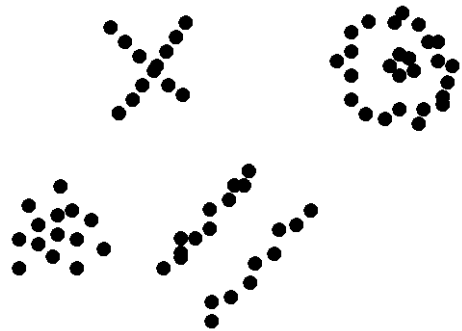
# 1. OHJAAMATON OPPIMINEN JA KLUSTEROINTI

Ohjaamattomassa oppimisessa on tavoitteena muodostaa hahmoista ryhmiä, klustereita, joiden sisällä hahmot ovat jossain mielessä samankaltaisia ja joiden välillä on selkeitä eroja

Erilaisia klustereita:



Käytettävissä oleville havainnoille ei ole valmiina mitään luokitustietoa ('labeling'). Myös klustereiden lkm voi olla tuntematon

Ohjaamattoman oppimisen menetelmiä voidaan myös käyttää ongelmissa, joissa havainnoilla on luokitustieto, mutta halutaan selvittää luokkien sisäinen rakenne

Ohjaamattoman oppimisen menetelmät voidaan jakaa lähestymistavoiltaan kahteen luokkaan:

- Parametriset menetelmät: yritetään yhtäaikaisesti selvittää sekä havaintojen luokittelu että luokkien tnjakaumien parametrit (mikstuurimallit & EM-algoritmi!)
- Epäparametriset menetelmät: jaetaan havainnot eri luokkia vastaaviksi osajoukoiksi, luokkien tnjakaumia ei estimoida

Ensimmäinen lähestymistapa on suoraviivainen laajennus tilastolliselle hahmontunnistukselle; jälkimmäinen lähestymistapa on tilastollisessa mielessä epäoptimaalinen, mutta laskennallisesti yleensä huomattavasti helpommin käsiteltävä

Klusterointiongelman ratkaiseminen voidaan jakaa seuraaviin vaiheisiin:

- Piirteiden valinta (esikäsittely & normalisointi!)
- Samankaltaisuus/erilaisuusmitan valinta havaintoparien, havaintojen ja klustereiden, sekä klusteriparien välille (piirteiden samanarvoinen kohtelu!)
- Klusterikriteerin valinta
- Klusterointialgoritmin valinta
- Tulosten validointi erilaisten testien avulla
- Tulosten tulkinta

Huom! Saatu ratkaisu on erittäin subjektiivinen, koska sama havaintojoukko voidaan jakaa hyvin erilaisiin klustereihin riippuen em valinnoista. Yleensä ratkaisun tulkinnan ja sen hyvyyden arvioinnin suorittaa sovellusalan asiantuntija

Havaintojen klusterointia voidaan käyttää luokittelun lisäksi seuraaviin tarkoituksiin:

- Datan tiivistys: havainto esitetään piirrevektorin sijasta klusterin koodinumeron avulla (LVQ!)
- Havaintoihin liittyvien hypoteesien muodostus (SOM!) ja testaus
- Havaintoihin perustuva ennustaminen: havainnon puuttuvat tiedot ennustetaan muiden samaan klusteriin kuuluvien havaintojen perusteella

Erilaisia klustereiden määrittelytapoja

- Luonnolliset klusterit ('natural clusters'): piirreavaruuden alueita, joissa havainnot ovat suhteellisen tiheässä ja joiden välissä havainnot ovat suhteellisen harvassa
- Todennäköisyyksiin perustuvat klusterit: havainto $\mathbf{x}$ kuuluu siihen klusteriin $C_k$, jonka *a posteriori* tn on korkein eli $P(C_k|\mathbf{x}) \geq P(C_i|\mathbf{x})\ \forall i = 1, \ldots, m$
- Yksikäsitteinen ('hard', 'crisp') klusterointi:

$$C_i \neq \emptyset,\ i = 1, \ldots, m \tag{1}$$
$$\cup_{i=1}^{m} C_i = X \tag{2}$$
$$C_i \cap C_j = \emptyset,\ i \neq j,\ i, j = 1, \ldots, m \tag{3}$$

($C_i$ on $i$. klusteri, $m$ klustereiden lkm ja $X$ kaikki havainnot)

- Sumea ('fuzzy') klusterointi. Havainnon kuulumisasteet eri klustereihin ilmaistaan jäsenyysfunktioiden ('membership functions') avulla:

$$u_j : X \to [0,1],\ j = 1, \ldots, m \tag{4}$$

$$\sum_{j=1}^{m} u_j(\mathbf{x}_i) = 1,\ i = 1, \ldots, N \tag{5}$$

$$0 < \sum_{i=1}^{N} u_j(\mathbf{x}_i) < N,\ j = 1, \ldots, m \tag{6}$$

($\mathbf{x}_i$ on $i$. havainto, $u_j$ klusteriin $C_j$ liittyvä kuuluvuusfunktio, $m$ klustereiden lkm, $X$ kaikki havainnot ja $N$ havaintojen lkm)

## 1.1 'Sequential clustering algorithms'

Havainnot $\mathbf{x}_1, \ldots, \mathbf{x}_N$ käydään läpi yksitellen, mahdollisesti useammin kuin yhden kerran

Tuottavat havainnoille yhden klusteroinnin

Klustereiden lkm:ää $m$ ei tarvitse tietää etukäteen

Havainnon ja klusterin samankaltaisuutta kuvataan funktion $d(\mathbf{x}, C)$ avulla

Klusteriparin samankaltaisuutta kuvataan funktion $d(C_i, C_j)$ avulla

## 'Basic Sequential Algorithmic Scheme', BSAS

Klusterointialgoritmin, jossa havainnot käsitellään yksitellen, perusversio:

- Määrätään klustereiden lkm:n yläraja $q$ ja kynnysarvo $\Theta$ havainnon ja klusterin samankaltaisuudelle
- Lähdetään liikkeelle yhdestä klusterista: $m = 1$, $C_m = \{\mathbf{x}_1\}$
- Käydään havainnot $\mathbf{x}_i$ läpi yksitellen $i = 2, \ldots, N$ ja
  - Etsitään klusteri $C_k$, jolle pätee $d(\mathbf{x}_i, C_k) = \min_{1 \leq j \leq m} d(\mathbf{x}_i, C_j)$
  - Jos $d(\mathbf{x}_i, C_k) > \Theta$ ja $m < q$, silloin $m = m + 1$ ja $C_m = \{\mathbf{x}_i\}$
  - Muuten, $C_k = C_k \cup \{\mathbf{x}_i\}$ ja päivitetään $d(\mathbf{x}, C_k)$

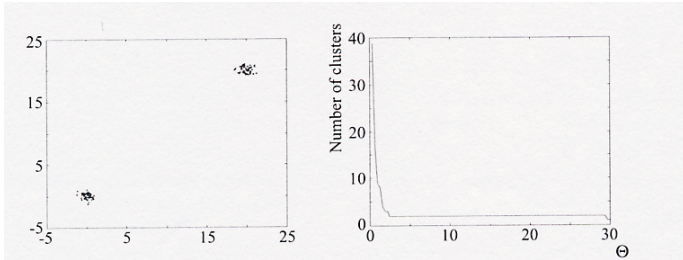
Havaintojen läpikäyntijärjestyksellä, funktion $d(\mathbf{x}, C)$ ja kynnysparametrin $\Theta$ valinnalla on huomattava vaikutus BSAS-tyyppisen klusterointialgoritmin löytämään ratkaisuun

Funktion $d(\mathbf{x}, C)$ valinta vaikuttaa löydettävien klustereiden rakenteeseen. Jos esim. klusteri $C_k$ esitetään yhden vektorin $\mathbf{m}_k$ avulla ja määritellään $d(\mathbf{x}, C_k) = d(\mathbf{x}, \mathbf{m}_k)$, algoritmilla on taipumus löytää kompakteja klustereita

Kynnysparametri $\Theta$ vaikuttaa suoraan löydettävien klustereiden lkm:ään

Sopiva $\Theta$:n arvo (tai klustereiden lkm) voidaan estimoida seuraavasti:

- Kokeillaan erilaisia kynnysparametrin arvoja $\Theta = a, a+c, a+2c, \ldots, b$
- Suoritetaan klusterointi jokaisella kynnysarvolla $N$ kertaa ja käydään havainnot läpi aina erilaisessa satunnaisessa järjestyksessä
- Lasketaan jokaiselle kynnysarvolle keskimääräinen klusterien lkm ja piirretään tuloksista kuvaaja $\bar{m}(\Theta)$
- Kohdat, joissa $\bar{m}(\Theta)$ on pitkään vakioarvoinen, ovat hyviä $\Theta$:n arvoja


25
15
5
-5
-5
5
15
25
Number of clusters
40
30
20
10
0
0
10
20
30
Θ

BSAS-tyyppisen klusterointialgoritmin varjopuolia:

- Päätös siitä, mihin klusteriin havainto $\mathbf{x}_i$ kuuluu, perustuu vain aikaisemmin käsiteltyihin havaintoihin $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}$ eikä koko havaintoaineistoon $X$
- Kerran tehtyä päätöstä uudesta klusterista ei voida perua. Siksi lopullisessa klusteroinnissa voi olla joitain hyvin samankaltaisia klustereita, jotka olisi järkevä yhdistää
- Lopputulos riippuu siitä, missä järjestyksessä havainnot käydään läpi

BSAS-tyyppisille klusterointialgoritmille onkin olemassa erilaisia variaatiota, jotka huomioivat paremmin em ongelmat:

- Muodostetaan havaintojen 1.:llä läpikäyntikierroksella $m$ kpl:tta yhdestä havainnosta koostuvia klustereita. Jaetaan seuraavilla kierroksilla loput havainnot näihin klustereihin
- Määrätään kynnysarvo klusteriparin väliselle samankaltaisuudelle ja tutkitaan onko ratkaisussa kaksi klusteria, jotka ovat keskenään liian samankaltaiset. Jos on, yhdistetään samankaltaisin pari ja päivitetään klustereiden numerointi ja samankaltaisuusmitat $d(C_i, C_j)$. Toistetaan, kunnes kaikki klusterit ovat riittävän erilaisia
- Tarkistetaan lopuksi, mitkä ovat eri havaintoja parhaiten vastaavat klusterit ja jaetaan havainnot uudestaan näihin parhaisiin klustereihin. Päivitetään havaintojen ja klustereiden samankaltaisuusmitat $d(\mathbf{x}, C)$. Toistetaan, kunnes havaintojen jako klustereihin ei enää muutu

## 1.2 Hierarkiset klusterointialgoritmit

Hierarkiset klusterointialgoritmit muodostavat havainnoille klusterointiratkaisujen sarjan, joilla on selkeä sisäkkäinen ('nested') rakenne

Algoritmit voidaan jakaa kahteen alaluokkaan: kasaaviin ('agglomerative') ja pilkkoviin ('divisive') algoritmeihin

Kasaavat algoritmit muodostavat klusterointiratkaisujen sarjan yhdistelemällä klustereita. Algoritmi lähtee liikkeelle tilanteesta, jossa jokainen havainto vastaa yhtä klusteria ($m = N$). Algoritmin edetessä klustereiden lkm pienenee, kunnes kaikki havainnot kuuluvat yhteen klusteriin ($m = 1$)

Pilkkovat algoritmit toimivat päinvastoin. Aluksi kaikki havainnot kuuluvat yhteen klusteriin ($m = 1$) ja algoritmin edetessä klustereita jaetaan kahtia, kunnes jokaisessa klusterissa on tasan yksi havainto ($m = N$)

Kasaava klusterointialgoritmi muodostaa klusterointiratkaisujen sarjan $R_0, \ldots, R_{N-1}$, joilla on seuraava sisäkkäinen rakenne:

$$R_0 \sqsubset R_1 \sqsubset \cdots \sqsubset R_{N-1} \tag{7}$$

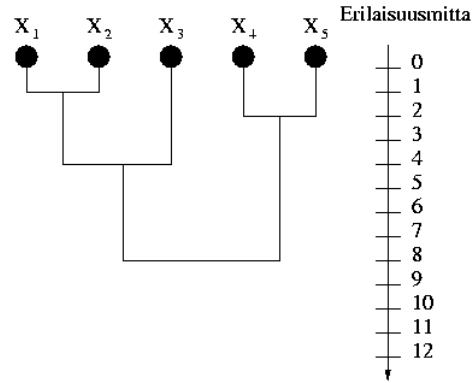
($R_0 = \{\{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_N\}\}$, $R_{N-1} = \{X\} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$)

Pilkkova klusterointialgoritmi muodostaa ratkaisujen sarjan $R_0, \ldots, R_{N-1}$, joilla on seuraava sisäkkäinen rakenne:

$$R_{N-1} \sqsubset R_{N-2} \sqsubset \cdots \sqsubset R_0 \tag{8}$$

($R_0 = \{X\} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $R_{N-1} = \{\{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_N\}\}$)

Hierarkisen klusterointialgoritmin tuottama ratkaisusarja voidaan esittää dendr
grammin avulla:




Dendrogrammin tasot vastaavat ratkaisuja, joissa on tietty määrä klustereita

Sopivan klustereiden lkm valitsee usein asiantuntija dendrogrammin avulla (pitkäikäiset klusterit!)

## 'Generalized Agglomerative Scheme' GAS

Kasaavan, hierarkisen klusterointialgoritmin yleistetty muoto:

- Klustereiden samankaltaisuutta/erilaisuutta mitataan funktion $g(C_i, C_j)$ avulla
- Alustus: aseta $R_0 = \{C_i = \{\mathbf{x}_i\},\ i = 1, \ldots, N\}$ ja $t = 0$
- Toista kunnes kaikki havainnot ovat yhdessä klusterissa eli $R_t = \{X\}$
  - t=t+1
  - Etsi ratkaisuun $R_{t-1}$ kuuluvista klustereista pari $(C_i.C_j)$, jolle

$$g(C_i, C_j) = \begin{cases} \min_{r,s} g(C_r, C_s), & \text{jos } g \text{ on erilaisuusmitta} \\ \max_{r,s} g(C_r, C_s), & \text{jos } g \text{ on samankaltaisuusmitta} \end{cases} \tag{9}$$

  - Määritä uusi klusteri $C_q = C_i \cup C_j$ ja ratkaisu $R_t = (R_{t-1} - \{C_i, C_j\}) \cup \{C_q\}$

– Päivitä klustereiden numerointi ja funktio $g$

Edellä esitetyn GAS-tyyppisen klusterointialgoritmin heikkous on se, että klustereiden yhdistämistä ei voida koskaan perua

Askeleella $t$ klustereiden lkm on $N-t$ ja yhdistettävän klusteriparin löytämiseksi on tehtävä

$$\binom{N-t}{2} = \frac{(N-1)(N-t-1)}{2} \tag{10}$$

klustereiden välistä vertailua

Kaikilla askelilla vertailuja tarvitaan yhteensä

$$\sum_{t=0}^{N-1} \binom{N-t}{2} = \sum_{k=1}^{N} \binom{k}{2} = \frac{(N-1)N(N+1)}{6} \tag{11}$$

## 'Matrix Updating Algorithmic Scheme' MUAS

Eräs GAS-tyyppisten klusterointialgoritmien alalaji on matriisiteoriaan perustuvat MUAS-tyyppiset algoritmit, joissa päivitetään klustereiden samankaltaisuus/erilaisuusmatriisia $P(X)$. $P$:n alkio $(i, j)$ on klustereiden $\mathbf{C}_i$ ja $\mathbf{C}_j$ samankaltaisuutta/erilaisuutta kuvaava funktio $d(\mathbf{C}_i, \mathbf{C}_j)$

MUAS-algoritmin perusversio:

- Alustus: $R_0 = \{\{\mathbf{x}_i\}, \ldots, \{\mathbf{x}_N\}\}$, $P_0 = P(X)$, $t = 0$
- Toista kunnes kaikki havainnot ovat yhdessä klusterissa eli $R_t = \{X\}$
  - t=t+1
  - Etsi klusterit $C_i$ ja $C_j$ s.e. $d(C_i, C_j) = \min_{r,s=1,\ldots,N, r \neq s} d(C_r, C_s)$
  - Määritä uusi klusteri $C_q = C_i \cup C_j$ ja ratkaisu $R_t = (R_{t-1} - \{C_i, C_j\}) \cup \{C_q\}$
  - Muodosta matriisi $P_t$ matriisista $P_{t-1}$

Usein klustereiden erilaisuutta kuvataan funktion $d(C_i, C_j)$ avulla, jonka päivityssääntö voidaan kirjoittaa seuraavassa muodossa:

$$\begin{aligned} d(C_q, C_s) =& a_i d(C_i, C_s) + a_j d(C_j, C_s) + b d(C_i, C_j) \\ &+ c|d(C_i, C_s) - d(C_j, C_s)| \end{aligned} \tag{12}$$

Parametrien $a_i$, $a_j$, $b$ ja $c$ arvot riippuvat valitusta funktiosta $d(C_i, C_j)$

Kun $a_i = a_j = 1/2$, $b = 0$ ja $c = -1/2$ eli

$$d(C_q, C_s) = \min\{d(C_i, C_s), d(C_j, C_s)\}, \tag{13}$$

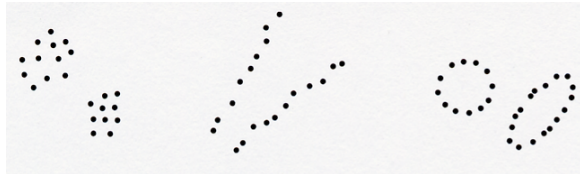
kyseessä on *'single link'* klusterointialgoritmi

Kun $a_i = a_j = 1/2$, $b = 0$ ja $c = 1/2$ eli

$$d(C_q, C_s) = \max\{d(C_i, C_s), d(C_j, C_s)\}, \tag{14}$$

kyseessä on *'complete link'* klusterointialgoritmi

'Single link' suosii pitkulaisia klustereita; 'complete link' taas kompakteja klustereita



**'Graph Theory-based Algorithmic Scheme' GTAS**

GTAS-tyyppiset algoritmit ovat GAS-tyyppisten algoritmien alalaji, jotka perustuvat graafiteoriaan

Näissä algoritmeissä havaintojoukko $X$ esitetään graafin avulla s.e. jokainen solmu vastaa yhtä havaintoa ja solmut, joiden välillä on polku, kuuluvat samaan klusteriin

Solmujen yhtenäisyyden lisäksi voidaan klustereille asettaa lisävaatimus $h(k)$, esim:

- Kaikki klusterin solmuparit on kytketty toisiinsa vähintään $k$:lla polulla, joilla ei ole yhteisiä solmuja
- Kaikki klusterin solmuparit on kytketty toisiinsa vähintään $k$:lla polulla, joilla ei ole yhteisiä kaaria
- Jokaisen solmun asteluku on vähintään $k$

GTAS-tyyppinen algoritmi saadaan suoraan GAS-algoritmin perusversiosta korvaamalla $g(C_i, C_j)$ rajoitusehdon $h(k)$ huomioivalla funktiolla $g_{h(k)}(C_i, C_j)$

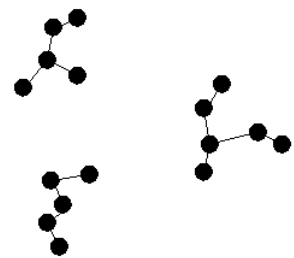
MST-algoritmi on eräs GTAS-tyyppinen algoritmi, joka perustuu minimivirittäjä-puuhun ('Minimum Spanning Tree') ja jolle

$$g(C_r, C_s) = \min_{i,j}\{w_{ij} : \mathbf{x}_i \in C_r, \mathbf{x}_j \in C_s\}, \tag{15}$$

missä $w_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$

MST-algoritmi ja 'single link' tuottavat saman ratkaisun, mikäli $d(\mathbf{x}_i, \mathbf{x}_j) \neq d(\mathbf{x}_k, \mathbf{x}_l)$, kun $i \neq k$ tai $j \neq l$

Esimerkki MST-algoritmilla muodostetusta klusteroinnista:



## 'Generalized Divisive Scheme' GDS

Pilkkovan klusterointialgoritmin yleistetty muoto:

- Klustereiden erilaisuutta mitataan funktion $g(C_i, C_j)$ avulla
- Alustus: aseta $R_0 = \{X\}$, $t = 0$
- Toista kunnes jokainen havainto vastaa yhtä klusteria eli $R_t = \{\{\mathbf{x}_1\}, \ldots \{\mathbf{x}_N\}\}$
  - $t = t + 1$
  - Käy läpi kaikki klusterit $C_i,\ i = 1, \ldots, t$, ja etsi jaot $(C_i^1, C_i^2)$, jotka maksimoivat $g(C_i^1, C_i^2)$:n
  - Pilko klusteri $C_j = \arg\max_{C_i, i=1,\ldots,t} g(C_i^1, C_i^2)$
  - Määritä uusi ratkaisu $R_t = (R_{t-1} - \{C_j\}) \cup \{C_j^1, C_j^2\}$
  - Päivitä klustereiden numerointi ja funktio $g$

GDS-tyyppinen klusterointialgoritmi on laskennallisesti hyvin vaativa