Classification Methods

Aijun An
York University, Canada

INTRODUCTION

Generally speaking, classification is the action of assigning an object to a category according to the characteristics of the object. In data mining, classification refers to the task of analyzing a set of pre-classified data objects to learn a model (or a function) that can be used to classify an unseen data object into one of several predefined classes. A data object, referred to as an example, is described by a set of attributes or variables. One of the attributes describes the class that an example belongs to and is thus called the class attribute or class variable. Other attributes are often called independent or predictor attributes (or variables). The set of examples used to learn the classification model is called the training data set. Tasks related to classification include regression, which builds a model from training data to predict numerical values, and clustering, which groups examples to form categories. Classification belongs to the category of supervised learning, distinguished from unsupervised learning. In supervised learning, the training data consists of pairs of input data (typically vectors), and desired outputs, while in unsupervised learning there is no a priori output.

Classification has various applications, such as learning from a patient database to diagnose a disease based on the symptoms of a patient, analyzing credit card transactions to identify fraudulent transactions, automatic recognition of letters or digits based on handwriting samples, and distinguishing highly active compounds from inactive ones based on the structures of compounds for drug discovery.

BACKGROUND

Classification has been studied in statistics and machine learning. In statistics, classification is also referred to as discrimination. Early work on classification focused on discriminant analysis, which constructs a set of discriminant functions, such as linear functions of the predictor variables, based on a set of training examples to discriminate among the groups defined by the class variable. Modern studies explore more flexible classes of models, such as providing an estimate of the join distribution of the features within each class (e.g. Bayesian classification), classifying an example based on distances in the feature space (e.g. the k-nearest neighbor method), and constructing a classification tree that classifies examples based on tests on one or more predictor variables (i.e., classification tree analysis).

In the field of machine learning, attention has more focused on generating classification expressions that are easily understood by humans. The most popular machine learning technique is decision tree learning, which learns the same tree structure as classification trees but uses different criteria during the learning process. The technique was developed in parallel with the classification tree analysis in statistics. Other machine learning techniques include classification rule learning, neural networks, Bayesian classification, instance-based learning, genetic algorithms, the rough set approach and support vector machines. These techniques mimic human reasoning in different aspects to provide insight into the learning process.

The data mining community inherits the classification techniques developed in statistics and machine learning, and applies them to various real world problems. Most statistical and machine learning algorithms are memory-based, in which the whole training data set is loaded into the main memory before learning starts. In data mining, much effort has been spent on scaling up the classification algorithms to deal with large data sets. There is also a new classification technique, called association-based classification, which is based on association rule learning.

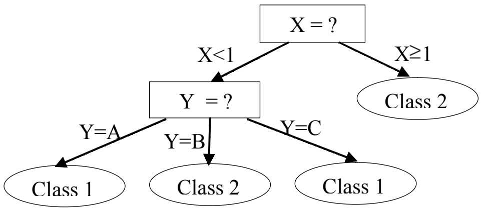
MAIN THRUST

Major classification techniques are described below. The techniques differ in the learning mechanism and in the representation of the learned model.

Decision Tree Learning

Decision tree learning is one of the most popular classification algorithms. It induces a decision tree from data. A decision tree is a tree structured prediction model where each internal node denotes a test on an attribute, each outgoing branch represents an outcome of the test, and each leaf node is labeled with a class or

Figure 1. A decision tree with tests on attributes X and Y



class distribution. A simple decision tree is shown in Figure 1. With a decision tree, an object is classified by following a path from the root to a leaf, taking the edges corresponding to the values of the attributes in the object.

A typical decision tree learning algorithm adopts a top-down recursive divide-and-conquer strategy to construct a decision tree. Starting from a root node representing the whole training data, the data is split into two or more subsets based on the values of an attribute chosen according to a splitting criterion. For each subset a child node is created and the subset is associated with the child. The process is then separately repeated on the data in each of the child nodes, and so on, until a termination criterion is satisfied. Many decision tree learning algorithms exist. They differ mainly in attribute-selection criteria, such as information gain, gain ratio (Quinlan, 1993), gini index (Breiman, Friedman, Olshen, & Stone, 1984), etc., termination criteria and post-pruning strategies. Post-pruning is a technique that removes some branches of the tree after the tree is constructed to prevent the tree from over-fitting the training data. Representative decision tree algorithms include CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993). There are also studies on fast and scalable construction of decision trees. Representative algorithms of such kind include RainForest (Gehrke, Ramakrishnan, & Ganti, 1998) and SPRINT (Shafer, Agrawal, & Mehta., 1996).

Decision Rule Learning

Decision rules are a set of if-then rules. They are the most expressive and human readable representation of classification models (Mitchell, 1997). An example of decision rules is "if X<1 and Y=B, then the example belongs to Class 2". This type of rules is referred to as propositional rules. Rules can be generated by translating a decision tree into a set of rules – one rule for each leaf node in the tree. A second way to generate rules is to learn rules directly from the training data. There is a variety of rule induction algorithms. The algorithms induce rules by searching in a hypothesis space for a hypothesis that best matches the training data. The algorithms differ in the search method (e.g. general-to-specific, specific-to-general, or two-way search), the search heuristics that control the search, and the pruning method used. The most widespread approach to rule induction is *sequential covering*, in which a greedy general-to-specific search is conducted to learn a disjunctive set of conjunctive rules. It is called sequential covering because it sequentially learns a set of rules that together cover the set of positive examples for a class. Algorithms belonging to this category include CN2 (Clark & Boswell, 1991), RIPPER (Cohen, 1995) and ELEM2 (An & Cercone, 1998).

Naive Bayesian Classifier

The naive Bayesian classifier is based on Bayes' theorem. Suppose that there are *m* classes, $C_1, C_2, \ldots, C_m$. The classifier predicts an unseen example X as belonging to the class having the highest posterior probability conditioned on X. In other words, X is assigned to class C_i if and only if

$$P(C_i|X) > P(C_j|X) \text{ for } 1 \leq j \leq m,\ j \neq i.$$

By Bayes' theorem, we have

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}.$$

As $P(X)$ is constant for all classes, only $P(X \mid C_i)P(C_i)$ needs to be maximized. Given a set of training data, $P(C_i)$ can be estimated by counting how often each class occurs in the training data. To reduce the computational expense in estimating $P(X|C_i)$ for all possible Xs, the classifier makes a naïve assumption that the attributes used in describing X are conditionally independent of each other given the class of X. Thus, given the attribute values $(x_1, x_2, \ldots x_n)$ that describe X, we have

$$P(X \mid C_i) = \prod_{j=1}^{n} P(x_j \mid C_i).$$

The probabilities $P(x_1|C_i), P(x_2|C_i), \ldots, P(x_n|C_i)$ can be estimated from the training data.

The naïve Bayesian classifier is simple to use and efficient to learn. It requires only one scan of the training data. Despite the fact that the independence assumption is often violated in practice, naïve Bayes often competes well with more sophisticated classifiers. Recent theoretical analysis has shown why the naive