Primjena algoritama za grupisanje

Hasanbegović Selma
shasanbego1@etf.unsa.ba
17753/1574
Odsjek za telekomunikacije
Elektrotehnički fakultet
Sarajevo

Mahovac Nerman
nmahovac1@etf.unsa.ba
17919/1575
Odsjek za telekomunikacije
Elektrotehnički fakultet
Sarajevo

Velić Nejra
nvelic3@etf.unsa.ba
17313/1634
Odsjek za telekomunikacije
Elektrotehnički fakultet
Sarajevo

***Sažetak*— Obrada podataka igra veliku ulogu u mnogim granama nauke i industrije. Često je potrebno set podataka vezan za određenu oblast statistike razvrstati po određenom kriteriju. Grupisanje predstavlja podjelu skupa ulaznih podataka u grupe na osnovu njihove sličnosti. Svaka grupa sadrži podatke koji su u dovoljnoj mjeri slični jedni drugima, te dovoljno različiti od podataka u drugim grupama. Kroz ovaj rad ćemo se osvrnuti na neke od najpoznatijih algoritama za grupisanje kao što su: *k-means,* DBSCAN te hijerarhijski algoritam. Osvrnućemo se na neke od primjena navedenih algoritama na relevantnim skupovima podataka.**

Ključne riječi- algoritam, grupisanje, k-means, DBSCAN, hijerarhijski, sličnost, udaljenost

I. UVOD

Svakodnevno se generiše ogromna količina informacija koja se pohranjuje i obrađuje kao skup podataka za daljnju analizu. Jedan od glavnih ciljeva jeste kategorizacija prikupljenih informacija u određene skupove. Generalno, kategorizacija se može izvršiti na dva načina: nadgledanim načinom, te nenadgledanim. Pojam koji se veže uz nadgledanu kategorizaciju jeste klasifikacija podataka, koja se sprovodi na način da se sistemu pruže ulazni podaci koji su već svrstani u određene grupe, te se na osnovu sličnosti sa viđenim podacima, neoznačeni podaci grupišu u predefinisane kategorije. Drugi način jeste da se sistem u potpunosti oslanja samo na neoznačene podatke te iz njih dobije informaciju o broju grupa kao i međusobnoj relaciji među ulaznim podacima, i ova vrsta kategorizacije se naziva grupisanje (*clustering*).

Grupisanje razvrstava ulazne podatke u podskupove na način da su podaci koji su slični jedni drugima grupisani u jedan skup, dok se oni dovoljno različiti razvrstavaju u druge skupove. Primjene grupisanja variraju od matematike i statistike, pa do biologije i genetike. Pri tome, svaka grana nauke ima svoj određeni generički naziv za cijeli skup ulaznih podataka (npr. biološke taksonomije te genetički genotipi) [1]. Cilj grupisanja jeste razdvajanje neoznačenog skupa podataka u ograničen, diskretan skup „prirodnih“, skrivenih podatkovnih struktura. Nenadgledano kategorisanje je po svojoj prirodi subjektivan proces, te se oslanja na relativnu efikasnost tehnika grupisanja. Objekti se grupišu u manje ili više homogene strukture na osnovu, često subjektivno određene, mjere sličnosti. Što se tiče same grupe, generalno ne postoji opća definicija. Većina naučnika i istraživača opisuje grupu uzimajući u obzir internu homogenost te eksternu razdvojenost, to jeste uzorci unutar iste grupe bi trebali biti slični jedni drugima dok ne bi trebali biti slični uzorcima unutar druge grupe. Sličnost i razlika među uzorcima bi se trebale moći provjeriti jasnom i preciznom metodom [2].

Kako grupisanje zahtjeva da podaci u istoj grupi budu slični, potrebne su određene mjere kako bi se odredila sličnost (ili razlika) među uzorcima. Postoje dva glavna tipa mjera za utvrđivanje navedenih relacija i to: mjere udaljenosti i mjere sličnosti [1].

Najčešće korištene mjere udaljenosti su:

- Minkowski udaljenost data izrazom: $d(x_i, x_j) = (\sum_{j=1}^{p} |x_i - x_j|^g)^{\frac{1}{g}}$. Specijalan slučaj jeste Euklidska udaljenost za g=2, te Manhattan udaljenost za g=1. U slučaju da ne nose svi atributi istu vrijednost, istima se mogu dodijeliti težine, te tada Minkowski udaljenost postaje jednaka: $d(x_i, x_j) = (\sum_{j=1}^{p} w_j|x_i - x_j|^g)^{\frac{1}{g}}$.
- Mjera udaljenosti za binarne atribute: $d(x_i, x_j) = \frac{r+s}{q+r+s+t}$, gdje je q broj atributa koji su jednaki 1 za oba objekta, t broj atributa koji su jednaki nula za oba objekta, a s i r broj atributa koji su nejednaki za oba objekta.
- Mjera udaljenosti za nominalne atribute: $d(x_i, x_j) = \frac{p-m}{p}$, gdje je p ukupan broj atributa, a m broj poklapanja.

Kao alternativa mjerama udaljenosti, mogu se koristiti mjere sličnosti kako bi se uporedila dva vektora. Funkcija sličnosti bi u pravilu trebala davati velike vrijednosti kada su dva vektora veoma slična i maksimalne vrijednosti kada se porede dva identična vektora. Neke od funkcija sličnosti su:

- Kosinusna mjera: $s(x_i, x_j) = \frac{x_i^T \cdot x_j}{||x_i|| \cdot ||x_j||}$, gdje je ugao između dva vektora mjera njihove sličnost.
- Pearson korelacijska mjera: $s(x_i, x_j) = \frac{(x_i - \bar{x}_i)^T \cdot (x_j - \bar{x}_j)}{||x_i - \bar{x}_i|| \cdot ||x_j - \bar{x}_j||}$, gdje $\bar{x}_i$ označava prosjek x preko svih dimenzija.
- Proširena Jaccard mjera: $s(x_i, x_j) = \frac{x_i^T \cdot x_j}{||x_i||^2 + ||x_j||^2 - x_i^T \cdot x_j}$

Također, korisno je nakon grupisanja, procijeniti koliko su zapravo dobri i ispravni dobijeni klasteri. Iako je u najčešćem

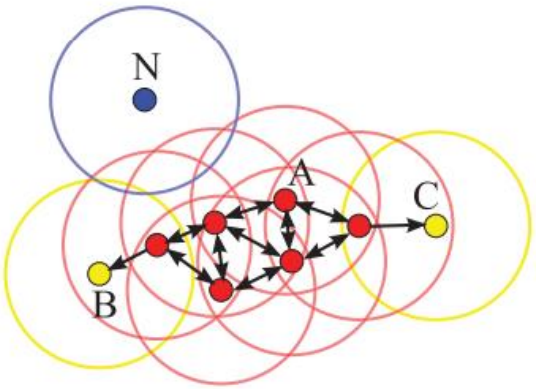
slučaju ocjena kvalitete grupisanja subjektivna, postoje određene evaluacijske mjere, koje se dijele na interne (gdje se ocjenjuje kompaktnost klastera) i eksterne. Neki od njih su suma kvadratne greške (SSE), kriterij raspršenosti, C-kriterij, mjera bazirana na zajedničkim informacijama, rand indeks i slične.
U nastavku ćemo opisati neke od najpoznatijih i najprimjenjenijih algoritama za grupisanje.

II. DBSCAN Algoritam

Prilikom baratanja sa klasterima različite gustoće, veličine i oblika, često je teško odrediti grupu tačaka koji pripadaju jednom klasteru. Taj zadatak može biti još teže izvršiti u slučaju da ulazni set podataka sadrži šum i *outlier*-e. U originalnom radu [3] Martin Ester i koautori su predstavili *Density Based Clustering of Applications with Noise* (DBSCAN). Prema njima, tri glavna razloga za korištenje ovog algoritma su [4]:

1. Zahtijeva minimalno domensko znanje
2. Može otkriti klastere proizvoljnog oblika
3. Efikasan za velike baze podataka (kompleksnost O(n*log(n)))

Model uveden u DBSCAN koristi estimaciju minimalne gustoće, bazirane na pragu broja susjednih tačaka, minPts, unutar nekog radijusa ε (sa proizvoljnom mjerom udaljenosti). Objekti sa više od minPts tačaka unutar radijusa (uključujući tačku pretrage), nazivaju se jezgrene tačke. Intuicija ovog algoritma jeste da se pronađu područja visoke gustoće, razdvojena područjima niske gustoće. Svi susjedi jezgrene tačke se smatraju tačkama istog klastera kao i jezgrena tačka (navedeno se naziva „direct density reachability"). Ukoliko je neki od susjeda jezgrena tačka u idućoj iteraciji, njihova susjedstva postaju dio istog klastera i to se naziva „*density reachability*". Ne-jezgrene tačke se nazivaju granične tačke i sve tačke unutar tog seta su „*density connected*". Tačke koje nisu „*density reachable*" ni iz jednog klastera se nazivaju šum [5]. Slika 1 ilustrira koncepte DBSCAN-a. Parametar minPts je 4, radijus je predstavljen krugovima, N je šum, A je jezgrena tačka a B i C su granične tačke. Strelice indiciraju „*direct density reachability*". Tačke B i C su „*density connected*", jer su obe tačke „*density reachable*" iz A. Tačka N nije „*density reachable*", te je stoga označena kao šum.

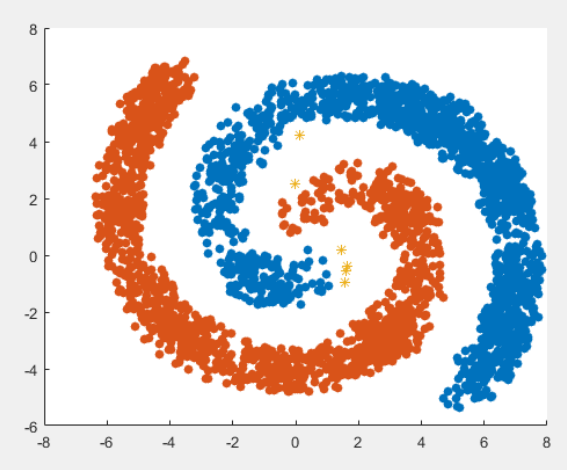


Slika 1.

Jedna od zanimljivih varijanti predstavljena u [6], posmatra ST-DBSCAN (engl. „*Spatial-temporal*"- prostorno-vremenski) gdje tačke za razliku od originalnog DBSCAN-a ne pripadaju samo nekom n-dimenzionalnom prostoru, već se dodatno razdvajaju u nekoj vremenskoj osi. Naprimjer, u vremenskoj domeni dvije susjedne tačke mogu biti dva dana zaredom, ili isti dan u dvije godine zaredom.
Kako bi se pronašao klaster, algoritam počinje sa proizvoljnom tačkom p, te prikuplja sve tačke koje su „*density reachable*" iz p (uz naravno definisane ε i minPts). Ukoliko je p jezgrena tačka formira se klaster, dok u slučaju da je p rubna tačka, ne postoje „*density reachable*" tačke iz p te algoritam prelazi na sljedeću tačku. U nastavku su prikazani rezultati eksperimenta za tri različita skupa podataka, gdje ćemo pokazati situacije u kojima je poželjno koristiti DBSCAN. Također, osvrnućemo se i na važnost odabira parametara samog algoritma. Za sve vrste podataka, parametar minPts je bio postavljen na 4, dok se parametar Eps mijenjao u zavisnosti od vrste podataka.

A. *Slučaj: Dvije spirale*

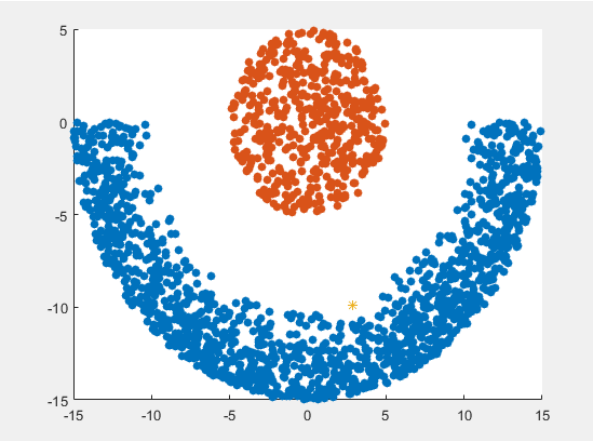
U ovom slučaju, parametar Eps je postavljen na vrijednost 0.5 iz razloga što su ulazni podaci gusto zbijeni, te bi dalje povećanje navedenog parametra prouzrokovalo i potencijalno spajanje klastera što bi predstavljalo neželjenu pojavu. Druga strana navedenog pristupa jeste pojava *outlier*-a, koji nemaju pripadnost niti jednom klasteru. U ovom slučaju njihova pojava predstavlja svojevrsno vaganje koristi i štete, iz razloga što bi u slučaju da nisu klasificirani na ovaj način, mogli uzrokovati spajanje klastera.



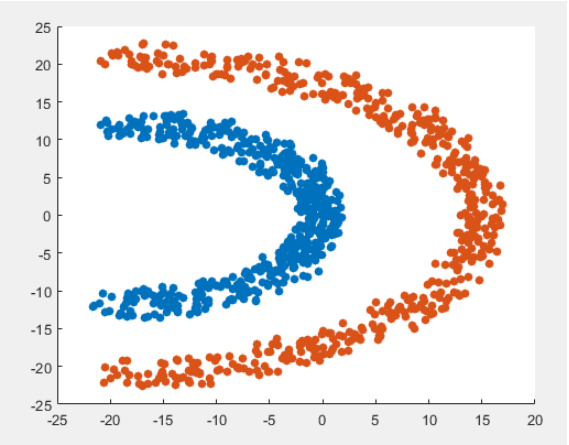
Slika 2. Klasteri raspoređeni u dvije spirale

B. *Slučaj:Polumjesec*

Kako su u ovom slučaju, klasteri dovoljno razmaknuti, korištena je vrijednost Eps=1, iako je mogla biti korištena i veća. Ako pogledamo grafik, možemo primjetiti kako su u ovom slučaju pojedine tačke u donjem dijelu raspršene i nalaze se nešto dalje od ostatka klastera iako mu pripadaju.



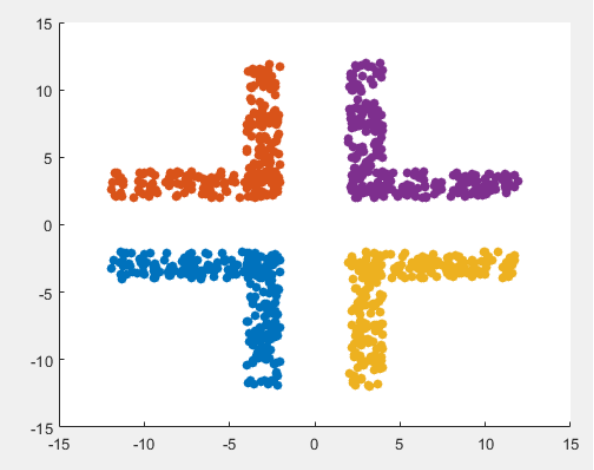
Slika 3. Polumjesec



Slika 4. Polukernel

C. *Slučaj: Ulica*

I u ovom slučaju je korištena vrijednost Eps=1. Kao što vidimo, pojavljuju se veliki procjepi u ulaznim podacima, te bi u slučaju npr. Eps=0.5, imali mnogo manjih klastera, iako oni logički pripadaju istom klasteru.



Slika 4. Ulica

D. *Slučaj: Half-kernel*

U ovom slučaju imamo rijetko raspoređene podatke u vanjskom obodu. Stoga, izabrana je vrijednost Eps=2. Vrijedi ista konstatacija kao i za prethodni slučaj, s tom razlikom što bi se pojavilo nebrojeno mnogo klastera koji vizuelno pripadaju jednom. U prethodnom slučaju bi potencijalno rješenje moglo biti i smanjenje broja tačaka koji se moraju pojaviti u Eps-okolini da bi bile povezane. Ovde to nije slučaj jer su tačke toliko raspršene, da ih jedino možemo spojiti povećanjem parametra Eps.

III. K-MEANS CLUSTERING ALGORITAM

Najjednostavniji oblik grupisanja je parcionisano grupisanje. Ovaj tip grupisanja ima za cilj podjelu datog skupa podataka u podsetove (klastere) tako da je svaki klaster optimiziran po nekom kriteriju. Najčešće korišten kriterij za klasterizaciju je kriterij greške. On za svaku tačku računa njegovu kvadratnu udaljenost od korespondirajućeg centra klastera te uzima sumu udaljenosti za sve tačke u jednom skupu podataka. Popularni metod koji minimizira ovu grešku je *k-means clustering* algoritam [6].

K-means algoritam grupisanja spade u grupu algoritama nenadgledanog učenja. Kod ovog tipa učenja ne postoji informacija o željenim izlaznim vrijednostima za date ulaze. U ovom radu, najprije je data teorijska analiza problema, kao i primjer primjene ovog mehanizma u programskom jeziku MATLAB.

K-means algoritam grupiše neoznačene setove podataka u različite klastere. *K* ovdje definiše broj pre-definisanih klastera koji treba da budu procesirani. Ako je K=2, tada postoje dva klastera, za K=3 postoje tri klastera itd. Moguće je podijeliti podatke u različite grupe te zaključiti kojim kategorijama pripadaju grupisani, neoznačeni podaci bez potrebe za treningom.

Alogritam je zasnovan na centroidu, gdje se svaki klaster povezuje sa jednim centroidom. Glavni cilj algoritma je tada minimizirati sumu udaljenosti između svih tački i korespondirajućeg centroida. On uzima neoznačene setove podataka kao ulaz, dijeli ih u *K* različitih grupa te ponavlja ovaj process dok ne nađe najbolje klastere [7].

K-means algoritam grupisanja obavlja dva zadatka:

- Iterativnim procesom procijeni najbolju vrijednost za *k* centralnih tačaka.
- Svim ostalim tačkama dodijeli najbliži *k* centar. Tada svi podaci dodijeljeni jednom centru kreiraju jedan klaster.

Sam algoritam može biti pokazan u nekoliko koraka, navedenih u nastavku.

Neka postoji sljedećih osam tačaka prikazanih na slici 1, te je na njih potrebno primijeniti opisani algoritam. Algoritam se može sprovesti kroz sljedeće korake [8]:

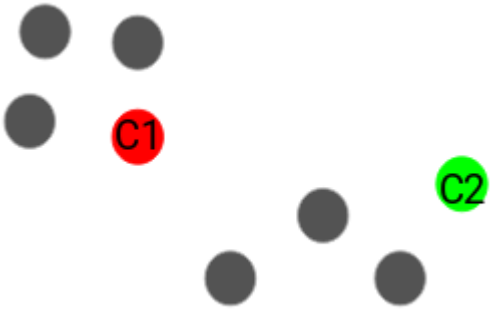
A. Odrediti K:

Slika 5.

Prvi korak u primjeni ovog algoritma jeste odrediti broj klastera koje treba primijeniti na dati set podataka. U ovom slučaju K=2.

B. Odrediti centroide

Odabir centroda je slučajan za svaki klaster. U ovom slučaju postoje dva klastera, tako da se za njih biraju dva slučajna centroida, kao na slici 6.

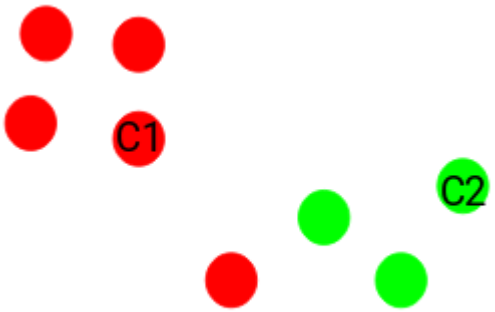


Slika 6.

Crveni i zeleni krug predstavljaju centroide za dva klastera.

C. Dodjeljivanje ostalih tačaka najbližim centroidima

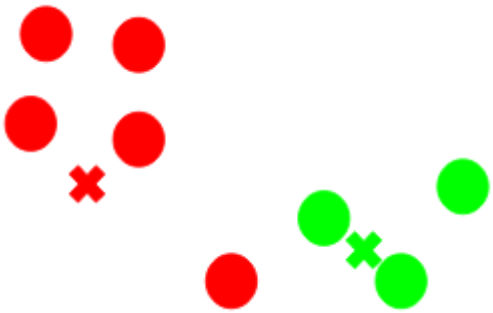
Nakon što se odrede centroidi (korak 2), svakom od njih se dodjeljuje najbliža tačka. Na slici 7 vidimo da tačke bliže crvenom centroidu su dodijeljenje njegovom klasteru, dok su one bliže zeleonom, dodijenjene njemu.



Slika 7.

D. Računanje centroida

Sada je potrebno izračunati centride novoformiranih klastera, jer su prvi izabrani slučajno. Novi centroidi su prikazani na slici 8, označeni znakom X.



Slika 8.

E. Ponavljati korake C i D:

Ponavljanjem koraka 3 i 4 dobijemo slučaj na slici 9.

Slika 9.

Ovakav algoritam, kada je pokrenut, može izvršavati iteracije beskonačno. Zbog toga je potrebno moći ga zaustaviti. Postoje tri kriterija za zaustavljanje rada algoritma [8]:

1. Centroidi novoformiranih klastera se ne mijenjaju – ako se nakon nekoliko iteracija uvijek dobiju isti centroidi za sve klastere, možemo reći da algoritam ne uči nikakav novi patern te zaustavljamo treniranje.
2. Tačke ostaju u istom klasteru – ako nakon više iteracija tačka ostaje u istom klasteru, potrebno je zaustaviti treniranje.
3. Dosegnut je maksimalan broj iteracija – ako postoji set od 100 iteracija, proces će se ponavljati dok ne dosegne taj broj iteracija i tada se zaustavlja.

Još jedno pitanje je ostalo neodgovoreno, a to je način na koji se bira broj klastera *k*.

Performanse razmatranog algoritma uveliko zavise od efikasnosti klastera koje formira. Postoje razni načini za određivanje klastera, no najjednostavniji je određivanjem broja klastera *k*. Najpoznatija metoda za rješavanje ovog problema je metoda lakta (engl. *Elbow method*) [2].

Ova metoda koristi concept WCSS (engl. *Within Cluster Sum of Squares*) vrijednosti, koja definira ukupan broj varijacija unutar klastera. Formula za računanje WCSS (za tri klastera) je:

$$WCSS = \sum_{P_i u\ klasteru\ 1} udaljenost\ (P_i, C_1)^2 + \sum_{P_i u\ klasteru\ 2} udaljenost\ (P_i, C_2)^2 + \sum_{P_i u\ klasteru\ 3} udaljenost\ (P_i, C_3)^2$$

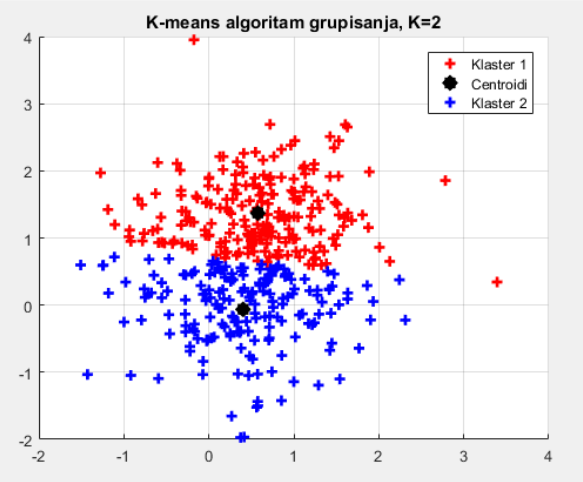
U navedenoj formuli:

$\sum_{P_i u\ klasteru\ 1} udaljenost\ (P_i, C_1)^2$ - predstavlja sumu kvadrata udaljenosit između svake tačke i njegovih centroida u klasteru 1. Isto je I za ostale klastere. Za računanje ovih udaljenosti najčešće se koristi Euklidska udaljenost.

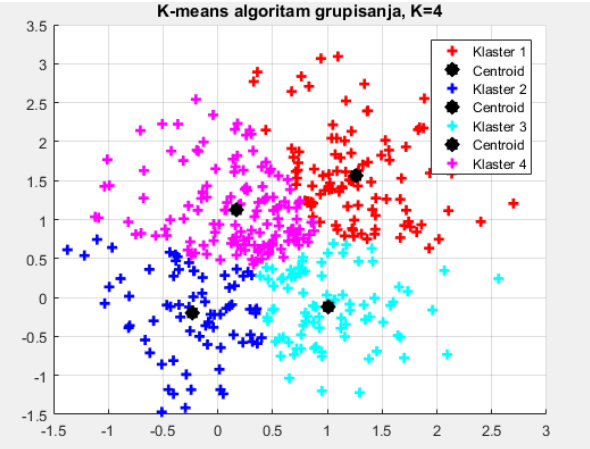
Da bi pronašao optimalnu vrijednost klastera, metoda lakta slijedi nekoliko koraka [7]:

1. Izvršava algoitam *k-means* klasterizacije za dati set podataka, za različite vrijednosti *k*, najčešće u opsegu (1,10).
2. Za svaku vrijednost *k* računa WCSS vrijednost.
3. Crta krivu koja prikazuje odnos između izračunate vrijednosti WCSS i broja *k*.
4. Oblik krive podsjeća na ruku (lakat), te tačka koja se nalazi "na laktu" predstavlja najbolju vrijednost *k*.

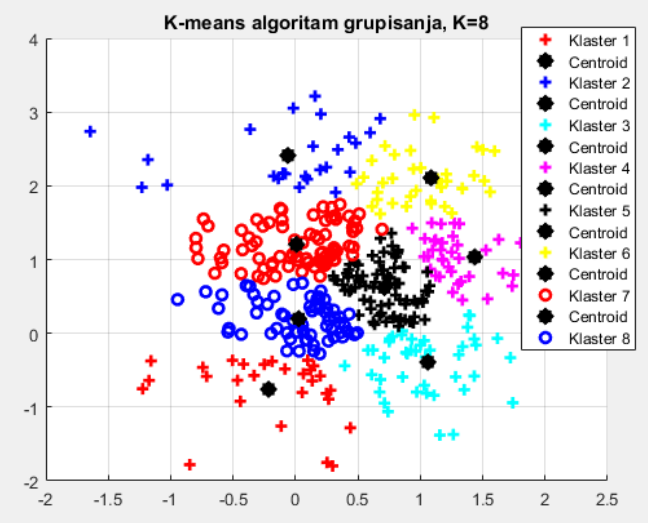
Praktična primjena navedenog algoritma je prikazana u programskom okruženju MATLAB. Prikaz *k-means* algoritma prikazan je na sljedeće tri slike, pri čemu je broj klastera mijenjan za isti set ulaznih podataka. Slika 10 prikazuje grupisanje u 2 klastera, slika 11 prikazuje grupisanje seta podataka u 4 klastera, dok je na slici 12 prikazano grupisanje u 8 klastera.



Slika 10. *K-means* algoritam grupisanja, 2 klastera



Slika 11. *K-means* algoritam grupisanja, 4 klastera

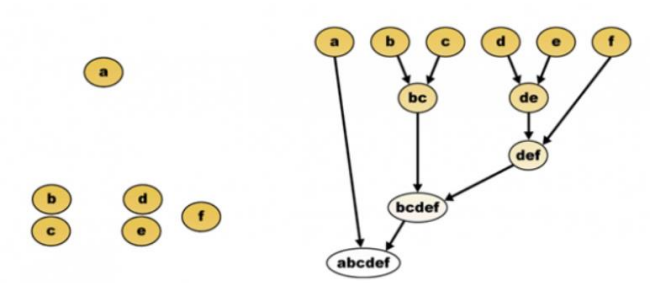


Slika 12. *K-means* algoritam grupisanja, 8 klastera

IV. AGLOMERATIVNO HIJERARHIJSKO GRUPISANJE

Algoritam za hijerarhijsko grupisanje predstavlja zasebnu vrstu nenadgledanog mašinskog učenja koji, za razliku od particijskog grupisanja, rezultuje hijerarhijom grupa. Algoritam se zasniva na rekurzivnom traženju podgrupa unutar grupa, kreirajući time hijerarhijsku strukturu unutar grupa. Hijerarhija grupa se može prikazati dendrogramom,

stablom kod kojeg listovi predstavljaju primjere, a linije odgovaraju povezivanjima na određenoj udaljeosti. Presijecanjem dendrograma na željenoj udaljenosti se može dobiti isti rezultat koji bi se dobio particijskim grupisanjem da datoj udaljenosti.
Poseban slučaj hijerarhijskog grupisanja, a koji se najčešće i koristi, jeste aglomerativno hijerarhijsko grupisanje (engl. *Hierarchical agglomerative clustering*, HAC). Algoritam prati *'bottom-up'* pristup, a započinje tako da se svaki primjer nalazi u zasepnoj grupi, a zatim u svakom koraku se stapaju dvije najsličnije grupe, po određenom kriteriju, sve dok svi primjeri ne budu unutar jednog klastera/grupe ili dok se ne dosegne unaprijed zadani broj grupa K [9]. Za K=1 algoritam će rezultirati potpunim dendrogramom koji se naknadno može presijecati na željenim udaljenostima.



Slika 13. Primjer korištenja aglomerativnog hijerarhijskog algoritma za grupisanje

Algoritam se može jednostavno objasniti pomoću slike 13 [10].

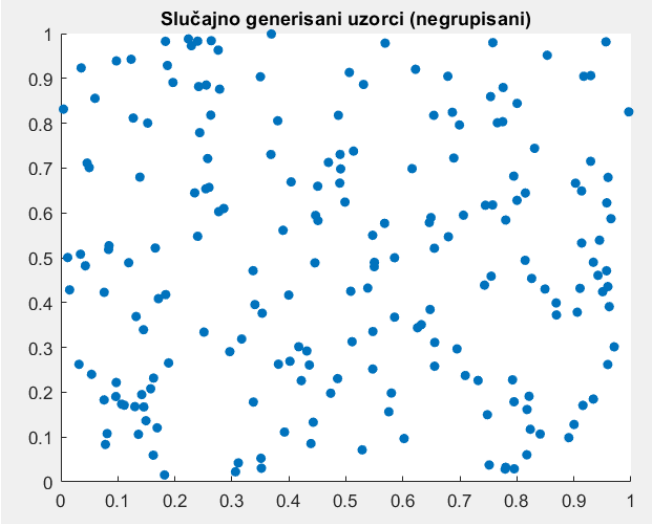
1. Svaki primjer a-f čini zaseban klaster
2. Formiraju se klasteri od po dva najbliža primjera, b/c i d/e
3. Klaster f se spaja sa najbližim klasterom d/e, formirajući novi klaster
4. Klasteri b/c i d/e/f su najbliži klasteri te se spajaju u jedan, novi klaster
5. Konačno, klaster a se spaja sa klasterom iz koraka 4 formirajući jedan zajednički, korijenski klaster

Hijerarhijsko aglometrijsko grupisanje se vrši na osnovu funkcije udaljenosti ili mjere sličnosti s ciljem da se pronađu grupe primjera koji su najsličniji jedan drugome. Kao metrike udaljenosti najčešće se koriste Minkowski, Euklidska, Manhattan, Chebyshev, kvadratna Euklidska udaljenost i slično, a u ovom radu će biti korištena Euklidska udaljenost. Postoji više načina za određivanje udaljenosti između grupa, G_i i G_j, koje sadrže jedan ili više primjera [11]:

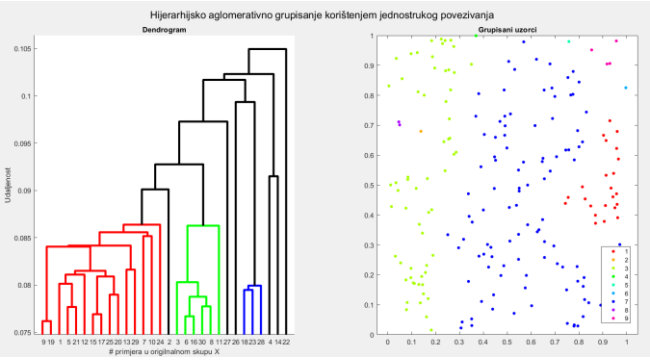
1. Jednostruko povezivanje (engl. *single linkage*) - proračun najmanje udaljenosti između pojedinačnih primjera u tim grupama. Još se naziva i grupisanje metodom najbližeg susjeda.
$$d_{min}(G_i, G_j) = \min d(\boldsymbol{x}, \boldsymbol{x}') \quad , \quad \boldsymbol{x} \in G_i, \boldsymbol{x}' \in G_j$$
2. Potpuno povezivanje (engl. *complete linkage*) - proračun najveće udaljenosti između pojedinačnih primjera u tim grupama. Još se naziva i grupisanje metodom najdaljeg susjeda.
$$d_{max}(G_i, G_j) = \max d(\boldsymbol{x}, \boldsymbol{x}') \quad , \quad \boldsymbol{x} \in G_i, \boldsymbol{x}' \in G_j$$
3. Prosječno povezivanje (engl. *average linkage*) - koristi se kada su grupe primjera podložne šumu, na koji su izuzetno osjetljivi jednostruko i poptpuno povezivanje.
$$d_{avg}(G_i, G_j) = \frac{1}{N_i N_j} \sum_{\boldsymbol{x} \in G_i} \sum_{\boldsymbol{x}' \in G_j} d(\boldsymbol{x}, \boldsymbol{x}')$$
gdje je N_i odnosno N_j broj primjera u grupi G_i odnosno G_j.
4. Ward metoda – proračun stope grupisanja u cilju minimiziranja povećanja EES (engl. *error sum of squares*) nakon spajanja dvije grupe u jednu. Koristi se kada postoji šum između klastera.
$$d_{ward}(G_i, G_j) = \frac{N_i N_j}{N_i + N_j} \|\bar{\boldsymbol{x}} - \bar{\boldsymbol{x}}'\|^2$$
pri čemu $\|\cdot\|$ predstavlja Euklidsku normu [12].

Prednost aglomerativnog hijerarhijskog algoritma koja je i razlog njegovog čestog korištenja jeste činjenica da nije potrebno unaprijed znati broj klastera. Međutim, kada se donese odluka o tome koja dva klastera će se spojiti, ne može se poništiti. Također, ovaj algoritam je primjenljiv samo na manje skupine podataka zbog svoje vremenske ($O(N^3)$) i prostorne ($O(N^2)$) kompleksnosti [13].

Za implementaciju aglomerativnog hijerarhijskog algoritma je korišteno MATLAB okruženje. Korišteno je 200 primjera, predstavljenih 2D koordinatama, koje je potrebno grupisati. Kao metrika udaljenosti korištena je Euklidska metrika, dok je kao kriterij povezivanja korišteno jednostruko, potpuno povezivanje i povezivanje Ward metodom. Za svaki od navedenih scenarija je prikazan skup grupisanih podataka, kao i odgovarajući dendrogram. Na slici 14 je prikazan skup negrupisanih primjera koji su slučajno generisani.

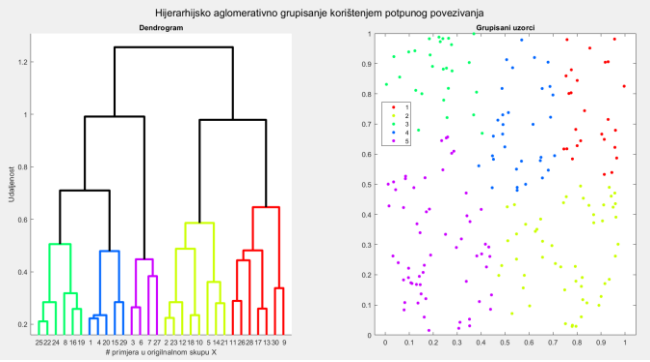


Slika 14. Prikaz slučajno generisanih uzoraka (negrupisani)



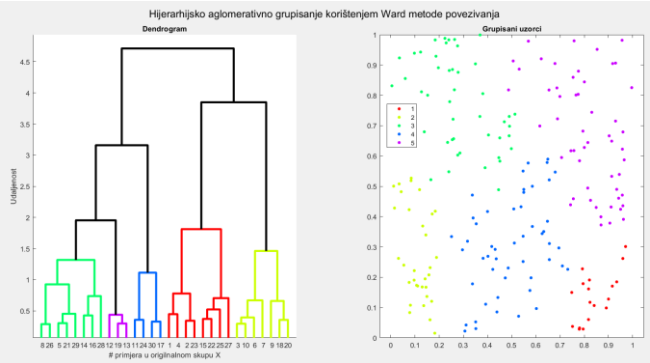
Slika 15. HAC algoritam sa jednostrukim povezivanjem (9 grupa)

Na slici 15 je prikazan dendrogram i grupisani uzorci u 9 grupa dobiveni korištenjem jednostrukog povezivanja. Može se uočiti da su primjeri grupisani uglavnom unutar jedne grupe.



Slika 16. HAC algoritam sa potpunim povezivanjem (5 grupa)

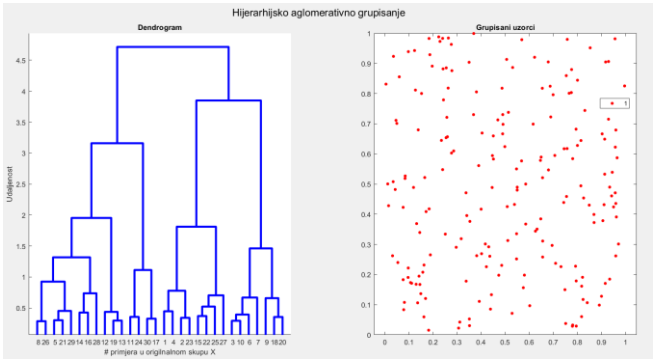
Slika 16 prikazuje dendrogram i grupisane primjere u 5 grupa dobivene korištenjem HAC algoritma sa potpunim povezivanjem. Uočava se da metoda sa potpunim povezivanjem daj poprilično dobre rezultate iako primjeri nisu adekvatno grupisani u potpunosti.



Slika 17. : HAC algoritam sa Ward metodom povezivanja (5 grupa)

Slika 17 prikazuje dendrogram i grupisane uzorke primjenom Ward metode povezivanja. Sa dijagrama klastera se može uočiti da su primjeri grupisani u 5 klastera na osnovu njihove sličnosti, a nije osjetljiv na postojanje šuma.

Na slici 18 prikazan je krajnji rezultat hijerarhijskog aglomerativnog grupisanja, gdje se uočava da je dendrogram potpun, to jeste da su sve grane međusobno povezane. Tačnije, svi primjeri su međusobno grupisani u jednu jedinstvenu grupu, što se može vidjeti i na dijagramu klastera. Ovakav potpuni dendrogram se može dalje presijecati na željenim udaljenostima u svrhu grupisanja primjera u više klastera.



Slika 18: HAC algoritam sa svim primjerima povezanim u jednu grupu

ZAKLJUČAK

Jedan od zadataka nenadgledanog učenja jeste grupisanje podataka, koje predstavlja postupak razdvajanja primjera u grupe (klastere) na osnovu njhove sličnosti. Svrha grupisanja jeste pronalaženje 'prirodnih' grupa u skupu neoznačenih podataka, a koristi se i kao pretprocesiranje za nadzorno učenje.

U radu smo se osvrnuli na neke primjenu nekih karakterističnih algoritama za grupisanje podataka kao što su *k-means*, DBSCAN i hijerarhijski aglometrijski algoritam.

DBSCAN algoritam prikazuje mogućnost grupisanja uzoraka koji oblikuju nekonvencionalne klastere. Ukazali smo na spektar primjene ovih algoritama te smo pokazali važnost adekvatne procjene skupa ulaznih podataka. Kvalitet grupisanja korištenjem posmatranog algoritma umnogome zavisi od gustine i rasporeda ulaznih podataka. Stoga je potrebno odabrati parametre Eps i minPts na takav način da istovremeno obuhvatimo klastere na pravi način, te izbjegnemo grupisanje outlier-a.

K-means tip grupisanja ima za cilj podjelu datog skupa podataka u podsetove (klastere) tako da je svaki klaster optimiziran po nekom kriteriju. algoritam grupiše neoznačene setove podataka u različite klastere. K ovdje definiše broj pre-definisanih klastera koji treba da budu procesirani. Veoma koristan u slučaju kada dovoljno poznajemo ulazni skup podataka. Predstavlja jednostavno implementacijsko rješenje, veoma efikasno za manji skup ulaznih podataka, međutim za large data-sets je potrebno razmotriti drugačija rješenja.

U radu je također obrađen hijerarhijski aglomerativni algoritam grupisanja sa jednostrukim, potpunim i Ward povezivanjem. HAC algoritam sa pojedinačnim povezivanjem rezultira grupisanjem većine uzoraka u jednu grupu. Navedena metoda ne rezultuje adekvatnim grupisanjem jer se grupisanje vrši na osnovu najbližih primjera iz dvije grupe, čak i ukoliko je samo jedan primjer blizu drugoj grupi. Rezultati dobiveni korištenjem potpunog povezivanja i povezivanja korištenjem Ward metode su gotovo isti, naročito kada su klasteri dobro odvojeni.

Međutim, Ward metoda daje malo bolje rezultate u slučajevima gdje dolazi do preklapanja klastera. Aglomerativni hijerarhijski algoritam grupisanja je najbolje koristiti prilikom traženja manjih grupa, a kako krajnji rezultat je dat u obliku dendrograma, vizuelizacija kreiranih grupa je jednostavna.

REFERENCE

[1] O. Maimon, L. Rokach (eds.), Data Mining and Knowledge Discovery Handbook, 2nd ed., (Springer Science+Business Media, LLC 2010)

[2] Xu, R., WunschII, D., *Survey of Clustering Algorithms* (IEEE Transactions on Neural Networks, 2005)

[3] Martin Ester, Hans-Peter Kriegel, Jiirg Sander, Xiaowei Xu, *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise* (University of Munich, 1996)

[4] https://towardsdatascience.com/dbscan-algorithm-complete-guide-and-application-with-python-scikit-learn-d690cbae4c5d

[5] Ester et.al., DBSCAN Revisited, Revisited: *Why and How You Should (Still) Use DBSCAN* (ACM Transactions on Database Systems, 1–21)

[6] Likas, A., Vlassis, N., J. Verbeek, J, *The global k-means clustering algorithm (Pattern Recognition*, 451–461, 2003)

[7] JavaTpoint. K-Means Clustering Algorithm. [Online] https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning

[8] Sharma, P. *The Most Comprehensive Guideto K-Means Clustering You'll Even Need* [Online] https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

[9] Sasirekha, K., and P. Baby. *Agglomerative hierarchical clustering algorithm- A Review*. International Journal of Scientific and Research Publications 83 (2013): 83.

[10] Minitab. *Cluster Analysis Tips* [Online] https://blog.minitab.com/en/quality-data-analysis-and-statistics/cluster-analysis-tips

[11] C. R. Patlolla. *Understanding the concept of Hierarchical clustering Technique* [Online] https://towardsdatascience.com/understanding-the-concept-of-hierarchical-clustering-technique-c6e8243758ec

[12] Vijaya, Sharma, S., Batra, N., *Comparative Study of Single Linkage, Complete Linkage, and Ward Method of Agglomerative Clustering* (International Conference on Machine Learning, Big Data, Cloud and Parallel Computing, 2019)

[13] Bouguettaya, A., Yu, Q., Liu, X., Zhou, X., & Song, A., *Efficient agglomerative hierarchical clustering* (Expert Systems with Applications, 2785–2797, 2015)