CS 478 - Tools for Machine Learning and Data Mining

Symbolic Clustering - COBWEB
COBWEB Overview

- Symbolic approach to category formation.
- Uses global quality metrics to determine number of clusters, depth of hierarchy, and category membership of new instances.
- Categories are probabilistic. Instead of category membership being defined as a set of feature values that must be matched by an object, COBWEB represents the probability with which each feature value is present.
- Incremental algorithm. Any time a new instance is presented, COBWEB considers the overall quality of either placing it in an existing category or modifying the hierarchy to accommodate it.
Category Utility

\[ CU = \sum_k \sum_j \sum_i P(F_i = v_{ij}) P(F_i = v_{ij} \mid C_k) P(C_k \mid F_i = v_{ij}) \]

- \( P(F_i = v_{ij} \mid C_k) \) is called the predictability. It is the probability that an object has value \( v_{ij} \) for feature \( F_i \) given that the object belongs to category \( C_k \). The greater this probability, the more likely two objects in a category share the same features.

- \( P(C_k \mid F_i = v_{ij}) \) is called the predictiveness. It is the probability with which an object belongs to category \( C_k \) given that it has value \( v_{ij} \) for feature \( F_i \). The greater this probability, the less likely objects not in the category will have those feature values.

- \( P(F_i = v_{ij}) \) serves as a weight. It ensures that frequently-occurring feature values exert a stronger influence on the evaluation.

CU maximizes the potential for inferring information while maximizing intra-class similarity and inter-class differences.
Tree Representation

- Each node stores:
  1. Its probability of occurrence, \( P(C_k) \) (= num. instances at node / total num. instances)
  2. All possible values of every feature observed in the instances, and for each such value, its predictability.
  3. Predictiveness is computed using Bayes rule (i.e.,
     \( P(A \mid B) = \frac{P(A)P(B \mid A)}{P(B)} \).

- Leaf nodes correspond to observed instances.
- All links are “is-a” links (i.e., no test on feature values).
- Tree is initialized with a single node whose probabilities are those of the first instance.
- For each subsequent instance \( I \), Cobweb(\( \text{Root}, I \)) is invoked.
COBWEB Algorithm

Algorithm Cobweb(Node, Instance)

If Node is a leaf
   Create 2 children, L₁ and L₂ of Node
   Set the probabilities of L₁ to those of Node
   Initialize the probabilities of L₂ to those of Instance
   Add Instance to Node, updating Node's probabilities
Else
   Add Instance to Node, updating Node's probabilities
   For each child C of Node
      Compute CU of taxonomy obtained by placing Instance in C
   Let S₁ be the score of the best categorization C₁
   Let S₂ be the score of the next best categorization C₂
   Let S₃ be the score of placing Instance in a new category
   Let S₄ be the score of merging C₁ and C₂ into one category
   Let S₅ be the score of splitting C₁
   If S₁ is the best score
      Cobweb(C₁, Instance)
   Else if S₃ is the best score
      Initialize new category's probabilities to those of Instance
   Else if S₄ is the best score
      Let Cₘ be the result of merging C₁ and C₂
      Cobweb(Cₘ, Instance)
   Else if S₅ is the best score
      Split C₁
      Cobweb(Node, Instance)
   Else
      Cobweb(C₂, Instance)
http://www-ai.cs.uni-dortmund.de/kdnet/auto?self=$81d91eaae317b2bebb
Discussion

- Nice probabilistic model with no parameters set a priori.
- Only handles nominal features (CLASSIT extends to numerical).
- Sensitive to order of presentation of instances.
- Retains each instance, which may cause problems with noisy data.