Introduction to Computational Learning

Akihiro Yamamoto 山本 章博
http://www.iip.ist.i.kyoto-u.ac.jp/member/akihiro/
akihiro@i.kyoto-u.ac.jp

2018

What is Computational Learning

- Computational learning is modeling “learning” in the same way as modeling computation.
- Practical applications of computational learning include learning or knowledge discovery from discrete data:

  strings: texts, DNA sequences,…

  trees: parsing trees, XML documents

  tables: relational data,

  graphs: …

Machine Learning

- Machine Learning originally means mechanisms which make machines wiser and wiser by training them more and more.
- Recently Machine Learning also (and mainly) means mechanisms with which we can discover rules or structures hidden behind data.

  "Make unvisible structure be visible"

  - Sometimes we fail in applying machine learning to a specific purpose because of what type of rules would be discovered.

Why discrete data in ML(1)?

- We are surrounded by full of strings, sentences, tables,…

Why discrete data in ML?(2)

- Computers work with strings(sequences) consisting of 0 and 1.

Binary	Oct	Dec	Hex	Glyph	Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@	110 0000	140	96	60	`
100 0001	101	65	41	A	110 0001	141	97	61	a
100 0010	102	66	42	B	110 0010	142	98	62	b
100 0011	103	67	43	C	110 0011	143	99	63	c
100 0100	104	68	44	D	110 0100	144	100	64	d
100 0101	105	69	45	E	110 0101	145	101	65	e
100 0110	106	70	46	F	110 0110	146	102	66	f
100 0111	107	71	47	G	110 0111	147	103	67	g
100 1000	110	72	48	H	110 1000	150	104	68	h
100 1001	111	73	49	I	110 1001	151	105	69	i
100 1010	112	74	4A	J	110 1010	152	106	6A	j
100 1011	113	75	4B	K	110 1011	153	107	6B	k
100 1100	114	76	4C	L	110 1100	154	108	6C	l
100 1101	115	77	4D	M	110 1101	155	109	6D	m
100 1110	116	78	4E	N	110 1110	156	110	6E	n
100 1111	117	79	4F	O	110 1111	157	111	6F	o
101 0000	120	80	50	P	111 0000	160	112	70	p
101 0001	121	81	51	Q	111 0001	161	113	71	q
101 0010	122	82	52	R	111 0010	162	114	72	r

[Wikipedia]

Why sequences in ML?(1)

- Sentences are strings(sequences) consisting of characters in an alphabet.

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikimedia Shop

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia

Toolbox

Print/export

Languages
Acèh
Afrikaans

Article Talk

Computer

From Wikipedia, the free

For other uses, see

"Computer technol

"Computer system"

A **computer** is a gener
operations. Since a se
of problem.

Conventionally, a comp
and some form of mem
sequencing and contro
devices allow informati
retrieved.

The first electronic digi
States. Originally they
personal computers (P

Modern computers bas
machines, and occupy
and mobile computers
the Information Age an
from toys to industrial

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

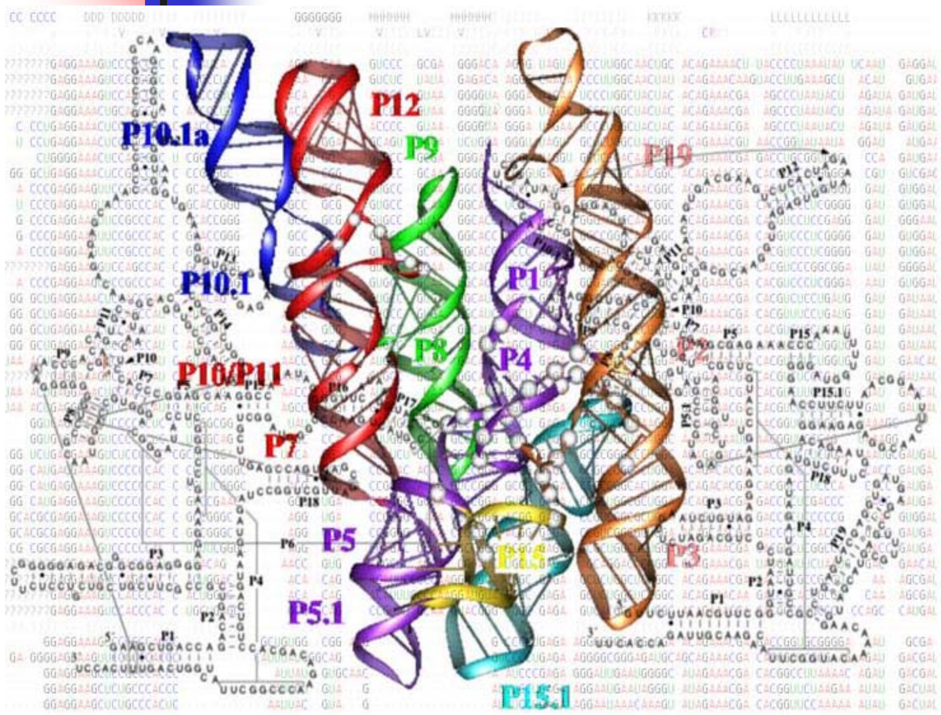
By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of com-

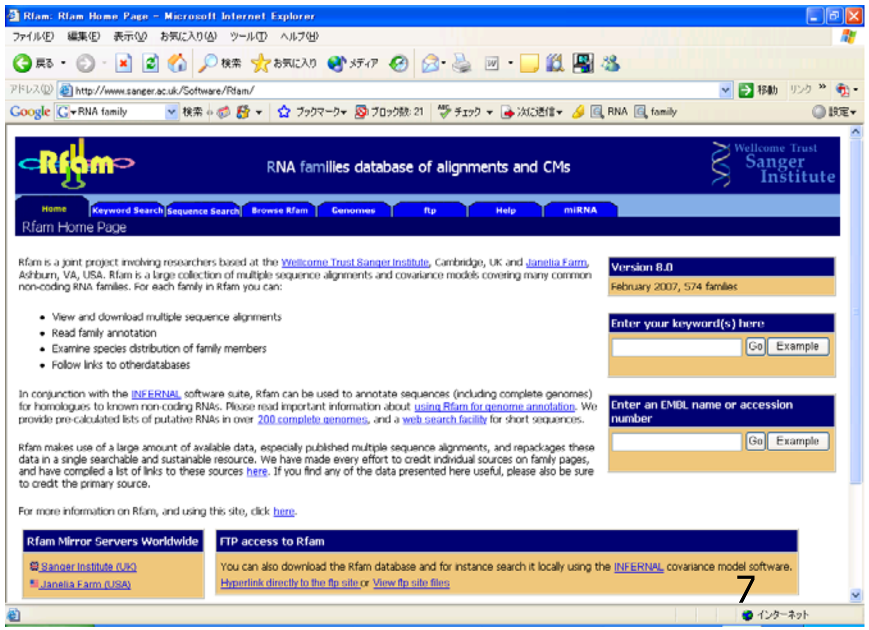
[Wikipedia] [Davis, M. : *The Unsolvable,* Raven Press]

Why sequences in ML?(2)

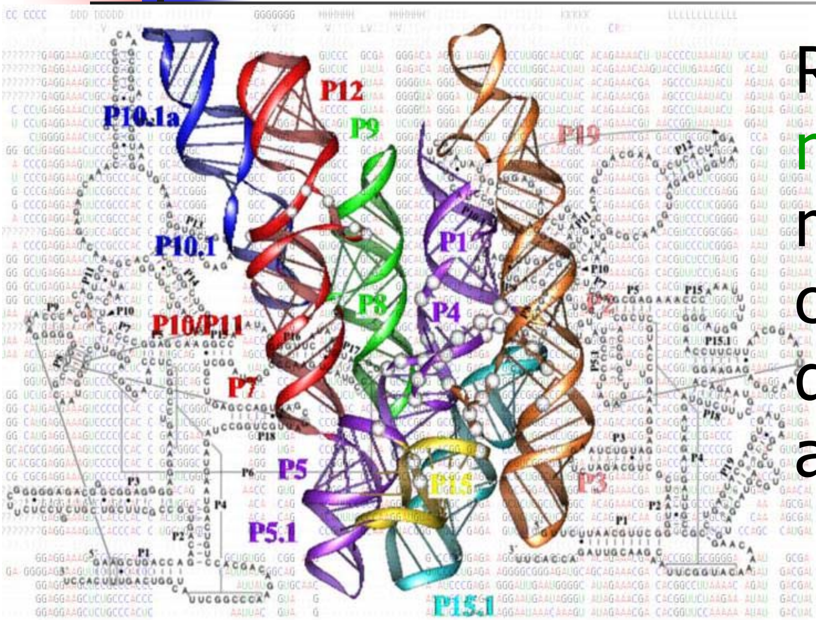


- Many data for academic research is now open. In particular, many string data are provided in the area of bio-informatics.

5′ CACAUGUACAAGACUU 3′



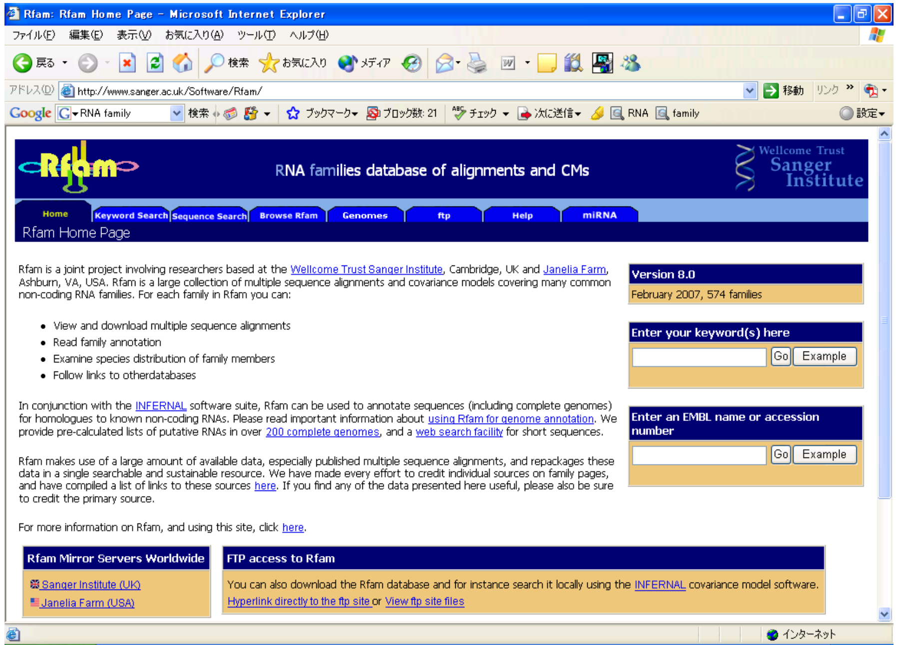
RNA Classification 1



Recently, in bioinformatics, classifying non-coding RNAs (ncRNAs) is paid to much attention, because they are considered to be a factor of the difference between higher organism and others.

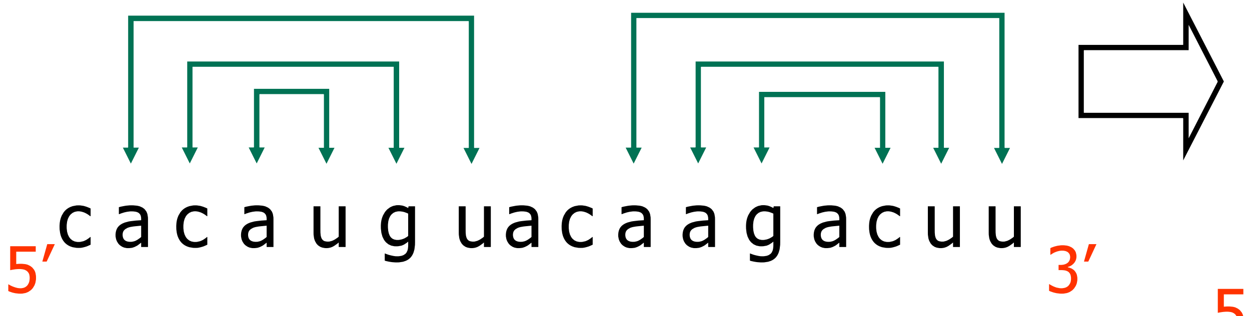
RNA sequences are accumulated in **RNA families**, and the members of each family have similar structures and functions.

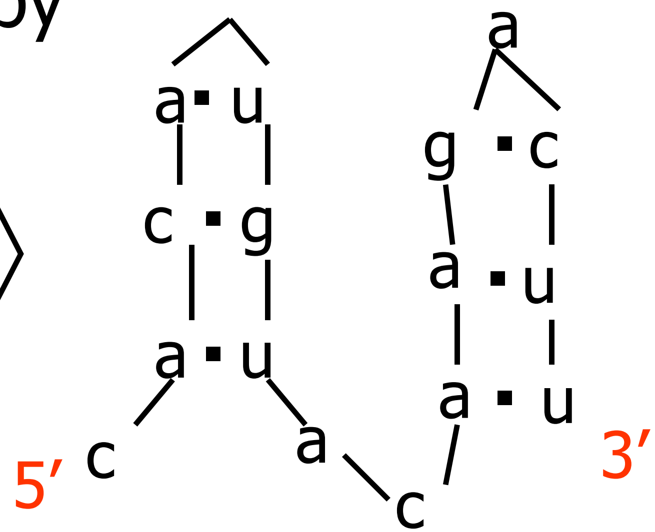
We can get the RNA sequences in Rfam database.



RNA Classification 2

In RNA classification,
the secondary structures detected by base pairs (a-u, c-g) are important.





RNA sequence

secondary structure

The purpose

To distinguish between the member sequences in a given RNA family and non-member sequences by taking secondary structures into account.

Learning from Numerical Data

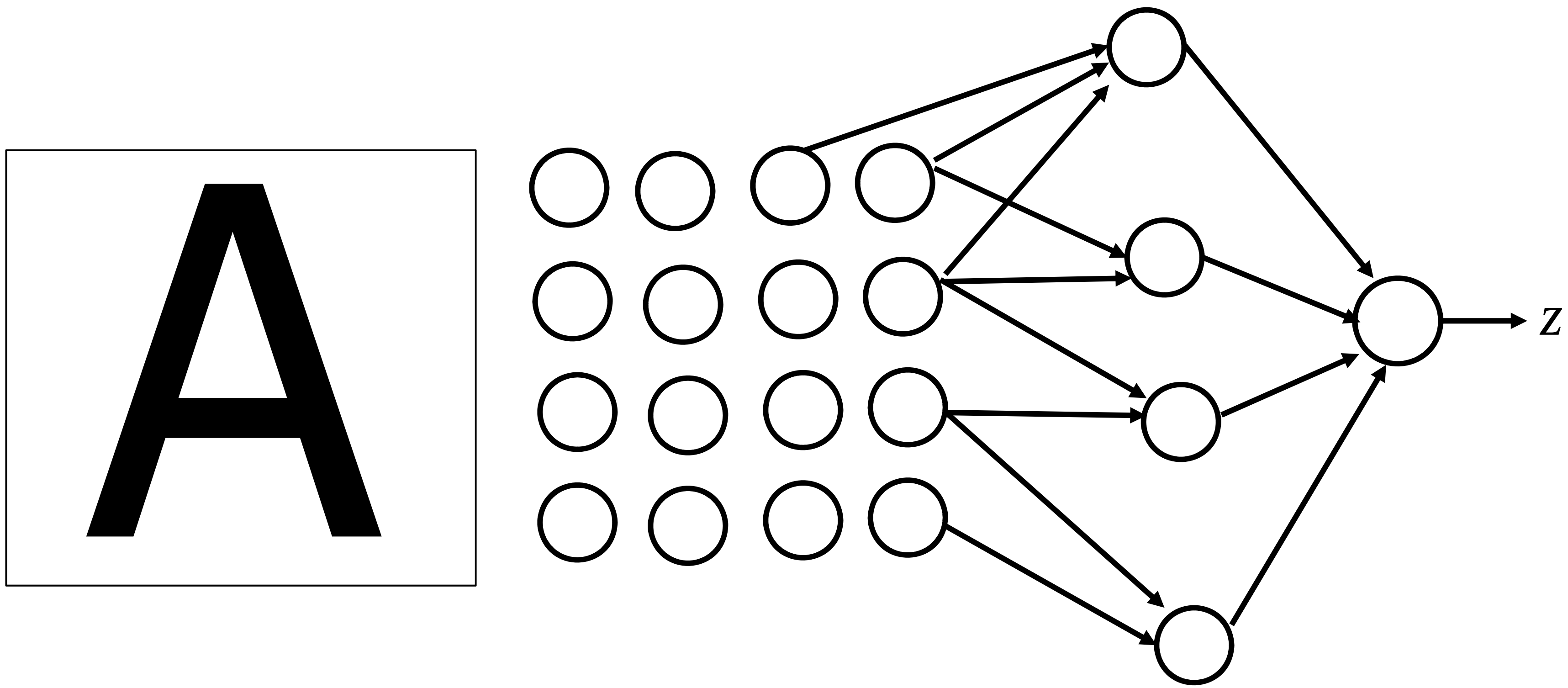
Task

- Every input signal is given to the perceptron with its ‘teaching’ or ‘target’ signal which tells ‘yes’ or ‘no’.

Example The target is A

Input	A	B
Teaching signal	yes	no

Learning Machine



A Simple Learning Method

- Revise the weights w_i according to the combination of the output of the perceptron network and the 'teaching' or 'target' signal.
  - Learning depends on the ways of the revision.
  - The so called "Perceptron Learning" adopt the revision method as follows:

    If the output coincides with the teaching signal, do nothing,

    otherwise add ρ to the weights w_i in the direction to the teaching signal.

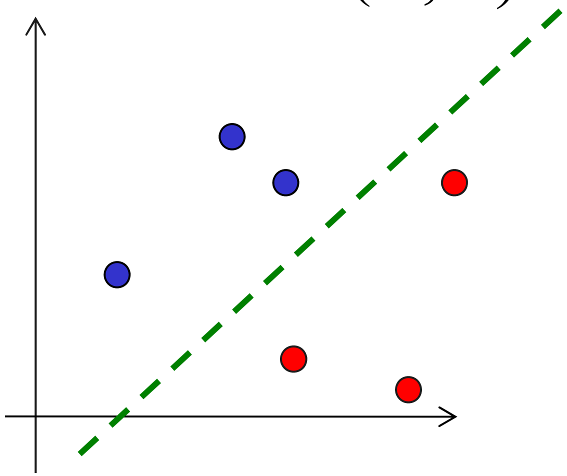
Mathematical Formalization

- In order to our discussion simple , we consider classification into two classes.

Formalization of the Learning Problem

For given two finite subsets C (yes), D (no) ($C \cap D = \emptyset$) in $\boldsymbol{R}^n$, find a line $\boldsymbol{px} + c = 0$ which satisfies

$$\boldsymbol{x} \in C \Rightarrow (\boldsymbol{w}, \boldsymbol{x}) + c > 0$$
$$\boldsymbol{x} \in D \Rightarrow (\boldsymbol{w}, \boldsymbol{x}) + c < 0$$



- In order to find c as well as $\boldsymbol{w}$, we regard every data $\boldsymbol{x}$ as $(\boldsymbol{x}, 1)$ and the target line as $(\boldsymbol{w}', \boldsymbol{x}) = 0$.

A Simple Learning Algorithm

1. Let the input data $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$
2. Initialize $\boldsymbol{w}$ as some value.
3. For $n = 1,2,\ldots, N$,

   if $\boldsymbol{x}_n \in C$ and $(\boldsymbol{w}, \boldsymbol{x}_n) < 0$

   replace $\boldsymbol{w}$ with $\boldsymbol{w} + \rho\,\boldsymbol{x}_n$

   else if $\boldsymbol{x}_n \in D$ and $(\boldsymbol{w}, \boldsymbol{x}_n) > 0$

   replace $\boldsymbol{w}$ with $\boldsymbol{w} - \rho\,\boldsymbol{x}_n$

   otherwise

   do nothing
4. For $n = 1,2,\ldots, N$

   if no $\boldsymbol{x}_n$ satisfies

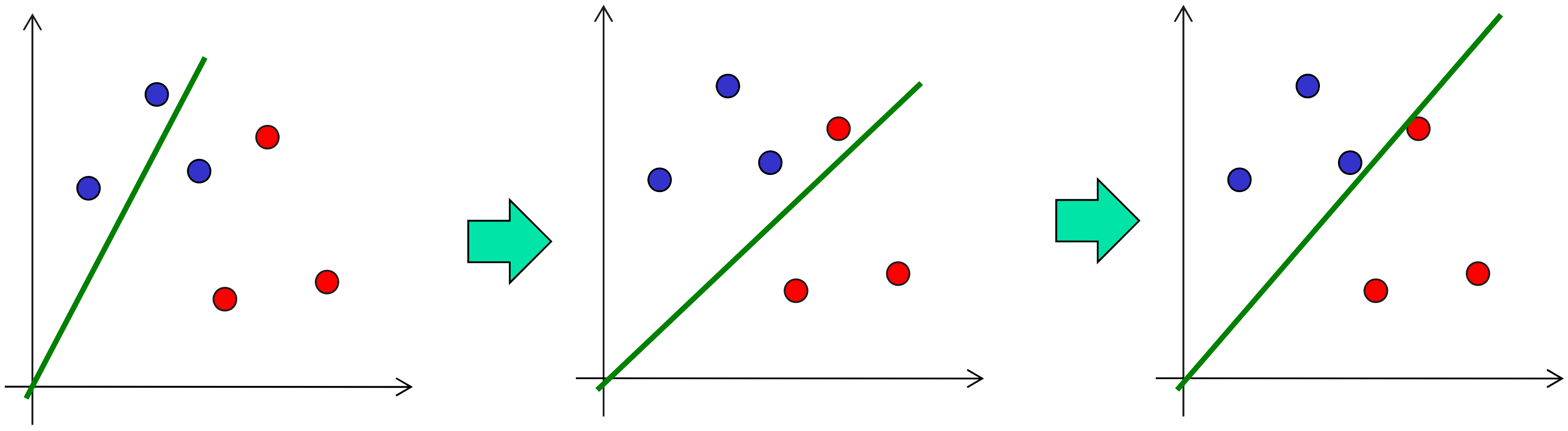
   $(\ \boldsymbol{x}_n \in C$ and $(\boldsymbol{w}, \boldsymbol{x}_n) < 0\ )$ or $(\ \boldsymbol{x}_n \in D$ and $(\boldsymbol{w}, \boldsymbol{x}_n) > 0\ )$

   terminates and return $\boldsymbol{w}$

   else

   go to 3.

Example



Learning from Discrete Data by Embedding

Two approaches

Two approaches can be considered towards learning from discrete data:

- By transforming discrete data into data in $\boldsymbol{R}^n$, in other words, embedding discrete data into $\boldsymbol{R}^n$,

  and apply learning methods for data in in $\boldsymbol{R}^n$.
- By analyzing properties of discrete data, in other words, and using mathematics on discrete data and develop new learning theories and methods for discrete data.

This course is along the second approach.

Example of Transformation (1)

- Let D be the domain of sequences in English.
- We fix a dictionary (bag of keywords) $W = (w_1, w_2, \ldots, w_k)$, and define a transformation Φ as:

$$\Phi(s) = (x_1, x_2, \ldots, x_k) \text{ where}$$

$$x_i = \text{ how many times the keyword } w_i \text{ appears in } s$$

$$\text{for } i = 1,2, \ldots , n$$

Example

W =(book, compute, is, paper, suppose, square, symbol, write)

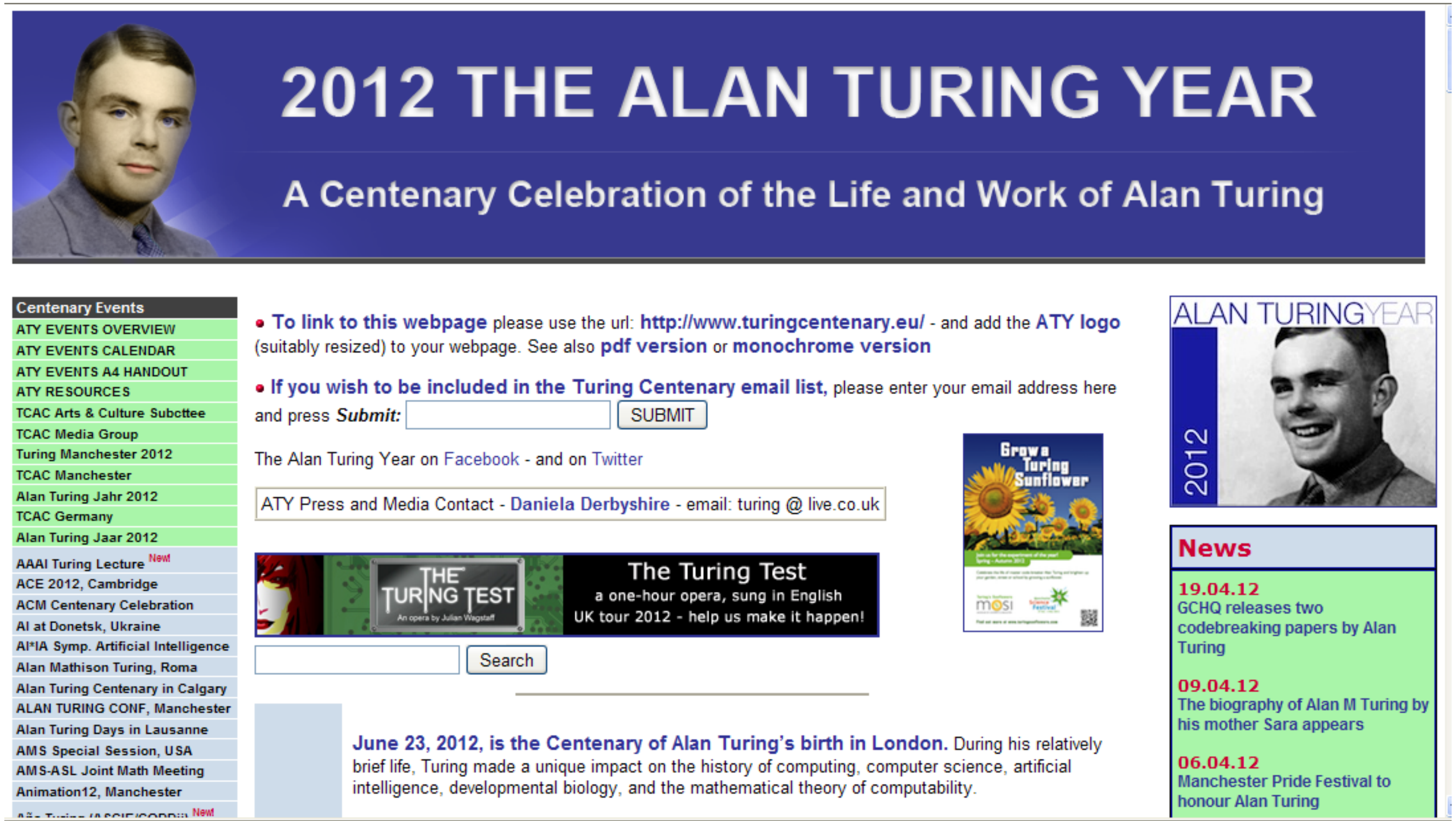
s_1: Computing is normally done by writing certain symbols on paper.

s_2: We may suppose this paper is divided into squares like a child's arithmetic book.

$$\Phi(s_1) = (0, 0, 1, 1, 1, 0, 0, 1, 1)$$

$$\Phi(s_2) = (1, 1, 1, 0, 1, 1, 1, 0, 0)$$

- Alan Turing: On Computable Numbers, with an Application to the Entscheidungsproblem: A correction”. Proceedings of the London Mathematical Society 43: pp. 544–6. 1937. doi:10.1112/plms/s2-43.6.544



Example of Transformation (2)

A simple method for embedding sequences into $\boldsymbol{R}^n$ is using N-grams.

- Let D be the domain of sequences consisting of characters a and b.
- An N-gram is a sequence consisting of N characters.

  For example 3-grams are aaa, aab, aba, …, bbb.
- We have 2^N different N-grams for the domain D, and enumerate them as $w_1, w_2, \ldots, w_n$, where $n = 2^N$.
- We define a transformation $\Phi : D \rightarrow \boldsymbol{R}^n$ as:

$$\Phi(s) = (x_1, x_2, \ldots, x_k) \text{ where}$$

$$x_i = \text{ how often } w_i \text{ appears in } s \quad \text{for } i = 1,2, \ldots , n$$

Example

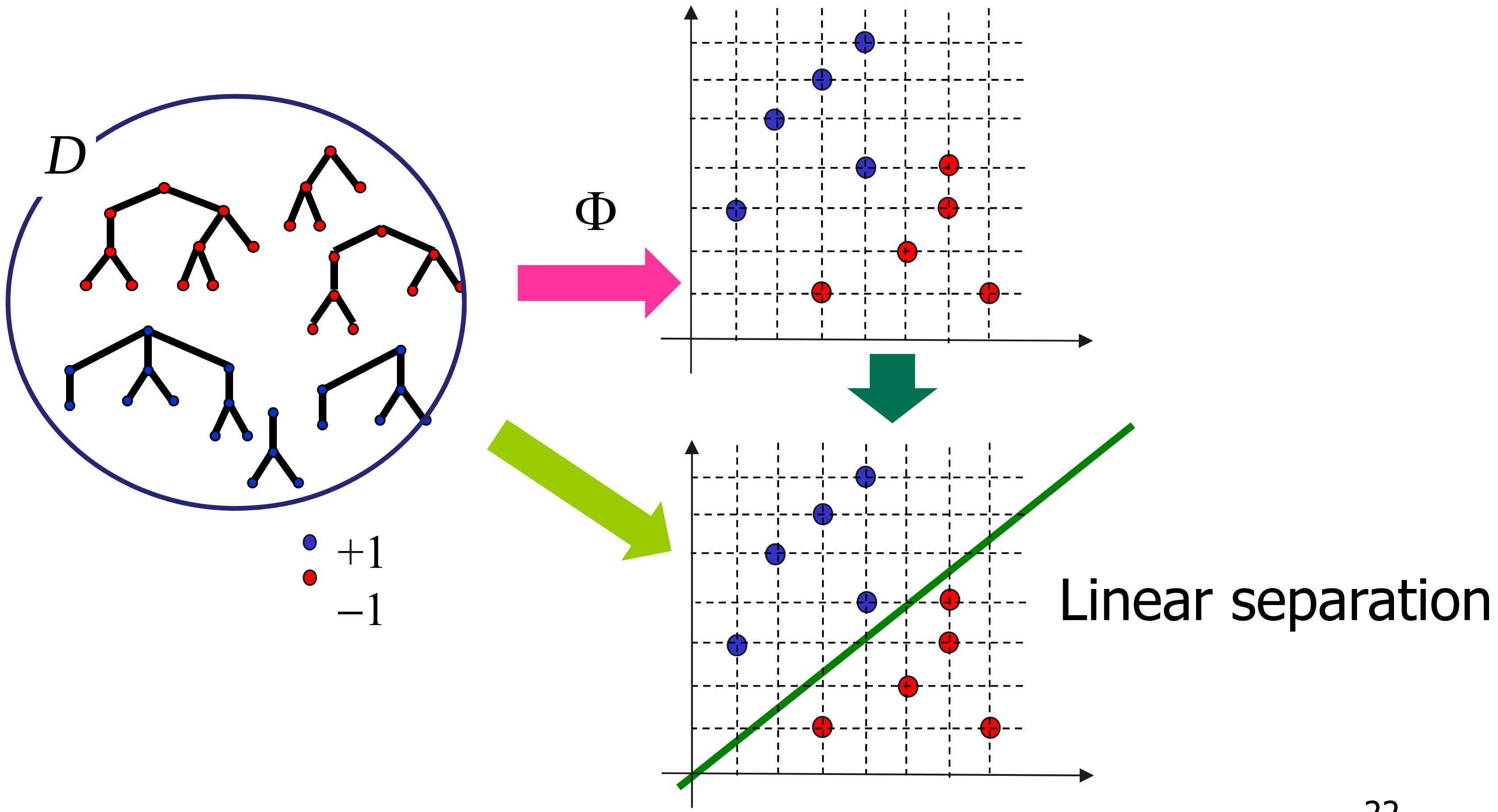
Let w_1= aaa, w_2 = aab, w_3 = aba, w_4 = abb, …, w_8 = bbb

and s = aabbaaabbb.

$$\Phi(\mathrm{s}_1) = (1, 2, 0, 1, \ldots, 1)$$

Example of Transformation (3)

- We do not need the expression of Φ, but need the value $K(x, y) = \Phi(x)\cdot\Phi(y)$, called the kernel function.



Support Vector Machine(1)

- **Input:** a set of numerical data

  $$\{(\boldsymbol{x}_1, c_1), (\boldsymbol{x}_2, c_2), \ldots, (\boldsymbol{x}_m, c_m)\} \quad \boldsymbol{x}_i \in \boldsymbol{R}^n$$

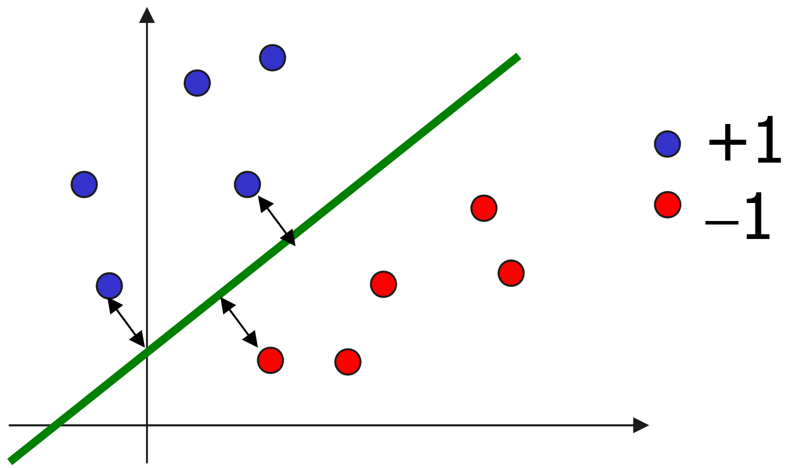
  where each $c_i \in \{+1, -1\}$ is a class signal for $\boldsymbol{x}_i$

  Output: find a liner function (hyper-plane)

  $$f(\boldsymbol{x}) = \Sigma\, w_i\, \boldsymbol{x}_i \cdot \boldsymbol{x} + b$$

  which $sign\,(f(\boldsymbol{x}_i)) = y_i$ for all i and

  maximize the margin $\min_{1 \le i \le m} \mathrm{d}(f, \boldsymbol{x}_i)$



Support Vector Machine(2)

- In order to find c as well as $\boldsymbol{w}$, we regard every data

  $$\boldsymbol{x}_i \text{ as } \boldsymbol{x}_i\text{'} = (\boldsymbol{x}_i, 1) \text{ and } f(\boldsymbol{x}_i\text{'}) = \boldsymbol{w}\text{'} \cdot \boldsymbol{x}\text{'}_i = 0.$$

  Moreover, we can represent two conditions

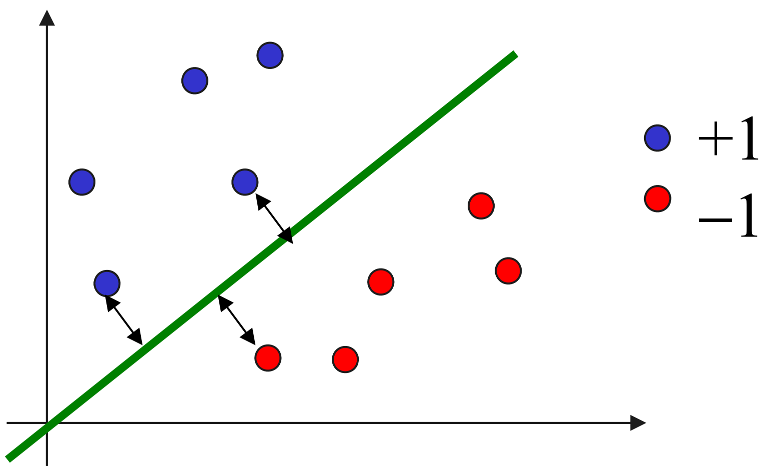
  $$c_i = +1 \Rightarrow \boldsymbol{w}\text{'} \cdot \boldsymbol{x}\text{'}_i \geq 0 \text{ and } c_i = -1 \Rightarrow \boldsymbol{w}\text{'} \cdot \boldsymbol{x}\text{'}_i \leq 0$$

  into one

  $$c_i \, (\boldsymbol{w}\text{'} \cdot \boldsymbol{x}\text{'}_i) \geq 0$$

- In this setting

  $$\mathrm{d}(f, \boldsymbol{x}\text{'}_i) = \| \boldsymbol{x}\text{'}_i \| \cos\theta_i = \frac{1}{\|\boldsymbol{w}\text{'}\|}$$



- In the followings, we write

  $\boldsymbol{x}_i$ for $\boldsymbol{x}_i$' and $\boldsymbol{w}_i$ for $\boldsymbol{w}_i$'.

Kernel function for Boolean Data

- If vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are boolean, the dot product $\boldsymbol{x} \cdot \boldsymbol{y}$ represents: how many coodinates of $\boldsymbol{x}$ and $\boldsymbol{y}$ coincide.

  Example

  $$\boldsymbol{x} = (0, 0, 1, 1, 1, 0, 0, 1, 1)$$
  $$\boldsymbol{y} = (1, 1, 1, 0, 1, 1, 1, 0, 0)$$
  $$\boldsymbol{x} \cdot \boldsymbol{y} = 2$$

  - For two boolean values x and y , the logical conjunction $x \cdot y$ coincides with the product $x\,y$ as real numbers.

- This dot product is too simple and the DNF kernel is developed [Sadohara01, Kahdon05]

  $$K(\boldsymbol{x}, \boldsymbol{y}) = 2^{(\boldsymbol{x} \cdot \boldsymbol{y})} - 1$$

Learning from Discrete Data by Discrete Mathematics

Problems on the first approach

- The obtained data in $\boldsymbol{R}^n$ might not locate densely.
  - They sometimes in $\boldsymbol{N}^n$.
- Even if a rule is obtained by some learning machine, it might be difficult to interpret the rule, or what the rule mean.

On the second approach

We have to know mathematics on discrete data.

- The mathematics may vary from type to type of data.
  - mathematics on sequences, mathematics on trees, mathematics on graphs,…
- We must notice that we need mathematics for machine learning.
  - We make machine learning more abstract, and then observe the correspondence between numerical data and discrete data.

More General Learning

- Recently a machine learning method is recognized as one to find

  $$\mathrm{argmin}_{f \in H} \left(\Sigma_{\boldsymbol{x} \in D} \mathrm{Loss}(f, \boldsymbol{x}) + \lambda \mathrm{P}(f) \right)$$

  where

  Loss(f, $\boldsymbol{x}$) is a loss function and

  P(f) = is a penalty function.

  - This definition is declarative.
  - This course we introduce some of the instances of Loss(f, $\boldsymbol{x}$) and P(f).

Abstract Classification

- A half-plane P which contains C (yes) and excludes D (no) is to be learned
- The half-plane P is represented as a pair $(\boldsymbol{w}, \mathrm{c})$ which means the linear inequation $(\boldsymbol{w}, \boldsymbol{x}) + c > 0$.
  - Let $C(p)=\{\boldsymbol{x} \in \boldsymbol{R}^n \mid p(\boldsymbol{x})\}$ for a predicate p.

    Then the search space (version space) is

    $$\mathrm{C} = \{C(\lambda\, \mathbf{x}.((\boldsymbol{w}, \boldsymbol{x}) + c > 0)) \mid \boldsymbol{w} \in \boldsymbol{R}^n, c \in \boldsymbol{R}^n\}.$$

    The set of parameter s are from

    $$\mathrm{H} = \{(\boldsymbol{w}, c) \mid \boldsymbol{w} \in \boldsymbol{R}^n, c \in \boldsymbol{R}^n\}.$$
- The training examples are provided as the sets C and D.
- A learning algorithm is provided.

Learning from string data

- Assume that we are treating data on the domain of sequences of characters.
- Then we treat the problem of classifying two finite sets of sequences C (yes), D (no) ($C \cap D = \emptyset$).

Example

Let D be the domain of sequences consisting of characters a and b.

C = {ab, aab, abaab, aaab, aaaabbbb,abab}

D = {a, b, bbbb, abba, baaaba, babbb}

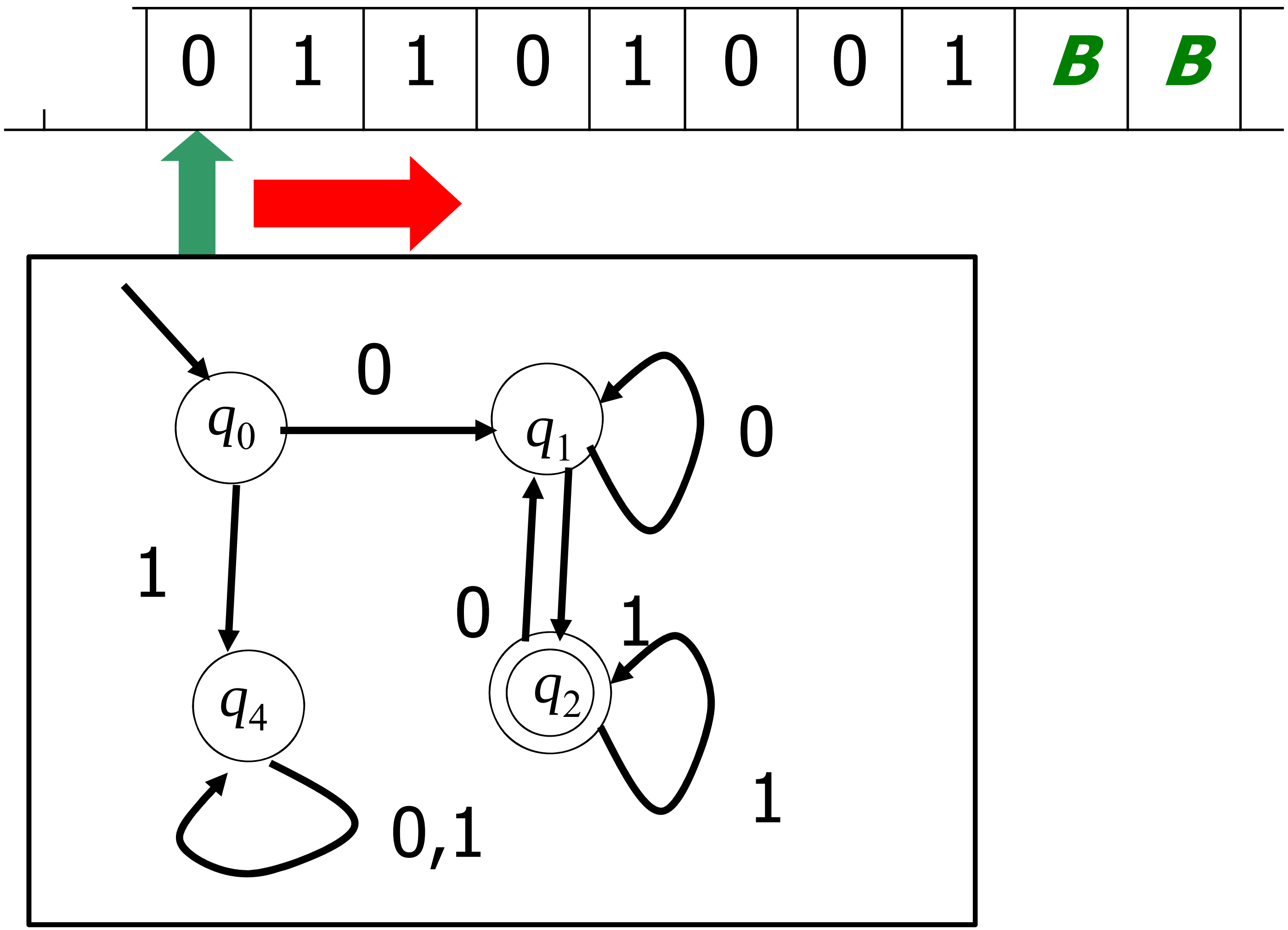
How to distinguish data

In this course we treat the following methods:

- Abstract machines to distinguish data
  - Finite state automata, Tuning Machine, …
- Formal grammar with production rules
  - Linear grammar, Context free grammar, …
- Regarding string data as mathematical objects
  - Based on the operation : aaba means a▪(a▪(b▪a)))

⇨ monomials (patterns), instead of linear combinations

Finite state automaton



Grammar with productions

- $S = \{ \text{a}^n\text{b}^n \mid n \geq 1 \} = \{ \underbrace{\text{a...a}}_{n \text{ times}}\underbrace{\text{b...b}}_{n \text{ times}} \mid n \geq 1 \}$

  $= \{\text{ab, aabb, aaabbb, aaaabbbb,...} \}$

- The language is defined with a set of productions:

  $S \rightarrow \text{ab}, \ S \rightarrow \text{a}S\text{b}$

- Some examples of derivations:

  $S \Rightarrow \text{ab}$

  $S \Rightarrow \text{a}S\text{b} \Rightarrow \text{aabb}$

  $S \Rightarrow \text{a}S\text{b} \Rightarrow \text{aa}S\text{bb} \Rightarrow \text{aaabbb}$

  $S \Rightarrow \text{a}S\text{b} \Rightarrow \text{aa}S\text{bb} \Rightarrow \text{aaa}S\text{bbb} \Rightarrow \text{aaaabbbb}$

- It is easy to show that there is no FA which accepts L.

Patterns (Monomials)

- A pattern π is a sequence consisting symbols and variables
  - Assuming that we can distinguish characters and varaibles.

Example

Characters: a, b, Variables : x, y,…

Patterns: axb, bxayb, aa$x$bybxa,...

The set defined with a pattern

L(axb) = {aab, abb, aaab, aabb, abab, abbb,... }

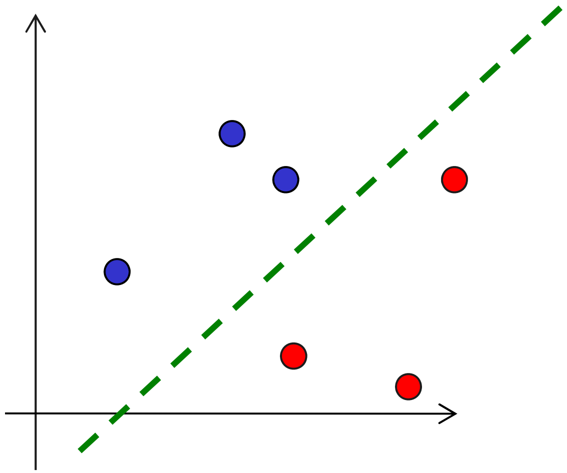
L(bxayb) = {baaab, baabb, baaaab, baaabb, baabab,...}

Making the learning be abstract

- In the case of treating sequences, what is the correspondence to the linear inequation?

$$\boldsymbol{x} \in C \Rightarrow \boxed{(\boldsymbol{w}, \boldsymbol{x}) + c \geq 0}$$ parameter $(\boldsymbol{w}, \boldsymbol{c})$

$$\boldsymbol{x} \in D \Rightarrow (\boldsymbol{w}, \boldsymbol{x}) + c \leq 0$$



A Learning Algorithm

1. Let the input data $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$
2. Initialize $\boldsymbol{w}$ as some value.
3. For $n = 1,2,\ldots, N$,

   if $\boldsymbol{x}_n \in C$ and $(\boldsymbol{w}, \boldsymbol{x}_n) < 0$

   replace $\boldsymbol{w}$ with $\boldsymbol{w} + \rho\, \boldsymbol{x}_n$

   else if $\boldsymbol{x}_n \in D$ and $(\boldsymbol{w}, \boldsymbol{x}_n) > 0$

   replace $\boldsymbol{w}$ with $\boldsymbol{w} - \rho\, \boldsymbol{x}_n$

   otherwise

   do nothing
4. For $n = 1,2,\ldots, N$

   if no $\boldsymbol{x}_n$ satisfies

   $(\boldsymbol{x}_n \in C$ and $(\boldsymbol{w}, \boldsymbol{x}_n) < 0)$ or $(\boldsymbol{x}_n \in D$ and $(\boldsymbol{w}, \boldsymbol{x}_n) > 0)$

   terminates and return $\boldsymbol{w}$

   else

   go to 3.

Machines with parameters

- We regard the inequation $(\boldsymbol{w}, \boldsymbol{x}) + c \geq 0$ as a machine to distinguish whether or not every datum $\boldsymbol{x}$ is in C.

- For the case of treating strings, we should adopt machines which can distinguish whether or not every string $\boldsymbol{x}$ is in C.

Learning Finite State Automata

1. Let the input data $x_1, x_2, \ldots, x_N$
2. Initialize L as some automaton.
3. For $n = 1,2,\ldots, N$,

    if $x_n \in C$ and M does not accept x_n

        replace M with another M'

    else if $x_n \in D$ and M accepts x_n

        replace M with another M'

    otherwise

        do nothing

4. For $n = 1,2,\ldots, N$

    if no x_n satisfies

    ($x_n \in C$ and M does not accept x_n) or ($x_n \in D$ and M accepts x_n)

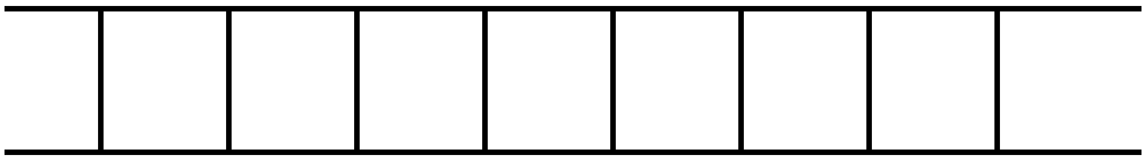
        terminates and return M

    else

        go to 3.

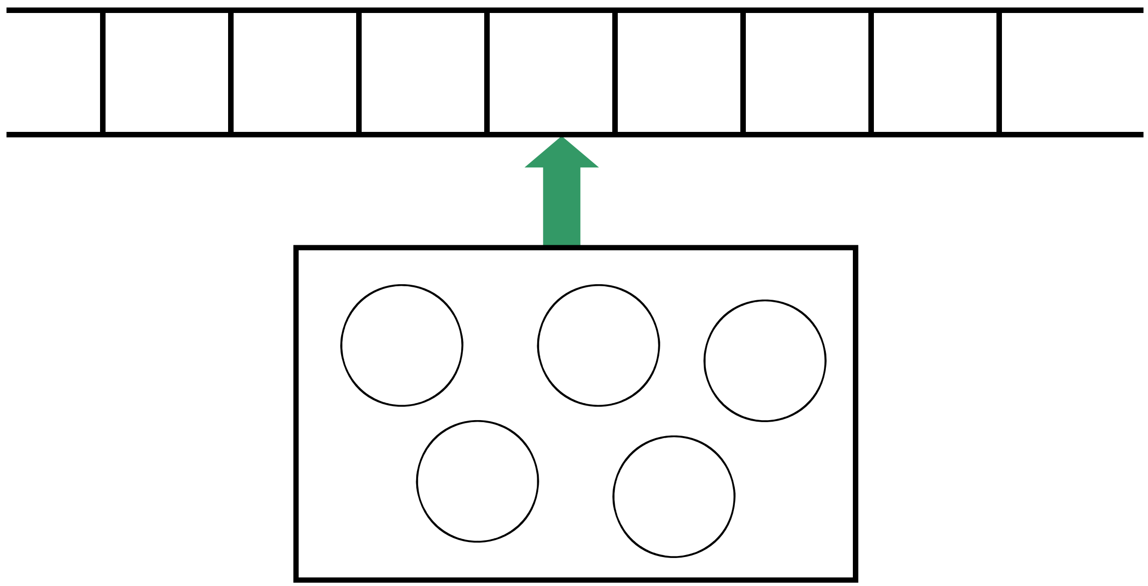
Turing Machine(2)

- Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book.
- I assume then that the computation is carried out on one-dimensional paper, i.e. on a tape divided into squares.

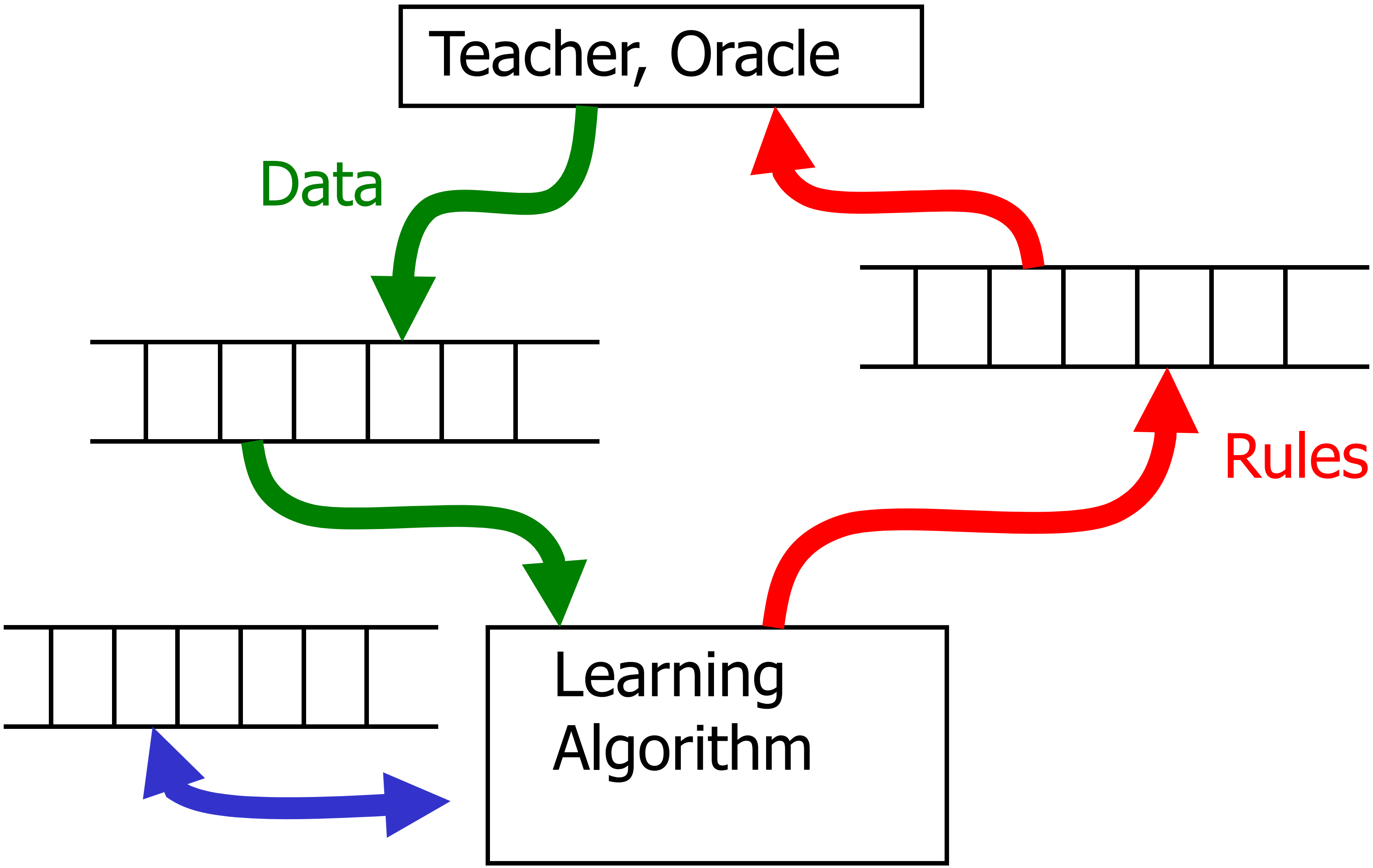


Turing Machine(3)

- The behaviour of the computer at any moment is determined by the symbols which he is observing and his "state of mind" at that moment.
- We will also suppose that the number of states of mind which need be taken into account is finite.



Learning Machine



Conclusion

Elements of Learning Theories

- A class of rules to be learned
- A uniform representation method of each rule
  - We assume that each rule is represented by an expression/a formula defined by a grammar.
- A representation method of training examples/observation
- A learning algorithm
- A method evaluation / some criteria of justification of the learning algorithm

References

- Colin de la Higuera, *Grammatical Inference*, Cambridge University press, 2010..
- 横森・榊原・小林:“計算論的学習” 培風館