Induction of Rules

Lecturer: JERZY STEFANOWSKI
Institute of Computing Sciences
Poznan University of Technology
Poznan, Poland

Lecture 6

SE Master Course 2008/2009

Update 2010

Rules - preliminaries

- **Rules** → the most popular symbolic representation of knowledge derived from data;
  - Natural and easy form of representation → possible inspection by human and their interpretation.
  - More comprehensive than any other knowledge representation!
- Standard form of rules
  IF *Conditions* THEN *Class*
- Other forms: Class IF Conditions; Conditions → Class

Example: The set of decision rules induced from PlaySport:

if outlook = overcast **then** Play = yes

if temperature = mild **and** humidity = normal **then** Play = yes

if outlook = rainy **and** windy = FALSE **then** Play = yes

if humidity = normal **and** windy = FALSE **then** Play = yes

if outlook = sunny **and** humidity = high **then** Play = no

if outlook = rainy **and** windy = TRUE **then** Play = no

Rules – more formal notations

- A rule corresponding to class K_j is represented as

  if P then Q

  where P = w_1 and w_2 and … and w_m is a condition part and Q is a decision part (object x satisfying P is assigned to class K_j)

- Elementary condition w_i (a *rel* v), where $a \in A$ and v is its value (or a set of values) and *rel* stands for an operator as $=, <, \leq, \geq, >$.

- $[P]$ is a cover of a condition part of a rule $\rightarrow$ a subset of examples satisfying P.

  - *if* ($a2$ = small) *and* ($a3 \leq 2$) *then* (d = C1) $\{x1, x7\}$

- A rule is certain / discriminant in DT iff $[P] = \bigcap [w_i] \subseteq [K_j]$, otherwise ($P \cap K_j \neq \emptyset$) the rule is partly discriminating.

An example of rules induced from data table

Minimal set of rules

- *if* (*a*2 = s) $\wedge$ (*a*3 $\leq$ 2) *then* (*d* = C1) {*x*1,*x*7}
- *if* (*a*2 = n) $\wedge$ (*a*4 = c) *then* (*d* = C1) {*x*3,*x*4}
- *if* (*a*2 = w) *then* (*d* = C2) {*x*2,*x*6}
- *if* (*a*1 = f) $\wedge$ (*a*4 = a) *then* (*d* = C2) {*x*5,*x*8}

Partly discriminating rule:

- *if* (*a1*=m) *then* (*d*=C1) {*x1*,*x3*,*x7* | *x6*} 3/4

id.	a_1	a_2	a_3	a_4	d
x_1	m	s	1	a	C1
x_2	f	w	1	b	C2
x_3	m	n	3	c	C1
x_4	f	n	2	c	C1
x_5	f	n	2	a	C2
x_6	m	w	2	c	C2
x_7	m	s	2	b	C1
x_8	f	s	3	a	C2

Why Decision Rules?

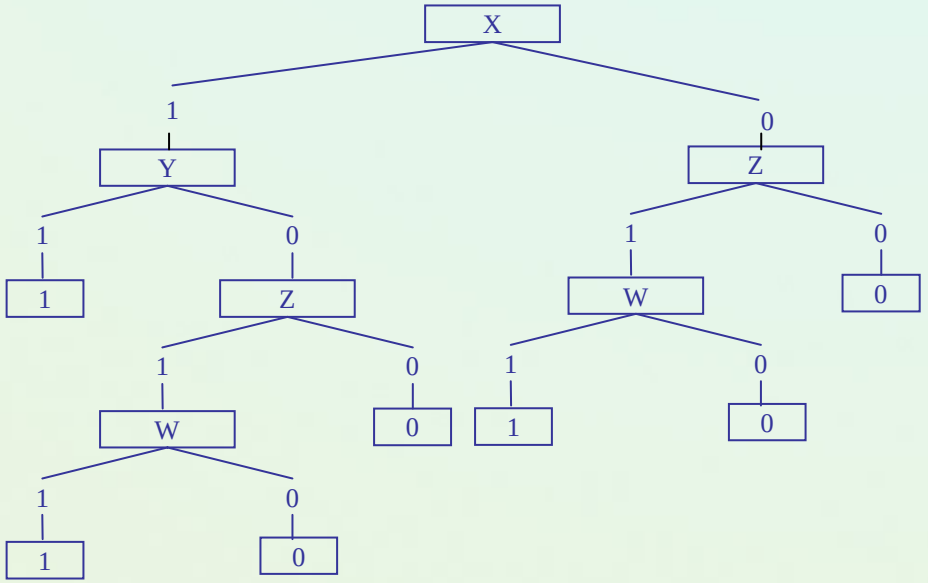
- Decision rules are more compact.
- Decision rules are more understandable and natural for human.
- Better for descriptive perspective in data mining.
- Can be nicely combined with background knowledge and more advanced operations, …

Example: Let $X \in \{0,1\}$, $Y \in \{0,1\}$, $Z \in \{0,1\}$, $W \in \{0,1\}$. The rules are:

if X=1 and Y=1 **then** 1

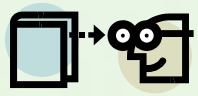
if Z=1 and W=1 **then** 1

Otherwise 0;



How to learn decision rules?

- Typical algorithms based on the scheme of a sequential covering and heuristically generate a minimal set of rule covering examples:
  - see, e.g., AQ, CN2, LEM, PRISM, MODLEM, Other ideas – PVM, R1 and RIPPER).
- Other approaches to induce „richer” sets of rules:
  - Satisfying some requirements (Explore, BRUTE, or modification of association rules, „Apriori-like”).
  - Based on local „reducts” $\rightarrow$ boolean reasoning or LDA.
- Specific optimization, eg. genetic approaches.
- Transformations of other representations:
  - Trees $\rightarrow$ rules.
  - Construction of (fuzzy) rules from ANN.



Covering algorithms

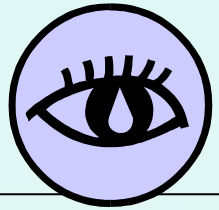
- A strategy for generating a rule set directly from data:
  - for each class in turn find a rule set that covers all examples in it (excluding examples not in the class).
- The main procedure is iteratively repeated for each class.
  - Positive examples from this class vs. negative examples.
- This approach is called a *covering* approach because at each stage a rule is identified that covers some of the examples (then these examples are skipped from consideration for the next rules).
- A sequential approach.
  - For a given class it conducts in a stepwise way a general to specific search for the best rules (learn-one-rule) guided by the evaluation measures.

General schema of inducing minimal set of rules

- The procedure conducts a general to specific (greedy) search for the best rules (**learn-one-rule**) guided by the evaluation measures.
- At each stage add to the current condition part next elementary tests that optimize possible rule's evaluation (no backtracking).

Procedure Sequential covering (K_j Class; *A* attributes; *E* examples,
τ - acceptance threshold);
begin
  R := $\emptyset$; {set of induced rules}
  r := ***learn-one-rule***(Y_j Class; *A* attributes; *E* examples)
  while ***evaluate***(*r*,*E*) > τ **do**
  begin
    R := *R* $\cup$ *r*;
    E := *E* \ [*R*]; {remove positive examples covered by *R*}
    r := ***learn-one-rule***(*K*j Class; *A* attributes; *E* examples);
  end;
return *R*
end.

The contact lenses data



Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	No	Reduced	None
Young	Myope	No	Normal	Soft
Young	Myope	Yes	Reduced	None
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	No	Reduced	None
Young	Hypermetrope	No	Normal	Soft
Young	Hypermetrope	Yes	Reduced	None
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	No	Reduced	None
Pre-presbyopic	Myope	No	Normal	Soft
Pre-presbyopic	Myope	Yes	Reduced	None
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	No	Reduced	None
Pre-presbyopic	Hypermetrope	No	Normal	Soft
Pre-presbyopic	Hypermetrope	Yes	Reduced	None
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	No	Reduced	None
Presbyopic	Myope	No	Normal	None
Presbyopic	Myope	Yes	Reduced	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	No	Reduced	None
Presbyopic	Hypermetrope	No	Normal	Soft
Presbyopic	Hypermetrope	Yes	Reduced	None
Presbyopic	Hypermetrope	Yes	Normal	None

Inducing rules by PRISM from contact lens data

- Rule we seek:

```
If ?
 then recommendation = hard
```

- Possible conditions:

Age = Young	2/8
Age = Pre-presbyopic	1/8
Age = Presbyopic	1/8
Spectacle prescription = Myope	3/12
Spectacle prescription = Hypermetrope	1/12
Astigmatism = no	0/12
Astigmatism = yes	4/12
Tear production rate = Reduced	0/12
Tear production rate = Normal	4/12

PRISM - Evaluation of candidates for a rule:

- High accuracy P(K|R);
- High coverage |[P]|

ACK: slides coming from witten&eibe WEKA

Modified candidate for a rule and covered data

- Condition part of the rule with the best elementary condition added:

```
If astigmatism = yes
    then recommendation = hard
```

- Examples covered by the first condition part:

Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	Yes	Reduced	None
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	Yes	Reduced	None
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	Yes	Reduced	None
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	Yes	Reduced	None
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	Yes	Reduced	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	Yes	Reduced	None
Presbyopic	Hypermetrope	Yes	Normal	None

Further specialization of conditions

- Current state:

```
If astigmatism = yes
    and ?
  then recommendation = hard
```

- Possible conditions:

Age = Young	2/4
Age = Pre-presbyopic	1/4
Age = Presbyopic	1/4
Spectacle prescription = Myope	3/6
Spectacle prescription = Hypermetrope	1/6
Tear production rate = Reduced	0/6
Tear production rate = Normal	4/6

Two conditions in the rule

- The rule with the next best condition added:

```
If astigmatism = yes
    and tear production rate = normal
  then recommendation = hard
```

- Examples covered by modified rule:

Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	Yes	Normal	None

Further specialization of the candidate for a rule

- The current state:

```
If astigmatism = yes
     and tear production rate = normal
     and ?
   then recommendation = hard
```

- Possible conditions:

Age = Young	2/2
Age = Pre-presbyopic	1/2
Age = Presbyopic	1/2
Spectacle prescription = Myope	3/3
Spectacle prescription = Hypermetrope	1/3

- Tie between the first and the fourth test
  - We choose the one with greater coverage

The result for class „hard”

- Final rule:

```
If astigmatism = yes
   and tear production rate = normal
   and spectacle prescription = myope
   then recommendation = hard
```

- Second rule for recommending “hard lenses”:
  (built from instances not covered by first rule)

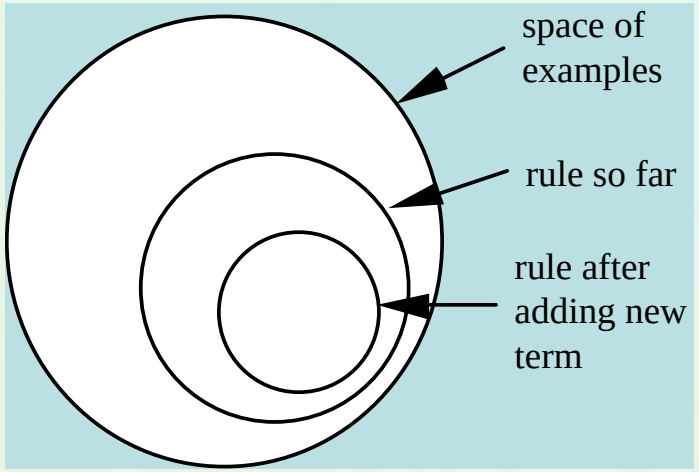
```
If age = young and astigmatism = yes
   and tear production rate = normal
   then recommendation = hard
```

- These two rules cover all “hard lenses”:
  - Process is repeated with other two classes

Thnaks to witten&eibe

A search in a simple covering algorithm

- Generates a rule by adding tests that maximize rule's accuracy
- Similar to situation in decision trees: problem of selecting an attribute to split on
  - But: decision tree inducer maximizes overall purity
- Each new term reduces rule's coverage:



LEM2 algorithm with rough approximations

- Grzymala 92; - induces rules from rough sets approximations of inconsistent decision classes.
- Sequential covering (similar to PRISM but another evaluation criteria)
- A heuristic approach to minimal set of rules; it is based on iterative computing the single local covering $\mathbf{T}$ (see it as a set of cond. parts) of each concept (approximation) in a decision table
- $\mathbf{T}$ is a local covering of K iff

  Each member $T \in \mathbf{T}$ is minimal

$$\bigcup_{T \in \mathbf{T}} [T] = K$$

  $\mathbf{T}$ is minimal, i.e. contains the smallest number of elements *T*.

LEM2 – An Example (1)

U	*Headache*	*Nausea*	*Temp.*	*Flu*
x1	no	no	normal	No
x2	yes	no	high	Yes
x3	yes	yes	high	Yes
x4	yes	no	normal	No
x5	no	no	high	No
x6	no	no	high	Yes

IND: {x1}, {x2}, {x3}, {x4}, {x5,x6}
YES: lower appr. {x2,x3}
upper {x2,x3,x5,x6}
NO: lower approx. {x1,x4}
upper {x1,x4,x5,x6}
Inconsistent boundary {x5,x6}

Certajn rules for **(Flue=Yes)**: Concept {x2,x3}

(headache,yes)	{x2,x3+ ; x4-}
(nausea,no)	{x2+ ; x1,x4,x5,x6-}
(nausea,yes)	{x3+ }
(temperature,high)	{x2,x3+ ; x5,x6-}

Choose t_1 (headache,yes) but it {x2,x3+ ; x4-} $\not\subset$ {x2,x3}, so look for next, new condition ; Add (temperature,high),
now t1$\cap$t2= {x2,x3+ ; x4-} $\cap$ {x2,x3+ ; x5,x6-} $\subseteq$ {x2,x3}
Finally, the rule **(headache=yes) $\cap$ (temperature=high) $\rightarrow$(Flue=Yes)** describes all examples from this concept

LEM2 – An Example (2)

U	*Headache*	*Nausea*	*Temp.*	*Flu*
x1	no	no	normal	No
x2	yes	no	high	Yes
x3	yes	yes	high	Yes
x4	yes	no	normal	No
x5	no	no	high	No
x6	no	no	high	Yes

IND: {x1}, {x2}, {x3}, {x4}, {x5,x6}
YES: lower appr. {x2,x3}
upper {x2,x3,x5,x6}
NO: lower approx. {x1,x4}
upper {x1,x4,x5,x6}

Certajn rules for **(Flue=No)**: Concept {x1,x4}

(headache,no}	{x1+; x5,x6-}
(headache,yes)	{x4+ ; x2,x3-}
(nausea,no)	{x1,x4+;x2,x5,x6-}
(temperature,normal)	{x1,x4+ ; ∅}

Choose t_1 (temperature,normal),
now t1= {x1,x4+ ; ∅-} $\subseteq$ {x1,x4}
Finally, the rule **(temperature=normal) →(Flue=No)** describes all examples from this concept

MODLEM – Algorithm for rule induction

- MODLEM [Stefanowski 98] generates a minimal set of rules.
- Its extra specificity – handling directly numerical attributes during rule induction; elementary conditions, e.g. $(a \geq v)$, $(a < v)$, $(a \in [v_1, v_2))$ or $(a = v)$.
- Elementary condition evaluated by one of three measures: class entropy, Laplace accuracy or Grzymala 2-LEF.

obj.	*a*1	*a*2	*a*3	*a*4	*D*
*x*1	m	2.0	1	a	C1
*x*2	f	2.5	1	b	C2
*x*3	m	1.5	3	c	C1
*x*4	f	2.3	2	c	C1
*x*5	f	1.4	2	a	C2
*x*6	m	3.2	2	c	C2
*x*7	m	1.9	2	b	C1
*x*8	f	2.0	3	a	C2

if (*a*1 = m) *and* (*a*2 ≤ 2.6) *then* (*D* = C1) {*x*1,*x*3,*x*7}

if (*a*2 ∈ [1.45, 2.4]) *and* (*a*3 ≤ 2) *then* (*D* = C1) {*x*1,*x*4,*x*7}

if (*a*2 ≥ 2.4) *then* (*D* = C2) {*x*2,*x*6}

if (*a*1 = f) *and* (*a*2 ≤ 2.15) *then* (*D* = C2) {*x*5,*x*8}

Mushroom data (UCI Repository)

- Mushroom records drawn from The Audubon Society Field Guide to North American Mushrooms (1981).
- This data set includes descriptions of hypothetical samples corresponding to 23 species of mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility.
- Number of examples: 8124.
- Number of attributes: 22 (all nominally valued)
- Missing attribute values: 2480 of them.
- Class Distribution:
  - -- edible: 4208 (51.8%)
  - -- poisonous: 3916 (48.2%)

MOLDEM rule set (Implemented in WEKA)

=== Classifier model (full training set) ===

Rule 1.(odor is in: {n, a, l})&(spore-print-color is in: {n, k, b, h, o, u, y, w})&(gill-size = b) => (class = e); [3920, 3920, 93.16%, 100%]

Rule 2.(odor is in: {n, a, l})&(spore-print-color is in: {n, h, k, u}) => (class = e); [3488, 3488, 82.89%, 100%]

Rule 3.(gill-spacing = w)&(cap-color is in: {c, n}) => (class = e); [304, 304, 7.22%, 100%]

Rule 4.(spore-print-color = r) => (class = p); [72, 72, 1.84%, 100%]

Rule 5.(stalk-surface-below-ring = y)&(gill-size = n) => (class = p); [40, 40, 1.02%, 100%]

Rule 6.(odor = n)&(gill-size = n)&(bruises? = t) => (class = p); [8, 8, 0.2%, 100%]

Rule 7.(odor is in: {f, s, y, p, c, m}) => (class = p); [3796, 3796, 96.94%, 100%]

Number of rules: 7

Number of conditions: 14

Approaches to Avoiding Overfitting

- **Pre-pruning:** stop learning the decision rules before they reach the point where they perfectly classify the training data

- **Post-pruning:** allow the decision rules to overfit the training data, and then post-prune the rules.

Pre-Pruning

The criteria for stopping learning rules can be:

- **minimum purity** criterion requires a certain percentage of the instances covered by the rule to be positive;
- **significance test** determines if there is a significant difference between the distribution of the instances covered by the rule and the distribution of the instances in the training sets.

Post-Pruning (Grow, IREP)

1. Split instances into *Growing Set* and *Pruning Set*;
2. Learn set *SR* of rules using *Growing Set*;
3. Find the best simplification *BSR* of *SR*.
4. **while** (Accuracy(*BSR*, *Pruning Set*) >
   Accuracy(*SR*, *Pruning Set*)) **do**
4.1 *SR* = *BSR*;
4.2 Find the best simplification *BSR* of *SR*.
5. **return** *BSR*;

Applying rule set to classify objects

- **Matching** a new object description *x* to condition parts of rules.
  - Either object's description satisfies all elementary conditions in a rule, or not.

  IF (a1=L) and (a3≥ 3) THEN Class +

  x → (a1=L),(a2=s),(a3=7),(a4=1)
- Two ways of assigning x to class K depending on the set of rules:
  - Unordered set of rules (AQ, CN2, PRISM, LEM)
  - Ordered list of rules (CN2, c4.5rules)

Applying rule set to classify objects

- The rules are ordered into priority decision list!

  Another way of rule induction – rules are learned by first determining Conditions and then Class (CN2)

Notice: mixed sequence of classes K1,…, K in a rule list

But: ordered execution when classifying a new instance: rules are sequentially tried and the first rule that ‘fires’ (covers the example) is used for final decision

Decision list {R1, R2, R3, …, D}: rules Ri are

interpreted as **if-then-else** rules

If no rule fires, then DefaultClass (majority class in input data)

Applying unordered rule set to classify objects

- An unordered set of rules $\rightarrow$ three situations:
  - Matching to rules indicating the same class.
  - Multiple matching to rules from different classes.
  - No matching to any rule.
- An example:
- e1={(Age=m), (Job=p),(Period=6),(Income=3000),(Purpose=K)}
  - rule 3: if (Period$\in$[3.5,12.5)) then (Dec=d) [2]
  - Exact matching to rule 3. $\rightarrow$ Class (Dec=d)
- e2={(Age=m), (Job=p),(Period=2),(Income=2600),(Purpose=M)}
  - No matching!

Solving conflict situations

- LERS classification strategy (Grzymala 94)
  - Multiple matching
    - Two factors: *Strength*(*R*) – number of learning examples correctly classified by *R* and final class *Support*(*Y*i):

$$\sum_{\text{matching rules R for Yi}} Strength(R)$$

  - Partial matching
    - Matching factor *MF*(*R*) and

$$\sum_{\text{partially match. rules R for Yi}} MF(R) \cdot Strength(R)$$

- e2={(Age=m), (Job=p), (Period=2),(Income=2600),(Purpose=M)}
  - Partial matching to rules 2 , 4 and 5 for all with MF = 0.5
  - Support(r) = 0.5·2 =1 ; Support(d) = 0.5·2+0.5·2=2
- Alternative approaches – e.g. nearest rules (Stefanowski 95)
- Instead of MF use a kind of normalized distance *x* to conditions of *r*

Different perspectives of rule application

- In a descriptive perspective
  - To present, analyse the relationships between values of attributes, to explain and understand classification patterns
- In a prediction/classification perspective,
  - To predict value of decision class for new (unseen) object)

Perspectives are different;
Moreover rules are evaluated in a different ways!

Explore algorithm (Stefanowski, Vanderpooten)

- Another aim of rule induction
  - to extract from data set inducing **all rules** that *satisfy* some *user's requirements* connected with *his interest* (regarding, e.g. the strength of the rule, level of confidence, length, sometimes also emphasis on the syntax of rules).
- Special technique of exploration the space of possible rules:
  - Progressively generation rules of increasing size using in the most efficient way some 'good' pruning and stopping condition that reject unnecessary candidates for rules.
- Similar to adaptations of Apriori principle for looking frequent itemsets [AIS94]; Brute [Etzioni]

Any questions, remarks?