Data mining, machine learning, and uncertainty reasoning

林偉川

Artificial discovery

- Artificial discovery is an instructive illustration of the general way of thinking in machine learning
- Supervised learning where the learner seeks to develop a concept description from examples that have been pre-classified by the teacher
- Unsupervised learning whose task is to generate conceptual taxonomies from non-classified objects

Artificial discovery

- The utility of the taxonomies and categories is obvious: any object that has been recognized as a member of a certain category inherits the general properties of the category
- In machine learning, the search for concept hidden in a set of objects is studied by the discipline known under the name of concept formation
- The further research is to discover not only concepts but also laws defining the relations among them

Concept formation

- Focus on the unsupervised concept learning and divides this field into 2 different subfields: concept discovery ➔deriving concepts from a batch, and incremental concept formation ➔ gradually forms the concepts from a stream of examples
- Concept discovery by conceptual clustering

Concept formation

- Conceptual clustering is a novel form of clustering in which clusters are not just collections of entities processing numerical similarity. Rather, the clusters are understood as groups of objects that together represent a concept
- Conceptual clustering produces not only clusters, but also descriptions of related concepts

AQ algorithm (Supervised learning)

1. Divide all examples into the subsets PE ⊕'s and NE ⊗'s
2. Choose randomly or by design one example from PE and call it the ***seed***
3. Find a set of maximally general rules characterizing the seed. The limit of the generalization is defined by the set NE: a generalized description of the seed is not allowed to cover any object from NE. the set of rules obtained is called the ***star***

AQ algorithm (Supervised learning)

4. According to some preference criterion, select the best rule in the star
5. If this rule, jointly with all previously generated rules, covers all objects from PE, then stop. Otherwise, find another seed among the uncovered examples in PE and go to step 3.

Step 3 is done by a special star generation procedure

It constructs a set of decision rules with different relationship among the individual rules

AQ algorithm (Supervised learning)

- Each of descriptions also covers some of negative examples. These rules are specialized so as to exclude these negative examples
- Multiplying out the current rules by the negations of negative examples and applying absorption law

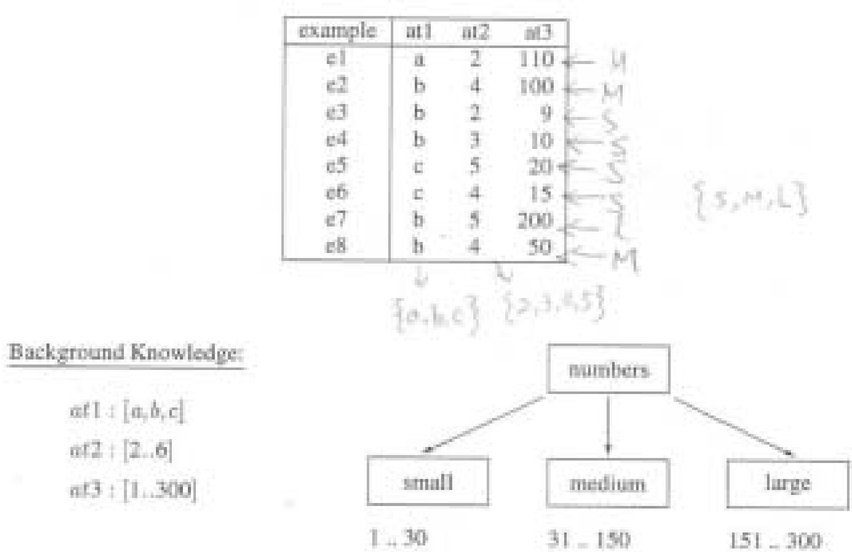
R1’: (at1=x V y) & (at3=r)

R2’: (at1=y) & (at3=r V s)

Concept formation

- The CLUSTER system is anchored in the same seed-and-star philosophy as AQ, and can be considered as its extension to the realm of non-classified examples
- 8 non-classified examples are described by 3 attributes. Attribute at1 is symbolic, attribute at2 acquires integer values, and attribute at3 acquires integer values that can be decomposed into 3 symbolic values
- Background knowledge provides the type and range for each of the attributes and defines the decomposition of at3

Concept formation



example	at1	at2	at3
e1	a	2	110
e2	b	4	100
e3	b	2	9
e4	b	3	10
e5	c	5	20
e6	c	4	15
e7	b	5	200
e8	b	4	50

Concept formation

The learner picks k seeds and treats them as if they represented k different clusters. The CLUSTER algorithm can be summarized as followed:

1. pick k seeds, where k is a user-specified parameter
2. Build k stars, each star being understood as a collection of the most general descriptions of one seed; the limits for the seed generalization are all the other seeds

Concept formation

3. Select from each star one rule so that each rule in the generated rule-set has the minimum logical intersection with the remaining rules, and the logical union of these rules covers the maximum number of instances

Concept formation

4. If there are any uncovered instances, find rules with they have the best ‘fit’. Refine the rules so that together they cover all instances and are all logically disjoint. Instances that belong to an intersection of rules are redistributed so that each is covered by one and only one rule. At this moment, each rule represents a set of examples. From each of these sets, select a new seed

Concept formation

5. Repeat the above procedure for the new seeds and keep repeating the entire procedure as long as each new solution makes an improvement over the previous solution. Repeat for several different values of k (k=2,3,…7) and determine the highest ‘quality’ solution, the quality being determined on the basis of various criteria such as the simplicity of the rules in a clustering and their sparseness (measuring the degree of generalization of each rule over the instances covered by the rule)

Concept formation

- Assume that numerical values have been replaced by symbolic values 'small', 'medium' or 'large'.
- The algorithm roughly perform the following steps (seed k assumed to be 2)
- Choose randomly 2 seeds, e1 and e5. their description are:
  des(e1): (at1=a) & (at2=2) & (at3=medium)
  des(e5): (at1=c) & (at2=5) & (at3=small)
- The initial stars are:
  star(e1): (at1≠c) & (at2≠5) & (at3≠small) (e1e2e8)
  star(e5): (at1≠a) & (at2 ≠2) & (at3 ≠ medium) (e4 e5 e6 e7)

Concept formation

- Each star has 3 one-condition rules and rules from different star intersect. From each star, one rule is selected and modified in such a way that the rules in the rule-set obtained are logically disjoint and their union covers all instances.
- The results is the following solution:
  Cluster1: (at1=a V b) & (at2=2 V 3)
  Instances: e1, e3, e4
  Cluster2: (at1=b V c) & (at2=4 V 5)
  Instances: e2, e5, e6, e7, e8

Concept formation

- The proposed solution for k=2. That is the example can be classified into 2 clusters
- A repetition of the algorithm for higher values of k also does not improve the solution, so that the above is the final result

Crisp concept hierarchies

- The algorithms for concept discovery from fixed sets of non-classified examples tend to prohibitively expensive
- Concept formation algorithms attempt to simulate the development of taxonomies in humans as closely as possible ➔ learn incrementally
- The emphasis of learning incrementally should lay on generating hierarchically ordered concepts

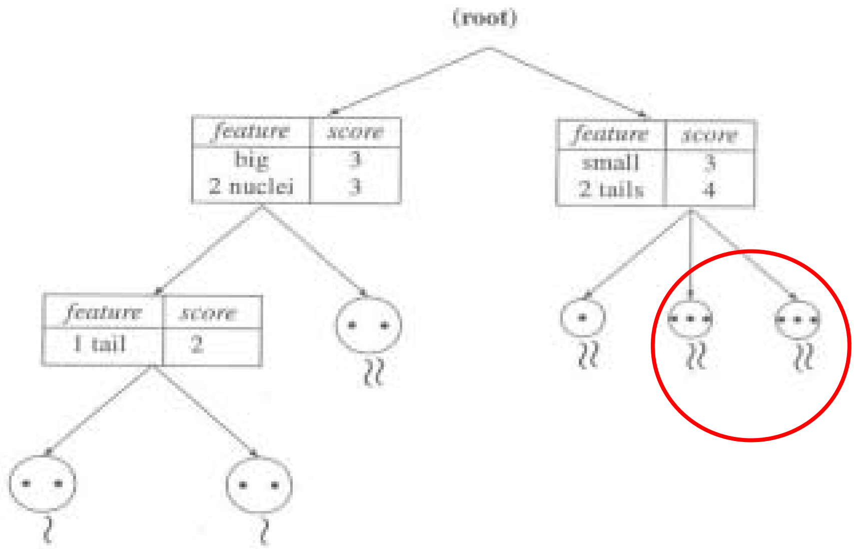
Crisp concept hierarchies

- Most of the systems combine the process of classification and learning: whenever a new example arrives, the system integrates it (classifies) into the current knowledge structure as proposed in UNIMEM system
- It has developed a taxonomy from 6 examples of cells, described by their size, number of nuclei, and number of tail
- When another example arrives (small, 2-tailed, 1 nucleus), it is found to be similar to the triplet in the right-hand branch of the knowledge tree.

Table for the above example

example	size	Nuclei	tail
C1	Big	2	1
C2	Big	2	1
C3	Big	2	2
C4	Small	1	2
C5	Small	2	2
C6	Small	3	2

Hierarchically ordered concept



Hierarchically ordered concept after 1 added

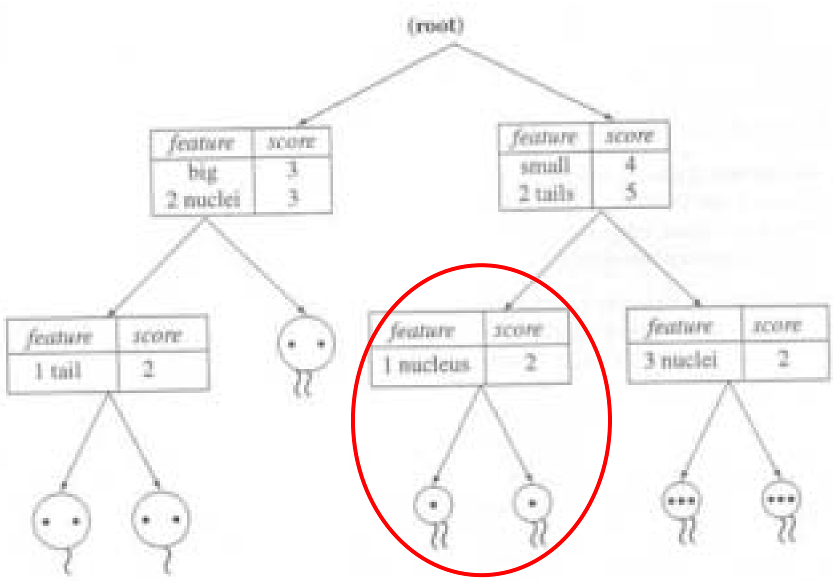


Table for the above example

example	size	Nuclei	tail
C1	Big	2	1
C2	Big	2	1
C3	Big	2	2
C4	Small	1	2
C5	Small	2	2
C6	Small	3	2
C7	Small	1	2

Concept formation algorithm

- Typically conceived as search systems – defined by initial state, termination criterion, search operators, search strategy, and representational issues
- Initial state is given by the description of the first example
- Final state is the knowledge structure after the last example – the system is supposed to learn as long as the examples keep coming

Concept formation algorithm

- The pictures above are self-explanatory. Each node (representing a concept formed by the system) is defined by a set of features such as size(big) [the literal is reduced to the attribute]
- Each feature is accompanied by an integer called score which tells the learner how many times the feature has so far been encountered

Concept formation algorithm

- The score also reflects examples that have been placed in other clusters – ‘2-tail’ example in the right-hand category.
- The score determines the strength of the feature ➔ A small score indicates that the feature is irrelevant and should be discarded. A high score suggests that the feature should be ‘fixed’ in the structure and cannot be deleted

Search operators for concept formation

- In UNIMEM, the concept formation process uses the following search operators:
  1. Store a new instance in the closest node
  2. Create a new node if it improves the value of some general criterion assessing the quality of the conceptual structure created

Search operators for concept formation

- 3. Fix a feature if its score exceeds a predefined threshold
  4. Delete a feature if its score is lower than the scores of the other features
  5. Delete an overly general node (contain only a few features)

Probabilistic conceptual hierarchies

- Other concept-formation systems differ from UNIMEM in the internal representational structure, in the description language (symbolic V.S. numeric attributes), in the search operator, and in the evaluation function guiding the search
- In the COBWEB systems, each node in the hierarchy contains complete information about the probability of the individual attribute values ➔ the probability are estimated simply as relative frequencies

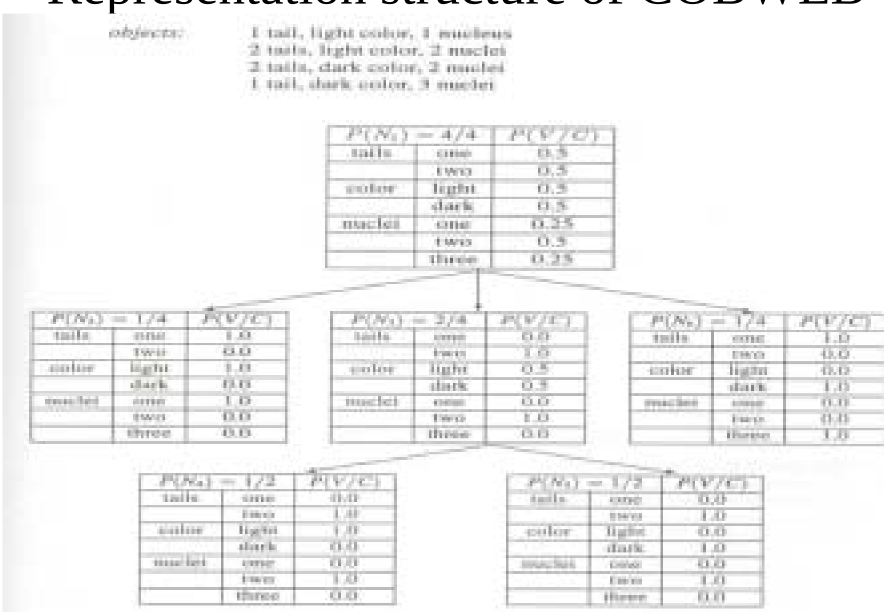
COBWEB hierarchy

- What is peculiar about this representation is that the system does not store crisp description. Rather, each attribute-value pair is accomplished by a number giving the probability that an instance of the concept will possess this particular attribute value
- Each node consists of a heading and 3 column table. The heading contains information about the frequency, $P(N_i)$, with which an example falls into his category. The table contains the relative frequency of the occurrence of any attribute-value pair

Table for the above example

example	Color	Nuclei	tail
C1	Light	1	1
C2	Light	2	2
C3	Dark	2	2
C4	Dark	3	1

Representation structure of COBWEB



COBWEB hierarchy

- COBWEB uses the following search operators:
  1. Incorporate the new example into some of the existing nodes
  2. Create a new node for the example
  3. Merge 2 nodes into one
  4. Split a node into 2 nodes
- Whenever a new example is encountered, the learner must decide which of the operators applies best

COBWEB hierarchy

- Knowing that each operator can change the conceptual hierarchy, the system uses the formula assessing utility of each of the potential new hierarchies:

$$\frac{IG - UG}{N}$$

- the UG (Uniformed Guess) is the expected number of attribute values that can be correctly guessed from an unordered set of objects; the IG (Informed Guess) is the expected number of attribute value that can be correctly guessed, given the conceptual hierarchy; and N is the number of categories that are currently present in the hierarchy

Conclusion of COBWEB hierarchy

- This probabilistic approach is to create a conceptual hierarchy that maximizes the number of attribute values which can be predicated in an unseen example, given the information about the category into which the example falls

Homework

- Apply slice 9 to express the concept structure according the slice 21
- Apply slice 9 to express the concept structure according the slice 32