Boolean Functions for Classification: Logical Analysis of Data

Yves Crama

HEC Management School, University of Liège, Belgium

and Endre Boros

Rutgers University, USA

Based on numerous papers by E. Boros, Y. Crama, P.L. Hammer, T. Ibaraki, A. Kogan, K. Makino, etc.

Outline

- Boolean Functions
- Learning from Examples
- Partially Defined Boolean Functions
- Logical Analysis of Data

Outline

- **Boolean Functions**
- Learning from Examples
- Partially Defined Boolean Functions
- Logical Analysis of Data

Boolean functions

Consider a function:

$$(x_1, x_2, \ldots, x_n) \to F(x_1, x_2, \ldots, x_n).$$

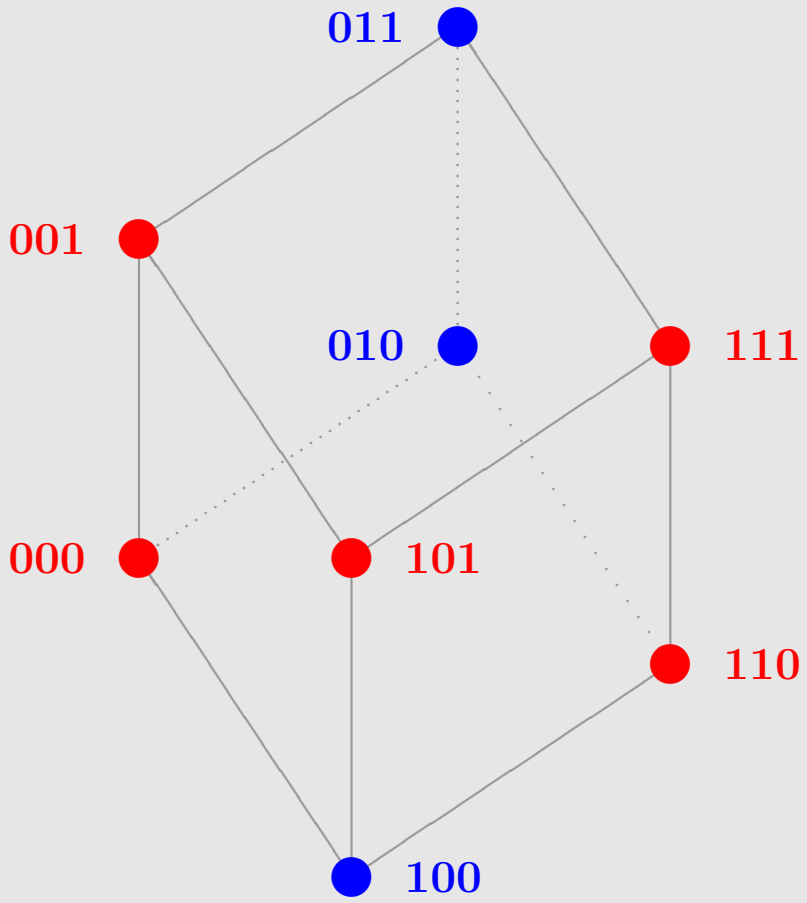
To be interesting,

- each variable should take at least two distinct values, and
- the function should take at least two distinct values.

So: the most elementary interesting functions are those for which each variable takes exactly two values, and for which the function itself can take exactly two values.

Named Boolean functions, after George Boole (1815-1864).

Example



red = 0

blue = 1

Applications of Boolean functions

- In spite of their simplicity, Boolean functions possess a rich theory and have found an amazing array of applications over the last 150 years.

- Boole was interested in modeling human reasoning (*Laws of Thought*): Each variable and output can be interpreted as "True" or "False".

- Electrical and electronic engineering: signal goes through a network depending on the state of intermediate gates (Open or Closed).

- Computer science: computation output is 0 or 1 depending on the initial input (in binary format: string of 0's and 1's).

- Game theory and social choice: a resolution is adopted (Yes or No) by a governing body depending on the votes (Yes or No) of individual members.

- Artificial intelligence: an action is taken (Yes or No) depending on the presence or absence of certain features (e.g, medical diagnosis: prescribe additional tests or not).

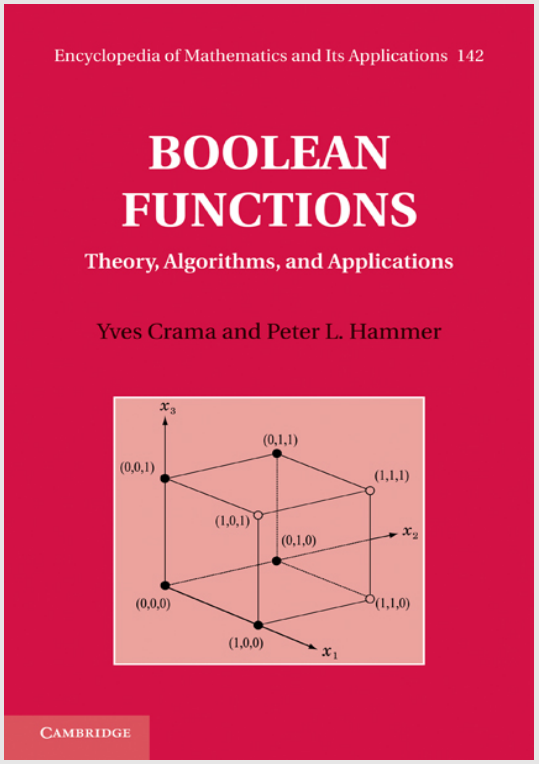
- Reliability: complex system operates (Yes or No) depending on the state of its elements (operating or failed).

Thousands of publications on related topics.

Two recent books

A monograph:

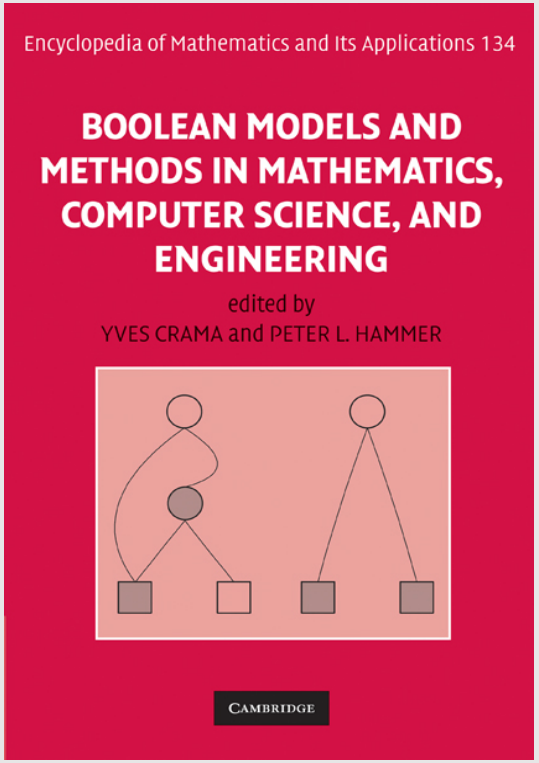
BOOLEAN FUNCTIONS
Theory, Algorithms, and Applications
Yves CRAMA and Peter L. HAMMER
Cambridge University Press, 2011, 710 pages (Available as e-Book)
with contributions by C. Benzaken, E. Boros, N. Brauner, M.C. Golumbic, V. Gurvich, L. Hellerstein, T. Ibaraki, A. Kogan, K. Makino, B. Simeone



Two recent books

A collection of surveys:

BOOLEAN MODELS AND METHODS
in Mathematics, Computer Science and Engineering
Yves CRAMA and Peter L. HAMMER, editors.
Cambridge University Press, 2010, 780 pages



Basic concepts

Attributes: $V = \{1, 2, ..., n\}$.

Boolean function: $f : \{0,1\}^V \longrightarrow \{0,1\}$.

True vectors of f: $T(f) = \{\mathbf{x} \in \{0,1\}^V \mid f(\mathbf{x}) = 1\}$.

False vectors of f: $F(f) = \{\mathbf{x} \in \{0,1\}^V \mid f(\mathbf{x}) = 0\}$.

$$T(f) \cap F(f) = \emptyset \quad \text{and} \quad T(f) \cup F(f) = \{0,1\}^V$$

Basic concepts

A **term** $\mathbf{t}$ is a Boolean function defined by an elementary conjunction **(AND)**

$$\mathbf{t}(\mathbf{x}) = \bigwedge_{j \in P} x_j \wedge \bigwedge_{j \in N} \overline{x}_j$$

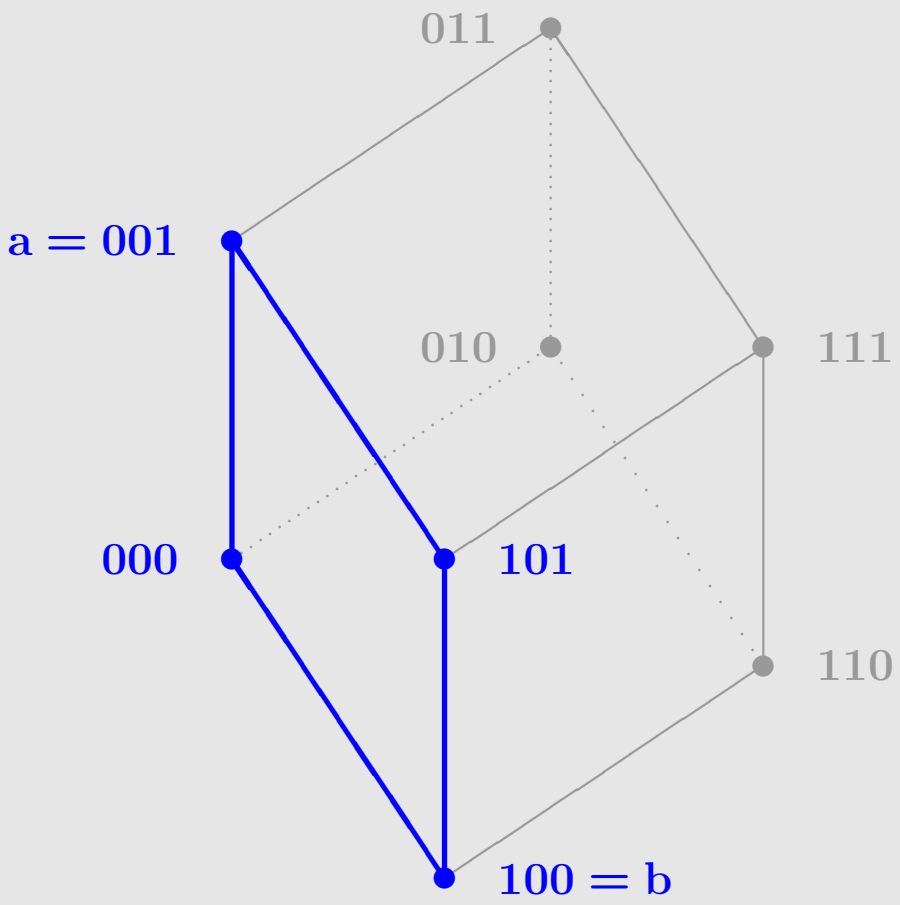
where $P, N \subseteq V$, and $\overline{x} = 1 - x$.

The conjunction takes value 1 (or "true") if and only if

$$x_j = 1 \text{ for all } j \in P \text{ and } x_j = 0 \text{ for all } j \in N.$$

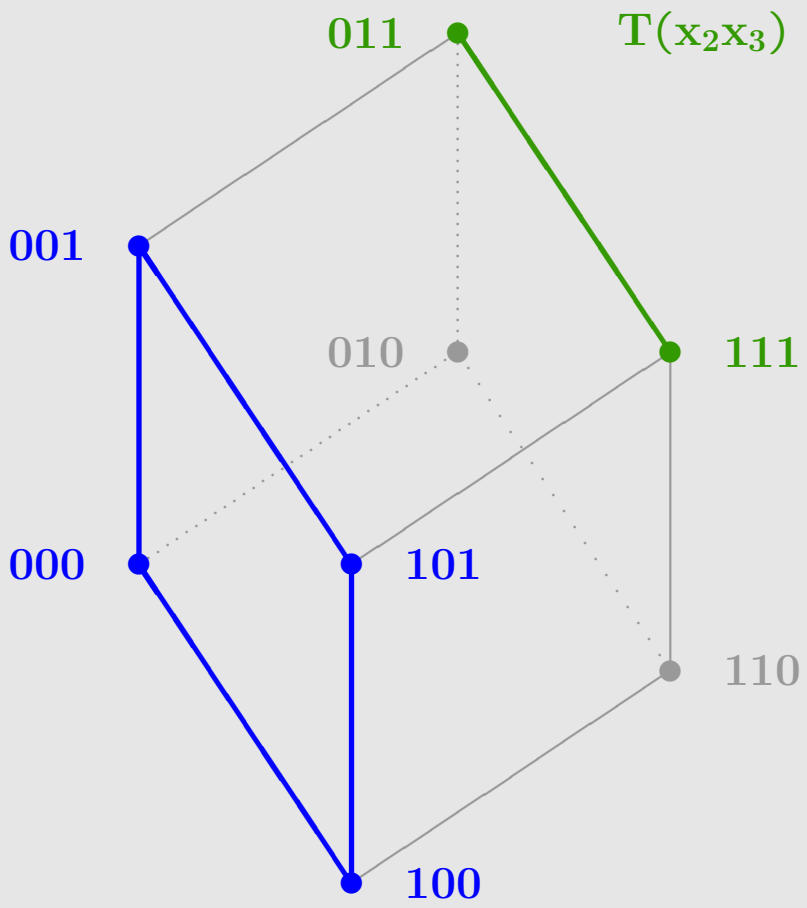
The set of true vectors of a term forms a **sub-cube** of $\{0,1\}^V$, and vice-versa, every sub-cube, is the set of true vectors of a Boolean function, defined by a unique term.

Sub-cubes and Terms in $\{0,1\}^3$



$$[a, b] = \{(000), (001), (100), (101)\} = (*0*) = T(\overline{x}_2)$$

Sub-cubes and Terms in $\{0,1\}^3$



$T(x_2x_3) = (*11) = [(011),(111)]$

Basic concepts

Every Boolean function can be represented as a **disjunctive normal form (DNF)**, that is, as a disjunction **(OR)** of terms (elementary conjunctions):

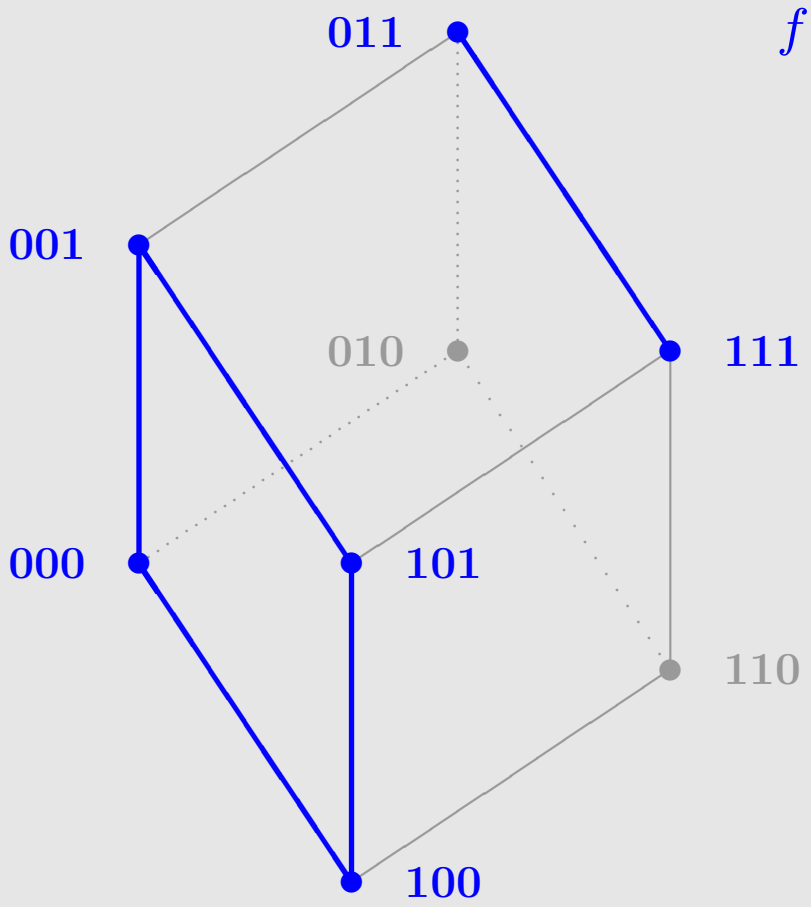
$$f(\mathbf{x}) = \bigvee_{(P,N)\in E} \Big(\bigwedge_{j\in P} x_j \wedge \bigwedge_{j\in N} \overline{x}_j \Big)$$

where $P, N \subseteq V$, and $\overline{x} = 1 - x$.

The DNF takes value 1 (or "true") if and only if at least one of its terms takes value 1.

Geometrically: the set of true vectors of f is covered by a union of subcubes of $\{0,1\}^V$.

Boolean functions as DNFs



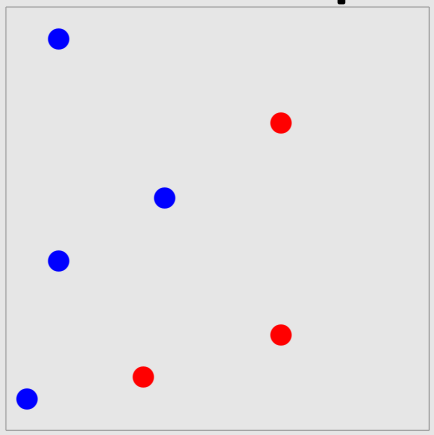
$f = \overline{x}_2 \vee x_2 x_3$

Outline

- Boolean Functions
- **Learning from Examples**
- Partially Defined Boolean Functions
- Logical Analysis of Data

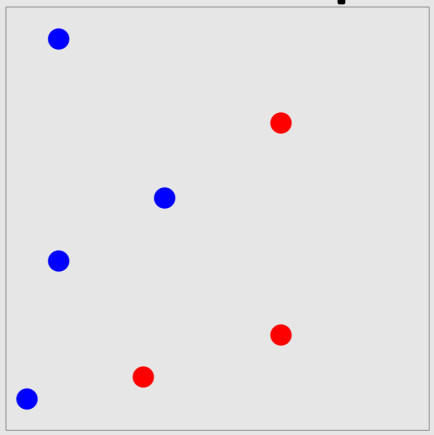
Process of Learning ...

Data: Examples



Process of Learning ...

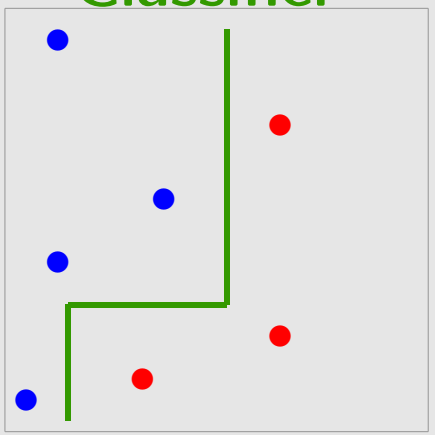
Data: Examples



Learning

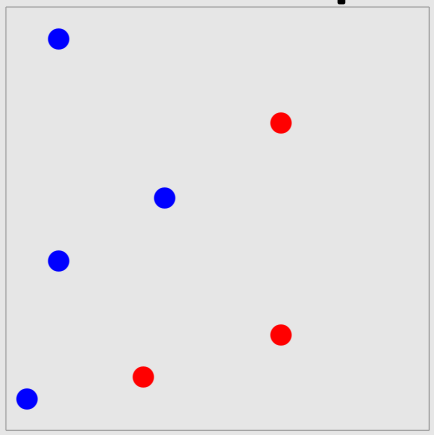
$\Rightarrow \cdots \Rightarrow$

Classifier



Process of Learning ...

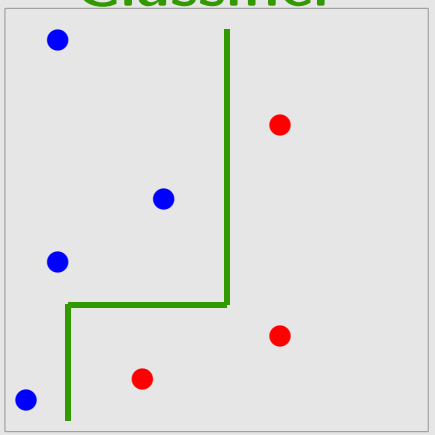
Data: Examples



Learning

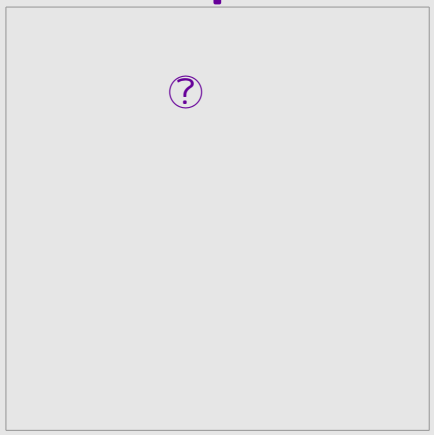
$\Rightarrow \cdots \Rightarrow$

Classifier



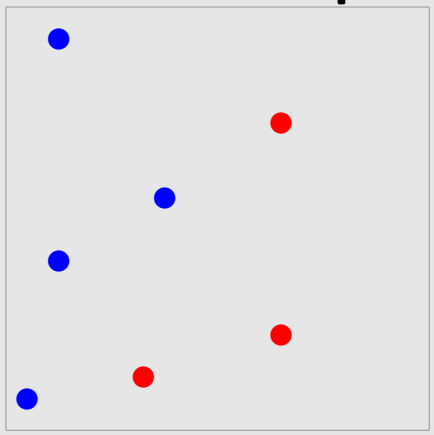
Testing:

Test point



Process of Learning ...

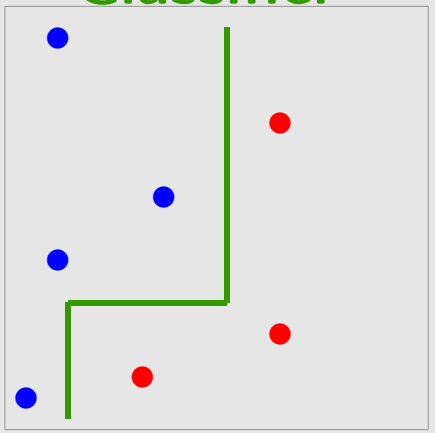
Data: Examples



Learning

$\Rightarrow \cdots \Rightarrow$

Classifier



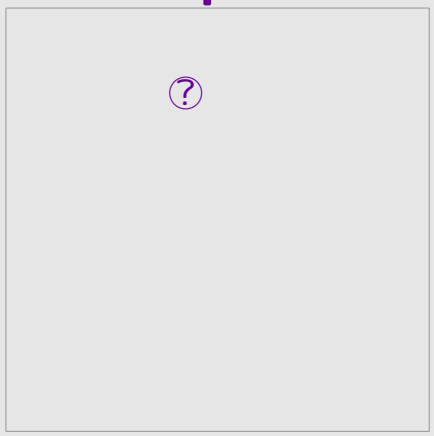
$\Downarrow$

$\vdots$

$\Downarrow$

Testing:

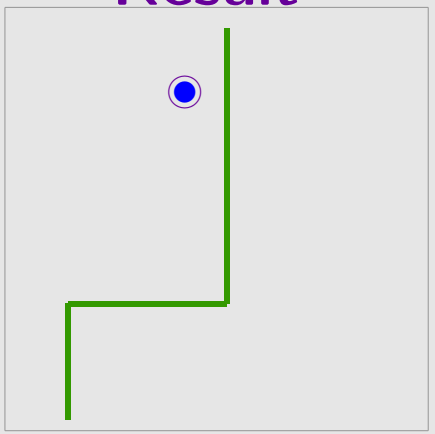
Test point:



Trial

$\Rightarrow \cdots \Rightarrow$

Result



Some typical examples

- **Credit approval.** Data: attributes of applicants for credit card vs. decision.

- **Customer targeting.** Data: attributes of customers vs. decision to buy.

- **Medical diagnosis.** Data: symptoms or bio-medical features vs. diagnosis.

WHAT CANNOT BE LEARNED FROM EXAMPLES!

"Just a darn minute! — Yesterday you said that X equals **two**!"

Data Sets and Classifiers

Attributes: $A, B, \ldots$ in domains A, B,...

Data Set: $\mathbf{D} = \left\{ \mathbf{X}^i = (X^i_A, X^i_B, \ldots) \mid i = 1, \ldots, M \right\}$

Class: $c : \mathbf{D} \longrightarrow \{C_1, \ldots, C_s\}$

Data Sets and Classifiers

Attributes: $A, B, ...$ in domains A, B,...

Data Set: $\mathbf{D} = \left\{ \mathbf{X}^i = (X^i_A, X^i_B, ...) \mid i = 1, ..., M \right\}$

Class: $c : \mathbf{D} \longmapsto \{C_1, \ldots, C_s\}$

Classifier: $f : \mathsf{A} \times \mathsf{B} \times \cdots \longmapsto \{C_1, \ldots, C_s\}$

Data Sets and Classifiers

Attributes: $A, B, \ldots$ in domains A, B,...

Data Set: $\mathbf{D} = \left\{ \mathbf{X}^i = (X^i_A, X^i_B, \ldots) \mid i = 1, \ldots, M \right\}$

Class: $c : \mathbf{D} \longmapsto \{C_1, \ldots, C_s\}$

Classifier: $f : \mathsf{A} \times \mathsf{B} \times \cdots \longmapsto \{C_1, \ldots, C_s\}$

We usually expect: $f(\mathbf{X}) = c(\mathbf{X})$ for all $\mathbf{X} \in \mathbf{D}$

Data Sets and Classifiers

Attributes: $A, B, ...$ in domains A, B,...

Data Set: $\mathbf{D} = \left\{ \mathbf{X}^i = (X^i_A, X^i_B, ...) \mid i = 1, ..., M \right\}$

Class: $c : \mathbf{D} \longmapsto \{C_1, \ldots, C_s\}$

Classifier: $f : \mathsf{A} \times \mathsf{B} \times \cdots \longmapsto \{C_1, \ldots, C_s\}$

We usually expect: $f(\mathbf{X}) = c(\mathbf{X})$ for all $\mathbf{X} \in \mathbf{D}$

There may be many classifiers for a same data set.

A small example

7 patients in the training set with 4 test results for each

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	1	1
	C	1	1	1	0
$\mathbb{F}$	T	0	0	1	1
	U	1	0	0	1
	V	1	0	1	0
	W	0	1	1	0

A small example

7 patients in the training set with 4 test results for each

	ID	Test Results				Dr. F
		x_1	x_2	x_3	x_4	$\mathbf{x_1 + x_2 + x_3 + x_4 \geq 3}$
$\mathbb{T}$	A	1	1	0	1	1
	B	0	1	1	1	1
	C	1	1	1	0	1
$\mathbb{F}$	T	0	0	1	1	0
	U	1	0	0	1	0
	V	1	0	1	0	0
	W	0	1	1	0	0

A small example

7 patients in the training set with 4 test results for each

	ID	Test Results				Dr. F	Dr. G
		x_1	x_2	x_3	x_4	$\mathbf{x_1 + x_2 + x_3 + x_4 \geq 3}$	$\mathbf{x_1 + 2x_2 + x_4 \geq 3}$
$\mathbb{T}$	A	1	1	0	1	1	1
	B	0	1	1	1	1	1
	C	1	1	1	0	1	1
$\mathbb{F}$	T	0	0	1	1	0	0
	U	1	0	0	1	0	0
	V	1	0	1	0	0	0
	W	0	1	1	0	0	0

A small example

7 patients in the training set with 4 test results for each

	ID	Test Results				Dr. F	Dr. G
		x_1	x_2	x_3	x_4	$x_1 + x_2 + x_3 + x_4 \geq 3$	$x_1 + 2x_2 + x_4 \geq 3$
$\mathbb{T}$	A	1	1	0	1	1	1
	B	0	1	1	1	1	1
	C	1	1	1	0	1	1
$\mathbb{F}$	T	0	0	1	1	0	0
	U	1	0	0	1	0	0
	V	1	0	1	0	0	0
	W	0	1	1	0	0	0
	Ms. Y	1	1	0	0		
	Mr. Z	1	0	1	1		

A small example

7 patients in the training set with 4 test results for each

	ID	Test Results				Dr. F	Dr. G
		x_1	x_2	x_3	x_4	$x_1 + x_2 + x_3 + x_4 \geq 3$	$x_1 + 2x_2 + x_4 \geq 3$
$\mathbb{T}$	A	1	1	0	1	1	1
	B	0	1	1	1	1	1
	C	1	1	1	0	1	1
$\mathbb{F}$	T	0	0	1	1	0	0
	U	1	0	0	1	0	0
	V	1	0	1	0	0	0
	W	0	1	1	0	0	0
	Ms. Y	1	1	0	0	**0**	**1**
	Mr. Z	1	0	1	1	**1**	**0**

Outline

- Boolean Functions
- Learning from Examples
- **Partially Defined Boolean Functions**
- Logical Analysis of Data

Partially Defined Boolean Functions

- **Definition**
- Support sets
- Patterns
- Theories

Definitions

Training Data: a pair of subsets $(\mathbf{T}, \mathbf{F})$ such that

$$\mathbf{T} \subseteq \{0,1\}^V, \quad \mathbf{F} \subseteq \{0,1\}^V, \quad \text{and} \quad \mathbf{T} \cap \mathbf{F} = \emptyset.$$

We call such a pair $(\mathbf{T}, \mathbf{F})$ a *partially defined Boolean function* (or **pdBf** in short).

Classifier: a Boolean function $f : \{0,1\}^V \longrightarrow \{0,1\}$, which is an **extension** of $(\mathbf{T}, \mathbf{F})$, i.e., for which

$$\mathbf{T} \subseteq T(f) \quad \text{and} \quad \mathbf{F} \subseteq F(f).$$

Let $\mathcal{E}(\mathbf{T}, \mathbf{F})$ denote the family of all extensions. We have

$$|\mathcal{E}(\mathbf{T}, \mathbf{F})| = 2^{2^n - |\mathbf{T} \cup \mathbf{F}|}$$

What can guide learning?

If $|V| = 20$ and $|(\mathbf{T}, \mathbf{F})| = 1000$, then

$$|\mathcal{E}(\mathbf{T}, \mathbf{F})| > \mathbf{2}^{\mathbf{1,000,000}}$$

What can guide learning?

If $|V| = 20$ and $|(\mathbf{T}, \mathbf{F})| = 1000$, then

$$|\mathcal{E}(\mathbf{T}, \mathbf{F})| > \mathbf{2}^{\mathbf{1,000,000}}$$

- Simplicity
  - Essential attributes
  - Efficient computation (DNF, CNF, decision tree, etc.)

What can guide learning?

If $|V| = 20$ and $|(\mathbf{T}, \mathbf{F})| = 1000$, then

$$|\mathcal{E}(\mathbf{T}, \mathbf{F})| > \mathbf{2}^{\mathbf{1,000,000}}$$

- Simplicity
  - Essential attributes
  - Efficient computation (DNF, CNF, decision tree, etc.)
- Interpretability
- Justifiability

Note on framework: we focus here on *unspecified models*, as opposed to *specified models* such as regression models (which assume prior knowledge about the relation between inputs and outputs).

Building reasonable extensions

Given $(\mathbf{T}, \mathbf{F})$, how can we build a reasonable extension $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$?

Many ways....

For example, **nearest neighbor** methods, **decision trees**, or **neural networks** build such classifiers.

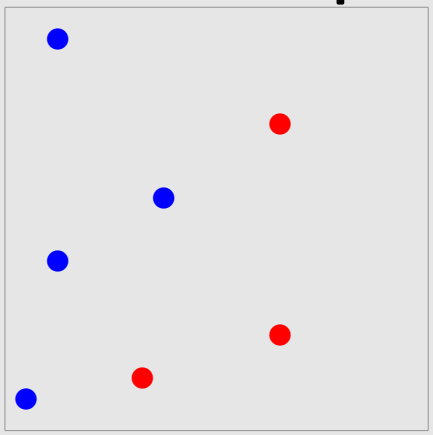
Nearest Neighbor classifiers

Define a notion of **distance** $\rho(X, Y)$ between any two points X, Y in the input space.

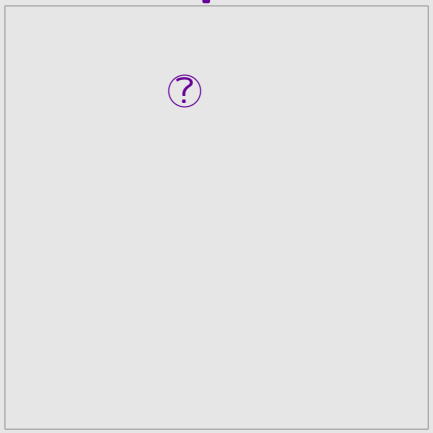
Nearest Neighbor classifiers

Define a notion of **distance** $\rho(X, Y)$ between any two points X, Y in the input space.

Data: Examples



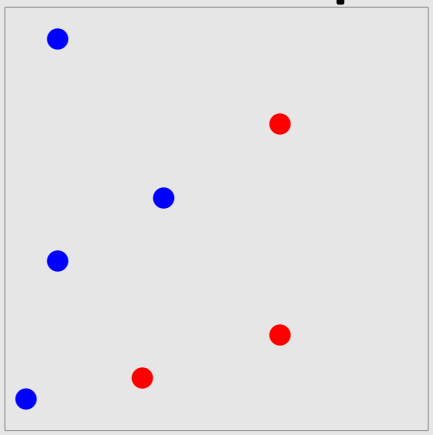
Test point:



Nearest Neighbor classifiers

Define a notion of **distance** $\rho(X, Y)$ between any two points X, Y in the input space.

Data: Examples

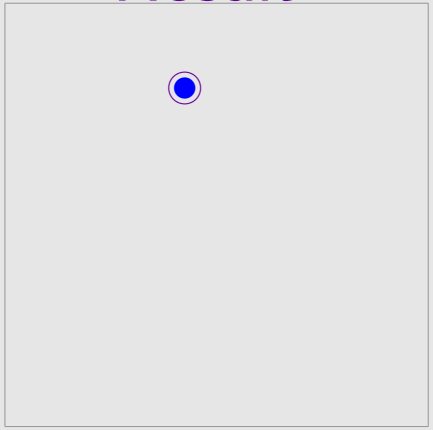


Test point:

Closest example

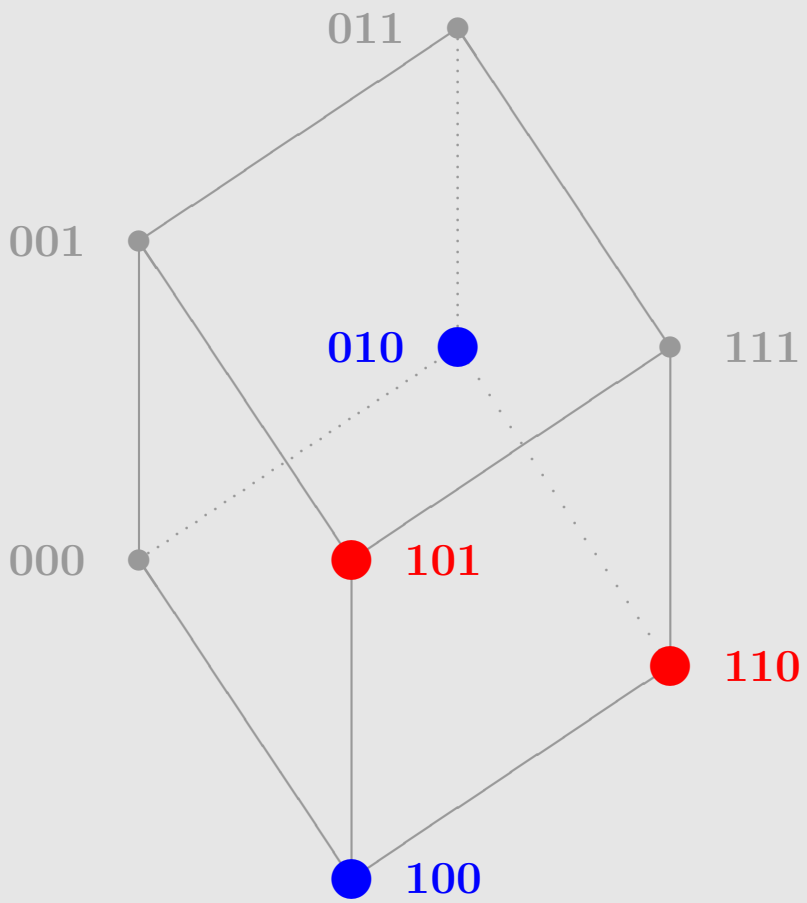
$\Rightarrow \cdots \Rightarrow$

Result



Nearest Neighbor classifiers

Example in the Boolean case.

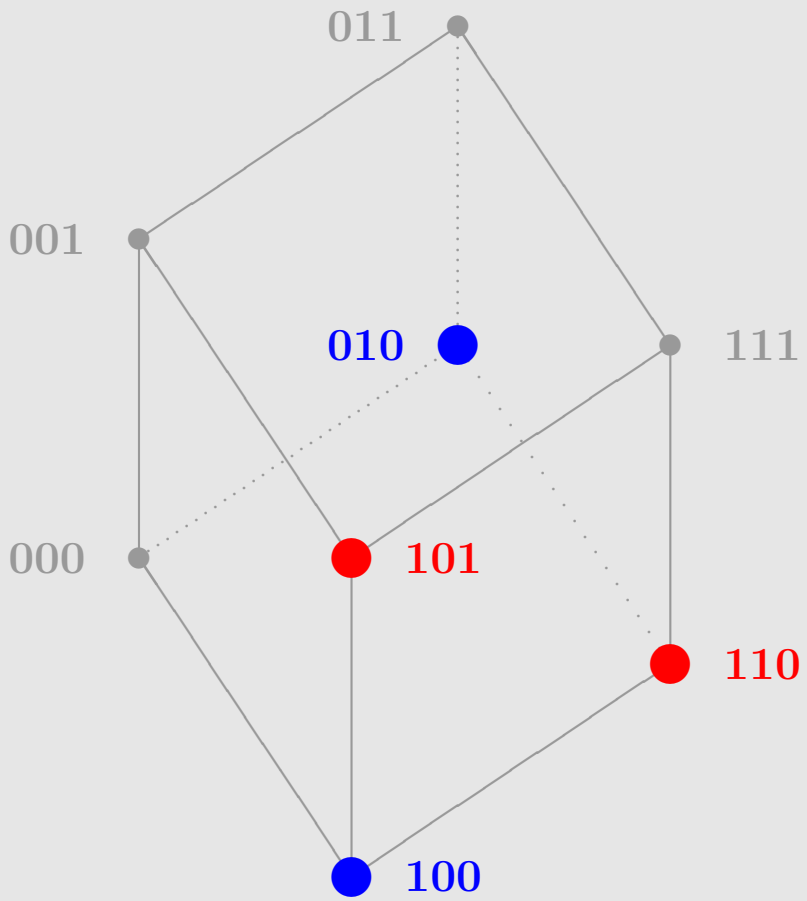


$\mathbf{F} = \{(110), (101)\}$

$\mathbf{T} = \{(010), (100)\}$

Nearest Neighbor classifiers

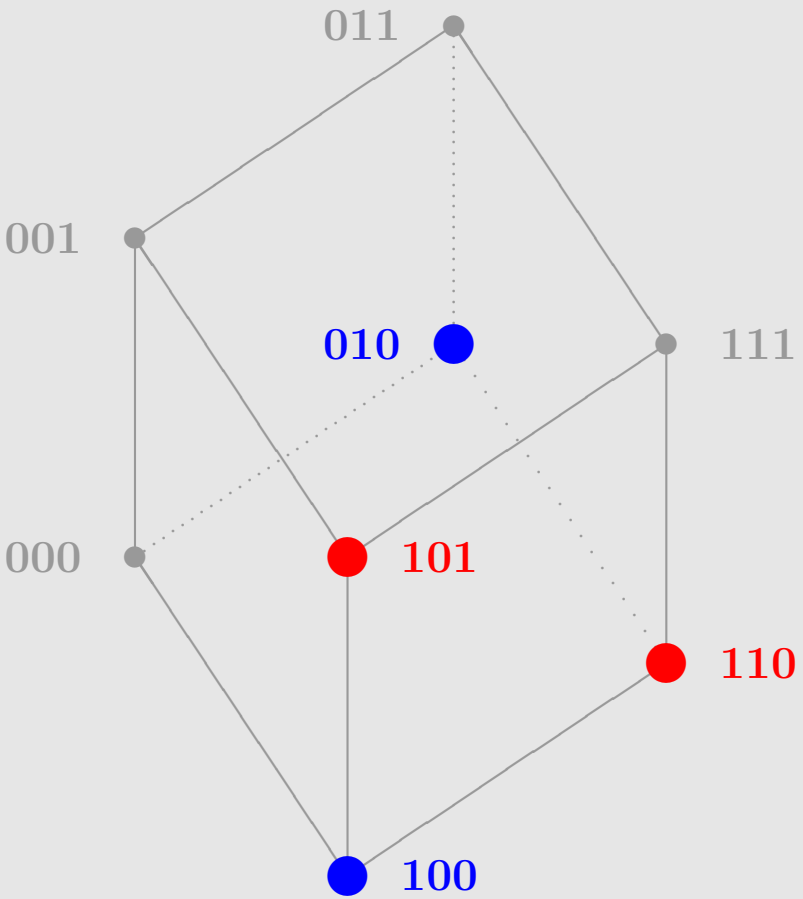
Example in the Boolean case.



(111) is classified as red (false)

(000) is classified as blue (true)

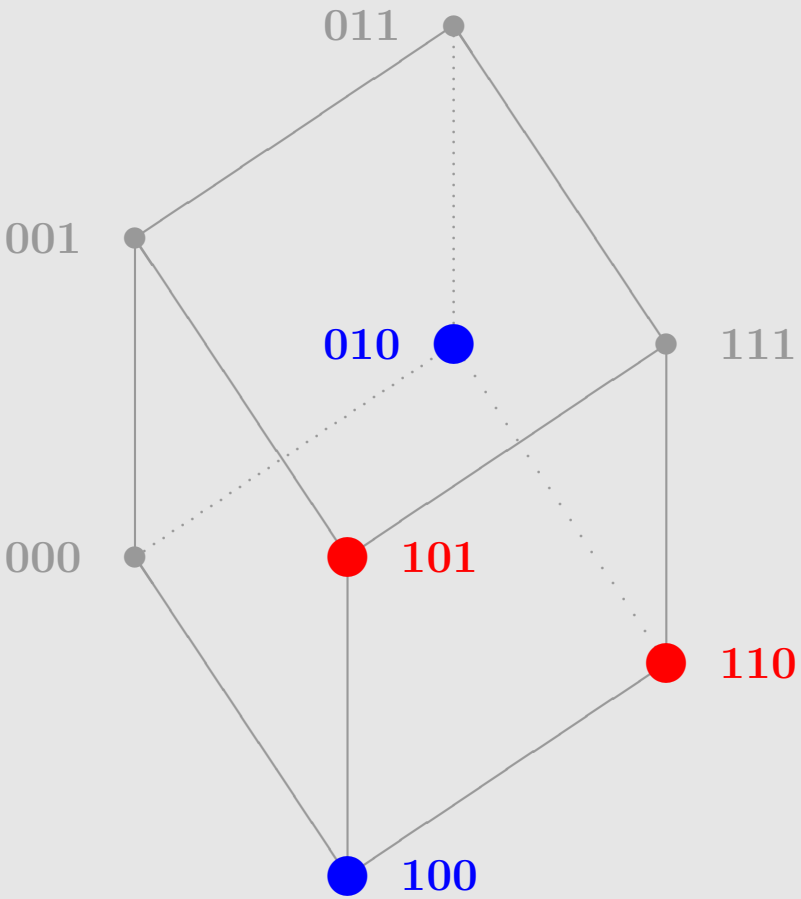
Decision Trees for pdBfs

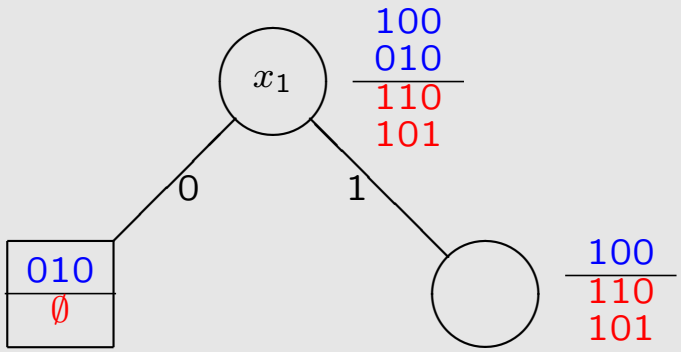


$F = \{(110), (101)\}$

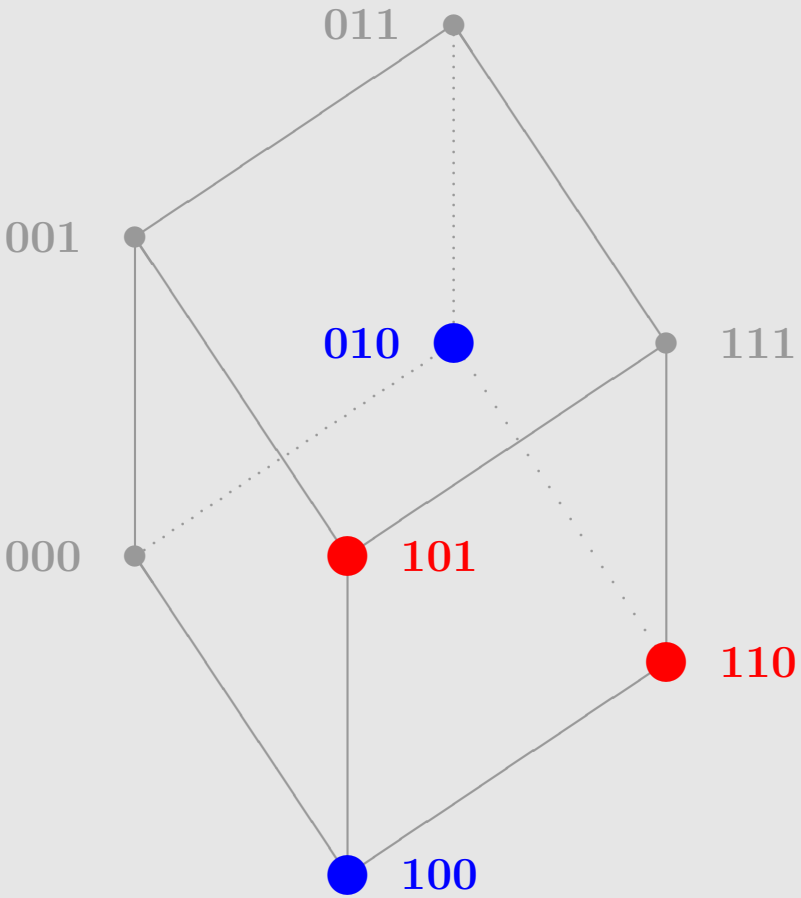
$T = \{(010), (100)\}$

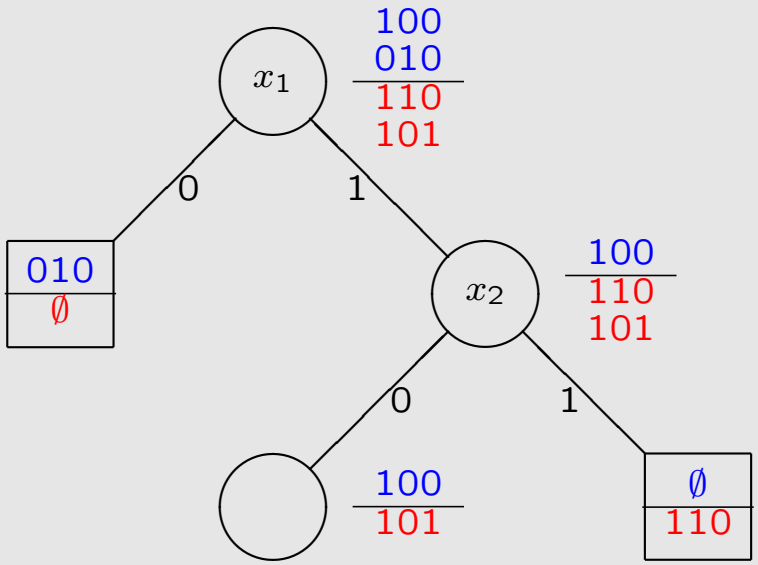
Decision Trees for pdBfs



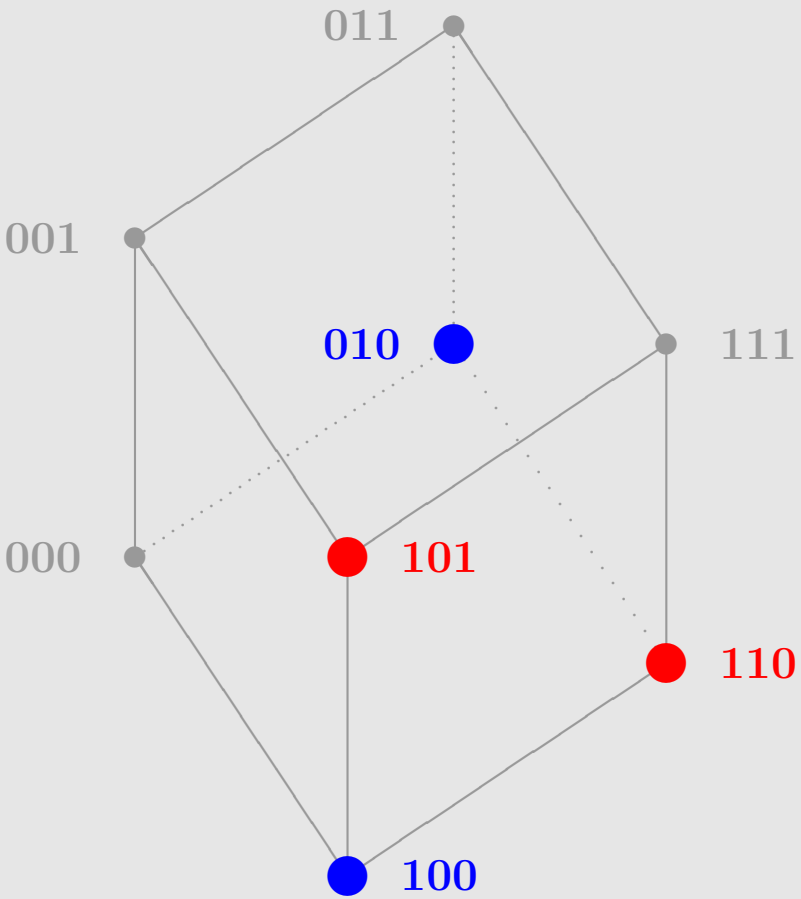


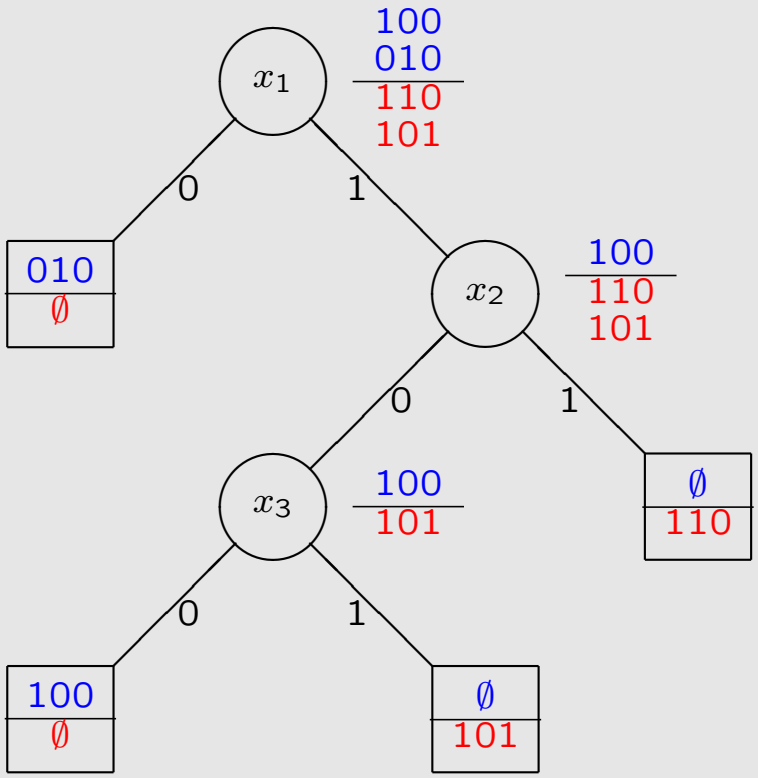
Decision Trees for pdBfs



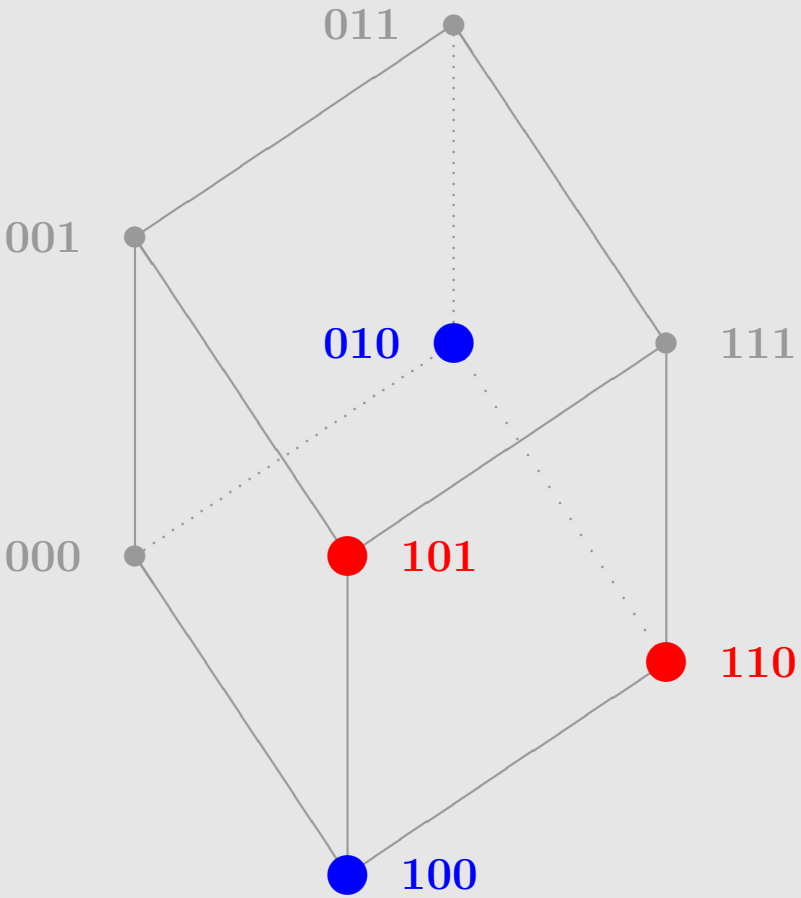


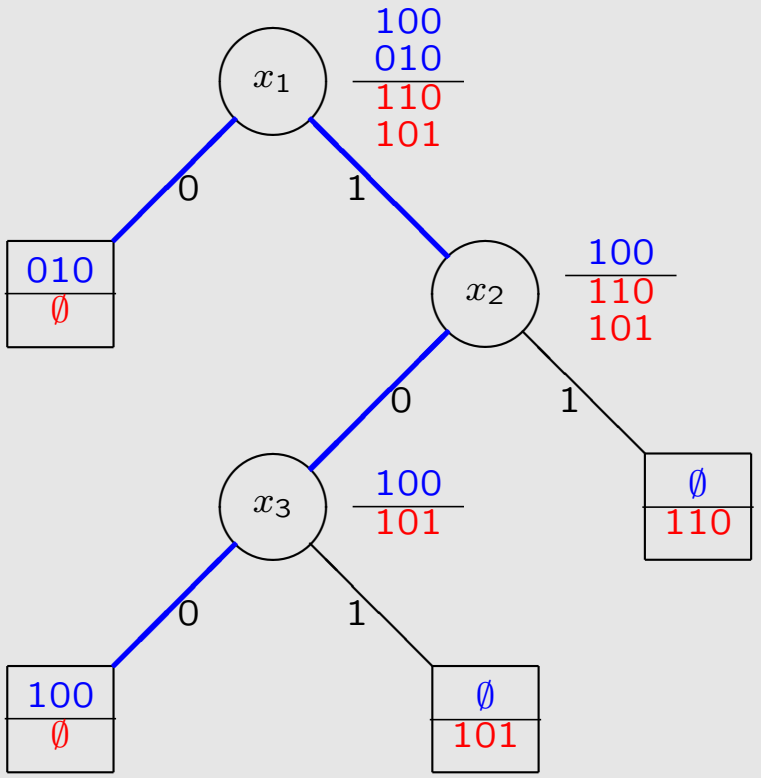
Decision Trees for pdBfs





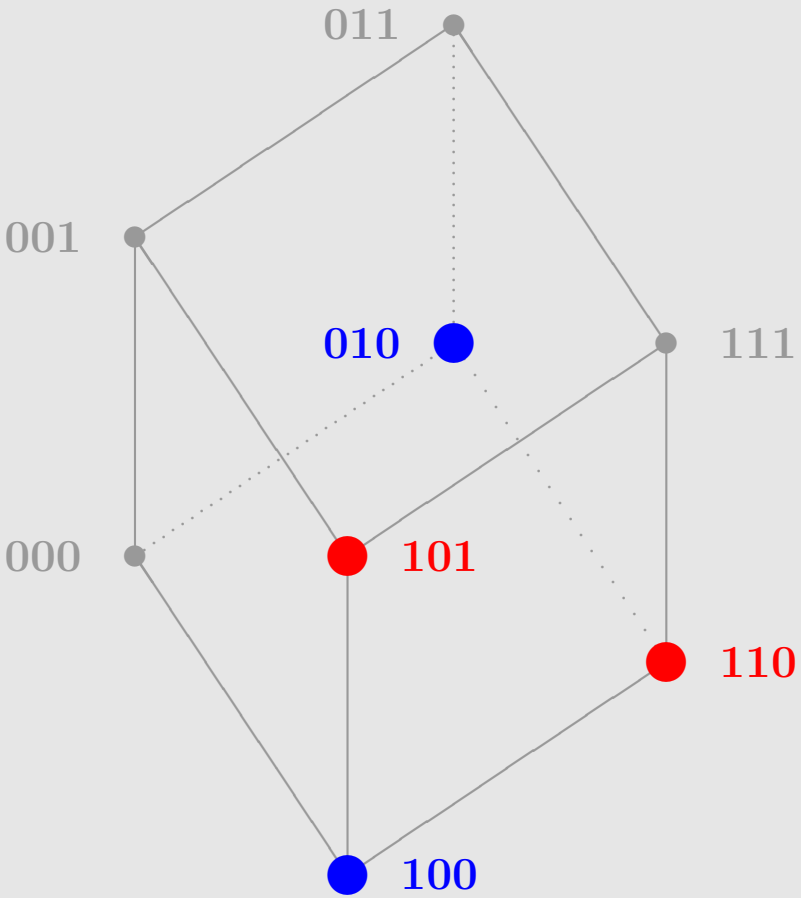
Decision Trees for pdBfs

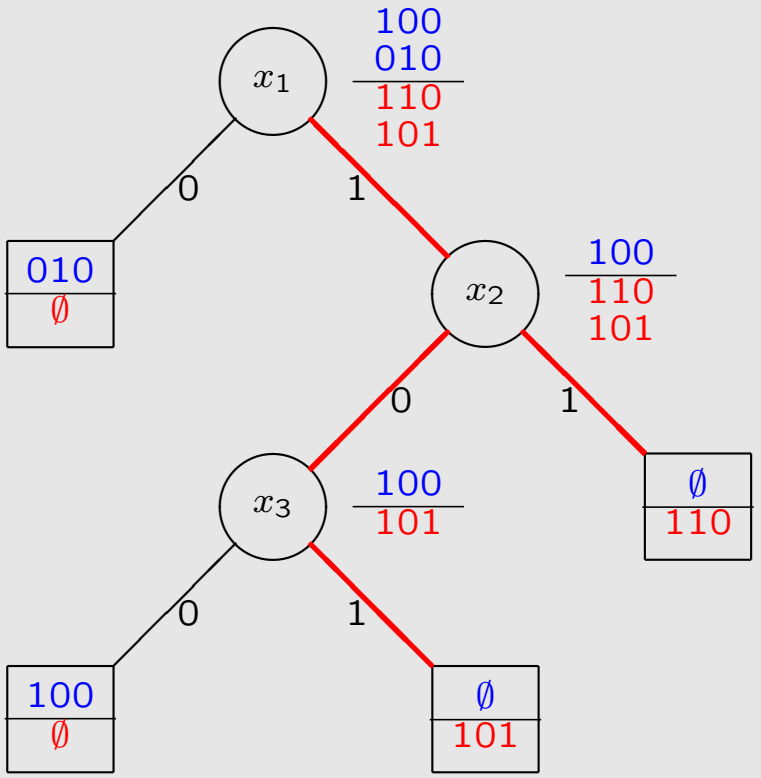




$$f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3$$

Decision Trees for pdBfs

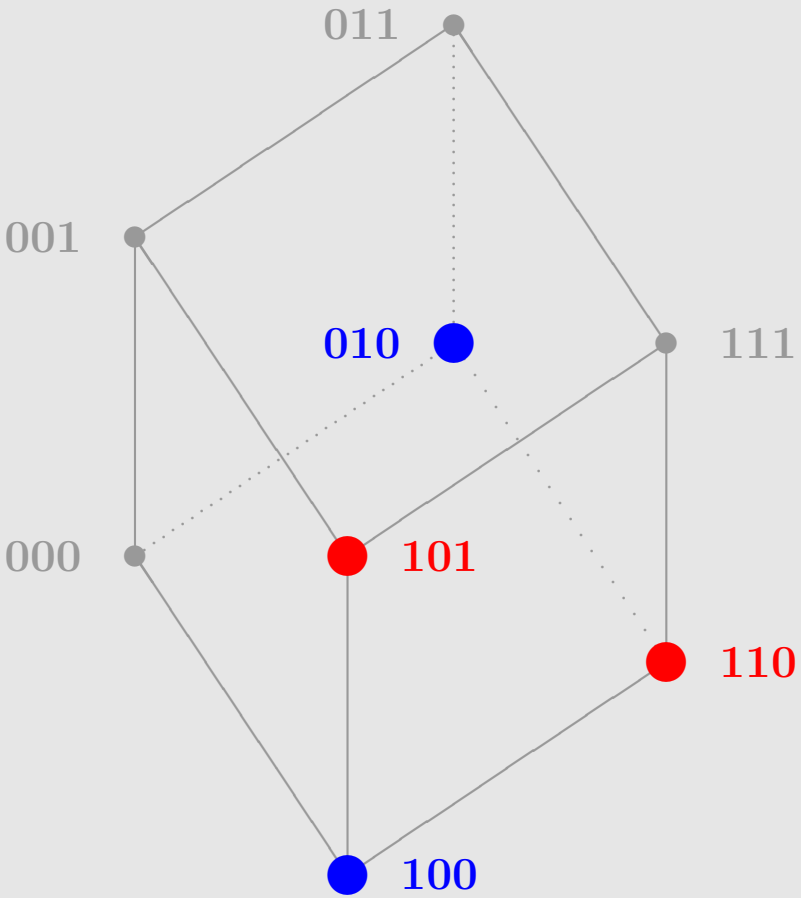


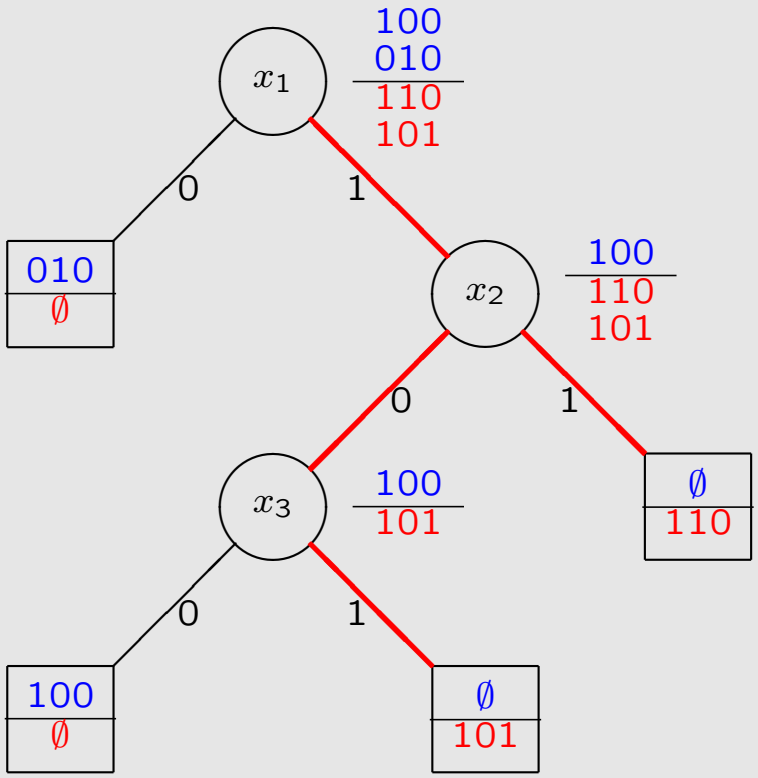


$$f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3$$

$$\overline{f}_D = x_1x_2 \vee x_1\overline{x}_2x_3 = x_1x_2 \vee x_1x_3$$

Decision Trees for pdBfs



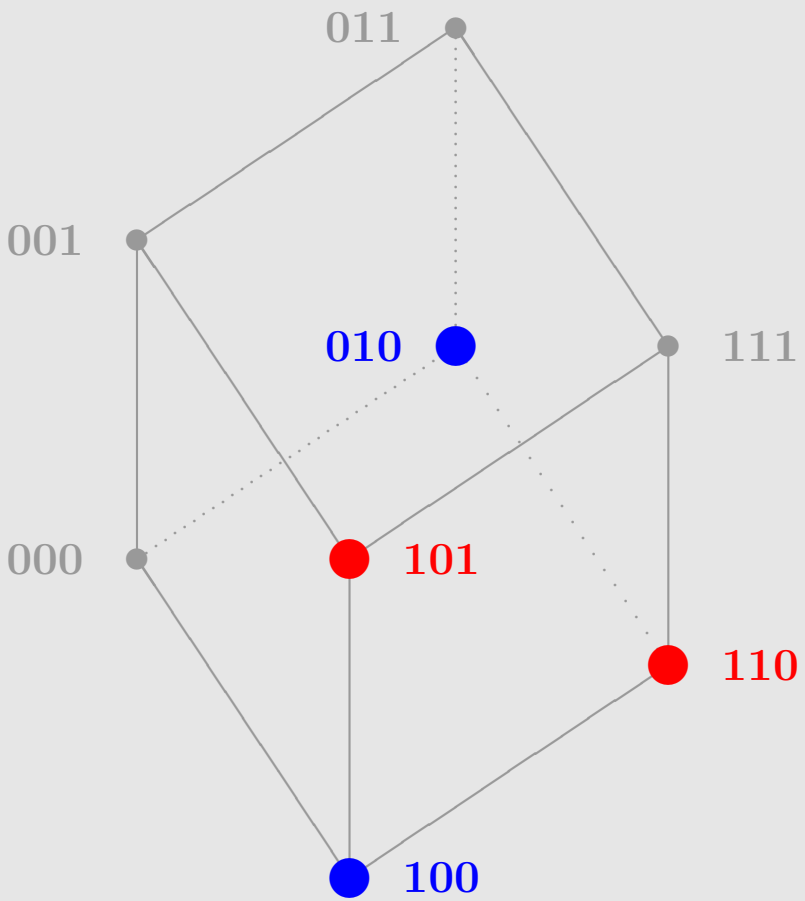


$$f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3$$

$$\overline{f}_D = x_1x_2 \vee x_1\overline{x}_2x_3 = x_1x_2 \vee x_1x_3$$

Note: (001) is classified differently by NN and by DT

Linear separator



$\mathbf{F} = \{(110), (101)\}$

$\mathbf{T} = \{(010), (100)\}$

Decide whether $\mathbf{T}$ and $\mathbf{F}$ can be separated by a hyperplane.

This is a simple linear programming problem.

Similar to recognizing a weighted majority game.

Outline

- Boolean Functions
- Learning from Examples
- Partially Defined Boolean Functions
- **Logical Analysis of Data**

Logical Analysis of Data

LAD: Introduced in Crama, Hammer and Ibaraki (1988).

More than a well-circumscribed mathematical theory, rather:

- a way of looking at classification problems through the Boolean prism,
- a collection of concepts, algorithms, and structural properties.

Based on the representation of extensions by DNFs and on selection of

- subsets of relevant variables (support sets)
- relevant terms (patterns)
- relevant disjunctions of terms (theories)

Similarities with other models used in data mining, concept learning, qualitative analysis, etc.

Partially Defined Boolean Functions

- Definition
- **Support sets**
- Patterns
- Theories

Finding Essential Attributes

- Select relevant features.

- Compress data.

Relevance and its evaluations

- Well defined for *complete* systems: an attribute is relevant, if changing its value changes the classification of some situations.

- Measures of relevance are based on counting such situations (with slight variations, e.g., *coalitions' power* in game theory; *voters' influence* in voting schemes, etc., Shapley (1954), Chow (1961), Banzhaf (1965), Winder (1971), Kahn, Kalai and Linial (1988), Hammer, Kogan and Rohtblum (2000))

- These definitions cannot be easily extended to incomplete data sets in a consistent way, see e.g., John, Kohavi and Pfleger (1994).

Data compression

- The simpler, the better! – "Occam's Razor:"
  Theories built on smaller attribute sets, generalize better.
  Blumer,Ehrenfeucht,Haussler and Warmuth (1987)

- Decreases the computational complexity of finding and using a classifier.

- Decreases the cost of future data collection.

Feature selection based on separating power

Find a small (smallest, if possible) subset of the attributes which distinguishes the sets $\mathbb{T}$ and $\mathbb{F}$. Such a subset is called a **support set**.

Crama, Hammer and Ibaraki (1988).

Feature selection based on separating power

Find a small (smallest, if possible) subset of the attributes which distinguishes the sets $\mathbb{T}$ and $\mathbb{F}$. Such a subset is called a **support set**.

Crama, Hammer and Ibaraki (1988).

$\mathbb{T}$	1	0	0	1	1	0	0	0	0	
	1	1	1	1	0	0	1	0	0	
	0	1	1	0	1	1	1	0	0	
	1	0	1	0	0	1	1	1	1	
$\mathbb{F}$	0	0	1	1	1	1	0	0	1	
	0	1	0	1	1	1	1	1	1	
	0	0	0	0	1	1	0	1	0	
	1	1	0	0	0	0	0	1	0	

Feature selection based on separating power

Find a small (smallest, if possible) subset of the attributes which distinguishes the sets $\mathbb{T}$ and $\mathbb{F}$. Such a subset is called a **support set**.

Crama, Hammer and Ibaraki (1988).

$\mathbb{T}$	**1**	0	0	**1**	1	0	**0**	0	0
	1	1	1	**1**	0	0	**1**	0	0
	0	1	1	**0**	1	1	**1**	0	0
	1	0	1	**0**	0	1	**1**	1	1
$\mathbb{F}$	**0**	0	1	**1**	1	1	**0**	0	1
	0	1	0	**1**	1	1	**1**	1	1
	0	0	0	**0**	1	1	**0**	1	0
	1	1	0	**0**	0	0	**0**	1	0

Feature selection based on separating power

Finding a smallest support set is NP-hard.

Algorithmic Approaches to find a small(est) support set:

- complete enumeration: FOCUS (Almuallim and Dietterich, 1994) ...
- greedy search: Rel-FSS (Bell and Wang, 2000) ...
- computing relevance index: (Kira and Rendell, 1992) ...
- etc., ... *over 40 references in the past decade.*

Note that decision trees automatically select a (small) support set.

Feature selection based on separating power

A set covering model to find a small(est) support set:

- associate a 0-1 variable a_i with each attribute A_i
- for every pair of false example X and true example Y, express that at least one of the attributes differentiating X from Y must be chosen:

$$\text{for all } X \in \mathbb{F}, Y \in \mathbb{T}, \quad \sum_{i:x_i \neq y_i} a_i \geq 1$$

- minimize $\sum_i a_i$.

This model can be solved either exactly, or heuristically.

Feature selection based on separating power

Questions to clarify

Why a small(est) feature set??

Which one??

How to measure the quality of a support set?

Why a small(est) feature set??

Which one??

- Typically, there are many support sets of different sizes.

Why a small(est) feature set??

Which one??

- Typically, there are many support sets of different sizes.

- The larger the data set, the less likely to have a small support set. (Tested in experiments by Boros, Horiyama, Ibaraki, Makino and Yagiura, 2003 on random data sets.)

Why a small(est) feature set??

Which one??

- Typically, there are many support sets of different sizes.
- The larger the data set, the less likely to have a small support set. (Tested in experiments by Boros, Horiyama, Ibaraki, Makino and Yagiura, 2003 on random data sets.)
- The larger the data set, the more surprising to have a small support set.

Why a small(est) feature set??

Which one??

- Typically, there are many support sets of different sizes.
- The larger the data set, the less likely to have a small support set. (Tested in experiments by Boros, Horiyama, Ibaraki, Makino and Yagiura, 2003 on random data sets.)
- The larger the data set, the more surprising to have a small support set.
- In practice, real-world data set often have small support sets, which can be expected to be meaningful.

Partially Defined Boolean Functions

- Definition
- Support sets
- **Patterns**
- Theories

Definitions

Remember: A **term** $\mathbf{t}$ is a Boolean function defined by an elementary conjunction **(AND)**

$$\mathbf{t}(\mathbf{x}) = \bigwedge_{j \in P} x_j \wedge \bigwedge_{j \in N} \overline{x}_j.$$

A term $\mathbf{t}$ is a **pattern** of $(\mathbf{T}, \mathbf{F})$ if

$$\mathbf{T} \cap T(\mathbf{t}) \neq \emptyset \quad \text{and} \quad \mathbf{F} \subseteq F(\mathbf{t}),$$

or

$$\mathbf{t}(\mathbf{x}) = 1 \text{ for at least one } \mathbf{x} \in \mathbf{T} \quad \text{and} \quad \mathbf{t}(\mathbf{x}) = 0 \text{ for all } \mathbf{x} \in \mathbf{F}.$$

A pattern corresponds to a combination of attributes which has been observed **at least once in a true point**, but **never in a false point**.

A pattern of $(\mathbf{F}, \mathbf{T})$ is called a **co-pattern** of $(\mathbf{T}, \mathbf{F})$.

Pat$(\mathbf{T}, \mathbf{F})$ and co-Pat$(\mathbf{T}, \mathbf{F})$ denote the families of all patterns and co-patterns of $(\mathbf{T}, \mathbf{F})$, respectively.

Returning to the medical example

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	1	1
	C	1	1	1	0
$\mathbb{F}$	T	0	0	1	1
	U	1	0	0	1
	V	1	0	1	0
	W	0	1	1	0

Some patterns:

$$x_1x_2,\ x_2\overline{x}_3,\ x_2x_4,\ldots$$

Some co-patterns:

$$\overline{x}_1\overline{x}_2,\ \overline{x}_2,\ \overline{x}_1\overline{x}_4,\ldots$$

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Coverage: $cov(\mathbf{P}) = 2 \quad > \quad 0$

number of positive examples covered

Precision: $\pi(\mathbf{P}) = 2/7 \quad > \quad 0$

fraction of data correctly classified

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Coverage: $cov(\mathbf{P}) = 2 \quad > \quad 0$

number of positive examples covered

Precision: $\pi(\mathbf{P}) = 2/7 \quad > \quad 0$

fraction of data correctly classified

Generating efficiently **all** patterns is possible (*in total time*).

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Coverage: $cov(\mathbf{P}) = 2 \quad > \quad 0$

number of positive examples covered

Precision: $\pi(\mathbf{P}) = 2/7 \quad > \quad 0$

fraction of data correctly classified

Generating efficiently **all** patterns is possible (*in total time*).

But there are too many!

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Coverage: $cov(\mathbf{P}) = 2 \quad > \quad 0$

number of positive examples covered

Precision: $\pi(\mathbf{P}) = 2/7 \quad > \quad 0$

fraction of data correctly classified

Ideally, we would like to generate **all** patterns with high coverage:

$$\mathcal{P}(\mathbf{T}, \mathbf{F}, \gamma) = \{\mathbf{P} \mid cov(\mathbf{P}) \geq \gamma |\mathbf{T}|\}$$

Pattern Generation

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	0	1
	C	1	0	0	0
$\mathbb{F}$	T	0	0	0	1
	U	1	0	1	1
	V	1	1	0	0
	W	0	1	0	0

Pattern: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

$\mathbf{P}(\mathbf{b}) = 0 \quad \forall\ \mathbf{b} \in \mathbb{F}$

Coverage: $cov(\mathbf{P}) = 2 \quad > \quad 0$

number of positive examples covered

Precision: $\pi(\mathbf{P}) = 2/7 \quad > \quad 0$

fraction of data correctly classified

Ideally, we would like to generate **all** patterns with high coverage:

$$\mathcal{P}(\mathbf{T}, \mathbf{F}, \gamma) = \{\mathbf{P} \mid cov(\mathbf{P}) \geq \gamma |\mathbf{T}|\}$$

NP-hard!

Patterns

In practice, generation heuristics concentrate for instance on patterns of small degree, high coverage, high precision.

There is considerable empirical evidence that patterns with high precision on a training (data) set **generalize** well, in the sense that they provide classifiers with high precision on subsequent test sets.

In fact, in many cases, even **a single pattern** can be used as a good classifier.

Mushroom Database

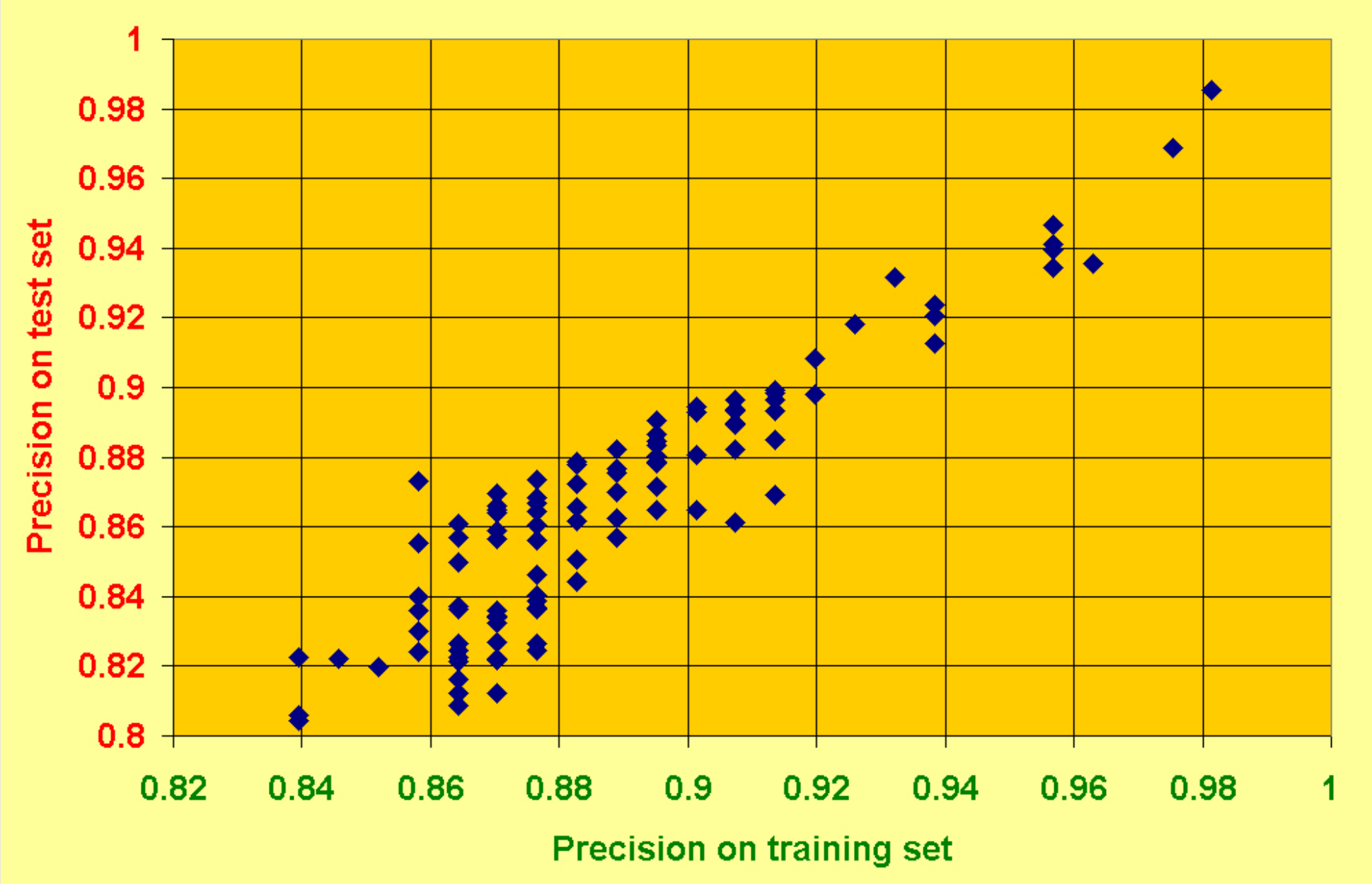
http://www.ics.uci.edu/ mlearn/MLRepository.html

- Number of Instances: 8124 (status of April 27, 1987)
- Number of Attributes: 22 with nominal values (126 categories)
- Missing attribute values: 2480, all for attribute stalk-root.
- Class distribution:
  - edible: **4208** (51.8%)
  - poisonous: **3916** (48.2%)

Mushroom Database

- Training set: **161** records ($\approx 2\%$)
- Attributes: **56** binary
- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.85$: **218** (of degrees $2-9$)

Mushroom Database



Mushroom Database

- Training set: **161** records ($\approx 2\%$)
- Attributes: **56** binary
- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.85$: **218** (of degrees $2-9$)
- Single **best** pattern: 98.5%-**classifier**!!

$$\mathbf{P}(\mathbf{X}) = (\text{Odor} \neq none) \wedge (\text{Odor} \neq anise) \wedge (\text{Odor} \neq almond)$$

- Best results reported in literature: $95-99\%$

Partially Defined Boolean Functions

- Definition
- Support sets
- Patterns
- **Theories**

Theories and Co-Theories

An extension $f \in \mathcal{E}(\mathbf{T}, \mathbf{F})$ is called a **theory** of $(\mathbf{T}, \mathbf{F})$ if it can be represented as a disjunction of some of the patterns of $(\mathbf{T}, \mathbf{F})$: it is a disjunction of patterns which cover all true points of $(\mathbf{T}, \mathbf{F})$.

A theory g of $(\mathbf{F}, \mathbf{T})$ is called a **co-theory** of $(\mathbf{T}, \mathbf{F})$: it is a disjunction of co-patterns which cover all false points of $(\mathbf{T}, \mathbf{F})$).

Denote by $\mathcal{E}_T(\mathbf{T}, \mathbf{F})$ and $\mathcal{E}_T(\mathbf{F}, \mathbf{T})$ the families of theories and co-theories of a given pdBf $(\mathbf{T}, \mathbf{F})$.

Typically we have

$$|\mathcal{E}_T(\mathbf{T}, \mathbf{F})| \quad << \quad |\mathcal{E}(\mathbf{T}, \mathbf{F})|$$

Examples

	ID	Test Results			
		x_1	x_2	x_3	x_4
$\mathbb{T}$	A	1	1	0	1
	B	0	1	1	1
	C	1	1	1	0
$\mathbb{F}$	T	0	0	1	1
	U	1	0	0	1
	V	1	0	1	0
	W	0	1	1	0

Some patterns: x_1x_2, $x_2\overline{x}_3$, $x_2x_4, \ldots$

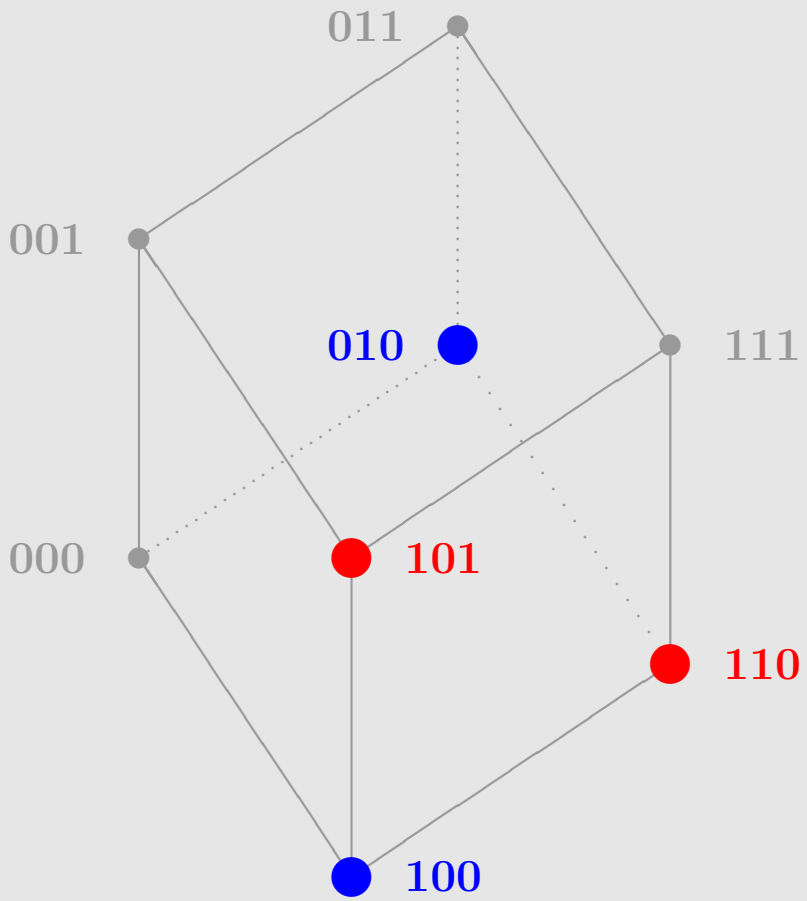
A theory: $x_1x_2 \vee x_2x_4$

Some co-patterns: $\overline{x}_1\overline{x}_2$, $\overline{x}_2$, $\overline{x}_1\overline{x}_4, \ldots$

A co-theory: $\overline{x}_1\overline{x}_2 \vee \overline{x}_1\overline{x}_4$

Examples

Nearest Neighbor classifier



(111) is classified as red (false)

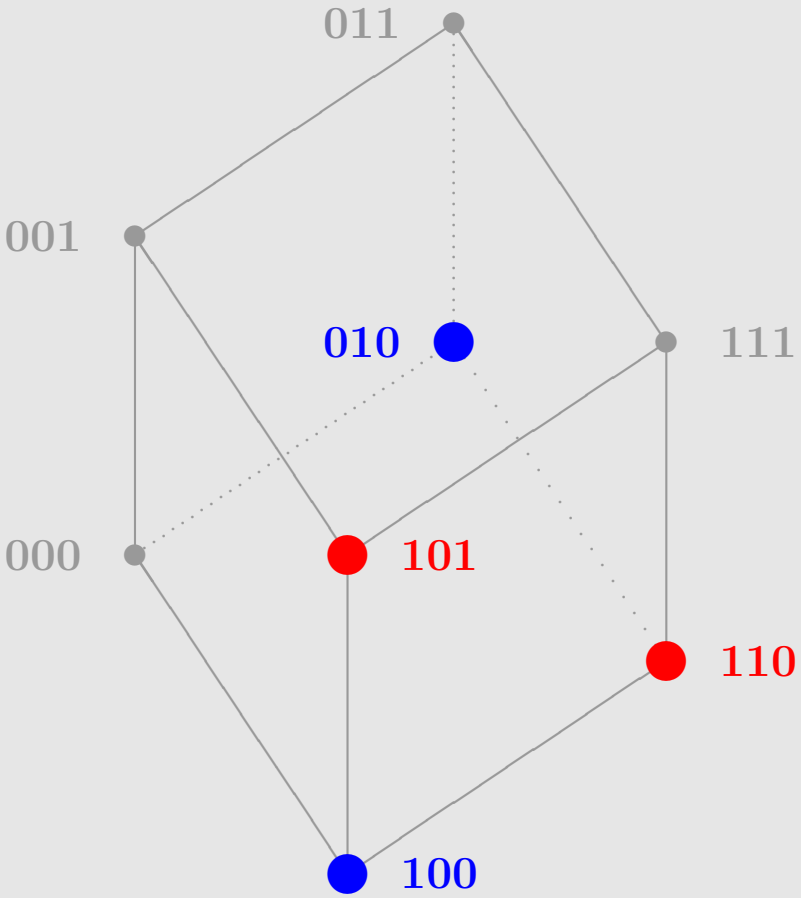
(000) is classified as blue (true)

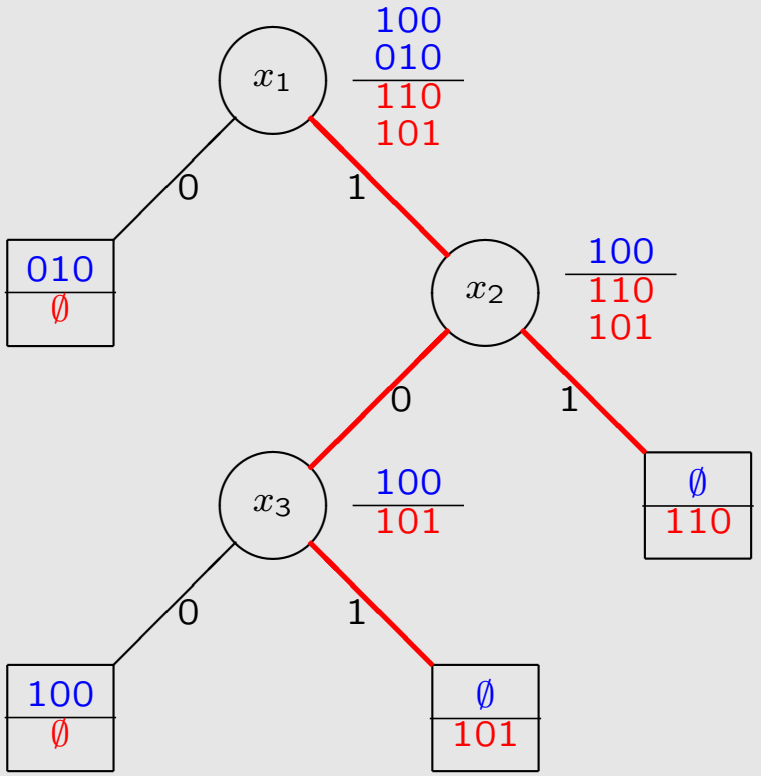
Classifier: $f_{NN} = \overline{x}_1 x_2 \vee \overline{x}_2 \overline{x}_3$.

This classifier is a theory.

Examples

Decision Trees for pdBfs





$$f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3$$

$$\overline{f}_D = x_1x_2 \vee x_1\overline{x}_2x_3 = x_1x_2 \vee x_1x_3$$

f_D is a theory and $\overline{f}_D$ is a co-theory .

Theories as justifiable classifiers

How can we jsutify the choice of a theory over another one?

Typically done experimentally, with measures of classifier quality that involve new data:

- **training - test partition, cross validation** (assumes distribution of future examples follows that of data)
- **simulation** (assumes knowledge of distribution of future examples)
- **clinical trial** (done "in the future")

Theories as justifiable classifiers

Theory f classifies an example $\mathbf{x}$ as a "positive" example if $f(\mathbf{x}) = 1$.

This is the case only if (at least) one pattern of f is "triggered" by $\mathbf{x}$, meaning that we have observed earlier another positive example displaying the same features, and we have never observed a negative example displaying these features.

Theories as justifiable classifiers

Theory f classifies an example $\mathbf{x}$ as a "positive" example if $f(\mathbf{x}) = 1$.

This is the case only if (at least) one pattern of f is "triggered" by $\mathbf{x}$, meaning that we have observed earlier another positive example displaying the same features, and we have never observed a negative example displaying these features.

But we don't necessarily have a good justification for the opposite classification.

Theories as justifiable classifiers

Theory f classifies an example $\mathbf{x}$ as a "positive" example if $f(\mathbf{x}) = 1$.

This is the case only if (at least) one pattern of f is "triggered" by $\mathbf{x}$, meaning that we have observed earlier another positive example displaying the same features, and we have never observed a negative example displaying these features.

But we don't necessarily have a good justification for the opposite classification.

Similarly, co-theory g classifies an example $\mathbf{x}$ as a "negative" example if $g(\mathbf{x}) = 1$.

Theories as justifiable classifiers

Theory f classifies an example $\mathbf{x}$ as a "positive" example if $f(\mathbf{x}) = 1$.

This is the case only if (at least) one pattern of f is "triggered" by $\mathbf{x}$, meaning that we have observed earlier another positive example displaying the same features, and we have never observed a negative example displaying these features.

But we don't necessarily have a good justification for the opposite classification.

Similarly, co-theory g classifies an example $\mathbf{x}$ as a "negative" example if $g(\mathbf{x}) = 1$.

In both cases, we can provide some explanation or justification for the classification, but not for the opposite one.

Theories, Co-Theories and Bi-Theories

A pair of a theory $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$ and a co-theory $g \in \mathcal{E}_T(\mathbf{F}, \mathbf{T})$ can be used to define a classifier F:

$$F_{f,g}(\mathbf{x}) = \begin{cases} 1 & \text{if } f(\mathbf{x}) = 1 \text{ and } g(\mathbf{x}) = 0, \\ 0 & \text{if } f(\mathbf{x}) = 0 \text{ and } g(\mathbf{x}) = 1, \\ ? & \textbf{otherwise} \end{cases}$$

Such a classifier can justify all its definite answers with evidence from $(\mathbf{T}, \mathbf{F})$, however, **it may not be able to give an answer for all** $\mathbf{x} \in \{0, 1\}^V$!

To avoid such uncertainties, ideally we would like to use a pair for which

$$\overline{g} = f$$

Theories, Co-Theories and Bi-Theories

A theory $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$ is called a **bi-theory** of $(\mathbf{T}, \mathbf{F})$ if $\overline{f}$ is a co-theory of $(\mathbf{T}, \mathbf{F})$.

Then, the pair $(f, \overline{f})$ defines a classifier $F_{f,\overline{f}}$ which always provides evidence to support its answers.

Theories, Co-Theories and Bi-Theories

A theory $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$ is called a **bi-theory** of $(\mathbf{T}, \mathbf{F})$ if $\overline{f}$ is a co-theory of $(\mathbf{T}, \mathbf{F})$.

Then, the pair $(f, \overline{f})$ defines a classifier $F_{f,\overline{f}}$ which always provides evidence to support its answers.

Do bi-theories always exist for all data sets?

Theories, Co-Theories and Bi-Theories

A theory $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$ is called a **bi-theory** of $(\mathbf{T}, \mathbf{F})$ if $\overline{f}$ is a co-theory of $(\mathbf{T}, \mathbf{F})$.

Then, the pair $(f, \overline{f})$ defines a classifier $F_{f,\overline{f}}$ which always provides evidence to support its answers.

Do bi-theories always exist for all data sets?

YES, in fact (most) **nearest neighbor** and **decision tree** algorithms build a classifier $F_{f,\overline{f}}$ for some bi-theory $f \in \mathcal{E}_B(\mathbf{T}, \mathbf{F})$.

Theories, Co-Theories and Bi-Theories

A theory $f \in \mathcal{E}_T(\mathbf{T}, \mathbf{F})$ is called a **bi-theory** of $(\mathbf{T}, \mathbf{F})$ if $\overline{f}$ is a co-theory of $(\mathbf{T}, \mathbf{F})$.

Then, the pair $(f, \overline{f})$ defines a classifier $F_{f,\overline{f}}$ which always provides evidence to support its answers.

Do bi-theories always exist for all data sets?

YES, in fact (most) **nearest neighbor** and **decision tree** algorithms build a classifier $F_{f,\overline{f}}$ for some bi-theory $f \in \mathcal{E}_B(\mathbf{T}, \mathbf{F})$.

But in general, **some bi-theories do not correspond to any decision tree nor to any nearest neighbor classifier.**

Theory Building

Theories can be built by selecting enough patterns to cover all positive examples.

This can be done for instance by solving an optimization problem, either exactly or in a greedy way.

Similarly for co-theories.

Many applications in the literature...

Theory Building

Theory formation: for each vector $\mathbf{a} \in \mathbf{T}$ we choose at most 5 patterns with the highest coverage from $\mathcal{P}(\mathbf{a}, \mathbf{T}, \mathbf{F}, \gamma)$.

Results of 10-fold cross validation

Data Set	Training	Test
AU CREDIT*	88.9%	85.4%
BCW	99.7%	97.4%
BUPA	97.4%	90.1%
DNA*	87.2%	87.5%
HEART	100.0%	96.3%
HEPATITIS	100.0%	87.0%
IONOSPHERE	99.9%	95.2%
PIMA	81.3%	77.9%
VEHICLE*†	93.2%	80.8%
VOTES	100.0%	98.3%
WINE	100.0%	97.9%

* STATLOG Data Collection

† 4 classes

Conclusions

In conclusion

♠ A **best pattern** alone is a **very good, simple and robust classifier**.

♡ **Theories** built as disjunctions of good patterns provide **excellent classifiers for a large variety of applications**.

♣ **Theories** provide classifications that are both **understandable and justifiable**.

♢ Several **software packages** have been developed.

References

E. Boros, Y. Crama, P.L. Hammer, T. Ibaraki, A. Kogan and K. Makino, Logical Analysis of Data: Classification with justification, Annals of Operations Research 188 (2011) 33-61.

Y. Crama, P. L. Hammer and T. Ibaraki, Cause-effect relationships and partially defined boolean functions, Annals of Operations Research 16 (1988) 299-326.

T. Ibaraki, Partially defined Boolean functions, Chapter 8 in: Y. Crama and P. L. Hammer, *Boolean Functions - Theory, Algorithms and Applications*, Cambridge University Press, New York, 2011.