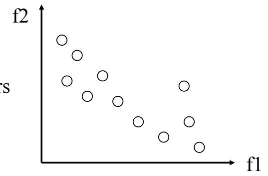
Classification & Clustering

Sergej Sizov
Information Retrieval
Summer term 2013

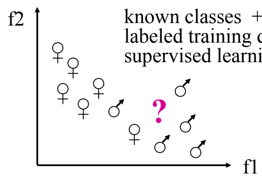
UNIVERSITÄT KOBLENZ · LANDAU

Classification Problem (Categorization)

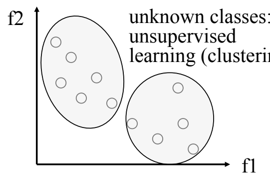
given:
feature vectors



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 = #docs that are classified into C and do belong to C
b = #docs that are classified into C but do not belong to C
c = #docs that are not classified into C but do belong to C
d = #docs that are not classified into C and do not belong to C

Acccuracy (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) = $\left(\frac{1}{precision}+\frac{1}{recall}\right)^{-1}$

For **manyway classification** with regard to classes C_1, ..., 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 $\vec{d}$

Step 2:
Assign $\vec{d}$ to class Cj for which the function value

$$f(\vec{d},C_j) = \sum_{\vec{v}\in kNN(\vec{d})} sim(\vec{d},\vec{v}) * \begin{cases} 1 & if\ \vec{v}\in C_j \\ 0 & otherwise \end{cases}$$

is maximized

With binary classification assign $\vec{d}$ to class C if
$f(\vec{d},C)$ is above some threshold δ (δ >0.5)

Automatic clustering

Clustering: Classification based on Unsupervised Learning

given:
n ***m-dimensional data records*** dj ∈D ⊆ dom(A1) × ... × dom(Am)
with attributes Ai (e.g. term frequency vectors ⊆ N_0 × ... × N_0)
or n ***data points*** with pair-wise ***distances (similarities)*** in a ***metric space***

wanted:
k **clusters** c1, ..., ck and an assignment D → {c1, ..., ck} such that the
average **intra-cluster similarity** $\frac{1}{k}\sum_k \left(\frac{1}{|c_k|} \sum_{\vec{d} \in c_k} sim(\vec{d}, \vec{c}_k) \right)$
is high and
the average **inter-cluster similarity** $\frac{1}{k(k-1)} \sum_{\substack{i,j \\ i \neq j}} sim(\vec{c}_i, \vec{c}_j)$
is low,
where the ***centroid*** $\vec{c}_k$ of ck is: $\vec{c}_k = \frac{1}{|c_k|} \sum_{\vec{d} \in c_k} \vec{d}$

Hierarchical vs. Flat Clustering

Hierarchical Clustering:
- detailed and insightful
- hierarchy built in natural manner from fairly simple algorithms
- relatively expensive
- 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

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_1, ..., c_n}; /* set of clusters */
while |C| > 1 do
  determine c_i, c_j ∈ C with maximal inter-cluster similarity;
  C := C – {c_i, c_j} ∪ {c_i ∪ c_j};
od;

Alternative Similarity Metrics for Clusters

given: similarity on data records - sim: D×D→R oder [0,1]
define: similarity between clusters – sim: $2^D \times 2^D$→R or [0,1]

Alternatives:
- **Centroid method**: sim (c,c‘) = sim(d, d‘) with centroid d of c and centroid d‘ of c‘
- **Single-Link method**: sim(c,c‘) = sim(d, d‘) with d ∈c, d‘∈c‘, such that d and d‘ have the highest similarity
- **Complete-Link method**: sim(c,c‘) = sim(d, d‘) with d ∈c, d‘∈c‘, such that d and d‘ have the lowest similarity
- **Group-Average method**: $\frac{1}{|c| \cdot |c'|} \sum_{d \in c, d' \in c'} sim(d, d')$

For hierarchical clustering the following axiom must hold:
max {sim(c,c‘), sim(c,c“)} ≥ sim(c, c‘∪ c“) for all c, c‘, c“ ∈ 2^D

Cluster Quality Measures (1)

With regard to **ground truth:**
known class labels L_1, …, L_g for data points d_1, …, d_n:
L(d_i) = L_j ∈{L_1, …, L_g}

With cluster assignment Γ(d_1), …, Γ(d_n) ∈ c_1, …, c_k
cluster c_j has **purity** $\max_{\nu=1..g} |\{d \in c_j \mid L(d) = L_\nu\}| / |c_j|$

Complete clustering has purity $\sum_{j=1..k} purity(c_j)/k$

Alternatives:
- **Entropy** within cluster $\sum_{\nu=1..g} \frac{|c_j \cap L_\nu|}{|c_j|} \log_2 \frac{|c_j|}{|c_j \cap L_\nu|}$
- **MI** between cluster and classes

$$\sum_{c \in \{c_j, \bar{c}_j\}, L \in \{L_1, \dots, L_g\}} \frac{|c \cap L|/n}{|c| \cdot |L|/n} \log_2 \frac{|c| \cdot |L|/n}{|c \cap L|/n}$$

Cluster Quality Measures (2)

Without any ground truth:

ratio of intra-cluster to inter-cluster similarities

$$\frac{1}{k}\sum_k \left(\frac{1}{|c_k|} \sum_{\vec{d} \in c_k} sim(\vec{d}, \vec{c}_k) \right) \Big/ \left(\frac{1}{k(k-1)} \sum_{\substack{i,j \\ i \neq j}} sim(\vec{c}_i, \vec{c}_j) \right)$$

or other ***cluster validity measures*** of this kind
(e.g. considering variance of intra- and inter-cluster distances)

Flat Clustering: Simple Single-Pass Method

<u>given:</u> data records d1, ..., dn
<u>wanted:</u> (up to) k clusters C:={c1, ..., ck}

C := {{d1}}; /* random choice for the first cluster */
for i:=2 to n do
  determine cluster cj ∈ C with the largest value of
  sim(di, cj) (e.g. sim(di, $\vec{c}_j$) with centroid $\vec{c}_j$);
   if sim(di, cj) ≥ threshold
   then assign di to cluster cj
   else if |C| < k
     then C := C ∪ {{di}}; /* create new cluster */
     else assign di to cluster cj
     fi
   fi
od

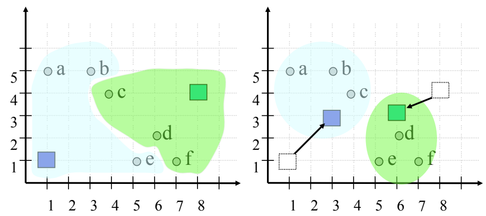
K-Means Method for Flat Clustering (1)

<u>Idea:</u>
- 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 $\vec{c}_1,\ldots,\vec{c}_k$
while not yet sufficiently stable do
  for i:=1 to n do
    assign di to cluster cj for which $sim(\vec{d}_i, \vec{c}_j)$ is minimal
  od;
  for j:=1 to k do $\vec{c}_j := \frac{1}{|c_j|} \sum_{\vec{d} \in c_j} \vec{d}$ od;
od;

Example for K-Means Clustering

K=2 data records prototype vectors



after 1st iteration after 2nd iteration