COMP2610/6261 - Information Theory

Lecture 15: Shannon-Fano-Elias and Interval Coding

Australian National University

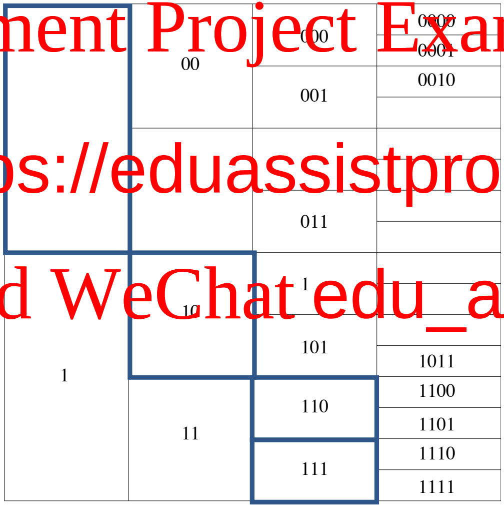
24 September, 2018

1. The Trouble with Huffman Coding

2. Interv
   - Sh
   - Lo
   - The Prefix Property and Intervals
   - Decoding
   - Expected Length

Prefix Codes as Trees (Recap)

$$C_2 = \{0, 10, 110, 111\}$$



The Source Coding Theorem for Symbol Codes

Source Coding Theorem for Symbol Codes

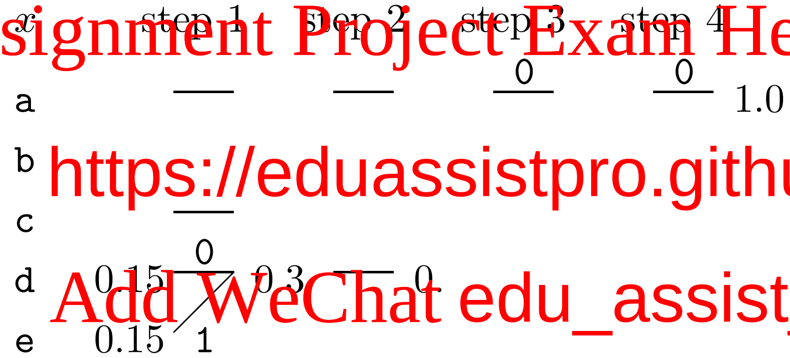
For any en

In particular, **Shannon codes** C — those wit

have *expected code length within 1 bit of the entrop*

Huffman Coding: Recap

$\mathcal{A}_X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}\}$ and $\mathcal{P}_X = \{0.25, 0.25, 0.2, 0.15, 0.15\}$



From Example 5.15 of MacKay

$C = \{00, 10, 11, 010, 011\}$

Huffman Coding: Advantages and Disadvantages

Advantages:

- Huffman Codes are provably optimal amongst prefix codes.
- Algorithm is simple and efficient.

Huffman Coding: Advantages and Disadvantages

Advantages:

- Huffman Codes are provably optimal amongst prefix codes

- Algorithm is simple and efficient

Disadva

- Ass

- The extra bit in the SCT
  - If $H(X)$ is large – not a problem

  - If $H(X)$ is small (e.g., $\sim$ 1 bit for Englis

Huffman codes are the best possible symbol code
but symbol coding is not always the best type of code

This time

A different way of coding (interval coding)

Shannon-Fano-Elias codes

Worse guarantee than Huffman codes, but will lead us to the powerful arithmeti

1 The Trouble with Huffman Coding

2 Interv

- Sh

- Lo
- The Prefix Property and Intervals
- Decoding
- Expected Length

Coding via Cumulative Probabilities

Suppose X is an ensemble with probabilities $(p_1, \ldots, p_{|X|})$

Define the cumulative distribution function by

Coding via Cumulative Probabilities

Suppose X is an ensemble with probabilities $(p_1, \ldots, p_{|X|})$

Define the cumulative distribution function by

and the modified cumulative distribution function by

$$\overline{F}(x) = \sum_{i<x} p_i + \frac{1}{2} \cdot p(x) =$$

Coding via Cumulative Probabilities

Suppose X is an ensemble with probabilities $(p_1, \ldots, p_{|X|})$

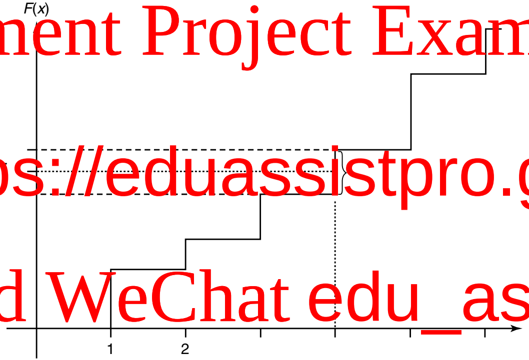
Define the cumulative distribution function by

and the modified cumulative distribution function by

$$\overline{F}(x) = \sum_{i<x} p_i + \frac{1}{2} \cdot p(x) =$$

We can losslessly code outcomes based on $\overline{F}$!

Coding via Cumulative Probabilities



$\overline{F}(x)$ will uniquely determine each outcome x (lossless code)

Example

Suppose X has outcomes (a_1, a_2, a_3, a_4) and probabilities $(2/9, 1/9, 1/3, 1/3)$

Define th

1			
a_2	$1/9$	$1/3$	5
a_3	$1/3$	$2/3$	1
a_4	$1/3$	1	5

Example

Suppose X has outcomes (a_1, a_2, a_3, a_4) and probabilities $(2/9, 1/9, 1/3, 1/3)$

Define th

1			
a_2	$1/9$	$1/3$	5
a_3	$1/3$	$2/3$	1
a_4	$1/3$	1	5

How do we code $\overline{F}(x)$ in binary though?

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$12_{10} \rightarrow 1 \times 10^{1} + 2 \times 10^{0}$$

$$3.7_{10} \rightarrow 3 \times 10^{0} + 7 \times 10^{-1}$$

$$0.94_{10} \qquad + 9 \quad 10^{-1} + 4 \quad 10^{-2}$$

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$12_{10} \rightarrow 1 \times 10^{1} + 2 \times 10^{0}$$

$$3.7_{10} \rightarrow 3 \times 10^{0} + 7 \times 10^{-1}$$

$$0.94_{10} \quad + 9 \quad 10^{-1} + 4 \quad 10^{-2}$$

Some rea

$\frac{1}{3} = 0.$ … $.\overline{142857}_{10}$

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$12_{10} \rightarrow 1 \times 10^{1} + 2 \times 10^{0}$$

$$3.7_{10} \rightarrow 3 \times 10^{0} + 7 \times 10^{-1}$$

$$0.94_{10} \qquad + 9 \quad 10^{-1} + 4 \quad 10^{-2}$$

Some rea

$\frac{1}{3} = 0.$ ... $.\overline{142857}_{10}$

Real numbers can also be similarly expressed in bin

$$3_{10} = 11_{2} \rightarrow 1 \times 2^{1} + 1 \times 2$$

$$1.5_{10} = 1.1_{2} \rightarrow \qquad 1 \times 2 \; + \quad \times$$

$$0.75_{10} = 0.11_{2} \rightarrow \qquad + 1 \times 2^{-1} + 1 \times 2^{-2}$$

$$\frac{1}{3} = 0.010101\ldots_{2} = 0.\overline{01}_{2} \quad \text{and} \quad \frac{22}{7} = 11.001001\ldots_{2} = 11.\overline{001}_{2}$$

Converting Decimal Fractions to Binary

To convert a fraction (e.g. $3/4$) to binary:

1. Multiply the fraction by 2. Take the whole number part of the result; this is the first bit of the binary expansion.

2. Throw away the whole number part of the result, and just retain the part
3. Re
   -
   - you detect an infinite loop

Converting Decimal Fractions to Binary

To convert a fraction (e.g. $3/4$) to binary:

1. Multiply the fraction by 2. Take the whole number part of the result; this is the first bit of the binary expansion.

2. Throw away the whole number part of the result, and just retain the part
3. Re

   - you detect an infinite loop

Example: for 0.625_{10},

- $2 \cdot 0.625 = 1.25$, so first bit is 1
- $2 \cdot 0.25 = 0.5$, so second bit is 0
- $2 \cdot 0.5 = 1.0$, so third bit is 1
- decimal part is zero, so stop

$\bar{F}(x)$ will uniquely determine each outcome...

Shannon-Fano-Elias Coding: To Infinity and Beyond

$\overline{F}(x)$ will uniquely determine each outcome... but coding $\overline{F}(x)$ naively could need infinitely many bits!

- e.g. i

Shannon-Fano-Elias Coding: To Infinity and Beyond

$\overline{F}(x)$ will uniquely determine each outcome... but coding $\overline{F}(x)$ naively could need infinitely many bits!

- e.g. i

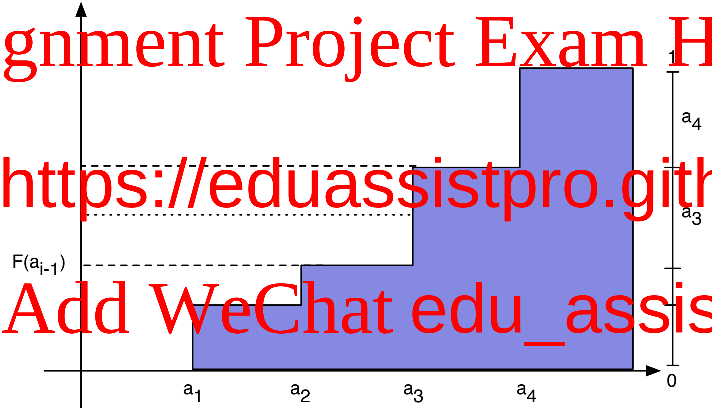
Fortunat tely

Shannon-Fano-Elias coding: code using the fi $\rceil + 1$ bits of $\overline{F}(x)$

- (Almost) Constructive procedure for a Shan

Cumulative Distribution

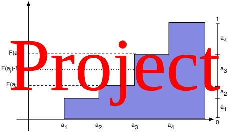
Example



Cumulative distribution for $\mathbf{p} = (\frac{2}{9}, \frac{1}{9}, \frac{1}{3}, \frac{1}{3})$

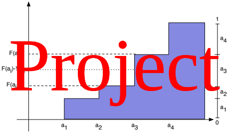
Shannon-Fano-Elias Coding

Example



Shannon-Fano-Elias Coding

Example



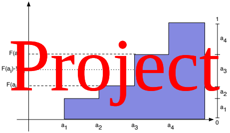
Define th ... $\overline{}$ ... $\log_2 \frac{1}{p_i}\Big\rceil + 1$.

Shan ... $\in$ ... $\ell(\ )$ ... s of $\overline{F}(x)$.

x	$p(x)$	$F(x)$	$\overline{F}(x)$	$\overline{F}$		
a_1	$2/9$	$2/9$	$1/9$	$0.\overline{00}\ \ _2$		
a_2	$1/9$	$1/3$	$5/18$	$0.01\overline{000111}_2$	5	01000
a_3	$1/3$	$2/3$	$1/2$	0.1_2	3	100
a_4	$1/3$	1	$5/6$	$0.1\overline{10}_2$	3	110

Shannon-Fano-Elias Coding

Example



Define th ... $\overline{}$ ... $\bar{}$ ... $\log_2 \frac{1}{p_i} \rceil + 1$.

Shan ... $\in$ ... $\ell(\ )$ ... s of $\overline{F}(x)$.

x	$p(x)$	$F(x)$	$\overline{F}(x)$	$\overline{F}$		
a_1	$2/9$	$2/9$	$1/9$	$0.0\overline{00\ldots}_2$		
a_2	$1/9$	$1/3$	$5/18$	$0.01\overline{000111}_2$	5	01000
a_3	$1/3$	$2/3$	$1/2$	0.1_2	3	100
a_4	$1/3$	1	$5/6$	$0.1\overline{10}_2$	3	110

Example: Sequence $\mathbf{x} = a_3 a_3 a_1$ coded as 100 100 0001.

Remaining questions

Encoding with a Shannon-Fano-Elias code is simple

But we hav

- is th
- is the code prefix-free?
- how do we decode a given codeword?

1 The Trouble with Huffman Coding

2 Interv

- Sh
- Lo

- The Prefix Property and Intervals
- Decoding
- Expected Length

Shannon-Fano-Elias Coding: Is it lossless?

Denote the Shannon-Fano-Elias code for an outcome x by

$$\lfloor \overline{F}(x) \rfloor_{\ell(x)},$$

where $\lfloor \cdot \rfloor_{\ell}$ means truncate to first ℓ bits

Shannon-Fano-Elias Coding: Is it lossless?

Denote the Shannon-Fano-Elias code for an outcome x by

$$\lfloor \overline{F}(x) \rfloor_{\ell(x)},$$

where $\lfloor \cdot \rfloor_\ell$ means truncate to first ℓ bits

Could it be t

Shannon-Fano-Elias Coding: Is it lossless?

Denote the Shannon-Fano-Elias code for an outcome x by

$$\lfloor \overline{F}(x) \rfloor_{\ell(x)},$$

where $\lfloor \cdot \rfloor_{\ell}$ means truncate to first ℓ bits

Could it be t

No, beca

$$F(x-1) < \lfloor \overline{F}(x) \rfloor_{\ell}$$

i.e. the codeword lies entirely in the interval between

- These intervals don't overlap for different outcomes

- The code is lossless!

1. The Trouble with Huffman Coding

2. Interv
   - Sh
   - Lo
   - The Prefix Property and Intervals
   - Decoding
   - Expected Length

Prefixes and Binary Strings

What is the set of binary strings that begin with $\mathbf{b} = \mathtt{b_1 \ldots b_n}$?

Prefixes and Binary Strings

What is the set of binary strings that begin with $\mathbf{b} = b_1 \ldots b_n$?

$b_1 \ldots b_n 0, b_1 \ldots b_n 1, b_1 \ldots b_n 01, b_1 \ldots b_n 11, \ldots$

Prefixes and Binary Strings

What is the set of binary strings that begin with $\mathbf{b} = b_1 \ldots b_n$?

$b_1 \ldots b_n 0, b_1 \ldots b_n 1, b_1 \ldots b_n 01, b_1 \ldots b_n 11, \ldots$

Basically, anything ranging from

These are the strings having $b_1 \ldots b_n$ as a pr

Prefixes and Binary Strings

We could equally associate $b_1 \dots b_n$ with the fraction $0.b_1 \dots b_n$

What is the set of binary strings that begin with $b = b_1 \dots b_n$?

Prefixes and Binary Strings

We could equally associate $b_1 \ldots b_n$ with the fraction $0.b_1 \ldots b_n$

What is the set of binary strings that begin with $\mathbf{b} = b_1 \ldots b_n$?

$0.b_1 \ldots b_n 0, 0.b_1 \ldots b_n 1, 0.b_1 \ldots b_n 01, 0.b_1 \ldots b_n 11, \ldots$

Prefixes and Binary Strings

We could equally associate $b_1 \dots b_n$ with the fraction $0.b_1 \dots b_n$

What is the set of binary strings that begin with $b = b_1 \dots b_n$?

$0.b_1 \dots b_n 0, 0.b_1 \dots b_n 1, 0.b_1 \dots b_n 01, 0.b_1 \dots b_n 11, \dots$

Basicall

$_1 \quad _n \quad _1 \quad _n$

i.e.

$0.b_1 \dots b_n$ to $0.b_1$

Prefixes and Binary Strings

We could equally associate $b_1 \dots b_n$ with the fraction $0.b_1 \dots b_n$

What is the set of binary strings that begin with $b = b_1 \dots b_n$?

$0.b_1 \dots b_n 0, 0.b_1 \dots b_n 1, 0.b_1 \dots b_n 01, 0.b_1 \dots b_n 11, \dots$

Basicall

i.e.

$$0.b_1 \dots b_n \text{ to } 0.b_1$$

Note that

$$0.b_1 \dots b_n \overline{1} = 0.b_1 \dots b_n + \frac{1}{2^n} = 0.b_1 \dots b_n + 0.0 \dots 1,$$

just like $0.1\overline{9}_{10} = 0.2$

Intervals: Definition

It will be useful to analyse the prefix property in terms of intervals

An interv *a* but
smaller th

$$[a, b) = \{x : a \leq$$

Examples: [0, 1), [0.3, 0.6), [0.2, 0.4).

Intervals in Binary

The set of numbers in $[0, 1)$ that start with a given sequence of bits $\mathbf{b} = b_1 \ldots b_n$ form the interval

$$\Big[0.\mathtt{b} \quad \ldots \quad 0.\mathtt{0}\ldots\mathtt{1}\Big)$$

- $\mathtt{1} \rightarrow [\ .\ ,\ .\ )$ $\qquad [0.5, 1]_{10}$
- $\mathtt{01} \rightarrow [0.\mathtt{01}, 0.10)$ $\qquad \ldots 5, 0.5)_{10}$
- $\mathtt{1101} \rightarrow [0.\mathtt{1101}, 0.1110)$ $\qquad [0.8125, 0.875)_{10}$

Prefix Property and Intervals

Prefix property (tree form): Once you pick a node in the binary tree, you cannot pick any of its descendants

Prefix property (interval form): Once you pick a codeword $b_1 b_2 \ldots b_n$ you cannot pick any codeword in

Why? This contains all binary strings for which

e.g. If we pick 0110, we cannot pick anything fro

$$
\begin{aligned}
[0.0110, 0.0111) &= [0.0110\overline{0}, 0.0110\overline{1}) \\
&= \{0.0110, 0.01101, 0.011001, 0.011011, \ldots,\}
\end{aligned}
$$

Prefix Property and Intervals

If $\mathbf{b}'$ is a prefix of $\mathbf{b}$, the interval for $\mathbf{b}$ is contained in the interval for $\mathbf{b}'$

e.g. $\mathbf{b}' = 01$ is prefix of $\mathbf{b} = 0101$ so $[0.0101, 0.0110) \quad \underbrace{[0.01, 0.10)}_{[0.25, 0.5)_{10}}$

Why? Be ... a prefix

- And if $\mathbf{b}$ has $\mathbf{b}'$ as a prefix, so does anythin

Prefix Property and Intervals

If $\mathbf{b}'$ is a prefix of $\mathbf{b}$, the interval for $\mathbf{b}$ is contained in the interval for $\mathbf{b}'$

e.g. $\mathbf{b}' = 01$ is prefix of $\mathbf{b} = 0101$ so $[0.0101, 0.0110)$ $\underbrace{[0.01, 0.10)}_{.25, 0.5)_{10}}$

Why? Be a prefix

- And if $\mathbf{b}$ has $\mathbf{b}'$ as a prefix, so does anythin

Implication: If intervals for $\mathbf{b}, \mathbf{b}'$ are disjoint, o another

Shannon-Fano-Elias Coding is Prefix-Free

We already know $\lfloor \overline{F}(x) \rfloor_{\ell(x)} > F(x-1)$. We also have

$$\lfloor \overline{F}(x) \rfloor_{\ell(x)} + \frac{1}{2^{\ell}} \leq \overline{F}(x) + \frac{1}{2^{\ell}}$$

$$= \quad \frac{p(x)}{\text{[illegible]}} \text{[illegible]}$$

and so

$$\left[\lfloor \overline{F}(x) \rfloor_{\ell(x)}, \lfloor \overline{F}(x) \rfloor_{\ell(x)} + \frac{1}{2^{\ell}} \right) \text{[illegible]}$$

The intervals for each codeword are thus trivially disjoint, since we know each of the $[F(x-1), F(x))$ intervals is disjoint

The SFE code is prefix-free!

Two Types of Interval

The **symbol interval** for some outcome x_i is (assuming $F(x_0) = 0$)

$$[F(x_{i-1}), F(x_i))$$

These intervals are disjoint for each outcome

The **cod**

$$\lfloor \overline{F}(x_i) \rfloor_{\ell(x_i)}, \lfloor \overline{F}(x_i) \rfloor_{\ell(} \quad + \text{———}$$

This is a strict subset of the symbol interval

All strings in the codeword interval start with the same prefix

- This is **not true** in general for the symbol interval

1. The Trouble with Huffman Coding

2. Interv
   - Sh
   - Lo
   - The Prefix Property and Intervals
   - **Decoding**
   - Expected Length

Shannon-Fano-Elias Decoding

To decode a given bitstring:

1. start with the first bit, and compute the corresponding binary interval

2. if the interval is strictly contained within that of a codeword:
    1.
    2.

    3. repeat (1) for the rest of the bitstring
3. else include next bit and compute the corresp

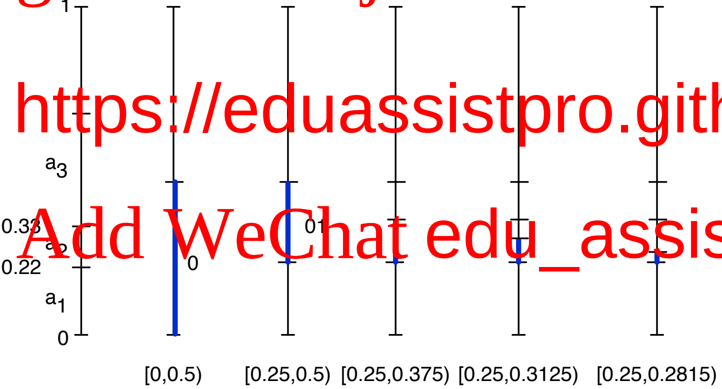
4. ⋮

We might be able to stop early owing to redundancies in SFE

Shannon-Fano-Elias Decoding

Let $\mathbf{p} = \{\frac{2}{9}, \frac{1}{9}, \frac{1}{3}, \frac{1}{3}\}$. Suppose we want to *decode* 01000:

Find symbol interval containing codeword interval for $01000 = [0.25, 0.28125)_{10}$



We could actually stop once we see 0100, since $[0.25, 0.3125) \subset [0.22, 0.33]$

1. The Trouble with Huffman Coding

2. Interv
   - Sh
   - Lo
   - The Prefix Property and Intervals
   - Decoding
   - Expected Length

Expected Code Length of SFE Code

The extra bit for the code lengths is because we code $\frac{p_i}{2}$ and

$$\log_2 \frac{2}{p_i} = \log_2 \frac{1}{p_i} + \log_2 2 = \log_2 \frac{1}{p_i} + 1$$

What is the with probabili

$$L(C, X) = \sum_{i=1} p_i \ell(a_i) = \sum_{i=1}^{K} \ldots$$

$$\leq \sum_{i=1} \ldots \frac{}{p_i}$$

$$= H(X) + 2$$

Similarly, $H(X) + 1 \leq L(C, X)$ for the SFE codes.

Why bother?

Let X be an ensemble, C_{SFE} be a Shannon-Fano-Elias code for X and C_H be a Huffman code for X

2

so why not just use Huffman codes?

SFE is a stepping stone to a more powerful type of code

- Roughly, try to apply SFE to a block of outcome

Summary and Reading

Main points:

- Problems with Huffman coding symbol distribution
- Binary strings to/from intervals in [0, 1]
- Shannon-Fano-Elias Coding:
  -
  -
- Extr

Summary and Reading

Main points:

- Problems with Huffman coding symbol distribution
- Binary strings to/from intervals in [0, 1]
- Shannon-Fano-Elias Coding:
  -
  -
- Extr

Reading:

- Interval coding: MacKay §6.1 and §6.
- Shannon-Fano-Elias Coding: Cover & Tho §

Summary and Reading

Main points:

- Problems with Huffman coding symbol distribution
- Binary strings to/from intervals in [0, 1]
- Shannon-Fano-Elias Coding:
  -
  -
- Extr

Reading:

- Interval coding: MacKay §6.1 and §6.
- Shannon-Fano-Elias Coding: Cover & Tho §

Next time:

Extending SFE Coding to sequences of symbols