kullback–Leibler divergence. This fact matches our intuition: to make q similar to p, we simply minimize the "distance" between them.

4.3 Feature selection

Mutual information is widely used in feature selection. Given D-dimensional examples $\boldsymbol{x}$, we treat each dimension as a feature. There might be ample reasons to reduce it to d-dimensional ($d \le D$ and usually $d \ll D$). For example, some dimensions could be noise; and D might be too large such that the CPU and memory costs are too high to process the problem in ordinary computers. Dimensionality reduction techniques such as PCA or FLD use all D dimensions to generate d new features (i.e., feature *extraction*). Instead, we may also choose a subset of d dimensions from the original D ones, which is feature *selection*.

Suppose the original feature set is indexed by $O = \{1, 2, \dots, D\}$ and $S \subseteq O$ is a subset of O. We use $f_1, f_2, \dots, f_D$ to denote the original feature dimensions, and $f_S = \{f_i | i \in S\}$ is the subset of features in S. If $\varphi(S)$ measures the fitness of S for our task, feature selection then seeks to maximize the fitness, as

$$\operatorname*{arg\,max}_{S \subseteq O} \varphi(S) \,. \tag{56}$$

For example, if y represents the labels of training examples in a classification problem, we can set

$$\varphi_1(S) = I(f_S; y)$$

to be the fitness measure. If $I(f_S; y)$ is high, this means f_S contains a large amount of information to describe the label y; hence mutual information is a suitable fitness measure for feature selection.

There is one major issue with this simple fitness measure: the complexity of estimating $I(f_S; y)$ grows exponentially with $|S|$, the size of S. The curse of dimensionality makes estimating $I(f_S; y)$ impractical. One idea is to use the marginal distributions to replace the joint distribution of f_S, that is,

$$\varphi_2(S) = \sum_{i \in S} I(f_i; y) \,.$$

The complexity now grows linearly with $|S|$.

However, another problem will occur: the selected features may be redundant. In an extreme example, if all dimensions in a problem are exactly the same, then all D dimensions will be selected. But, any single dimension will give the same classification accuracy as the accuracy using all dimensions. Hence, *redundant* features are not welcome in feature selection. One obvious treatment is: requiring the selected features to have minimal redundancy among themselves.

The redundancy among two selected features f_i and f_j can be measured as $I(f_i; f_j)$. To avoid the curse of dimensionality, we can use these pairwise redundancy numbers to measure the redundancy in a set S:

$$\sum_{i\in S, j\in S} I(f_i; f_j)\,.$$

Hence, the overall objective can be a mix of $\sum_{i\in S} I(f_i;y)$ (to be maximized) and $\sum_{i\in S,j\in S} I(f_i;f_j)$ (to be minimized).

Now, we have all necessary components to introduce the mRMR (Minimum-redundancy-maximum-relevance) feature selection method. We introduce D binary variables $\boldsymbol{s} = (s_1, s_2, \dots, s_D)^T$, $s_i \in \{0,1\}$; $s_i = 1$ means the i-th dimension is selected, and $s_i = 0$ means it is not. The mRMR method is

$$\max_{\boldsymbol{s}\in\{0,1\}^D} \left(\frac{\sum_{i=1}^n s_i I(f_i;y)}{\sum_{i=1}^D s_i} - \frac{\sum_{i=1}^D \sum_{j=1}^D s_i s_j I(f_i;f_j)}{\left(\sum_{i=1}^D s_i\right)^2} \right)\,. \tag{57}$$

Note that the selected subset S is formed by those i with $s_i = 1$.

Since $\boldsymbol{s} \in \{0,1\}^D$ has 2^D candidates, the above optimization is difficult to solve. Various approaches are used in feature selection methods to handle this type of difficulty. A greedy method can first select one single feature with the largest fitness, then add features one by one. The added feature maximizes the difference of two terms: its own fitness, and the sum of its pairwise relevance with features that are already chosen. The greedy method is suboptimal, and leads the optimization process to a local minimum.[5] We can also solve the optimization problem approximately using advanced algorithms.

Feature selection is a huge subject, and we only introduced one feature selection method in this section. There are a lot of research papers, software packages, and surveys available for feature selection.

5 Decision trees

The decision tree is an important method in machine learning and pattern recognition. We will introduce the decision tree model, but only one component is introduced in detail: node splitting using the information gain criterion.

5.1 The XOR problem and its decision tree model

The XOR problem is famous for being *linearly inseparable*. As shown in Figure 3a, four points $(1,1)$, $(1,-1)$, $(-1,1)$ and $(-1,-1)$ belong to 2 classes (blue circle and red square).[6] It is obvious that given a point (x_1,x_2), it belongs to the positive class (denoted by blue circles) if $x_1x_2 = 1$, and the negative

[5]However, for a certain class of functions called the submodular functions, the greedy algorithm works fine. We can roughly treat the minimization of submodular functions as the counterpart of convex optimization in the discrete space.

[6]The original XOR problem uses coordinates 1 and 0, not 1 and -1. However, the problem shown in Figure 3a is equivalent to the original XOR problem.

class (denoted by red squares) if $x_1x_2 = -1$. It is also obvious that any single line cannot separate examples of the two classes perfectly—e.g., linear SVM. Although we can use more complex classifiers such as kernel SVM to solve the XOR problem, the decision tree is a conceptually simple method to solve it (and many other problems).

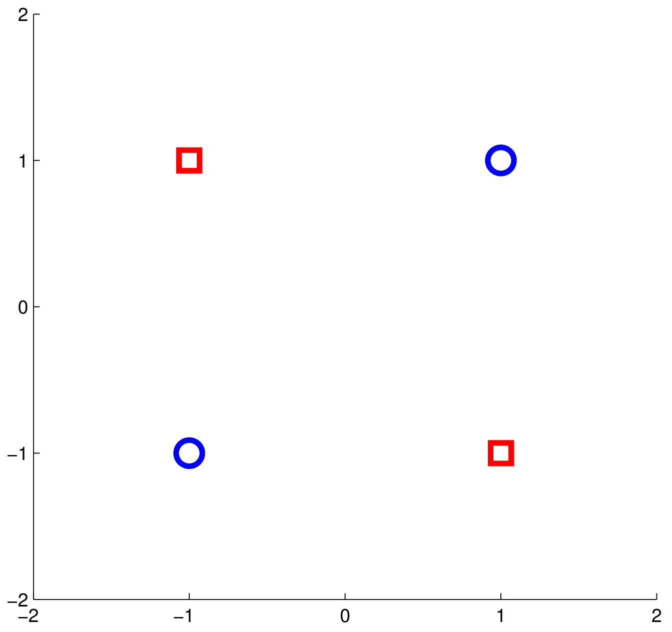
Figure 3b shows the decision tree model for the XOR problem. The model is organized in a tree structure. Given an example $(x_1, x_2) = (1, -1)$, we start from the root of the tree. The root node depends on x_1, and is split into two subtrees depending on whether the value of x_1 is 1 or -1. In our example, $x_1 = 1$ and we traverse to the left subtree. This node depends on x_2. Since $x_2 = -1$ in our example, we traverse to its right subtree. The right subtree is a single node, which provides a decision (red square, or negative). Hence, our example is classified as negative by the decision tree model.

A decision tree has internal nodes and leaf nodes. An internal node depends on an attribute, which is usually one dimension in the input feature vector but can also be a function of several dimensions. Based on the value of an example, an internal node can route it to one of the child subtrees. An internal node can have two or more subtrees. If the root of the selected subtree is still an internal node, the routing process is iterated until the example reaches a leaf node, which gives a prediction.

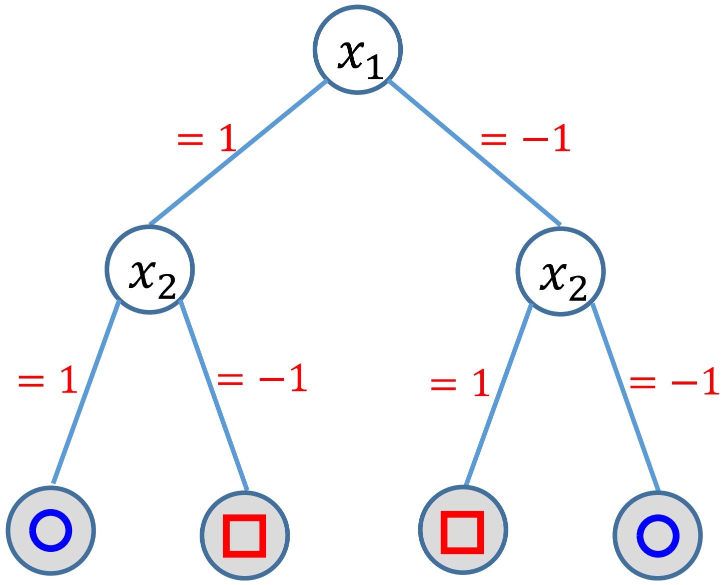
Given a training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, the root node will process all n training examples. Depending on the evaluation result on the root node, training examples will be routed to different subtrees, until they reach leaf nodes. If a leaf node has n' training examples, they will jointly determine the prediction of this leaf node—e.g., by voting of these n' examples. Suppose a decision tree is a perfect binary tree (i.e., every node has 2 subtrees and all leaf nodes have the same depth D), then the number of leaf nodes is 2^{D-1}. Hence, decision trees can model complex nonlinear decisions. And, if the evaluations at internal nodes depend on a single feature, the prediction of a decision tree model is very fast. Furthermore, decision tree algorithms can naturally deal with categorical data.

However, we also anticipate many issues in decision tree learning, for example,

- How to choose the split criterion of an internal node?
- When to stop splitting? We can split an internal node if there are two or more training examples associated with it. However, if a leaf node only has one training example, its prediction may be heavily affected by small amounts of noise in the training data. That is, if the tree depth is too large, the decision tree will overfit. Similarly, a too shallow tree may underfit.
- How can a decision tree deal with real-valued variables?
- If the input feature vector has many dimensions and every dimension is only marginally useful for classification, a decision tree using one dimension for node split is inefficient.



(a) The XOR problem



(b) The XOR decision tree

Figure 3: The XOR problem and its decision tree model. ()

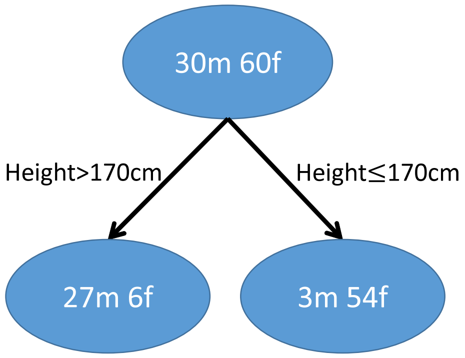


Figure 4: Information gain based internal node split.

- There exist problems that are difficult for single decision tree to handle.

We will only discuss one proposal for the first issue: use information gain to decide the routing/splitting strategy for one node when the features are all categorical. In the decision tree literature, many proposals have been made to solve these issues. For example, one can create a deep tree and prune nodes that may cause overfitting. There are also ways to split nodes using real-valued features and the combination of a set of features. When a problem is difficult for a single tree, the ensemble of many trees (i.e., a decision forest) usually have higher accuracy than a single tree. Readers interested in decision trees can find more details in the literature.

5.2 Information gain based node split

Let us examine a hypothetical example. Suppose 30 male and 60 female students are in a classroom, and we want to predict their gender based on some feature values. If we use the height as the root node's attribute and use 170cm as a threshold, we get the two subtrees in Figure 4.

The root node is split into two new nodes, which seems more friendly for gender classification. For example, if we set the right child node as a leaf node and predict all examples in that node as female, the error rate is only $\frac{3}{57} \approx 5\%$. The left child node, if predicted as male, has a relatively higher error rate of $\frac{6}{33} = 18\%$, which may require further split. In the root node, the best possible error rate is $\frac{30}{90} \approx 33\%$, which is higher than that of both its children.

The classification error is reduced after a node split, because the new nodes are more *pure*—i.e., the distribution of labels of training examples become more concentrated. In information theory, we can say that the entropy of the new nodes are both smaller than the root node. Hence, the information gain criterion tries to find a split that will make the difference in entropy greatest.

Let us suppose an internal node is associated with a set of training examples T. It is split based on a categorical feature, which has K possible values. There are K new nodes, each associated with a set of training examples T_i,

$\bigcup_{k=1}^K T_i = T$ and $T_i \cap T_j = \emptyset$ if $i \neq j$ (in which $\emptyset$ is the empty set). The proportion of examples in T_i with respect to T is

$$w_i = \frac{|T_i|}{\sum_{k=1}^K |T_k|} \,,$$

and $\sum_{k=1}^K w_k = 1$. The *information gain* measures the reduction in uncertainty of our prediction, defined as

$$H(T) - \sum_{k=1}^K w_i H(T_i) \,, \tag{58}$$

in which $H(T)$ is the entropy of labels in the set T.

In our hypothetical example,

$$H(T) = -\frac{1}{3}\log_2(\frac{1}{3}) - \frac{2}{3}\log_2\frac{2}{3} = 0.9183 \,.$$

The left child has $w_1 = \frac{33}{90}$, and

$$H(T_1) = -\frac{27}{33}\log_2\frac{27}{33} - \frac{6}{33}\log_2\frac{6}{33} = 0.6840 \,.$$

Similarly, the right child has $w_2 = \frac{57}{90}$ and

$$H(T_2) = -\frac{3}{57}\log_2\frac{3}{57} - \frac{54}{57}\log_2\frac{54}{57} = 0.2975 \,.$$

Hence, the information gain is

$$\begin{aligned} & H(T) - w_1 H(T_1) - w_2 H(T_2) \\ =& 0.9183 - \frac{33}{90} \times 0.6840 - \frac{57}{90} \times 0.2975 \\ =& 0.4793 \,. \end{aligned}$$

At an internal node, we can test all features and use the one with the greatest information gain to split it. If we denote the classification label using a random variable L and the feature that is used to calculate the information gain as F, then $\sum_{k=1}^K w_i H(T_i)$ is in fact the conditional entropy $H(L|F)$ computed using the training set. Hence, the information gain is

$$H(L) - H(L|F) = I(L;F) \,,$$

in other words, the uncertainty of L that is reduced by knowing F. Hence, it is reasonable to choose the feature that leads to the largest reduction in uncertainty, which equals the mutual information. Because $I(L;F) \geq 0$, the information gain is always non-negative.

Of course, we can also find the feature that minimizes $w_i H(T_i)$, because $H(T)$ does not change when different features are used.

Exercises

1. (Huffman tree for Fibonacci numbers) The Fibonacci numbers are a sequence of number F_n ($n \geq 1$), defined by

$$F_n = F_{n-1} + F_{n-2}, \forall n > 2 \in \mathbb{N} \tag{59}$$
$$F_1 = F_2 = 1\,. \tag{60}$$

(a) Write out the first six Fibonacci numbers.

(b) Prove that for any positive integer n,

$$F_n = \frac{\alpha^n - \beta^n}{\sqrt{5}} = \frac{\alpha^n - \beta^n}{\alpha - \beta}\,,$$

in which $\alpha = \frac{1+\sqrt{5}}{2} \approx 1.618$ and $\beta = \frac{1-\sqrt{5}}{2} \approx -0.618$.

(c) Prove that

$$\sum_{i=1}^{n} F_i = F_{n+2} - 1\,.$$

Hence, $\frac{F_i}{F_{n+2}-1}$ ($1 \leq i \leq n$) forms the p.m.f. of a valid discrete distribution.

(d) Prove that

$$\sum_{i=1}^{n} iF_i = nF_{n+2} - F_{n+3} + 2\,.$$

(e) $\frac{F_i}{F_7-1}$ ($1 \leq i \leq 5$) forms the p.m.f. of a discrete distribution. Draw the Huffman tree for this distribution. In a more general case, what will the Huffman tree be for the distribution $\frac{F_i}{F_{n+2}-1}$ ($1 \leq i \leq n$)?

(f) Prove that the average number of required bits for the tree corresponding to $\frac{F_i}{F_{n+2}-1}$ ($1 \leq i \leq n$) is

$$B_n = \frac{F_{n+4} - (n+4)}{F_{n+2} - 1}\,.$$

(g) What is $\lim_{n\to\infty} B_n$? That is, if n is large, how many bits will the Huffman tree use to encode the distribution $\frac{F_i}{F_{n+2}-1}$ ($1 \leq i \leq n$)?

2. Answer the following questions.

(a) For a function d to be a distance metric, what are the properties that d must satisfy?

(b) Is the KL divergence a valid distance metric? Use the following example to illustrate which properties are satisfied (or not satisfied) by the KL divergence.

In this example, we consider three discrete distributions which have only two possible outcomes. The p.m.f. of these three distributions are specified by

$$A = \left(\frac{1}{2}, \frac{1}{2}\right) , \tag{61}$$

$$B = \left(\frac{1}{4}, \frac{3}{4}\right) , \tag{62}$$

$$C = \left(\frac{1}{8}, \frac{7}{8}\right) . \tag{63}$$

Use pen and paper to check each of the properties on this example.

(c) Write a program to verify your calculations. Be succinct in writing the program.

3. Prove that

$$\mathrm{CE}(p, q) \geq h(p) .$$

When will the equality hold? (Hint: Use the fact $\mathrm{CE}(p, q) = h(p) + \mathrm{KL}(p||q)$.)

4. (Log sum inequality) In this problem, we will prove the log sum inequality and show some of its applications to information theory.

(a) Prove that $f(x) = x \log_2 x$ ($x > 0$) is strictly convex.

(b) Let us define $0 \log_2 0 = 0$, $0 \log_2 \frac{0}{0} = 0$ and $x \log_2 \frac{x}{0} = \infty$ for $x > 0$. These definitions are reasonable. For example, because $\lim_{x \to 0^+} x \log_2 x = 0$, it makes sense to define $0 \log_2 0 = 0$.

Consider two sequences of non-negative numbers $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$. Prove

$$\sum_{i=1}^{n} a_i \log_2 \frac{a_i}{b_i} \geq \left(\sum_{i=1}^{n} a_i\right) \log_2 \left(\frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}\right) . \tag{64}$$

This inequality is called the *log sum inequality*. Although we use $\log_2$ here, any other base can be used in the logarithm. Show that the equality holds if and only if there exists a constant c, such that $a_i = b_i c$ for all $1 \leq i \leq n$.

(c) (Gibbs' inequality) *Gibbs' inequality* is named after the American scientist Josiah Willard Gibbs. Gibbs' inequality states that for any two distributions p and q, the entropy of p is less than or equal to the cross entropy $\mathrm{CE}(p, q)$, and the equality holds if and only if $p = q$. That is,

$$H(p) \leq \mathrm{CE}(p, q). \tag{65}$$

In the previous problem, we have already proved this fact by using 1) $\mathrm{CE}(p, q) = H(p) + \mathrm{KL}(p||q)$ and 2) $\mathrm{KL}(p||q) \geq 0$.

Now, use the log sum inequality to prove Gibbs' inequality. Note that this provides another way to prove $\mathrm{KL}(p||q) \geq 0$.

5. In this problem, we present yet another way to prove the non-negative property of the Kullback–Leibler distance.

   (a) Prove that for any $x > 0$, we have $\ln x \leq x - 1$. When will the equality hold?

   (b) Use this result to prove $\text{KL}(p||q) \geq 0$ and that the equality holds if and only if $p = q$. You can limit your proof to the discrete random variable case.

6. Let X be a continuous random variable with a p.d.f. $q(x)$. Suppose $q(x) > 0$ when $x \geq 0$, and $q(x) = 0$ for $x < 0$. Furthermore, suppose the mean of X is $\mu > 0$ and its entropy exists.

   Show that the exponential distribution with parameter $\lambda = \frac{1}{\mu}$ is the maximum entropy distribution under these constraints.