Partially Defined Boolean Functions, Classification, and Logical Analysis of Data

Yves Crama

HEC Management School, University of Liège

based on earlier presentations by Endre Boros

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

Outline

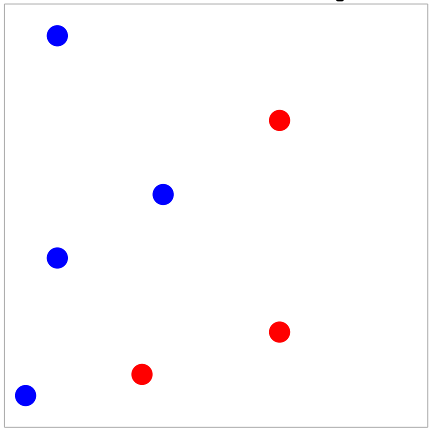
- **Learning from Examples**

- Partially Defined Boolean Functions

- Logical Analysis of Data

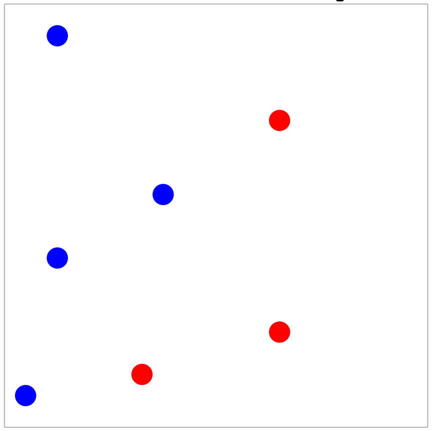
Process of Learning ...

Data: Examples



Process of Learning ...

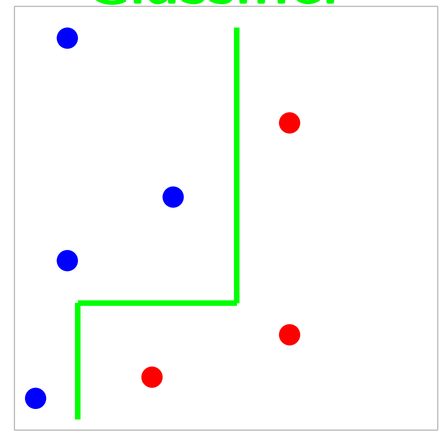
Data: Examples



Learning

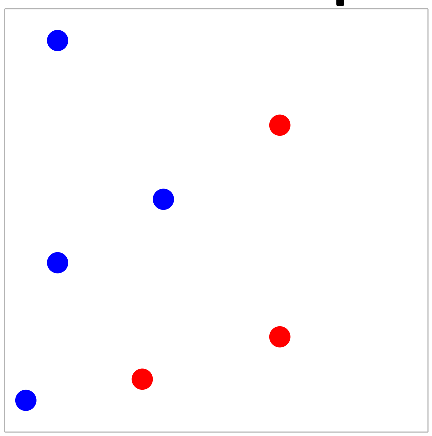
$\Rightarrow \cdots \Rightarrow$

Classifier



Process of Learning ...

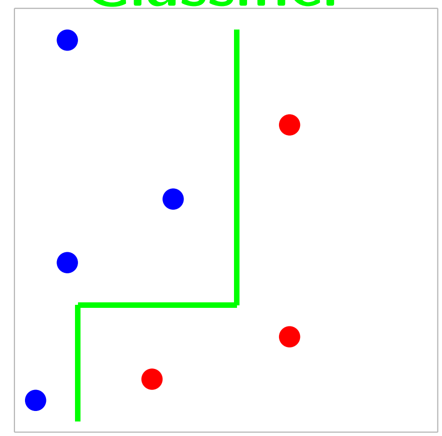
Data: Examples



Learning

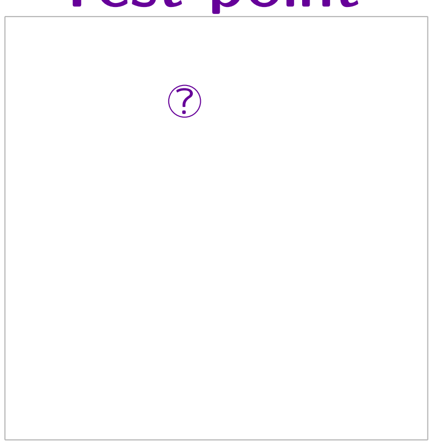
$\Rightarrow \cdots \Rightarrow$

Classifier



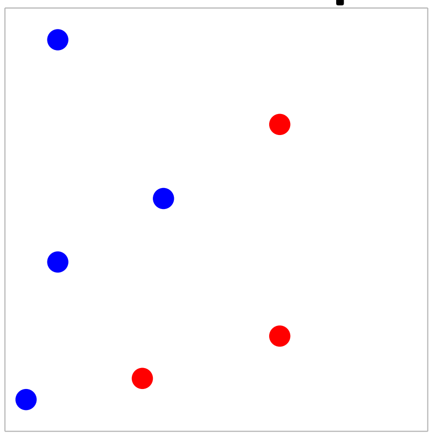
Testing:

Test point



Process of Learning ...

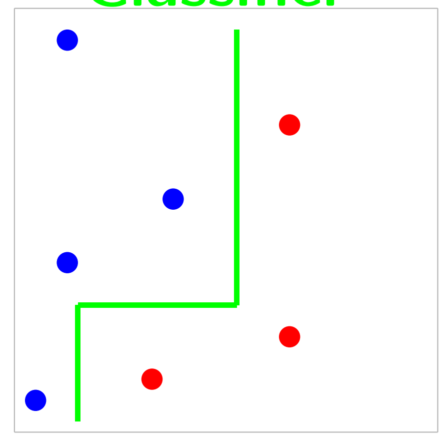
Data: Examples



Learning

$\Rightarrow \cdots \Rightarrow$

Classifier



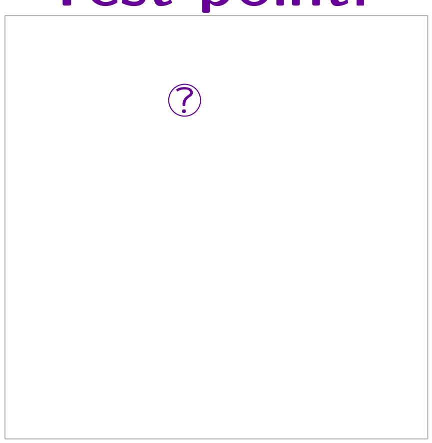
$\Downarrow$

$\vdots$

$\Downarrow$

Testing:

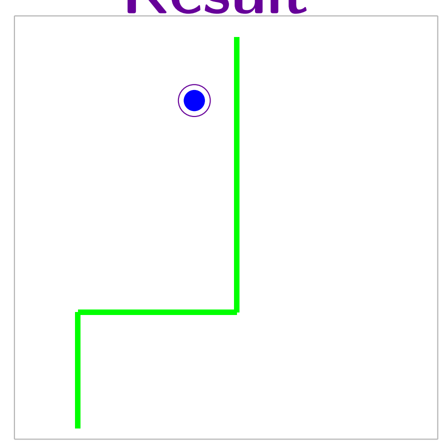
Test point:



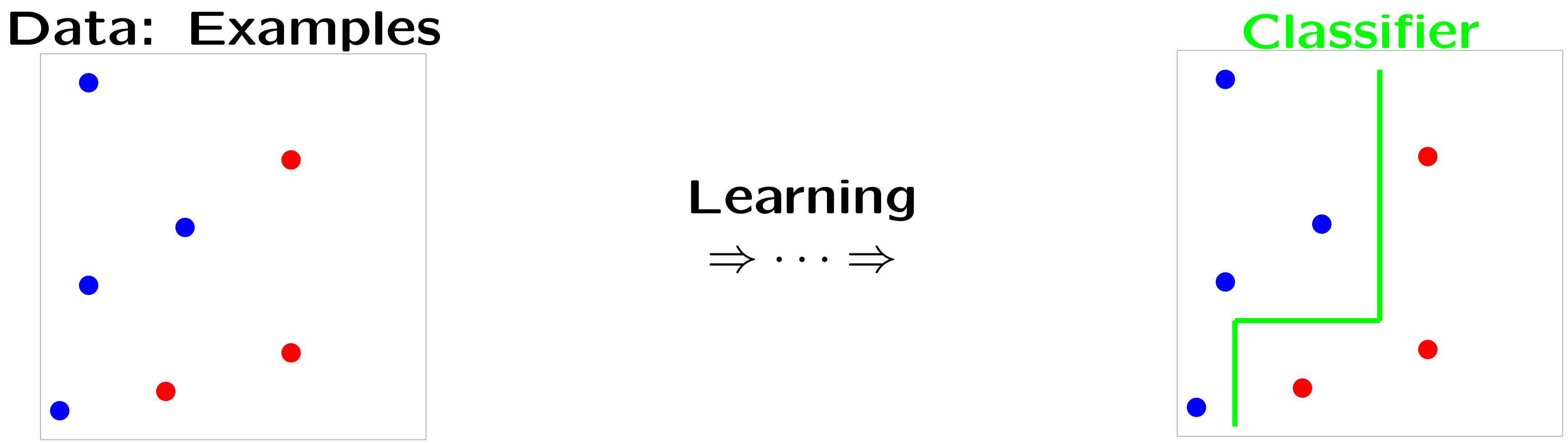
Trial

$\Rightarrow \cdots \Rightarrow$

Result



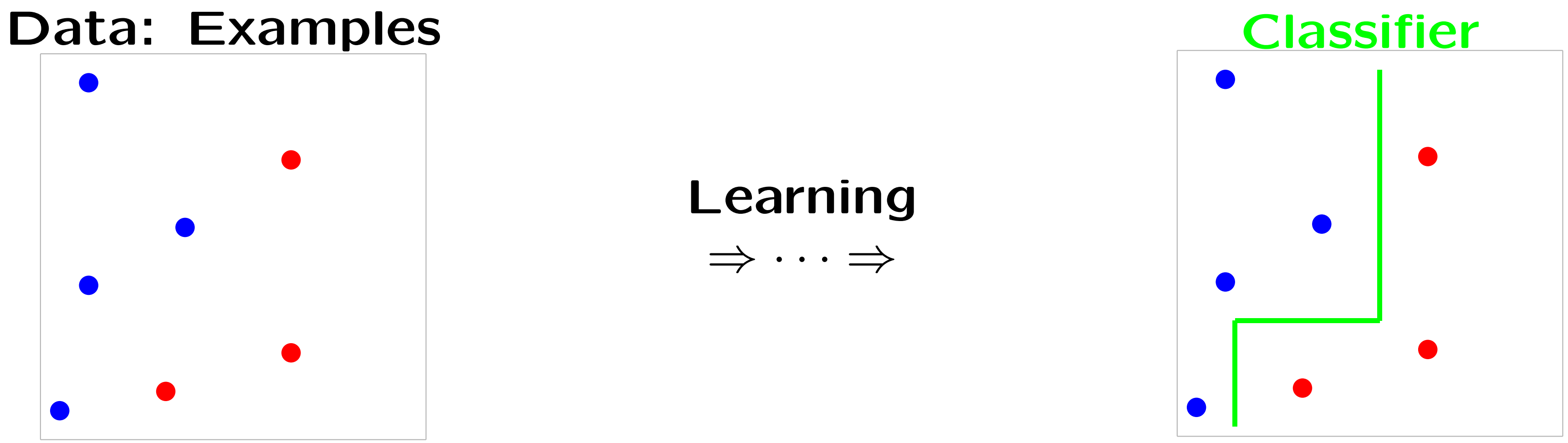
Process of Learning ...



Common measures of classifier quality involve new data:

- **training - test partition, cross validation** (assumes distribution of future examples follows that of data)

Process of Learning ...



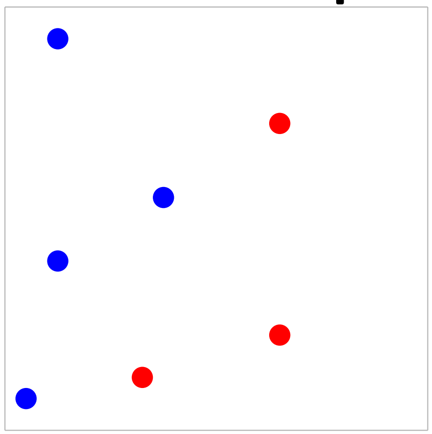
Common measures of classifier quality involve new data:

- **training - test partition, cross validation** (assumes distribution of future examples follows that of data)

- **simulation** (assumes knowledge of distribution of future examples)

Process of Learning ...

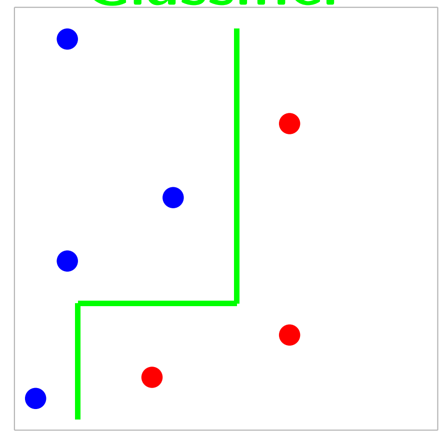
Data: Examples



Learning

$\Rightarrow \cdots \Rightarrow$

Classifier



Common measures of classifier quality 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")

WHAT CANNOT BE LEARNED FROM EXAMPLES!

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

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.

Data Sets and Classifiers

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

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

Class: $c : \mathbf{D} \longrightarrow \{0, 1\}$

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: $c : \mathbf{D} \longmapsto \{0, 1\}$

Classifier: $f : \mathsf{A} \times \mathsf{B} \times \cdots \longmapsto \{0, 1\}$

Data Sets and Classifiers

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

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

Class: $c : \mathbf{D} \longmapsto \{0, 1\}$

Classifier: f : A $\times$ B $\times \cdots \longmapsto \{0, 1\}$

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: $c : \mathbf{D} \longmapsto \{0, 1\}$

Classifier: f : A $\times$ B $\times \cdots \longmapsto \{0, 1\}$

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	$\mathbf{x_1 + x_2 + x_3 + x_4 \geq 3}$	$\mathbf{x_1 + 2x_2 + x_4 \geq 3}$
$\mathbb{T}$	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

	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 x_1	x_2	x_3	x_4	Dr. F $\mathbf{x_1 + x_2 + x_3 + x_4 \geq 3}$	Dr. G $\mathbf{x_1 + 2x_2 + x_4 \geq 3}$
$\mathbb{T}$	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

	Ms. Y	1	1	0	0	**0**	**1**
	Mr. Z	1	0	1	1	**1**	**0**

Partially Defined Boolean Functions

- **Definition**

- Support sets

- Patterns

- Theories

Definitions

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

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

Definitions

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

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

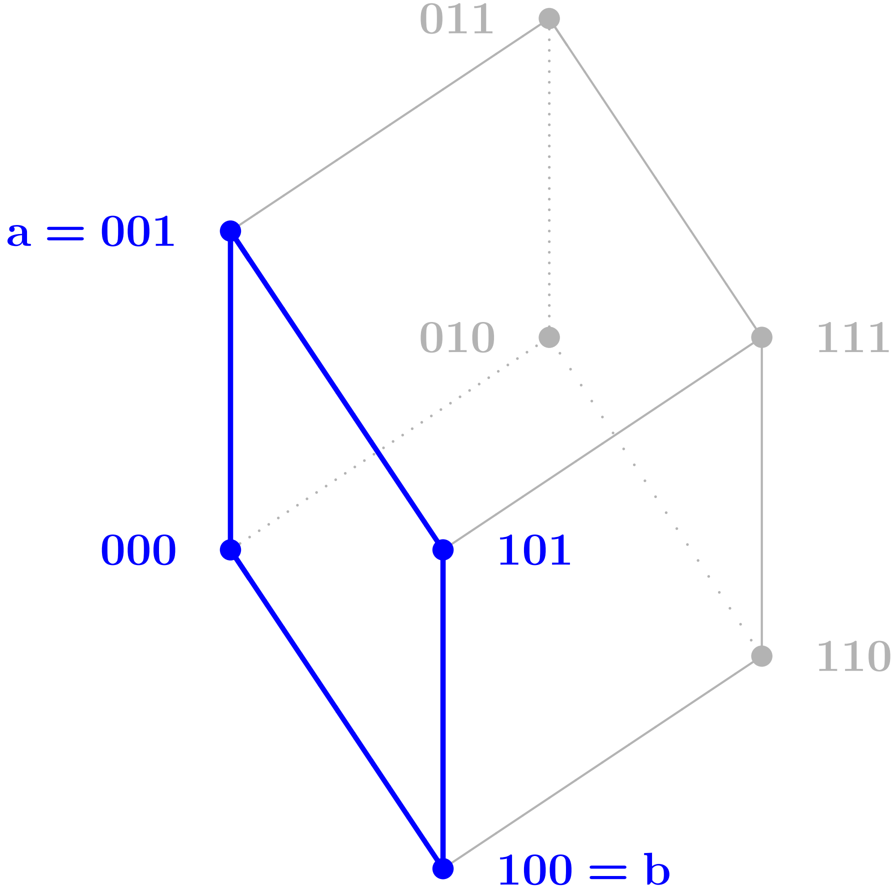
where $P, N \subseteq V$, and $\overline{x} = 1 - x$.

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

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

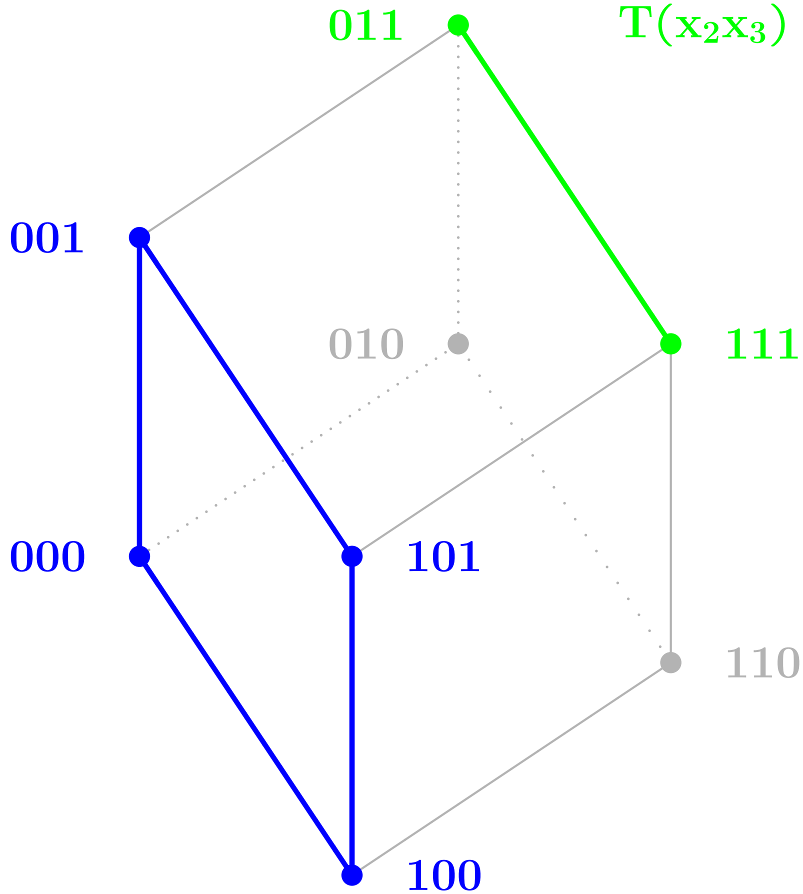
The set of true vectors of a term forms a **sub-cube** of $\{0, 1\}^V$, and vice-versa, every sub-cube, is the set of true vectors of a Boolean function, defined by a unique term.

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



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

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



$$\mathbf{T(x_2x_3) = (*11) = [(011),(111)]}$$

Definitions

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

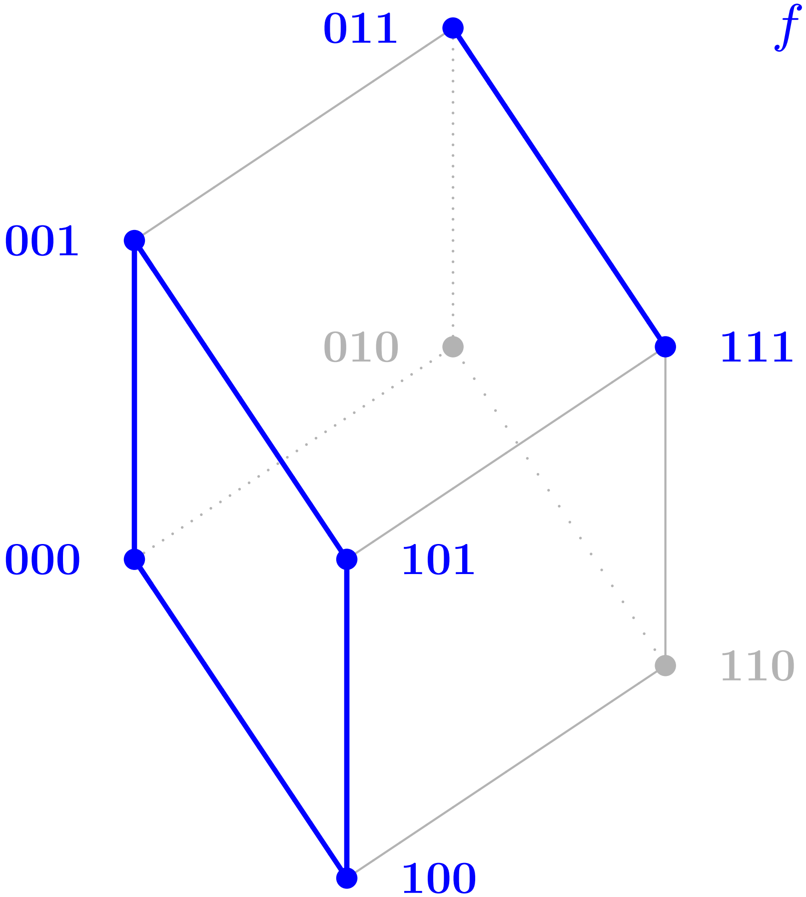
$$f(\mathbf{x}) = \bigvee_{(P,N)\in E} \Big(\bigwedge_{j\in P} x_j \wedge \bigwedge_{j\in N} \overline{x}_j \Big)$$

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



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

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. Clearly, 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

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

- Simplicity
  - Essential attributes
  - Function $f \in \mathcal{E}(\mathbf{T}, \mathbf{F})$
  - Representation (DNF, CNF, decision tree, etc.)

What can guide learning?

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

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

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

- Justifiability

Note on framework: we mostly speak here of building *unspecified models*, as opposed to *specified models* such as regression models (which assume a priori 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.

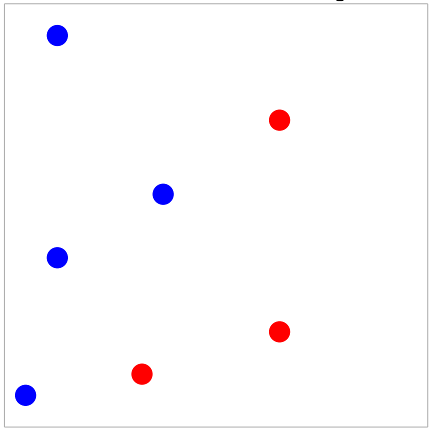
Nearest Neighbor classifiers

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

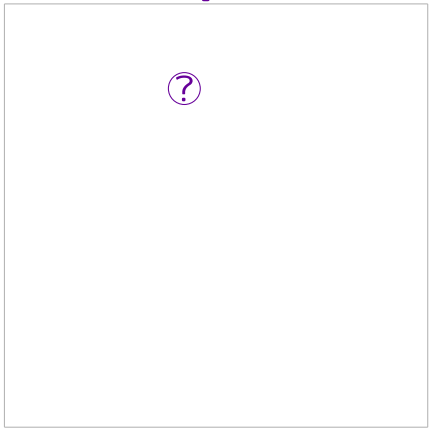
Nearest Neighbor classifiers

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

Data: Examples



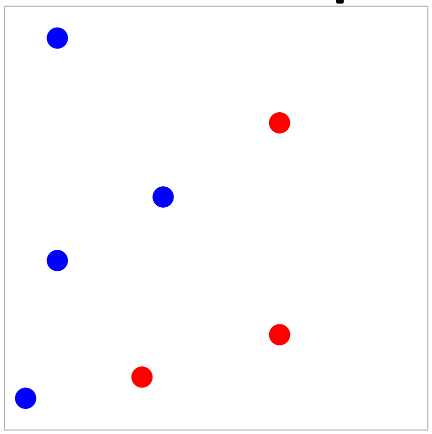
Test point:



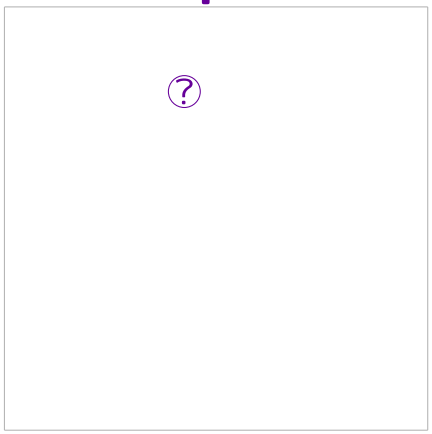
Nearest Neighbor classifiers

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

Data: Examples



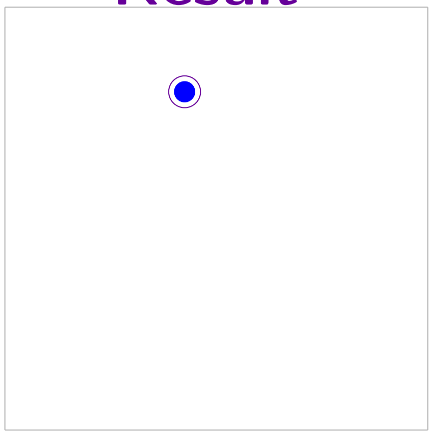
Test point:



Closest example

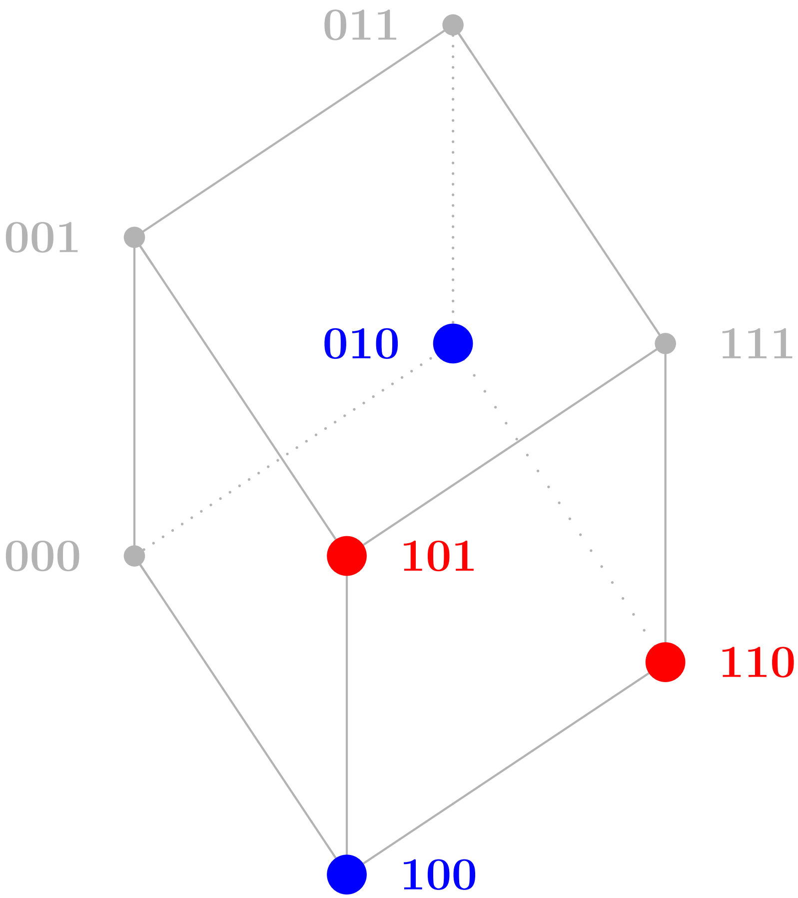
$\Rightarrow \cdots \Rightarrow$

Result



Nearest Neighbor classifiers

Example in the Boolean case.

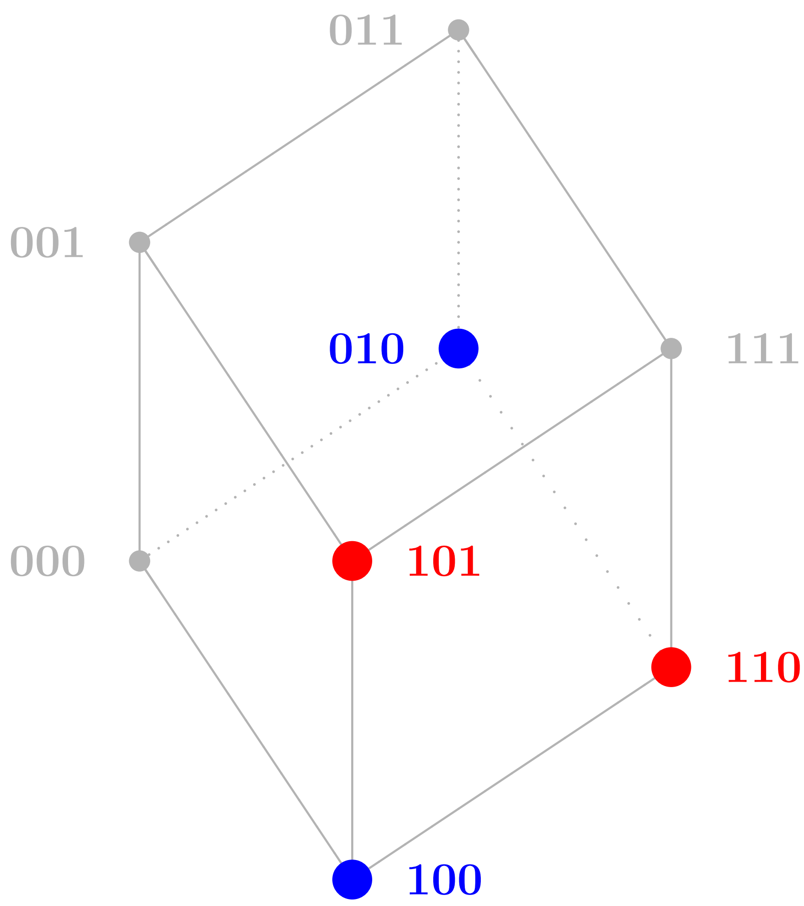


$$\mathbf{F} = \{(\mathbf{110}), (\mathbf{101})\}$$

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

Nearest Neighbor classifiers

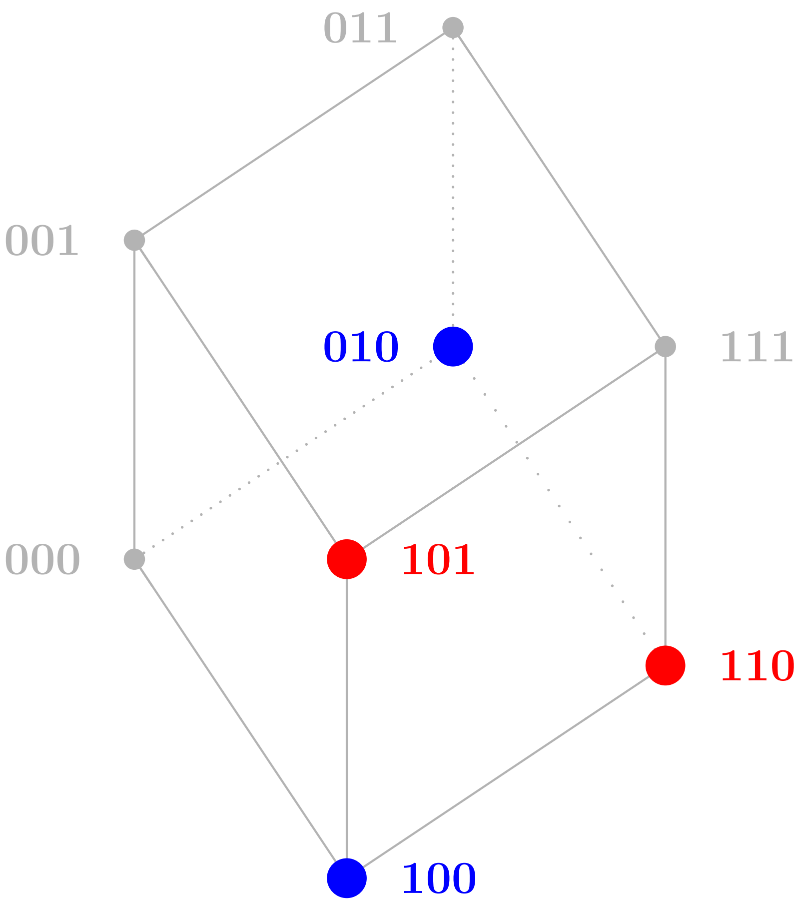
Example in the Boolean case.



(111) is classified as red (false)

(000) is classified as blue (true)

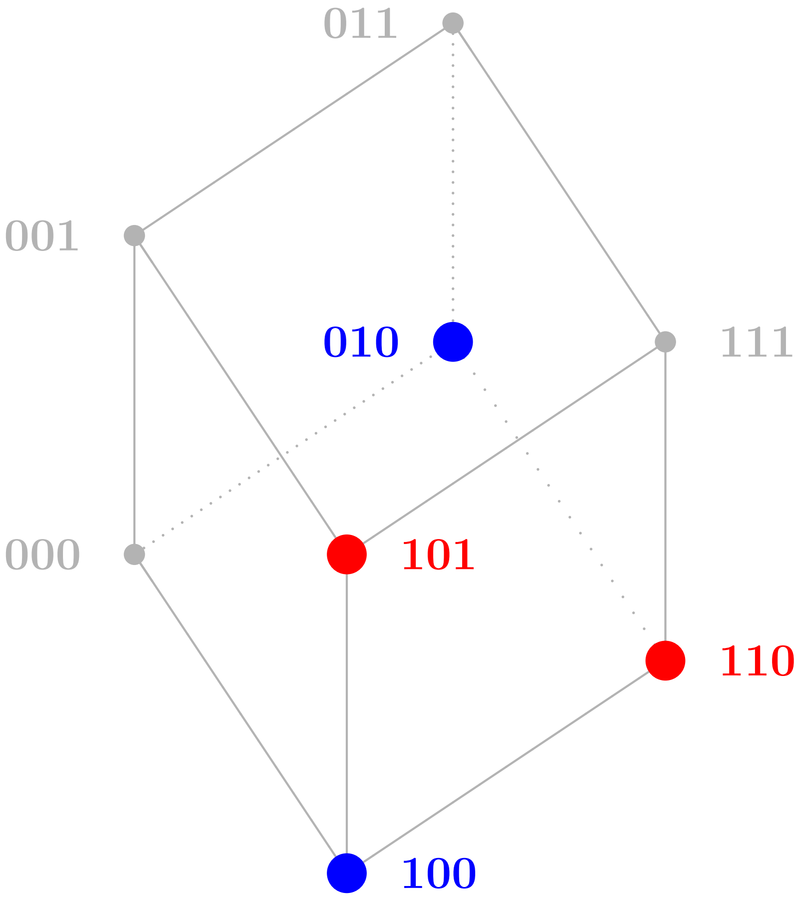
Decision Trees for pdBfs

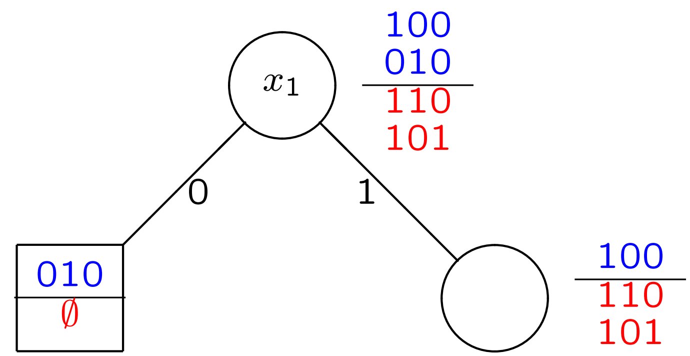


$\mathbf{F} = \{(\mathbf{110}), (\mathbf{101})\}$

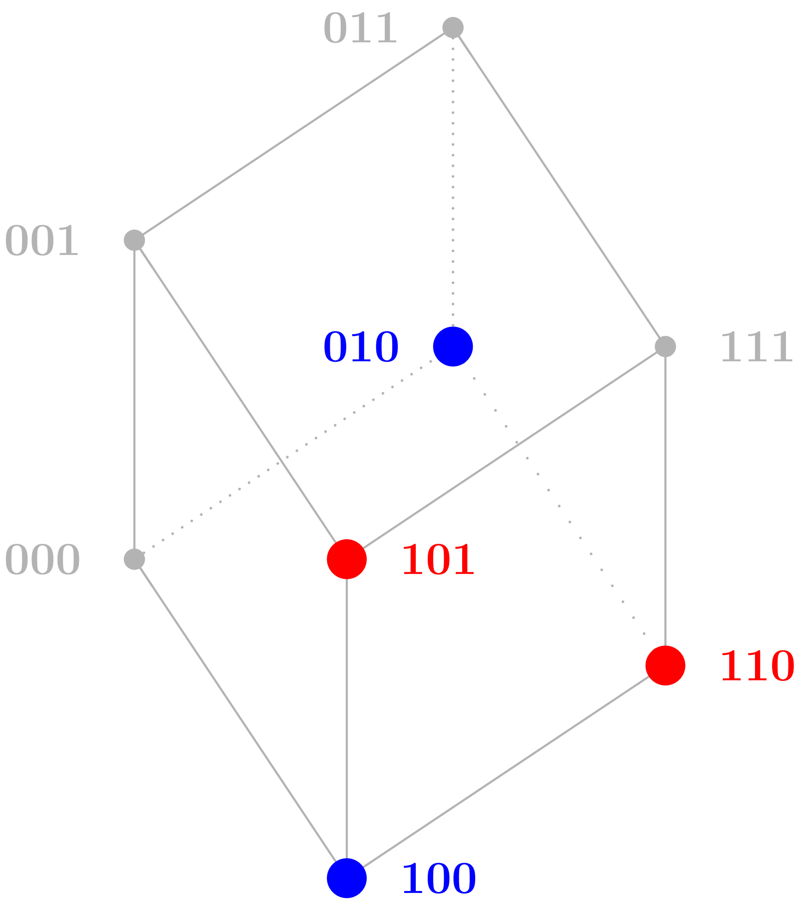
$\mathbf{T} = \{(\mathbf{010}), (\mathbf{100})\}$

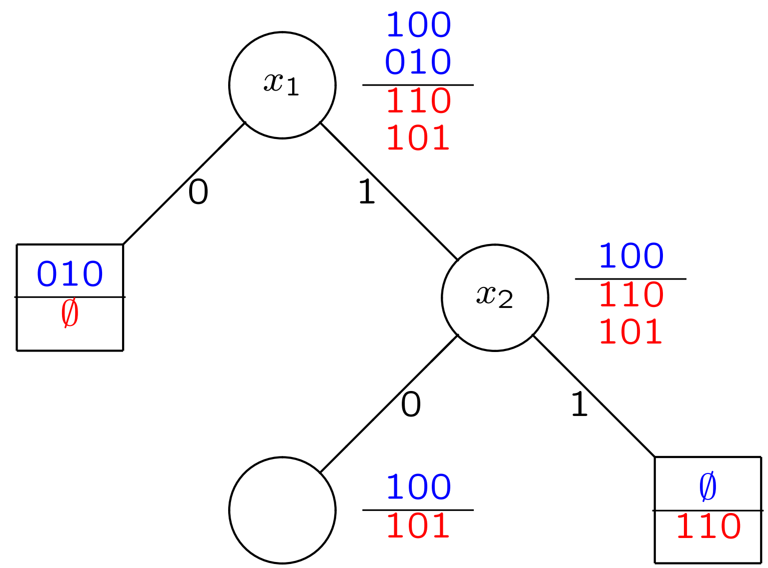
Decision Trees for pdBfs



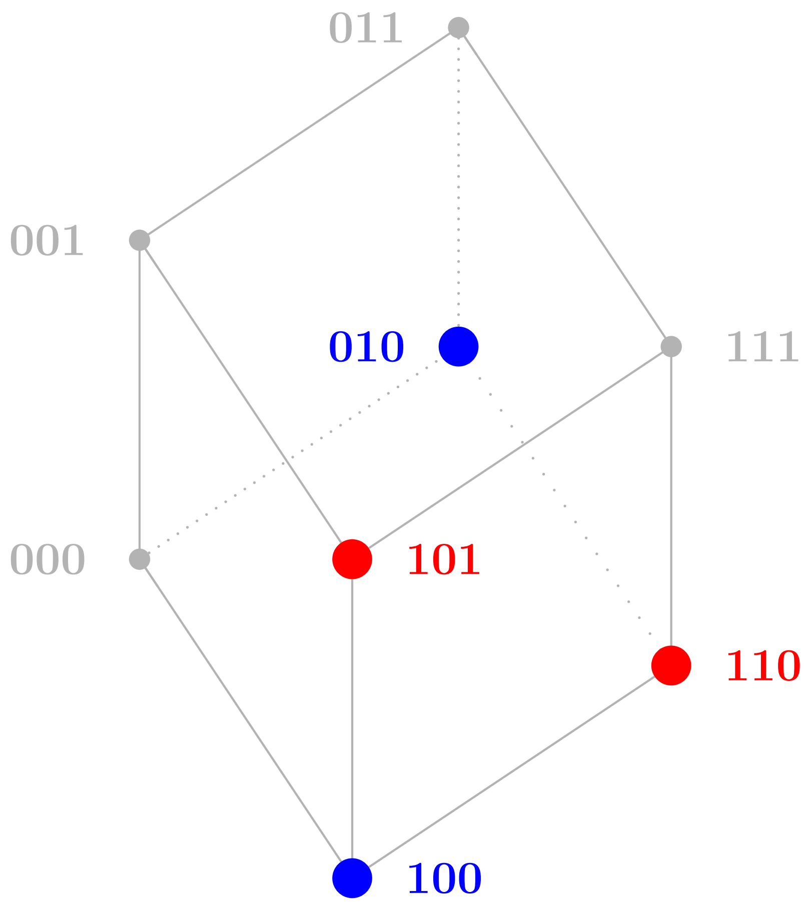


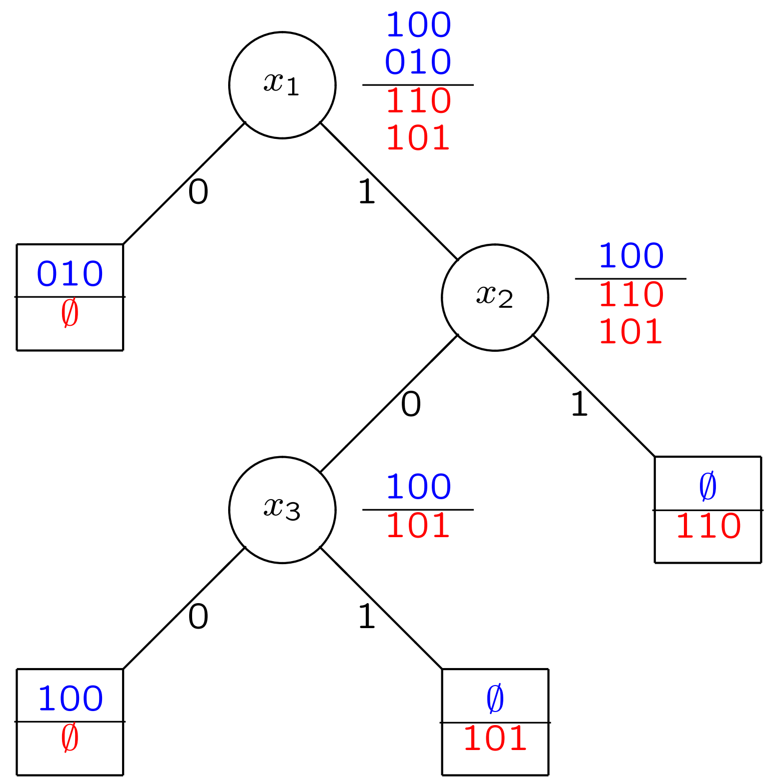
Decision Trees for pdBfs



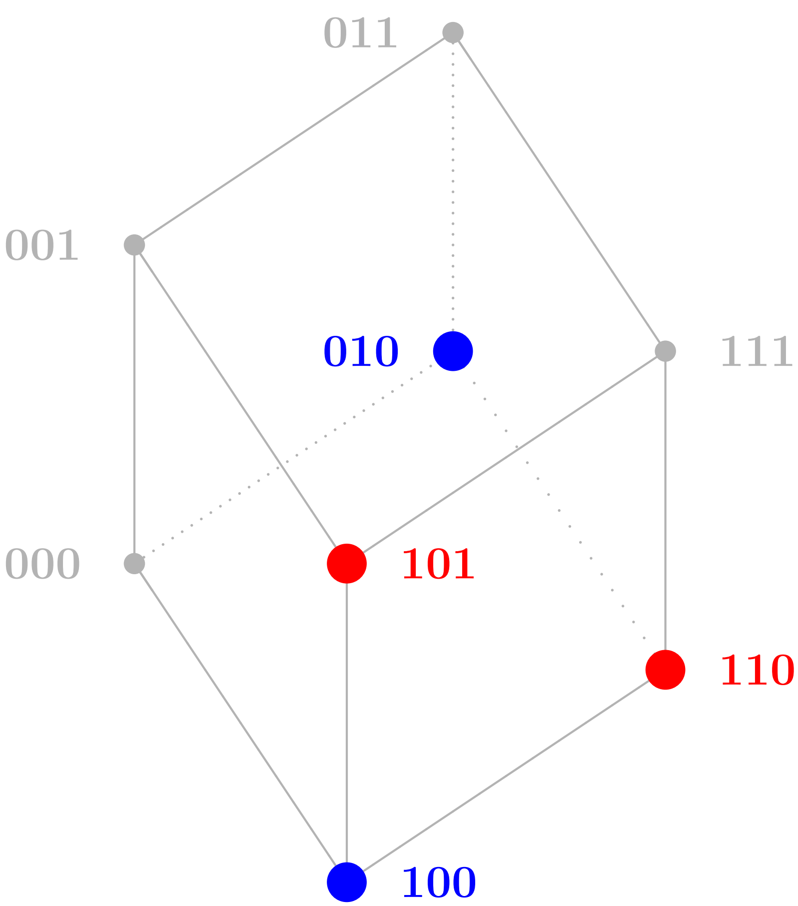


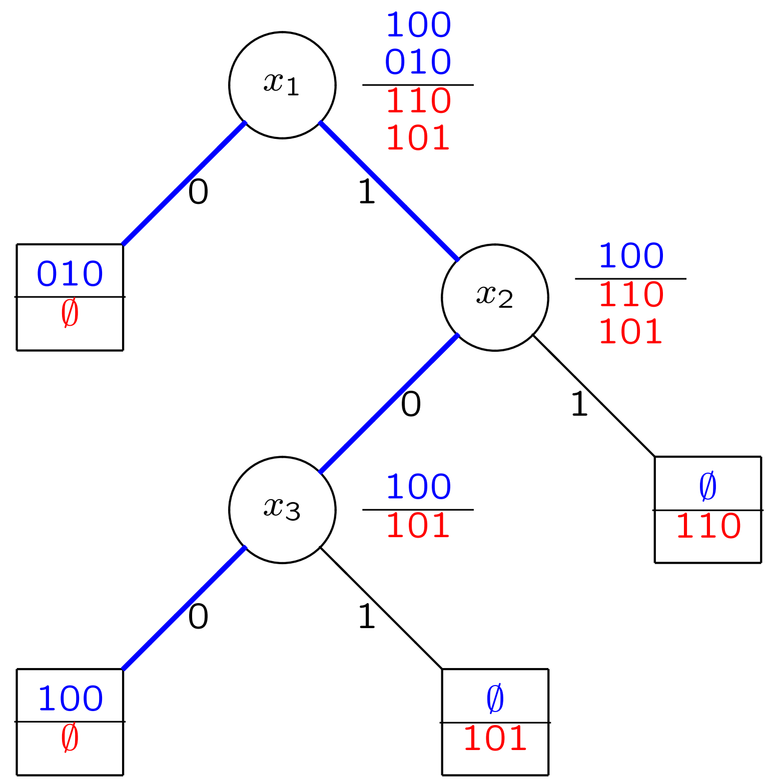
Decision Trees for pdBfs





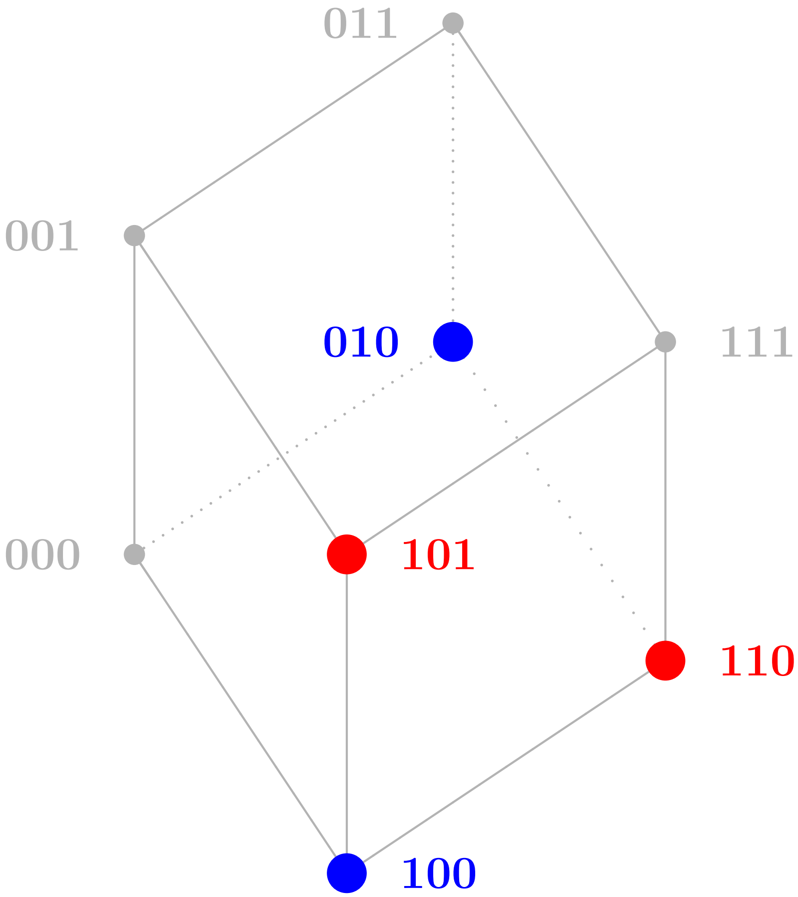
Decision Trees for pdBfs

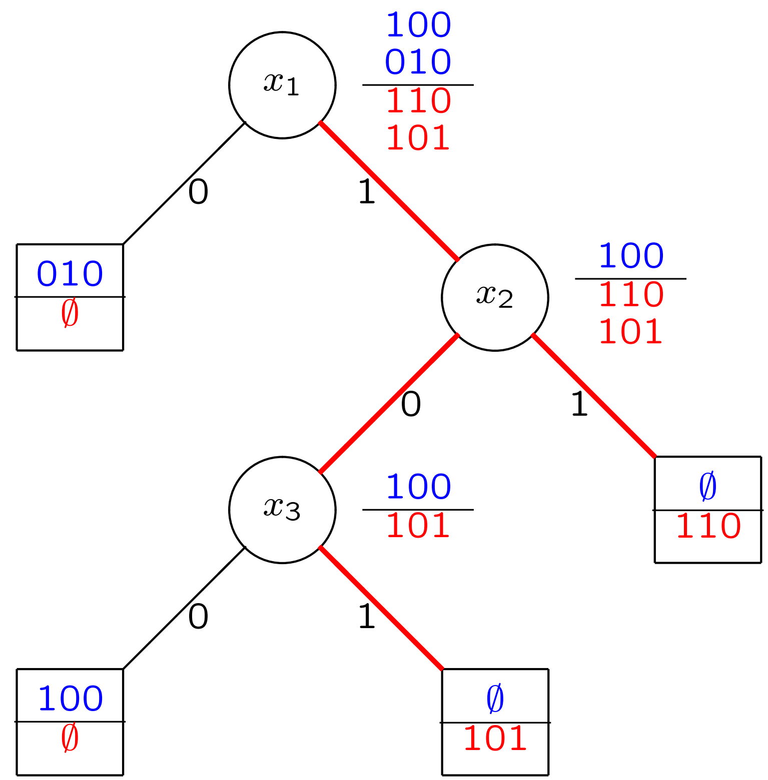




$$f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3$$

Decision Trees for pdBfs



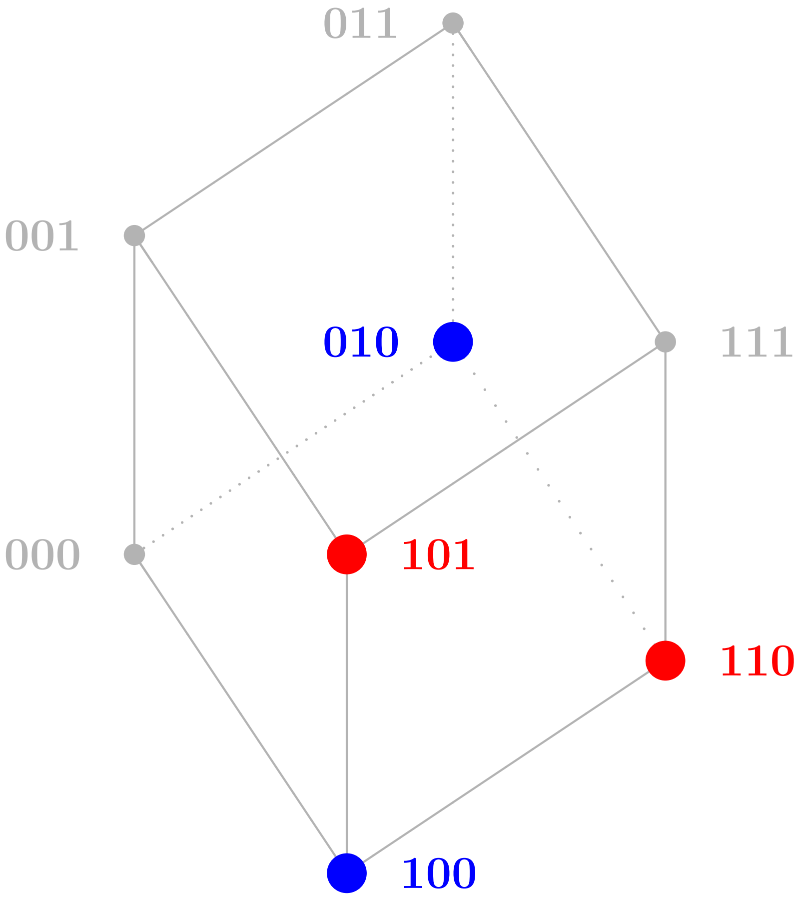


$$\mathbf{f_D = \overline{x}_1 \vee x_1\overline{x}_2\overline{x}_3 = \overline{x}_1 \vee \overline{x}_2\overline{x}_3}$$

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

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.

Logical Analysis of Data

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

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)

Partially Defined Boolean Functions

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

Finding Essential Attributes

- Select relevant features.
- Eliminate noise.
- 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).

Eliminate Noise: What is it?

- A random attribute?

- An irrelevant attribute?

- An (almost) constant attribute?

- A dependent attribute?

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.

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.

- 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.

- The larger the data set, the more **surprising** to have a small support set.

- OUR SURPRISE $\approx$ INFORMATION IN DATA

The Good News

Boros, Horiyama, Ibaraki, Makino and Yagiura, 2003

Distribution of Support Sets in Randomly Generated Data

	K	5	6	7	8	9	10	11	12
$n = 10$	LB	22	34	46	60	66			
	UB	60	58	68	80	84			
$n = 15$	LB	26	42	62	90	126	176	238	312
	UB	∞	82	98	126	164	214	278	352
$n = 20$	LB	30	46	70	102	150	216	306	432
	UB	∞	102	116	150	198	268	364	494
$n = 40$	LB	34	54	84	126	198	278	408	594
	UB	∞	∞	156	196	262	358	498	694
$n = 100$	LB	38	62	96	148	226	336	500	734
	UB	∞	∞	236	252	330	450	624	876
$n = 1000$	LB	46	76	122	190	292	442	662	982
	UB	∞	∞	∞	420	480	672	874	1220

Lower and upper bounds on the threshold size of data sets (assuming $|\mathbb{T}| = |\mathbb{F}|$ and uniform random generation) above which support sets of size K are unlikely to exists.

The Good News

Distribution of Support Sets in Randomly Generated Data

	K	5	**6**	7	8	9	10	11	12
$n = 10$	LB	22	34	46	60	66			
	UB	60	58	68	80	84			
$n = 15$	LB	26	42	62	90	126	176	238	312
	UB	∞	82	98	126	164	214	278	352
$n = \mathbf{20}$	LB	30	**46**	70	102	150	216	306	432
	UB	∞	**102**	116	150	198	268	364	494
$n = 40$	LB	34	54	84	126	198	278	408	594
	UB	∞	∞	156	196	262	358	498	694
$n = 100$	LB	38	62	96	148	226	336	500	734
	UB	∞	∞	236	252	330	450	624	876
$n = 1000$	LB	46	76	122	190	292	442	662	982
	UB	∞	∞	∞	420	480	672	874	1220

- If we have 20 attributes and less than 46 records, then it is very likely to have many support sets of size 6 or smaller.

- If we have more than 102 records, then it is very unlikely to have a support set of size 6 or smaller $\Longrightarrow$ **If there is one ...**

The Good News

Distribution of Support Sets in Randomly Generated Data

	K	5	6	7	8	9	**10**	11	12
$n = 10$	LB	22	34	46	60	66			
	UB	60	58	68	80	84			
$n = 15$	LB	26	42	62	90	126	176	238	312
	UB	∞	82	98	126	164	214	278	352
$n = 20$	LB	30	46	70	102	150	216	306	432
	UB	∞	102	116	150	198	268	364	494
$n = 40$	LB	34	54	84	126	198	278	408	594
	UB	∞	∞	156	196	262	358	498	694
$n = 100$	LB	38	62	96	148	226	336	500	734
	UB	∞	∞	236	252	330	450	624	876
$n = 1000$	LB	46	76	122	190	292	**442**	662	982
	UB	∞	∞	∞	420	480	**672**	874	1220

- If, in a data set with 1000 attributes and more than 672 records, we find a **support set of size** 10, then
  - it might be a unique one;
  - it is probably related to the structure of the data, and not to random noise.

Partially Defined Boolean Functions

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

Definitions

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 data point, but which never occurs in a false data 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
	A	1	1	0	1
$\mathbb{T}$	B	0	1	1	1
	C	1	1	1	0
	T	0	0	1	1
$\mathbb{F}$	U	1	0	0	1
	V	1	0	1	0
	W	0	1	1	0
	Ms. Y	1	1	0	0
	Mr. Z	1	0	1	1

Some patterns:

$$x_1 x_2,\ x_2 \overline{x}_3,\ x_2 x_4, \ldots$$

Some co-patterns:

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

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 $(\mathbf{T}, \mathbf{F})$.

A theory g of $(\mathbf{F}, \mathbf{T})$ is called a **co-theory** of $(\mathbf{T}, \mathbf{F})$ (it can be represented by a disjunction of some of the co-patterns of $(\mathbf{T}, \mathbf{F})$).

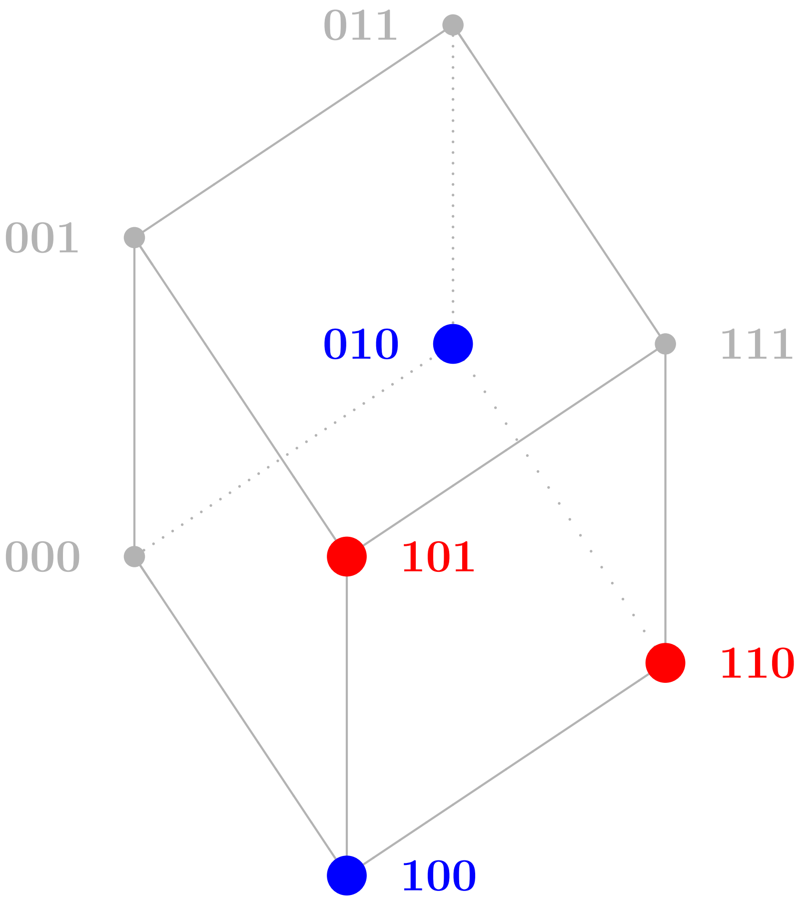
Denote by $\mathcal{E}_T(\mathbf{T}, \mathbf{F})$ and $\mathcal{E}_T(\mathbf{F}, \mathbf{T})$ the families of theories and co-theories of a given pdBf $(\mathbf{T}, \mathbf{F})$.

Typically we have

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

Examples

Nearest Neighbor classifier



(111) is classified as red (false)

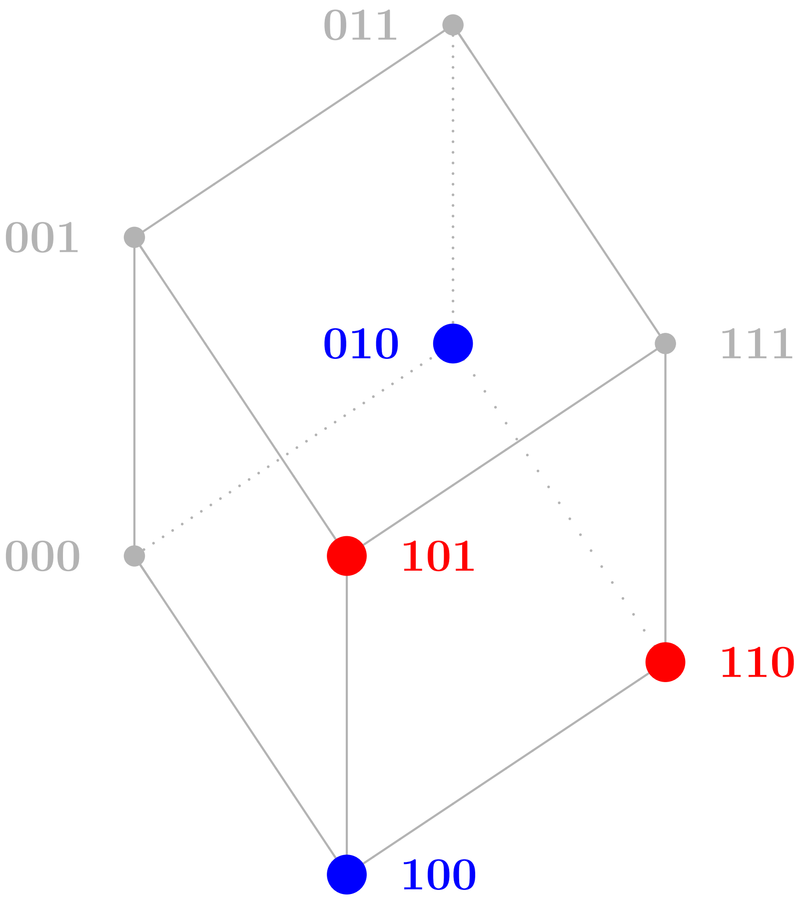
(000) is classified as blue (true)

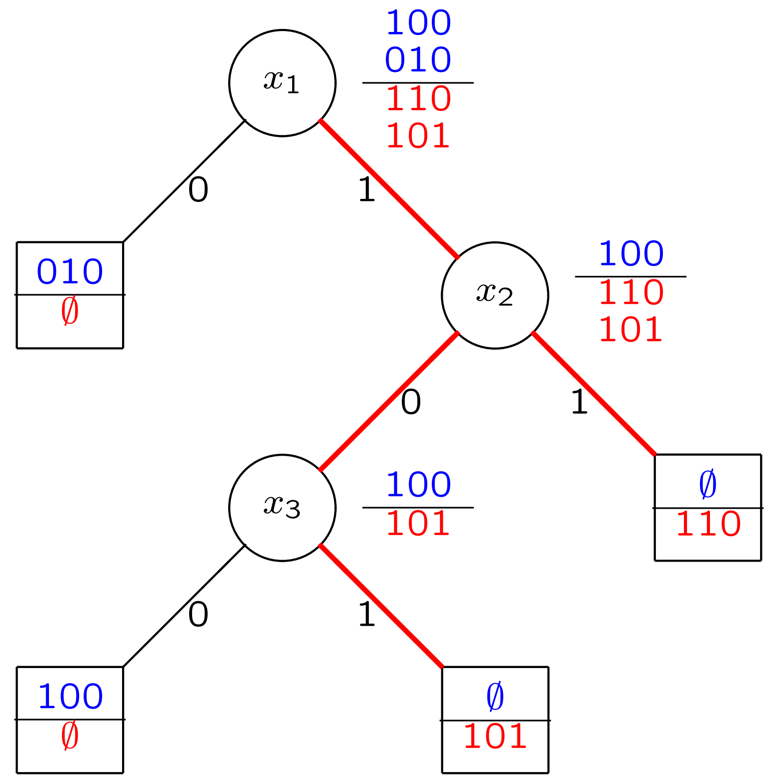
Classifier: $f_{NN} = \overline{x}_1 x_2 \vee \overline{x}_2 \overline{x}_3.$

This classifier is a theory.

Examples

Decision Trees for pdBfs





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

$$\overline{\mathbf{f}}_\mathbf{D} = \mathbf{x}_1\mathbf{x}_2 \vee \mathbf{x}_1\overline{\mathbf{x}}_2\mathbf{x}_3 = \mathbf{x}_1\mathbf{x}_2 \vee \mathbf{x}_1\mathbf{x}_3$$

$\mathbf{f_D}$ is a theory and $\overline{\mathbf{f}}_\mathbf{D}$ is a co-theory .

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 we always have bi-theories?

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 we always have bi-theories?

YES, in fact (most) **nearest neighbor** approaches and **decision tree** based methods 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 we always have bi-theories?

YES, in fact (most) **nearest neighbor** approaches and **decision tree** based methods 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.**

Conclusions

- Bi-theories and decision trees are very strongly related.

Conclusions

- **Bi-theories and decision trees are very strongly related.**

- **Bi-theories and nearest neighbor approaches are very strongly related.**

Conclusions

- Bi-theories and decision trees are very strongly related.
- Bi-theories and nearest neighbor approaches are very strongly related.
- Patterns and co-patterns are the basic building blocks in all these methods.

Logical Analysis of Data

Extensions and applications of LAD

- **Binarization of numerical attributes**

- **Binarization of categorical attributes**

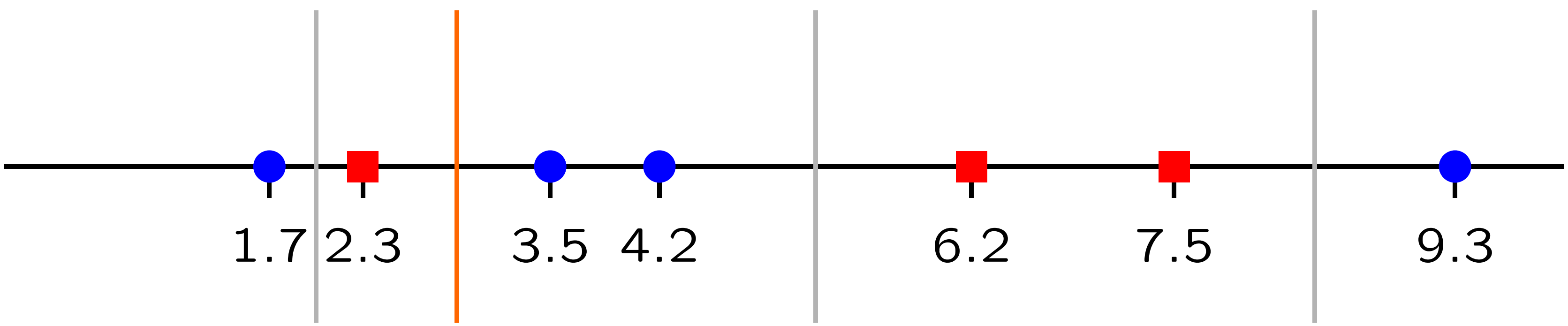
- **Pattern generation**

- **Theory building**

Binarization – Numerical Attributes

	ID	Attributes: $\mathbf{A}$	$\cdots$
$\mathbb{S}^+$	001	1.7	
	002	3.5	
	003	4.2	
	004	9.3	
$\mathbb{S}^-$	991	2.3	
	992	6.2	
	993	7.5	

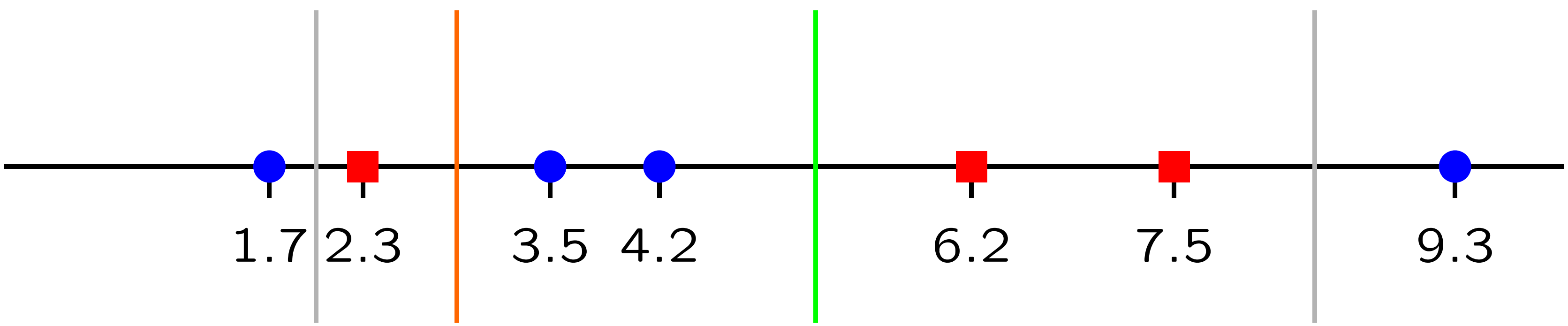
	ID	Binarized Attributes: $\mathbf{A} \geq \mathbf{2.9}$	$\cdots$
$\mathbb{T}$	001	0	
	002	1	
	003	1	
	004	1	
$\mathbb{F}$	991	0	
	992	1	
	993	1	



Binarization − Numerical Attributes

	ID	Attributes: **A**	⋯
$\mathbb{S}^+$	001	1.7	
	002	3.5	
	003	4.2	
	004	9.3	
$\mathbb{S}^-$	991	2.3	
	992	6.2	
	993	7.5	

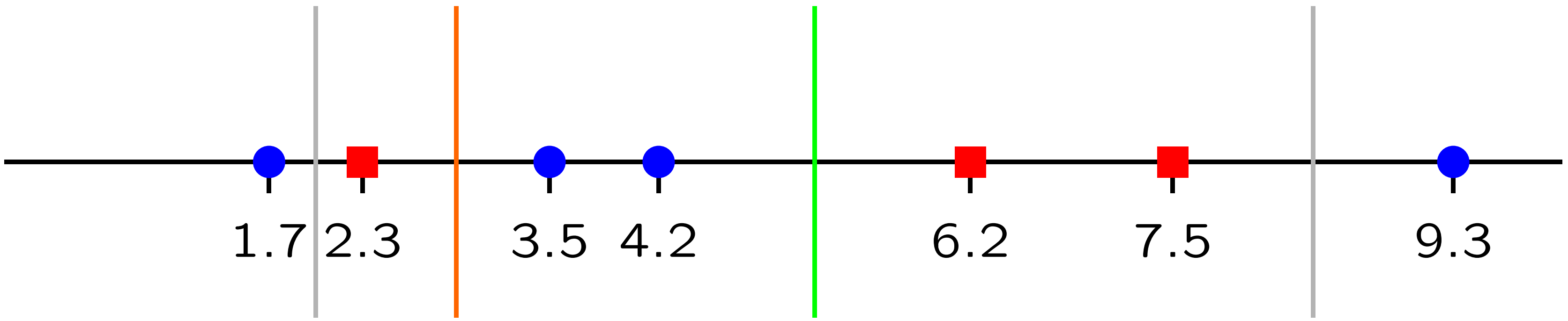
	ID	Binarized Attributes: $\mathbf{A} \geq \mathbf{2.9}$	$\mathbf{A} \geq \mathbf{5.2}$	⋯
$\mathbb{T}$	001	0	0	
	002	1	0	
	003	1	0	
	004	1	1	
$\mathbb{F}$	991	0	0	
	992	1	1	
	993	1	1	



Binarization – Numerical Attributes

	ID	Attributes	
		A	…
$\mathbb{S}^+$	001	1.7	
	002	3.5	
	003	4.2	
	004	9.3	
$\mathbb{S}^-$	991	2.3	
	992	6.2	
	993	7.5	

	ID	Binarized Attributes		
		$A \geq 2.9$	$A \geq 5.2$	…
$\mathbb{T}$	001	0	0	
	002	1	0	
	003	1	0	
	004	1	1	
$\mathbb{F}$	991	0	0	
	992	1	1	
	993	1	1	



Linear time generation; up to 40 cut points per attribute

Binarization

$c(\mathbf{X})$	A	B	C
1	5.7	3.1	blue
1	1.2	4.2	green
1	3.1	5.1	blue
1	2.8	3.2	green
0	7.1	7.3	red
0	5.9	3.6	yellow
0	6.4	4.2	blue
0	3.4	1.6	green

Binarization

$c(\mathbf{X})$	A	B	C
1	5.7	3.1	blue
1	1.2	4.2	green
1	3.1	5.1	blue
1	2.8	3.2	green
0	7.1	7.3	red
0	5.9	3.6	yellow
0	6.4	4.2	blue
0	3.4	1.6	green

$\longrightarrow$

$A \geq 6$	$\cdots$	$B \geq 3$	$\cdots$	$C = blue$	$\cdots$
0		1		1	
0		1		0	
0	$\cdots$	1	$\cdots$	1	$\cdots$
0		1		0	
1		1		0	
0		1		0	
1	$\cdots$	1	$\cdots$	1	$\cdots$
0		0		0	

Binarization

$c(\mathbf{X})$	A	B	C
1	5.7	3.1	blue
1	1.2	4.2	green
1	3.1	5.1	blue
1	2.8	3.2	green
0	7.1	7.3	red
0	5.9	3.6	yellow
0	6.4	4.2	blue
0	3.4	1.6	green

$\longrightarrow$

$A \geq 6$	$\cdots$	$B \geq 3$	$\cdots$	$C = blue$	$\cdots$
0		1		1	
0		1		0	
0	$\cdots$	1	$\cdots$	1	$\cdots$
0		1		0	
1		1		0	
0		1		0	
1	$\cdots$	1	$\cdots$	1	$\cdots$
0		0		0	

$$c(\mathbf{X}) = (A < 6) \wedge (B \geq 3) \wedge (C \in \{blue, green\})$$

Logical Analysis of Data

Extensions and applications of LAD

- **Binarization of numerical attributes**
- **Binarization of categorical attributes**
- **Pattern generation**
- **Theory building**

Pattern Generation

	ID	Test Results			
		$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$	$\mathbf{x_4}$
$\mathbb{T}$	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			
		$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$	$\mathbf{x_4}$
$\mathbb{T}$	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: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

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

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			
		$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$	$\mathbf{x_4}$
$\mathbb{T}$	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: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

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

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			
		$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
$\mathbb{T}$	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*).

TOO MANY!

Pattern Generation

	ID	Test Results			
		$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$	$\mathbf{x_4}$
$\mathbb{T}$	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			
		$\mathbf{x_1}$	$\mathbf{x_2}$	$\mathbf{x_3}$	$\mathbf{x_4}$
$\mathbb{T}$	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: $\mathbf{P}(\mathbf{x}) = x_2 x_4$

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

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!

Even if $|\mathcal{P}(\mathbf{T}, \mathbf{F}, \gamma)|$ is *small*, we cannot guarantee finding them all, unless P=NP!

Patterns

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

Remember:

$cov(\mathbf{P})$ = number of positive examples covered by $\mathbf{P}$

$\pi(\mathbf{P})$ = fraction of examples correctly classified by $\mathbf{P}$

$c(\mathbf{X})$	A_1	A_2	A_3	A_4	A_5	$\mathbf{P(X)} = \overline{\mathbf{A}}_1 \wedge \mathbf{A_3} \wedge \mathbf{A_4}$
1	**0**	0	**1**	**1**	1	**1**
1	0	1	1	0	1	**0**
1	**0**	0	**1**	**1**	0	**1**
1	0	0	1	0	1	**0**
1	**0**	1	**1**	**1**	0	**1**
0	**1**	0	1	1	0	0
0	**1**	1	1	1	1	0
0	0	1	1	**0**	0	0
0	0	0	**0**	**0**	0	0
0	0	1	**0**	1	1	0

Coverage: $cov(\mathbf{P}) = 3$ **Precision**: $\pi(\mathbf{P}) = 0.8$

Patterns

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.

Wisconsin Breast Cancer Data Set

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

O. L. Mangasarian and W. H. Wolberg: "Cancer diagnosis via linear programming", *SIAM News*, Volume **23**, Number 5, September 1990, pp. 1-18.

- Number of Instances: 699 (status of 15 July 1992)
- Number of Attributes: 9 with integer values between 1 and 10
- Missing attribute values: 16, all for attribute **"Bare-nuclei"**.
- Class distribution:
  - Benign: **458** (65.5%)
  - Malignant: **241** (34.5%)

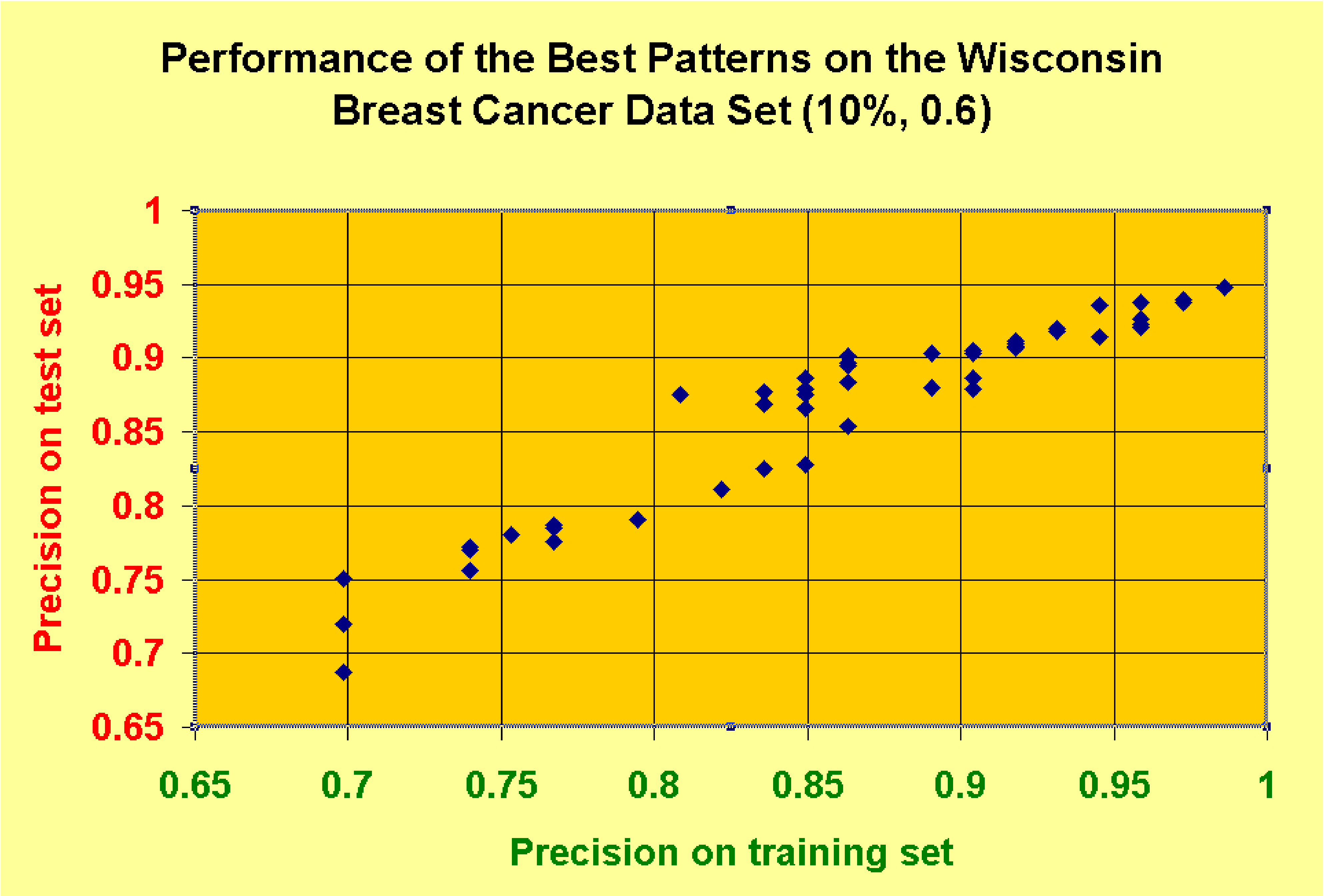
Wisconsin Breast Cancer Data Set

- Training set: **63** records ($\approx 10\%$)

- Attributes: **13** binary

- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.7$: **36** (of degrees $2-5$)

Wisconsin Breast Cancer Data Set



Wisconsin Breast Cancer Data Set

- Training set: **63** records ($\approx$ 10%)

- Attributes: **13** binary

- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.7$: **36** (of degrees $2-5$)

- Single **best** pattern: 95.1%-**classifier**!!

$$\mathbf{P}_1(\mathbf{X}) = (\text{Clump-Thickness} \leq 6) \wedge (\text{Bare-Nuclei} \leq 4) \wedge (\text{Normal-Nucleoli} \leq 3)$$

- Misclassifies only 5 malignant cases (all with missing data!)

- Best results reported in literature: $95-98\%$

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%)

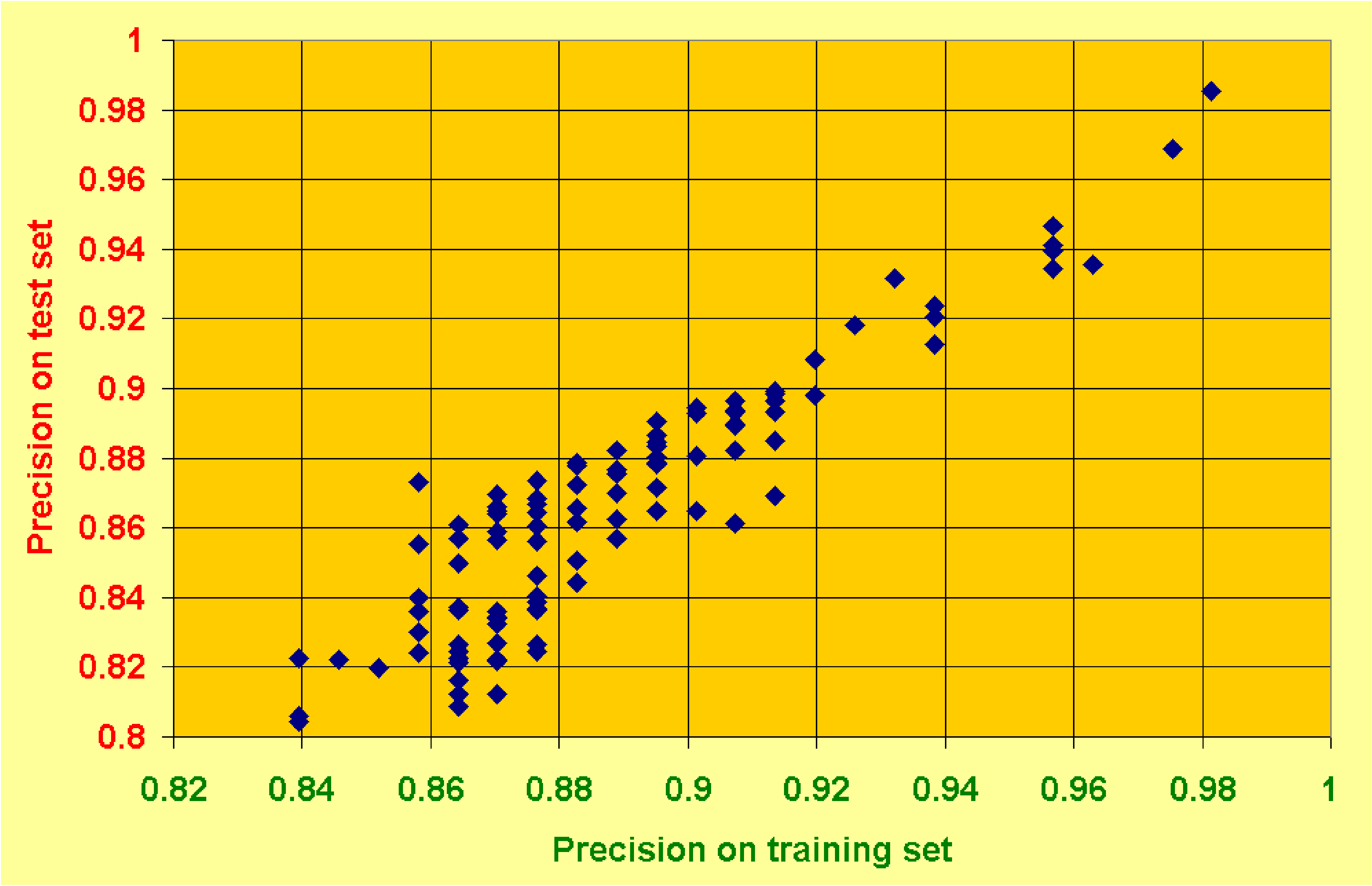
Mushroom Database

- 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: $95-99\%$

Australian Credit Card Data Set

STATLOG: http://www.ncc.up.pt/liacc/ML/statlog/datasets.html

- Number of Instances: 690

- Number of Attributes: 14 (6 numerical and 8 categorical)

- Missing attribute values: none

- Class distribution:

  - positive: **307** (44.5%)

  - negative: **383** (55.5%)

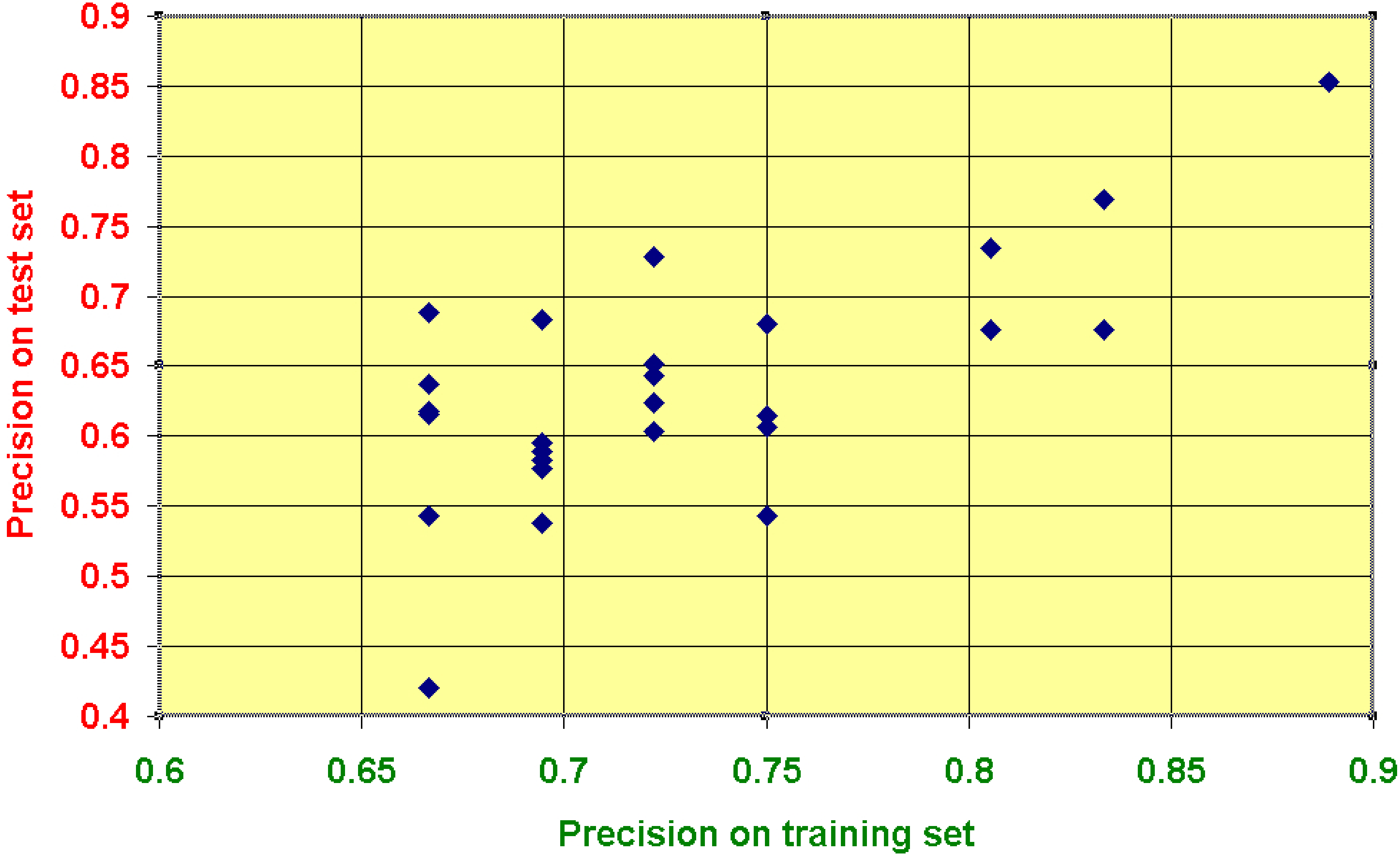
Australian Credit Card Data Set

- Training set: **36** records ($\approx 5\%$)

- Attributes: **12** binary

- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.60$: **26** (of degrees $1-4$)

Australian Credit Card Data Set



Australian Credit Card Data Set

- Training set: **36** records ($\approx 5\%$)

- Attributes: **12** binary

- Number of patterns $\mathbf{P}$ with $\pi(\mathbf{P}) \geq 0.60$: **26** (of degrees $1-4$)

- Single **best** pattern: 85.4%-**classifier**!!

$$\mathbf{P}(\mathbf{X}) = (\mathsf{A8} = 0)$$

- Best results reported in literature: $80-87\%$

STATLOG Results: Australian Credit Card

Algorithm	Error Rate on training	on test
Cal5	0.132	0.131
Itrule	0.162	0.137
LogDisc	0.125	0.141
Discrim	0.139	0.141
Dipol92	0.139	0.141
Radial	0.107	0.145
Cart	0.145	0.145
Best Pattern	**0.111**	**0.146**
Castle	0.144	0.148
Bayes	0.136	0.151
IndCart	0.081	0.152
BackProp	0.087	0.154
C4.5	0.099	0.155
Smart	0.090	0.158
BayTree	0.000	0.171
KNN	0.000	0.181
Ac2	0.000	0.181
NewId	0.000	0.181
LVQ	0.065	0.197
Alloc80	0.194	0.201
Cn2	0.001	0.204
QuaDisc	0.185	0.207
Default	0.440	0.440
Cascade	?	100.0
Kohonen	?	100.0

STATLOG: Vehicle Data Set

STATLOG: http://www.ncc.up.pt/liacc/ML/statlog/datasets.html

- Number of Instances: 846
- Number of Attributes: 18 numerical
- Missing attribute values: none
- Class distribution:
  - OPEL: **212** (25.06%)
  - SAAB: **217** (25.65%)
  - BUS: **218** (25.77%)
  - VAN: **199** (23.52%)

STATLOG: Vehicle Data Set

Averages over the 4 classes

Algorithm	Error Rate on training	on test
QuaDisc	0.085	0.150
Dipol92	0.079	0.151
Alloc80	0.000	0.173
Best Patterns	**0.068**	**0.192**
LogDisc	0.167	0.192
BackProp	0.168	0.207
Discrim	0.202	0.216
Smart	0.062	0.217
Cart	0.284	0.235
C4.5	0.065	0.266
BayTree	0.079	0.271
KNN	0.000	0.275
Cal5	0.068	0.279
Cascade	0.263	0.280
LVQ	0.171	0.287
Ac2	?	0.296
IndCart	0.047	0.298
NewId	0.030	0.298
Radial	0.098	0.307
Cn2	0.018	0.314
Itrule	?	0.324
Kohonen	0.115	0.340
Castle	0.545	0.505
Bayes	0.519	0.558
Default	0.750	0.750

When Best is Good

When **Best** is **Good**

♣ A pattern is **good**, if its precision on the test set is **high**.

When **Best** is **Good**

♣ A pattern is **good**, if its precision on the test set is **high**.

♢ A pattern is **best**, if its precision on the training set is **high**.

When **Best** is **Good**

♣ A pattern is **good**, if its precision on the test set is **high**.

♢ A pattern is **best**, if its precision on the training set is **high**.

♡ **Empirical evidence** suggests that **best** patterns generalize **very well**.

The average precision on the test set, as a function of the precision on the training set, is increasing; its variance, as a function of the precision on the training set, is decreasing.

When Best is Good

♣ A pattern is **good**, if its precision on the test set is **high**.

♢ A pattern is **best**, if its precision on the training set is **high**.

♡ **Empirical evidence** suggests that **best** patterns generalize **very well**.

The average precision on the test set, as a function of the precision on the training set, is increasing; its variance, as a function of the precision on the training set, is decreasing.

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

Logical Analysis of Data

- Binarization of numerical attributes

- Binarization of categorical attributes

- Pattern generation

- Theory building

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.

Software

Software available from Christhophe Meyer's LAD site

http://www.gerad.ca/~christop/LAD_en.html

1) **Datascope** (package written by Sorin Alexe in Visual Basic for Windows)

2) **LADTools** (program in C++ written by Eddy Mayoraz; may need CPLEX)

3) **Ladoscope** Gang (a set of programs written in Objective Caml by Pierre Lemaire)

Software available by request from E. Boros

4) **PLAD** (a PERL LAD Tool package, "use at your own risk research-ware")

References

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

E. Boros, Y. Crama, P.L. Hammer, T. Ibaraki, A. Kogan and K. Makino, Logical Analysis of Data: Classification with justification, Working paper HEC-ULg 200902-02.

Y. Crama and P. L. Hammer, *Boolean Functions Theory, Algorithms and Applications*, Cambridge University Press, Cambridge, U.K., to appear, 2010.

Y. Crama and P. L. Hammer, eds., *Boolean Models and Methods in Mathematics, Computer Science, and Engineering,* Cambridge University Press, Cambridge, U.K., 2010.

References

E. Boros, V. Gurvich, P.L. Hammer, T. Ibaraki and A. Kogan. Decomposability of partially defined Boolean functions. Discrete Applied Mathematics 62 pp. 51-75 (1995).

E. Boros, P.L. Hammer, T. Ibaraki and A. Kogan. Logical analysis of numerical data. Mathematical Programming 79 pp. 163-190 (August, 1997).

E. Boros, T. Ibaraki and K. Makino. Error-free and best-fit extensions of partially defined Boolean functions. Information and Computation 140 (2) pp. 254-283 (February, 1998).

E. Boros, T. Ibaraki and K. Makino. Logical analysis of binary data with missing bits. Artificial Intelligence 107 (2) pp. 219-263 (February, 1999).

E. Boros, P.L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz and I. Muchnik. An Implementation of Logical Analysis of Data. IEEE Transactions on Knowledge and Data Engineering 12 (2) pp. 292-306 (May, 2000).

References

E. Boros, T. Ibaraki and K. Makino. Variations on Extending Partially Defined Boolean Functions with Missing Bits. Information and Computation, 180 pp. 53-70 (January, 2003).

E. Boros, T. Horiyama, T. Ibaraki, K. Makino and M. Yagiura. Finding Essential Attributes from Binary Data. Annals of Mathematics and Artificial Intelligence, 39(3) (2003) pp. 223-257.

E. Boros and V. Menkov. Exact and approximate discrete optimization algorithms for finding useful disjunctions of categorical predicates in data analysis. Discrete Applied Mathematics, 144 (2004) pp. 43-58.

Gabriela Alexe, Sorin Alexe, Tibrius O. Bonates, and Alexander Kogan. Logical analysis of data the vision of Peter L. Hammer. Annals of Mathematics and Artificial Intelligence 49 (1-4) pp. 265-312 (April 2007).

T. O. Bonates and P. L. Hammer. Large Margin LAD Classifiers. RUTCOR Research Reports 22-2007.