UNIVERSITÄT BIELEFELD
Faculty of Technology

Introduction to Data Mining
06 - Clustering II

Benjamin Paaßen

WS 2023/2024, Bielefeld University

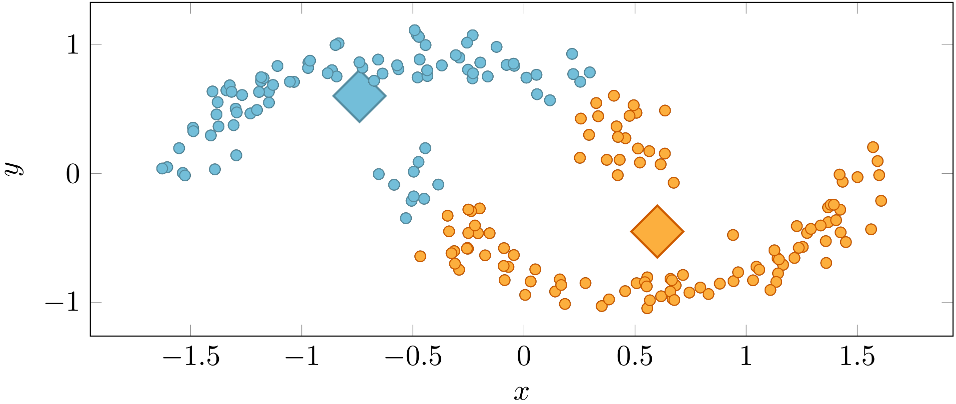
1. Agglomerative Clustering
2. Relational K-Means
3. Practical Story: Word Clustering

Agglomerative Clustering

Motivation: Non-spherical clusters

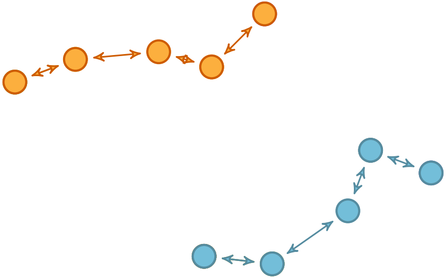
UNIVERSITÄT BIELEFELD

Faculty of Technology



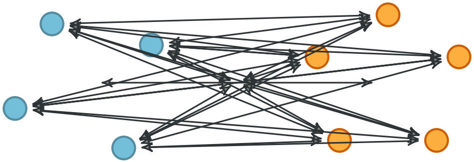
Setup

- Key idea: Each point is its own cluster; then merge closest clusters until only K clusters are left



What does “closeness” mean?

UNIVERSITÄT
BIELEFELD
Faculty of Technology



- single linkage: distance between closest points; flexible cluster shapes, sensitive to 'bridges'
- complete linkage: distance between farthest points; spherical clusters
- centroid linkage: (squared) distance between means
- average linkage: all pairwise (squared) distances
- Ward's method: (squared) distance to shared mean (i.e. variance)

Algorithm

function AgglomerativeClustering(cluster distance function d, desired number of clusters K)

  Initialize $\mathcal{C}_i = \{i\}$ for all $i \in \{1, \ldots, N\}$.

  for $N - K$ repeats **do**

    Find k, l that minimize $d(\mathcal{C}_k, \mathcal{C}_l)$.

    Set $\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \mathcal{C}_l$.

    Remove $\mathcal{C}_l$ and update the cluster numbering.

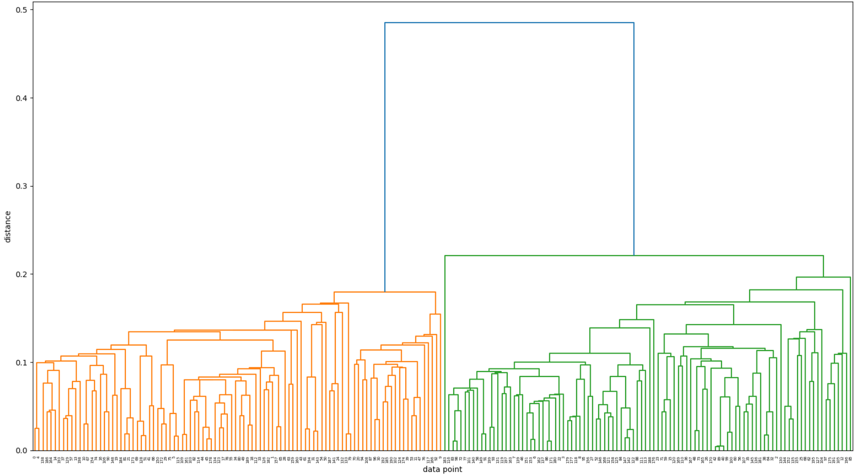
  end for

  return $\mathcal{C}_1, \ldots, \mathcal{C}_K$.

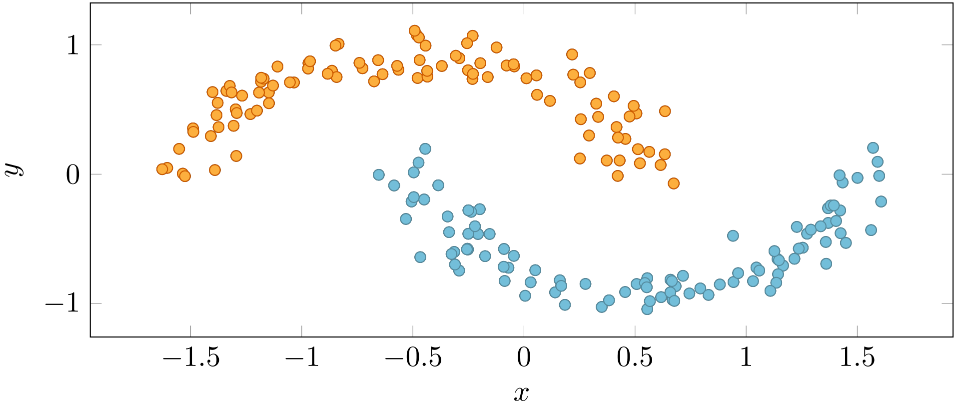
end function

Single Linkage Dendrogram

hnology



Single Linkage Result



Runtime complexity

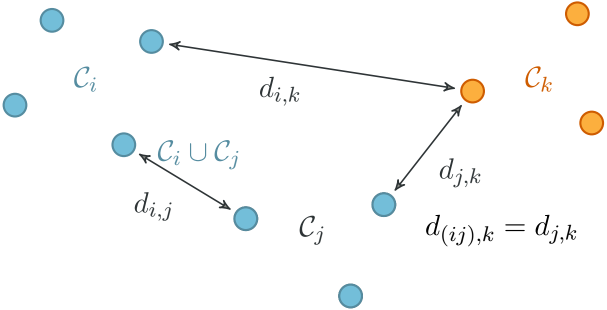
UNIVERSITÄT BIELEFELD
Faculty of Technology

- $N - K$ iterations (one cluster vanishes per iteration)
- computing $d(\mathcal{C}_k, \mathcal{C}_l)$ takes $\mathcal{O}(N^2)$
- we need to evaluate $\mathcal{O}(N^2)$ cluster-to-cluster distances

$\Rightarrow$ naive implementation can be as bad as $\mathcal{O}(N^5)$:(

Example: Single-linkage clustering

- Key idea: Speed up computation by computing cluster-to-cluster purely based on prior cluster-to-cluster distances (recursively)



$$d_{(ij),k} = \min\{d_{i,k}, d_{j,k}\}$$
$$= \frac{1}{2} \cdot (d_{i,k} + d_{j,k}) - \frac{1}{2} \cdot |d_{i,k} - d_{j,k}|$$

Recursive formula

Lance and Williams (1966)

Let $d_{i,j}$, $d_{i,k}$, and $d_{j,k}$ be the pairwise distances between clusters $\mathcal{C}_i$, $\mathcal{C}_j$, and $\mathcal{C}_k$.
Then, the distance $d_{(ij),k}$ between$\mathcal{C}_i \cup \mathcal{C}_j$ and $\mathcal{C}_k$ is

$$d_{(ij),k} = \alpha_1 \cdot d_{i,k} + \alpha_2 \cdot d_{j,k} + \beta \cdot d_{i,j} + \gamma \cdot |d_{i,k} - d_{j,k}|$$

for suitable parameters $\alpha_1, \alpha_2, \beta, \gamma \in \mathbb{R}$.

variant	α_1	α_2	β	γ
single linkage	$\frac{1}{2}$	$\frac{1}{2}$	0	$-\frac{1}{2}$
complete linkage	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$
centroid linkage	$\frac{N_i}{N_i+N_j}$	$\frac{N_j}{N_i+N_j}$	$-\frac{N_i \cdot N_j}{(N_i+N_j)^2}$	0
Ward's method	$\frac{N_i+N_k}{N_i+N_j+N_k}$	$\frac{N_j+N_k}{N_i+N_j+N_k}$	$-\frac{N_k}{N_i+N_j+N_k}$	0

Comments

- **note:** aggl. clustering requires no vectors, only distances!
- Dendrogram can be used to find no. of clusters
- With clever data structures, aggl. clustering can be improved to $\mathcal{O}(N^2 \cdot \log(N))$; in practice, even faster (Bouguettaya et al. 2015)

Relational K-Means

Motivation: Non-vectorial data

- Task: write a sorting program ⇒ How to cluster the answers?

```python
def bsort(A):
    for i in range(1,len(A)):
        for j in range(1,len(A)):
            if(A[j-1] > A[j]):
                tmp = A[j]
                A[j] = A[j-1]
                A[j-1] = tmp
    return A
```

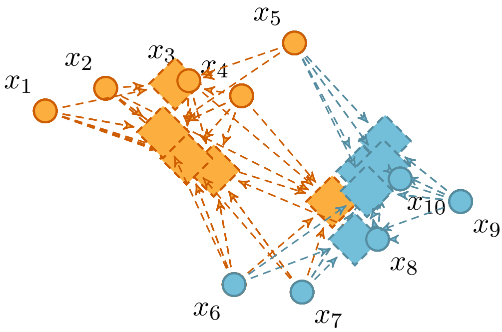
```python
def isort2(A):
    for i in range(1,len(A)):
        tmp = A[i]
        j = i-1
        while(j >= 0 and tmp < A[j]):
            A[j+1] = A[j]
            j -= 1
        A[j+1] = tmp
    return A
```

- We may not have a vector representation but pairwise distances (like tree edit distance)

Setup

- Basic idea: Same as K-means, but using only distances
- Trick 1: represent each prototype μ_k only via coefficients $\alpha_{k,1}, \ldots, \alpha_{k,N}$, such that $\mu_k = \sum_{i=1}^N \alpha_{k,i} \cdot x_i$
- Trick 2: compute $d(x_i, \mu_k)$ only based on distances between data points and $\alpha_{k,1}, \ldots, \alpha_{k,N}$

Visualization



mean step / assignment step

i	$\alpha_{1,i}$	$\alpha_{2,i}$
1	0.25 / 0.14	0
2	0.25 / 0.14	0
3	0.25 / 0.14	0
4	0.25 / 0.14	0
5	0 / 0.05	0.125 / 0.13
6	0.25 / 0.14	0 / 0.2
7	0.15 / 0.14	0 / 0.2
8	0.15 / 0.14	0.25 / 0.2
9	0 / 0.1	0.25 / 0.3
10	0 / 0.1	0.25 / 0.3

Relational distance formula

UNIVERSITÄT BIELEFELD
Faculty of Technology

- Can we compute a distance between a (weighted) mean and a point purely based on point-to-point distances?

Relational Distance Formula (Hammer and Hasenfuss 2010)

Let $x_1, \ldots, x_N \in \mathcal{X}$. Let $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$ such that $\sum_{i=1}^N \alpha_i = 1$. Finally, let $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ be a symmetric and self-equal function. Then, the point $\mu = \sum_{i=1}^N \alpha_i \cdot x_i$ is well-defined and for any $x \in \mathcal{X}$, it holds:

$$d(\mu, x)^2 = \sum_{i=1}^N \alpha_i \cdot d(x_i, x)^2 - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \cdot \alpha_j \cdot d(x_i, x_j)^2 \tag{1}$$

Proof Sketch (Part 1)

UNIVERSITÄT
BIELEFELD
Faculty of Technology

- For full details, refer to Paaßen (2019, chapter 2.1)
- Starting point: for any symmetric and self-equal d, we can construct a bi-linear $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, such that $d(x, y)^2 = s(x, x) - 2s(x, y) + s(y, y)$.
- Starting from the right-hand-side, we obtain:

$$
\begin{aligned}
&\sum_{i=1}^{N} \alpha_i \cdot d(x_i, x)^2 - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \cdot \alpha_j \cdot d(x_i, x_j)^2 \\
=& \sum_{i=1}^{N} \alpha_i \cdot \Big(s(x_i, x_i) - 2s(x_i, x) + s(x, x) \Big) \\
&- \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \cdot \alpha_j \cdot \Big(s(x_i, x_i) - 2s(x_i, x_j) + s(x_j, x_j) \Big)
\end{aligned}
$$

Proof Sketch (Part 2)

$$= \sum_{i=1}^{N} \alpha_i \cdot s(x_i, x_i) - 2\sum_{i=1}^{N} \alpha_i \cdot s(x_i, x) + s(x, x) \cdot \Big(\sum_{i=1}^{N} \alpha_i\Big)$$

$$-\frac{1}{2}\sum_{i=1}^{N} \alpha_i \cdot s(x_i, x_i) \cdot \Big(\sum_{j=1}^{N} \alpha_j\Big) + \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \cdot \alpha_j \cdot s(x_i, x_j) - \frac{1}{2}\Big(\sum_{i=1}^{N} \alpha_i\Big) \cdot \sum_{j=1}^{N} \alpha_j \cdot s(x_j, x_j)$$

$$= -\,2\sum_{i=1}^{N} \alpha_i \cdot s(x_i, x) + s(x, x) + \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \cdot \alpha_j \cdot s(x_i, x_j)$$

$$= -\,2s\Big(\sum_{i=1}^{N} \alpha_i \cdot x_i, x\Big) + s(x, x) + s\Big(\sum_{i=1}^{N} \alpha_i \cdot x_i, \sum_{j=1}^{N} \alpha_j \cdot x_j\Big)$$

$$= -\,2s(\mu, x) + s(x, x) + s(\mu, \mu) = d(\mu, x)^2 \qquad \square$$

Relational K-Means procedure

UNIVERSITÄT BIELEFELD
Faculty of Technology

function Relational KMeans(matrix of squared distances $\boldsymbol{D}^2$ with N rows and columns, desired number of clusters K)

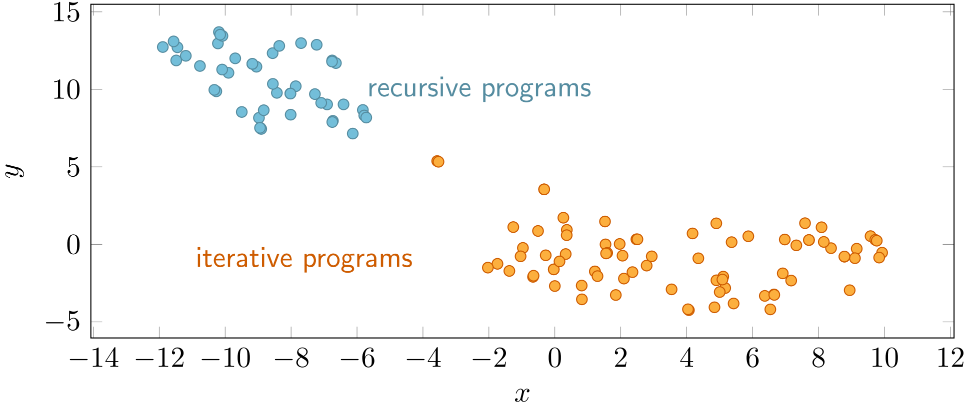
   Randomly initialize $\boldsymbol{A}$ as $N \times K$ matrix of positive numbers.
   Divide columns of $\boldsymbol{A}$ by column sums.
   while $\boldsymbol{A}$ still changes **do**
      Compute $d(\mu_k, x_i)^2$ using Equation (1) for all i and k.
      Compute $z_i \leftarrow \arg\min_k d(\mu_k, x_i)^2$.
      Set $\alpha_{k,i} = 1$ if $z_i = k$ and 0, otherwise.
      Divide columns of $\boldsymbol{A}$ by column sums.
   end while
   return $\boldsymbol{A}$.
end function

Comments

- Almost the same as K-means, just with coefficient representation for prototypes and relational distance formula (1)
- Yields exactly (!) the same results as regular K-means for Euclidean distance

Visualization: Programming Clustering

- distance function: adapted tree edit distance (Paaßen 2019)
- t-SNE embedding; color indicates relational K-means with $K = 2$



Practical Story: Word Clustering

Setup

- Without thinking too hard, what is the first word you associate with **Bielefeld**

lots of nature

trees

teutoburg forest

grey sky

rain

cloudy

olderdissen animal park

sparrenburg

pudding

oetker

frozen pizza

bielefeld conspiracy

does not exist

non-existance

- How to cluster these answers – without manual labor?

Step 1: Define a distance function

Try to think of a good distance function for words

- bag-of-words?
- edit distance/Levenshtein distance?
- normalized compression distance?

$\Rightarrow$ cosine distance on **word embeddings** by `BAAI/bge-large-en-v1.5` language model from huggingface

Step 2: Clustering

- Select K via silhouette score and BIC
- Apply relational K-Means (actually: just K-means on normalized embeddings)

$\Rightarrow$ Problem: very large number of clusters :(

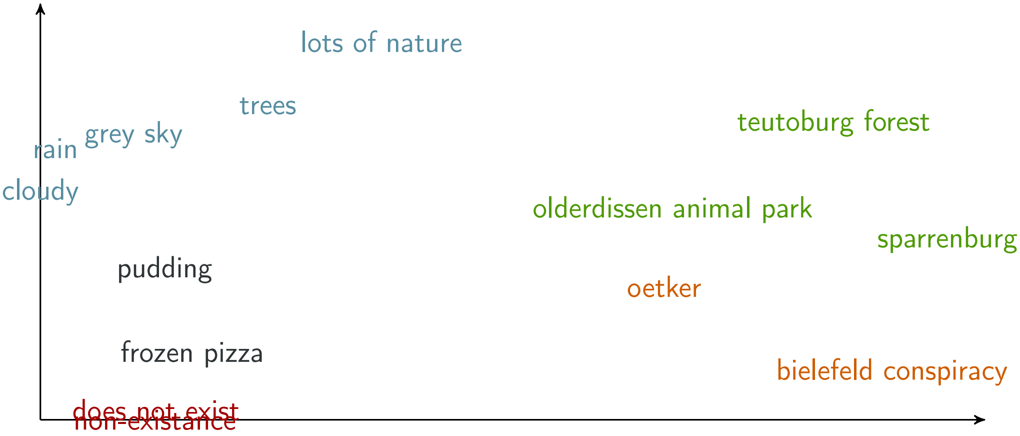
Step 3: Agglomerative clustering

- merge all clusters above similarity threshold via aggl. clustering
- re-compute cluster means
- interpret clusters via words closest to center

Visualization

UNIVERSITÄT BIELEFELD
Faculty of Technology

- PCA on word embeddings, color indicates K-means clustering with $K = 5$



Summary

UNIVERSITÄT BIELEFELD
Faculty of Technology

- Via aggl. clustering and relational K-means, clustering is possible even on purely distance-based data
- example: program clustering, word clustering
- challenge 1: efficiency (at least $\mathcal{O}(N^2)$)
- challenge 2: interpretability (e.g. via closest-to-mean, largest coefficient; Hofmann et al. 2014)

Literature I

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bouguettaya, Athman et al. (2015). "Efficient agglomerative hierarchical clustering". In: **Expert Systems with Applications** 42.5, pp. 2785–2797. doi: 10.1016/j.eswa.2014.09.054.

Cormack, R. M. (1971). "A Review of Classification". In: **Journal of the Royal Statistical Society: Series A (General)** 134.3, pp. 321–353. doi: 10.2307/2344237.

Hammer, Barbara and Alexander Hasenfuss (2010). "Topographic Mapping of Large Dissimilarity Data Sets". In: **Neural Computation** 22.9, pp. 2229–2284. doi: 10.1162/NECO_a_00012.

Hofmann, Daniela et al. (2014). "Learning interpretable kernelized prototype-based models". In: **Neurocomputing** 141, pp. 84–96. doi: 10.1016/j.neucom.2014.03.003.

Lance, GN and WT Williams (1966). "A generalized sorting strategy for computer classifications". In: **Nature** 212.5058, pp. 218–218. doi: 10.1038/212218a0.

Literature II

Paaßen, Benjamin (2019). "Metric Learning for Structured Data". Dissertation. Bielefeld University. doi: 10.4119/unibi/2935545.