Classification & Clustering

Sergej Sizov
Information Retrieval
Summer term 2013

Classification Problem (Categorization)

given:
feature vectors

determine class/topic membership(s) of feature vectors

known classes + labeled training data: supervised learning
unknown classes: unsupervised learning (clustering)

Assessment of Classification Quality

empirical by automatic classification of documents that do not belong to the training data (but in benchmarks class labels of test data are usually known)

For binary classification with regard to class C:

\[ a = \#\text{docs that are classified into } C \text{ and do belong to } C \]
\[ b = \#\text{docs that are classified into } C \text{ but do not belong to } C \]
\[ c = \#\text{docs that are not classified into } C \text{ but do belong to } C \]
\[ d = \#\text{docs that are not classified into } C \text{ and do not belong to } C \]

Accuracy (Genauigkeit) = \( \frac{a + d}{a + b + c + d} \)
Error (Fehler) = 1-accuracy

Precision (Präzision) = \( \frac{a}{a + b} \)
Recall (Ausbeute) = \( \frac{a}{a + c} \)

F1 (harmonic mean of precision and recall) = \( \left( \frac{1}{\text{precision}} + \frac{1}{\text{recall}} \right)^{-1} \)

For manyway classification with regard to classes \( C_1, \ldots, C_k \):

• macro average over \( k \) classes or
• micro average over \( k \) classes

Estimation of Classifier Quality

use benchmark collection of completely labeled documents (e.g., Reuters newswire data from TREC benchmark)

cross-validation (with held-out training data):
• partition training data into \( k \) equally sized (randomized) parts,
• for every possible choice of \( k-1 \) partitions
  • train with \( k-1 \) partitions and apply classifier to \( k \)th partition
  • determine precision, recall, etc.
• compute micro-averaged quality measures

leave-one-out validation/estimation:
variant of cross-validation with two partitions of unequal size:
use \( n-1 \) documents for training and classify the \( n \)th document

Distance-based Classifiers: k-Nearest-Neighbor Method (kNN)

Step 1:
find among the training documents of all classes the \( k \) (e.g. 10-100) most similar documents (e.g., based on cosine similarity): the \( k \) nearest neighbors of \( d \)

Step 2:
Assign \( d \) to class \( C_j \) for which the function value

\[
 f(d, C_j) = \sum_{\tilde{v} \in \text{NN}(d)} \text{sim}(d, \tilde{v}) \times \begin{cases} 1 & \text{if } \tilde{v} \in C_j \\ 0 & \text{otherwise} \end{cases}
\]
is maximized

With binary classification assign \( d \) to class \( C \) if

\[ f(d, C) \text{ is above some threshold } \delta (\delta > 0.5) \]
Hierarchical Clustering: Agglomerative Bottom-up Clustering (HAC)

Principle:
• start with each d_i forming its own singleton cluster c_i
• in each iteration combine the most similar clusters c_i, c_j
  into a new, single cluster
for i=1 to n do c_i := {d_i}; od,
C := {c_i, ..., c_k}; /* set of clusters */
while |C| > 1 do
  determine c_j, c_k ∈ C with maximal inter-cluster similarity;
  C := C - {c_j, c_k} ∪ {c_j ∪ c_k};
end!

Hierarchical vs. Flat Clustering

Hierarchical Clustering:
• detailed and insightful
• hierarchy built in natural manner
• from fairly simple algorithms
• no prevalent algorithm

Flat Clustering:
• data overview & coarse analysis
• level of detail depends on the choice of the number of clusters
• relatively efficient
• K-Means and EM are simple standard algorithms

Cluster Quality Measures (1)

With regard to ground truth:
known class labels L_1, ..., L_n for data points d_1, ..., d_n:
L(d_i) = L_j ∈ {L_1, ..., L_n}
With cluster assignment Γ(d_1), ..., Γ(d_n) ∈ c_1, ..., c_k
cluster c_j has purity
max_{i=1..k} \{ |\{d \in c_j | L(d) = L_j\}| / |c_j| \}
Complete clustering has purity \sum_{j=1..k} purity(c_j) / k

Alternatives:
• Entropy within cluster \sum_{i=1..k} \frac{|c_j \cap L_j|}{|c_j|} \log_2 \left( \frac{|c_j \cap L_j|}{|c_j|} \right)
• MI between cluster and classes
\sum_{c_j \in C} \frac{|c \cap L_j| / n \log_2 \left( \frac{|c \cap L_j| / n}{|c| \cap |L_j| / n} \right) }{ |c| \cap |L_j| / n}
Flat Clustering: Simple Single-Pass Method

given: data records $d_1, \ldots, d_n$

wanted: (up to) $k$ clusters $C := \{c_1, \ldots, c_k\}$

$C := \{\{d_1\}\}$; /* random choice for the first cluster */

for $i := 2$ to $n$ do

determine cluster $c_j \in C$ with the largest value of $\text{sim}(d_i, c_j)$ (e.g. $\text{sim}(d_i, \bar{c}_j)$ with centroid $\bar{c}_j$);

if $\text{sim}(d_i, c_j) \geq$ threshold
then assign $d_i$ to cluster $c_j$
else if $|C| < k$
then $C := C \cup \{d_i\}$; /* create new cluster */
else assign $d_i$ to cluster $c_j$
fi
od

K-Means Method for Flat Clustering (1)

Idea:

• determine $k$ prototype vectors, one for each cluster
• assign each data record to the most similar prototype vector and compute new prototype vector (e.g. by averaging over the vectors assigned to a prototype)
• iterate until clusters are sufficiently stable

randomly choose $k$ prototype vectors $\bar{c}_1, \ldots, \bar{c}_k$
while not yet sufficiently stable do

for $i := 1$ to $n$ do

assign $d_i$ to cluster $c_j$ for which $\text{sim}(d_i, \bar{c}_j)$ is minimal
od;

for $j := 1$ to $k$ do

$\bar{c}_j := \frac{1}{\sum_{d_i \in c_j} \text{sim}(d_i, \bar{c}_j)} \sum_{d_i \in c_j} d_i$
od;

fi

Example for K-Means Clustering

K=2

data records

prototype vectors

after 1st iteration

after 2nd iteration