Codifica aritmetica / 05-05

Una debolezza di Huffman è sopra la distribuzione di probabilità a priori. Tale codifica si adatta invece sulla frequenza dei simboli presenti, applicandosi su stream di byte. Chiede però quanti valori si quantizzano

Dati quattro valori $x_1 = 1$ $x_2 = 2$ $x_3 = 3$ $x_4 = 4$ e 321124

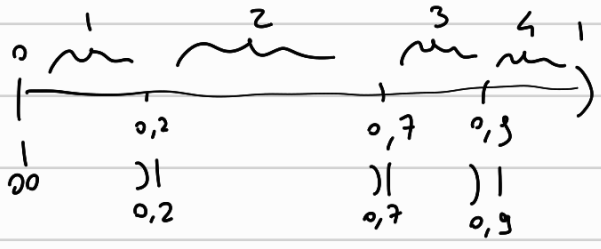
$p_1 = 0{,}2$ $p_2 = 0{,}5$ $p_3 = 0{,}2$ $p = 0{,}1$

$p(x|6) = p(3)\,p(2)\,p(1)\,p(1)\,p(2)\,p(4)$

(n° simboli)

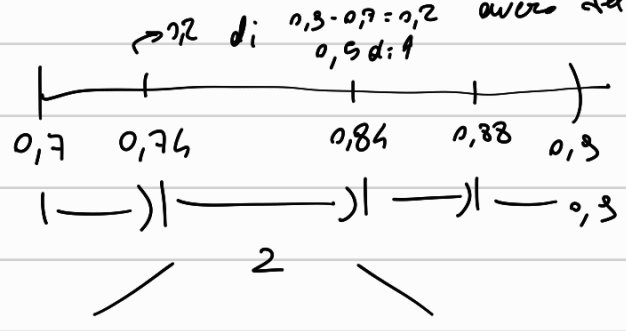
↓ se ho tanti simboli improbabile, anche $p(x|6)$ sarebbe piccolo

L'intervallo sarà compreso fra $[0, 1)$

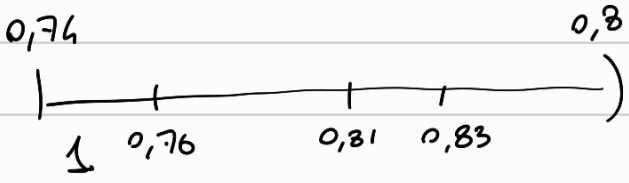


distribuendo le probabilità nell'intervallo

Analizzo la sequenza, ho un 3 quindi ci si aspetta che rimanda fra 0,7 e 0,9



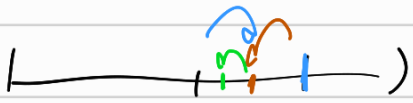
distribuisco le prob. precedenti riadimensionate su questo intervallo



è continuo: finché non finisco non conosco gli estremi

$[\ 0{,}7426074728)$

l'idea è l'approccio binario (maggiore o minore della metà?)



risparmio gli zeri dopo la virgola 12 simboli

0.10111111000105 codifica aritmetica

0,7426758125

$H = \sum p_i \log \frac{1}{p_i} = 1{,}76$ per sei caratteri $= 10{,}56$ vicino a 12

avere lunghezza media di caratteri che mi aspetto

→ migliora la precisione con sep più lunghe

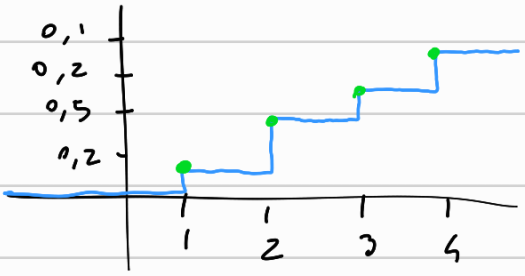
$\overline{\Phi}_k$ è l'intervallo di iterazione k

$\overline{\Phi}_0 = [0,1)$ iniziale $\quad \overline{\Phi}_N$ la meta dell'esplorazione

$\delta_k = \beta_k - \alpha_k$ empirico intervallo $\quad \overline{\Phi}_k = [\alpha_k, \beta_k)$

cdf

• il valore è assunto in quel x



Algoritmo (stringhe di numeri naturali, la cdf è su dei numeri)

for $k = 1 \dots N$

$$\mathbb{D}_k [\alpha_k, \beta_k) = [\alpha_{k-1} + \lambda_{k-1} \ cdf(x_k - 1), \alpha_{k-1} + \lambda_k \ cdf(x_k))$$

Inoltre $\mathbb{D}_0 = [0,1) \quad \lambda_0 = 1$

$\mathbb{D}_1 = [0 + 1 \cdot 0{,}7, 0 + 1 \cdot 0{,}3) \quad \lambda_1 = 0{,}2$

↳ il 1° simbolo della seq. è 3

$\dots [0{,}7426, 0{,}7428)$

Correttezza

$$\log_2 \frac{1}{p(x|6)} = \log_2 \frac{1}{p(3)} + \log_2 \frac{1}{p(2)} + \dots$$

$$\log_2 \frac{1}{p(x|N)} = (\#3) \log \frac{1}{p(3)} + (\#4) \log \frac{1}{p(4)} + (\#1) \log \frac{1}{p(1)} + (\#2) \log \frac{1}{p(N)}$$

$$\frac{\log_2 \frac{1}{p(x|N)}}{N} \simeq p(3) \log \frac{1}{p(3)} + p(1) \log \frac{1}{p(1)} + \dots = H(X)$$

quindi codifica ottimale

Decodifica → devo sapere sempre quanti valori ho

$\vartheta = \vartheta_0$ rappresentazione della codifica

for $k = 1 \dots N$

$$x_k = \{ x \in V_x \ cdf(x-1) < \vartheta_{k-1} < cdf(x) \}$$

$$\vartheta_k = \frac{\vartheta_{k-1} - cdf(x_k - 1)}{p(x_k)}$$

va a vedere $0,7426757\,8125$ dove cade : fino a 2 e 3

$X_k = 3$ al primo giro

Non è necessario sapere la probabilità a priori

0 1

01 | 11001000011 ....

$b_1 b_2 b_3 \ldots$

Codifica binaria $\quad p_0(0) = p_0(1) = \frac{1}{2}$

$$p_k(0) = p_{k-1}(0) + \frac{1}{k+2}\left((1-b_k) - p_{k-1}(0)\right)$$

$$p_k(1) = p_{k-1}(1) + \frac{1}{k+2}\left(b_k - p_{k-1}(1)\right)$$

(si basa sullo stesso concetto di media $m_{11} = m_{10} + \frac{1}{11}(24 - m_{10})$

$\rightarrow \frac{27 \cdot 10 + 24}{11}$)

Tale processo viene eseguito a runtime, senza attese