Part II. Predictive DM techniques

- Decision tree learning
- Bayesian Classifier
- Rule learning
- Evaluation

Predictive DM - Classification

- data are objects, characterized with attributes - they belong to different classes (discrete labels)
- given objects described with attribute values, induce a model to predict different classes
- decision trees, if-then rules, discriminant analysis, ...

Predictive DM - classification formulated as a machine learning task

- Given a set of labeled **training examples** (n-tuples of attribute values, labeled by class name)

	A1	A2	A3	Class
example1	$v_{1,1}$	$v_{1,2}$	$v_{1,3}$	C_1
example2	$v_{2,1}$	$v_{2,2}$	$v_{2,3}$	C_2

. .

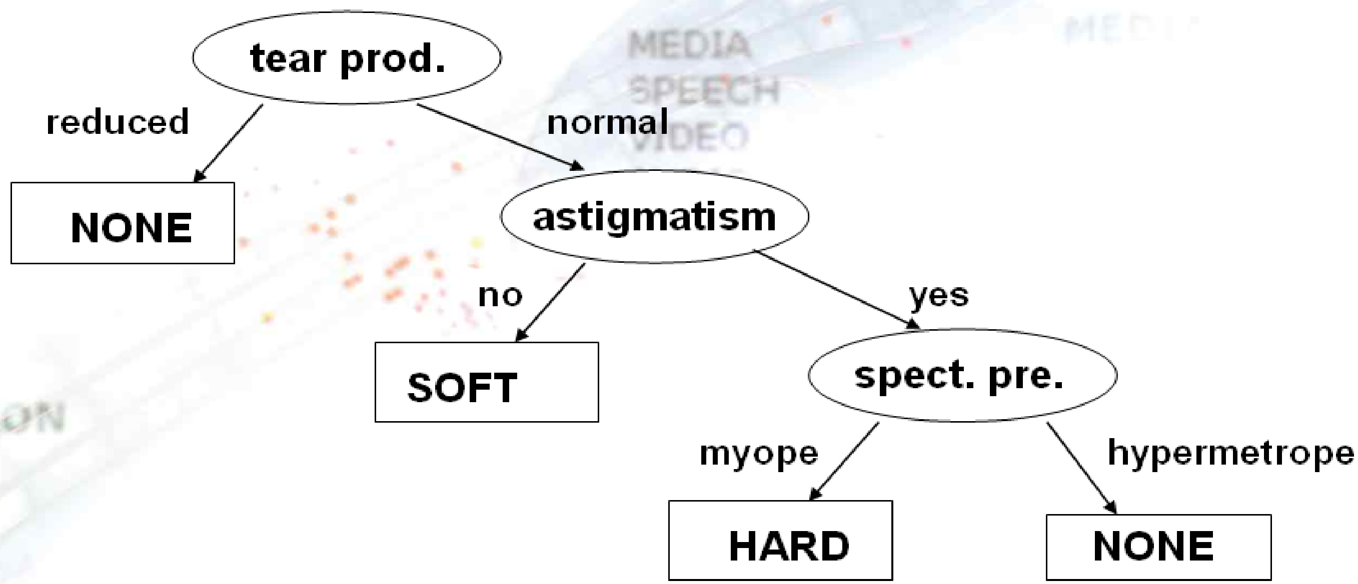
- Performing generalization from examples (induction)
- Find a **hypothesis** (a decision tree or classification rules) which explains the training examples, e.g. decision trees or classification rules of the form:

IF $(A_i = v_{i,k})$ & $(A_j = v_{j,l})$ & ... THEN Class = C_n

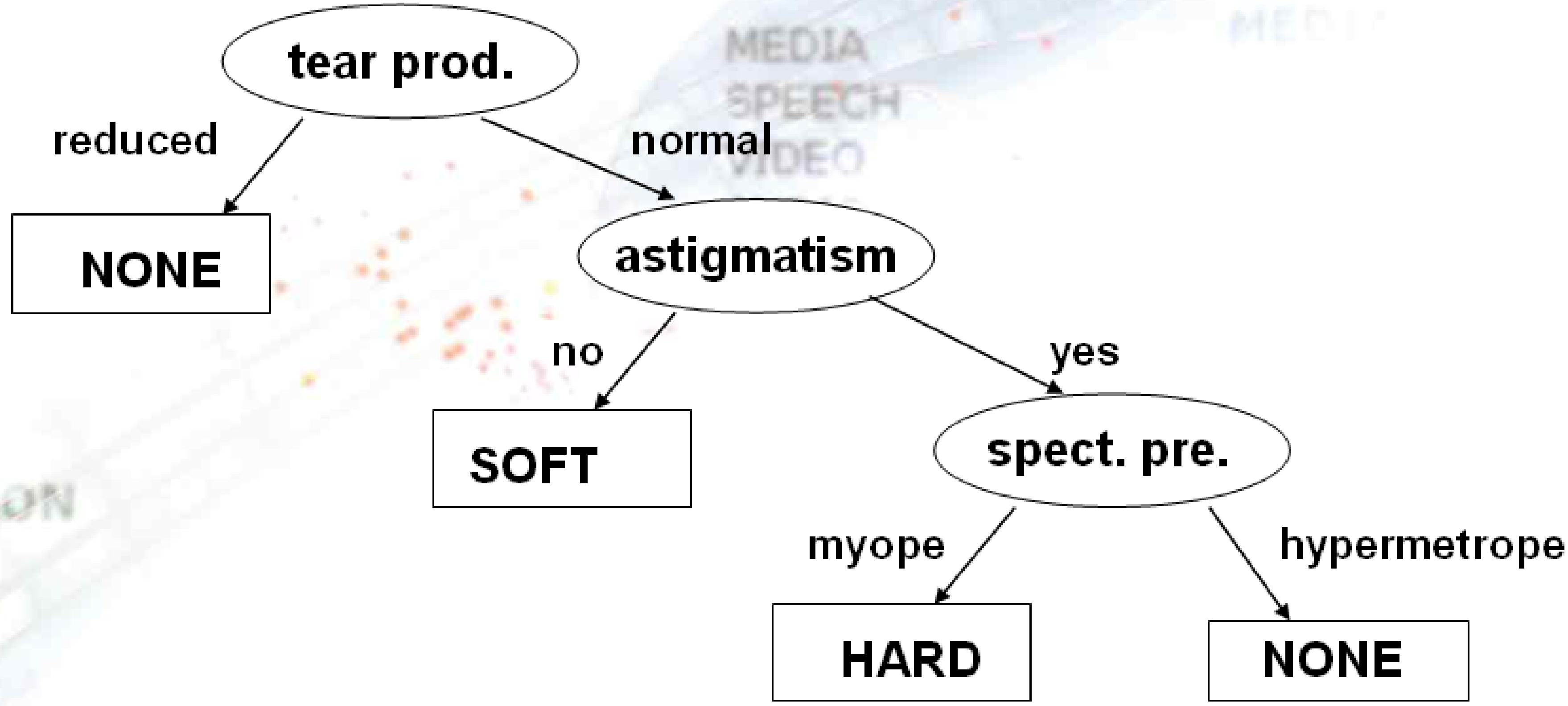
Decision Tree Learning

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

Data Mining

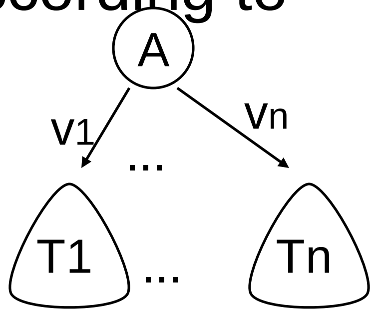


Decision Tree classifier



Decision tree learning algorithm

- ID3 (Quinlan 1979), CART (Breiman et al. 1984), C4.5, J48 in WEKA, ...
  – create the root node of the tree
  – if all examples from S belong to the same class Cj
    - then label the root with Cj
  – else
    - select the ‘most informative’ attribute **A** with values **v1, v2, … vn**
    - divide training set **S** into **S1,… , Sn** according to values **v1,…,vn**
    - recursively build sub-trees **T1,…,Tn** for **S1,…,Sn**



Decision tree search heuristics

- Central choice in decision tree algorithms: Which attribute to test at each node in the tree ? The attribute that is most useful for classifying examples.
- Define a statistical property, called **information gain**, measuring how well a given attribute separates the training examples w.r.t their target classification.
- First define a measure commonly used in information theory, called **entropy**, to characterize the (im)purity of an arbitrary collection of examples.

Entropy

- **S** - training set, $\mathbf{C_1,\ldots,C_N}$ - classes
- **Entropy E(S)** – measure of the impurity of training set S

$$E(S) = -\sum_{c=1}^{N} p_c . \log_2 p_c$$

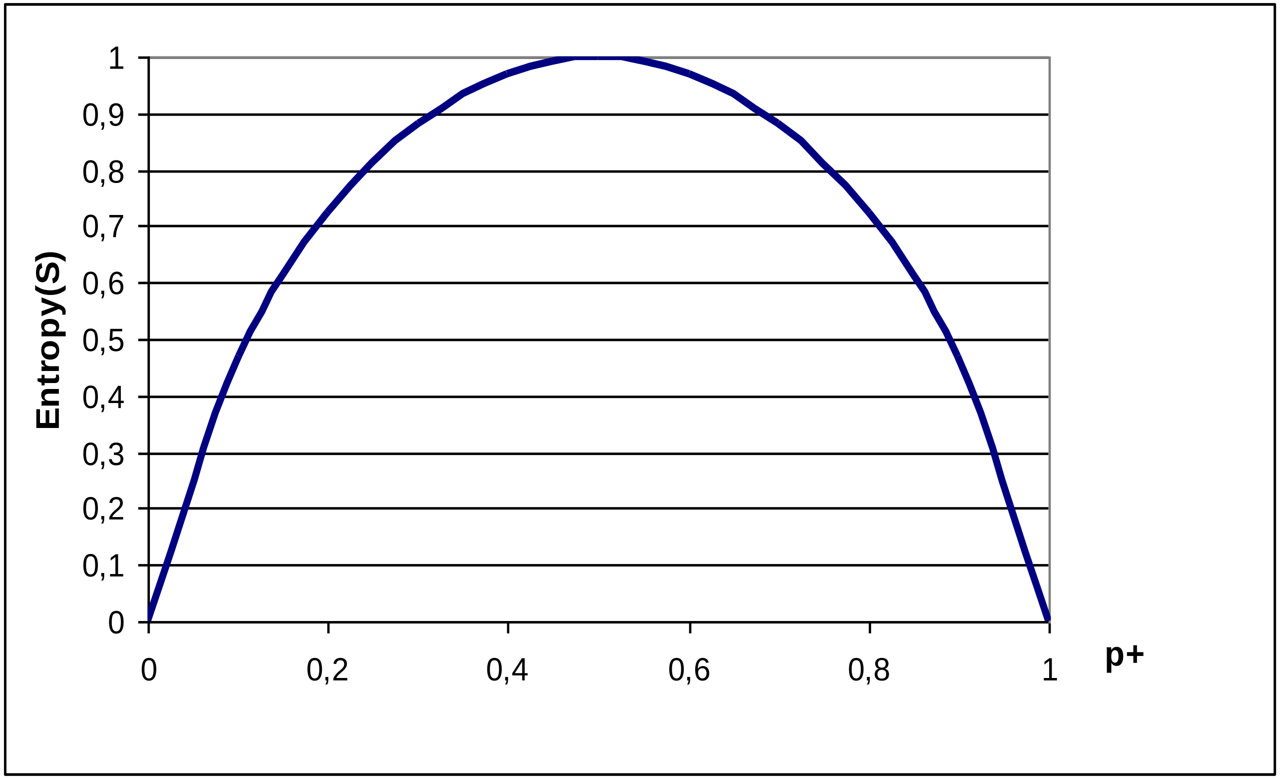
$\mathbf{p_c}$ - prior probability of class $\mathbf{C_c}$
(relative frequency of $\mathbf{C_c}$ in **S**)

- Entropy in binary classification problems

$$\mathbf{E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-}$$

Entropy

- $E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-$
- The entropy function relative to a Boolean classification, as the proportion **p_+** of positive examples varies between 0 and 1



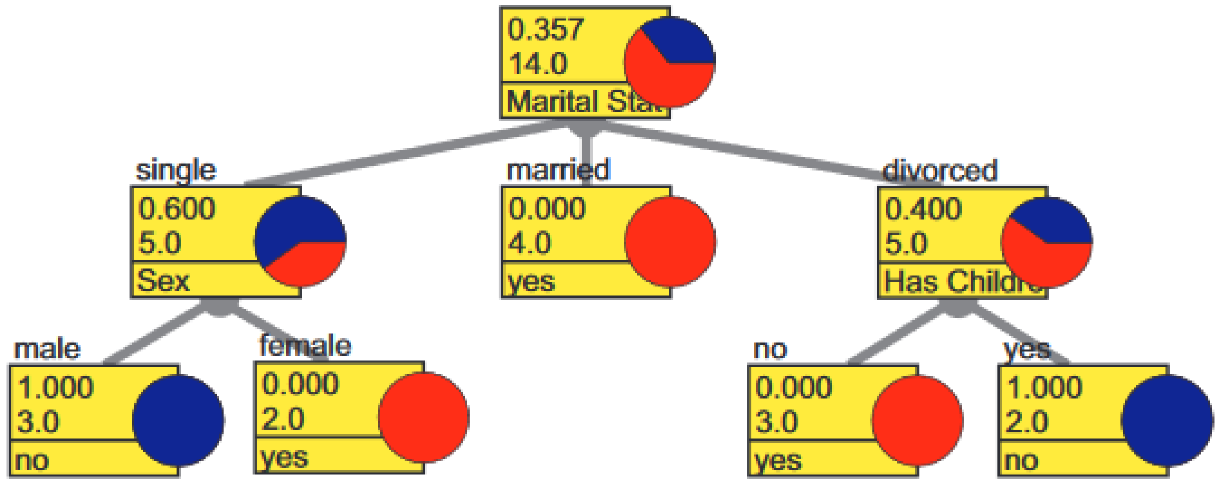
Entropy – why ?

- **Entropy E(S) =** expected amount of information (in bits) needed to assign a class to a randomly drawn object in S (under the optimal, shortest-length code)
- Why ?
- Information theory: optimal length code assigns $-\log_2 p$ bits to a message having probability p
- So, in binary classification problems, the expected number of bits to encode + or – of a random member of S is:

$$p_+ (- \log_2 p_+) + p_- (- \log_2 p_-) = - p_+ \log_2 p_+ - p_- \log_2 p_-$$

Binary classification problem

Education	Marital Status	Sex	Has Children	Approved
primary	single	male	no	no
primary	single	male	yes	no
primary	married	male	no	yes
university	divorced	female	no	yes
university	married	female	yes	yes
secondary	single	male	no	no
university	single	female	no	yes
secondary	divorced	female	no	yes
secondary	single	female	yes	yes
secondary	married	male	yes	yes
primary	married	female	no	yes
secondary	divorced	male	yes	no
university	divorced	female	yes	no
secondary	divorced	male	no	yes



Entropy – example calculation

- Training set S: 14 examples (9 pos., 5 neg.)
- Notation: S = [9+, 5-]
- $E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-$
- Computing entropy, if probability is estimated by relative frequency

$$E(S) = -\left(\frac{|S_+|}{|S|} \cdot \log \frac{|S_+|}{|S|} \right) - \left(\frac{|S_-|}{|S|} \cdot \log \frac{|S_-|}{|S|} \right)$$

- $E([9+,5-]) = - (9/14) \log_2(9/14) - (5/14) \log_2(5/14)$

  $= 0.940$

Information gain search heuristic

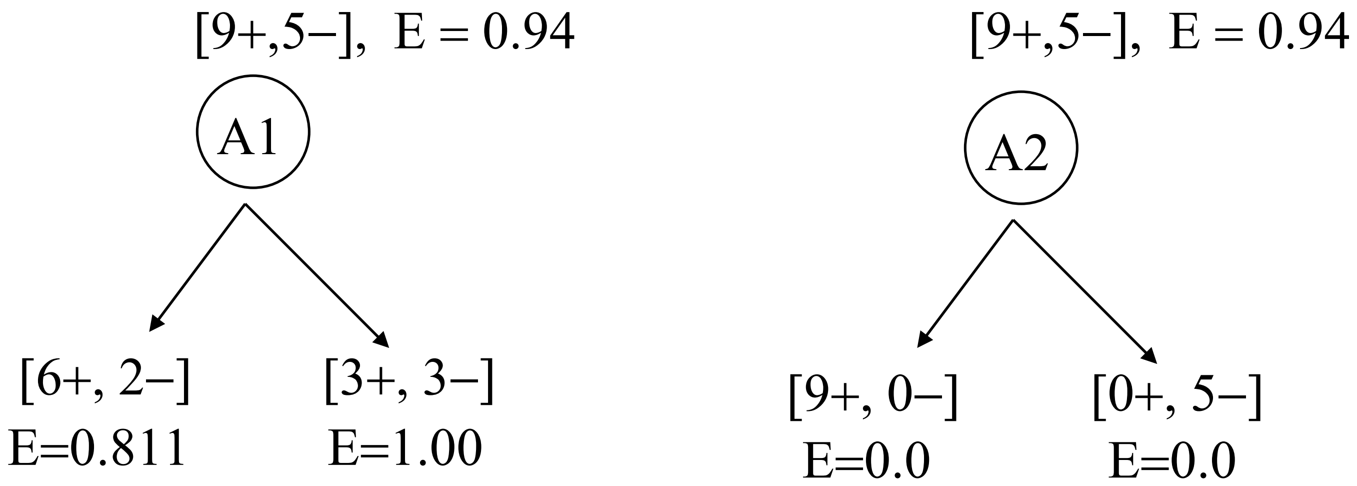
- **Information gain** measure is aimed to minimize the number of tests needed for the classification of a new object
- **Gain(S,A)** – expected reduction in entropy of S due to sorting on A

$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

- **Most informative** attribute: **max Gain(S,A)**

Information gain search heuristic

- Which attribute is more informative, A1 or A2 ?



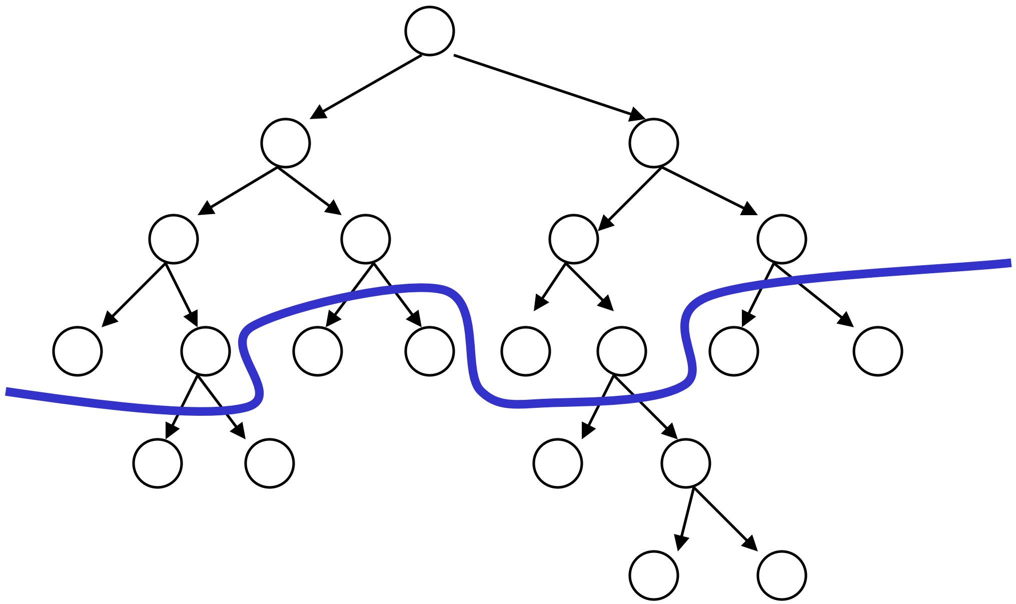
- Gain(S,A1) = 0.94 – (8/14 x 0.811 + 6/14 x 1.00) = 0.048
- Gain(S,A2) = 0.94 – 0 = 0.94 A2 has max Gain

Heuristic search in ID3

- **Search bias:** Search the space of decision trees from simplest to increasingly complex (greedy search, no backtracking, prefer small trees)
- **Search heuristics:** At a node, select the attribute that is most useful for classifying examples, split the node accordingly
- **Stopping criteria:** A node becomes a leaf
  - if all examples belong to same class C_j, label the leaf with C_j
  - if all attributes were used, label the leaf with the most common value C_k of examples in the node
- **Extension to ID3:** handling noise - tree pruning

Pruning of decision trees

- Avoid overfitting the data by tree pruning
- Pruned trees are
  - less accurate on training data
  - more accurate when classifying unseen data



Handling noise – Tree pruning

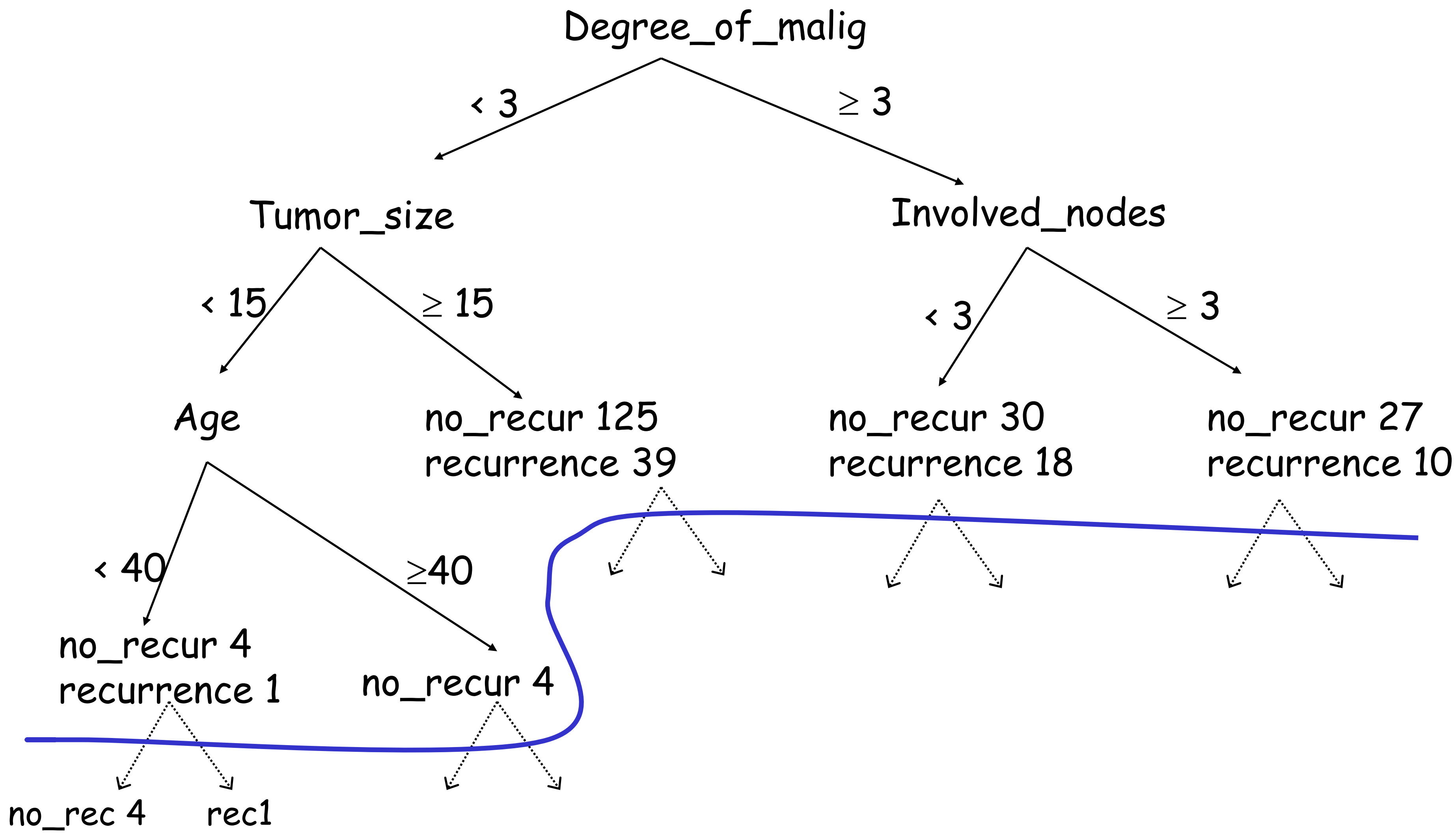
Sources of imperfection

1. Random errors (noise) in training examples
   - erroneous attribute values
   - erroneous classification
2. Too sparse training examples (incompleteness)
3. Inappropriate/insufficient set of attributes (inexactness)
4. Missing attribute values in training examples

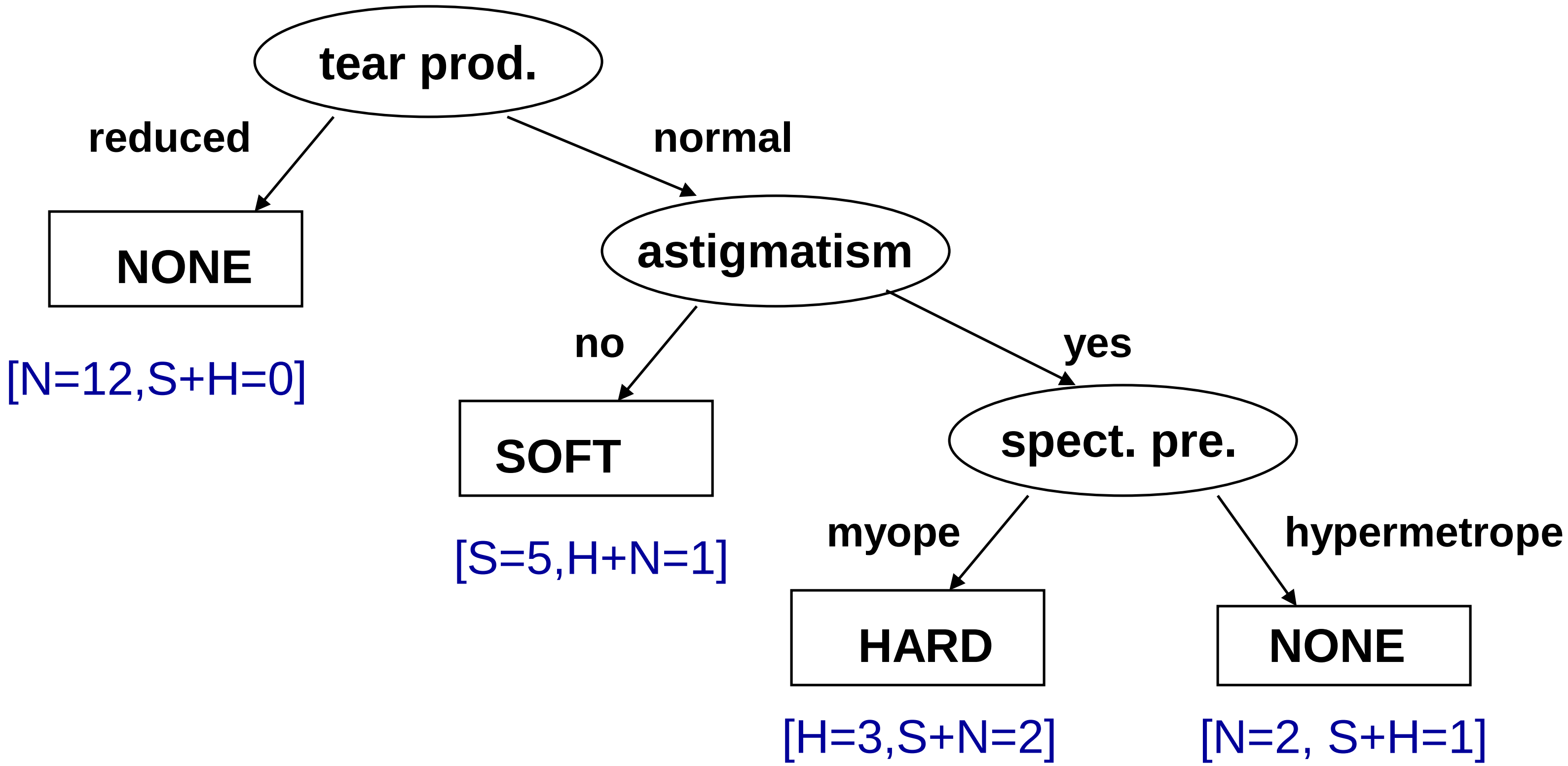
Handling noise – Tree pruning

- Handling imperfect data
  - handling imperfections of type 1-3
    - pre-pruning (stopping criteria)
    - post-pruning / rule truncation
  - handling missing values
- Pruning avoids perfectly fitting noisy data: relaxing the completeness (fitting all +) and consistency (fitting all -) criteria in ID3

Prediction of breast cancer recurrence: Tree pruning



Pruned decision tree for contact lenses recommendation

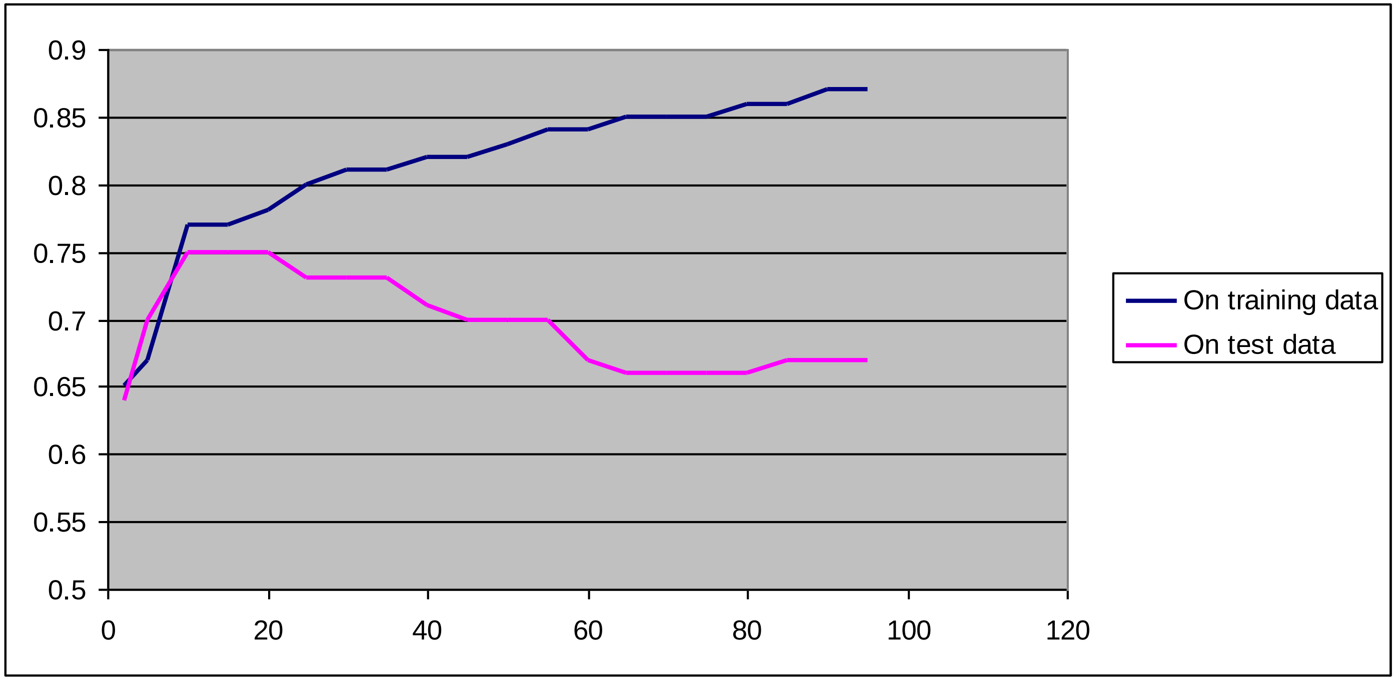


Accuracy and error

- Accuracy: percentage of correct classifications
  - on the training set
  - on unseen instances
- How accurate is a decision tree when classifying unseen instances
  - An estimate of accuracy on unseen instances can be computed, e.g., by averaging over 4 runs:
    - split the example set into training set (e.g. 70%) and test set (e.g. 30%)
    - induce a decision tree from training set, compute its accuracy on test set
- Error = 1 - Accuracy
- High error may indicate data overfitting

Overfitting and accuracy

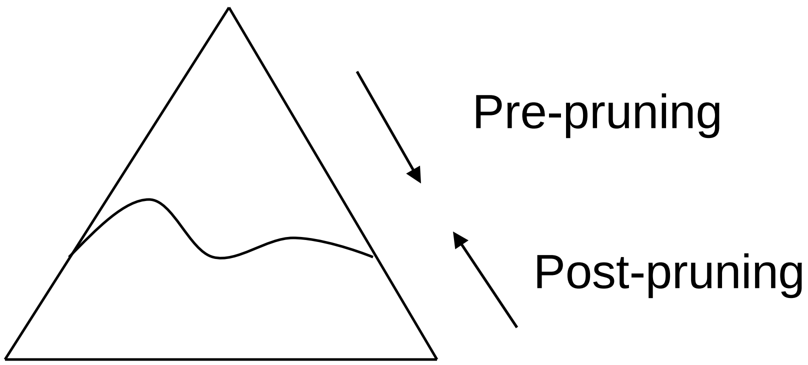
- Typical relation between tree size and accuracy



- Question: how to prune optimally?

Avoiding overfitting

- How can we avoid overfitting?
  - Pre-pruning (forward pruning): stop growing the tree e.g., when data split not statistically significant or too few examples are in a split
  - Post-pruning: grow full tree, then post-prune



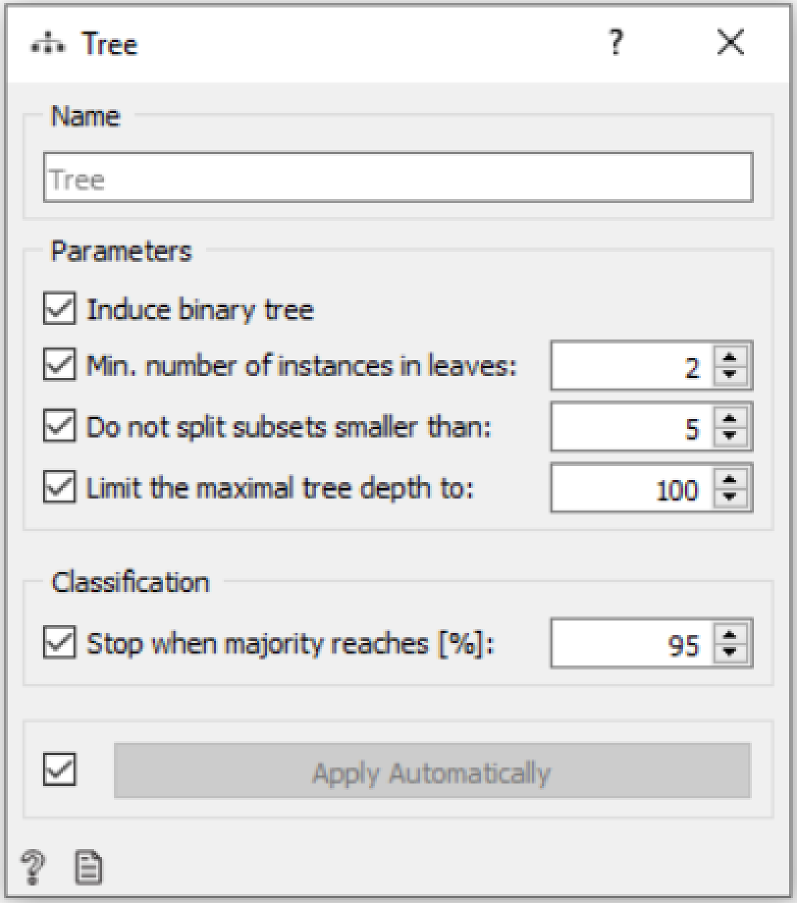
- forward pruning considered inferior (myopic)
- post pruning makes use of sub trees

Selected decision tree learners

- Decision tree learners
  - ID3 (Quinlan 1979)
  - CART (Breiman et al. 1984)
  - Assistant (Cestnik et al. 1987)
  - C4.5 (Quinlan 1993), C5 (See5, Quinlan)
  - J48 (in WEKA)
  - Tree (in Orange)

Selected decision tree learners

- Decision tree learners: Tree (in Orange)



Features of C4.5 and J48

- Implemented as part of the WEKA data mining workbench
- Handling noisy data: post-pruning
- Handling incompletely specified training instances: ‘unknown’ values (**?**)
  - in learning assign conditional probability of value v: p(v|C) = p(vC) / p(C)
  - in classification: follow all branches, weighted by prior prob. of missing attribute values

Other features of C4.5

- Binarization of attribute values
  - for continuous values select a boundary value maximally increasing the informativity of the attribute: sort the values and try every possible split (done automaticaly)
  - for discrete values try grouping the values until two groups remain *
- ‘Majority’ classification in NULL leaf (with no corresponding training example)
  - if an example ‘falls’ into a NULL leaf during classification, the class assigned to this example is the majority class of the parent of the NULL leaf

* the basic C4.5 doesn’t support binarisation of discrete attributes, it supports grouping

Appropriate problems for decision tree learning

- Classification problems: classify an instance into one of a discrete set of possible categories (medical diagnosis, classifying loan applicants, …)
- Characteristics:
  - instances described by attribute-value pairs
    (discrete or real-valued attributes)
  - target function has discrete output values
    (boolean or multi-valued, if real-valued then regression trees)
  - disjunctive hypothesis may be required
  - training data may be noisy
    (classification errors and/or errors in attribute values)
  - training data may contain missing attribute values

Classifier evaluation

- **Use of induced models**
  - discovery of new patterns, new knowledge
  - classification of new objects
- **Evaluating the quality of induced models**
  - Accuracy, Error = 1 - Accuracy
  - classification accuracy on testing examples = percentage of correctly classified instances
    - split the example set into training set (e.g. 70%) to induce a concept, and test set (e.g. 30%) to test its accuracy
    - more elaborate strategies: 10-fold cross validation, leave-one-out, ...
  - comprehensibility (compactness)
  - information contents (information score), significance

n-fold cross validation

- A method for accuracy estimation of classifiers
- Partition set D into n disjoint, almost equally-sized folds T_i where U_i T_i = D
- **for** i = 1, ..., n **do**
  - form a training set out of n-1 folds: Di = D\T_i
  - induce classifier H_i from examples in Di
  - use fold T_i for testing the accuracy of H_i
- Estimate the accuracy of the classifier by averaging accuracies over 10 folds T_i

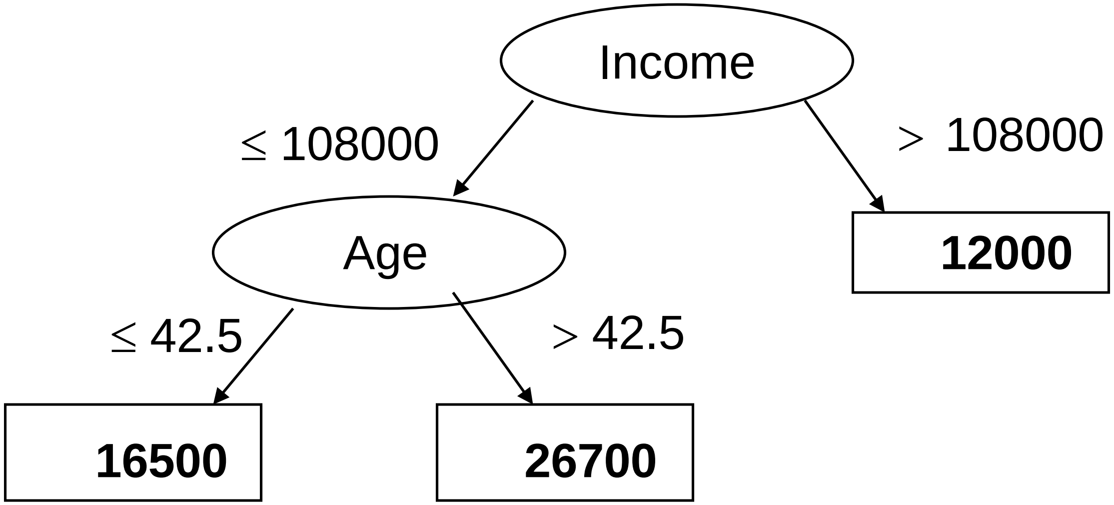
Regression tree learning

- Estimation or regression task: given objects described with attribute values, induce a model to predict the numeric class value
- Data are objects, characterized with attributes (discrete or continuous), classes of objects are continuous (numeric)
- Regression tree learners, model tree learners:
  - M5
  - M5P (implemented in WEKA)
  - Tree (in Orange)

Estimation/regression example: Customer data

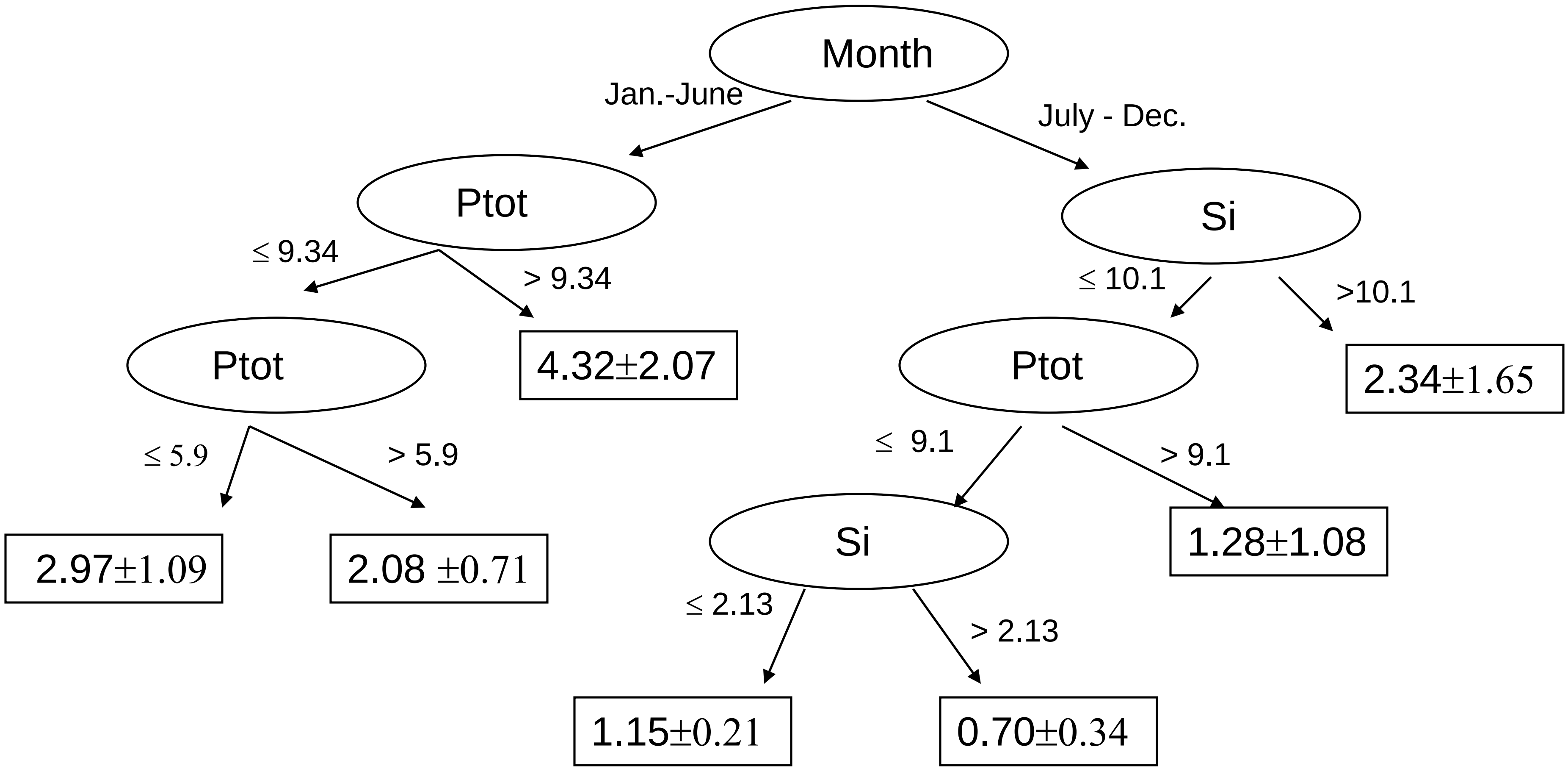
Customer	Gender	Age	Income	Spent	
c1	male	30	214000	18800	
c2	female	19	139000	15100	
c3	male	55	50000	12400	
c4	female	48	26000	8600	
c5	male	63	191000	28100	
O6-O13	...	...	...	...	
c14	female	61	95000	18100	
c15	male	56	44000	12000	
c16	male	36	102000	13800	
c17	female	57	215000	29300	
c18	male	33	67000	9700	
c19	female	26	95000	11000	
c20	female	55	214000	28800	

Customer data: regression tree



In the nodes one usually has Predicted value +- st. deviation

Predicting algal biomass: regression tree



Regression	Classification
Data: attribute-value description	
Target variable: Continuous	**Target variable**: Categorical (nominal)
Evaluation: cross validation, separate test set, …	
Error: MSE, MAE, RMSE, …	**Error**: 1-accuracy
Algorithms: Linear regression, regression trees,…	**Algorithms**: Decision trees, Naïve Bayes, …
Baseline predictor: Mean of the target variable	**Baseline predictor**: Majority class

Part II. Predictive DM techniques

- Decision tree learning
- Bayesian Classifier (more by Petra Kralj Novak)
- Rule learning
- Evaluation

Bayesian methods

- Bayesian methods – simple but powerful classification methods
  - Based on Bayesian formula

$$p(H \mid D) = \frac{p(D \mid H)}{p(D)} p(H)$$

- Main methods:
  - Naive Bayesian classifier
  - Semi-naïve Bayesian classifier
  - Bayesian networks *

* Out of scope of this course

Naïve Bayesian classifier

- Probability of class, for given attribute values

$$p(c_j \mid v_1 \ldots v_n) = p(c_j) \cdot \frac{p(v_1 \ldots v_n \mid c_j)}{p(v_1 \ldots v_n)}$$

- For all C_j compute probability $p(C_j)$, given values v_i of all attributes describing the example which we want to classify (assumption: conditional independence of attributes, when estimating $p(C_j)$ and $p(C_j \mid v_i)$)

$$p(c_j \mid v_1 \ldots v_n) \approx p(c_j) \cdot \prod_i \frac{p(c_j \mid v_i)}{p(c_j)}$$

- Output C_{MAX} with maximal posterior probability of class:

$$C_{MAX} = \arg\max_{Cj} p(c_j \mid v_1 \ldots v_n)$$

Semi-naïve Bayesian classifier

- Naive Bayesian estimation of probabilities (reliable)

$$\frac{p(c_j \mid v_i)}{p(c_j)} \cdot \frac{p(c_j \mid v_k)}{p(c_j)}$$

- Semi-naïve Bayesian estimation of probabilities (less reliable)

$$\frac{p(c_j \mid v_i, v_k)}{p(c_j)}$$

Probability estimation

- Relative frequency:

$$p(c_j) = \frac{n(c_j)}{N}, p(c_j \mid v_i) = \frac{n(c_j, v_i)}{n(v_i)}$$

j = 1. . k, for k classes

[6+,1-] (7) = 6/7

[2+,0-] (2) = 2/2 = 1

problems with small samples

- Laplace estimate (prior probability):

$$p(c_j) = \frac{n(c_j) + 1}{N + k}$$

assumes uniform prior distribution of k classes

[6+,1-] (7) = 6+1 / 7+2 = 7/9

[2+,0-] (2) = 2+1 / 2+2 = 3/4

Probability estimation

- Relative frequency:

$$p(c_j) = \frac{n(c_j)}{N}, \; p(c_j \mid v_i) = \frac{n(c_j, v_i)}{n(v_i)}$$

j = 1. . k, for k classes

- Prior probability: Laplace law

$$p(c_j) = \frac{n(c_j) + 1}{N + k}$$

- m-estimate:

$$p(c_j) = \frac{n(c_j) + m \cdot p_a(c_j)}{N + m}$$

Probability estimation: intuition

- Experiment with N trials, n successful
- Estimate probability of success of next trial
- **Relative frequency: n/N**
  - reliable estimate when number of trials is large
  - Unreliable when number of trials is small, e.g., 1/1=1
- **Laplace: (n+1)/(N+2), (n+1)/(N+k),** k classes
  - Assumes uniform distribution of classes
- **m-estimate: $(n+m.p_a)/(N+m)$**
  - Prior probability of success p_a, parameter m (weight of prior probability, i.e., number of ‘virtual’ examples)

Explanation of Bayesian classifier

- Based on information theory
  - Expected number of bits needed to encode a message = optimal code length -log p for a message, whose probability is p (*)
- Explanation based of the sum of information gains of individual attribute values v_i (Kononenko and Bratko 1991, Kononenko 1993)

$$-\log(p(c_j \mid v_1 \ldots v_n)) =$$

$$= -\log(p(c_j)) - \sum_{i=1}^{n} (-\log p(c_j) + \log(p(c_j \mid v_i))$$

* log p denotes binary logarithm

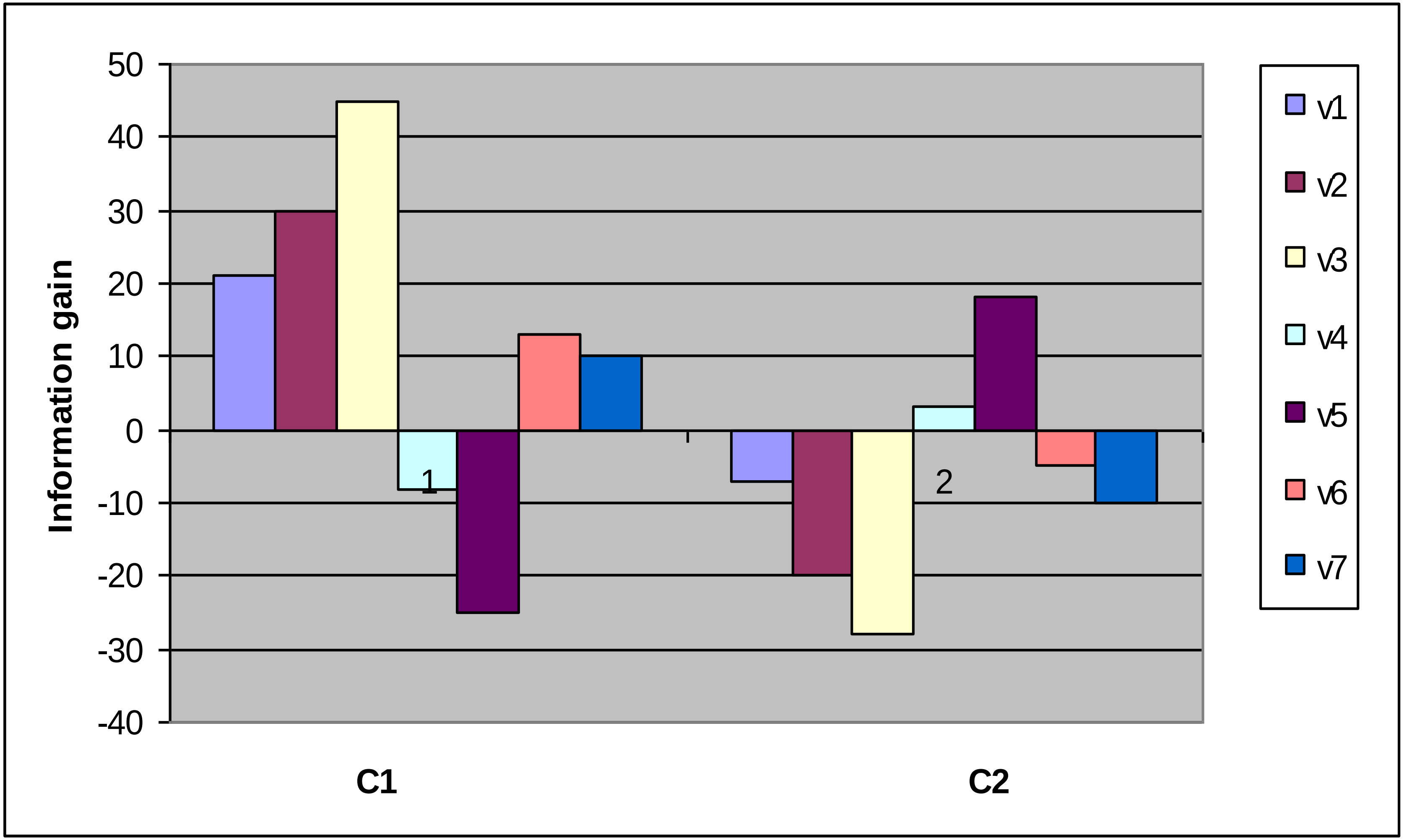
Example of explanation of semi-naïve Bayesian classifier

Hip surgery prognosis

Class = no (“no complications”, most probable class, 2 class problem)

Attribute value	For decision (bit)	Against (bit)
Age = 70-80	0.07	
Sex = Female		-0.19
Mobility before injury = Fully mobile	0.04	
State of health before injury = Other	0.52	
Mechanism of injury = Simple fall		-0.08
Additional injuries = None	0	
Time between injury and operation > 10 days	0.42	
Fracture classification acc. To Garden = Garden III		-0.3
Fracture classification acc. To Pauwels = Pauwels III		-0.14
Transfusion = Yes	0.07	
Antibiotic profilaxies = Yes		-0.32
Hospital rehabilitation = Yes	0.05	
General complications = None		0
Combination:	0.21	
Time between injury and examination < 6 hours		
AND Hospitalization time between 4 and 5 weeks		
Combination:	0.63	
Therapy = Artroplastic AND anticoagulant therapy = Yes		

Visualization of information gains for/against C_i



Naïve Bayesian classifier

- Naïve Bayesian classifier can be used
  - when we have sufficient number of training examples for reliable probability estimation
- It achieves good classification accuracy
  - can be used as ‘gold standard’ for comparison with other classifiers
- Resistant to noise (errors)
  - Reliable probability estimation
  - Uses all available information
- Successful in many application domains
  - Web page and document classification
  - Medical diagnosis and prognosis, …

Improved classification accuracy due to using m-estimate

	Primary tumor	Breast cancer	thyroid	Rheumatology
#instan	339	288	884	355
#class	22	2	4	6
#attrib	17	10	15	32
#values	2	2.7	9.1	9.1
majority	25%	80%	56%	66%
entropy	3.64	0.72	1.59	1.7

	Relative freq.	m-estimate
Primary tumor	48.20%	52.50%
Breast cancer	77.40%	79.70%
hepatitis	58.40%	90.00%
lymphography	79.70%	87.70%

Part II. Predictive DM techniques

- Decision tree learning
- Bayesian Classifier
- Rule learning
- Evaluation

Rule Learning

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

data

knowledge discovery
from data

Rule learning

Model: a set of rules

Patterns: individual rules

Given: transaction data table, relational database (a set of objects, described by attribute values)

Find: a classification model in the form of a set of rules;
or a set of interesting patterns in the form of individual rules

Rule set representation

- Rule base is a disjunctive set of conjunctive rules
- Standard form of rules:

  IF Condition THEN Class

  Class IF Conditions

  Class $\leftarrow$ Conditions

- Form of CN2 rules:

  IF Conditions THEN MajClass [ClassDistr]

- Rule base: {R1, R2, R3, …, DefaultRule}

Contact lens data: Classification rules

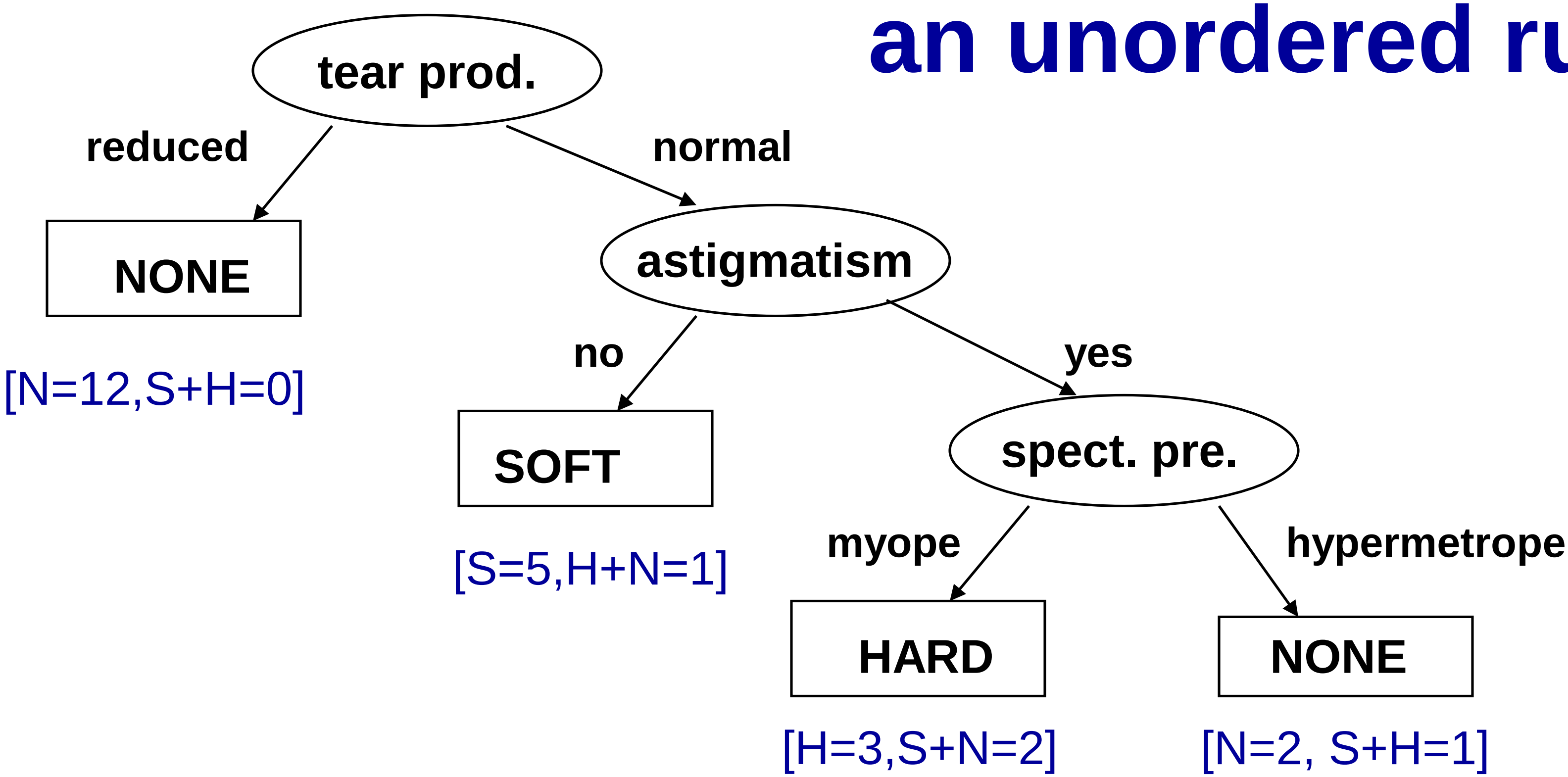
Type of task: prediction and classification
Hypothesis language: rules X ➔ C, if X then C
X conjunction of attribute values, C class

tear production=reduced → lenses=NONE
tear production=normal & astigmatism=yes & spect. pre.=hypermetrope → lenses=NONE
tear production=normal & astigmatism=no → lenses=SOFT
tear production=normal & astigmatism=yes & spect. pre.=myope → lenses=HARD
DEFAULT lenses=NONE

Rule learning

- Two rule learning approaches:
  - Learn decision tree, convert to rules
  - Learn set/list of rules
    - Learning an unordered set of rules
    - Learning an ordered list of rules
- Heuristics, overfitting, pruning

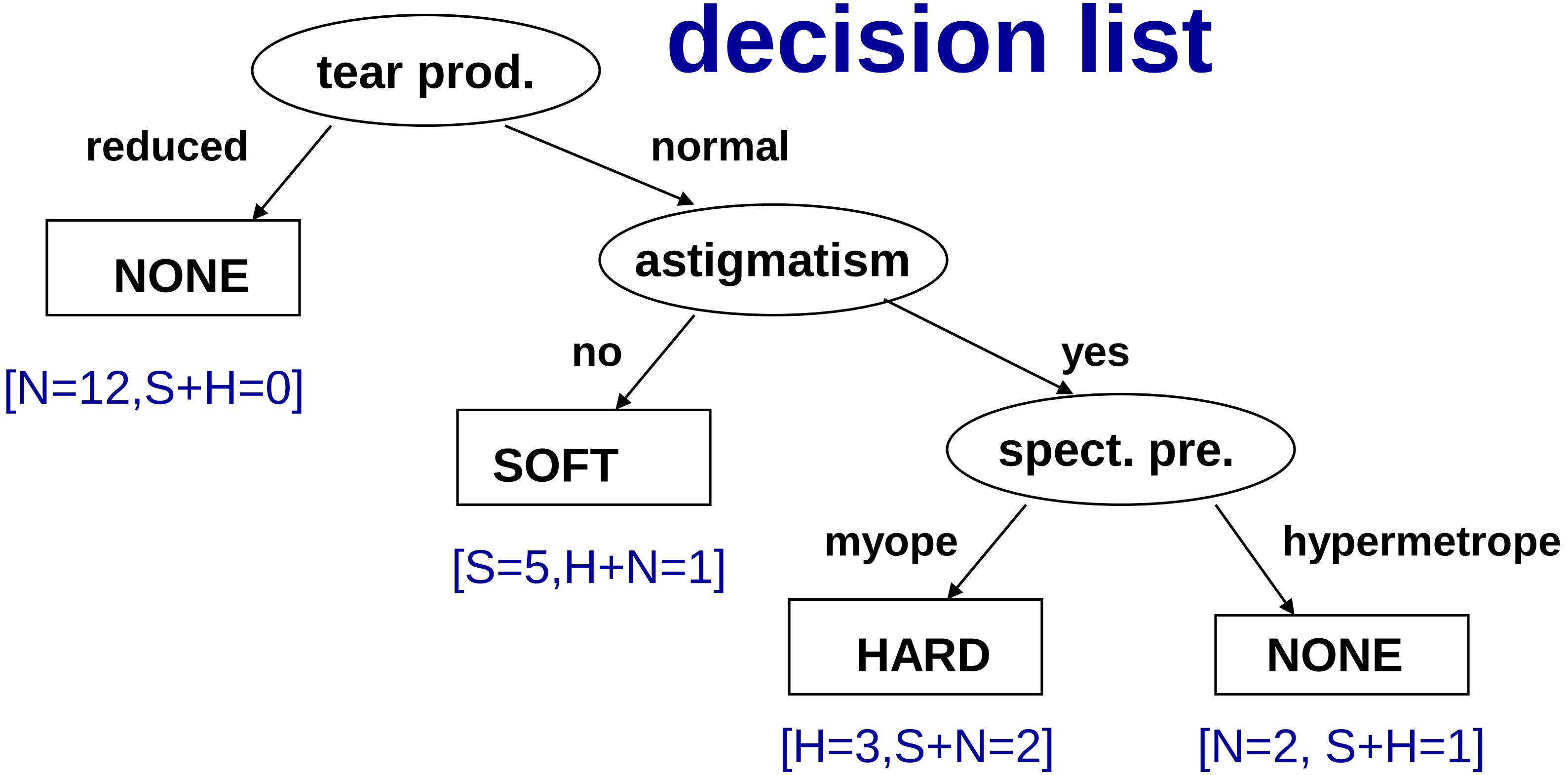
Contact lenses: convert decision tree to an unordered rule set



tear production=reduced **=>** lenses=NONE [S=0,H=0,N=12]
tear production=normal & astigmatism=yes & spect. pre.=hypermetrope **=>** lenses=NONE [S=0,H=1,N=2]
tear production=normal & astigmatism=no **=>** lenses=SOFT [S=5,H=0,N=1]
tear production=normal & astigmatism=yes & spect. pre.=myope **=>** lenses=HARD [S=0,H=3,N=2]
DEFAULT lenses=NONE

Order independent rule set (may overlap)

Contact lenses: convert decision tree to decision list



IF tear production=reduced THEN lenses=NONE
ELSE /*tear production=normal*/
IF astigmatism=no THEN lenses=SOFT
ELSE /*astigmatism=yes*/
IF spect. pre.=myope THEN lenses=HARD
ELSE /* spect.pre.=hypermetrope*/
lenses=NONE

Ordered (order dependent) rule list

Converting decision tree to rules, and rule post-pruning (Quinlan 1993)

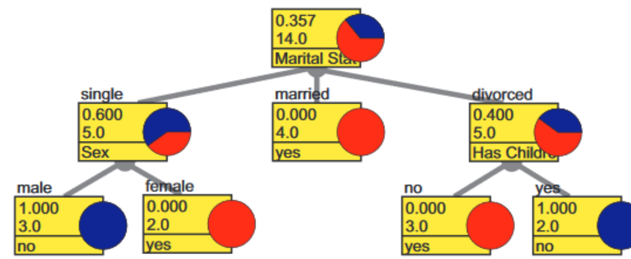
- Very frequently used method, e.g., in C4.5 and J48
- Procedure:
  - grow a full tree (allowing overfitting)
  - convert the tree to an equivalent set of rules
  - prune each rule independently of others
  - sort final rules into a desired sequence for use

Concept learning: Task reformulation for rule learning: (pos. vs. neg. examples of Target class)

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	17	myope	no	reduced	NO
O2	23	myope	no	normal	YES
O3	22	myope	yes	reduced	NO
O4	27	myope	yes	normal	YES
O5	19	hypermetrope	no	reduced	NO
O6-O13	...	...	...	...	...
O14	35	hypermetrope	no	normal	YES
O15	43	hypermetrope	yes	reduced	NO
O16	39	hypermetrope	yes	normal	NO
O17	54	myope	no	reduced	NO
O18	62	myope	no	normal	NO
O19-O23	...	...	...	...	...
O24	56	hypermetrope	yes	normal	NO

Learning predictive rules

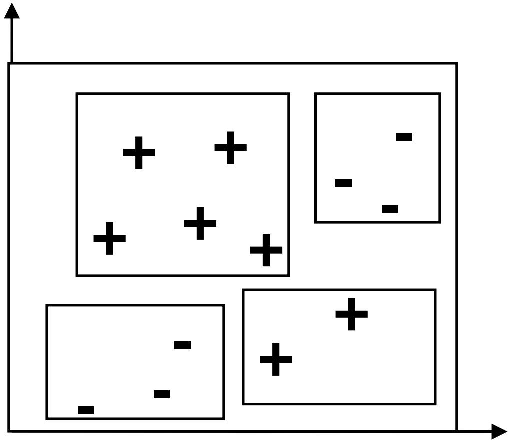
Education	Marital Status	Sex	Has Children	Approved
primary	single	male	no	no
primary	single	male	yes	no
primary	married	male	no	yes
university	divorced	female	no	yes
university	married	female	yes	yes
secondary	single	male	no	no
university	single	female	no	yes
secondary	divorced	female	no	yes
secondary	single	female	yes	yes
secondary	married	male	yes	yes
primary	married	female	no	yes
secondary	divorced	male	yes	no
university	divorced	female	yes	no
secondary	divorced	male	no	yes




```
Sex = female  →  Approved = yes
MaritalStatus = single  AND  Sex = male  →  Approved = no
MaritalStatus = married  →  Approved = yes
MaritalStatus = divorced  AND  HasChildren = yes  →  Approved = no
MaritalStatus = divorced  AND  HasChildren = no  →  Approved = yes
```

Figure 2: A set of predictive rules, modeling the data set shown in Table 1.

Original covering algorithm (AQ, Michalski 1969,86)



Given examples of N classes $C_1, \ldots, C_N$

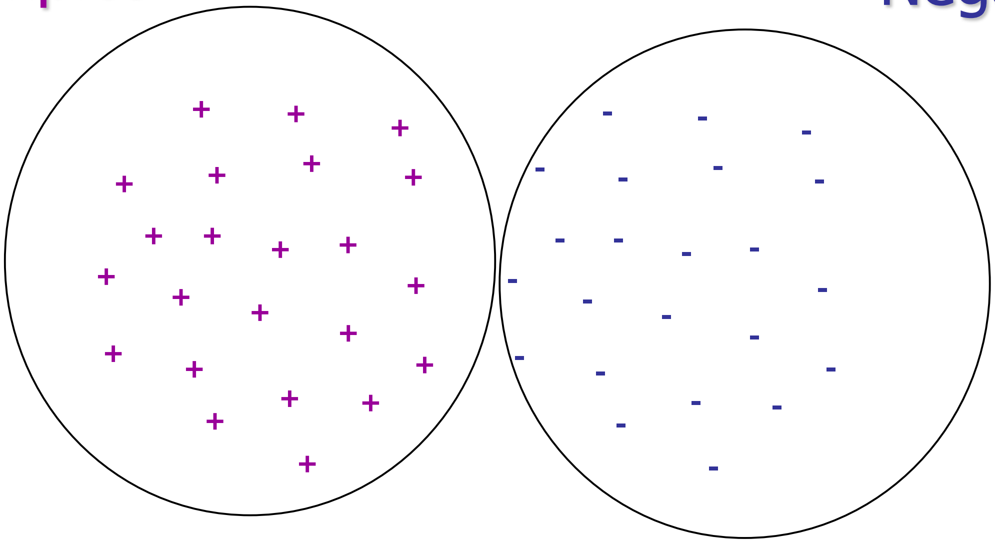
for each class Ci **do**

- Ei := Pi U Ni (Pi pos., Ni neg.)
- RuleBase(Ci) := empty
- **repeat {learn-set-of-rules}**
  - **learn-one-rule** R covering some positive examples and no negatives
  - add R to RuleBase(Ci)
  - delete from Pi all pos. ex. covered by R
- **until** Pi = empty

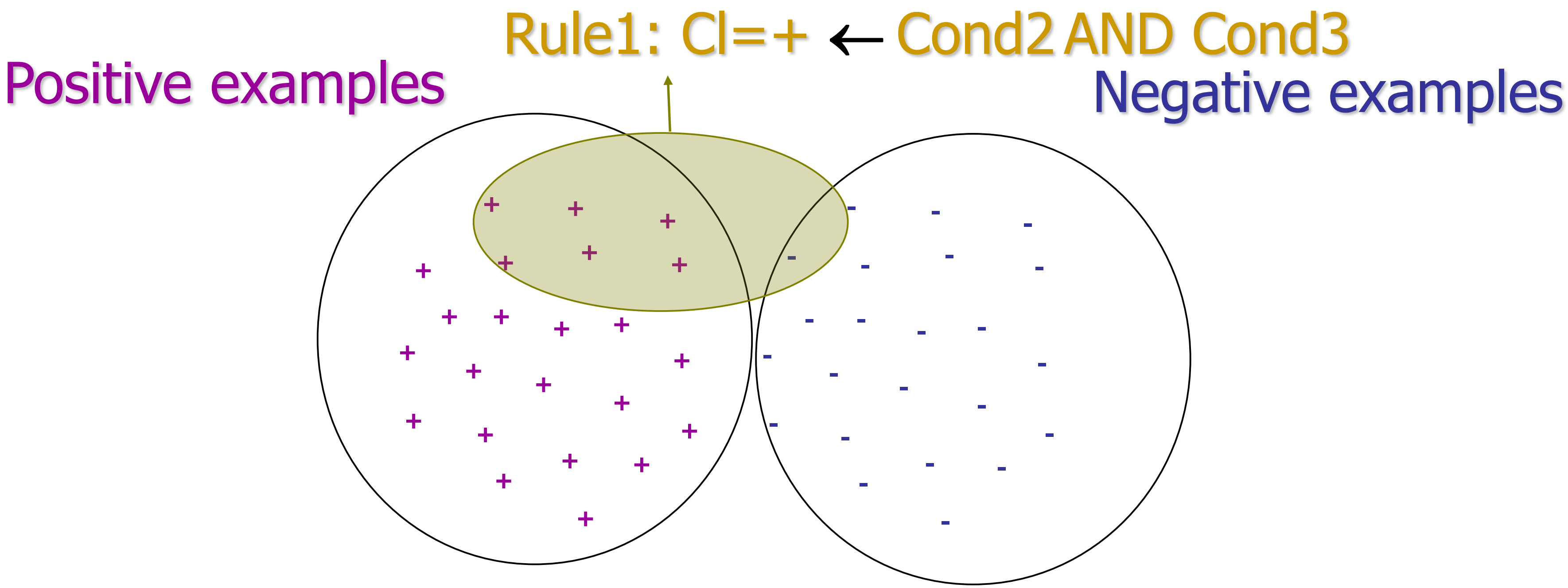
Covering algorithm

Positive examples

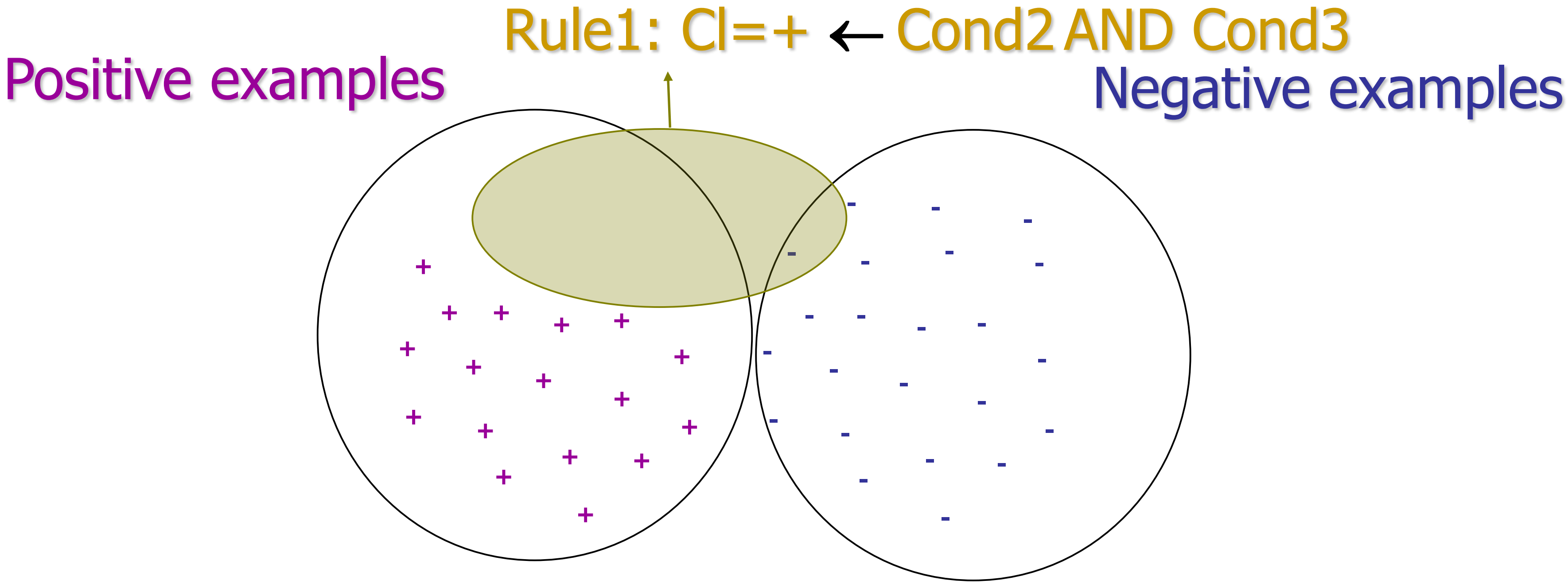
Negative examples



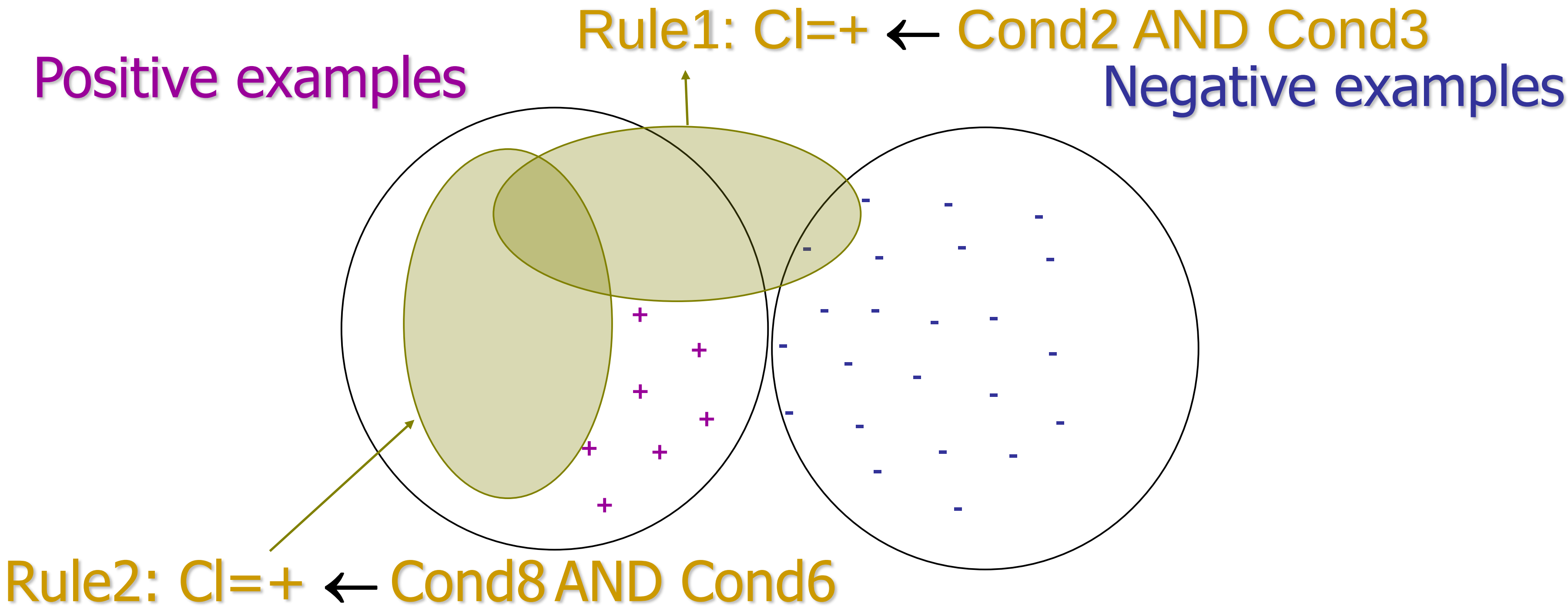
Covering algorithm



Covering algorithm



Covering algorithm



Probability estimates

- **Relative frequency** :
  - problems with small samples

$$p(Class \mid Cond) =$$
$$= \frac{n(Class.Cond)}{n(Cond)}$$

[6+,1-] (7) = 6/7

[2+,0-] (2) = 2/2 = 1

- **Laplace estimate** :
  - assumes uniform prior distribution of k classes

$$= \frac{n(Class.Cond)+1}{n(Cond)+k} \qquad k = 2$$

[6+,1-] (7) = 6+1 / 7+2 = 7/9

[2+,0-] (2) = 2+1 / 2+2 = 3/4

Learn-one-rule: search heuristics

- Assume a two-class problem
- Two classes (+,-), learn rules for + class (Cl).
- Search for specializations R' of a rule R = Cl $\leftarrow$ Cond from the RuleBase.
- Specializarion R' of rule R = Cl $\leftarrow$ Cond

  has the form R' = Cl $\leftarrow$ Cond & Cond'
- Heuristic search for rules: find the 'best' Cond' to be added to the current rule R, such that rule accuracy is improved, e.g., such that Acc(R') > Acc(R)
  - where the expected **classification accuracy** can be estimated as A(R) = p(Cl|Cond)

Learn-one-rule: Greedy vs. beam search

- learn-one-rule by greedy general-to-specific search, at each step selecting the `best' descendant, no backtracking
  - e.g., the best descendant of the initial rule

    lenses=NONE ←
  - is rule lenses=NONE ← tear production=reduced

- beam search: maintain a list of k best candidates at each step; descendants (specializations) of each of these k candidates are generated, and the resulting set is again reduced to k best candidates

What is "high" rule accuracy (rule precision) ?

- Rule evaluation measures:
  - aimed at maximizing classification accuracy
  - minimizing Error = 1 - Accuracy
  - avoiding overfitting
- BUT: Rule accuracy/precision should be traded off against the "default" accuracy/precision of the rule **Cl ←true**
  - 68% accuracy is OK if there are 20% examples of that class in the training set, but bad if there are 80%
- **Relative accuracy** *(relative precision)*
  - RAcc(Cl ←Cond) = p(Cl | Cond) – p(Cl)

Learn-one-rule: search heuristics

- Assume two classes (+,-), learn rules for + class (Cl). Search for specializations of one rule R = Cl ← Cond from RuleBase.
- Expected **classification accuracy**: A(R) = p(Cl|Cond)
- **Informativity** (info needed to specify that example covered by Cond belongs to Cl): I(R) = $-\log_2 p(Cl|Cond)$
- **Accuracy gain** (increase in expected accuracy):

  AG(R',R) = p(Cl|Cond') - p(Cl|Cond)
- **Information gain** (decrease in the information needed):

  IG(R',R) = $\log_2 p(Cl|Cond') - \log_2 p(Cl|Cond)$
- **Weighted** measures favoring more general rules: WAG, WIG

  WAG(R',R) =

  p(Cond')/p(Cond) . (p(Cl|Cond') - p(Cl|Cond))
- **Weighted relative accuracy** trades off coverage and relative accuracy WRAcc(R) = p(Cond).(p(Cl|Cond) - p(Cl))

Ordered set of rules: if-then-else rules

- rule Class IF Conditions is learned by first determining Conditions and then Class
- **Notice:** mixed sequence of classes C1, …, Cn in RuleBase
- **But: ordered** execution when classifying a new instance: rules are sequentially tried and the first rule that `fires' (covers the example) is used for classification
- **Decision list {R1, R2, R3, …, D}:** rules Ri are interpreted as **if-then-else** rules
- If no rule fires, then DefaultClass (majority class in E_{cur})

Sequential covering algorithm

- RuleBase := empty
- E_{cur}:= E
- **repeat**
  - learn-one-rule R
  - RuleBase := RuleBase U R
  - E_{cur} := E_{cur} - {examples covered and correctly classified by R} **(DELETE ONLY POS. EX.!)**
  - **until** performance(R, E_{cur}) < ThresholdR
- RuleBase := sort RuleBase by performance(R,E)
- return RuleBase

Learn ordered set of rules (CN2, Clark and Niblett 1989)

- RuleBase := empty
- E_{cur}:= E
- **repeat**
  - learn-one-rule R
  - RuleBase := RuleBase U R
  - E_{cur} := E_{cur} - {all examples covered by R}
    (NOT ONLY POS. EX.!)
- **until** performance(R, E_{cur}) < ThresholdR
- RuleBase := sort RuleBase by performance(R,E)
- RuleBase := RuleBase U DefaultRule(E_{cur})

Learn-one-rule:
Beam search in CN2

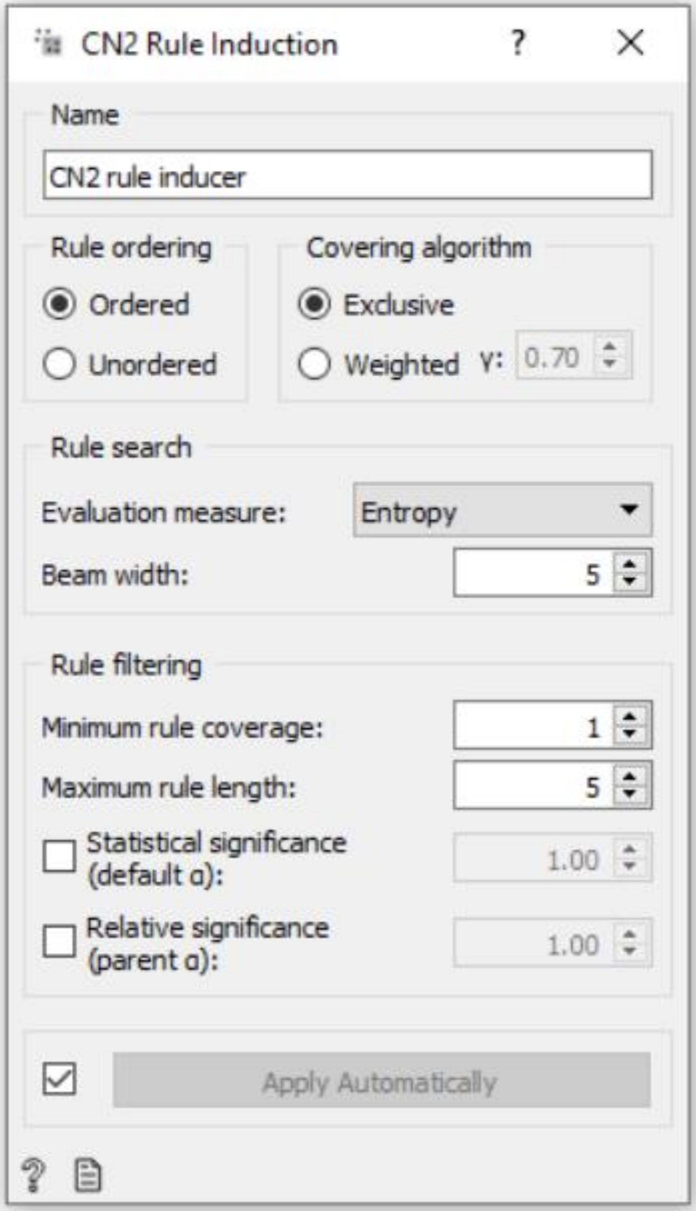
- Beam search in CN2 learn-one-rule algo.:
  - construct BeamSize of best rule bodies (conjunctive conditions) that are statistically significant
  - BestBody - min. entropy of examples covered by Body
  - construct best rule R := Head $\leftarrow$ BestBody by adding majority class of examples covered by BestBody in rule Head
- performance (R, E_{cur}) : - Entropy(E_{cur})
  - performance(R, E_{cur}) < ThresholdR (neg. num.)
  - Why? Ent. > t is bad, Perf. = -Ent < -t is bad

Variations

- Sequential vs. simultaneous covering of data (as in TDIDT): choosing between attribute-values vs. choosing attributes
- Learning rules vs. learning decision trees and converting them to rules
- Pre-pruning vs. post-pruning of rules
- What statistical evaluation functions to use
- Probabilistic classification

- Best performing rule learning algorithm: Ripper
- JRip implementation of Ripper in WEKA, available in ClowdFlows

CN2 rule learner in Orange



Probabilistic classification

- In the ordered case of standard CN2 rules are interpreted in an `IF-THEN-ELSE` fashion, and the first fired rule assigns the class.
- In the unordered case all rules are tried and all rules which fire are collected. If a clash occurs, a probabilistic method is used to resolve the clash.
- A simplified example:
  1. tear production=reduced **=>** lenses=NONE [S=0,H=0,N=12]
  2. tear production=normal & astigmatism=yes & spect. pre.=hypermetrope **=>** lenses=NONE [S=0,H=1,N=2]
  3. tear production=normal & astigmatism=no **=>** lenses=SOFT [S=5,H=0,N=1]
  4. tear production=normal & astigmatism=yes & spect. pre.=myope **=>** lenses=HARD [S=0,H=3,N=2]
  5. DEFAULT lenses=NONE

Suppose we want to classify a person with normal tear production and astigmatism. Two rules fire: rule 2 with coverage [S=0,H=1,N=2] and rule 4 with coverage [S=0,H=3,N=2]. The classifier computes total coverage as [S=0,H=4,N=4], resulting in probabilistic classification into class H with probability 0.5 and N with probability 0.5. In this case, the clash can not be resolved, as both probabilities are equal.

Part II. Predictive DM techniques

- Decision tree learning
- Bayesian Classifier
- Rule learning
- Evaluation (more by Petra Kralj Novak)

Course Outline

I. Introduction
- Data Mining and KDD process
- Introduction to Data Mining
- Data Mining platforms

II. Predictive DM Techniques
- Decision Tree learning
- Bayesian classifier
- Classification rule learning
- Classifier Evaluation

III. Regression

IV. Descriptive DM
- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
  Hierarchical clustering

V. Relational Data Mining
- RDM and Inductive Logic Programming
- Propositionalization
- Semantic data mining

VI. Advanced Topics

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Descriptive DM: Subgroup discovery example - Customer data

Customer	Gender	Age	Income	Spent	BigSpender
c1	male	30	214000	18800	yes
c2	female	19	139000	15100	yes
c3	male	55	50000	12400	no
c4	female	48	26000	8600	no
c5	male	63	191000	28100	yes
O6-O13	...	...	...	...	...
c14	female	61	95000	18100	yes
c15	male	56	44000	12000	no
c16	male	36	102000	13800	no
c17	female	57	215000	29300	yes
c18	male	33	67000	9700	no
c19	female	26	95000	11000	no
c20	female	55	214000	28800	yes

Customer data: Subgroup discovery

Type of task: description (pattern discovery)

Hypothesis language: rules **X ➔ Y,** if X then Y

X is conjunctions of items, Y is target class

Age > 52 & Sex = male ➔ BigSpender = no

Age > 52 & Sex = male & Income ≤ 73250
➔ BigSpender = no

Descriptive DM: Association rule learning example - Customer data

Customer	Gender	Age	Income	Spent	BigSpender
c1	male	30	214000	18800	yes
c2	female	19	139000	15100	yes
c3	male	55	50000	12400	no
c4	female	48	26000	8600	no
c5	male	63	191000	28100	yes
O6-O13	...	...	...	...	...
c14	female	61	95000	18100	yes
c15	male	56	44000	12000	no
c16	male	36	102000	13800	no
c17	female	57	215000	29300	yes
c18	male	33	67000	9700	no
c19	female	26	95000	11000	no
c20	female	55	214000	28800	yes

Customer data: Association rules

Type of task: description (pattern discovery)

Hypothesis language: rules **X → Y,** if X then Y

X, Y conjunctions of items

1. Age > 52 & BigSpender = no → Sex = male
2. Age > 52 & BigSpender = no →
   Sex = male & Income ≤ 73250
3. Sex = male & Age > 52 & Income ≤ 73250 →
   BigSpender = no

Descriptive DM: Clustering and association rule learning example - Customer data

Customer	Gender	Age	Income	Spent	BigSpender
c1	male	30	214000	18800	yes
c2	female	19	139000	15100	yes
c3	male	55	50000	12400	no
c4	female	48	26000	8600	no
c5	male	63	191000	28100	yes
O6-O13	...	...	...	...	...
c14	female	61	95000	18100	yes
c15	male	56	44000	12000	no
c16	male	36	102000	13800	no
c17	female	57	215000	29300	yes
c18	male	33	67000	9700	no
c19	female	26	95000	11000	no
c20	female	55	214000	28800	yes

Predictive vs. descriptive induction

- **Predictive induction:** Inducing classifiers for solving classification and prediction tasks,
  - Classification rule learning, Decision tree learning, ...
  - Bayesian classifier, ANN, SVM, ...
  - Data analysis through hypothesis generation and testing
- **Descriptive induction:** Discovering interesting regularities in the data, uncovering patterns, ... for solving KDD tasks
  - Symbolic clustering, Association rule learning, Subgroup discovery, ...
  - Exploratory data analysis

Descriptive DM

- Often used for preliminary explanatory data analysis
- User gets feel for the data and its structure
- Aims at deriving descriptions of characteristics of the data
- Visualization and descriptive statistical techniques can be used

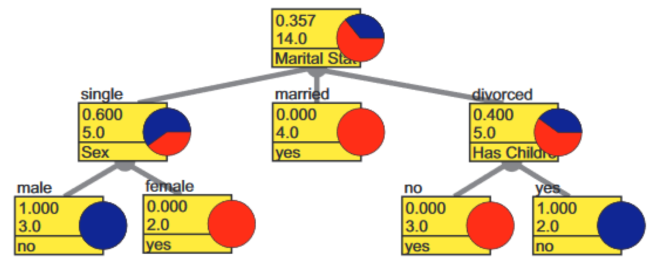
Predictive vs. descriptive DM: Summary from a rule learning perspective

- **Predictive DM:** Induces **rulesets** acting as classifiers for solving classification and prediction tasks
- **Descriptive DM:** Discovers **individual rules** describing interesting regularities in the data

- **Therefore:** Different goals, different heuristics, different evaluation criteria

Learning descriptive rules

Education	Marital Status	Sex	Has Children	Approved
primary	single	male	no	no
primary	single	male	yes	no
primary	married	male	no	yes
university	divorced	female	no	yes
university	married	female	yes	yes
secondary	single	male	no	no
university	single	female	no	yes
secondary	divorced	female	no	yes
secondary	single	female	yes	yes
secondary	married	male	yes	yes
primary	married	female	no	yes
secondary	divorced	male	yes	no
university	divorced	female	yes	no
secondary	divorced	male	no	yes



MaritalStatus = single AND Sex = male $\longrightarrow$ Approved = no

Sex = male $\longrightarrow$ Approved = no

Sex = female $\longrightarrow$ Approved = yes

MaritalStatus = married $\longrightarrow$ Approved = yes

MaritalStatus = divorced AND HasChildren = yes $\longrightarrow$ Approved = no

MaritalStatus = single $\longrightarrow$ Approved = no

Figure 3: Selected descriptive rules, describing individual patterns in the data of Table 1.

Descriptive DM

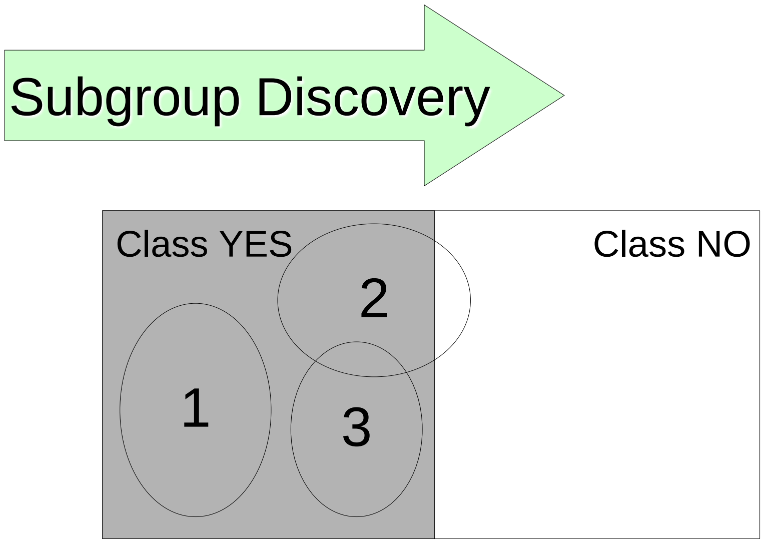
- **Description**
  - Data description and summarization: describe elementary and aggregated data characteristics (statistics, …)
  - Dependency analysis:
    - describe associations, dependencies, …
    - discovery of properties and constraints
- **Segmentation**
  - Clustering: separate objects into subsets according to distance and/or similarity (clustering, SOM, visualization, ...)
  - Subgroup discovery: find unusual subgroups that are significantly different from the majority (deviation detection w.r.t. overall class distribution)

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Subgroup Discovery

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	17	myope	no	reduced	NO
O2	23	myope	no	normal	YES
O3	22	myope	yes	reduced	NO
O4	27	myope	yes	normal	YES
O5	19	hypermetrope	no	reduced	NO
O6-O13	...	...	...	...	...
O14	35	hypermetrope	no	normal	YES
O15	43	hypermetrope	yes	reduced	NO
O16	39	hypermetrope	yes	normal	NO
O17	54	myope	no	reduced	NO
O18	62	myope	no	normal	NO
O19-O23	...	...	...	...	...
O24	56	hypermetrope	yes	normal	NO

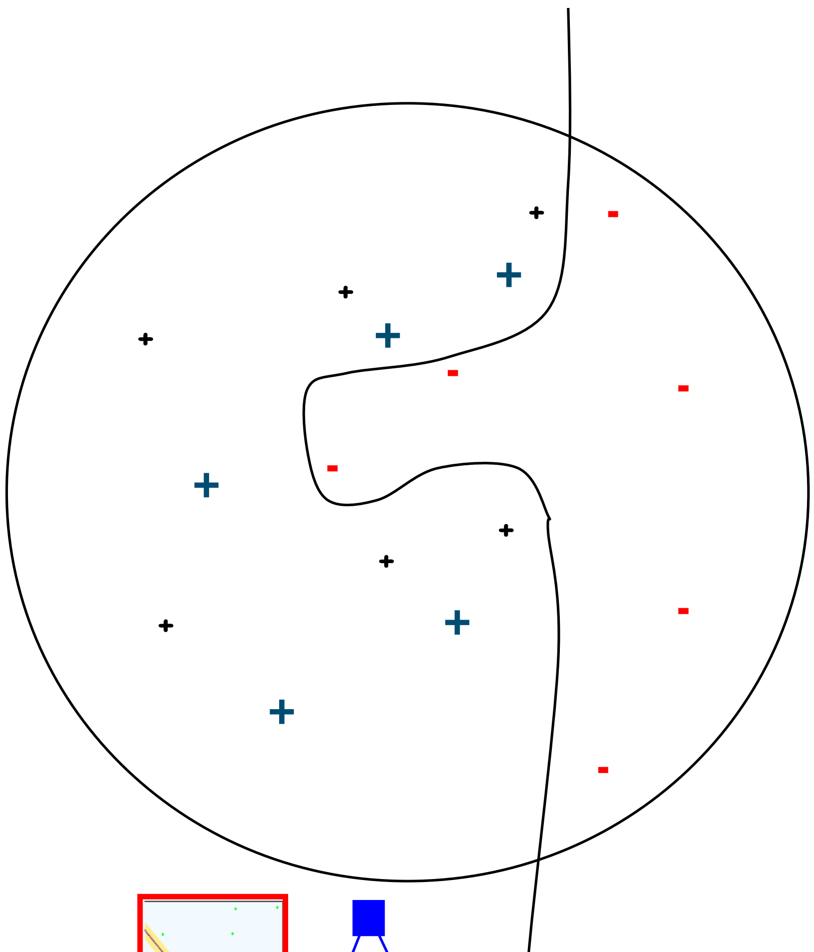


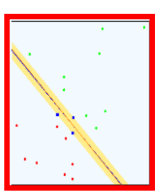
- A task in which individual interpretable patterns in the form of rules are induced from data, labeled by a predefined property of interest.
- SD algorithms learn several independent rules that describe groups of target class examples
  - subgroups must be large and significant

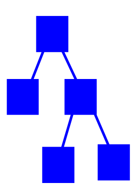
Classification versus Subgroup Discovery

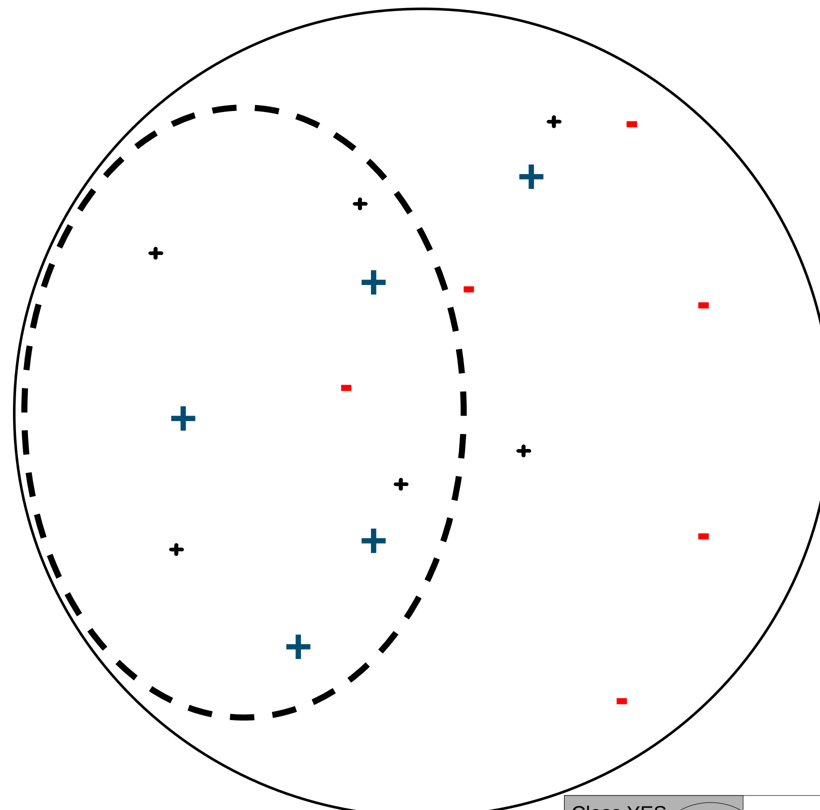
- **Classification (predictive induction) - constructing sets of classification rules**
  - aimed at learning a model for classification or prediction
  - rules are dependent
- **Subgroup discovery (descriptive induction) – constructing individual subgroup describing rules**
  - aimed at finding interesting patterns in target class examples
    - large subgroups (high target class coverage)
    - with significantly different distribution of target class examples (high TP/FP ratio, high significance, high WRAcc
  - each rule (pattern) is an independent chunk of knowledge

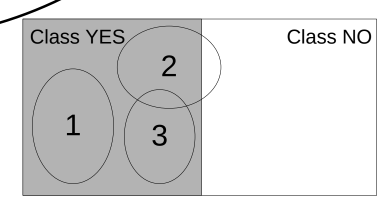
Classification versus Subgroup discovery











Subgroup discovery in High CHD Risk Group Detection

Input: Patient records described by anamnestic, laboratory and ECG attributes

Task: Find and characterize population subgroups with high CHD risk (large enough, distributionaly unusual)

From **best induced descriptions**, five were selected by the expert as **most actionable** for CHD risk screening (by GPs):

high-CHD-risk ← male & pos. fam. history & age > 46

high-CHD-risk ← female & bodymassIndex > 25 & age > 63

high-CHD-risk ← ...

high-CHD-risk ← ...

high-CHD-risk ← ...

(Gamberger & Lavrač, JAIR 2002)

Subgroup Discovery: Medical Use Case

- **Find and characterize population subgroups with high risk for coronary heart disease (CHD)** (Gamberger, Lavrač, Krstačić)
- **A1** for males: **principal risk factors**

  CHD ← pos. fam. history & age > 46
- **A2** for females: **principal risk factors**

  CHD ← bodyMassIndex > 25 & age >63
- **A1, A2** (anamnestic info only), **B1, B2** (an. and physical examination), **C1** (an., phy. and ECG)
- **A1: supporting factors** (found by statistical analysis): psychosocial stress, as well as cigarette smoking, hypertension and overweight

Subgroup discovery in functional genomics

- Functional genomics is a typical scientific discovery domain, studying genes and their functions
- Very large number of attributes (genes)
- Interesting subgroup describing patterns discovered by SD algorithm

CancerType = Leukemia	
IF	KIAA0128 = DIFF. EXPRESSED
AND	prostoglandin d2 synthase = NOT_ DIFF. EXPRESSED

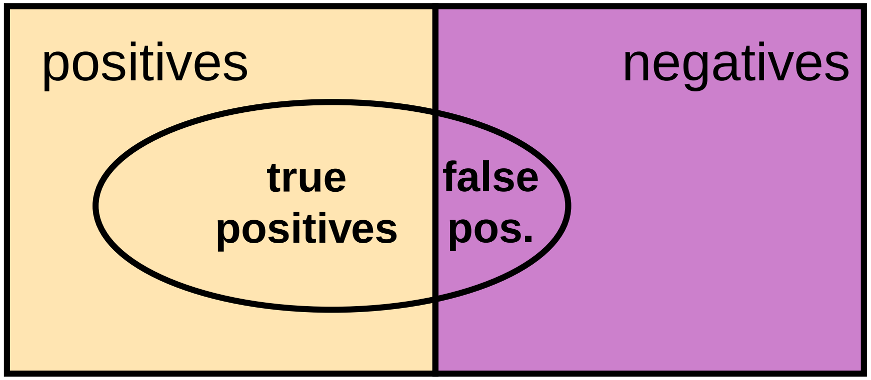
- Interpretable by biologists

D. Gamberger, N. Lavrač, F. Železný, J. Tolar

Journal of Biomedical Informatics 37(5):269-284, 2004

Subgroups vs. classifiers

- Classifiers:
  - Classification rules aim at pure subgroups
  - A set of rules forms a domain model
- Subgroups:
  - Rules describing subgroups aim at significantly higher proportion of positives
  - Each rule is an independent chunk of knowledge
- Link
  - SD can be viewed as cost-sensitive classification
  - Instead of *FNcost* we aim at increased *TPprofit*



Classification Rule Learning for Subgroup Discovery: Deficiencies

- Only first few rules induced by the covering algorithm have sufficient support (coverage)
- Subsequent rules are induced from smaller and strongly biased example subsets (pos. examples not covered by previously induced rules), which hinders their ability to detect population subgroups
- ‘Ordered’ rules are induced and interpreted sequentially as a **if-then-else** decision list

CN2-SD: Adapting CN2 Rule Learning to Subgroup Discovery

- Weighted covering algorithm
- Weighted relative accuracy (WRAcc) search heuristics, with added example weights
- Probabilistic classification
- Evaluation with different interestingness measures

CN2-SD: CN2 Adaptations

- General-to-specific search (beam search) for best rules
- Rule quality measure:
  - CN2: Laplace: Acc(Class $\leftarrow$ Cond) =

    = p(Class|Cond) = $(n_c+1)/(n_{rule}+k)$
  - CN2-SD: Weighted Relative Accuracy

    WRAcc(Class $\leftarrow$ Cond) =

    p(Cond) (p(Class|Cond) - p(Class))
- Weighted covering approach (example weights)
- Significance testing (likelihood ratio statistics)
- Output: Unordered rule sets (probabilistic classification)

CN2-SD: Weighted Covering

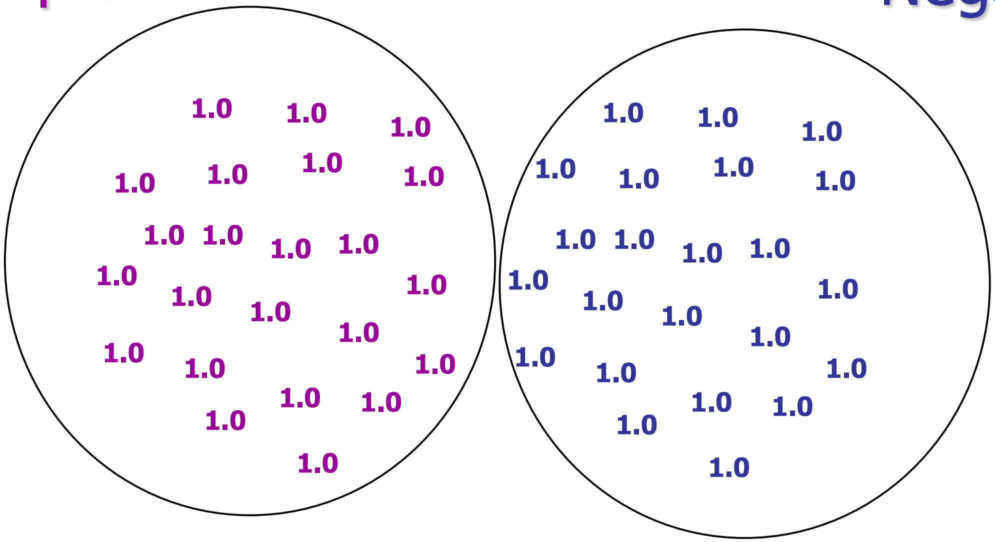
- Standard covering approach:
  covered examples are deleted from current training set

- Weighted covering approach:
  - weights assigned to examples
  - covered pos. examples are re-weighted:
    in all covering loop iterations, store
    count i how many times (with how many
    rules induced so far) a pos. example has
    been covered: w(e,i), w(e,0)=1
    - **Additive weights:** `w(e,i) = 1/(i+1)`
      `w(e,i)` – pos. example e being covered `i` times

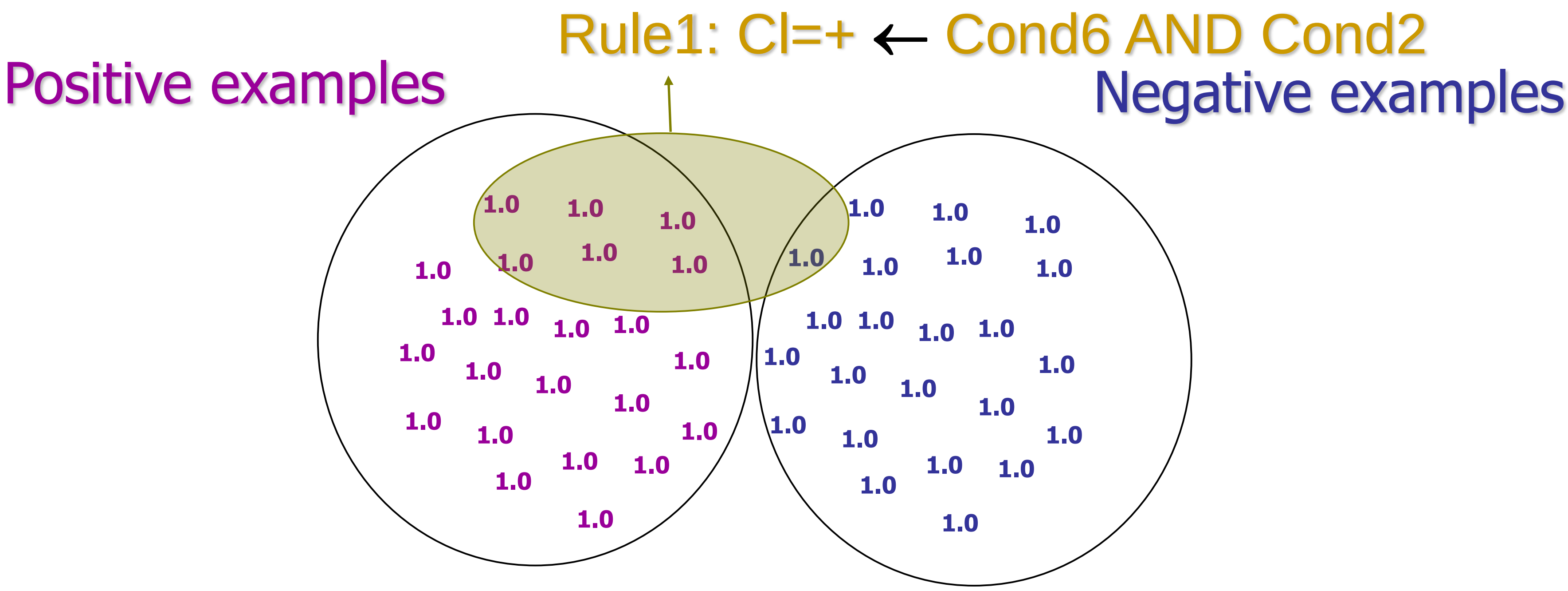
Subgroup Discovery

Positive examples

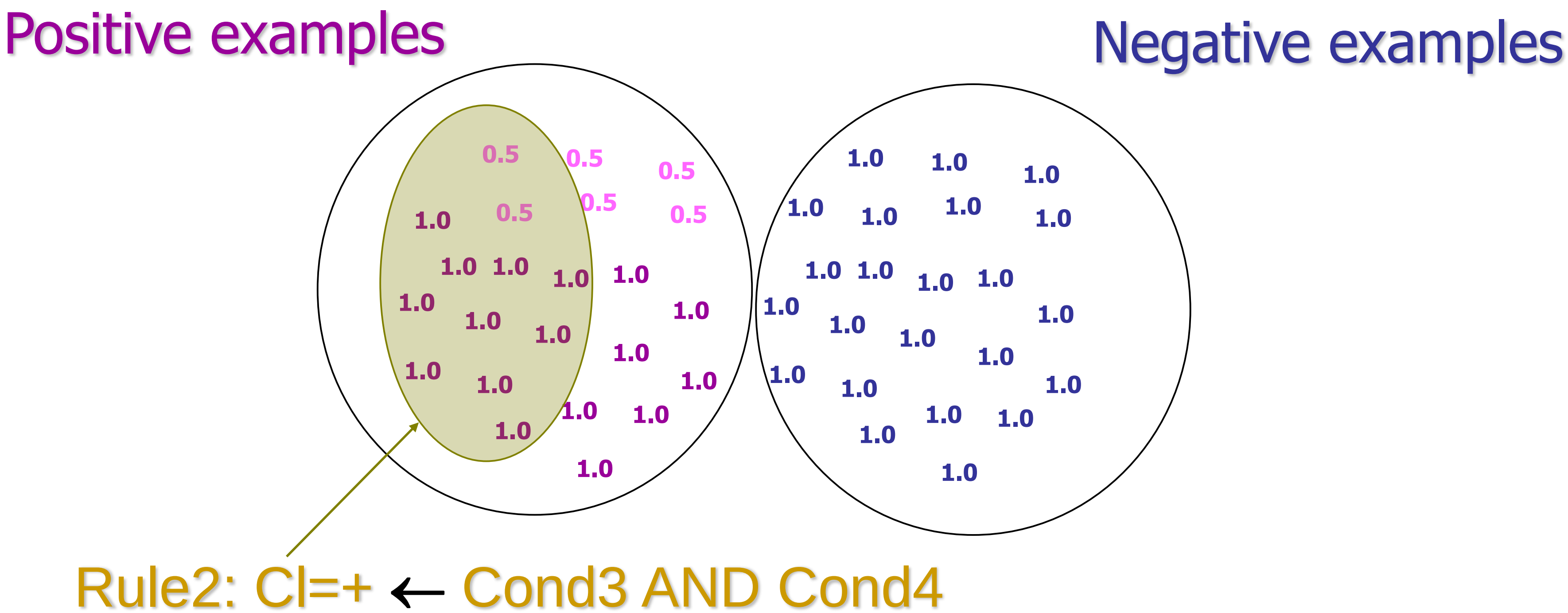
Negative examples



Subgroup Discovery



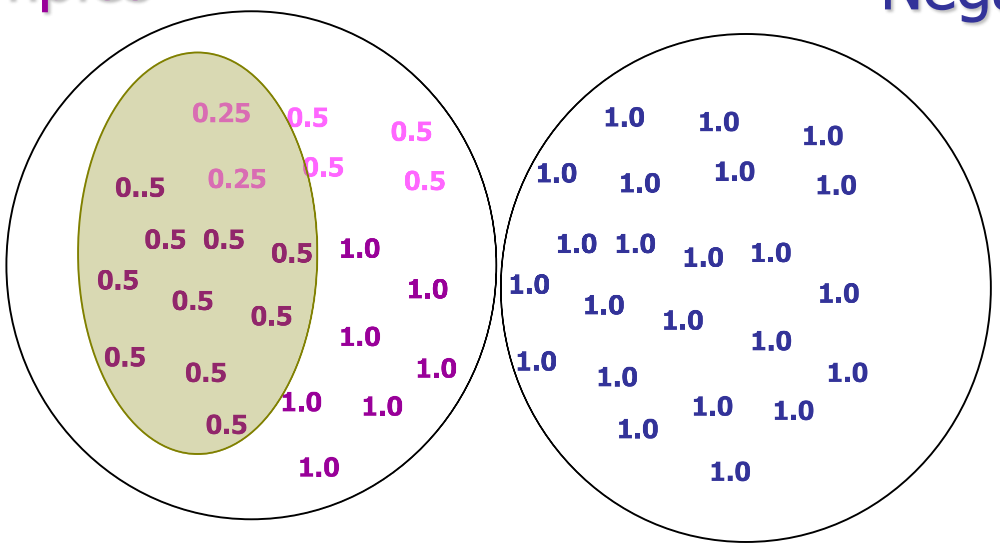
Subgroup Discovery



Subgroup Discovery

Positive examples

Negative examples



CN2-SD: Weighted WRAcc Search Heuristic

- **Weighted relative accuracy (WRAcc) search heuristics, with added example weights**

  WRAcc(Cl ← Cond) = p(Cond) (p(Cl|Cond) - p(Cl))

  increased coverage, decreased # of rules, approx. equal accuracy (PKDD-2000)

- In WRAcc computation, probabilities are estimated with relative frequencies, adapt:

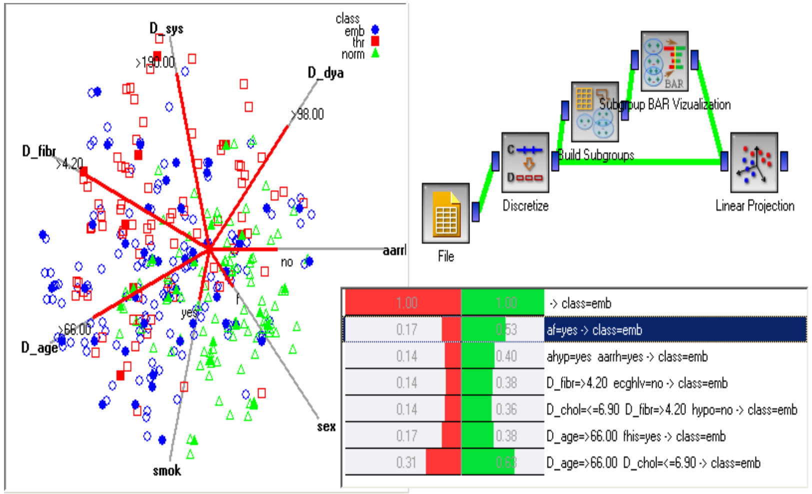
  WRAcc(Cl ← Cond) = p(Cond) (p(Cl|Cond) - p(Cl)) =

  n'(Cond)/N' (n'(Cl.Cond)/n'(Cond) - n'(Cl)/N')

  - N' : sum of weights of examples
  - n'(Cond) : sum of weights of all covered examples
  - n'(Cl.Cond) : sum of weights of all correctly covered examples

SD algorithms in the Orange DM Platform

- **Orange** data mining toolkit
  - classification and subgroup discovery algorithms
  - data mining workflows
  - visualization



- **SD Algorithms in Orange**
  - SD (Gamberger & Lavrač, JAIR 2002)
  - Apriori-SD (Kavšek & Lavrač, AAI 2006)
  - CN2-SD (Lavrač et al., JMLR 2004): Adapting CN2 classification rule learner to Subgroup Discovery

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning (more by Petra Kralj Novak)
- Hierarchical clustering

Association Rule Learning

Rules: X =>Y, if X then Y

X and Y are itemsets (records, conjunction of items), where items/features are binary-valued attributes)

Given: Transactions

itemsets (records)

	i1	i2	....	i50
t1	1	1		0
t2	0	1		0
	…	…	....	…

Find: A set of association rules in the form X =>Y

Example: Market basket analysis

beer & coke **=>** peanuts & chips (0.05, 0.65)

- Support: Sup(X,Y) = #XY**/**#D = p(XY)
- Confidence: Conf(X,Y) = #XY**/**#X = Sup(X,Y)/Sup(X) = = p(XY)**/**p(X) = p(Y|X)

Association Rule Learning: Examples

- Market basket analysis
  - beer & coke $\Rightarrow$ peanuts & chips (5%, 65%)

    (IF beer AND coke THEN peanuts AND chips)
  - Support 5%: 5% of all customers buy all four items
  - Confidence 65%: 65% of customers that buy beer and coke also buy peanuts and chips
- Insurance
  - mortgage & loans & savings $\Rightarrow$ insurance (2%, 62%)
  - Support 2%: 2% of all customers have all four
  - Confidence 62%: 62% of all customers that have mortgage, loan and savings also have insurance

Association Rule Learning

Given: a set of transactions D

Find: all association rules that hold on the set of transactions that have

- user defined minimum support, i.e., support > MinSup, and
- user defined minimum confidence, i.e., confidence > MinConf

It is a form of exploratory data analysis, rather than hypothesis verification

Searching for the associations

- Find all large itemsets
- Use the large itemsets to generate association rules
- If XY is a large itemset, compute

  r =support(XY) / support(X)
- If r > MinConf, then X $\Rightarrow$ Y holds

  (support > MinSup, as XY is large)

Large itemsets

- Large itemsets are itemsets that appear in at least MinSup transaction
- All subsets of a large itemset are large itemsets (e.g., if A,B appears in at least MinSup transactions, so do A and B)
- This observation is the basis for very efficient algorithms for association rules discovery (linear in the number of transactions)

Association rules vs. Classification rules

Association rules

- Exploration of dependencies
- Different combinations of dependent and independent attributes
- Complete search (all rules found)

Classification rules

- Focused prediction
- Predict one attribute (class) from the others
- Heuristic search (subset of rules found)

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering (more by Petra Kralj Novak)

Hierarchical clustering

- Algorithm (agglomerative hierarchical clustering):

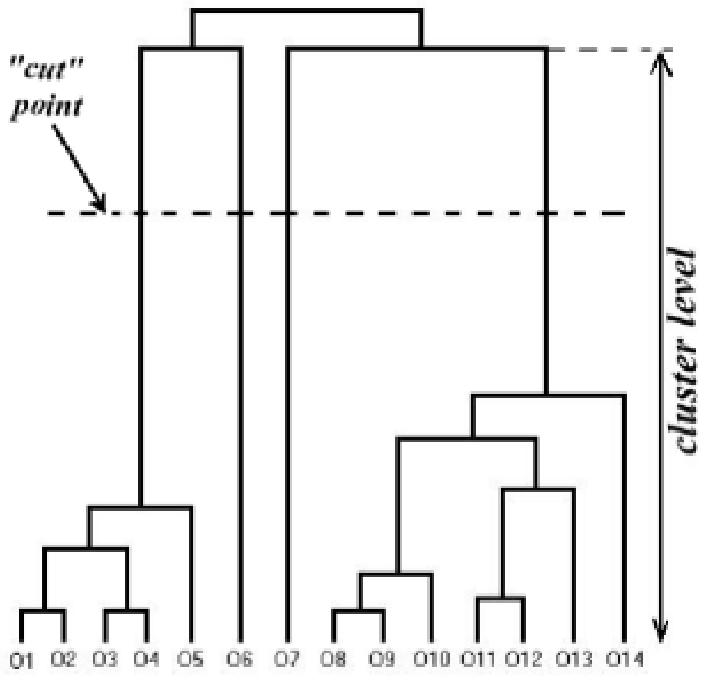
Each instance is a cluster;

repeat

find *nearest* pair C_i in C_j;
***fuse* C_i in C_j in a new cluster**
$C_r = C_i \cup C_j$;
determine *dissimilarities* between
C_r and other clusters;

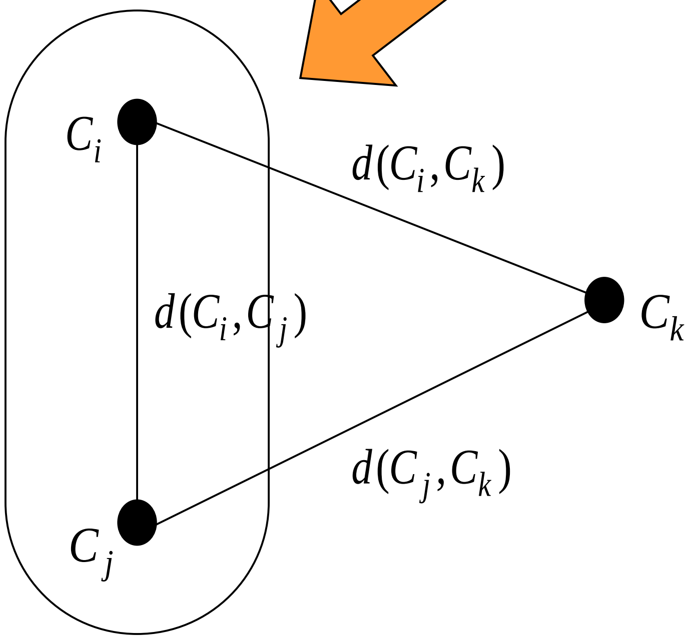
until one cluster left;

- Dendogram:



Hierarchical clustering

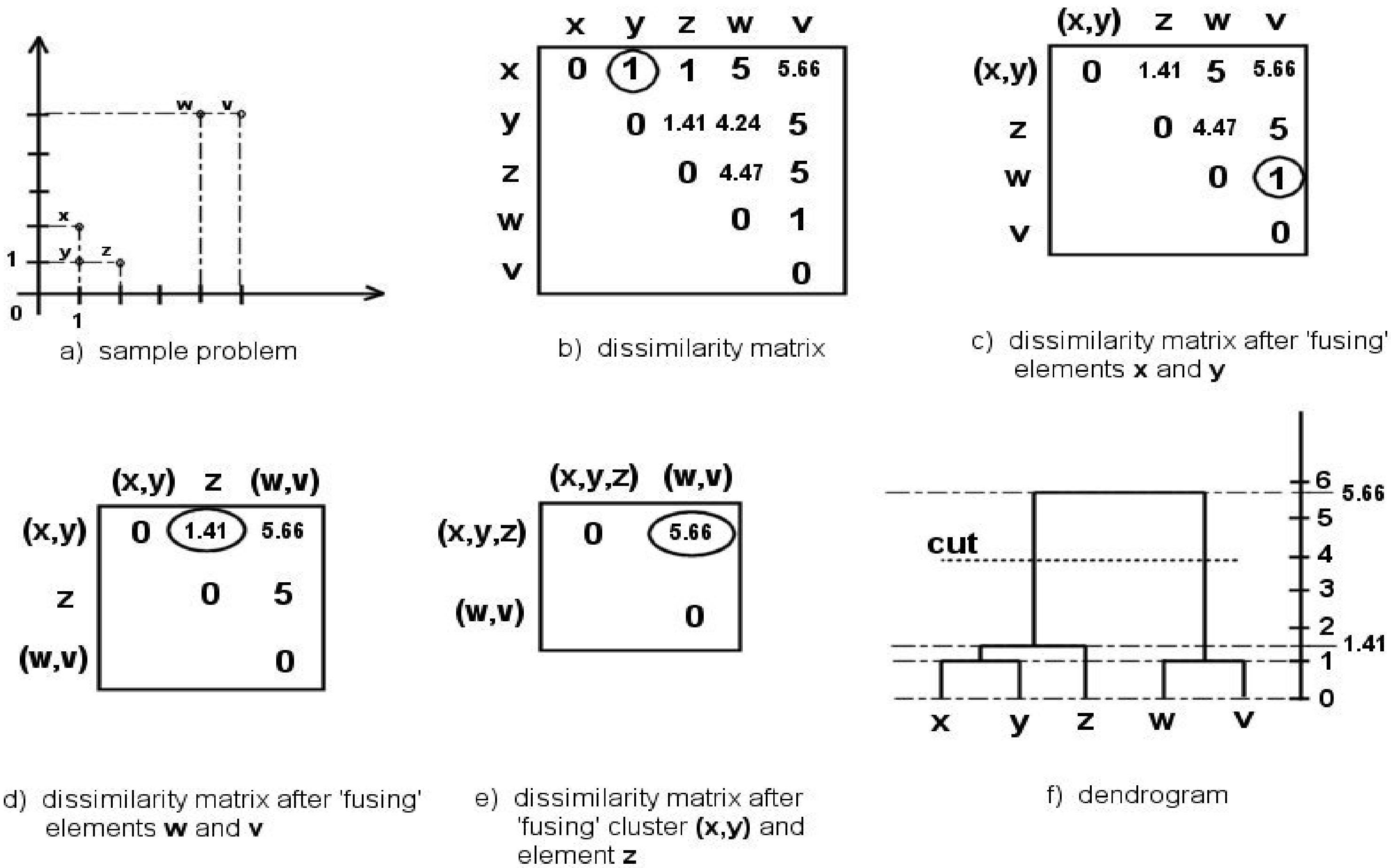
- Fusing the nearest pair of clusters



- Minimizing intra-cluster similarity
- Maximizing inter-cluster similarity

- Computing the dissimilarities from the “new” cluster

Hierarchical clustering: example



a) sample problem

b) dissimilarity matrix

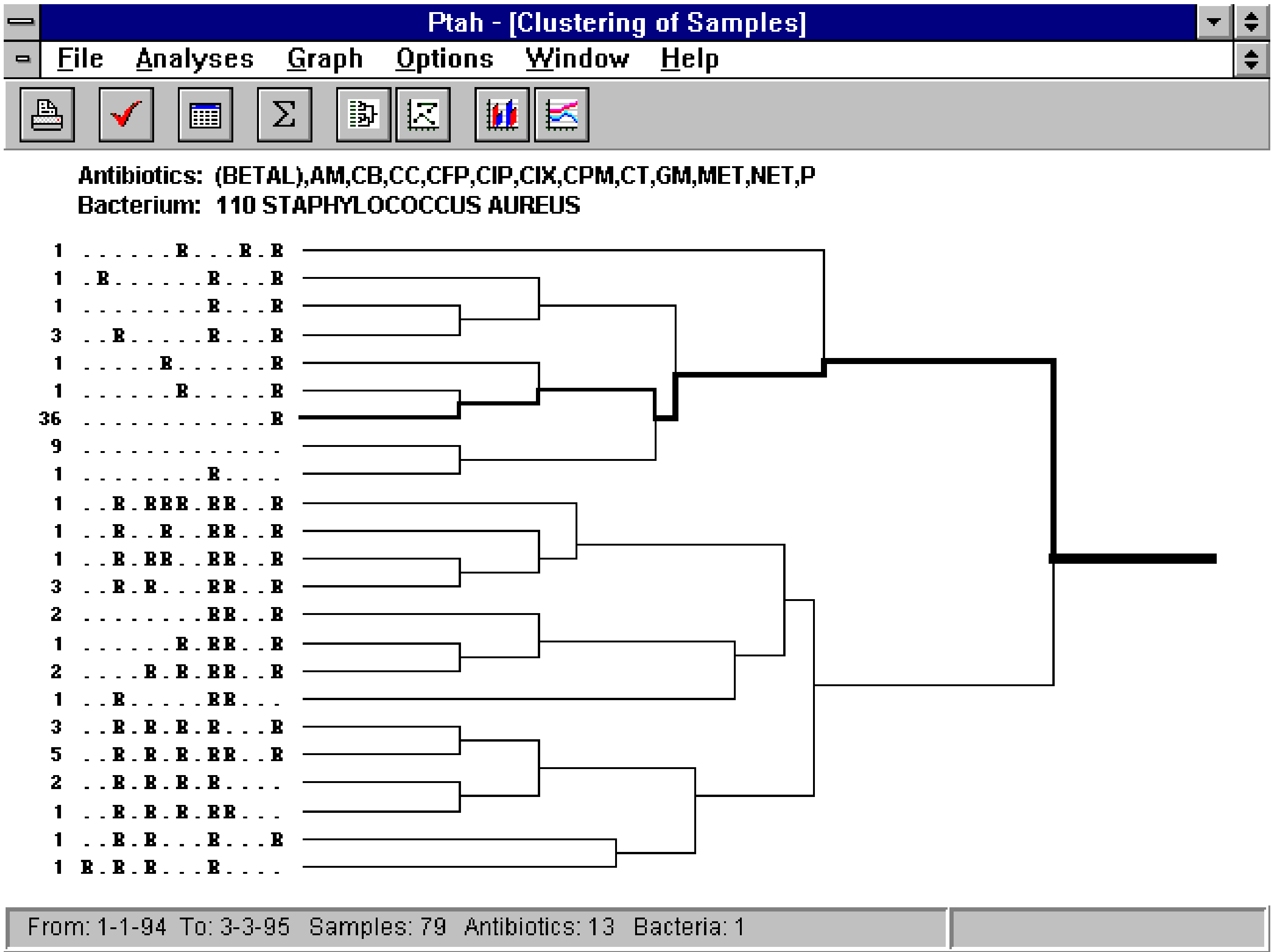
c) dissimilarity matrix after 'fusing' elements **x** and **y**

d) dissimilarity matrix after 'fusing' elements **w** and **v**

e) dissimilarity matrix after 'fusing' cluster **(x,y)** and element **z**

f) dendrogram

Results of clustering



A dendogram of resistance vectors

[Bohanec et al., “PTAH: A system for supporting nosocomial infection therapy”, IDAMAP book, 1997]

Course Outline

I. Introduction

- Data Mining and KDD process
- Introduction to Data Mining
- Data Mining platforms

II. Predictive DM Techniques

- Decision Tree learning
- Bayesian classifier
- Classification rule learning
- Classifier Evaluation

III. Regression

IV. Descriptive DM

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
  Hierarchical clustering

V. Relational Data Mining

- RDM and Inductive Logic Programming
- Propositionalization
- Semantic data mining

VI. Advanced Topics