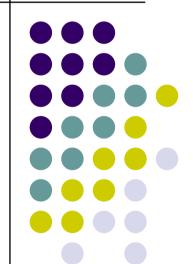
Rules Models

J. Savoy University of Neuchâtel

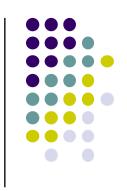


Ian H. Witten, Eibe Frank, M.A: Hall: Data Mining. Practical Machine Learning Tools and Techniques. Morgan Kaufmann

A. Rajaraman, D. Ullman: Mining of Massive Datasets. Cambridge University Press

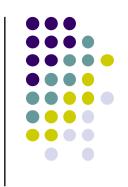
P. Flach: Machine Learning Cambridge University Press, 2012





- Represent the knowledge in the form of a set of rules
- The body of the rule indicates the conditions
 - Logical formula with connectors $\cup \cap \neg$
 - Constraints can be imposed in some methods (e.g., only conjunctive form)
 - May have literals (atomic expression) or more complex expressions





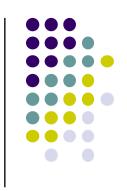
- Follow a general pattern
 IF (left) THEN (right)
 left = left-hand side = LHS = antecedent
 right = right-hand side = RHS = conclusion
- The head indicates the conclusion / class / label
- The body indicates the conditions
- The set of rules maybe ordered or unordered
- Example
 - If (length > 2) ∩ (dangerous=yes) then shark





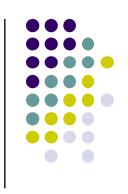
- Ordered Rule Lists
- Unordered Rule Lists





- Represent the instance set in the form of a set of rules
- The *order* of the rules is *important*
- We want a small set of rules
- Each rule must cover the largest possible subset of the training set
- The principle
 Build a conjunctive rule by adding a literal that improves
 its homogeneity
 - stop when some homogeneity criterion is satisfied or when encounter only simple cases





- Homogeneity criterion (or purity criterion)?
- Proportion of positives examples covered by the literal (assuming a binary classification)
 Can be represented by a probability estimate
- Other purity measures are possible but tend to produce the same result
- The main idea:
 - Select the purest literal, form a rule
 - From the training set, ignore examples covered by the proposed rule
 - Continue until some stopping criterion is reached





- Discriminate between two classes
- Attributes / values
 - Length = 3/4/5
 - Gills = True / False
 - Beak = True / False
 - Teeth = Many / Few
- Species
 - Dolphin (or not)

the body length
present or absent
present or absent
how many teeth





Set of examples (training set)

Length	Gills	Beak	Teeth	Dolphin
3	no	yes	many	yes
4	no	yes	many	yes
3	no	yes	few	yes
5	no	yes	many	yes
5	no	yes	few	yes
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no

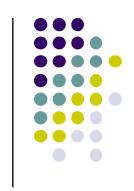


- Starting point: 5 pos, 5 neg
- List of possible literal

length = 3	2 pos, 0 neg
gills = <i>yes</i>	0 pos, 4 neg
beak = <i>yes</i>	5 pos, 3 neg
beak = <i>no</i>	0 pos, 2 neg
teeth = few	2 pos, 1 neg
teeth = many	3 pos, 4 neg

. . .

- How many literal can we have? $3 \times 2 \times 2 \times 2 = 24$
- Which one is the purest?
 As measure: min(p, 1-p) or ½ |p ½|
 with p = probability of the pos. category



- Starting point: 5 pos, 5 neg
- List of possible literal

length = 3	2 pos, 0 neg: $p = 2h$	¹ 2, q=0/2
gills = yes	0 pos, 4 neg: $p = 0$	⁷ 4, q=4/4
beak = <i>yes</i>	5 pos, 3 neg: $p = 5h$	⁷ 8, q=3/8
beak = <i>no</i>	0 pos, 2 neg: $p = 0$	¹ 2, q=2/2
teeth = many	3 pos, $4 neg$: $p = 3/4$	7, q=4/7

. . .

Not clear!

IF (gills = yes) THEN class \otimes IF (beak = no) THEN class \otimes IF (length = 3) THEN class \oplus



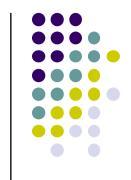


 We select the rule covering the largest number of observations

R1: IF (gills = yes) THEN class \otimes

The remaining training set

Length	Gills	Beak	Teeth	Dolphin
3	no	yes	many	yes
4	no	yes	many	yes
3	no	yes	few	yes
5	no	yes	many	yes
5	no	yes	few	yes
4	no	yes	few	no



 We select the rule covering the largest number of observations

R2: IF (teeth = many) THEN class ⊕

The remaining training set

Length	Gills	Beak	Teeth	Dolphin
3	no	yes	few	yes
5	no	yes	few	yes
4	no	yes	few	no





 We select the rule covering the largest number of observations

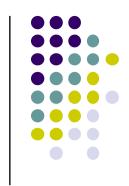
R3: IF (length = 4) THEN class \otimes

• The remaining training set

Length	Gills	Beak	Teeth	Dolphin
3	no	yes	few	yes
5	no	yes	few	yes

all instances belong to the same class

R4: ELSE class ⊕



We have found the ordered rule list

R1: IF (gills = yes) THEN class \otimes

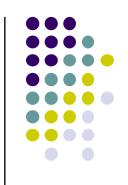
R2: IF (teeth = many) THEN class ⊕

R3: IF (length = 4) THEN class \otimes

R4: ELSE class ⊕

- More formally, we have two algorithms
 - one to generate the list
 - one to generate one rule
- This is *separate-and-conquer* approach





LearnRuleList(D) — learn an ordered list of rules

input: labeled training data D

output: rule list R

- 1. $R \leftarrow \emptyset$ # list is empty
- 2. while $(D \neq \emptyset)$ do # set not empty
- 3. $r \leftarrow \text{LearnRule}(D)$
- 4. append r to the end of R
- 5. $D \leftarrow D \setminus \{x \in D \mid x \text{ is covered by } r\}$
- 6. end
- 7. return R # return the rule list

LearnRule(D) - learn one rule

input: labeled training data D; output: rule r

- 1. $b \leftarrow \text{true}$
- 2. L \leftarrow set of available literals
- 3. while not Homogeneous (D) do
- 4. $l \leftarrow BestLiteral(D, L)$
- 5. $b \leftarrow b \cap 1$
- 6. D \leftarrow D \ $\{x \in D \mid x \text{ is covered by } b\}$
- 7. $\bot \leftarrow \bot \setminus \{l' \in L \mid l' \text{ uses same feature as } l\}$
- 8. end
- 9. $c \leftarrow Label(D)$
- 10. $r \leftarrow \text{if } b \text{ then } class=c$
- 11. return r







With our example we have found the ordered rule list

R1: IF (gills = yes) THEN class \otimes

R2: IF (teeth = many) THEN class ⊕

R3: IF (length = 4) THEN class \otimes

R4: ELSE class ⊕

- Can we form a single rule?
- Can we form an unordered rule list?



In one rule

IF (gills = yes) THEN class
$$\otimes$$

ELSE IF (teeth = many) THEN class \oplus
ELSE IF (length = 4) THEN class \otimes
ELSE class \oplus

The unordered rule list

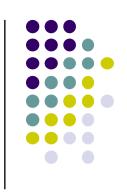
R1: IF (gills=yes) THEN class ⊗

R2': IF (gills=no) ∩ (teeth=many) THEN class ⊕

R3': IF (gills=no) \cap (teeth=few) \cap (length=4) THEN class \otimes

R4': IF (gills=no) ∩ (teeth=few) ∩ (length≠4) THEN class ⊕

Length	Gills	Beak	Teeth	Dolphin
3	yes	yes	many	?



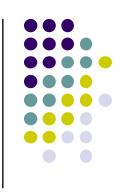
- Many variants do exist
- Instead of being limited to literal, we can consider two literals together, e.g. (p \cap q)
- Relationships with decision tree





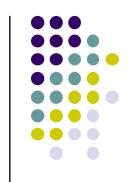
- Ordered Rule Lists
- Unordered Rule Lists





- The main idea is to learn one class at the time
- Instead of min(p, 1-p), we can maximize p (empirical probability of the given class)
- Instead of being limited to literal, we can consider two literals together, e.g. $(p \cap q)$
- Relationships with decision tree





Our training set

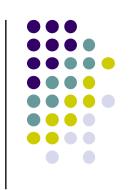
Length	Gills	Beak	Teeth	Dolphin
3	no	yes	many	yes
4	no	yes	many	yes
3	no	yes	few	yes
5	no	yes	many	yes
5	no	yes	few	yes
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no



- We start with the positive class
- We can begin with
 IF (length = 3) THEN class = ⊕
- We then remove the two examples covered by the rule

Length	Gills	Beak	Teeth	Dolphin
4	no	yes	many	yes
5	no	yes	many	yes
5	no	yes	few	yes
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no



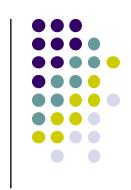


• We continue with?

Length	Gills	Beak	Teeth	Dolphin
4	no	yes	many	yes
5	no	yes	many	yes
5	no	yes	few	yes
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no

• No single literal...



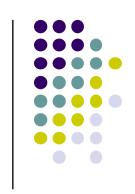


- Thus we continue with
 IF (gills = no) ∩ (length = 5) THEN class = ⊕
- We remove two more examples...

Length	Gills	Beak	Teeth	Dolphin
4	no	yes	many	yes
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no

We finish with
 IF (gills = no) ∩ (teeth = many) THEN class = ⊕



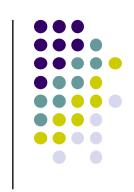


And for the rest?

Length	Gills	Beak	Teeth	Dolphin
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no

- Default rule: ELSE class = ⊗
- Introduces a bias in favor of the second class
- Can we have a better description of the second class?





- Instead of simply adding
 ELSE class = no
- Start the learning with the second class (no) but with the remaining examples

Length	Gills	Beak	Teeth	Dolphin
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no





- First?
- IF (gills = yes) THEN class = ⊗

Length	Gills	Beak	Teeth	Dolphin
5	yes	yes	many	no
4	yes	yes	many	no
5	yes	no	many	no
4	yes	no	many	no
4	no	yes	few	no

Then?
 IF (length = 4) ∩ (teeth = few) THEN class = ⊗





The final set of rules is

R1: IF (length = 3) THEN class = ⊕

R2: IF (gills = no) \cap (length = 5) THEN class = \oplus

R3: IF (gills = no) \cap (teeth = many) THEN class = \oplus

R4: IF (gills = yes) THEN class = \otimes

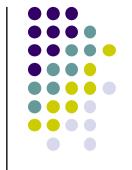
R5: IF (length = 4) \cap (teeth = few) THEN class = \otimes

Length	Gills	Beak	Teeth	Dolphin
3	yes	yes	many	?

Unordered Rule List



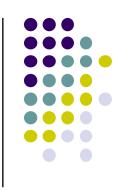
- More formally
- 1. We need to iterate over all possible classes
- 2. For each class, we determine the set of needed rules
- But, in this second step, we remove covered examples and remove the used literal(s).



Unordered Rule List

- **LearnRuleSet**(D) learn an unordered list of rules input: labeled training data D; output: rule set R
 - 1. $R \leftarrow \emptyset$
 - 2. for every class c_i do
 - 3. Di \leftarrow D # select the subset of class ci
 - 4. while Di contains example of class c_i do
 - 5. $r \leftarrow \text{LearnRuleForClass(Di, } c_i)$
 - 6. $R \leftarrow R \cup \{r\}$
 - 7. Di \leftarrow Di $\setminus \{x \in c_i \mid x \text{ is covered by } r\}$
 - 8. end
 - 9. **end**
- 10. return R

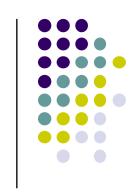




LearnRuleForClass(D, c_i) – learn one rule for a class input: labeled training data D, class c_i ; output: rule r

- 1. $b \leftarrow \text{true}$
- 2. L \leftarrow set of available literals
- 3. while not Homogeneous (D) do
- 4. $l \leftarrow BestLiteral(D, L, c_i)$
- 5. $b \leftarrow b \cap 1$
- 6. D \leftarrow D \ $\{x \in D \mid x \text{ is covered by } b\}$
- 7. $\bot \leftarrow \bot \setminus \{l' \in L \mid l' \text{ uses same feature as } l\}$
- 8. end
- 9. $r \leftarrow if b then class = c_i$
- 10. return r





• The suggested algorithm is myopia...

we have formed

R1: IF (length = 3) THEN class = ⊕

but a better choice could be

R1': IF (gills = no) \cap (teeth = many) THEN class = \oplus

- We can apply a beam search to improve the overall quality of the rules
- And with a new observation?

Length	Gills	Beak	Teeth	Dolphin
4	yes	yes	many	?





Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no ³⁴





R1: IF (outlook = overcast) THEN class = yes

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
rainy	mild	high	true	no ³⁵

Example: Weather Problem

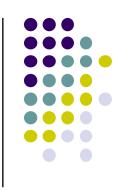
R1: IF (outlook = overcast) THEN class = yes

R2: IF (temperature = hot) THEN class = not

Outlook	Temperature	Humidity	Windy	Play
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
rainy	mild	high	true	no

30





R1: IF (outlook = overcast) THEN class = yes

R2: IF (temperature = hot) THEN class = no

R3: IF (outlook = rainy) \cap (windy = false) THEN class = yes

Outlook	Temperature	Humidity	Windy	Play
rainy	cool	normal	true	no
sunny	mild	high	false	no
sunny	cool	normal	false	yes
sunny	mild	normal	true	yes
rainy	mild	high	true	no





R1: IF (outlook = overcast) THEN class = yes

R2: IF (temperature = hot) THEN class = no

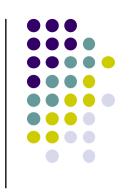
R3: IF (outlook = rainy) \cap (windy = false) THEN class = yes

R4: IF (humidity=normal) ∩ (outlook=sunny) THEN class = yes

R5: ELSE class = no

Outlook	Temperature	Humidity	Windy	Play
rainy	cool	normal	true	no
sunny	mild	high	false	no
rainy	mild	high	true	no





- Generating a set of rules is a simple approach
- But could be time comsuming O(n) = 2ⁿ
- Called also concept learning
- May work well in some contexts, and poorly in others
- Assume that all attribute-value pairs can be used to predict the correct classifiaction
- The decision can be easily explained to the final user