SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU ELEKTROTEHNIČKI FAKULTET

Diplomski studij računarstva

Seminarski rad

Grupiranje podataka

Dumančić Andrija, DRB

Osijek,15.09.2016

Sadržaj

1. Uvod 3
2. Metode grupiranja podataka 4
3. K-means (K srednjih vrijednosti) 10
3.1. Primjer pseudo koda K-means algoritma 11
3.2. Određivanje parametra modela tj. centroida 12
3.3 Primjer K-means algoritma 14
3.4. Specifičnosti K-means algoritma 17
4. Zaključak 20
5. Literatura 21

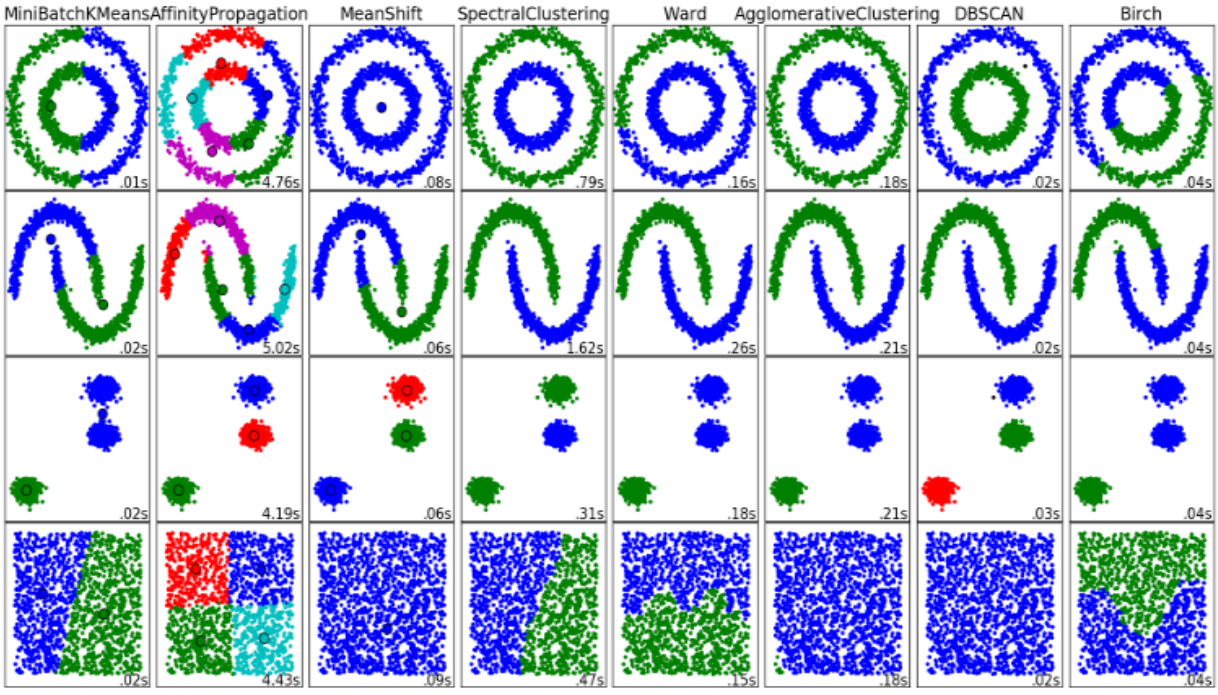
1. Uvod

Grupiranje podataka odnosno klasteriranje (engl. *clustering*) je zadatak kojim je potrebno grupirati podatke na način da su objekti istoj grupi (klasteri) više slični međusobno nego s onima iz druge skupine. To je glavna zadaća grupiranja podataka, a uobičajena tehnika u raznim primjenama. Tehnika se koristi u mnogim područjima poput strojnog učenja, raspoznavanja uzoraka, analize slike, pronalaženja informacija, bioinformatike, kompresije podataka te računalne grafike.Grupiranje nije specifičan algoritam, ali je zadatak koji treba riješiti. Rješenje se može pomoću različitih algoritama koji se znatno razlikuju u njihovom pojmu što čini jednu skupinu te na način kako ih učinkovito pronaći. Klasteri su skupine s malim udaljenostima među članovima (podacima) klastera, gusta područja prostora podataka, intervalne ili pojedine statističke distribucije. Grupiranje se može formulirati u obliku multi-objektivnog problema optimizacije. Klaster analiza je stoga najčešće iterativni proces u kojem se nastoji optimirati dani. Često je potrebno prije samog procesa grupiranja podataka provesti predobradbu raspoloživih podataka, a i vrijednosti različitih parametara postupka grananja je često potrebno mijenjati kako bi se postigao željeni rezultat.

2. Metode grupiranja podataka

Postoje razne metode grupiranja[1] poput:

- K-Means [2]
- Affinity propagation[3]
- Mean-shift [4]
- Spectral clustering [5]
- Ward hierarchical clustering [6]
- Agglomerative clustering
- DBSCAN [7]
- Gaussian mixtures
- Birch [8]



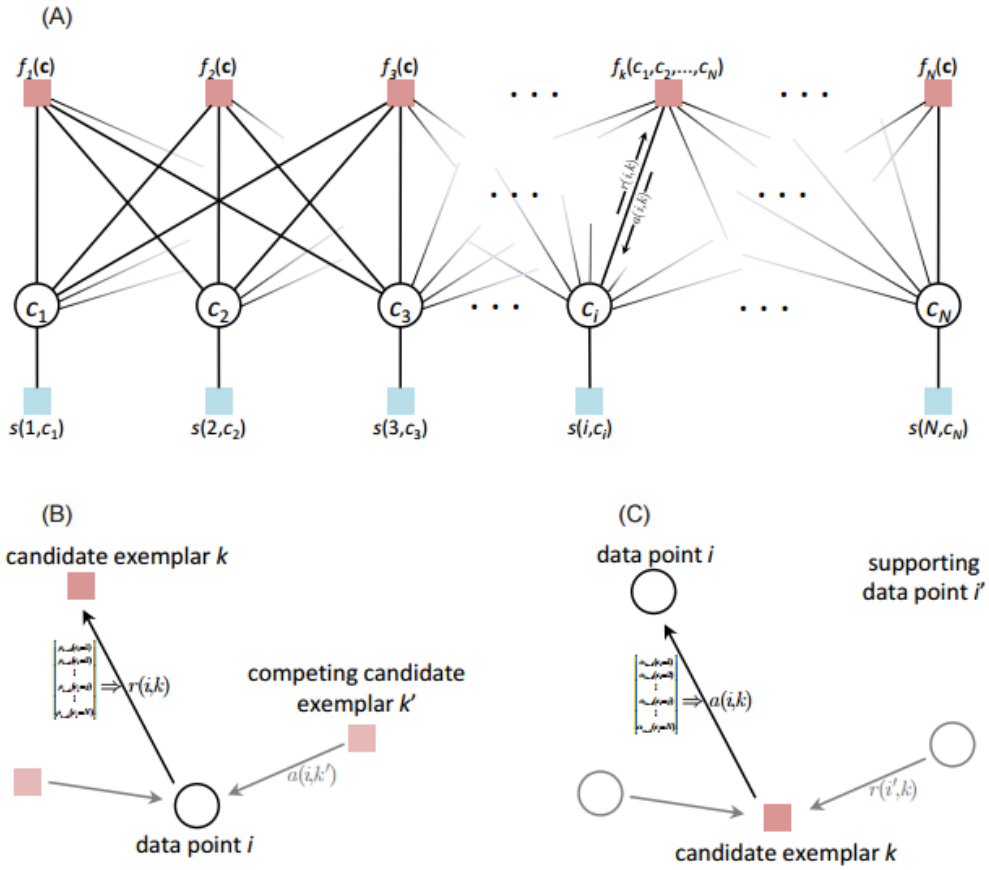
Sl. 2.1. Primjer grupiranja dvodimenzionalnih podataka različitim metodama [1]

Vidi se da metode spectral clustering, agglomerative clustering i pravilno grupiraju podatke dok kod ostalih metoda to nije slučaj. Također, kao što vidimo najbolje je spojen uzorak kod agglomerative klasteriranja gdje je u slučaju na slici spojio plavi uzorak sa plavim dok kod

spectral klasteriranja nije to slučaj. Dobar algoiram je i DBSCAN koji je posložio tri uzorka u svoje skupine (npr. plava, zelena i crvena što vidimo na slici 2.1)

Affinity propagation [3]

Algoritam na ulazu uzima skup realnih sličnosti između centra podataka {s(i, k)}, gdje svaka sličnost s (i, k) pokazuje koliko je dobar centar podataka s indeksom k te koliko je pogodan da bude primjer za točke koje prikazuju i. Svaka točka je u paru sa promjenjivim čvorom. Vrijednosti C_i = k te 6 = k ukazuju na to da nam se podaci dodjeljuju u skupinu sa točkom k kao njegov uzorak.



Sl. 2.2. Primjer na bazi klaster algoritma koji obavlja širenje na grafu

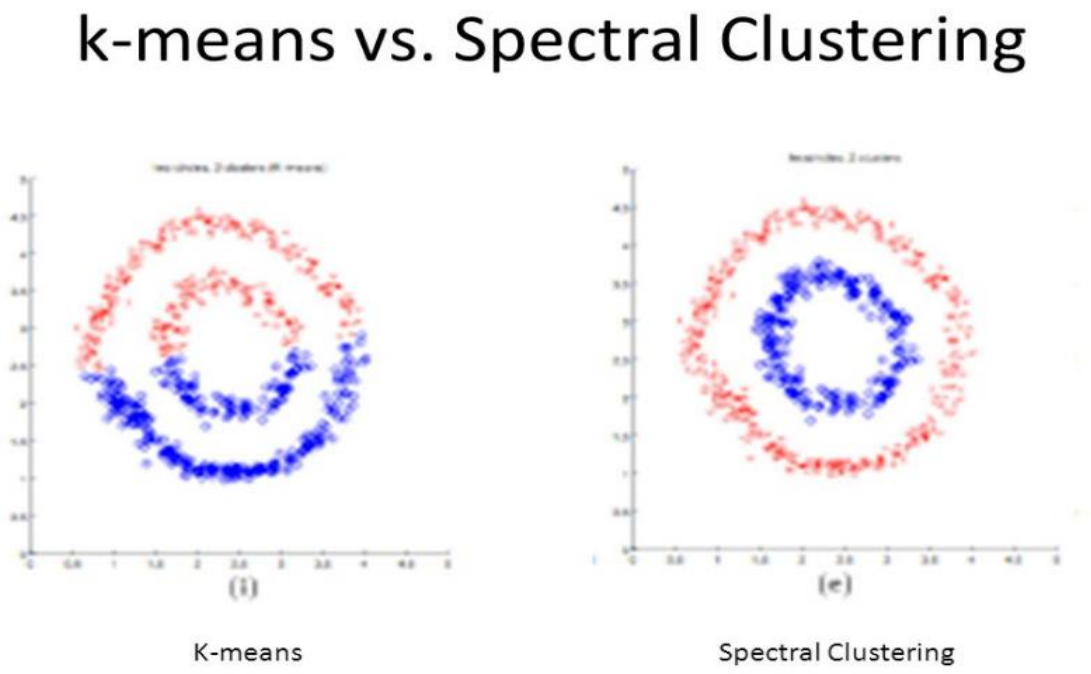
Primjer na bazi klastera algoritam koji obavlja širenje na grafu faktor koji je prikazan u (A) koji se nalazi na sl. 2.2. Prenose se dvije vrste poruka u grafikonu odgovornosti (B) s promjenjivim čvorovima. Dostupnost se prenosi iz funkcijskih čvorova na promjenjive čvorove (C), koji se očituju s primjerima kandidata podatkovnih točaka.

Mean- shift

Mean-shift [4] je ne parametarska tehnika za analizu gdje se locira maksimum funkcije gustoće. To je algoritam koji se temelji na centroidu, radi na način da se ažuriraju podaci za centroide koji imaju srednju vrijednost unutar određenog područja. Kanidati se filtriraju u fazi nakon obrade kako bi se uklonili duplikati za konačni skup centroida. Vidljivo je da metoda ne pretpostavlja neki konkretni oblik klastera već može prepoznati i dosta složene klastere. Za razliku od drugih algoritama ovaj je algoritam neovisan alat pogodan za stvarne analize podataka. Ne uzima se nikakav određeni oblik unaprijed na podacima klastera te se postupak oslanja samo na parametar propusnosti.

Spectral clustering [5]

Cilj prilikom realizacije ovog algoritma je okupiti podatke koji su spojeni, ali ne i nužno kopaktni ili grupirani unutar granice.



Sl. 2.3. S lijeve strane je prikaz K-means algoritma, dok se sa desne nalazi Specrtal clustering

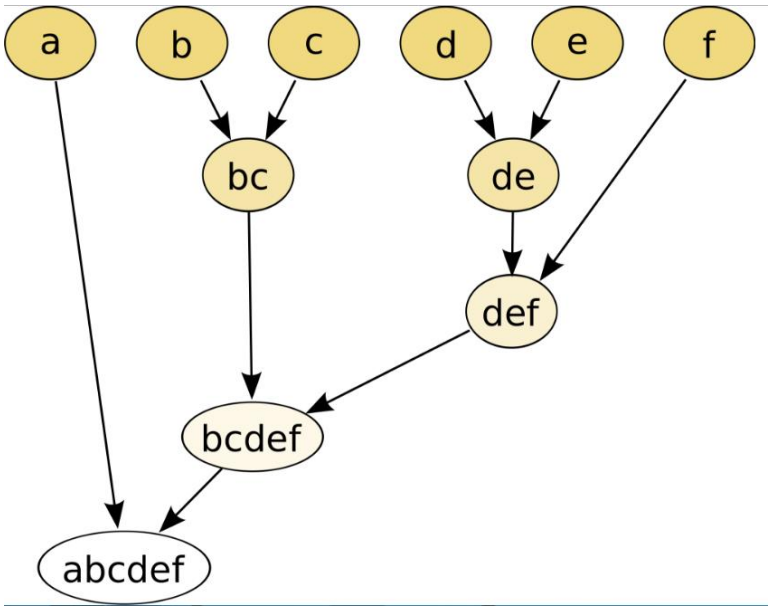
Na slici 2.3 vidimo prikaz K-means algoritma koji je rasporedio uzorke na način da je 2 klastera spojio skupa ali ih poredao po uzorcima što su u našem slučaju plava i crvena boja. S desne strane imamo algorita spectral clusterin koji je 2 klastera razdvojio te zasebno postavio crvene te plave uzorke.

Hierarchical clustering

Hijerarhijsko klasteriranje [6] je opća skupina klaster algoritma koja gradi ugniježđene klastere sukcesivnim spajanjem ili razdvajanjem dviju grupa podataka (mjerni uzorak). Tijek grupiranja podataka ovom metodom moguće je prikazati u obliku stabla. Korijen stabla je jedinstveni klaster koji sadrži sve uzorke, listovi čine nakupine od samo jednog uzorka.

Kriterij povezanosti određuje princip koji se koristi za strategiju spajanja:

- Ward minimizira zbroj kvadrata unutar svih klastera.
- Maksimalna ili potpuna veza smanjuje maksimalnu udaljenost između klastera.
- Prosjek povezanosti smanjuje prosječnu udaljenost između svih parova klastera.

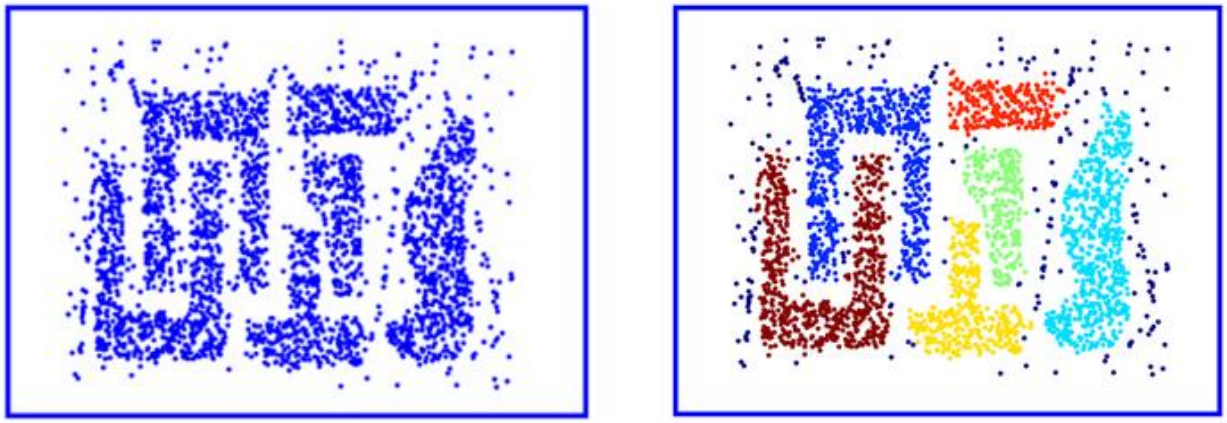


Sl. 2.4. Hijerarhijsko grupiranje u obliku gdje su nam zadani uzorci u obliku slova [14]

Na slici 2.4. je prikazano hijerarhijsko grupiranje podataka sa podacima u obliku slova. Cilje je grupirati podatke (slova) po abecednom redu. Kao što vidimo u drugoj liniji grupiranja, povezujemo podatke *b* i *c* te podatke *d* i *e*. Nakon toga u trećoj liniji povezujemo podatak *de* te podatak *f* u skupinu *def*. Nakon što smo došli u četvrti korak povezali smo podatke *def* te *bc* i dobili skup podatka po abecednom redu *bcdef*. Zadnji korak je konačno rješenje gdje smo povezali podatak *a* sa podatko *bcdef*.

DBSCAN [7]

Ovaj algoritam gleda klastere kao područja visoke gustoće odvojenih od područja sa niskom gustoćom. Zbog tog općenitog pogleda, klasteri pronađeni pomoću DBSCAN algoritma mogu biti bilo kakvog oblika. Središnja komponenta DBSCAN algoritma je koncept osnovnih uzoraka, a to su uzorci koji se nalaze u području visoke gustoće. Klaster je skup osnovnih uzoraka koji se nalaze jedan do drugoga. Ovaj algoritam ima dva parametra, a to su min_samples koji je minimalni broj podataka koji mora biti na manjoj udaljenosti od eps u okolini nekog drugog podatka da bi se taj podatak smatrao osnovnim uzorkom.



Sl. 2.5. S lijeve strane prikazan je originalni skup podataka dok se sa desne nalaze podaci koji su kreirani pomoću algoritma DBSCAN [13]

Na slici 2.5 je prikaz skupa podataka koji je zadan na početku prije izvršavanja algoritma DBSCAN. Desna slika nam prikazuje izvršen algoritam DBSCAN koji je skupu klastera dodjelio određenu boju kako bi ih raspoznali. Kao što vidite, smetnje koje se nalaze oko vanjskih parametara su riješene na odgovarajući način. DBSCAN ne radi dobro u slučaju kad se radi o klasterima različite gustoće ili visoko dimenzionalnih podataka.

Birch [8]

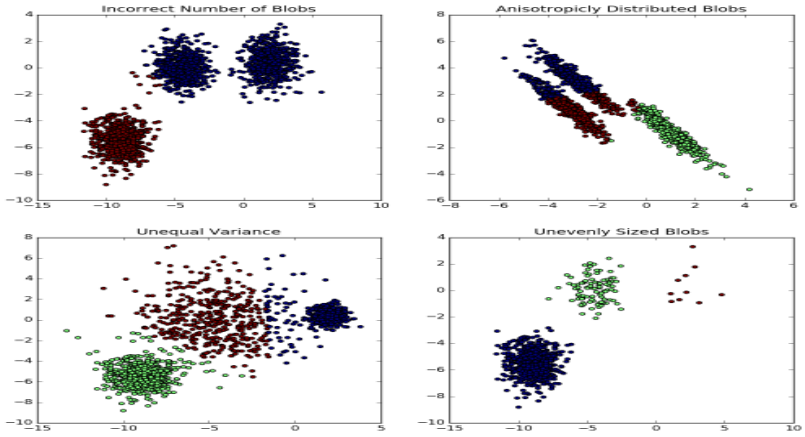
Birch izgrađuje stablo nazvano CFT (Characteristic Feature Tree) za dobivene podatke. Algoritam je bez nadzorna vrsta grupiranja podataka koji se koristi kod hijerarhijskih klastera tijekom velikih skupina podataka

3. K-means (K srednjih vrijednosti)

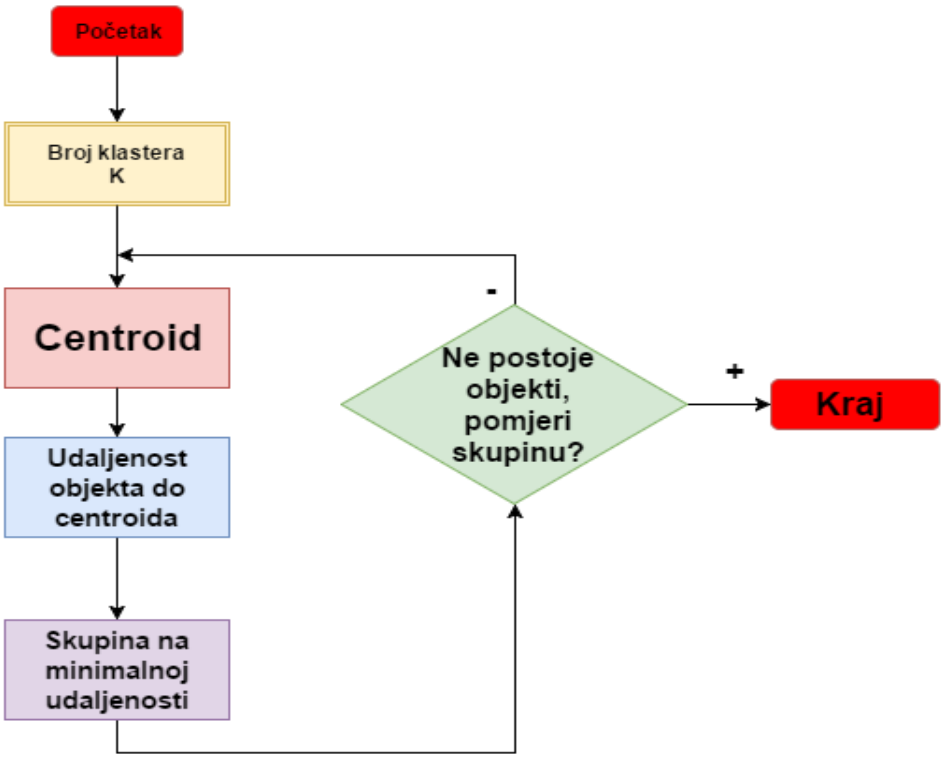
Podaci pomoću algoritma K srednjih vrijednosti[3] pokušavaju razdvojiti uzorke u *n* grupa jednakih varijanci, umanjuje se kriterij poznat kao inercija ili unutar klasterni zbroj kvadrata. Ovaj algoritam zahtjeva da se zada određeni broj klastera. Ovaj algoritam zahtjeva veliki broj uzoraka i korišten je na području velikog raspona u mnogim različitim područjima. Algoritam K srednjih vrijednosti dijeli skup od N uzoraka X u K disjunktnih klastera C, svaki je opisan sa srednjom vrijednosti centra klastera. Algoritam K srednjih vrijednosti odabire centroid koji minimiziraju inerciju ili unutar klasterni zbroj kvadrata.

$$\sum_{i=0}^{n} \min(\left|\left|x_j - \mu_i\right|\right|^2$$

K srednjih vrijednosti se još nazivaju i kao Lloyd algoritam. To je algoritam od tri koraka. Prilikom prvog koraka odabiru se inicijalni centroidi te se uz najosnovnije metode odabiru uzorci k iz skupa podataka X.Nakon inicijalizacije, K srednjih vrijednosti sastoji se od petlje između druga dva koraka. Prvi korak dodjeljuje svaki uzorak najbližem centru. Drugi korak stvara nove centroide uzimanjem srednjih vrijednosti svih uzoraka dodijeljenih svakom prethodnom težištu. Razlika između starih i novih centroida se računa i algoritam ponavlja posljednja dva koraka sve dok je ta vrijednost manja od praga. Nakon toga centroidi ažuriraju prosjek svakom segmentu. Algoritam ponavlja sve dok jedan od kriterija zaustavljanja nije ispunjen. Algoritam se zaustavlja kada je relativno smanjenje funkcije cilja između ponavljanja manja od zadane tolerancije vrijednosti.



Sl. 3.1. Primjeri rješenja K-means algoritma



Sl. 3.2. Blok dijagram K- means algoritma

3.1. Primjer pseudo koda K-means algoritma

1. Označi početno težište klastera

2. Ponovi

3. Ponoviti to za svaku točku x_i u skupu podataka

4. Ponoviti to za sve

5. Izračunaj različitost

6. Završi

7. Dodjeliti točku najbližu klasteru

8. Završi

9. Završi za sve

3.2. Određivanje parametra modela tj. centroida

Primjer [12] određivanja centroida u K-means algoritmu na osnovu sljedećeg skupa podataka koji se sastoji od dvije varijable (A i B) za sedam različitih osoba.

SUBJECT	A	B
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Tab 3.1

Podaci su grupirani u dva klastera. Prvi korak je pronalazak razumne inicijalne particije, neka vrijednost A i B za dvije osove koje su najdalje (koristimo Euklidovo mjerenje udaljenosti) te definiramo incijalna klaster sredstva.

	Individual	Mean vector (centroid)
Group 1	1	(1.0 1.0)
Group 2	4	(5.0 7.0)

Tab 3.2

Preostali pojedinci se ispituju u nizu te su smješteni do klatera koji su najbliži, u smislu Euklidove udaljenosti. Srednji vektor se izračunava svaki puta kada se dodaje novi član. To dovodi do sljedećeg niza koraka.

	Cluster 1		Cluster 2	
	Individual	**Mean vector (centroid)**	**Individual**	**Mean vector (centroid)**
1	1	(1.0 1.0)	4	(5.0 7.0)
2	1, 2	(1.2 1.5)	4	(5.0 7.0)
3	1, 2, 3	(1.8 2.3)	4	(5.0 7.0)
4	1, 2, 3	(1.8 2.3)	4, 5	(4.2 6.0)
5	1, 2, 3	(1.8 2.3)	4, 5, 6	(4.3 5.7)
6	1, 2, 3	(1.8 2.3)	4, 5, 6 ,7	(5.1 5.4)

Tab 3.3

Nakon što je inicijalna particija promjenjena, dva klastera u ovoj fazi dobijaju sljedeće karakteristike:

	Individual	**Mean vector (centroid)**
Cluster 1	1, 2, 3	(1.8 2.3)
Cluster 2	4, 5, 6, 7	(4.1 5.4)

Tab 3.4

Nakon ovog koraka još ne možemo sa sigurnošću reći koji pojedinac je dodjeljen pravom klasteru. Nastavljamo uspoređivati svaku udaljenost pojedinca sa njegovim klasterom te sa suprotnim klasterom te pronalazimo sljedeće rješenje.

Individual	**Distance to mean (centroid) of Cluster 1**	**Distance to mean (centroid) of Cluster 2**
1	1.5	5.4
2	0.4	4.3
3	2.1	1.8
4	5.7	1.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

Tab 3.5

Vidimo da je pojedinac tri najbliži suprotnom klasteru (Klaster 2) nego njegovom klasteru (Klateru 1). Drugim riječima, svaka udaljenost pojedinca za njego klaster bi trebala biti manja nego udaljenost do suprotnog klastera (što nije slučaj sa pojedincem 3). Pojedinacu tri se dodaje nova lokacija u klaster 2 što dovodi do novog konačnog rješenja.

	Individual	**Mean vector (centroid)**
Cluster 1	1, 2	(1.3 1.5)
Cluster 2	3, 4, 5, 6, 7	(3.9 5.1)

Tab 3.6

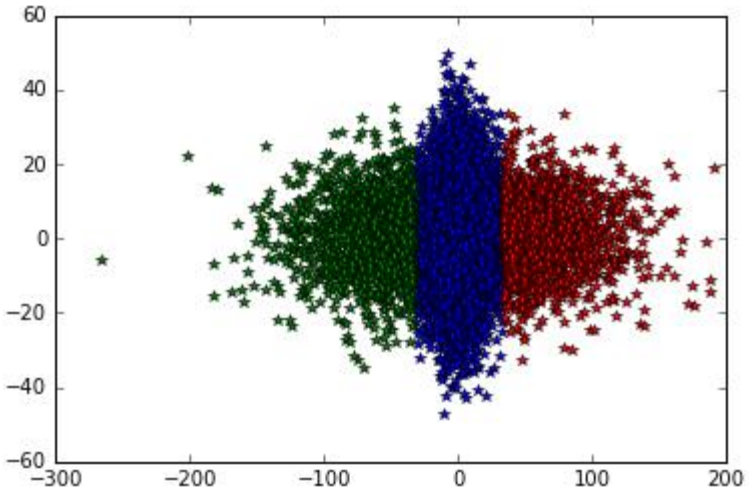
3.3 Primjer K-means algoritma

K srednjih vrijednosti je izveden u pythonu 2.7. pomoću sljedećeg koda koji se nalazi ispod:

```
"""
@author: Andrija
"""

import numpy as np
import pylab as plt

mean = [0, 0]
cov = [[320, 10], [200, 100]]
mean2 = [0, 1]
cov2 = [[100, 20], [100, 400]]
mean3 = [0, 1]
cov3 = [[300, 35], [5, 100]]


K = 3
maxIters = 10


data1 = np.random.multivariate_normal(mean, cov, 50)
data2 = np.random.multivariate_normal(mean2, cov2, 5000)
data3 = np.random.multivariate_normal(mean2, cov3, 5000)
X = np.vstack((data1, np.vstack((data2, data3))))
np.random.shuffle(X)

centroids = X[np.random.choice(np.arange(len(X)), K), :]
for i in range(maxIters):
C = np.array([np.argmin([np.dot(x_i - y_k, x_i - y_k) for y_k in centroids])
for x_i in X])
centroids = [X[C == k].mean(axis=0) for k in range(K)]
centroids = np.array(centroids)

plt.ion()
plt.cla()
plt.plot(X[C == 0, 0], X[C == 0, 1], '*b',
X[C == 1, 0], X[C == 1, 1], '*r',
X[C == 2, 0], X[C == 2, 1], '*g')
plt.draw()
plt.ioff()
plt.show()
```

Listing 3.1. Primjer grupiranja podataka u Pythonu

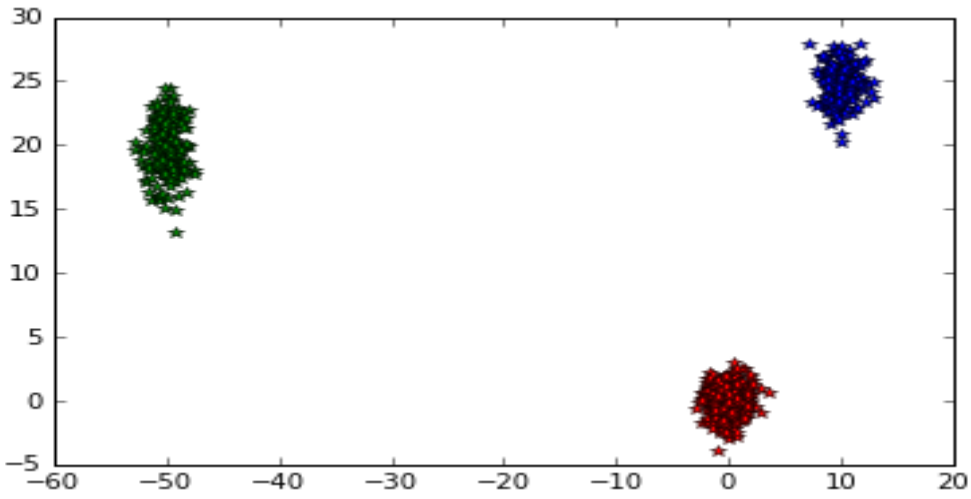
Rezultat koji smo dobili prilkom realizacije gore navedenog koda:



Sl. 3.3 Grafički prikaz 3 klastera prilikom izvršavanja gore navedenog koda

Primjer K-means algoritma kada su klasteri razdvojeni

```
import numpy as np
import pylab as plt

mean = [10, 25]
cov = [[1, 0], [0, 2]]
mean2 = [0, 0]
cov2 = [[1, 0], [0, 1]]
mean3 = [-50, 20]
cov3 = [[1, 0], [0, 4]]

K = 3
maxIters = 10

data1 = np.random.multivariate_normal(mean, cov, 200)
data2 = np.random.multivariate_normal(mean2, cov2, 500)
data3 = np.random.multivariate_normal(mean3, cov3, 200)
X = np.vstack((data1, np.vstack((data2, data3))))
....
```

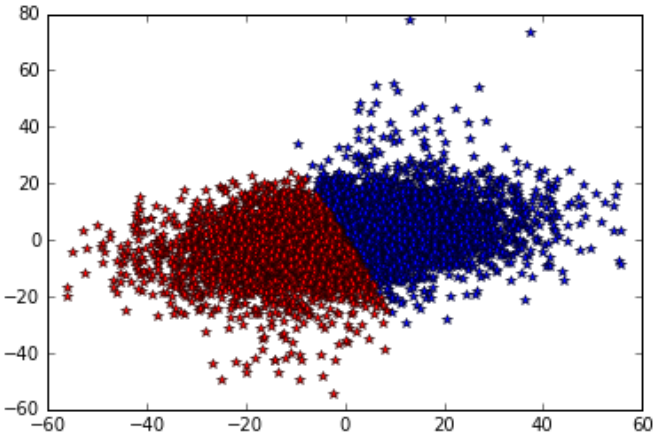


Sl. 3.4. Izgleda K-means algoritma kada su klasteri razdvojeni

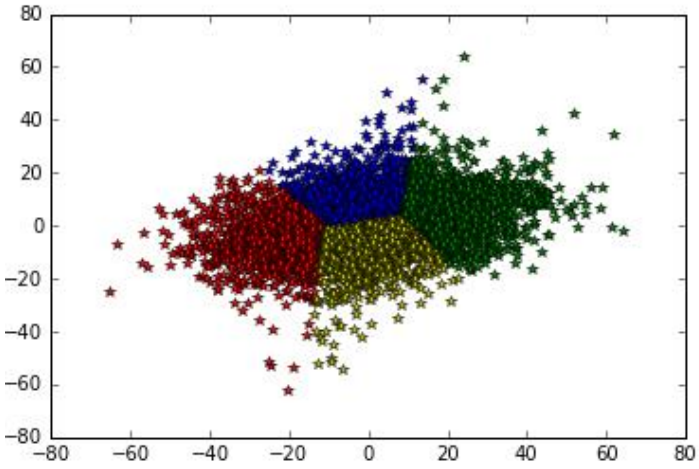
3.4. Specifičnosti K-means algoritma

- Potrebno je unaprijed zadati broj klastera i njihove početne vrijednosti

  mean = [0, 1]
  cov = [[320,200], [200, 100]]
  mean2 = [0, 1]
  cov2 = [[100, 120], [100, 400]]
  mean3 = [0, 1]
  cov3 = [[300, 35], [100, 100]]
  K = 2 =>***Određivanje broja klastera***



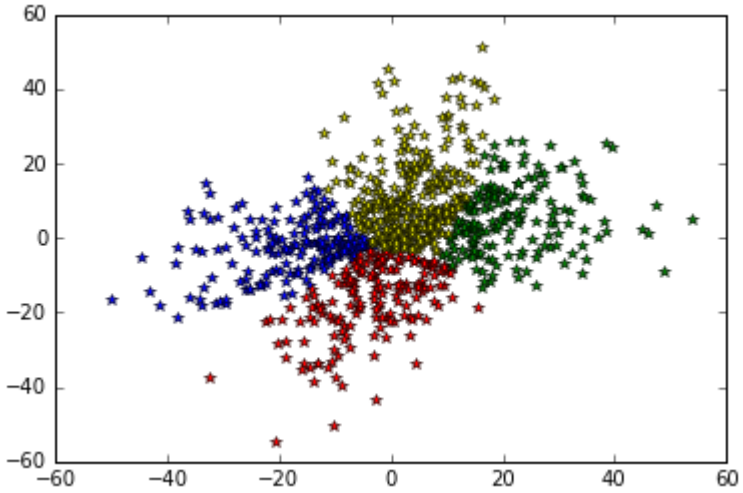
Sl. 3.5 Rješenje sa 2 klastera



Sl. 3.6 Rješenje sa 4 klastera

- Određivanje gustoće K-means klastera
  - *Rješenje sa parametrima data 1 (50), data2 (300), data 3(500)*

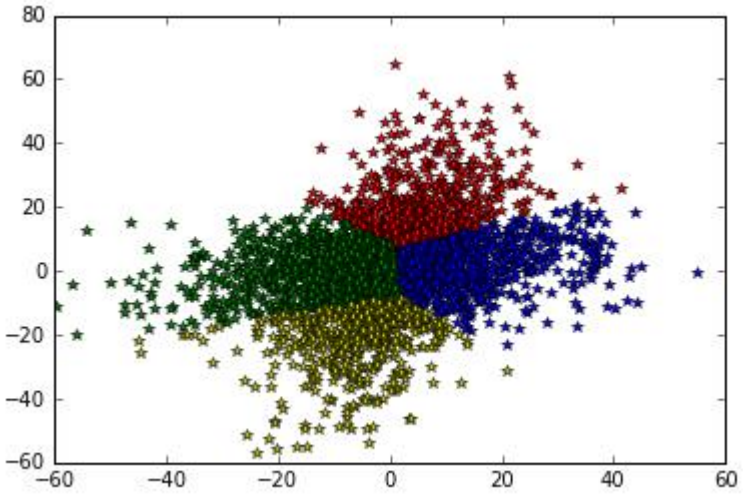
```
data1 = np.random.multivariate_normal(mean, cov, 50)
data2 = np.random.multivariate_normal(mean2, cov2, 300)
data3 = np.random.multivariate_normal(mean2, cov3, 500)
X = np.vstack((data1, np.vstack((data2, data3))))
np.random.shuffle(X)
```



Sl. 3.7 K-means algoritam sa manjom gustoćom

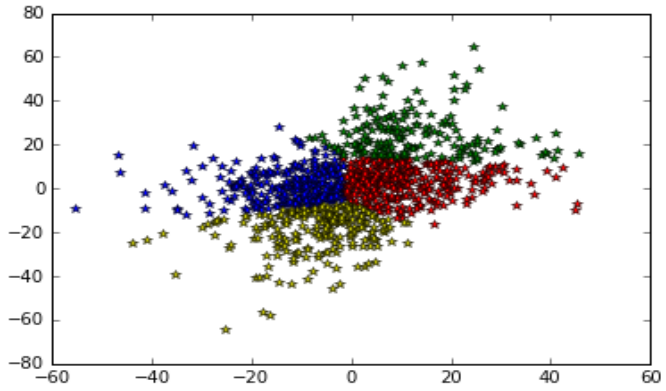
  - *Rješenje sa parametrima data 1 (50), data2 (1300), data 3(1500)*

```
data1 = np.random.multivariate_normal(mean, cov, 50)
data2 = np.random.multivariate_normal(mean2, cov2, 1300)
data3 = np.random.multivariate_normal(mean2, cov3, 1500)
```

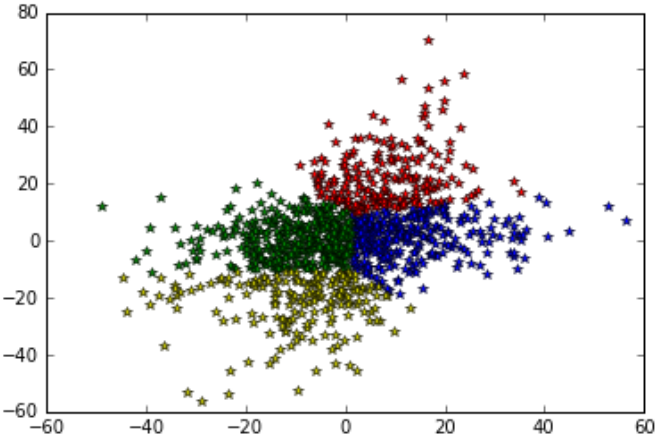


Sl. 3.8 K-means algoritam sa većom gustoćom

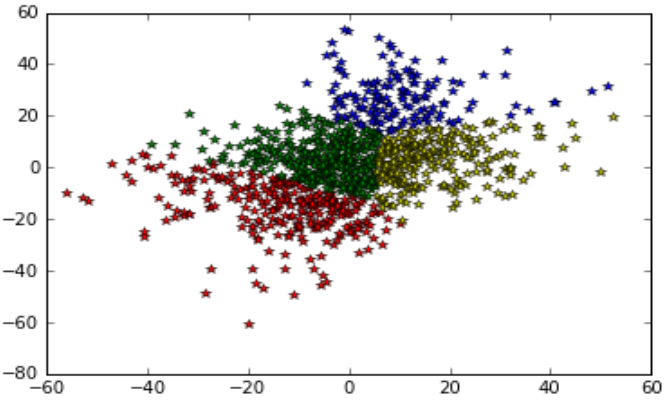
- *Rješenje nije uvijek isto*
  -Rješenje prilikom izvršavanja K-means algoritma nam nije uvijek isto kao što vidimo u primjeru ispod gdje su parametri uvijek bili jednaki.



Sl. 3.9 Prilikom prvog pokretanja



Sl. 3.10 Prilikom drugog pokretanja



Sl. 3.11 Prilikom trećeg pokretanja

4. Zaključak

Izrada projektnog zadatka mi je pomogla u upoznavanju pythona kojeg nisam koristio u velikoj mjeri. Prilikom ovog zadatka susreo sam se sa mnogim problemima poput same realizacije koda ali sam i naučio mnoge vrste grupiranja podataka među kojima sam se najviše posvetio K-means algoritmu. Prilikom izvršavanja K means algoritma najbolji pregled sam imao kad su postojale male količine uzoraka jer tada sam dobio preglednija rješenja iako se sa velikim količinama uzoraka dobijaju bolja riješenja. Jedna od stvari koje su bitne kod K-means algoritma je ta što je potrebno unaprijed zadati broj klastera koji će se realizirati. Također, prilikom svakog izvršavanja algoritma dobili smo drugačiji izgled riješenja iako su parametri ostajali isti što znači da nikad nećemo dobiti isto riješenje.

5. Literatura

[1] http://scikit-learn.org/stable/modules/
Pristupio: 15.09.2016

[2] http://scikit-learn.org/stable/modules/clustering.html#k-means
Pristupio: 15.09.2016

[3] http://scikit-learn.org/stable/modules/clustering.html#affinity-propagation
Pristupio: 15.09.2016

[4] http://scikit-learn.org/stable/modules/clustering.html#mean-shift
Pristupio: 15.09.2016

[5] http://scikit-learn.org/stable/modules/clustering.html#spectral-clustering
Pristupio: 15.09.2016

[6] http://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering
Pristupio: 15.09.2016

[7] http://scikit-learn.org/stable/modules/clustering.html#dbscan
Pristupio: 15.09.2016

[8] http://scikit-learn.org/stable/modules/clustering.html#birch
Pristupio: 15.09.2016

[9] https://en.wikipedia.org/wiki/K-means_clustering
Pristupio: 15.09.2016

[10] https://en.wikipedia.org/wiki/Affinity_propagation
Pristupio: 15.09.2016

[11] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.150.5946&rep=rep1&type=pdf
Pristupio: 26.09.2016

[12] http://mnemstudio.org/clustering-k-means-example-1.htm
Pristupio: 26.09.2016

[13]http://www.hypertextbookshop.com/dataminingbook/public_version/contents/chapters/chapter004/section004/blue/page003.html
Pristupio 29.09.016

[14]https://upload.wikimedia.org/wikipedia/commons/thumb/a/ad/Hierarchical_clustering_simple_diagram.svg/2000px-Hierarchical_clustering_simple_diagram.svg.png
Pristupio 29.09.016