Day 14 – Properties of the Entropy (cont.)

Elements of Information Theory

Important inequalities for entropy

1. For a random variable $X$ which takes only $k$ values we always have

$$H(X) \leq \log_2(k)$$

with equality if and only if $X$ takes all its values with equal probability.

**Meaning:** Maximum uncertainty about $X$ occurs when all its values are equally likely.

**Fact:** If $m_1 + m_2 + \ldots + m_k = 1$ and $f$ is a concave fun then

$$\sum_{i=1}^{k} m_i \cdot f(x_i) \leq f \left( \sum_{i=1}^{k} m_i \cdot x_i \right)$$

Here $x_i$’s are claims for $f$.

Here, $\log_2(\cdot)$ is a concave fun.

$$H(X) = \sum_{\alpha \in \text{all outcomes}} P(X=\alpha) \cdot \log_2 \left( \frac{1}{P(X=\alpha)} \right)$$

\[ \leq \sum_{\alpha} P(X=\alpha) \cdot \log_2 \left( \frac{1}{P(X=\alpha)} \right) = \log_2 \left( k \right) \]

total # of outcomes for $X$
2. For any two random variables $X$ and $Y$ we always have
\[ H(X|Y) \leq H(X) \]
and equality holds if and only if $X$ and $Y$ are independent.

**Meaning**: The info we gain from learning $X$ after we know $Y$ is less than the amount of information we would gain from learning $X$ if we did not know $Y$.

\[
H(X|Y) = \sum_b p(y = b) H(X|y = b) = \sum_b p(y = b) \sum_a p(x = a | y = b) \log_2 \left( \frac{1}{p(x = a | y = b)} \right)
\]

\[
= \sum_b p(y = b) \sum_a p(x = a \cap y = b) \log_2 \left( \frac{1}{p(x = a | y = b)} \right)
\]

\[
= \sum_a p(x = a) \sum_b p(y = b | x = a) \log_2 \left( \frac{1}{p(y = b | x = a)} \right)
\]

\[
= \sum_a p(x = a) \log_2 \left( \sum_b p(y = b | x = a) \right)
\]

\[
= \sum_a p(x = a) \log_2 p(x = a)^2
\]

\[
= H(X)
\]

\[
\therefore H(X|Y) \leq H(X).
\]
3. For any two random variables $X$ and $Y$ we always have

$$H(X,Y) \leq H(X) + H(Y)$$

and equality holds if and only if $X$ and $Y$ are independent.

**Meaning:** the info we gain on learning $X$ and $Y$ simultaneously is less than the info we would gain if we learned them separately.

$$H(x,y) = H(x) + H(y|x) \leq H(x) + H(y)$$

2nd inequality.
### Entropy of English

Below is the frequency table for the letters in a sample writing of about 1600 English letters (the Emancipation Proclamation):

<table>
<thead>
<tr>
<th>Letter</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>78</td>
</tr>
<tr>
<td>b</td>
<td>74</td>
</tr>
<tr>
<td>c</td>
<td>27</td>
</tr>
<tr>
<td>d</td>
<td>37</td>
</tr>
<tr>
<td>e</td>
<td>77</td>
</tr>
<tr>
<td>f</td>
<td>63</td>
</tr>
<tr>
<td>g</td>
<td>93</td>
</tr>
<tr>
<td>h</td>
<td>27</td>
</tr>
<tr>
<td>i</td>
<td>13</td>
</tr>
<tr>
<td>j</td>
<td>6</td>
</tr>
<tr>
<td>k</td>
<td>5</td>
</tr>
<tr>
<td>l</td>
<td>10</td>
</tr>
<tr>
<td>m</td>
<td>10</td>
</tr>
<tr>
<td>n</td>
<td>8</td>
</tr>
<tr>
<td>o</td>
<td>30</td>
</tr>
<tr>
<td>p</td>
<td>19</td>
</tr>
<tr>
<td>q</td>
<td>1</td>
</tr>
<tr>
<td>r</td>
<td>21</td>
</tr>
<tr>
<td>s</td>
<td>27</td>
</tr>
<tr>
<td>t</td>
<td>19</td>
</tr>
<tr>
<td>u</td>
<td>16</td>
</tr>
<tr>
<td>v</td>
<td>5</td>
</tr>
<tr>
<td>w</td>
<td>2</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>y</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>1</td>
</tr>
</tbody>
</table>

The entropy of English is given by

\[
H(X) = -\sum_{x \in X} p(X = x) \cdot \log_2 \left( \frac{1}{p(X = x)} \right)
\]

\[
= p(a) \cdot \log_2 \left( \frac{1}{p(a)} \right) + p(b) \cdot \log_2 \left( \frac{1}{p(b)} \right) + \ldots + p(z) \cdot \log_2 \left( \frac{1}{p(z)} \right)
\]

\[
= 0.073 \cdot \log_2 \left( \frac{1}{0.073} \right) + 0.009 \cdot \log_2 \left( \frac{1}{0.009} \right) + \ldots + 0.001 \cdot \log_2 \left( \frac{1}{0.001} \right)
\]

The entropy of English is approximately 4.1621. Optimally, we need 4.1621 bits to store a letter in English, on average.
International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is five units.
5. The space between words is a full 10 units.

\[ \cdot \cdot \cdot \Rightarrow H, \ EEEE, \ SE, \ ES, \ldots \]
\[ \ldots \Rightarrow \ldots \]

\[ 1+1+3+3 \Rightarrow A \]
\[ 1+3 \Rightarrow B \]

\[ 3+1+3+1+1+1+3 \Rightarrow U \]

Convert
\[ * := 1 - \text{bil} \]
\[ - := 11 - \text{bil} \]
\[ \text{space} := 0 - \text{bil} \]
\[ \text{end} := 000 - \text{bil} \]

\[ 0 = -- \]
\[ 10101011000 = 11011011000 \]

\[ t_m = \text{Length of this string} \]

on average, we need \( \approx 7 \text{ bits} \) for 1 letter under Morse code.
The Huffman Code

Definition 1. A rooted tree is a connected directed graph in which one vertex has been designated the root, which has no incoming edges, and every other vertex has exactly one incoming edge.

Definition 2. A binary tree is a rooted tree where every internal vertex has no more than 2 children.

A full binary tree is a rooted tree where every internal vertex has exactly 2 children.
Definition 3. A binary code is *comma-free* if concatenation of two code words contains a valid code word that overlaps both.

[Diagram of a binary tree with nodes labeled A, B, C, D, E, F, G and code words for A, B, C, D, E, F, G, and CA = 0100000.]

To decrypt:
1. Start from the root and trace through the branches according to the bits in the code word.
2. If we hit a leaf, replace with the leaf label.
3. Restart from the root.

Question: Given the letter frequencies of a file, which tree will require the least amount of bits? — The Huffman code.

The following algorithm gives the optimal tree:

1. Replace each letter by a node/vertex and label these nodes based on the frequency of each letter. Then sort the nodes by their values in increasing order when reading from left to right.
2. Starting from left to right, group the two smallest numbers together and replace them by their sum.
3. Sort the resulting nodes by their values again. Then repeat these steps until all the nodes are connected.
4. Once we obtain the binary tree, replace the vertex numbers with corresponding letters. Then we label the branches with 0 to the left and 1 to the right.
5. Lastly, we trace along the paths to obtain the code for each letter.
Example. Suppose a certain file contains only the letter with the following frequencies:

<table>
<thead>
<tr>
<th>Letter</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

Construct the context-free code that enables you to compress the file so that you can store it using the least number of bits.

\[
\begin{align*}
4 \cdot N_A + 4 \cdot N_B + 3 \cdot N_C + \ldots + 2 \cdot N_G \\
= 4 \cdot 1 + 4 \cdot 2 + 3 \cdot 2 + \ldots + 2 \cdot 6 = 64
\end{align*}
\]

File length is (after encrypted)

\[
\frac{64}{29} \approx 2.24
\]

Average number of bits per letter is

\[
\frac{64}{29} \approx 2.24
\]

Compare to the entropy of the file

\[
\sum p(n) \log_2 \left( \frac{1}{p(n)} \right) = \frac{1}{24} \log_2 \left( \frac{1}{7/24} \right) + \frac{1}{24} \log_2 \left( \frac{1}{2/24} \right) + \frac{1}{24} \log_2 \left( \frac{1}{1/24} \right) + \frac{5}{24} \log_2 \left( \frac{1}{5/24} \right) + \frac{6}{24} \log_2 \left( \frac{1}{6/24} \right)
\]
This is the optimal $\#$ of bits

given

\[
\frac{4}{24} \log \left( \frac{1}{1/23} \right) + \frac{4}{21} \log \left( \frac{1}{4/21} \right) \\
+ \frac{6}{24} \log \left( \frac{1}{5/24} \right) + \frac{6}{21} \log \left( \frac{1}{6/21} \right)
\]