Algorithmic probability, Part 1 of n

A presentation to the Maths Study Group at London South Bank University 09/09/2015

Motivation

Effective clustering – the partitioning of a collection of objects such that each object within a cluster has more in common with each other object within the same cluster than with any object not in the cluster

And

Categorisation – assigning new objects to an appropriate cluster so that an appropriate response can be supplied

Are essential techniques of machine learning

Problem

Given a collection of objects and data about each object partition the objects into discrete clusters given that:

Some, possibly many, objects will be irrelevant

Some possibly many, data fields will be irrelevant

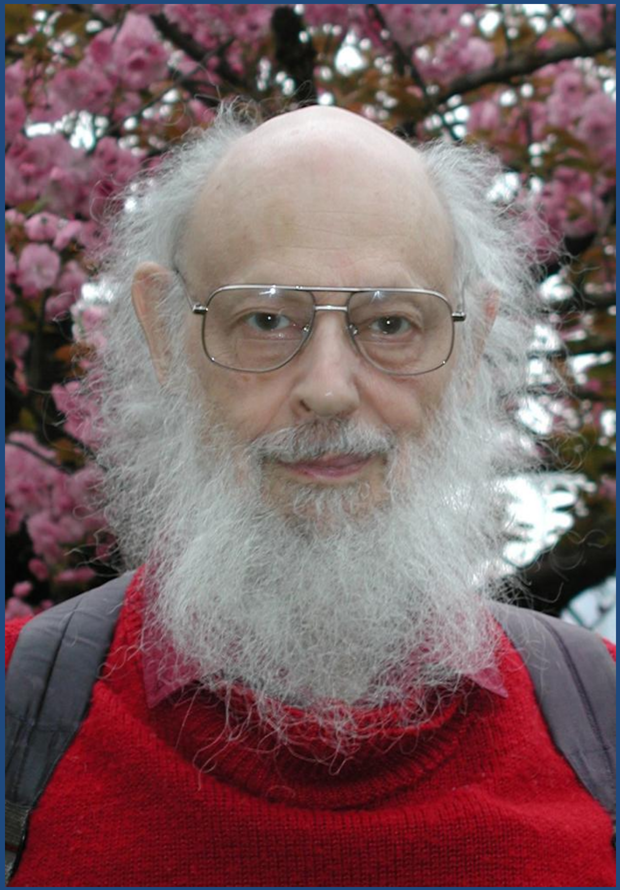
Data might be noisy and or incomplete

Given noiseless, complete and relevant data on relevant objects there may be many possible partitions

How do we determine the most probably correct partition?

Algorithmic Probability Theory

Ray [Solomonoff]



$$P_M(x) = \sum_{i}^{\infty} 2^{-|s_i(x)|}$$

Provides a possible solution

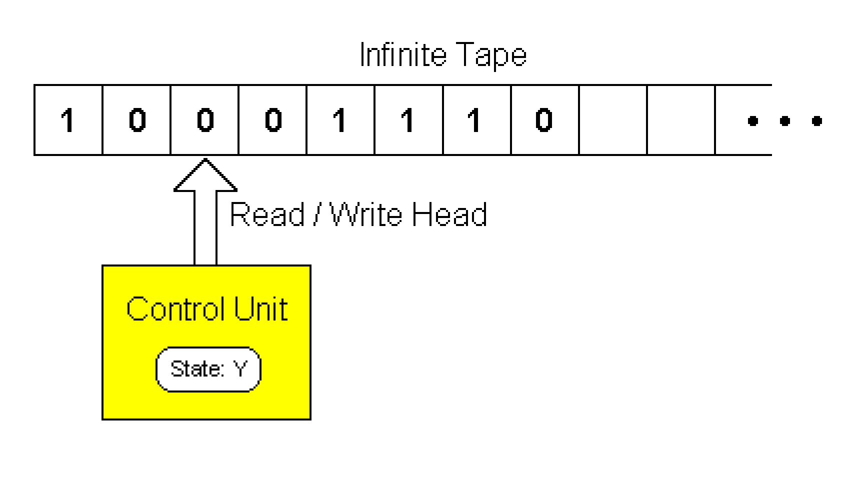
three

are ~~two~~

There ~~is~~ only ~~one~~ kinds of probabilistic induction problem

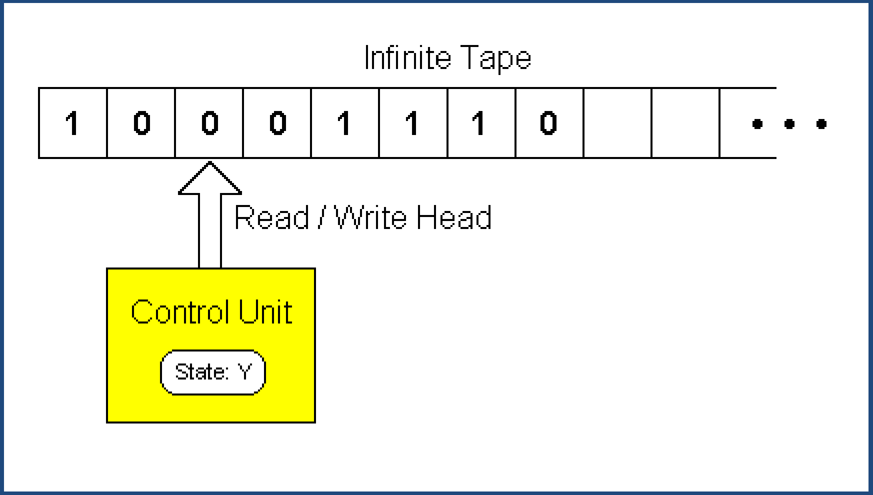
Turing Machine

An infinite tape of discrete cells, a read/write head and a control mechanism. The machine can perform simple operations – read the current symbol, replace it with a new symbol, move the head to the left or right on the tape.



Turing Machine

From its initial state and tape with symbols to the left of the head the machine will read the symbols change state and write new symbols until its stops, at which point the symbols on the tape to the right of the head constitute the output of that machine for the given input.



Turing Machine

The instructions for a TM can be encoded in a quintuple (q, s, q’, s’, d) where:

q = current state

s = symbol currently under the read/write head

q’ = the state to change to

s’ = the symbol to print in the current position

d = the direction in which to move

Universal Turing Machine

The table of rules of a Turing machine T can be encoded as a symbol string <*T*>.

The set of Turing machines can therefore be enumerated $\{T_1, T_2, \ldots\}$

A universal Turing machine can ‘simulate’ all other Turing machines

U simulates machine T_i with input d if fed with input i’d

Universal UIO machines

It can be useful to design slightly more complicated machines

The Universal UIO machine has three tapes and three read/write heads

A unidirectional input tape, a unidirectional output tape and a bidirectional work tape.

The following is from section 2 of Solomonoff, R. J. 1978, **Complexity-Based Induction Systems: Comparisons and Convergence Theorems**, IEEE Transactions on Information Systems Theory Vol. IT-24, No. 4, July 1978

Let M equal a UIO machine with input alphabet 0, 1, b (blank) designed such that it will always stop if it encounters b. So b marks the end of a finite input sequence.

Let x(n) be a possible output sequence of n symbols

Let s be a possible input sequence

s is a code of x(n) with respect to M if the first n symbols of M(s) are identical to those of x(n).

Note: because the output tape is unidirectional once an output symbol has been printed it cannot be changed by subsequent actions.

s is a minimal code of x(n) if

1. s is a code of x(n)
2. When the last symbol of s is removed, the resultant string is not a code of x(n)

All codes for x(n) are of the form $s_i\alpha$, where s_i is a minimal code of x(n) and α may a be null, finite or infinite string.

"It is easy to show that for each n the minimal codes for all strings of length n form a prefix code."

Let N(M, x(n), *i*) be the number of bits in the *i*th minimal code of x(n), with respect to machine M.

N(M, x(n), *i*) = ∞ if there is no code for x(n) on M

Let $x_j(n)$ be the *j*th of the 2^n strings of length n.

N(M, $x_j(n)$, *i*) is the number of bits in the *i*th minimal code of the *j*th string of length n

For a universal machine M we have

$$P'_M(x(n)) \triangleq \frac{\sum_{i=1}^{\infty} 2^{-N(M,x(n),i)}}{\sum_{j=1}^{2^n} \sum_{i=1}^{\infty} 2^{-N(M,x_j(n),i)}} \qquad (1)$$

Unfortunately, due to the halting problem, neither the numerator nor the denominator of (1) are effectively computable

Also, while

$$P'_M(x(n)) \triangleq \frac{\sum_{i=1}^{\infty} 2^{-N(M,x(n),i)}}{\sum_{j=1}^{2^n} \sum_{i=1}^{\infty} 2^{-N(M,x_j(n),i)}}$$

$$\sum_{j=1}^{2^n} P'_M\left(x_j(n)\right) = 1 \qquad (2)$$

It doesn't satisfy

$$P'_M\left(x(n)\right) = P'_M(x(n)0) + P'_M(x(n)1) \qquad (3)$$

FORs (Frames of Reference) are limited machines that do not suffer from the halting problem.

(Willis, D. G. Computational complexity and probability constructions, J. Ass. Comput. Mach., pp 241-259, Apr. 1970)

M_T is the same as the universal UIO machine M except that M_T always stops at time T.

Willis's result for a FOR *R* is:

$$P^R\big(x(n)\big) = \sum_i 2^{-N(R,x(n),i)} \qquad (4)$$

This is unnormalised and does not satisfy (2) or (3)

Usually

$$\sum_{j=1}^{2^n} P^R(x_j(n)) < 1$$

Define $\widetilde{P'}_M$ to be the numerator of (1)

$$\widetilde{P'}_M(x(n)) \equiv \lim_{T\to\infty} \sum_i 2^{-N(M_T,x(n),i)} \qquad (5)$$

Theorem 1: The limit in (5) exists

Proof: The minimum codes for sequences of length n form a prefix set, so by Kraft's inequality,

$$\sum_i 2^{-N(M_T,x(n),i)} \leq 1$$

This quantity is increasing as, as T increases, more and more codes for x(n) can be found. Since any monotonically increasing function that is bounded above must approach a limit, the theorem is proved.

To normalise P'_M, define

$$P'_M\big(x(n)\big) \triangleq \widetilde{P'}_M\big(x(n)\big)C\big(x(n)\big)$$

$$\triangleq \widetilde{P'}_M(x(n)) \prod_{i=0}^{n-1} \frac{\widetilde{P'}_M(x(i))}{\widetilde{P'}_M(x(i)0) + \widetilde{P'}_M(x(i)1)} \qquad (6)$$

P'_M satisfies (2) and (3) for n ≥ 1.

It is readily verified from (6) that *P'_M* satisfies (3) for n ≥ 1.

To show (2) is true for n ≥ 1

Define

$$\widetilde{P'}_M\big(x(0)\big) \triangleq 1, x(0) \textit{the sequence of length zero}$$

$$P'_M(0) = \widetilde{P'}_M(0)\frac{\widetilde{P'}_M(x(0))}{\widetilde{P'}_M(0) + \widetilde{P'}_M(1)}$$

$$P'_M(1) = \widetilde{P'}_M(1)\frac{\widetilde{P'}_M(x(0))}{\widetilde{P'}_M(0) + \widetilde{P'}_M(1)}$$

$$so\ P'_M(0) + P'_M(1) = P'_M\big(x(n)\big) = 1$$
$$and\ thus\ (2) is\ true\ for\ n = 1.$$

(3) Implies that if (2) is true for n, it must be true for n+1. Since (2) is true for n = 1, it must be true for all n QED

References

De Leeuw, K. Moore, E. F. Shannon, C. E. Shapiro, N. 1956, **Computability by Probabilistic Machines**, in Automata Studies, Annals of Mathematics Studies No. 34, Editors, Shannon, C. E. and McCarthy, J.Princeton University Press

Hutter, M. 2005, **Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability,** Springer-Verlag Berlin Heidelberg.

Solomonoff, R. J. 1978, **Complexity-Based Induction Systems: Comparisons and Convergence Theorems**, IEEE Transactions on Information Systems Theory Vol. IT-24, No. 4, July 1978

Willis, D. G. 1970, **Computational complexity and probability constructions**, J. Ass. Comput. Mach., pp 241-259, Apr. 1970