Knowledge Discovery in Data: Partially Supervised Learning

Overview

Partially supervised learning problem. Given a dataset $D = \{ldotsd_1, \ldots, d_N\}$ where some subset $T \subset D$ has category labels associated with it: $class(d) = c_i$; for $d \in T$, group the data in D by similarity (learn the labels for each $d \notin T$.

Why "partially observable"? The input data for these methods is the same as the input data for supervised learning algorithms (i.e., a dataset, a training set and a collection of class labels). What is different is **how** learning is conducted.

- Typically, **partially observable learning algorithms** are used in situations where the training set is drastically smaller than the entire dataset.
- On each step, the **partially observable learning** algorithms estimate the category for each of the data points $d \notin T$. This data is used to enhance the training set on the next iteration of the learning algorithm.

LU learning. **LU learning methods** are methods of partially observable learning that rely on **labeled** and **unlabeled** examples.

Expectation-Maximization Algorithm

One of the traditional **LU learning techniques** is **EM Algorithm** (Expectation-Maximization Algorithm).

In a nutshell. Given a dataset D and a (small) training T of labeled examples, the EM algorithm works as follows:

Algorithm EM($L, U, V = \{t_1, \ldots t_M\}$); $|L| = N_L$
Let $L = L_1 \cup \ldots L_k$, $d \in L_i \rightarrow class(d) = c_i$;
begin
$\alpha = 0$;
Learn an Naïve Bayesian classifier using L as the training set:

$$Pr^0(t_s|c_i, \hat{\Theta}) = \frac{\lambda + \sum_{j=1}^{N_L} freq(t_s, d_j) \cdot Pr(c_i|d_j)}{\lambda \cdot M + \sum_{t \in V} \sum_{j=1}^{N_L} freq(t, d_j) \cdot Pr(c_i|d_j)};$$

$$\phi_i^0 = Pr^0(c_i|\hat{\Theta}) = \frac{\sum_{j=1}^{N_L} Pr^0(c_i|d_j)}{N_L} = \frac{|L_i|}{N_L};$$

repeat
for each $d_j \in U$ `// Expectation step`
for each $c_i \in \{c_1, \ldots c_k\}$

$$Pr^\alpha(c_i|d_j;\hat{\Theta}) = \frac{Pr^\alpha(c_i|\hat{\Theta}) \cdot Pr^\alpha(d_j|c_i;\hat{\Theta})}{Pr(d_j|\hat{\Theta})} = \frac{Pr^\alpha(c_i|\hat{\Theta}) \cdot \prod_{t \in d_j} freq(t, d_j) \cdot Pr^\alpha(t|c_i;\hat{\Theta})}{\sum_{r=1}^{k} \left(Pr^\alpha(c_r|\hat{\Theta}) \cdot \prod_{t \in d_j} freq(t, d_j) \cdot Pr^\alpha(t|c_r;\hat{\Theta}) \right)};$$

endfor
endfor
`// Maximization step`

$$Pr^{\alpha+1}(t_s|c_i, \hat{\Theta}) = \frac{\lambda + \sum_{j=1}^{N} freq(t_s, d_j) \cdot Pr^\alpha(c_i|d_j)}{\lambda \cdot M + \sum_{t \in V} \sum_{j=1}^{N} freq(t, d_j) \cdot Pr^\alpha(c_i|d_j)};$$

$$\phi_i^{\alpha+1} = Pr^{\alpha+1}(c_i|\hat{\Theta}) = \frac{\sum_{j=1}^{N} Pr^\alpha(c_i|d_j)}{N};$$

$\alpha = \alpha + 1$;
until $Pr^\alpha(t_s|c_i;\hat{\Theta})$ and ϕ_i^α stabilize;
end

Figure 1: E-M Algorithm for Naïve Bayes classifier.

- **Expectation step.** A supervised learning algorithm is run on the current training set to construct a classifier for D.

  This step *fills in* the class variables for each data point $d \in D - T$ that were **missing** from the original data.

- **Maximization step.** The parameters of the supervised classfier are updated based on the current state of the predictions. The parameters are updated to **maximize the likliehood** of the data being constructed given the parameter values.

E-M Algorithm with Naïve Bayes Classifier

Naïve Bayes classifiers work really great within the **E-M framework**. These classfiers *estimate the probabilities* $P(c_i|d_j; T\hat{h}eta)$ *of generating a document (instance)* d_j using the mixture component c_i (i.e., the probability of $class(d_j) = c_i$.

We note that given a fixed document/instance $d_j \in D - T$,

$$\sum_{i=1}^{k} P(c_i|d_j) = 1,$$

i.e., we assume that a document/instance was generated using **some** mixture component.

This relaxes the original idea of a category as a binary function $Pr(class(d_j) = c_i) \in 0, 1$ to $Pr(class(d_j) = c_i) \in [0, 1]$.

The **Expectation step** of the E-M algorithm uses current information about $Pr(c_i|d_j)$ estimates (starting with the training set) to fill in the missing data and/or recompute $Pr(c_i|d_j)$ on next step.

The **Maximization step** recomputes mixture model parameters $Pr(t_s|c_i; T\hat{h}eta)$:

> the probabilities of encountering/*generating* a keyword t_s in a document from category c_i.

These parameters are recomputed to **maximize the likliehood of the collection of documents D being generated from a given mixture model**.

Pseudocode. Figure **??** shows the pseudocode for the **E-M Algorithm for Naïve Bayes classifier**.