# How to...

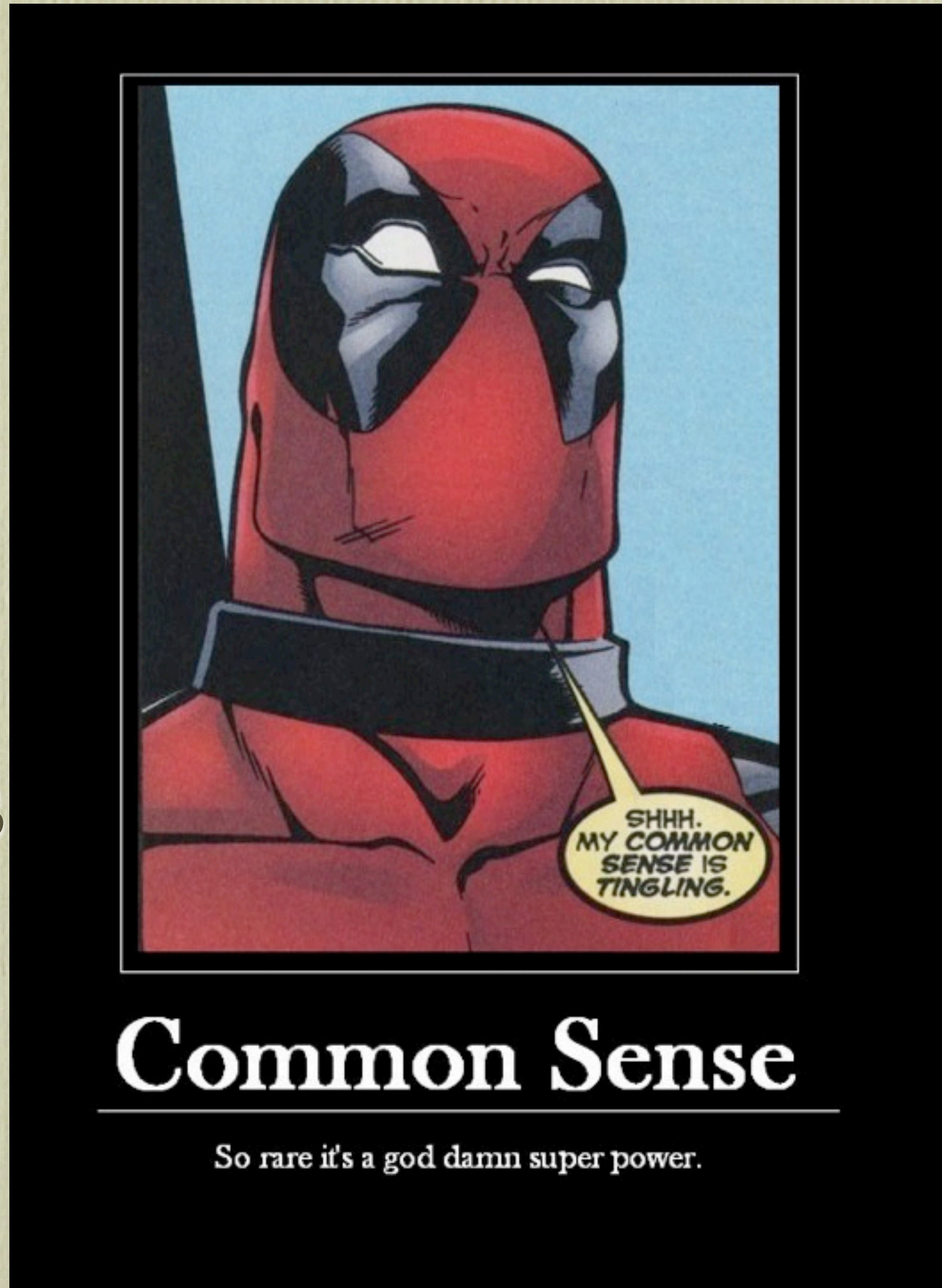
build a poster

Albrecht Zimmermann
26.03.2012
KU Leuven, Belgium

# Disclaimer

**It's just common sense**

# It’s


SHHH. MY COMMON SENSE IS TINGLING.
Common Sense
So rare it's a god damn super power.

se

# Typical Poster

Paper

Size A0 or A1

Fixed to Wall/Posterboard

Might be framed

# Purpose

Displayed **with others** in a room

# Purpose

Displayed **with others** in a room

# Purpose

Displayed **with others** in a room

*(often* ***many*** *people, food/drink)*

# Purpose

Displayed **with others** in a room

*(often **many** people, food/drink)*

Communicate **scientific** information

Ideally **stand-alone**

**Enhanced** by explanation

# Why make one?

Interesting research yet not enough for presentation

Augmenting given presentation

*(parallel tracks, not enough time for details/questions)*

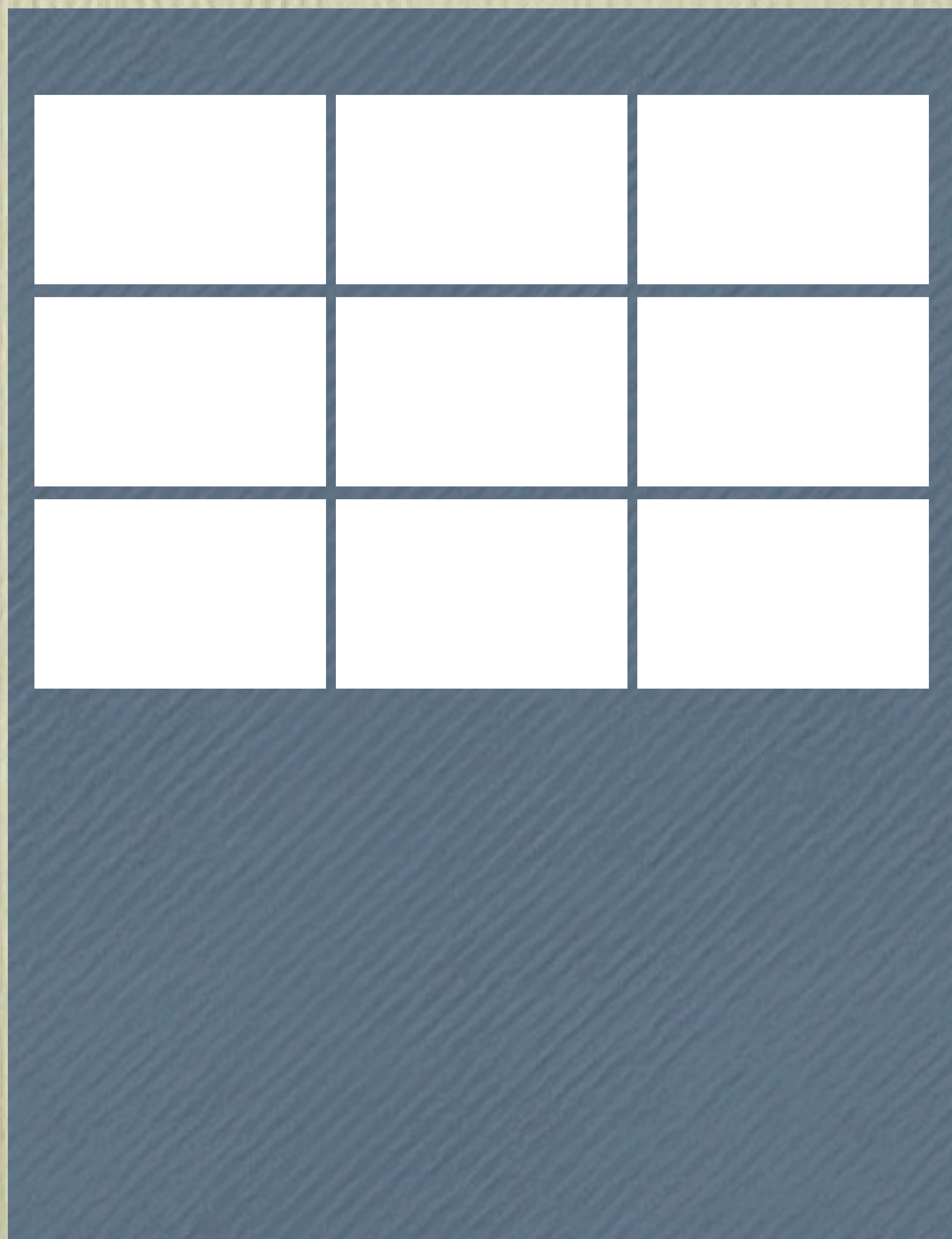
Relieves from presentation duty

*(weak presenter, late riser)*

Because you have to

*(organizer/supervisor requirement)*

# Some Approaches



Take your slides and tape them to a wall.

Very efficient!

# Poster != Slides

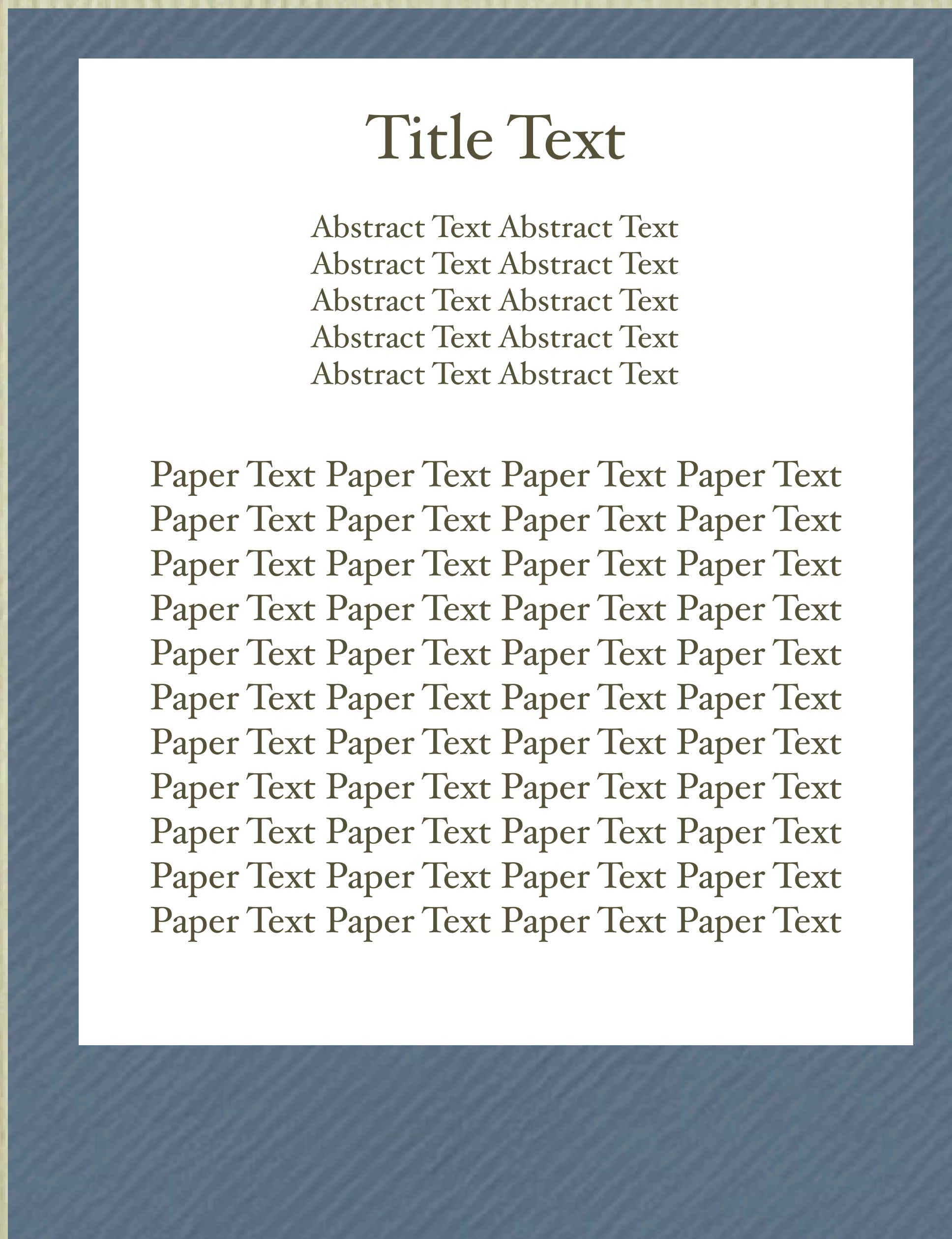
Slides **always** explained

less completeness needed

Slides **part** of a talk

guaranteed time slot, can focus on details

# Some Approaches cont.




Summarize your paper on the poster.

Very informative!

# Poster != Paper

Paper **doesn't have to engage**

*at distance* *against competition*

Paper will be **read over hours/days**

Paper normally **not explained** by author

# How should it achieve its purpose?

Get (**non**-informed) audience interested

Get **informed** audience interested

Make used approach **understandable**

**Justify** approach/**reward** audience

# What goes on it?

- **Title** (name, affiliation)

**Get** the (un)informed ones

- **Problem/question** to be addressed

**Keep** the informed ones

- **Solution** developed

**Navigate** the solution

- **Results** obtained/conclusion drawn

**Reward!**

# How to...

get it right

1 — 2 — 3 — 4

# Deliver the idea

- **Title** — Get

  large enough to read at distance, right colors

- **State problem concisely** — Keep

  colors, size, positioning

- **State main insight concisely** — Navigate

  if possible, give short description

- **Results** — Reward

  as above, at least make clear where they are shown

# Recognizability

- **Title**

  large enough to read at distance, right colors

- **State problem concisely**

  colors, size, positioning

**ID3 - A Decision Tree Learner**

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

Keep

1 2 3 4

# Give Context... but not too much

## Statistical Pattern Recognition

A specific instantiation of the method for structured data is investigated. We combine a graph mining algorithm and a graph decomposition SVM algorithm to exploit the different inductive biases.

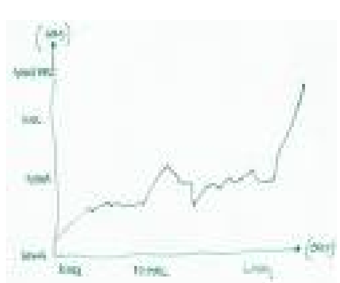
*A specific instantiation of the method for structured data is investigated. We combine a graph mining algorithm and a graph decomposition SVM algorithm to exploit the different inductive biases.*

### Algorithm

This paper proposes a meta-algorithm that allows a tight integration of pattern mining and robust statistical learning algorithms. A predictive model is build in an incremental fashion alternating a pattern mining and a learning phase. A novel, mutually recursive processing strategy allows the error of the incremental model to guide the mining algorithm.

Experimental results are reported on both artificial test cases and on a real world bio-informatics classification task. The incremental approach compares favorably with a pure graph decomposition kernel SVM as well as with a linear predictor based on mined.

### Experiments



This paper proposes a meta-algorithm that allows a tight integration of pattern mining and robust statistical learning algorithms. A predictive model is build in an incremental fashion alternating a pattern mining phase and a learning phase. A novel, mutually recursive processing strategy allows the error of the incremental model to guide the mining algorithm.

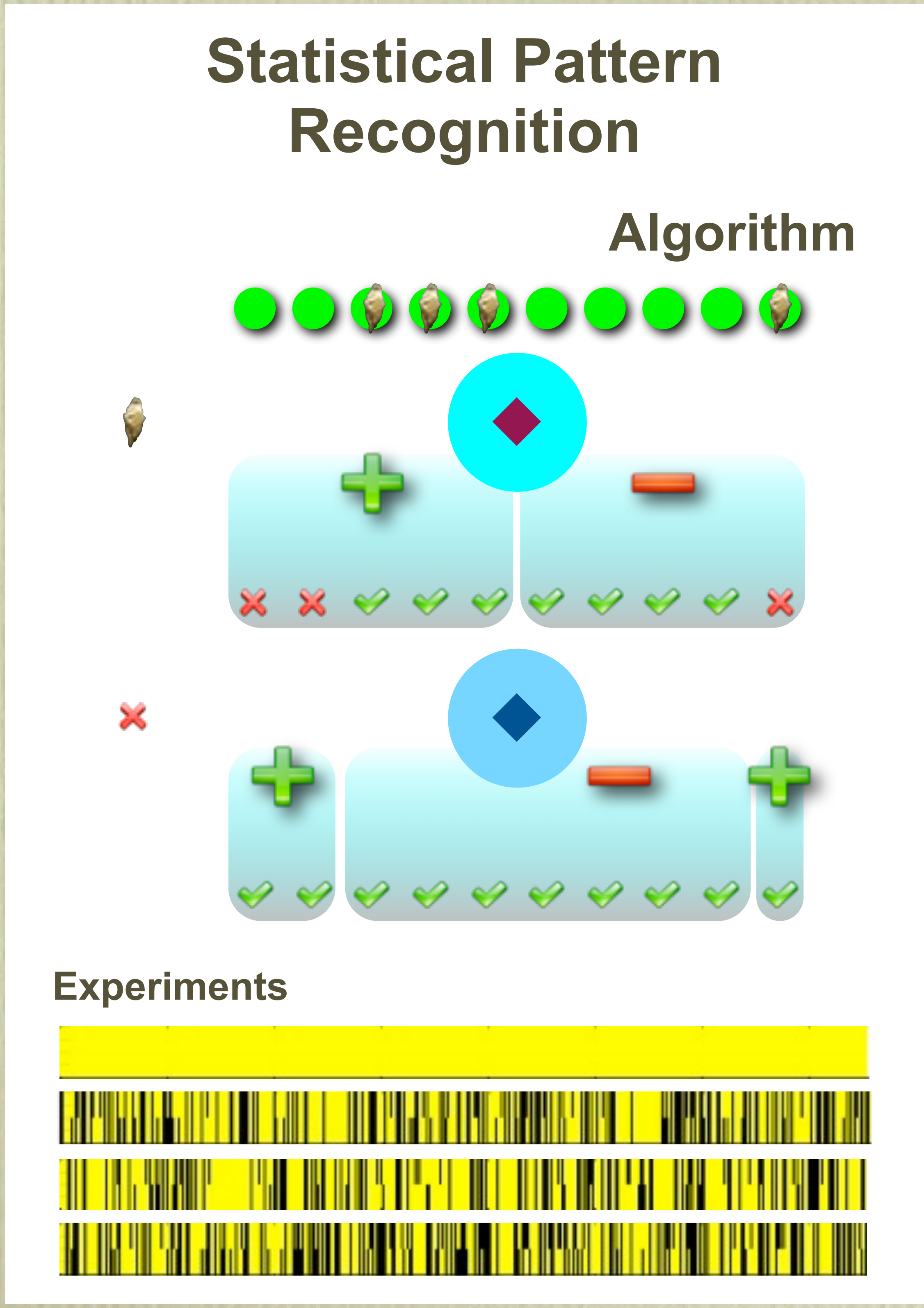
- Paper-style
- **no-one will read this**

$\Rightarrow$ not stand-alone

Keep

1 2 3 4

# …and not too little context!




- Slide-style
- always **needs explanation**

⇒ not stand-alone

Keep

1 2 3 4

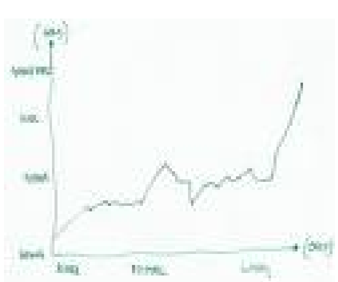
# Give details… but not too many

## Statistical Pattern Recognition

Experimental results are reported on both artificial test cases and on a real world bio-informatics classification task. The incremental approach compares favorably with a pure graph decomposition kernel SVM as well as with a linear predictor based on mined.

*A specific instantiation of the method for structured data is investigated. We combine a graph mining algorithm and a graph decomposition SVM algorithm to exploit the different inductive biases.*

A specific instantiation of the method for structured data is investigated. We combine a graph mining algorithm and a graph decomposition SVM algorithm to exploit the different inductive biases.



**Experiments**

### Algorithm

```
int GGraphFunction::getNextMatch() {
     for (int pi = 0; pi < this->_height; ++pi)
#ifdef boundary
     if (this->_nodeBound[pi] != GMatrixFree)
     {
          this->_nodeMatchCount[pi] = 1;     } else
#endif
     {
          int cnt = 0;
          for (int gi = 0; gi < this->_width; ++gi)
          if (this->getM(pi, gi) == 1)
               { cnt++; this->_matrixMask[gi] =
               GMatrixMark; }
          else
          if (pi == 0)
               this->_matrixMask[gi] = GMatrixFree;
          this->_nodeMatchCount[pi] = cnt;
          if (cnt == 0) return -2;
     }
     int nmc = this->_nodeMatchCount.size();
     for (int i = 0; i < nmc; ++i)
     {
          if (_nodeMatchCount[i] == 0) return -2;
          if ((_nodeMatchCount[i] < min)) {
               min = this->_nodeMatchCount[i];
               pos = i;
          }
     }
     return pos;
}
```

- Details in the paper
- explanation not needed here

**Keep**

**Navigate**

1 2 3 4

Title → 

Problem →

Insight →

Results →

Graph

Text

Formulae

Chart

# ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

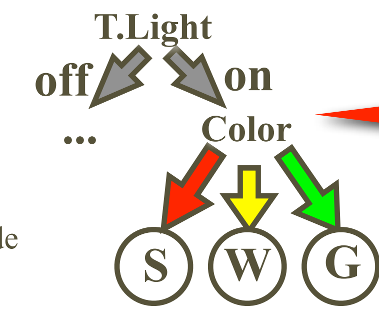
**Representation:**

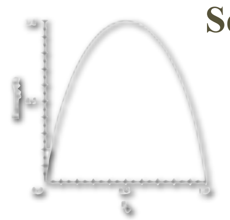
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**



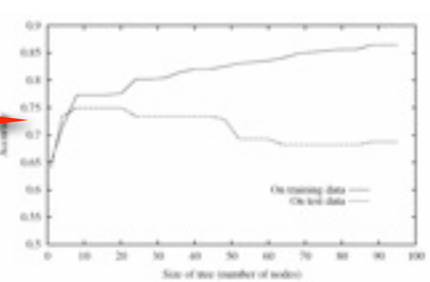
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

Entropy(S) = - ($p_+ \, log_2 \, p_+$) - ($p_- \, log_2 \, p_-$)

Gain(S, A) = Entropy(S) - $\sum_{Values(A)} \frac{|S_v|}{|S|}$Entropy($S_v$)

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

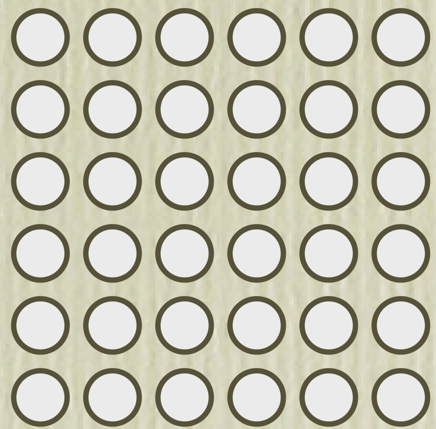
**References: Slides by Luc De Raedt, Freiburg 2003**

# Elements

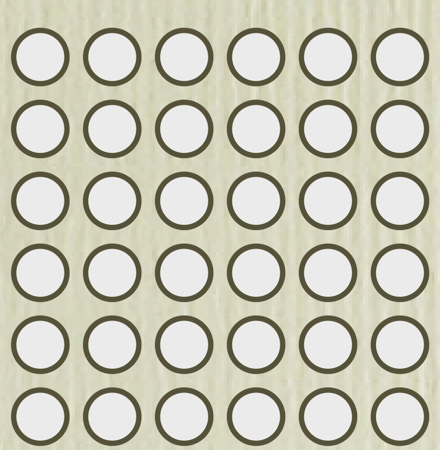
- **Content**
  - Text, Graphs, Formulae, Images
- **Grouping**
  - Shapes, Colors, Icons, Whitespace
- **Highlighting**
  - Size, Colors, Whitespace, Position, Icons

# Organizing without Cluttering

- Law of Proximity:

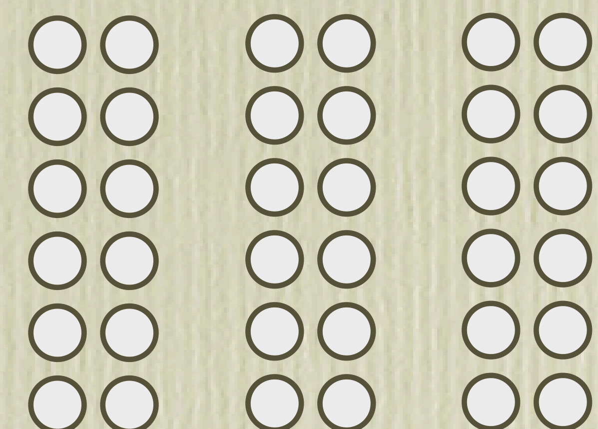


- Law of Similarity:

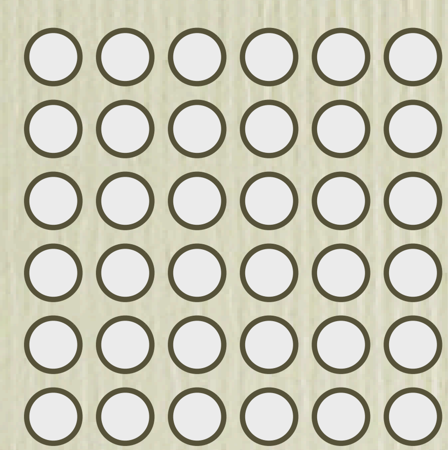


# Organizing without Cluttering

- Law of Proximity:

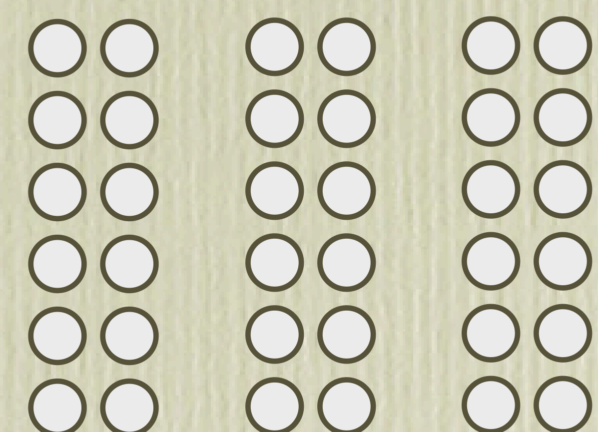


- Law of Similarity:

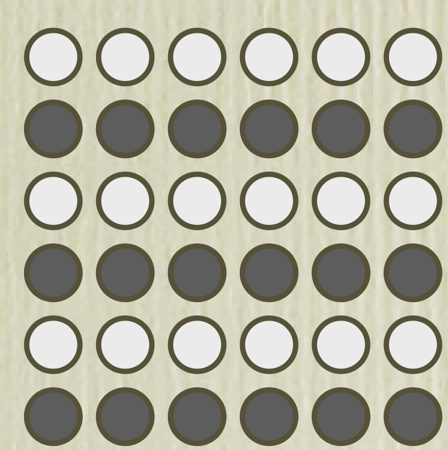


# Organizing without Cluttering

- Law of Proximity:



- Law of Similarity:



# ID3 - A Decision Tree Learner

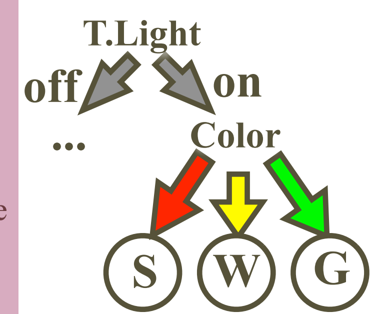
Björn Bringmann / Albrecht Zimmermann, KU Leuven

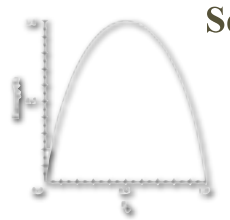
**Given** a set of classified training examples
**Find** a model to predict unseen examples

Representation:
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

Algorithm:
1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**



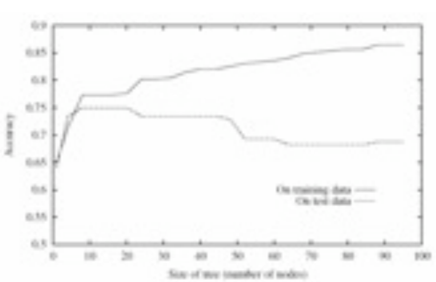
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

Entropy(S) = $-(p_+ \, log_2 \, p_+) - (p_- \, log_2 \, p_-)$

Gain(S, A) = Entropy(S) $- \sum_{Values(A)} \frac{|S_v|}{|S|}$ Entropy($S_v$)

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

# ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

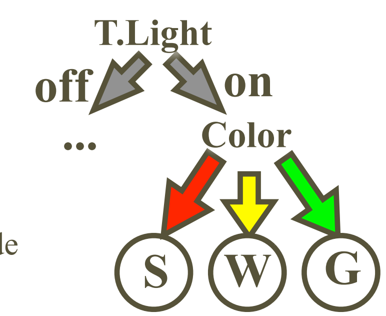
**Representation:**

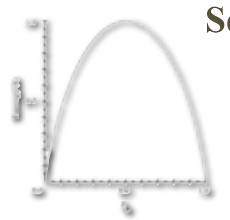
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**

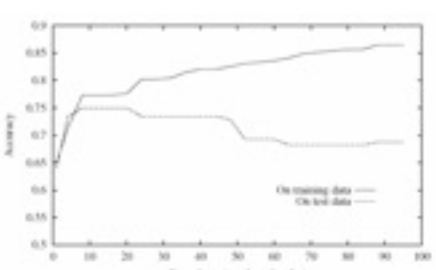


**Selecting best decision attribute:**

Calculate Information Gain for each attribute
Entropy(S) = $-(p_+ \, log_2 \, p_+) - (p_- \, log_2 \, p_-)$
Gain(S, A) = Entropy(S) $- \sum_{Values(A)} \frac{|S_v|}{|S|}$ Entropy($S_v$)

**Experiments / Conclusions**



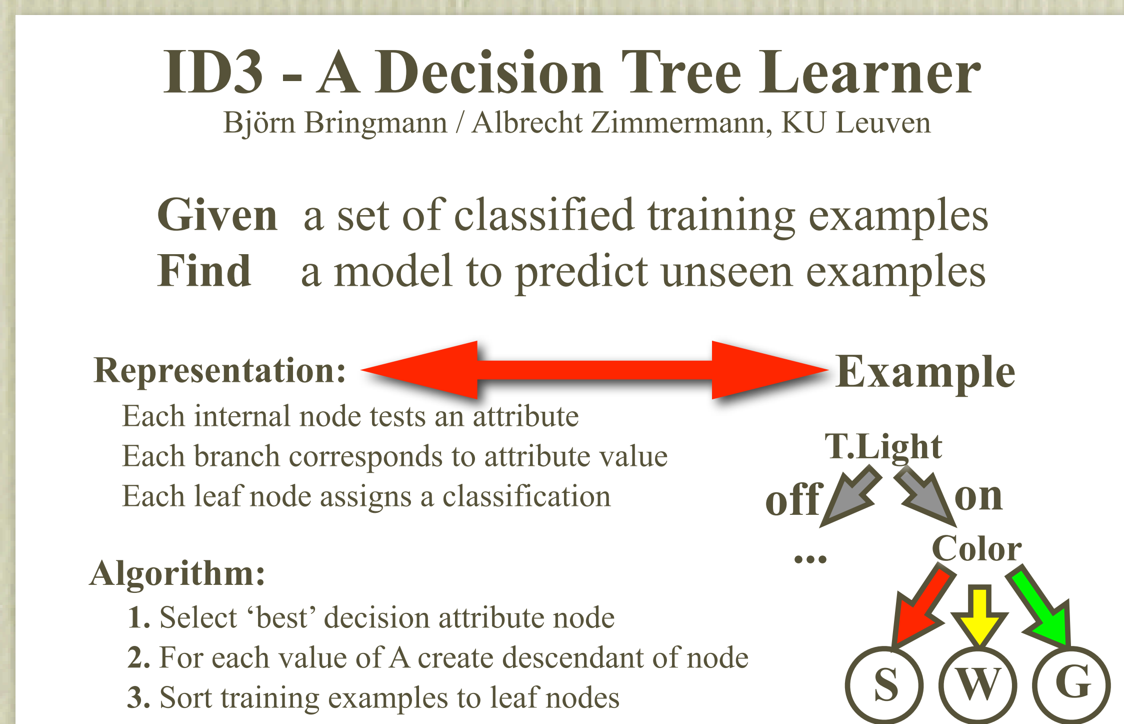
- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

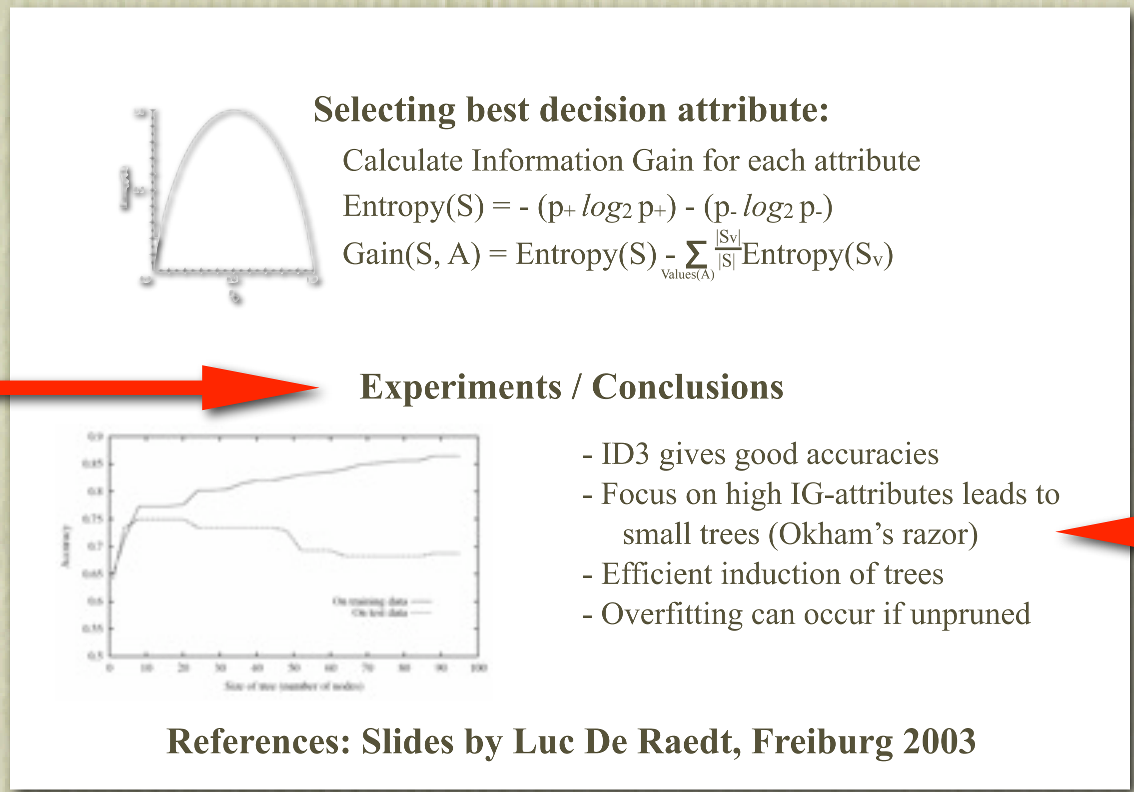
# Order

- Westerners expect to read **left to right, top to bottom**

⇒ Start related blocks at same level



# Point out conclusions




- Results need some detail to be understood
- Conclusion should be remembered!
  Draw attention, make sure it's read (stand-alone)

# The final package

**Title**

**Problem**

**Insight**

**Result**

## ID3 - A Decision Tree Learner

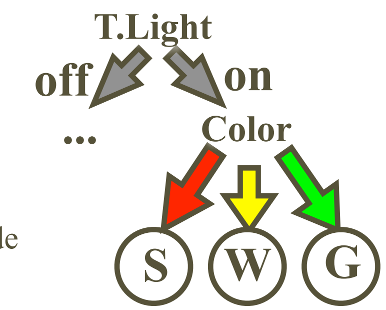
Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

**Representation:**

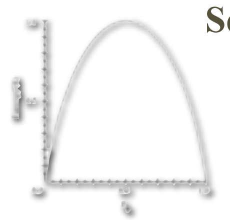
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Example**




**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node



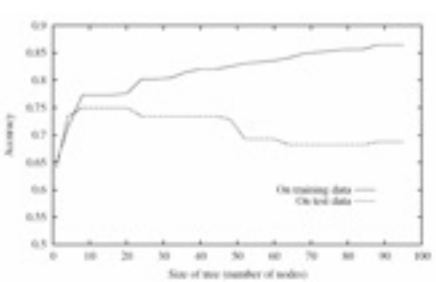
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

$\text{Entropy}(S) = -\,(p_+ \, log_2 \, p_+) - (p_- \, log_2 \, p_-)$

$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{Values(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

# The final package

**Size**

## ID3 - A Decision Tree Learner

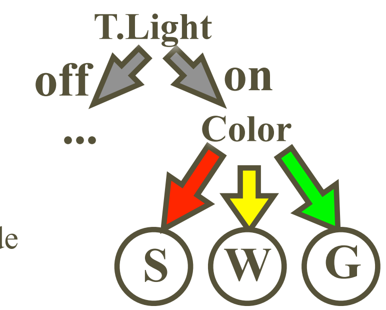
Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

**Representation:**

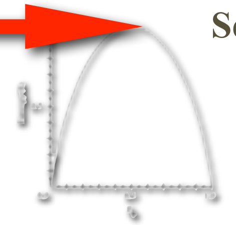
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Example**




**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node



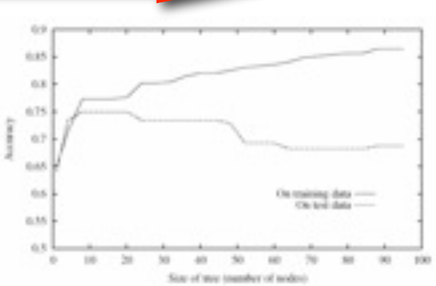
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

$\text{Entropy}(S) = -(p_+ \log_2 p_+) - (p_- \log_2 p_-)$

$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{\text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

# The final package

## ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

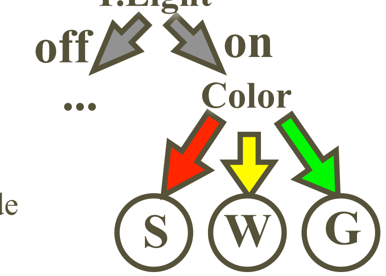
**Representation:**

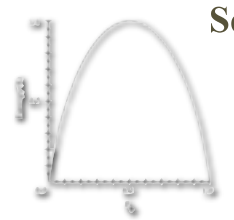
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**



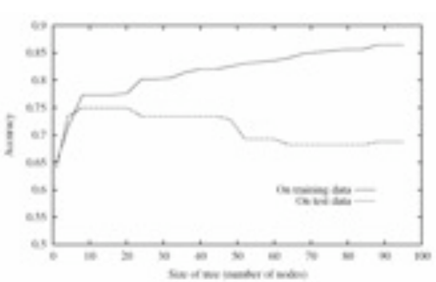
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

$\text{Entropy}(S) = -(p_+ \, log_2 \, p_+) - (p_- \, log_2 \, p_-)$

$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{Values(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

# The final package

## ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

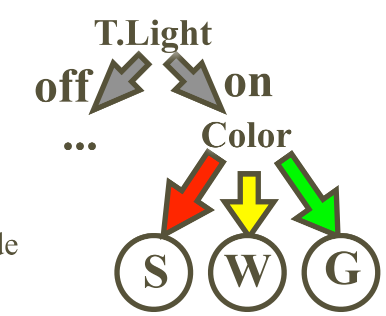
**Representation:**

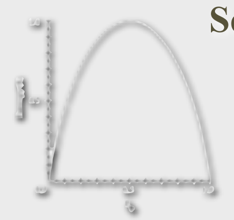
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**



**Selecting best decision attribute:**

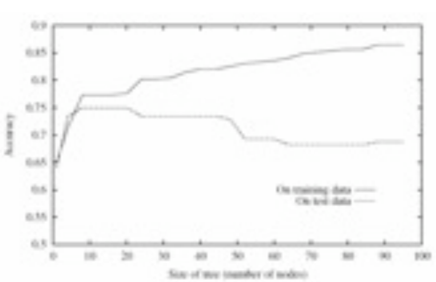
Calculate Information Gain for each attribute

Entropy(S) = - ($p_+ \, log_2 \, p_+$) - ($p_- \, log_2 \, p_-$)

Gain(S, A) = Entropy(S) - $\sum_{Values(A)} \frac{|S_v|}{|S|}$ Entropy($S_v$)

Shape

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

# The final package

## ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

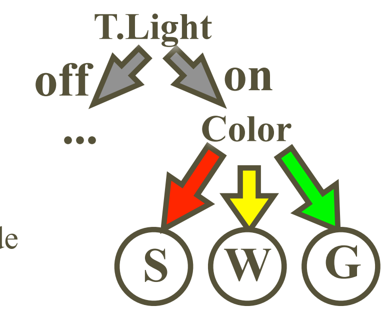
**Representation:**

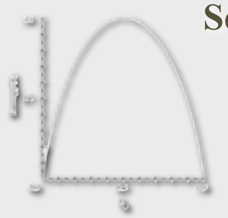
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

1. Select 'best' decision attribute node
2. For each value of A create descendant of node
3. Sort training examples to leaf nodes
4. Repeat for each unpure leaf node

**Example**



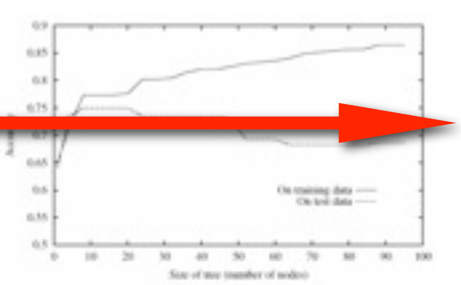
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

$\mathrm{Entropy}(S) = -(p_+ \log_2 p_+) - (p_- \log_2 p_-)$

$\mathrm{Gain}(S, A) = \mathrm{Entropy}(S) - \sum_{\mathrm{Values}(A)} \frac{|S_v|}{|S|} \mathrm{Entropy}(S_v)$

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

Icons

# The final package

## ID3 - A Decision Tree Learner

Björn Bringmann / Albrecht Zimmermann, KU Leuven

**Given** a set of classified training examples
**Find** a model to predict unseen examples

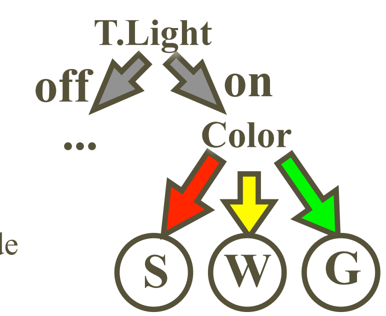
**Representation:**

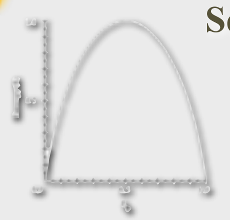
Each internal node tests an attribute
Each branch corresponds to attribute value
Each leaf node assigns a classification

**Algorithm:**

**1.** Select 'best' decision attribute node
**2.** For each value of A create descendant of node
**3.** Sort training examples to leaf nodes
**4.** Repeat for each unpure leaf node

**Example**



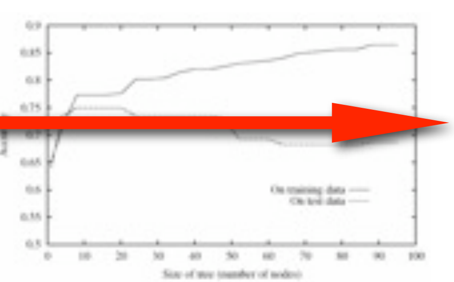
**Selecting best decision attribute:**

Calculate Information Gain for each attribute

$\text{Entropy}(S) = -(p_+ \, log_2 \, p_+) - (p_- \, log_2 \, p_-)$

$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{\text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$

**Experiments / Conclusions**



- ID3 gives good accuracies
- Focus on high IG-attributes leads to small trees (Okham's razor)
- Efficient induction of trees
- Overfitting can occur if unpruned

**References: Slides by Luc De Raedt, Freiburg 2003**

Icons

# Highlighting

- if everybody is special, nobody is -

Do not highlight everything!

# Highlighting

- if everybody is special, nobody is -

Do not highlight everything!

# Highlighting

- if everybody is special, nobody is -

**- Dashiell Parr ( The Incredibles )**

Do not highlight everybody and their mama!

# Hints

- First draw a few drafts on **PAPER**
- Begin with arranging main parts, fill more details later
- **“Restraint enforces discipline”**
  - **Minimum** fontsize of 18, at most 3 fonts
  - Stick to few colors, subtle ones (pastel, gradients)
- **Work with whitespace** - Empty parts **are not** wasted space

Hit ***SAVE*** very regularly

# Tools

## Presentation Software

(PowerPoint, Keynote, ...)

Provide *simple* elements, drag&drop, alignment supported

## LaTeX

*For LaTeX Gurus?*

*It's a text-setting system, not a drawing tool*

## Vector Oriented Drawing Software

(CorelDraw, OmniGraffl, Illustrator, ...)

***Best option***, but requires understanding of the tool.

Different layers for different parts of poster.

# Tools

## Presentation Software

(PowerPoint, Keynote, ...)

Pro rted

**NEVER...**

try to use every feature of the Software

## Vector Oriented Drawing Software

(CorelDraw, OmniGraffl, Illustrator, ...)

***Best option***, but requires understanding of the tool.

Different layers for different parts of poster.

# Conclusions

- **Draw attention** but DONT SCREAM
- Order & structure the Content
- Explanation supporting ***and*** stand-alone

the end

**http://people.cs.kuleuven.be/~albrecht.zimmermann/presentations/how-to-make-a-poster.pdf**

this page is intentionally left blank