Alexander Hinneburg,
hinneburg@informatik.uni-halle.de

23.12.2005

Übung zur Vorlesung „Data Mining in Datenbanken“

Übungsblatt 9[1]

Clusteranalyse

Abgabe am 12.1.2006

9.1 Implementieren Sie Laufzeit und Speicherplatz-effizient den Algorithmus für hierarchisches Clustering Complete-Linkage. Die Eingabe sei die Distanzmatrix mit allen paarweisen Distanzen zwischen den Instanzen $D = \{d_{x,y}\}$ mit $x, y = 1 \ldots, n$.

Der generische Algorithmus für bottom-up hierarchisches Clustering initialisiert die Liste der aktuellen Cluster mit den einzelnen Instanzen (Einer-Mengen) als initialen Clustern. Die Distanzen zwischen den initialen Clustern ist durch D gegeben. In $n-1$ Schritten wird eine Hierarchie (binärer Baum) aufgebaut, der die n initialen Cluster als Blätter hat. In jedem Schritt werden die zwei Cluster zusammengefaßt, welche die kleinste Distanz zu einander haben, s.d. die Liste der aktuellen Cluster um eins schrumpft. Nach $n-1$ Schritten bleibt ein Cluster übrig, der alle Instanzen enthält.

Bei Complete Linkage ist die Distanz $d(C_i, C_j)$ zwischen zwei Clustern C_i und C_j die maximale paarweise Distanz zwischen zwei Instanzen aus C_i und C_j:

$$d(C_i, C_j) = \max\{d(x, y) \colon x \in C_i, y \in C_j\} \tag{1}$$

Seien im Schritt l $(1 \leq l \leq n-1)$ C_i und C_j die zwei Cluster mit der kleinsten Distanz zueinander, dann werden sie vereinigt und bilden einen neuen Cluster $C_l = C_i \cup C_j$. Für den neuen Cluster C_l muß die Distanz zu allen noch aktiven Clustern C_k berechnet werden. Dies wird effizient mit dem Lance-Williams Aktualisierungsschema für Complete Linkage gemacht

$$d(C_l, C_k) = \max\{d(C_i, C_k), d(C_j, C_k)\} \tag{2}$$

Dies ist viel effizienter, als wenn man die Distanzen über die Original-Definition (1) neuberechnet. Überlegen Sie sich ein Beispiel dafür, um die Idee zu verstehen. Nach der Aktualisierung der Distanzen zu dem neuen Cluster C_l werden die zusammengefaßten Cluster C_i und C_j aus der Liste der aktuellen Cluster gestrichen (und auch die Distanzen zu diesen Clustern werden nicht mehr gebraucht).

Die Ausgabe soll eine Tabelle mit drei Spalten sein `O(idx1,idx2,linkdist)`. Die beiden Indizes x verweisen auf die zusammengefaßten Cluster und linkdist speichert die Distanz zwischen diesen Clustern. Die initialen Cluster sollen nicht in der Ausgabe auftauchen, werden aber implizit mitgedacht indem die Indizes auf sie verweisen.

[1] Achten Sie auch auf die Form Ihrer Lösungen, z.B. daß jedes Bild eine Bildunterschrift hat. Bitte geben Sie keine losen Blätter ab. Um die Übungen in angemessener Zeit zu diskutieren, bereiten Sie neben Ihrer Lösung ein kleine Präsentation vor, mit der Sie dann Ihre Ergebnisse zeigen können.

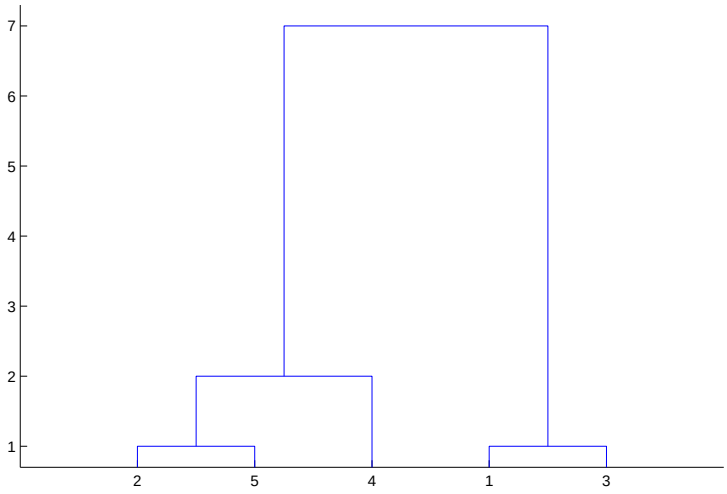
Ein Beispiel, die Instanzen sind durch ein-dimensionale Vektoren (also Zahlen) beschrieben $X = \{4, 9, 3, 10, 8\}$. Die Distanzmatrix D ist:

$$d = \begin{pmatrix} 0 & 5 & 1 & 6 & 4 \\ 5 & 0 & 6 & 1 & 1 \\ 1 & 6 & 0 & 7 & 5 \\ 6 & 1 & 7 & 0 & 2 \\ 4 & 1 & 5 & 2 & 0 \end{pmatrix}$$

Speicherplatz-effizienter ist es nur die obere Dreiecksmatrix als Vektor zu speichern: $D = (5, 1, 6, 4, 6, 1, 1, 7, 5, 2)$. Die Ausgabe von Complete-Linkage ist dann:

idx1	idx2	LinkDist
2	5	1
1	3	1
4	6	2
7	8	7

Als Drendrogramm sieht das Ganze so aus, die X-Achse gibt den Index in der Datenmenge X an.



Hinweise und Links zur weiterer Literatur finden Sie auf der Vorlesungswebseite. Wenden Sie Ihre Implementierung auf die transponierten Gen-Expressionsdaten an, wobei die Gene die Instanzen sind. Nutzen Sie nur den Teil für die Trainingsmenge, s.d. Sie 7071 Vektoren der Dimension 38 erhalten. Berechnen Sie die Distanz zwischen den Vektoren mittels Euklidischer Distanz. Messen Sie die Laufzeit Ihrer Implementierung für diese Aufgabe.