Complements on Unsupervised Learning

Badih Ghattas

Outline

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Supervised and unsupervised learning

- Supervised and unsupervised learning share a lot of features within the algorithms they use to estimate the underlying models.
- In supervised learning a target variable Y is available (Regression = $Y \in \mathbb{R}$, Classification = $Y \in 1, ..., J$).
- Unsupervised learning concerns Clustering and Density estimation, the second often resolves the first.

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Density or distribution estimators

- *Parametric Estimators*, mixtures of densities...
  Typically one assumes a specific form for the density, leading to estimating parameters using for instance ML; or Expectation Maximization.

$$\sum_{k=1}^{K} \alpha_k f_k(x)$$

- *Non parametric estimators*: Histogram, KDE

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathsf{K}\left(\frac{x - x_i}{h}\right)$$

- *Ensemble methods*: Averaging histograms, boosting, stacking....

Density or distribution estimators

- *Parametric Estimators*, mixtures of densities...
  Typically one assumes a specific form for the density, leading to estimating parameters using for instance ML; or Expectation Maximization.

$$\sum_{k=1}^{K} \alpha_k f_k(x)$$

- *Non parametric estimators*: Histogram, KDE

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathsf{K}\left(\frac{x - x_i}{h}\right)$$

- *Ensemble methods*: Averaging histograms, boosting, stacking....

Density or distribution estimators

- *Parametric Estimators*, mixtures of densities...
  Typically one assumes a specific form for the density, leading to estimating parameters using for instance ML; or Expectation Maximization.

$$\sum_{k=1}^{K} \alpha_k f_k(x)$$

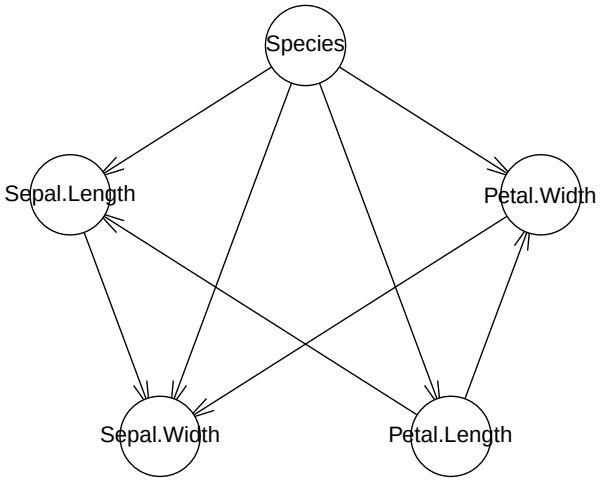
- *Non parametric estimators*: Histogram, KDE

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathsf{K}\left(\frac{x - x_i}{h}\right)$$

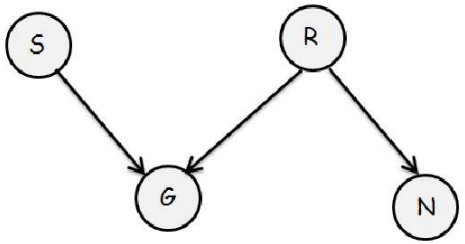
- *Ensemble methods*: Averaging histograms, boosting, stacking....

What about Bayesian Networks, Iris example ...

Objective: Model Joint distribution of a set of variables.



What about Bayesian Networks ...



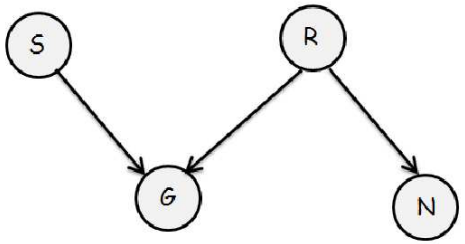
S	$\bar{S}$
0.2	0.8

R	$\bar{R}$
0.4	0.6

	S		$\bar{S}$	
	R	$\bar{R}$	R	$\bar{R}$
G	1	1	1	0
$\bar{G}$	0	0	0	1

	R	$\bar{R}$
N	1	0
$\bar{N}$	0	1

What about Bayesian Networks ...



S	$\bar{S}$
0.2	0.8

R	$\bar{R}$
0.4	0.6

	S		$\bar{S}$	
	R	$\bar{R}$	R	$\bar{R}$
G	1	1	1	0
$\bar{G}$	0	0	0	1

	R	$\bar{R}$
N	1	0
$\bar{N}$	0	1

Chain Rule of probabilities

$$
\begin{aligned}
P(X_1, X_2, ..., X_n) &= P(X_1|X_2, ..., X_n)P(X_2, ..., X_n) \\
P(X_2, ..., X_n) &= P(X_2|X_3, ..., X_n)P(X_3, ..., X_n) \\
\vdots &= \vdots \\
P(X_{n-1}, X_n) &= P(X_{n-1}|X_n)P(X_n) \\
P(X_n) &= P(X_n)
\end{aligned}
$$

$$\Rightarrow \quad P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n-1} P(X_i|X_{i+1}, ..., X_n)P(X_n)$$

This is the general factorization which depends on the ordering of variables.

Definition

A Bayesian Network (BN) $\mathcal{B} = (\mathcal{G}, \theta)$ is a directed acyclic graph $\mathcal{G} = (X, E)$ where the nodes correspond to the set of variables $X = (X_1, \cdots, X_n)$ and $\theta = \{\theta_1, \cdots, \theta_n\}$ where $\theta_i = \mathbb{P}(X_i \mid X_{P_a(X_i)})$ is the conditional probability of the node i over its parents $P_a(X_i)$ within $\mathcal{G}$. Thus the BN $\mathcal{B}$ represents a specific decomposition of the joint distribution of the data:

$$P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i | Pa(X_i)).$$

The underlying assumption of this decomposition is that each variable X_i is independent of its non-descendants conditionally on his parents in the graph $\mathcal{G}$. The graph encodes assertions of conditional independence.

BN, How to fit, how to use ?

- Learn parameters (Conditional distributions) from data
- Learn Structure and parameters from data
- Inference....

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Clustering methods

Clustering aims to partition the data. Some supervised methods are based also on partitioning the space of explanatory variables (CART, SVM, LDA). Partitioning may be:

- *Hierarchical*: find successive groups splitting or joining previously established groups ("bottom-up", "top-down")
- *Partitional- non hierarchical*: k-means.
- *Density based methods*: Mixture models, DBSCAN.

Clustering methods

- Hierarchical or non hierarchical (kmeans, mixture models, density based models).
- Works often with dissimilarity matrix.
- Often needs the number k of clusters to be fixed in advance.
- There exist some heuristics to estimate k (pseudo T, pseudo F).
- We need also to estimate the quality of a partition and to compare partitions.

Often we need to deal with ...

- qualitative datasets, or mixed
- functional data

The most widely used method: The k-means algorithm.

k-means

- Choose arbitrary centers,
- Repeat the two following steps until the centers do not change.
  - assign each observation to the closest center,
  - compute the new center of each class using the observations assigned to that class,
- Repeat the previous steps 10 times and keep the partition with the minimum within-cluster sum of squares given by,

$$\sum_{i=1}^{n}\sum_{j=1}^{k}\|\mathbf{X}_i - c_j\|^2 \mathbf{1}_{\{\mathbf{X}_i \in G_j\}}$$

where G_j is the j–th group and c_j is the corresponding center.

k-modes/k-medians

Extensions of the well-known k-means algorithm [22].

- The k-modes algorithm [16] for categorical data that extends the k-means algorithm. It seeks to partition the observations into k groups such that the distance from the observations to the cluster modes is minimized. Simple-matching distance to assess the dissimilarity between pairs of observations is often used (Number of disagreements between pairs/Total number of pairs).
  See the *klaR* package [31] from **R**.
- The k-medians approach is recommended in [20] for addressing ordinal data. It is similar to the k-means algorithm except that it uses medians rather than means as centers for the clusters. Manhattan distance ($\sum_i |x_i - y_i|$) may be used to assess the dissimilarity between the observations and the cluster medians.
  See the *flexclust* package [20] from **R**.

Hierarchical Clustering

Hierarchical methods [25] aim to construct a dendrogram, given a distance d, and they can be ascendant ("bottom-top") or descendant ("top-down"). In ascendant hierarchical clustering, the hierarchy is constructed by iterative aggregation of pairs of nearest clusters. The distance between two clusters may be defined in several ways, the most popular being simple, average or complete linkage. For ordinal data one may use the Manhattan distance as the dissimilarity measure between the observations, or the mutual information for nominal data.
The algorithm is implemented in the *hclust* function from **R**.

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Data simulation models

$k = 3$, number of groups

$p = 9$, number of variables

$n \in \{100, 300, 500, 1000\}$ for the sample sizes

$m_j = m = 5$ number of levels for $j \in \{1, ..., p\}$

$\left\lfloor \frac{n}{k} \right\rfloor$, number of observations per group

LC simulation, Model M1

Three clusters are defined, each characterized by a high frequency of one level: Level 1,3, and 5 are the most frequent for cluster 1, 2 and 3 resp. The other levels are uniformly distributed. For example, The distribution for each variable in cluster one is given by:

$$\begin{aligned} P(X_{.j} = 1) &= q \\ P(X_{.j} = l) &= \frac{1-q}{m-1} \qquad \text{for} \qquad l \neq 1 \end{aligned}$$

The same is done for the other clusters. We fix $q = 0.8$.

IRT-based simulation, M2

Item Response Theory (IRT) approach. We use the generalized partial credit model (GPCM):

$$p_{jx}(\theta) = P(X_{ij} = x|\theta) = \frac{\exp\left(\sum_{k=0}^{x} \alpha_j(\theta_i - \beta_{jk})\right)}{\sum_{r=0}^{m_j} \exp\left(\sum_{k=0}^{r} \alpha_j(\theta_i - \beta_{jk})\right)}$$

where

θ_i is the individual parameter
β_{jk} is the threshold parameter for the kth level of item j
α_j is the discrimination parameter of the jth item

For each class c ($c \in \{1, 2, 3\}$) we simulate a vector of θ_i values using $N(\mu_c, \sigma^2)$, $\mu = (-3, 0, 3)$ and $\sigma^2 = 0.1$.

For $j \in \{1, ..., p\}$, we fix $\alpha_j = 1.2$ and $\beta_j = (-1, -\frac{1}{3}, \frac{1}{3}, 1)$.

A tree model, M3

$p = 3$, $k = 4$. Each variable X_j, $j \in \{1, ..., p\}$, has $m = 6$ levels. Each level is coded as an integer, and we distinguish odd and even levels. The partition used for the simulation is shown in figure 1. Clusters are defined as follows:

- $C1$: x_1 and x_2 have odd levels, and x_3 is arbitrary
- $C2$: x_1 has odd levels, x_2 has even levels, and x_3 is arbitrary
- $C3$: x_1 has even levels, x_3 has odd levels, and x_2 is arbitrary
- $C4$: x_1 and x_3 have even levels, and x_2 is arbitrary

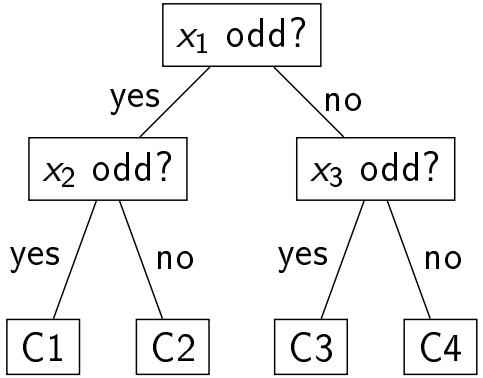


Figure: Tree structure used for data simulation model M3.

Another tree model, M4

$p = 3$, $k = 4$. Here, each variable $X_{.j}$, $j \in \{1, ..., p\}$, has $m = 4$ levels. The only difference with M3is that variable levels are not uniformly distributed in each cluster. Here, we consider a parameter p_0 that controls the non-uniformity of the distribution of levels, for example $p_0 = 0.8$. Clusters are defined as follows:

- $C1$: x_1 and x_2 have odd levels with $P(x_1 = 1) = P(x_2 = 1) = p_0$, and x_3 is arbitrary
- $C2$: x_1 has odd levels, x_2 has even levels with $P(x_1 = 1) = P(x_2 = 2) = p_0$, and x_3 is arbitrary
- $C3$: x_1 has even levels, x_3 has odd levels with $P(x_1 = 2) = P(x_3 = 1) = p_0$, and x_2 is arbitrary
- $C4$: x_1 and x_3 have even levels with $P(x_1 = 2) = P(x_3 = 2) = p_0$, and x_2 is arbitrary

Category Utility

Consider a partition, $C = C_k$, $k = 1, ..., K$, found by a clustering algorithm based on given attributes A_i, $i = 1, ..., p$. The attributes are assumed nominal so that each A_j has a set of attribute values or categories, V_{jl}. The category utility function scores partition C against the set of variables according to formula:

$$CU(C) = \frac{1}{K} \sum_{k=1}^{K} P(C_k) \left[\sum_j \sum_l P(A_j = V_{jl} | C_k)^2 - \sum_j \sum_l P(A_j = V_{jl})^2 \right]$$

Probability matching is random decision strategy using the empirical distribution, which is different from the bayesian majority vote prediction. The probability of winning when distribution is given by p_j, is $\sum_j p_j^2$

The matching error

We assess the performance of the different methods using the matching error. Let $y_1, ..., y_n$ be the class labels of each observation, and let $\hat{y}_1, ..., \hat{y}_n$ be the labels assigned to the n observations by a clustering algorithm. We denote by Σ the set of all possible permutations of the set of labels. The misclassification error rate, also called the "matching error" is defined as follows:

$$MCE = \min_{\sigma \in \Sigma} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \neq \sigma(\hat{y}_i)\}}$$

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

DBSCAN

Density-based spatial clustering of applications with noise [9] is a density-based, non-hierarchical algorithm. It uses two parameters: a reachability distance ϵ and a minimal number of points *MinPts*, used to define the ϵ-neighborhood to be considered as a cluster. For an arbitrary fixed point from the data, the ϵ-neighborhood is constructed. This point and all its neighbors are considered to be a cluster when there are *MinPts* or more points in the ϵ-neighborhood. If a point is in the ϵ-neighborhood of one or more cluster points, it is added to this cluster. Otherwise, it is used to create a new cluster using the same procedure. Manhattan distance is often used as the dissimilarity measure in the ordinal case and the mutual information in the nominal case. The reachability distance ϵ is chosen experimentally through a series of tests. The minimal number of points to constitute a cluster *MinPts* is fixed by the user.
See the *dbscan* function of the *fpc* package [15] from **R**.

COBWEB

COBWEB [10] is a conceptual clustering approach based on the category utility measure [5]. This incremental hierarchical algorithm is designed for qualitative data. It constructs a tree dynamically, inserting one individual at a time in the tree construction. At each individual insertion, COBWEB has four available options: inserting the individual to an existing cluster, creating a new cluster (represented by the new individual), merging two nodes, or splitting one node. For each option, the category utility is computed for each corresponding partition. The partition that provides the maximum value for this measure is selected. Then, the next individual in the dataset is considered. The stopping criterion is a minimal value of the category utility that can not be exceeded.
The *Cobweb* function [14] of the *RWeka* package from **R** can be used.

DIVCLUS-T

DIVCLUS-T [4] is a top-down hierarchical clustering method. It is monothetic, i.e subsets of observations in the dataset are split using single variable. Its goal is to optimize the same criterion as for classical CAH using Ward's method. It is designed for quantitative, qualitative and mixed data. To compute the splitting criterion, i.e the within-class inertia, DIVCLUS-T uses the euclidean distance in the quantitative case while it uses the Khi-2 distance in the qualitative case. The main advantage of this method, in comparison with other hierarchical methods, is that in addition to providing a dendrogram, each node is labeled by its binary splitting rule. So the dendogram provided by DIVCLUS-T can be read exactly like a binary decision tree.
The algorithm is available with the *divclust* function of the *divclust* package from **R**.

What is CUBT ?

CUBT, top-down hierarchical clustering method that consists of three stages:

Growing stage grows a "maximal tree" by recursively splitting the dataset into several subsets of observations

Pruning stage prunes the tree by aggregating pairs of sibling nodes that have a minimal dissimilarity measure

Joining stage aggregates non sibling nodes by using either the heterogeneity criterion or the dissimilarity measure

Similarities with CART

CUBT shares with CART various advantages:

- efficiency
- flexibility
- interpretability
- good convergence properties

Notations

$X \in E = \prod_{j=1}^{p} \{1, ..., m_j\}$ random p-dimensional discrete vector

$X_{.j}$ j^{th} component of X, $j \in \{1, ..., p\}$

$m_j \in \mathbb{N}$ number of categories of the jth variable

X_i independent variable identically distributed as X

S set of n random independent realizations of the variables

X_{ij} i^{th} observation of the jth variable of S

t node of a tree, subset of En

$n_t = card(t)$ size of the node t

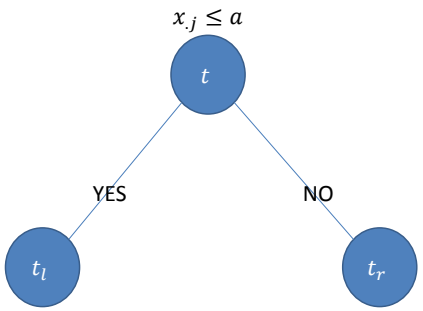
Growing stage: Heterogeneity measure

For any node t, let $X^{(t)}$ be the restriction of X to node t i.e. $X^{(t)} = X|\{X \in t\}$, we define $R(t)$, the heterogeneity measure of t, the *deviance*, as follows:

$$\begin{aligned} R(t) &= trace(\mathbf{MI}(X^{(t)})) \\ &= \sum_{j=1}^{p}(-\sum_{i \in t} P(X_{ij}) \log_2 P(X_{ij})) \end{aligned}$$

where $\mathbf{MI}(X^{(t)})$ is the mutual information matrix of $X^{(t)}$.

Splitting a node



$$t_l = \{x \in E : x_{.j} \leq a\} \qquad t_r = \{x \in E : x_{.j} > a\}$$

$$\underset{(j,a)\in\{1,\ldots,p\}\times\{1,\ldots,m_j\}}{\operatorname{argmax}} \{\Delta(t,j,a)\}$$

with

$$\Delta(t,j,a) = R(t) - R(t_l) - R(t_r)$$

Stopping rules

The splitting process is stopped when one of the two following criteria is satisfied:

- $n_t < minsize$ ($minsize \in \mathbb{N}$)
- $\Delta(t, j, a) < mindev \times R(S)$ ($mindev \in [0, 1]$)

A partition of the initial dataset is obtained and each leaf from the tree corresponds to a cluster.

Dissimilarity measure

For each $X_i \in t_r$ and $X_{i'} \in t_l$ we consider:

$$\tilde{d}_i = \min_{x \in t_l} d(x, X_i) \qquad \text{and} \qquad \tilde{d}_{i'} = \min_{x \in t_r} d(x, X_{i'})$$

and their ordered versions d_i and $d_{i'}$.

$d(X, Y)$ can be either the *Manhattan distance* or the *mutual information* between two random variables X and Y.

Dissimilarity measure

For $\delta \in [0, 1]$, we define:

$$\overline{d}_l^{\delta} = \frac{1}{\delta n_l} \sum_{i=1}^{\delta n_l} d_i \qquad \text{and} \qquad \overline{d}_r^{\delta} = \frac{1}{\delta n_r} \sum_{i=1}^{\delta n_r} d_j$$

Thus, the empirical dissimilarity measure between t_l and t_r is computed as follows:

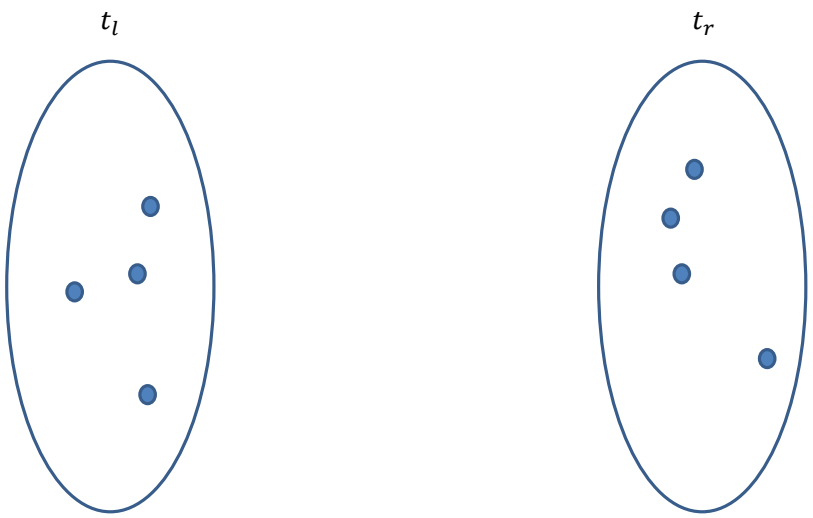
$$d^{\delta}(l, r) = d^{\delta}(t_l, t_r) = \max(\overline{d}_l^{\delta}, \overline{d}_r^{\delta}).$$

Pruning criterion

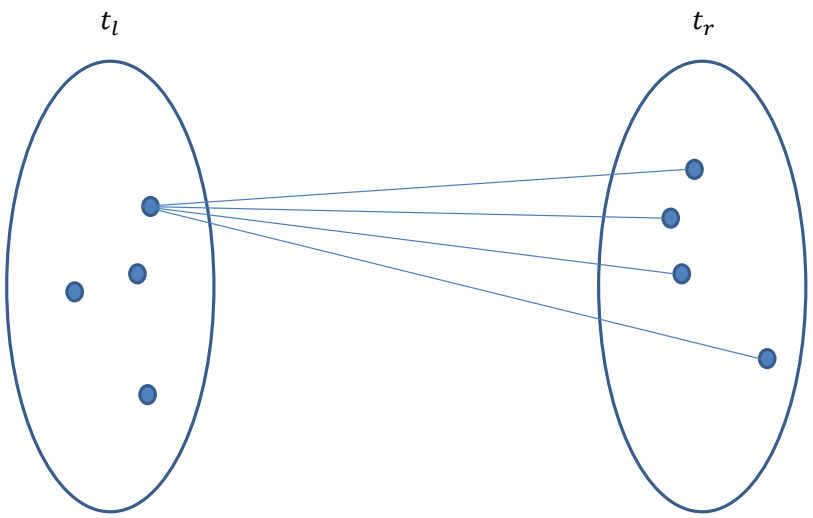
If $d^{\delta}(l, r) \leq mindist$ ($mindist > 0$) then t_l and t_r are aggregated.

Dissimilarity measure: illustration

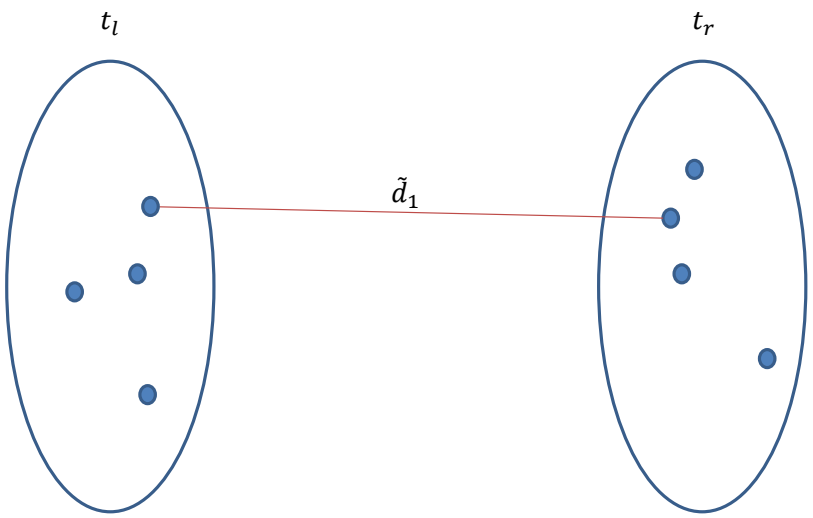
Dissimilarity measure: illustration



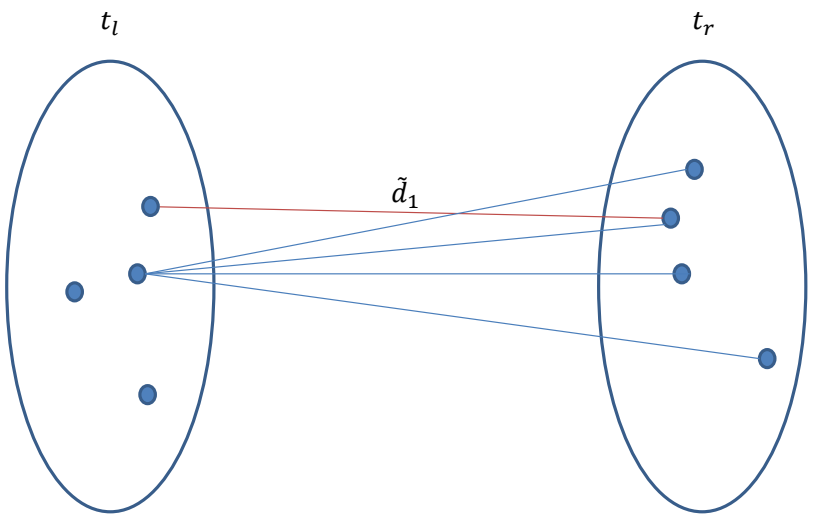
Dissimilarity measure: illustration



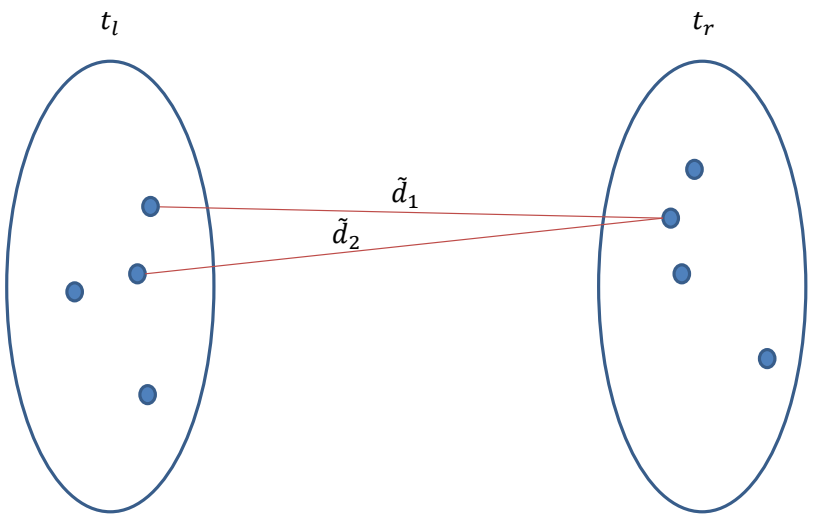
Dissimilarity measure: illustration



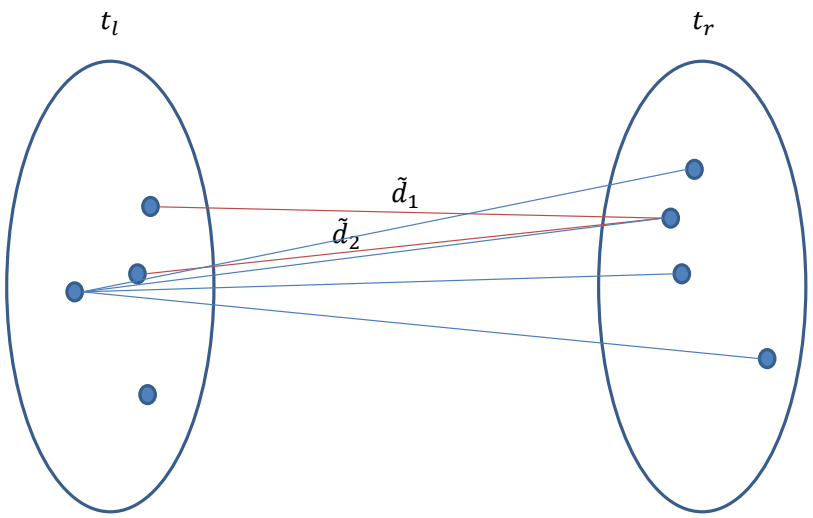
Dissimilarity measure: illustration



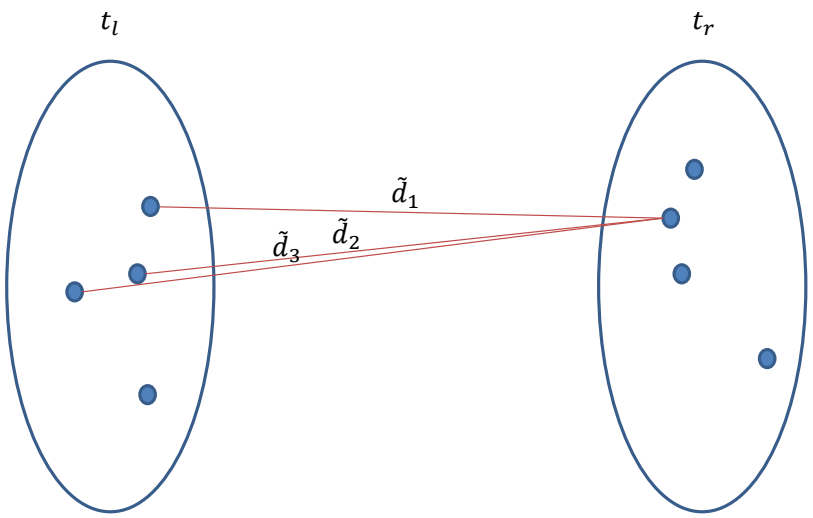
Dissimilarity measure: illustration



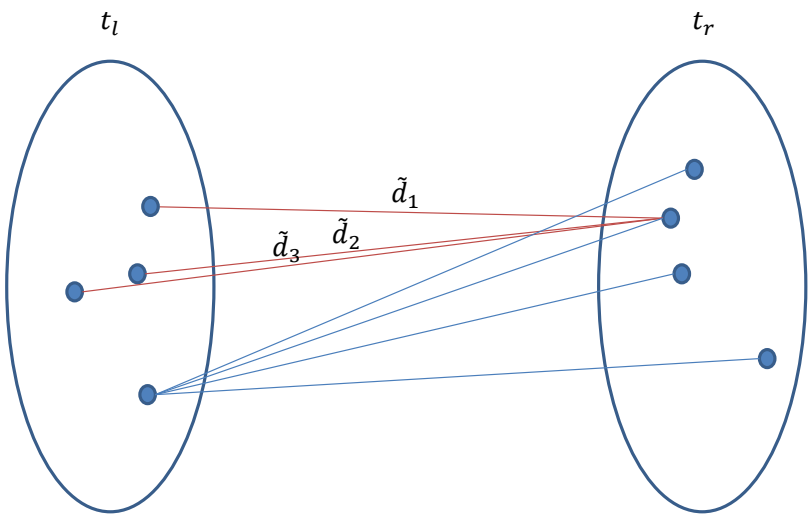
Dissimilarity measure: illustration



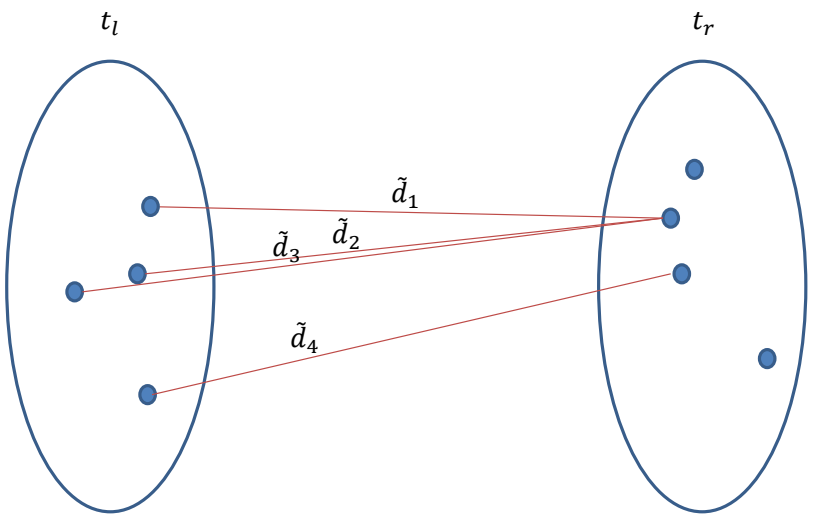
Dissimilarity measure: illustration



Dissimilarity measure: illustration



Dissimilarity measure: illustration



Joining stage

Aggregating pairs of non sibling nodes

Two joining criteria

Each pair of nodes t_l and t_r is compared using:

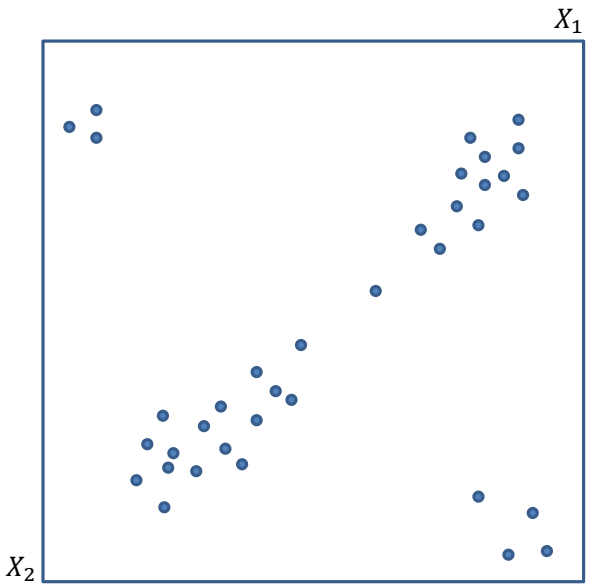
1. $\Delta(t_l, t_r) = R(t_l \cup t_r) - R(t_l) - R(t_r)$
2. $\Delta(t_l, t_r) = d^{\delta}(l, r)$

Let N_L be the total number of leaves and k the a priori number of clusters. For each pair (l, r) $(l, r \in \{1, ..., N_L\}$ and $l \neq r$, we have
$(\tilde{l}, \tilde{r}) = \text{argmin}_{l,r} \{\Delta(t_l, t_r)\}$

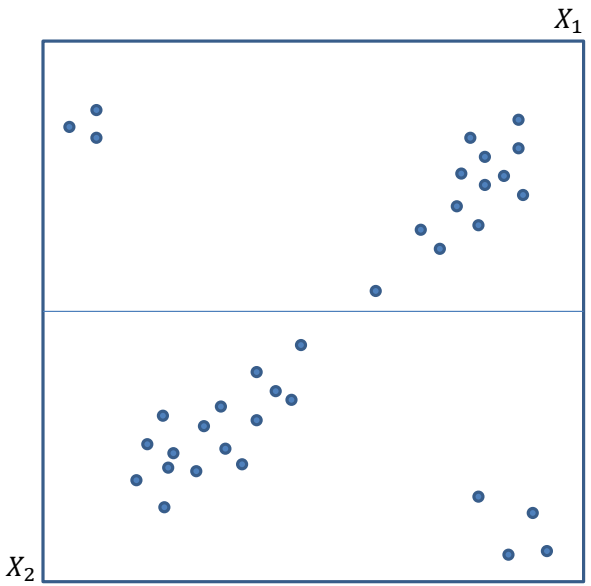
Node aggregation

$t_{\tilde{l}}$ and $t_{\tilde{r}}$ are replaced by their union $t_{\tilde{l}} \cup t_{\tilde{r}}$ and $N_L \leftarrow N_L - 1$. The process stops when $N_L = k$.

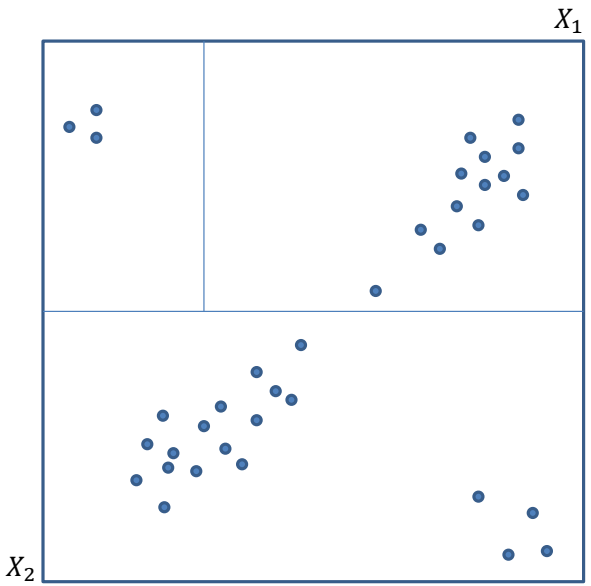
Joining stage: illustration



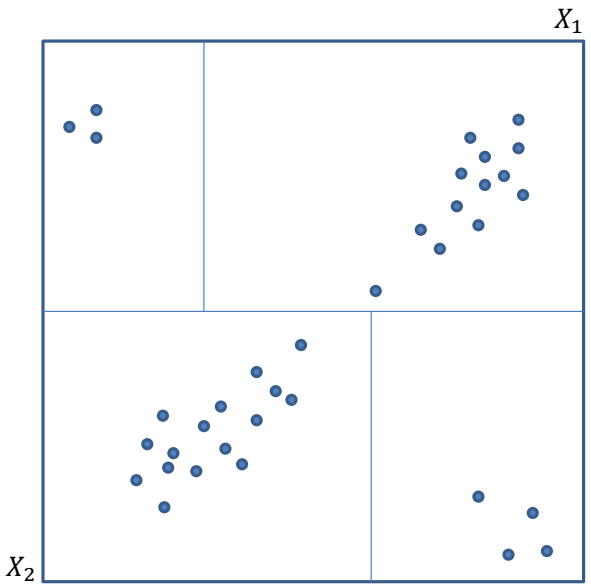
Joining stage: illustration



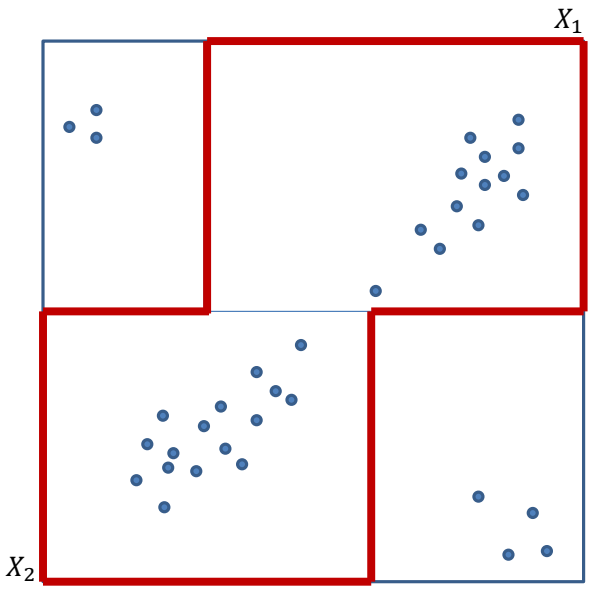
Joining stage: illustration



Joining stage: illustration



Joining stage: illustration



Data simulation models

$k = 3$, number of groups

$p = 9$, number of variables

$n \in \{100, 300, 500, 1000\}$ for the sample sizes

$m_j = m = 5$ number of levels for $j \in \{1, ..., p\}$

$\left\lfloor \frac{n}{k} \right\rfloor$, number of observations per group

Tuning the method

- For the growing stage we use $minsize = \lfloor ln(n) \rfloor$ and $mindev = 0.001$.
- For the pruning stage we fix $\delta = 0.3$ and *mindist* as the fourth quantile of the distribution of distances between sibling nodes.
- We choose mutual information as the measure of dissimilarity.
- Finally, the only parameter to be fixed in the joining stage is the number of classes $k = 3$.

We perform 100 replicates for each model. We compare our results with the results obtained using different existing methods.

Three clustering methods

k-modes We use the simple-matching distance to assess the dissimilarity between pairs of observations.

k-medians We use the Manhattan distance to assess the dissimilarity between the observations and the cluster medians.

Hierarchical clustering We use the complete linkage option to aggregate clusters and the Manhattan distance as the dissimilarity measure of the observations.

model	n	HCA	k-modes	k-medians	CUBT
Frequentist	100	0.064	0.008	0.084	0.150
	300	0.130	0.008	0.120	0.100
	500	0.160	0.022	0.074	0.150
	1000	0.150	0.005	0.075	0.160
IRT-based	100	0.023	0.062	0.009	0.005
	300	0.023	0.078	0.005	0.001
	500	0.025	0.062	0.005	0.001
	1000	0.024	0.077	0.005	0.001

Table: Simulations results: Matching Error for both frequentist and IRT models

CUBT for nominal data

The criteria for growing pruning and joining are the same as for the ordinal case. The only difference is for the choice of splits. This is done in CART,

$$X_{.j} \in \mathcal{A}$$

where $\mathcal{A}$ is a subset of the levels of X.
Models used for simulations: a theoretical tree, and Nominal IRT.
Results:

n	Model	hca.err	cubt.err	hca.RI	cubt.RI
100	tree	0.74	0.58	$1.3e-03$	0.130
300	tree	0.75	0.60	$1.4e-04$	0.120
500	tree	0.75	0.66	$4.8e-05$	0.040
100	nrm	0.74	0.59	$1.3e-03$	0.083
300	nrm	0.75	0.63	$1.4e-04$	0.028
500	nrm	0.75	0.66	$4.8e-05$	0.016

Plan

1 Introduction

2 Density estimation

3 Clustering

4 Simulations and performance evaluation

5 Recent approaches

6 Extensions

Substitution splits

Let s be any apparent split of a node t of the actual tree, splitting t into t_L and t_R. Let s_j a split over the jth variable chosen from the set S_j of all the possible splits (for continuous variables the splits having the form $x_j < a$ or $x_j > a$). s_j gives rise to two sub nodes t'_L and t'_R. The probability for an observation to be at the left for both splits is:

$$p(t_L \cap t'_L) = \frac{\#\{t_L \cap t'_L\}}{n_t}$$

The probability that both splits send an observation to the left is:

$$p_{LL}(s, s_j) = \frac{p(t_L \cap t'_L)}{p(t)}$$

p_{RR} maybe defined equivalently.

Substitution splits and Variable Importance

The probability that s_j predicts well s is

$$p(s, s_j) = p_{LL} + p_{RR}$$

$\tilde{s}_j$ is a substitution split for s if

$$p(s, \tilde{s}_j) = \max_{S_j} p(s, s_j)$$

The importance of variable j is given by:

$$I(X_{.j}) = \sum_t \Delta(R(\tilde{s}_j, t))$$

which the total of the deviance reduction induced if each split in the tree was replaced by the substitution split over $X_{.j}$.

Handling Missing values

For the prediction case, one may use substitution splits to follow a path into the tree when the observation of the apparent variable is missing.
Another possibility is to use the empirical proportion of observations at each side of a node in order to choose the direction to follow.

CUBT and functional data

Let $X \in L^2(\mathbf{R})$, observed at some discrete points, not necessarily the same for each observation. Two issues must be addressed to apply CUBT:

- How to split a node ?
- Which criterion ?

The first may be addressed using ideas from Yamada *et. al.* (CART with time series), and Roche A.:

$$D(x, x_0) \leq a$$

where x is an observation to route from a node, x_0 and a are a reference observation and a threshold to determine, and D is a similarity measure. DTW may be a good choice for the latter.

Functional CUBT ?

Bibliography

Blockeel, H. and De Raedt, L. (1998). Top-down induction of first-order logical decision trees. Artificial Intelligence, 101, 1-2:285-297. doi:10.1016/S0004-3702(98)00033-6.

Bock, R. D. (1972). Estimating item parameters and latent ability when responses are scored in two or more nominal categories. Psychometrika, 37, 29-51.

Breiman, L., Friedman, J., Stone, C.J. and Olshen, R.A. (1984). Classification and regression trees. Editions Chapman & Hall/CRC, Monterey, CA.

Chavent, M., Lechevallier, Y. and Briant, O. (2007). DIVCLUS-T: a monothetic divisive hierarchical clustering method. Computational statistics and data analysis, 52(12), 687-701.

Corter, J. E. and Gluck, M. A. (1992). Explaining basic categories: Feature predictability and information. Psychological Bulletin, 111(2), 291-303.

Dempster, A.P., Laird, N.M. and Rubin, D.B. (1977). Maximum Likelihood from Incomplete Data via the EM algorithm. Journal of the Royal Statistical Society, Series B, vol 39, 1:1-38.

De Raedt, L. and Blockeel, H. (1997). Using Logical Decision Trees for Clustering. In Proceedings of the 7th International Workshop on Inductive Logic Programming, 133-141, Springer-Verlag.

Dunn, J. C. (1973). A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics 3, 3:32-57. doi:10.1080/01969747308546227.

Ester, M., Kriegel, H. P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 226-231.

Fisher, D. H. (1987). Knowledge Acquisition Via Incremental Conceptual Clustering. Machine Learning 2, 2:139-172. doi:10.1023/A:1022852608280.

Fraiman, R., Ghattas, B. and Svarc, M. (2013). Interpretable clustering using unsupervised binary trees. Advances in data analysis and classification, 7, 125-145.

Ghattas, B., Svarc, M. and Fraiman, R. (2013). R package for interpretable clustering using binary trees. http://laminar.univ-mrs.fr/ghattas/CUBT.html

Gluck, M. A. and Corter, J. E. (1985). Information, uncertainty and the utility of categories. In Proceedings of the Seventh Annual Conference of the Cognitive Science Society, Hillsdale NJ: Lawrence Erlbaum.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11, Issue 1.

Hennig, C. (2014). fpc: Flexible procedures for clustering. R package version 2.1-9, http://CRAN.R-project.org/package=fpc.

Huang, Z. (1998). Extensions to the k-modes algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2(3), 281-314.

Hubert, L. and Arabie, P. (1985). Comparing partitions. Journal of Classification, 2(1), 193-218.

Jagannatha Reddy, M. V. and Kavitha, B. (2012). Clustering the Mixed Numerical and Categorical Dataset using Similarity Weighted Fuzzy Method International Journal of Database Theory and Applications, 5(1), 111-124.

Kedzierska, T. M. and Markusen, R. R. (2012). Partitioning Clustering algorithm for handling numerical and categorical data: a review, eXiv, arXiv:1211.7231v1.

Lebret, F. (2013). A model for K-centroids clustering in R. Computational Statistics and Data Analysis, 61(2), 131-146.

Liu, B., Xia, Y., and Yu, P. S. (2000). Clustering Through Decision Tree Construction. In Proceedings of the Ninth International Conference on Information and Knowledge Management, 20-29, CIKM 2000, New York, NY, USA: ACM. doi:10.1145/354756.354775.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Editions L. M. Le Cam & J. Neyman, 1, 281-297.

Michalski, R.S. (1983). Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. International Journal of Policy Analysis and Information Systems, 4(2), 125-161.

Muraki, E. (1992). A generalized partial credit model: application of an EM algorithm. Applied Psychological Measurement, 16, 159-176.

Mirkin, B. (2005). Multidimensional Clustering Algorithms. COMPSTAT Lectures 4, Physica-Verlag, Vienna.

Papadimitriou, C. and Steiglitz, K. (1982). Combinatorial Optimization: Algorithms and Complexity. Englewood Cliffs: Prentice Hall.

Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, Inc.

Rizopoulos, D. (2006). ltm: An R package for Latent Variable Modelling and Item Response Theory Analyses. Journal of Statistical Software, 17, 5, 1-25. http://www.jstatsoft.org/v17/i05/.

Rokach, L. (2010). A survey of Clustering Algorithms. In Maimon, O. and Rokach, L. (eds.), Data Mining and Knowledge Discovery Handbook, 269-298, Springer-US.

Weihs, C., Ligges, U., Luebke, K. and Raabe, N. (2005). klaR Analyzing German Business Cycles. In Baier, D., Decker, R. and Schmidt-Thieme, L. (eds.), Data Analysis and Decision Support, 335-343, Springer-Verlag, Berlin.

Zimmermann, A. and De Raedt, L. (2009). Cluster-Grouping: From Subgroup Discovery to Clustering. Machine Learning 77, 1:125-159. doi:10.1007/s10994-009-5132-5.