Introduction to source coding

Dr. Yoann Altmann

B39AX – Fall 2023
Heriot-Watt University

LEADERS IN IDEAS AND SOLUTIONS

Plan

- Types of compression
- Lossless compression
  - Expected code length
  - Prefix codes
  - Optimal codes
  - Shannon source coding theorem (symbol code)
  - Huffman code
- Based on the book:
  “Information Theory,. **Inference**, and **Learning Algorithms”**. David J.C. MacKay (Chap. 4-5)

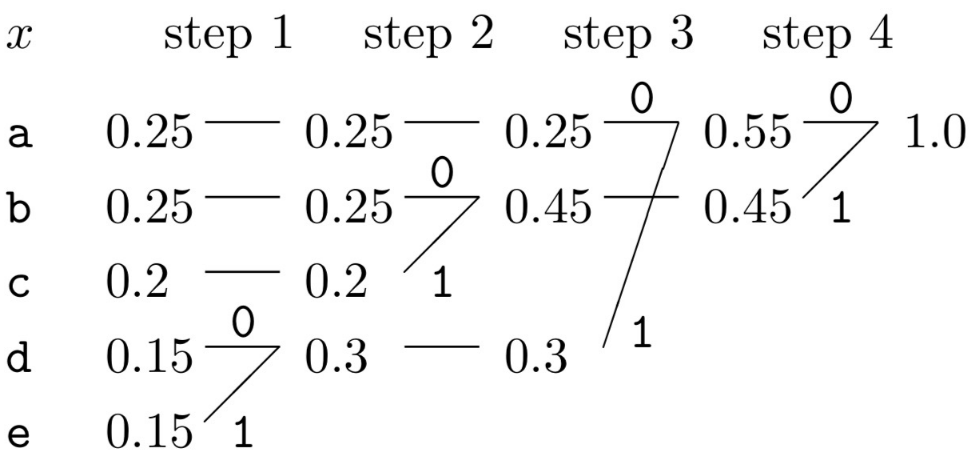
Construction of an optimal code

- So far, we have proved the existence of “good” prefix codes
- We can assess if a code is uniquely decodable
- How can we construct an optimal code?
  - Huffman code

Huffman code

1. Take two least probable symbols in the alphabet
2. Give them the longest codewords differing only in the last digit
3. Combine them into a single symbol
4. Go back to 1.

Example 1



a_i	p_i	$I(a_i)$	$c(a_i)$	l_i
a	0.25	2	00	2
b	0.25	2	10	2
c	0.2	2.3	11	2
d	0.15	2.7	010	3
e	0.15	2.7	011	3

$H(X) = 2.2855$ bits

$L(X, C) = 2.30$ bits

Example 2

$$H(X) = 1.0298 \text{ bits}$$

$$L(X,C) = 1.31 \text{ bits}$$

a_i	p_i	$I(a_i)$	$c(a_i)$	l_i
a	0.8	0.32	0	1
b	0.09	2.41	10	2
c	0.05	3.00	110	3
d	0.06	2.81	111	3

Huffman code

- Optimal: minimizes $L(X, C)$
- Prefix code (easy to decode)

- Limitations:
  – Overhead (between 0 and 1) important if $H(X)$ is small: compression of blocks of symbols instead to increase $H(X)$
  – Context not used (symbol code vs stream code)

Summary

- Lossless compression of symbols
  - Expected code length
  - Prefix codes
  - Optimal codes
  - Shannon source coding theorem (symbol code)
  - Huffman code