Optimal binary encoding

November 6, 2023

1 Optimal binary encoding

1.1 Discrete uniformly distributed information source

Construct a discrete information source with an alphabet with a given length n and probabilities $p_i = p(x_i)$ of symbols x_i such that p_i is uniformly distributed, i.e. proportional to i. Since p_i represents a probability, ensure that the sum of all p_i s is equal to one

$$\sum_{i=1}^{n} p_i = 1.$$

Additionally, compute the entropy of this source.

```
Symbol  p_i
a       0.52920
b       0.23137
c       0.09111
d       0.06238
e       0.05998
f       0.02596
Sum over all probabilities = 1.00000
Entropy = 1.91927 bit
```

1.2 Shannon coding

Generate the length N_i for the Shannon binary encoding and determine the average length of the coding. Note that

$$H(X) \leq N_{avg} < H(X) + 1$$

as shown by Shannon's theorem.

PS: It is not necessary to determine the binary prefix code that corresponds with N_i.

```
Symbol  p_i     N_i
a       0.52920 1.00000
b       0.23137 3.00000
c       0.09111 4.00000
d       0.06238 5.00000
e       0.05998 5.00000
f       0.02596 6.00000
N_avg = 2.35530 bit
```

1.3 Huffman coding

Write the algorithm to generate the huffman coding for

$$x_i = ["a", "b", "c", "d", "e"]$$

and

$$p_i = [0.35, 0.3, 0.2, 0.1, 0.05].$$

Creating a class “Nodes” that represents the nodes of the Binary Huffman Tree. Each node consists of

1. a symbol
2. the associated probability,
3. the left and right child in the Huffman tree, and
4. the resulting binary Huffman code.

Define a function “CalculateCodes” which runs recursively from the root to the leaf nodes and generates the binary coding depending on the side chosen (e.g. left for 0 and right for 1).

Implement a small helper function “ReturnLeafNodes” which returns a list which contains all the leaf nodes (= nodes without left or right children).

Implement the Huffman coding by

1. Populating a priority list with the symbols
2. Looping over the priority list until reaching the root node
3. Calculating the codes (CalculateCodes) starting from the root node

The loop over the priority list

1. Picks the 2 nodes x_1, x_2 with the smallest probability p_1, p_2.
2. Creates a new node with x' with $p' = p_1 + p_2$
3. Removes x_1 and x_2 from the priority list and
4. Adds x' to the priority list.

```
Symbol  p_i     code
a       0.35000 00
b       0.30000 01
c       0.20000 10
d       0.10000 110
e       0.05000 111
```