Klasteriranje podataka

Matej Mihelčić

Prirodoslovno-matematički fakultet, Sveučilište u Zagrebu

matmih@math.hr

13. svibnja, 2026.

Hijerarhijsko klasteriranje

- Algoritmi hijerarhijskog klasteriranja su razvijeni s ciljem otklanjanja nekih nedostataka particijskog (običnog) klasteriranja.
- Metode particijskog klasteriranja zahtjevaju korisnike da zadaju parametar K za uspješno klasteriranje. Te metode su često nedeterminističke.
- Hijerarhijski algoritmi su konstruirani da ponude deterministične i fleksibilnije mehanizme za klasteriranje entiteta iz skupova podataka.
- Hijerarhijske metode dijelimo na **aglomerativne** i **divizijske** metode.
- Aglomerativne metode startaju od klastera koji odgovaraju individualnim entitetima. Iterativno spajaju po dva klastera, stvarajući hijerarhiju klastera (od dna prema vrhu).
- Divizijske metode kreću od klastera koji sadrži sve entitete iz skupa podataka i iterativno dijele klastere u dvije grupe, generirajući hijerarhiju klastera (od vrha prema dnu).

Hijerarhijsko klasteriranje

- Hijerarhija klastera je binarno stablo koje u korjenu sadrži skup svih entiteta.
- Svaka razina predstavlja skupinu klastera. Entitete koji pripadaju svakom klasteru možemo dobiti tako da iz čvora klastera prijeđemo u čvorove koji odgovaraju listovima stabla.
- Navedena hijerarhija klastera se zove **dendrogram**.
- Bitna prednost u odnosu na particijsko klasteriranje je što hijerarhiju klastera možemo odrezati na proizvoljnoj razini, čime dobijemo klasteriranje određene rezolucije.
- Zbog toga kod hijerarhijskog klasteriranja ne trebamo definirati parametar K (broj klastera).

Aglomerativno klasteriranje

- Osnovni koraci aglomerativnog hijerarhijskog klasteriranja su:
  - izračun matrice razlika korištenjem zadane mjere udaljenosti, te crtanje točaka na dnu dendrograma.
  - spajanje najbližih skupova klastera i ažuriranje matrice razlika.
- Navedeni postupak aglomerativnog spajanja se ponavlja dok ne stvorimo konačan, maksimalan klaster koji sadrži sve entitete iz skupa podataka.

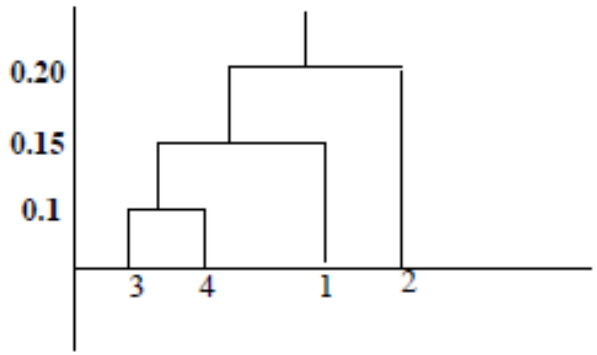
Aglomerativno klasteriranje jednom i potpunom vezom

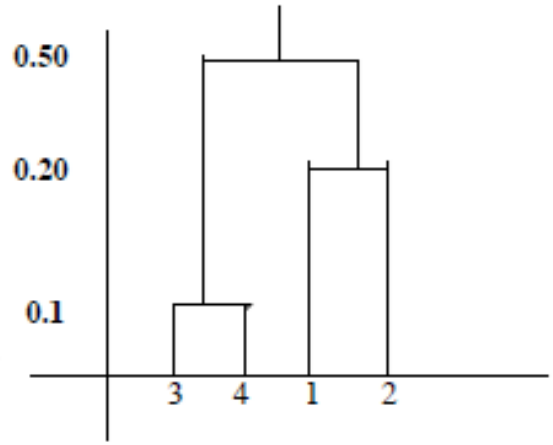
- Najpopularnije metode aglomerativnog klasteriranja su klasteriranje **jednom vezom** i klasteriranje **potpunom vezom**.
- Kod klasteriranja jednom vezom, sličnost parova klastera se definira kao sličnost između najbližih članova tog para. Veća važnost se daje regijama gdje su klasteri međusobno najbliži, zanemarujući ostale aspekte strukture klastera. Iz tog razloga klasteriranje jednom vezom spada u kategoriju klastering metoda baziranih na **lokalnoj sličnosti**.
- Zbog svojih lokalnih svojstava, klasteriranje jednom vezom može efektivno klasterirati ne-eliptične, izdužene grupe entiteta.
- Najveće mane pristupa su osjetljivost na šum i stršeće vrijednosti u podacima.
- Klasteriranje potpunom vezom mjeri sličnost para klastera kao sličnost njihovih najrazličitijih (najudaljenijih) članova.
- Biramo par klastera čiji spoj ima najmanji dijametar.

Aglomerativno klasteriranje jednom i potpunom vezom

- Klasteriranje potpunom vezom uzima u obzir strukturu klastera, stoga pripada ne-lokalnim metodama. Dobiva klastere kompaktnog oblika.
- Metoda je osjetljiva na stršeće vrijednosti.
- Obje metode (klasteriranje s jednom i potpunom vezom) imaju interpretaciju iz aspekta teorije grafova. Klasteri koje dobijemo metodom jedne veze odgovaraju komponentama povezanosti, klasteri dobiveni metodom potpune veze odgovaraju maksimalnim klikama grafa.

	1	2	3	4
1	0.0	0.20	0.15	0.30
2	0.20	0.0	0.40	0.50
3	0.15	0.40	0.0	0.10
4	0.30	0.50	0.10	0.0





Aglomerativno klasteriranje bazirano na prosjeku grupe i centroidu

- Aglomerativno klasteriranje **bazirano na prosjeku grupe** računa prosječnu udaljenost između svih parova svih točaka elemenata para klastera.
- Označimo li s C_a i C_b par klastera, $C_{a\cup b} = C_a \cup C_b$. Centroide tih klastera označimo c_a i c_b, tada $c_{a\cup b} = \frac{|C_a|c_a + |C_b|c_b}{|C_a| + |C_b|}$.
- Mjera sličnosti koju koristi klasteriranje bazirano na prosjeku grupe se definira kao:
$$S(C_a, C_b) = \frac{1}{(N_a + N_b)(N_a + N_b - 1))} \sum_{i \in C_a \cup C_b} \sum_{j \in C_a \cup C_b, i \neq j} d(i, j).$$
- Udaljenost između dva klastera je prosjek svih udaljenosti po parovima, stoga je **računalno skupo** računati navedeno klasteriranje.
- Aglomerativno klasteriranje bazirano na centroidima računa sličnost parova klastera kao sličnost (udaljenost) između centroida tih klastera.

Wardov kriterij

- Wardov kriterij mjeri udaljenost dva klastera prilikom aglomerativnog klasteriranja.
- Postupak spajanja klastera korištenjem Wardova kriterija se zove Wardova aglomeracija.
- Koristi sumu kvadratne pogreške (kao K-means) za određivanje udaljenosti.
- Za par klastera C_a i C_b mjerimo Wardovu udaljenost kao

$$\Delta W(C_a, C_b) = \frac{N_a N_b}{N_a + N_b} \sum_{v=1}^{|A|} (c_{a,v} - c_{b,v})^2 = \frac{N_a N_b}{N_a + N_b} d(c_a, c_b).$$

- Spajamo par klastera za koje Wardov Δ poprima najmanju vrijednost. Time spajamo klastere tako da rezultantni klaster ima najmanje raspršenje točaka.

Generalni algoritam aglomerativnog klasteriranja

Algoritam Aglomerativno hijerarhijsko klasteriranje

Izračunaj matricu razlika (udaljenosti) između svih točaka u skupu podataka.

repeat

Spoji klastere $C_{a\cup b} = C_a \cup C_b$ prema kriteriju spajanja.

Ubaci novi redak i stupac koji sadrži udaljenosti između $C_{a\cup b}$ i preostalih klastera.

until Ne ostane samo jedan, maksimalan klaster.

Lance-Williamsova formula za računanje udaljenosti

- Formula rješava problem ponovnog računanja udaljenosti između novo kreiranog klastera i postojećih klastera.
- Omogućava računanje udaljenosti koristeći ranije izračunate udaljenosti između klastera.
- $d(i \cup j, k) = \alpha_i d(i,k) + \alpha_j d(j,k) + \beta d(i,j) + \gamma |d(i,k) - d(j,k)|$.
  $\alpha_i,\ \alpha_j\ \beta,\ \gamma$ definiraju kriterije aglomeracije.

Ime metode	**Lance–Williams-ovi kriteriji**
Jedna veza	$\alpha_i = 0.5$; $\beta = 0$; i $\gamma = -0.5$
Potpuna veza	$\alpha_i = 0.5$; $\beta = 0$; i $\gamma = 0.5$
Prosjek grupe	$\alpha_i = \frac{\lvert i\rvert}{\lvert i\rvert+\lvert j\rvert}$; $\beta = 0$; i $\gamma = 0$
Centroid	$\alpha_i = \frac{\lvert i\rvert}{\lvert i\rvert+\lvert j\rvert}$; $\beta = -\frac{\lvert i\rvert\lvert j\rvert}{(\lvert i\rvert+\lvert j\rvert)^2}$; i $\gamma = 0$
Ward	$\alpha_i = \frac{\lvert i\rvert+\lvert k\rvert}{\lvert i\rvert+\lvert j\rvert+\lvert k\rvert}$; $\beta = -\frac{\lvert k\rvert}{\lvert i\rvert+\lvert j\rvert+\lvert k\rvert}$; i $\gamma = 0$

Divizijsko klasteriranje

- Divizijsko hijerarhijsko klasteriranje je pristup klasteriranja od vrha prema dnu (eng. top-down) kod kojeg krećemo od korjena koji sadrži sve entitete skupa podataka i rekurzivno dijelimo te tako konstruiramo dendrogram.
- Divizijsko klasteriranje ima veću učinkovitost u usporedbi s aglomerativnim klasteriranjem, pogotovo kada ne trebamo generirati potpunu hijerarhiju skroz do listova.
- Možemo ga smatrati globalnim pristupom jer sadrži potpunu informaciju prije dijeljenja podataka.

Divizijsko klasteriranje

Na divizijsko klasteriranje utječe nekoliko faktora:

- **Kriterij dijeljenja** - koristimo Wardov kriterij. Želimo pronaći dva centroida koji maksimiziraju Wardov kriterij. To će minimizirati sumu kvadratne pogreške rezultantnih klastera, odnosno maksimizirati razliku u sumi kvadratne pogreške između početnog skupa klastera i skupa klastera nakon podjele. U izračunu za kategorijske i binarne atribute koristimo Gini index (kao kod stabala odlučivanja).
- **Metoda podjele** - pravilna metoda može reducirati vrijeme potrebno za evaluaciju Wardovog kriterija. Koristi se bisekcijski K-means. Pokrećemo K-means uz $K = 2$ s centroidima određenim u prethodnom koraku.
- **Odabir klastera za podjelu** - računamo sume kvadratnih pogrešaka klastera, te biramo klaster s najvećom sumom.

Divizijsko klasteriranje

- **Umanjivanje utjecaja šuma** - šum može dovesti do stvaranja abnormalnih klastera. Utjecaj šuma možemo umanjiti tako da definiramo prag minimalnog poboljšanja sume kvadratnih pogrešaka (kriterij zaustavljanja). Kada je poboljšanje nakon dijeljenja manje od definiranog praga, prestajemo rekurzivno dijeliti klaster.

Algoritam Osnovni algoritam divizijskog klasteriranja

Krećemo od korjena koji sadrži sve entitete iz skupa podataka

repeat

Dijelimo čvor roditelja na dva dijela C_1 i C_2 koristeći bisekcijski K-means s ciljem maksimiziranja Wardove udaljenosti $W(C_1, C_2)$.

Konstruiramo dendrogram. Od postojećih klastera, biramo klaster s najvećom sumom kvadratne pogreške.

until Ne dobijemo listove koji sadrže samo jedan entitet ili ne postoji niti jedan klaster koji se može dalje dijeliti s obzirom na prag minimalnog poboljšanja.

Probabilističko klasteriranje

- Postupci klasteriranja bazirani na vjerojatnosnim modelima imaju široku primjenu.
- Korišteni su za segmentaciju slika, prepoznavanje pisma, klasteriranje dokumenata, modeliranje tema, dohvaćanje informacija, itd.
- Pristupi ove kategorije pokušavaju optimizirati slaganje između dostupnih podataka i nekog matematičkog modela koristeći vjerojatnosni pristup.
- Takve metode su bazirane na pretpostavci da su podaci generirani mješavinom vjerojatnosnih distribucija.
- Svaki klaster možemo reprezentirati parametriziranom vjerojatnosnom distribucijom, npr. Gaussovom ili Poissonovom distribucijom.
- Problem klasteriranja se transformira u problem procjene parametara, pošto cijeli skup podataka modeliramo mješavinom distribucija K komponenata.
- Entiteti (točke) koje najvjerojatnije pripadaju istoj distribuciji možemo definirati kao klaster.

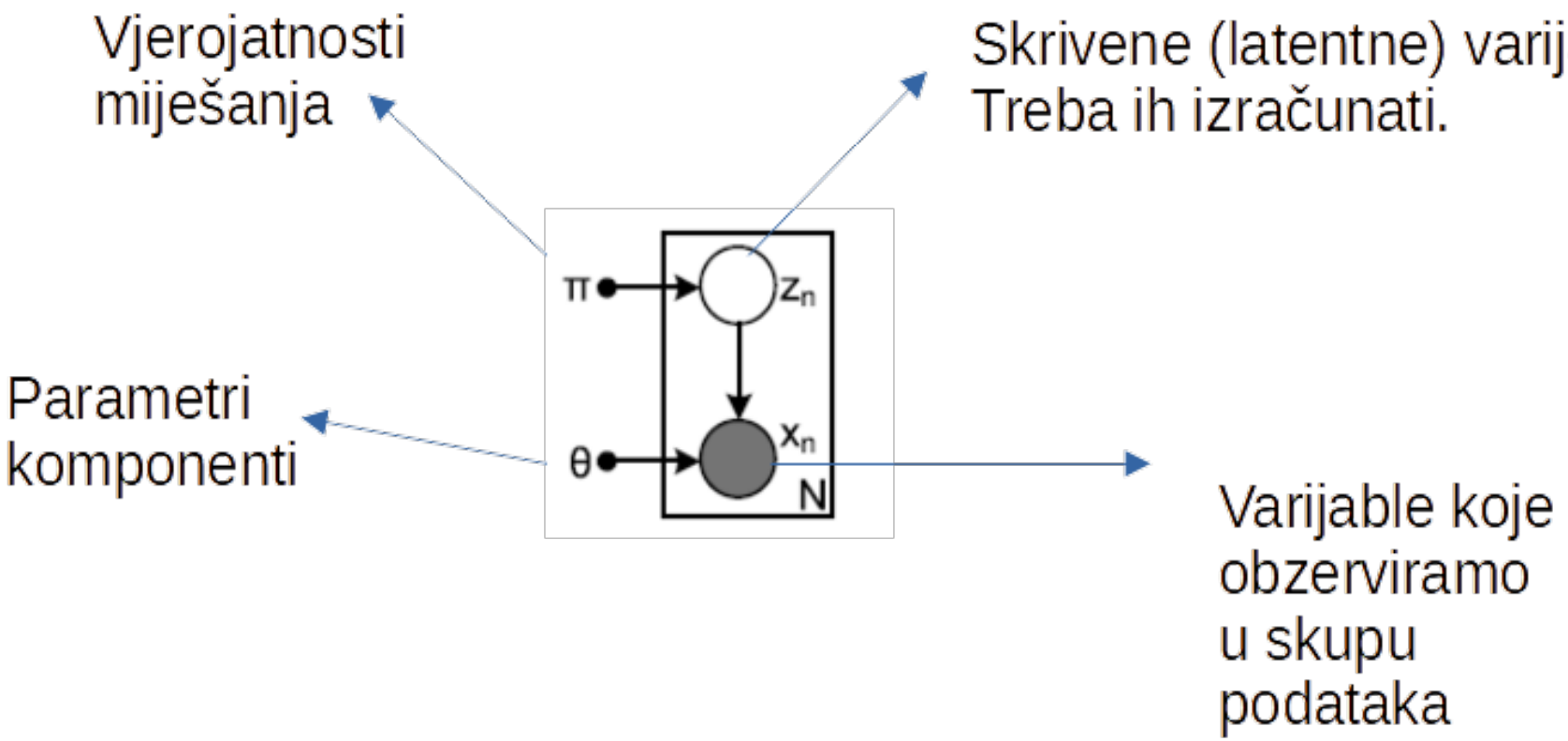
Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- Pretpostavka ovih modela je da su primjeri koje klasteriramo dobiveni iz jedne od nekoliko komponenti, a naš problem je procijeniti parametre svake komponente tako da dobijemo što bolje slaganje s dostupnim podacima.
- Računanje parametara komponenti i identifikacija komponente koja odgovara svakom primjeru u skupu podataka daje klasteriranje skupa entiteta.
- Pretpostavimo da imamo skup točaka $X = \{x_1, \ldots, x_N\}$ koji se sastoji od N opažanja D dimensionalne slučajne varijable x.
- Slučajna varijabla x_n je po pretpostavci distribuirana u skladu s mješavinom K komponenti.
- Svaka komponenta (klaster) se matematički predstavlja parametriziranom distribucijom.
- Distribucija koja modelira specifičan klaster se često naziva distribucija komponente.

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- Cijeli skup podataka modeliramo kao mješavinu takvih distribucija.
- Mješavina distribucija, odnosno vjerojatnosna funkcija gustoće od x_n je: $p(x_n) = \sum_{k=1}^{K} \pi_k p(x_n|\Theta_k)$. $\pi_1, \ldots, \pi_K$ su vjerojatnosti miješanja (možemo ih shvatiti i kao koeficijente miješanja ili težine), Θ_k je skup parametara koji specificiraju k-tu komponentu, a $p(x_n|\Theta_k)$ je komponenta distribucije.
- Vjerojatnosti miješanja moraju zadovoljavati $0 < \pi_k < 1$ $(k = 1, \ldots, K)$ i $\sum_{k=1}^{K} \pi_k = 1$.

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija



Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- Pretpostavimo da je z_n kategorička slučajna varijabla koja poprima vrijednosti $1, \ldots, K$ s vjerojatnostima $p(z_n = k) = \pi_k$.
- Pretpostavimo da je uvjetna distribucija od x_n uz dani $z_n = k$ jednaka $p(x_n|\Theta_k)$.
- Tada marginalnu distribuciju od x_n možemo dobiti sumiranjem zajedničke distribucije po svim mogućim vrijednostima od z_n,
$$p(x_n) = \sum_{k=1}^{K} \pi_k p(x_n|\Theta_k).$$
- z_n možemo smatrati vrijednošću labele klastera ili komponente (labela definira pripadnost primjera klasteru) slučajnog uzorka x_n.
- Umjesto korištenja jedne kategorijske varijable z_n uvodimo K-dimenzionalan binarni slučajni vektor z_n i njime označavamo labelu komponenata od x_n.

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- K-dimenzionalna slučajna varijabla z_n ima reprezentaciju oblika 1 od K, točno jedan element ima vrijednost 1, a preostali elementi vrijednost 0.
- Za $K = 5$ klastera i primjer za koji je $z_4 = 1$, pripadni vektor će biti reprezentiran kao $z_n = (0, 0, 0, 1, 0)^t$.
- Pošto za z_n imamo 1 od K reprezentaciju, tada je marginalna distribucija od z_n jednaka: $p(z_n) = \pi_1^{z_{n1}} \pi_2^{z_{n2}} \dots \pi_k^{z_{nK}} = \prod_{k=1}^{K} \pi_k^{z_{nk}}$.
- Uvjetna distribucija od x_n uz dani z_n je: $p(x_n|z_n) = \prod_{k=1}^{K} p(x_n|\Theta_k)^{z_{nk}}$.
- Zajednička distribucija je zadana s $p(z_n)p(x_n|z_n)$, a marginalna distribucija od x_n je $p(x_n) = \sum_{z_n} p(z_n)p(x_n|z_n) = \sum_{k=1}^{K} \pi_k p(x_n|\Theta_k)$.

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- Dani skup točaka je potencijalno generiran ponavljanjem procedure N puta, jednom za svaku točku x_n:
  - Biramo labelu skrivene komponente (klastera) $z_n \sim Mult_K(1, \pi)$. Time izabiremo k-tu komponentu iz koje generiramo x_n.
  - Uzorkujemo točku x_n iz k-te komponente prema uvjetnoj distribuciji $p(x_n|\Theta_k)$.
- Pošto je marginalna distribucija u obliku $p(x_n) = \sum_{z_n} p(x_n, z_n)$, slijedi da za svaku opaženu točku x_n postoji odgovarajuća latentna varijabla z_n.
- Korištenjem Bayesovog teorema možemo izračunati uvjetnu vjerojatnost da je $z_n = 1$ uz dani x_n:

$$p(z_{nk} = 1|\mathrm{x}_n) = \frac{p(z_{nk} = 1)p(\mathrm{x}_n|z_{nk} = 1)}{\sum_{j=1}^{K} p(z_{nj} = 1)p(\mathrm{x}_n|z_{nj} = 1)} = \frac{\pi_k p(\mathrm{x}_n|\theta_k)}{\sum_{j=1}^{K} \pi_j p(\mathrm{x}_n|\theta_j)}.$$

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- π_k je apriori vjerojatnost da je točka x_n generirana iz komponente k, a $p(z_{nk} = 1|x_n)$ je aposteriori vjerojatnost. $p(z_{nk} = 1|x_n)$ ćemo označavati s $\gamma(z_{nk})$. $\gamma(z_{nk})$ možemo razumjeti kao odgovornost koju komponenta k preuzima za objašnjavanje opažanja x_n.
- Zadatak je otkriti skup parametara iz opažanja, vjerojatnosti miješanja π_k i parametre distribucija komponenti Θ_k. Pretpostavljamo da je broj komponenti K fiksan.
- Svi parametri modela mješavine vjerojatnosnih distribucija su $\Theta = \{\pi_1, \ldots, \pi_K, \Theta_1, \ldots, \Theta_K\}$.
- Uz pretpostavku nezavisnog izabira točaka iz distribucije, vjerojatnost generiranja svih točaka možemo zapisati kao:

$$p(\mathrm{X}|\Theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k p(x_n|\Theta_k).$$

Modeli za klasteriranje bazirani na mješavini vjerojatnosnih distribucija

- Gornji izraz često zapisujemo u logaritamskom obliku:
$$\log p(\mathrm{X}|\Theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k p(x_n|\Theta_k).$$
- Parametre Θ_k ćemo procijeniti tako da računamo $\Theta_{ML} = arg \max_{\Theta} \log p(\mathrm{X}|\Theta)$. Ovaj postupak se zove maksimiziranje logaritma vjerodostojnosti.
- Ponekad imamo zadane apriori informacije o parametrima $p(\Theta)$ koje možemo ubaciti u model. Tako dobijemo maksimalnu aposteriori procjenu: $\Theta_{MAP} = arg \max_{\Theta} \log p(\mathrm{X}|\Theta) + \log p(\Theta)$.
- Mješavinu možemo konstruirati koristeći proizvoljne komponente (distribucije), međutim često se u praksi koriste mješavine gdje sve komponente dolaze iz iste parametrizirane familije distribucija, s potencijalno različitim parametrima (npr. Gaussove s različitim srednjim vrijednostima i varijancama).

Model baziran na mješavini Gaussovih distribucija

- Svaka komponenta u mješavini Gaussovih distribucija je normalna (Gaussova) distribucija.
- $\mathcal{N}(\mathrm{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{det(\boldsymbol{\Sigma})^{1/2}} \exp\left\{-\frac{1}{2}(\mathrm{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathrm{x}-\boldsymbol{\mu})\right\}$ - multivarijatna Gaussova razdioba, gdje $\boldsymbol{\Sigma}$ označava $D \times D$ kovarijacijsku matricu, a $\boldsymbol{\mu}$ D-dimenzionalni vektor očekivanja.
- $p(\mathrm{x}_n|\Theta) = p(\mathrm{x}_n|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$
- $l(\Theta) = \log p(\mathrm{X}|\Theta) = \sum_{n=1}^{N} \log p(\mathrm{x}_n|\Theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$
- Pošto imamo zadane distribucije u mješavini, sada možemo potražiti formulu za pronalazak parametara koji maksimiziraju funkciju log-vjerodostojnosti.

Model baziran na mješavini Gaussovih distribucija

- Da bi odredili parametre koji vrijede u lokalnom maksimumu računamo derivaciju od $\log p(\mathrm{X}|\pi, \mu, \Sigma)$ s obzirom na parametre π_k, μ_k, Σ_k.
- Derivacija s obzirom na μ_k je definirana kao: $\frac{\partial l}{\partial \boldsymbol{\mu}_k} =$

$$\sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1}(\mathrm{x}_n - \boldsymbol{\mu}_k) = \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{\Sigma}_k^{-1}(\mathrm{x}_n - \boldsymbol{\mu}_k).$$

- Iz toga dobijemo: $\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) \mathrm{x}_n}{\sum_{n=1}^{N} \gamma(z_{nk})}$.
- Također, $\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathrm{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$
- Na sličan način izračunamo Σ_k deriviranjem $l(\Theta)$ po tom parametru: $\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(\mathrm{x}_n - \boldsymbol{\mu}_k)(\mathrm{x}_n - \boldsymbol{\mu}_k)^T}{\sum_{n=1}^{N} \gamma(z_{nk})}$.

Model baziran na mješavini Gaussovih distribucija

- Derivacija od $l(\Theta)$ s obzirom na varijable π se ne može direktno računati jer imamo ograničenje na vrijednosti varijabli $0 < \pi < 1$, te uvjet $\sum_{k=1}^{K} \pi_k = 1$.
- Zbog toga koristimo Lagrangeove multiplikatore i maksimiziramo izraz: $\log p(X|\pi, \mu, \Sigma) + \lambda(\sum_{k=1}^{K} \pi_k - 1)$.
- Dobijemo: $\pi_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})}{N}$.
- Formule koje smo dobili ne daju zatvoreno rješenje za parametre μ, Σ, π, već imamo ovisnosti s $\gamma(z_{nk})$.
- Parametre možemo tražiti iterativno koristeći algoritam maksimiziranja očekivanja (eng. *Expectation-Maximization algorithm - EM*).

Model baziran na mješavini Gaussovih distribucija

Cilj

Zadan je skup podatkovnih točaka i model mješavine Gaussovih distribucija. Cilj je pronaći parametre Θ koji maksimiziraju log-vjerodostojnost.

Algoritam Maksimizacije očekivanja za Gaussiansku mješavinu

Inicijaliziraj očekivanja $\boldsymbol{\mu}_k^0$, kovarijance $\boldsymbol{\Sigma}_k^0$ i vjerojatnosti miješanja π_k^0.
repeat
  E-korak: Izračunaj odgovornosti $\gamma(z_{nk})$ pomoću trenutnih parametara.
  M-korak: Ažuriraj parametre pomoću trenutnih odgovornosti. Prvo ažuriraj očekivanja, zatim iskoristi te vrijednosti za računanje kovarijanci i na kraju procijeni vjerojatnosti miješanja.
  Izračunaj log-vjerodostojnost i provjeri konvergenciju algoritma.
until Kriterij konvergencije je zadovoljen
return Konačni parametri modela.

- K određujemo dodavanjem penalizacijskog člana $\mathcal{P}(k)$ i minimiziranjem $C(\Theta_k, k) = -\log p(\mathrm{X}|\Theta_k) + \mathcal{P}(k)$.

Općeniti algoritam maksimizacije očekivanja

Zadano: Statistički model koji se sastoji od skupa opaženih varijabli X, skupa neopaženih latentnih varijabli Z i vektora nepoznatih parametara Θ. Cilj je pronaći parametre Θ koji maksimiziraju log-vjerodostojnost.

Algoritam Generalni algoritam maksimizacije očekivanja

1: Započni s početnom procjenom parametara $\Theta^{(0)}$ i izračunaj početnu log-vjerodostojnost $\log p(\mathrm{X}|\Theta^{(0)})$.

repeat

2: E-korak: Izračunaj $q^{(t)} = \arg\max_q \mathcal{L}(q, \Theta^{(t)}) = p(\mathrm{z}_n|\mathrm{x}_n, \Theta^{(t)})$.

3: M-korak: Ažuriraj parametre $\Theta^{(t+1)} = \arg\max_\Theta Q(\Theta, \Theta^{(t)})$.

4: Izračunaj log-vjerodostojnost $\log p(\mathrm{X}|\Theta^{(t+1)})$ i provjeri konvergenciju algoritma.

until Kriterij konvergencije je zadovoljen

return Konačni parametri Θ.

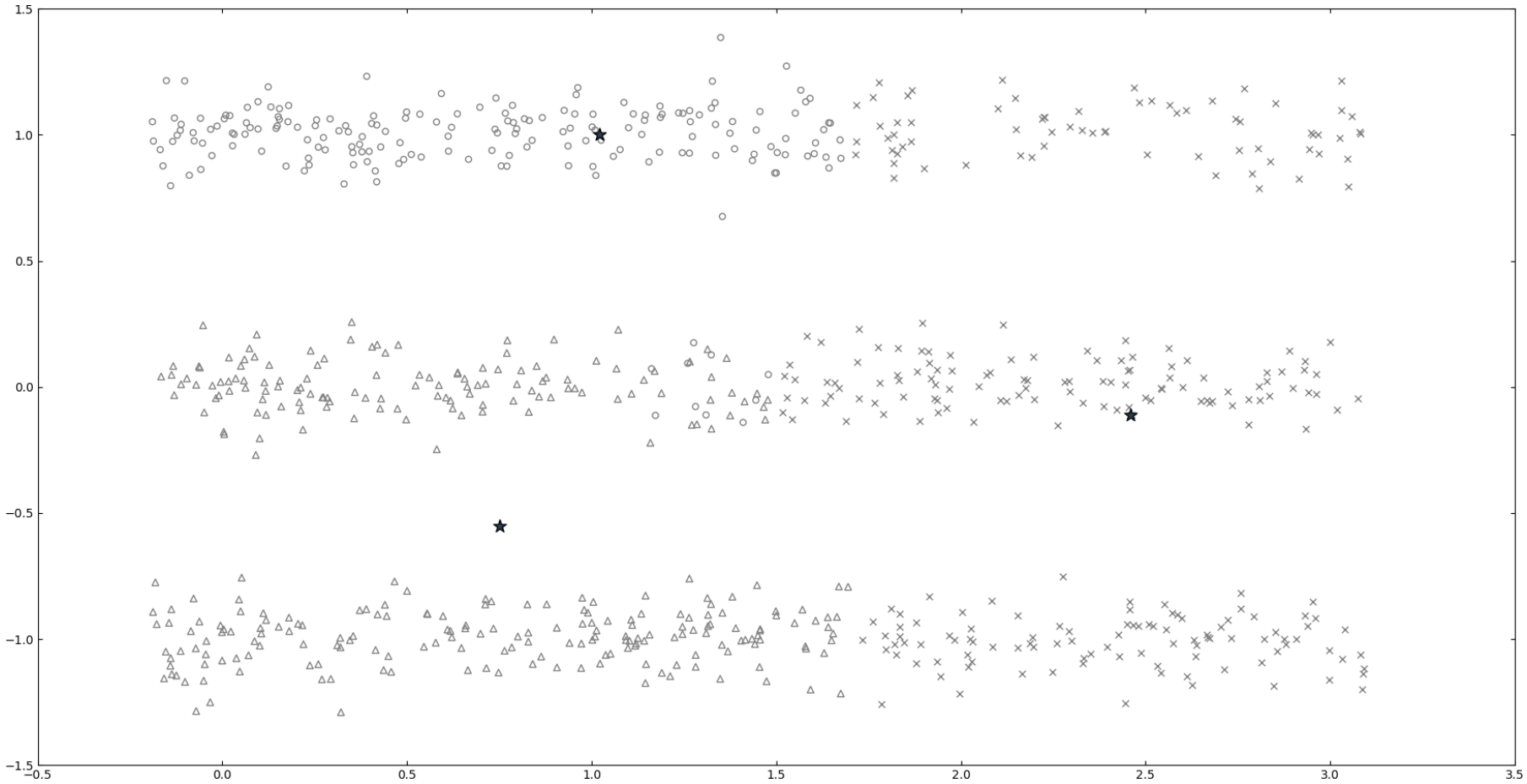
Algoritam maksimizacije očekivanja

- Algoritam maksimizacije očekivanja je jednostavan i ima garanciju monotonog rasta vjerodostojnosti skupa za treniranje tijekom optimizacije.
- Algoritam konvergira u lokalni maksimum, a njegovo rješenje snažno ovisi o inicijalizaciji. Zbog toga može proizvesti suboptimalne procjene maksimalne vjerodostojnosti.
- Jedna strategija za poboljšanje je korištenje višestrukih slučajnih početnih stanja iz kojih se iteriranjem izračuna konačna procjena vjerodostojnosti. Biramo procjenu s najvišom vrijednosti vjerodostojnosti.
- Druga strategija je koristiti klastering algoritam za inicijalizaciju.

Klasteriranje bazirano na gustoći

- Mnogi algoritmi klasteriranja pretpostavljaju da su podaci generirani iz točno zadane vjerojatnosne distribucije. Npr. mješavina normalnih distribucija i K-means pretpostavljaju normalnu distribuciju.
- Zbog pretpostavki, takvi algoritmi stvaraju sferne klastere i ne daju točne klastere na skupovim podataka gdje stvarni klasteri imaju ne sferični oblik.
- Ne sferični klasteri su česti kod prostornih podataka - podaci u dvije ili tri dimenzije vezani uz neki aspekt svijeta.
- Primjene uključuju geo-marketing (npr. pronalazak kuća sa zadanim karakteristikama), analiza kriminala (detektiranje žarišta) itd.
- Pretpostavke na broj ili oblik klastera mogu biti nevaljane čak i na visokodimenzionalnim podacima. U takvom slučaju će algoritmi kao npr. K-means razlamati ili spajati stvarne klastere što dovodi do netočnih rezultata.

Klasteriranje bazirano na gustoći



Klasteriranje bazirano na gustoći

- Uz nemogućnost particijskih i nekih vjerojatnosnih algoritama da ispravno identificiraju nepravilne klastere, javlja se problem skalabilnosti na velike baze, te problemi s detekcijom i otklanjanjem šuma i stršećih vrijednosti.
- Klasteriranje **bazirano na gustoći** je predloženo kao rješenje svih navedenih problema.
- Klasteriranje bazirano na gustoći je neparametarska metoda, ne radi pretpostavke na broj klastera i njihovu distribuciju.
- Klasteri bazirani na gustoći su **povezana, gusta područja** u prostoru podataka međusobno separirana rjeđim područjima.
- Pretpostavljamo da je gustoća šuma niža od gustoće bilo kojeg klastera. Klasteri mogu imati proizvoljan oblik.
- Algoritmi klasteriranja bazirani na gustoći moraju odgovoriti na nekoliko pitanja prilikom dizajna: a) kako procjeniti gustoću?, b) kako definirati povezanost?, c) koje podatkovne strukture omogućuju učinkovito implementiranje algoritma.

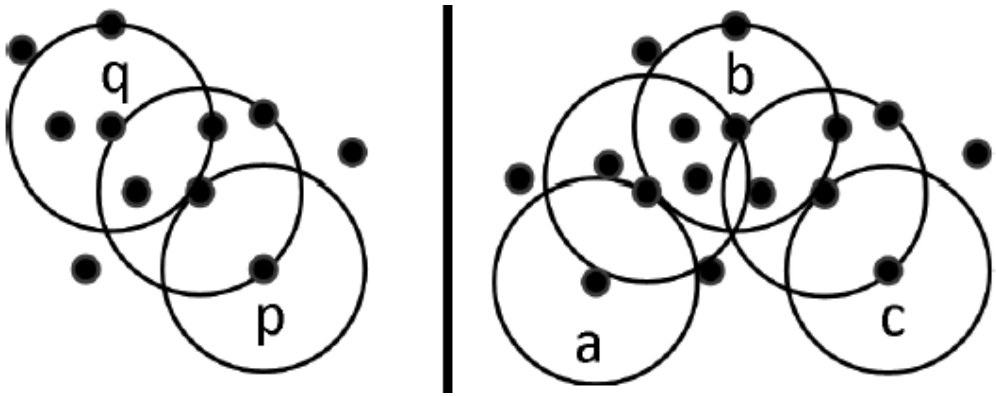
Klasteriranje bazirano na gustoći

- Algoritam DBSCAN je najpoznatiji algoritam klasteriranja baziran na gustoći, te jedan od najkorištenijih algoritama klasteriranja.
- DBSCAN procjenjuje gustoću brojanjem točaka u susjedstvu fiksnog radijusa. Dvije točke smatra povezanim ukoliko se nalaze unutar svojih međusobnih susjedstava.
- Točku nazivamo **jezgrenom točkom** ukoliko susjedstvo radijusa ε sadrži najmanje `MinT` točaka. Time osiguramo da gustoća točaka u susjedstvu prelazi predefinirani prag.
- Točka q je **direktno dohvatljiva s obzirom na gustoću** od strane jezgrene točke p ukoliko je q unutar ε susjedstva od p. Za zadanu mjeru udaljenosti $d(p, q)$, $SEps(p) = \{q \in D |\ d(p, q) \leq \varepsilon\}$.
- **Dohvatljivost s obzirom na gustoću** se definira kao tranzitivno zatvorenje direktne dohvatljivosti s obzirom na gustoću. Za niz točaka $p_1, \ldots, p_n$, gdje $p_1 = q$, a $p_n = p$, kažemo da je p dohvatljiva s obzirom na gustoću iz q, ako za svaki i, p_{i+1} je direktno dohvatljiv s obirom na gustoću od strane p_i.

Klasteriranje bazirano na gustoći

- Dvije točke p i q su **povezane s obzirom na gustoću** ukoliko postoji treća točka o takva da su i p i q dohvatljive s obzirom na gustoću iz o.
- Klaster je skup točaka koje su povezane s obzirom na gustoću i koji je maksimalan s obzirom na dohvatljivost s obzirom na gustoću. Za klaster C mora vrijediti:
  - $\forall p, q$, ako $p \in C$ i q je dohvatljiv s obzirom na gustoću iz p s obzirom na zadane parametre ε i $MinT$, tada $q \in C$ (**svojstvo maksimalnosti**).
  - $\forall p, q \in C$, p je povezan s obzirom na gustoću s q s obzirom na parametre ε i $MinT$ (**svojstvo povezanosti**).
- **Šum** definiramo kao skup točaka iz baze koje ne pripadaju niti jednom klasteru.
- Zadatak klasteriranja baziranog na gustoći je pronaći sve klastere s obzirom na parametre ε i $MinT$ u danoj bazi (skupu podataka).
- Kod klasteriranja baziranog na gustoći razlikujemo tri različita tipa točaka: a) **jezgrene točke** (točke sa susjedstvom $|SEps(p)| \geq MinT$, b) granične točke (točke koje pripadaju klasteru, međutim nemaju gusto susjedstvo), c) točke šuma (ne pripadaju niti jednom klasteru).

Klasteriranje bazirano na gustoći



- Točke q na slici lijevo i b na slici desno su jezgrene točke.
- Točke p na slici lijevo, te a i c na slici desno su granične točke.
- Važna činjenica koja omogućava učinkovito računanje klastera kod DBSCAN algoritma je da je svaka točka klastera C dohvatljiva s obzirom na gustoću od proizvoljne jezgrene točke klastera C.
- DBSCAN kreće od proizvoljne točke p iz baze podataka i dohvaća sve točke koje su dohvatljive s obzirom na gustoću iz p, prema parametrima ε i $MinT$. Dohvaćanje se provodi provodeći lokalizirane upite za p i potencijalno njegove direktne i indirektne susjede.

Klasteriranje bazirano na gustoći

- Ukoliko je p jezgrena točka, tada će pretraga pronaći klaster s obzirom na ε i $MinT$.
- Ako p nije jezgrena točka, ne postoje točke koje su dohvatljive s obzirom na gustoću iz p, stoga će p biti proglašen šumom. Ukoliko je p dohvatljiv s obzirom na gustoću iz neke druge jezgrene točke, bit će dodijeljen klasteru kojem pripada ta jezgrena točka, odnosno bit će rubna točka.
- Algoritam završava kada su sve točke dodijeljene klasteru ili proglašene šumom.
- DBSCAN koristi prostorni indeks, često implementiran strukturama kao što su R-stabla ili X-stabla. Navedena stabla omogućavaju učinkovito dohvaćanje točaka u ε susjedstvu zadane točke.
- Vremenska složenost u najgorem slučaju (za visokodimenzionalne podatke) je $\mathcal{O}(n^2)$, dok je vremenska složenost u slučaju podataka s malo atributa $\mathcal{O}(n \log n)$. Glavni razlog je povećanje složenosti indeksiranja u visokodimenzionalnim prostorima.

DBSCAN

Algoritam DBSCAN(SkupTocaka, ε, $MinT$)

```
KlasterId := sljedeciId(NOISE)
for i ← 1 to SkupTocaka.size do
    Tocka := SkupTocaka.get(i)
    if Tocka.KlId = UNCLASSIFIED then
        if ProsiriKlaster(SkupTocaka, Tocka, KlasterId, ε, MinT) then
            KlasterId := sljedeciId(KlasterId)
        end if
    end if
end for
```

Algoritam ProsiriKlaster(SkupTocaka, Tocka, KlId, ε, *MinT*) : Boolean

```
sjeme := SkupTocaka.prostorniUpit(Tocka, ε)
if sjeme.size < MinT then
    SkupTocaka.promijeniKlId(Tocka, NOISE)
    return false
else
    SkupTocaka.promijeniKlId(sjeme, KlId)
    sjeme.delete(Tocka)
    while sjeme ≠ ∅ do
        trenutnaT := sjeme.first()
        rezultat := SkupTocaka.prostorniUpit(trenutnaT, ε)
        if rezultat.size ≥ MinT then
            for i ← 1 to rezultat.size do
                rezultatT := rezultat.get(i)
                if rezultatT.KlId ∈ {UNCLASSIFIED, NOISE} then
                    if rezultatT.KlId = UNCLASSIFIED then
                        sjeme.append(rezultatT)
                    end if
                    SkupTocaka.promijeniKlId(rezultatT, KlId)
                end if
            end for
        end if
        sjeme.delete(trenutnaT)
    end while
    return true
end if
```