# DATA MINING - 1DL105, 1DL111

## Fall 2007

An introductory class in data mining

http://user.it.uu.se/~udbl/dm-ht2007/
alt. http://www.it.uu.se/edu/course/homepage/infoutv/ht07

Kjell  Orsborn
Uppsala Database Laboratory
Department of Information Technology, Uppsala University,
Uppsala, Sweden

# Data Mining
# Classification: Alternative Techniques

## (Tan, Steinbach, Kumar ch. 5)

Kjell Orsborn

Department of Information Technology
Uppsala University, Uppsala, Sweden

# Rule-based classifier

- Classify records by using a collection of “if…then…” rules
- Rule: $(Condition) \rightarrow y$
  - where
    - *Condition* is a conjunctions of attributes
    - *y* is the class label
  - *LHS*: rule antecedent or condition
  - *RHS*: rule consequent
  - Examples of classification rules:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) → Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) → Evade=No

# Rule-based classifier example

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---|---|---|---|---|---|
| human | warm | yes | no | no | mammals |
| python | cold | no | no | no | reptiles |
| salmon | cold | no | no | yes | fishes |
| whale | warm | yes | no | yes | mammals |
| frog | cold | no | no | sometimes | amphibians |
| komodo | cold | no | no | no | reptiles |
| bat | warm | yes | yes | no | mammals |
| pigeon | warm | no | yes | no | birds |
| cat | warm | yes | no | no | mammals |
| leopard shark | cold | yes | no | yes | fishes |
| turtle | cold | no | no | sometimes | reptiles |
| penguin | warm | no | no | sometimes | birds |
| porcupine | warm | yes | no | no | mammals |
| eel | cold | no | no | yes | fishes |
| salamander | cold | no | no | sometimes | amphibians |
| gila monster | cold | no | no | no | reptiles |
| platypus | warm | no | no | no | mammals |
| owl | warm | no | yes | no | birds |
| dolphin | warm | yes | no | yes | mammals |
| eagle | warm | no | yes | no | birds |

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) → Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) → Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

# Application of Rule-based classifier

- A rule *r* covers an instance **x** if the attributes of the instance satisfy the condition of the rule

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds

R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals

R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---|---|---|---|---|---|
| hawk | warm | no | yes | no | ? |
| grizzly bear | warm | yes | no | no | ? |

The rule R1 covers a hawk => Bird

The rule R3 covers the grizzly bear => Mammal

UPPSALA UNIVERSITET

# Rule coverage and accuracy

- Coverage of a rule:
  - Fraction of records that satisfy the antecedent of a rule
- Accuracy of a rule:
  - Fraction of records that satisfy both the antecedent and consequent of a rule

| *Tid* | Refund | Marital Status | Taxable Income | Class |
|---|---|---|---|---|
| 1 | Yes | **Single** | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | **Single** | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | **Single** | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | **Single** | 90K | **Yes** |

**(Status=Single) → No**

**Coverage = 40%, Accuracy = 50%**

UPPSALA UNIVERSITET

# How does rule-based classifier work?

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds

R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals

R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---|---|---|---|---|---|
| lemur | warm | yes | no | no | ? |
| turtle | cold | no | no | sometimes | ? |
| dogfish shark | cold | yes | no | yes | ? |

A lemur triggers rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

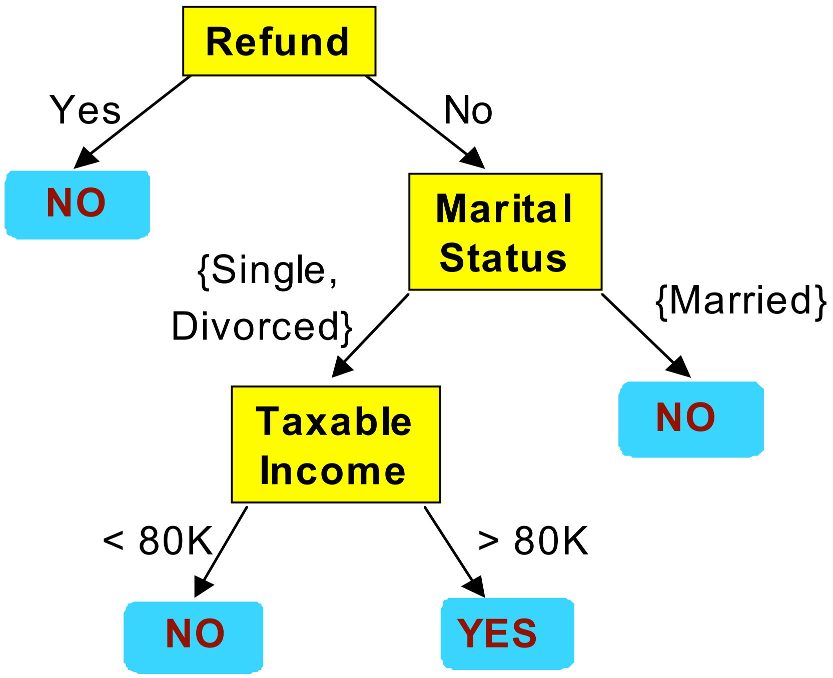
A dogfish shark triggers none of the rules

# Characteristics of Rule-based classifier

- Mutually exclusive rules
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered by at most one rule

- Exhaustive rules
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered by at least one rule

# From Decision trees to Rules



## Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

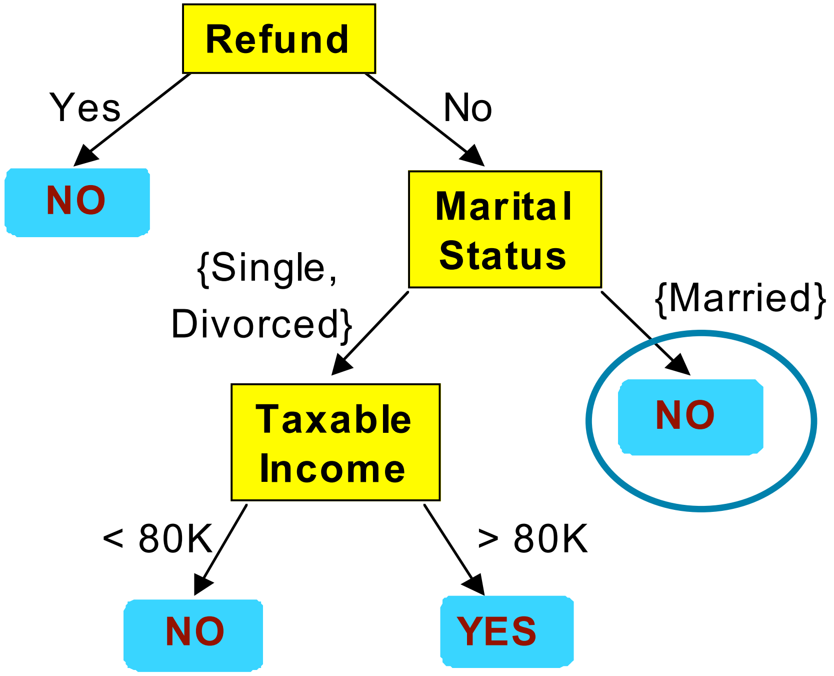
(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

**Rules are mutually exclusive and exhaustive**

**Rule set contains as much information as the tree**

# Rules can be simplified



| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |

**Initial Rule:** **(Refund=No) ∧ (Status=Married) → No**

**Simplified Rule:** **(Status=Married) → No**

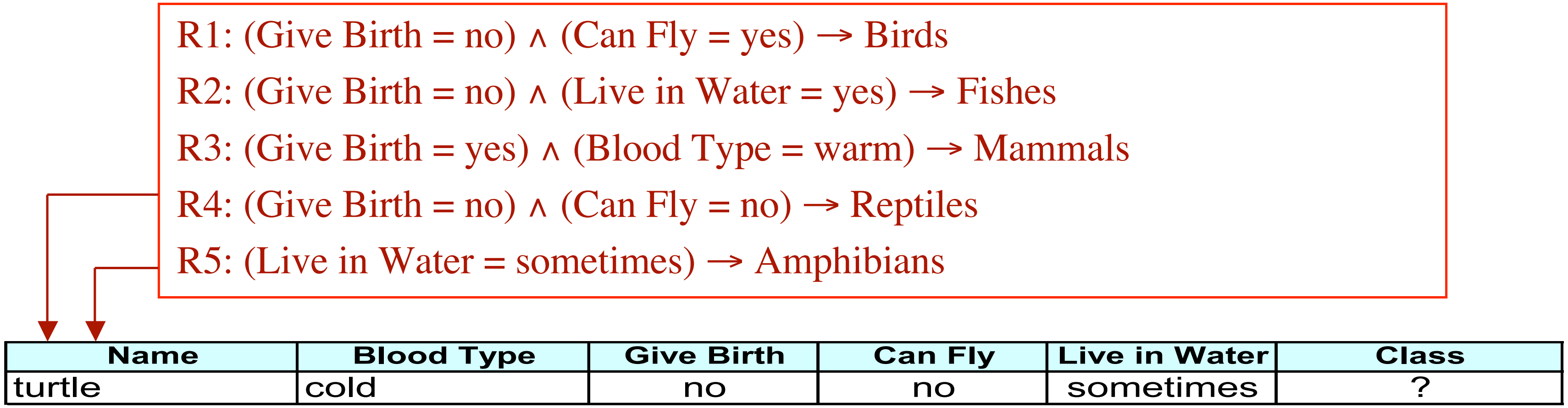
# Effect of rule simplification

- Rules are no longer mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes

- Rules are no longer exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class

# Ordered rule set

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a decision list
- When a test record is presented to the classifier
  - It is assigned to the class label of the highest ranked rule it has triggered
  - If none of the rules fired, it is assigned to the default class



R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds

R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals

R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---|---|---|---|---|---|
| turtle | cold | no | no | sometimes | ? |

# Rule ordering schemes

- Rule-based ordering
  - Individual rules are ranked based on their quality
- Class-based ordering
  - Rules that belong to the same class appear together

### Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

### Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

# Building classification rules

- Direct Method:
    - Extract rules directly from data
    - e.g.: RIPPER, CN2, Holte’s 1R
- Indirect Method:
    - Extract rules from other classification models (e.g. decision trees, neural networks, etc).
    - e.g: C4.5rules

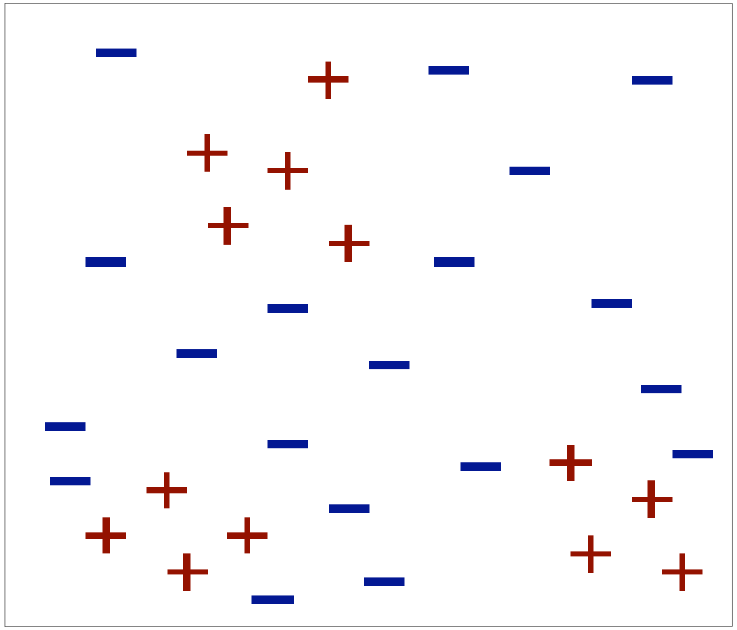
# Direct method: Sequential covering

- Start from an empty rule
- Grow a rule using the Learn-One-Rule function
- Remove training records covered by the rule
- Repeat Step (2) and (3) until stopping criterion is met

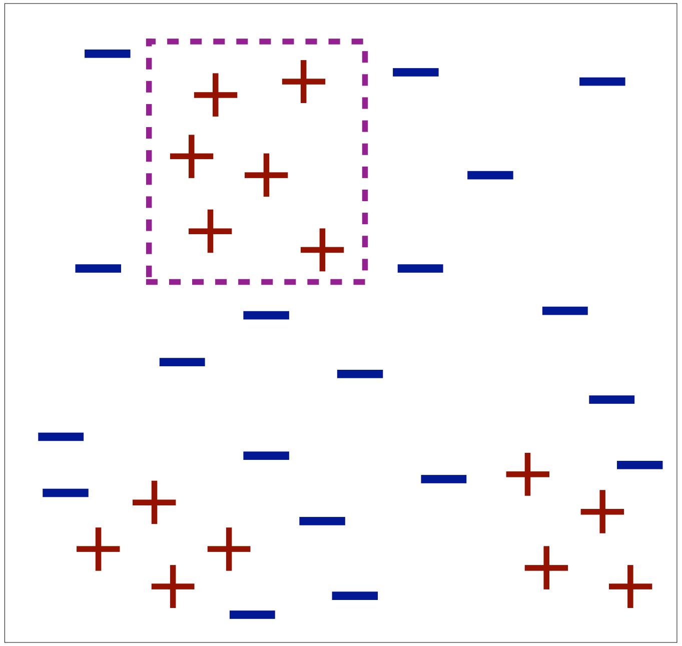
# Learning One Rule

- To learn one rule we use one of the strategies below:
  - Top-down:
    - Start with maximally general rule
    - Add literals one by one
  - Bottom-up:
    - Start with maximally specific rule
    - Remove literals one by one
  - Combination of top-down and bottom-up:
    - Candidate-elimination algorithm

# Example of sequential covering

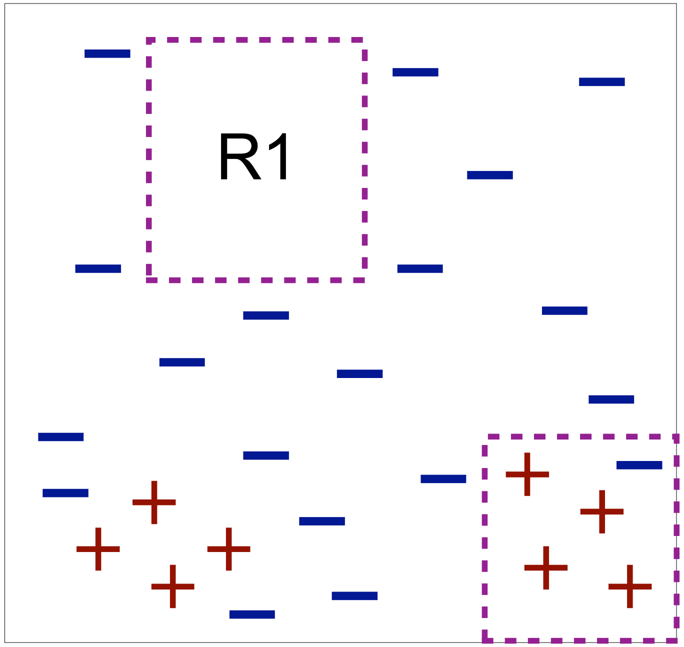


(i) Original Data

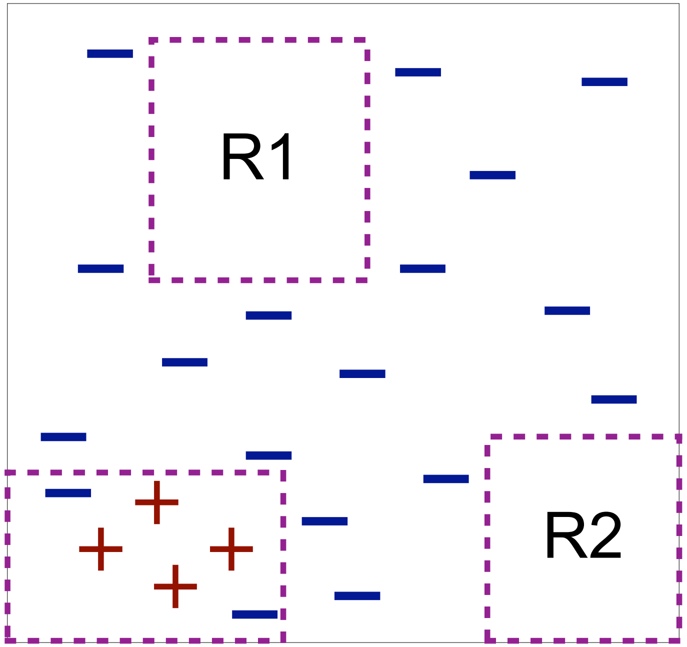


(ii) Step 1

# Example of sequential covering…



(iii) Step 2



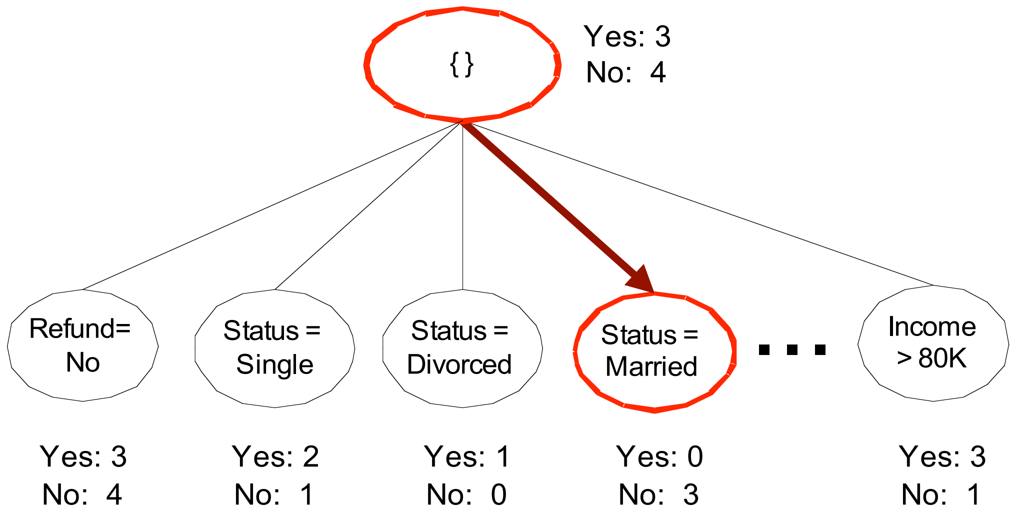
(iv) Step 3

# Aspects of sequential covering

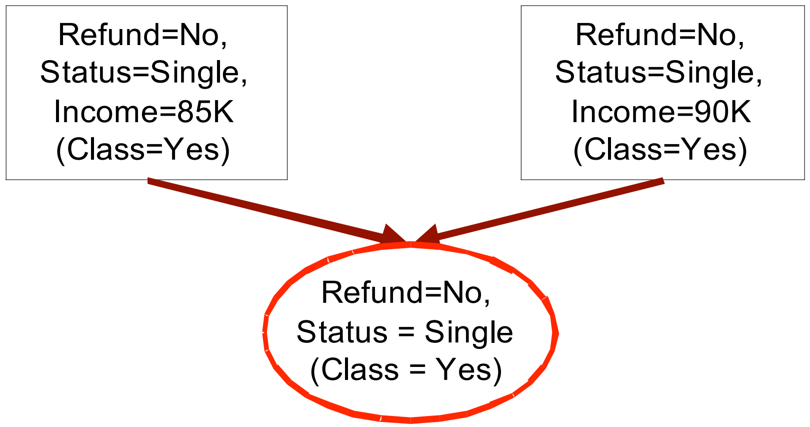
- Rule Growing
- Instance Elimination
- Rule Evaluation
- Stopping Criterion
- Rule Pruning

# Rule growing

## Two common strategies



(a) General-to-specific


Refund=No, Status=Single, Income=85K (Class=Yes)
Refund=No, Status=Single, Income=90K (Class=Yes)
Refund=No, Status = Single (Class = Yes)

(b) Specific-to-general

# Rule growing examples

- CN2 Algorithm:
  - Start from an empty conjunct: {}
  - Add conjuncts that minimizes the entropy measure: {A}, {A,B}, …
  - Determine the rule consequent by taking majority class of instances covered by the rule
- RIPPER Algorithm:
  - Start from an empty rule: {} => class
  - Add conjuncts that maximizes FOIL's information gain measure:
    - R0: {} => class (initial rule)
    - R1: {A} => class (rule after adding conjunct)
    - Gain(R0, R1) = t [ log (p1/(p1+n1)) – log (p0/(p0 + n0)) ]
    - where t: number of positive instances covered by both R0 and R1

      p0: number of positive instances covered by R0

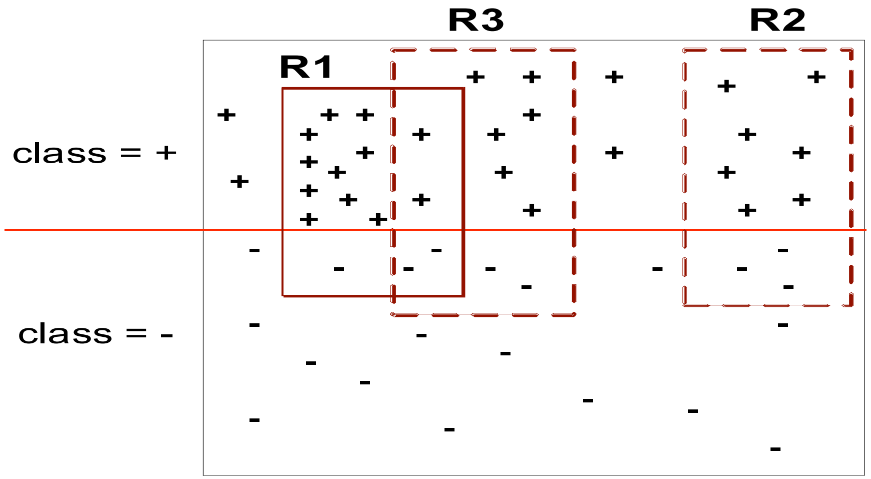
      n0: number of negative instances covered by R0

      p1: number of positive instances covered by R1

      n1: number of negative instances covered by R1

# Instance elimination

- Why do we need to eliminate instances?
  - Otherwise, the next rule is identical to previous rule
- Why do we remove positive instances?
  - Ensure that the next rule is different
- Why do we remove negative instances?
  - Prevent underestimating accuracy of rule
  - Compare rules R2 and R3 in the diagram



# Rule evaluation

- Heuristics for Learning One Rule - When is a rule “good”?
  - High accuracy;
  - Less important: high coverage.
- Metrics:
  - Accuracy, (relative frequency): $n_c/n$
  - Laplace: $(n_c+ 1)/(n+k)$
  - M-estimate of accuracy: $(n_c+ kp)/(n+k)$,
    - where $n_c$ is the number of correctly classified instances, and
    - n is the number of instances covered by the rule, and
    - p is the prior probablity of the class predicted by the rule, and
    - k is the number of classes or the weight of p.
  - Entropy
- The Laplace, M-estimate and Entropy metrics take rule coverage into account

# Stopping criterion and rule pruning

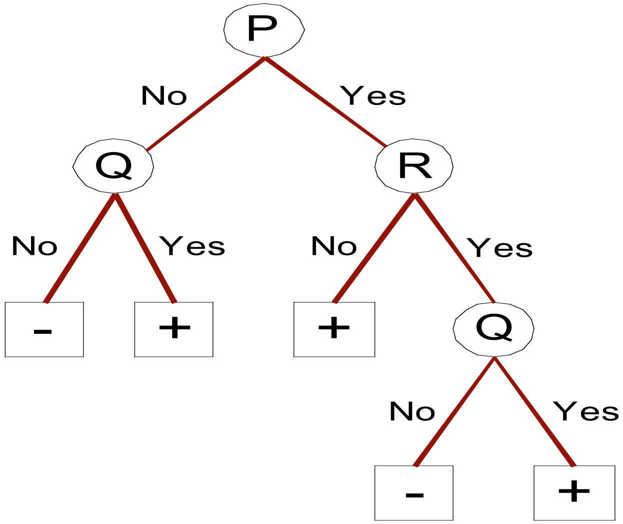
- Stopping criterion
  - Compute the gain
  - If gain is not significant, discard the new rule

- Rule Pruning
  - Similar to post-pruning of decision trees
  - Reduced Error Pruning:
    - Remove one of the conjuncts in the rule
    - Compare error rate on validation set before and after pruning
    - If error improves, prune the conjunct

# Summary of direct method

- Grow a single rule
- Remove instances from rule
- Prune the rule (if necessary)
- Add rule to Current Rule Set
- Repeat

# Indirect methods



**Rule Set**

r1: (P=No,Q=No) ==> -
r2: (P=No,Q=Yes) ==> +
r3: (P=Yes,R=No) ==> +
r4: (P=Yes,R=Yes,Q=No) ==> -
r5: (P=Yes,R=Yes,Q=Yes) ==> +

# Advantages of rule-based classifiers

- As highly expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees