Primjena klaster analize na segmentaciju slike

Šimunović, Ines

Sveučilište Josipa Jurja Strossmayera u Osijeku

Fakultet primijenjene matematike i informatike

Sveučilišni prijediplomski studij Matematika i računarstvo

Primjena klaster analize na segmentaciju slike

Završni rad

Mentor:

prof. dr. sc. Kristian Sabo

Student:

Ines Šimunović

Osijek, 2024

Sadržaj

1 | Uvod

Klasteriranje je jedna od ključnih tehnika u području strojnog učenja i analize podataka koja omogućava grupiranje sličnih objekata u iste skupove, poznate kao klasteri. Cilj klasteriranja je organizirati podatke na način koji omogućuje lakše razumijevanje, analizu i donošenje zaključaka. Klasteri se formiraju tako da su objekti unutar istog klastera slični jedni drugima prema određenom kriteriju, dok su objekti iz različitih klastera međusobno različiti.
Važnost klasteriranja očituje se u tome što omogućava otkrivanje skrivenih obrazaca u podacima bez potrebe za prethodnim znanjem o tim podacima. Osim što pomaže u boljem razumijevanju podataka, klasteriranje također omogućava smanjenje složenosti podataka i poboljšava točnost predviđanja u modelima strojnog učenja.
Tehnika klasteriranja ima široku primjenu. U poslovnom svijetu, često se primjenjuje za segmentaciju tržišta, gdje pomaže identificirati različite grupe kupaca na temelju njihovih karakteristika i ponašanja, omogućujući tvrtkama da prilagode svoje marketinške strategije. U medicini, klasteriranje se koristi za grupiranje pacijenata sličnih simptoma ili bioloških markera, što pomaže u dijagnostici i personalizaciji liječenja. U bioinformatici, klasteriranje omogućava grupiranje gena sličnih osobina kako bi se identificirale funkcionalne grupe. Također, koristi se i za obradu i segmentaciju slika (što će biti obrađeno u ovome radu) i prepoznavanju objekata.
Završni rad podijeljen je u četiri glavna poglavlja. U poglavlju nakon uvoda, koje se sastoji od nekoliko potpoglavlja, objašnjen je pojam klaster analize i vrste klasteriranja. Također, objašnjeni su osnovni pojmovi potrebni za uvođenje i razumijevanje k-means algoritma, poput particije skupa, najboljeg reprezentanta skupa, kriterijske funkcije i optimalne particije. Na kraju poglavlja objašnjen je k-means algoritam koji je ilustriran primjerima i odgovarajućim programskim kodom u Pythonu. Treće poglavlje opisuje izbor primjerenog broja klastera u particiji te su obrađena tri kriterija: Calinski–Harabasz indeks, Davies–Bouldin indeks i kriterij širine siluete. Korištenje indeksa prikazano je primjerom te odgovarajućim grafovima. Zadnje poglavlje odnosi se na segmentaciju crno-bijele slike i slike u boji.

2 | Klasteriranje

Klasteriranje ili klaster analiza (engl. *cluster analysis*) prvi se puta spominje 1939. godine te podrazumijeva grupiranje objekata na temelju sličnosti karakteristika koje posjeduju. Cilj klasteriranja je pronalaženje optimalog grupiranja podataka. Nakon završene klaster analize, identificiraju se grupe koje se nazivaju i klasteri, pri čemu vrijedi da je sličnost podataka u istim klasterima maksimalna, a sličnost između različitih klastera minimalna. Važno je navesti da proces klasteriranja ne zahtijeva uvijek pretpostavke o broju i karakteristikama klastera (grupa) u koje će podaci biti raspoređeni. Grupiranje se vrši na temelju sličnosti, što ponekad sa sobom može donijeti probleme o tome kako definirati i promatrati sličnost.
Algoritmi koji se koriste za klasteriranje dijele se u dvije skupine:

- particijski
- hijerarhijski [5].

Particijski algoritmi dijele skup podataka na n klastera, gdje je n unaprijed određen broj klastera te svaki podatak pripada isključivo jednom klasteru. Ova vrsta algoritama većinom pokušava optimizirati određenu funkciju cilja. S druge strane, postoje hijerarhijski algoritmi i oni grade hijerarhiju klastera koji su organizirani u strukturi sličnom stablu, gdje se klasteri mogu nalaziti unutar drugih klastera. Ovi algoritmi ne zahtijevaju unaprijed definiran broj klastera. Particijski algoritmi su korisni kada je broj klastera unaprijed poznat i kada je važna brzina, dok hijerarhijski algoritmi pružaju fleksibilnost i dublje razumijevanje strukture podataka pa su stoga i zahtjevniji za implementaciju.
Nadalje, particijski algoritmi dijele se na:

- tvrdo klasteriranje (engl. *hard clustering*)
- meko klasteriranje (engl. *soft clustering*) [2].

Tvrdo klasteriranje je pristup u kojem se svaki podatak dodjeljuje isključivo jednom klasteru. Dakle, svaki objekt ima jedinstvenu pripadnost klasteru i ne može pripadati niti jednom drugom klasteru. Kod mekog klasteriranja umjesto točno određenih granica, koristi se pristup u kojem se dodjeljuju vjerojatnosti ili stupnjevi pripadnosti svakom klasteru.
U ovome radu fokus je na tvrdom klasteriranju i njegovim karakteristikama.

2.1 Tvrdo klasteriranje

2.1.1 Particija skupa

Definicija 1 (vidjeti [8, Definicija 3.1.]). *Neka je* $\mathcal{X} = \{x_i \in \mathbb{R}^n : i = 1, \ldots, m\}$ *skup koji sadrži* $m \geq 2$ *elemenata. Rastav skupa* $\mathcal{X}$ *na* $1 \leq k \leq m$ *disjunktnih nepraznih podskupova* $\pi_1, \ldots, \pi_k$*, takvih da je*

$$\bigcup_{j=1}^{k} \pi_j = \mathcal{X}, \quad \pi_r \cap \pi_s = \emptyset, \quad r \neq s, \quad |\pi_j| \geq 1, \quad j = 1, \ldots, k,$$

zovemo k-particija skupa $\mathcal{X}$ *i označavamo s* $\Pi = \{\pi_1, \ldots, \pi_k\}$*. Elemente particije zovemo klasteri, a skup svih particija skupa* $\mathcal{X}$ *sastavljenih od k klastera označavamo s* $\mathcal{P}(\mathcal{X}; k)$*.*

Teorem 1 (vidjeti [8, Teorem 3.1.]). *Broj svih particija skupa* $\mathcal{X}$ *sastavljenih od k klastera jednak je Stirlingovom broju druge vrste*

$$|\mathcal{P}(\mathcal{X}, k)| = \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{k}{j} j^m.$$

Primjer 1. *Približni broj svih k-particija skupa* $\mathcal{X}$ *koje zadovoljavaju Definiciju 1 za* $m = 5, 50, 1200$ *i* $k = 2, 4, 6, 10$ *prikazan je u Tablici 2.1.*

	$k=2$	$k=4$	$k=6$	$k=10$
$m=5$	15	10	-	-
$m=50$	10^{15}	10^{29}	10^{36}	10^{44}
$m=1200$	10^{361}	10^{721}	10^{931}	10^{1193}

Tablica 2.1: Približan broj k-particija skupa $\mathcal{X}$ koji ima m elemenata za broj klastera k

2.1.2 Najbolji reprezentant skupa

Definicija 2. *Funkciju* $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$, $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$*, sa svojstvima:*

(i) $d(x, y) = 0 \iff x = y$,

(ii) $x \mapsto d(x, y)$ *je neprekidna na* $\mathbb{R}^n$*, za svaki fiksni* $y \in \mathbb{R}^n$,

(iii) $\lim_{\|x\| \to +\infty} = +\infty$*, za svaki fiksni* $y \in \mathbb{R}^n$.

zovemo kvazimetrička funkcija (*engl. distance like function*).

Dvije najčešće korištene kvazimetričke funkcije na $\mathbb{R}$ su:

- kvazimetrička funkcija namjanjih kvadrata (engl. *least squares distance like function*): $d_{LS}(x, y) = (x - y)^2$

- ℓ_1-metrička funkcija ili *Manhattan metrička funkcija*: $d_1(x,y) = |x-y|$.

Definicija 3. *Neka je* $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$, $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$ *kvazimetrička funkcija. Najbolji reprezentant skupa* $\Pi_j = \{x_i \in \mathbb{R}^n, i = 1,\ldots,m_j\}, m_j = |\Pi_j|$ *je vektor* $\mu_j^* \in \operatorname{argmin}_{x\in\mathbb{R}} \sum_{i=1}^{m_j} d(\mu, x_i)$.

Primjerice, najbolji reprezentant skupa možemo promatrati u kontekstu:

- $\mu_j^* = \frac{1}{m_j}\sum_{i=1}^{m_j} x_i$ (prosjek, za kojeg se može pokazati da je najbolji reprezentant podataka ako za kvazimetričku funkciju koristimo LS-kvazimetričku funkciju)
- $\mu_j^* = \operatorname{med}_{x_i\in\Pi_j} x_i$ (medijan, za kojeg se može pokazati da je najbolji reprezentant podataka ako za kvazimetričku funkciju koristimo ℓ_1-metričku funkciju) [8].

2.1.3 Kriterijska funkcija i optimalna particija

Ukoliko je dan skup $\mathcal{X} = \{x_i \in \mathbb{R}^n : i = 1,\ldots,m\} \subset \mathbb{R}^n$ te je k broj klastera, tada možemo definirati kriterijsku funkciju $\mathcal{F} : \mathcal{P}(\mathcal{X};k) \to [0,+\infty)$

$$\mathcal{F}(\Pi) = \sum_{j=1}^{k}\sum_{x_i\in\Pi_j} d(\mu_j, x_i),$$

gdje je μ_j reprezentant klastera Π_j. Cilj je da kriterijska funkcija $\mathcal{F}(\Pi)$ ima minimalnu vrijednost.

Definicija 4. *Za particiju* $\Pi^* \in \mathcal{P}(\mathcal{X};k)$ *kažemo da je optimalna ako je* $\mathcal{F}(\Pi^*) = \mathcal{F}(\Pi), \forall\, \Pi \in \mathcal{P}(\mathcal{X};k)$.

2.1.4 K-means algoritam

K-means je algoritam strojnog učenja (nenadzirano učenje) koji se koristi za grupiranje, tj. klasteriranje podataka. Algoritam daje lokalno optimalno rješenje koje ovisi o izboru početne aproksimacije te se višestruko pokreće dok rješenje ne bude optimalno.

Koraci algoritma:

1. Odrediti početne centre $\mu_j, j = 1,\ldots,k$
2. Odrediti klastere $\Pi_j = \{x_i \in \mathcal{X} : d(\mu_j, x_i) \leq d(\mu_s, x_i), \forall\, s = 1,\ldots,k\}, j = 1,\ldots,k$ (assignment step)
3. Odrediti centre $\mu_j \in \operatorname{argmin}_{\mu} \sum_{x_i\in\Pi_j} d(\mu, x_i)$ (update step)
4. Ponavljati korake 2. i 3. dok se centri/klasteri ne podudaraju

Dakle, najprije se odabere broj k koji određuje u koliko grupa, tj. klastera naši podaci trebaju biti raspoređeni. Nakon toga slučajno se odabiru centri - k točaka. Te točke predstavljaju središte svakog klastera. Nakon što su određeni početni centri, formiraju se klasteri na način da je svaka točka dodijeljenja najbližem centru. Potom se u update step-u određuju novi centri tako da se minimizira udaljenost svih točaka u klasteru Π_i od centra. Drugim riječima, novi centar μ_j definiran je kao točka koja minimizira sumu udaljenosti između te točke i svih točaka unutar klastera Π_j. Ponovnim ažuriranjem centara algoritam konvergira prema optimalnim klasterima u kojima su točke u klasteru najbliže svom centru. Budući da k-means algoritam konvergira lokalno, obično se koristi višestruko pokretanje sa slučajno generiranim početnim centrima [7].

Primjer 2. *Primjenom k-means algoritma potrebno je pronaći ℓ_1-optimalnu particiju skupa $\mathcal{X} = \{1, 3, 7, 9, 11, 14, 18\}$ za $k = 3$.*

Rješenje. Najprije slučajno odaberemo 3 centra budući da je zadan $k = 3$. Neka to budu 3 vrijednosti iz skupa: $\mu_1 = 1, \mu_2 = 9, \mu_3 = 18$. Sada određujemo klastere obzirom na centre tako da svaku točku dodijelimo najbližem centru:

- $x_1 = 1$
  $d_1(1,1) = 0, d_1(1,9) = 8, d_1(1,18) = 17 \Rightarrow$ najbliži je μ_1 pa pripada klasteru Π_1
- $x_1 = 3$
  $d_1(3,1) = 2, d_1(3,9) = 6, d_1(3,18) = 15 \Rightarrow$ najbliži je μ_1 pa pripada klasteru Π_1
- $x_1 = 7$
  $d_1(7,1) = 6, d_1(7,9) = 2, d_1(7,18) = 11 \Rightarrow$ najbliži je μ_2 pa pripada klasteru Π_2
- $x_1 = 9$
  $d_1(9,1) = 8, d_1(9,9) = 0, d_1(9,18) = 9 \Rightarrow$ najbliži je μ_2 pa pripada klasteru Π_2
- $x_1 = 11$
  $d_1(11,1) = 10, d_1(11,9) = 2, d_1(11,18) = 7 \Rightarrow$ najbliži je μ_2 pa pripada klasteru Π_2
- $x_1 = 14$
  $d_1(14,1) = 13, d_1(14,9) = 5, d_1(14,18) = 4 \Rightarrow$ najbliži je μ_3 pa pripada klasteru Π_3
- $x_1 = 18$
  $d_1(18,1) = 17, d_1(18,9) = 9, d_1(18,18) = 0 \Rightarrow$ najbliži je μ_3 pa pripada klasteru Π_3

Dakle, nakon ove iteracije imamo: $\Pi_1 = \{1,3\}, \Pi_2 = \{7,9,11\}, \Pi_3 = \{14,18\}$. Sada određujemo nove centre tako da novi centar za svaki klaster bude medijan

vrijednosti unutar klastera:
$med(1,3) = 2, med(7,9,11) = 9, med(14,18) = 16$.
Dakle, sada imamo centre: $\mu_1 = 2, \mu_2 = 9, \mu_3 = 16$.
Sada ponavljamo 2. korak algoritma na analogan način kao za početno odabrane centre (računamo udaljenosti). Nakon ovog koraka klasteri se nisu promijenili, tj. vrijednosti pripadaju istim klasterima kao u prethodnoj iteraciji: $\Pi_1 = \{1,3\}, \Pi_2 = \{7,9,11\}, \Pi_3 = \{14,18\}$ iz čega se zaključuje da je upravo ovo optimalna particija danog skupa.

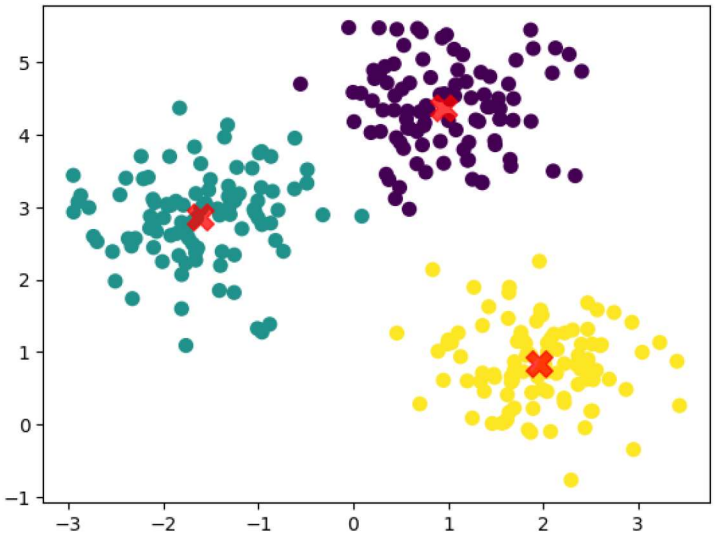
Primjer 3. *Korištenje k-means algoritma u Pythonu prikazano je sljedećim kodom:*

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

np.random.seed(42)
X, y = make_blobs(n_samples=300, centers=None, cluster_std=0.60,
    random_state=0)

kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=200, alpha
    =0.75, marker='X', label='Centroids')
```



Slika 2.1: Nasumično generirani podaci raspoređeni u 3 klastera nakon poziva *k*-means algoritma

U ovome kodu korišten je ugrađeni *k*-means algoritam koji je uvezen iz Scikit-Learn biblioteke. Najprije je generirano 300 podataka te je na njih pozvan algoritam kojemu je proslijeđen parametar 3 (broj klastera). Na Slici 2.1 prikazano je konačno rješenje algoritma gdje se vidi kako su se točke rasporedile u 3 klastera te su također označena 3 pripadna centra.

3 | Primjereni broj klastera u particiji

Tijekom klaster analize postavlja se važno pitanje: U koliko klastera je najprihvatljivije grupirati dani skup podataka, odnosno kako izabrati particiju s najprikladnijim brojem klastera?
Optimalna konfiguracija klastera definira se kao "najznačajnija" asocijacija svih mogućih kombinacija grupiranja. To se smatra temeljnim i vrlo teškim problemom klaster analize. Jedan od razloga je taj što klasteriranje treba biti izvedeno bez prethodnog razumijevanja unutarnje strukture podataka [6].
Odgovor na gore postavljeno pitanje obično se dobiva ispitivanjem različitih pokazatelja koje jednostavno nazivamo indeksi [8]. Neki od njih su: Calinski–Harabasz indeks, Davies–Bouldin indeks, kriterij širine siluete...

3.1 Calinski–Harabasz indeks

Calinski–Harabasz (CH) indeks definira se tako da interno kompaktnija particija čiji su klasteri dobro međusobno razdvojeni ima veću CH vrijednost [8].

Kada se određuje optimalna k-particija $\Pi^* = \{\pi_1^*, \ldots, \pi_k^*\}$ koristeći LS-kvazimetričku funkciju $d_{LS}(x,y) = \|x-y\|_2^2$, tada se odgovarajuća kriterijska funkcija $\mathcal{F}$ može napisati na sljedeći način:

$$\mathcal{F}_{LS}(\Pi) = \sum_{j=1}^{k} \sum_{x_i \in \Pi_j} \|\mu_j - x_i\|_2^2.$$

Vrijednost funkcije $\mathcal{F}_{LS}$ na optimalnoj particiji Π^* pokazuje ukupno "rasipanje" elemenata svih klastera $\pi_1^*, \ldots, \pi_k^*$ te particije do njihovih centara $\mu_1^*, \ldots, \mu_k^*$. Što je vrijednost funkcije F_{LS} manja, time je "rasipanje" manje, što znači da su klasteri interno kompaktniji. Stoga se može pretpostaviti da je CH-indeks optimalne particije Π^* obrnuto proporcionalan vrijednosti kriterijske funckije $\mathcal{F}_{LS}(\Pi^*)$.

Osim što možemo promatrati minimizaciju kriterijske funkcije $\mathcal{F}_{LS}$, pri traženju optimalne particije možemo promatrati i maksimum odgovarajuće dualne funkcije:

$$\mathcal{G}(\Pi) = \sum_{j=1}^{k} m_j \|\mu_j - \mu\|_2^2,$$

gdje je $m_j = |\pi_j|$, $\mu_j = \frac{1}{m_j}\sum_{x_i \in \Pi_j}$, $\mu = \frac{1}{m}\sum_{i=1}^{m} x_i$.
Vrijednost funkcije $\mathcal{G}$ na particiji Π^* govori nam o ukupnoj težinskoj razdvojenosti centara $\mu_1^*, \ldots, \mu_k^*$ klastera $\pi_1^*, \ldots, \pi_k^*$. Što je vrijednost funkcije $\mathcal{G}$ veća, time su i LS-udaljenosti centara μ_j^* do centara cijelog skupa μ^* veće. Stoga se može pretpostaviti da je CH-indeks optimalne particije Π^* proporcionalan vrijednosti kriterijske funckije $\mathcal{G}(\Pi^*)$.

Broj

$$\mathrm{CH}(k) = \frac{\frac{1}{k-1}\mathcal{G}(\Pi^*)}{\frac{1}{m-k}\mathcal{F}_L S(\Pi^*)}$$

nazivamo CH-indeks particije Π^*, a particiju s najvećim CH-indeksom smatramo particijom s najprikladnijim brojem klastera [8].

3.2 Davies–Bouldin indeks

Za svaki klaster π_j računa se razina sličnosti podataka u tom klasteru s podacima u ostalim klasterima i najveća izračunata vrijednost dodjeljuje se klasteru π_j. Zatim se Davies–Bouldin (DB) indeks dobiva prosjekom svih izračunatih sličnosti klastera. Što je indeks manji, to je rezultat klasteriranja bolji. Minimiziranjem ovog indeksa, klasteri su međusobno najrazličitiji i, prema tome, postiže se optimalna particija [1].

Neka je $\mu \in \mathbb{R}^2$ točka u ravnini oko koje je primjenom Gaussove normalne distribucije s varijancom σ^2 generirano m slučajnih točaka x_i. Ovaj skup točaka čini sferičan skup podataka i označimo ga s $\mathcal{X}$.
Poznato je da se u krugu $K(\mu, \sigma)$ sa središtem u točki μ i radijusom σ (standardna devijacija) nalazi oko 68% točaka skupa $\mathcal{X}$. Ovaj krug zvat ćemo glavni krug skupa podataka $\mathcal{X}$.
Pretpostavimo da su za dvije različite točke $\mu_1, \mu_2 \in \mathbb{R}^2$ i dvije različite varijance σ_1^2, σ_2^2 na prethodno opisan način generirana dva sferična skupa podataka $\mathcal{X}_1, \mathcal{X}_2$ i da su $K_1(\mu_1, \sigma_1)$, $K_2(\mu_2, \sigma_2)$ njihovi odgovarajući glavni krugovi. Radijus σ_1 predstavlja standardnu devijaciju skupa $\mathcal{X}_1$, a radijus drugog kruga σ_2 standardna je devijacija skupa $\mathcal{X}_2$.
Mogući odnosi skupova $\mathcal{X}_1$ i $\mathcal{X}_2$ obzirom na međusobni položaj njihovih glavnih krugova $K_1(\mu_1, \sigma_1)$ i $K_2(\mu_2, \sigma_2)$ su sljedeći:

- glavni krugovi se presijecaju (imaju neprazan presjek): $\|\mu_1 - \mu_2\|_2 \leq \sigma_1 + \sigma_2$
- glavni krugovi se dodiruju: $\|\mu_1 - \mu_2\|_2 = \sigma_1 + \sigma_2$
- glavni krugovi su razdvojeni: $\frac{\sigma_1 + \sigma_2}{\|\mu_1 - \mu_2\|_2} < 1$.

Nadalje, promotrimo optimalnu particiju Π^* skupa $\mathcal{X}$ s klasterima $\pi_1^*, \ldots, \pi_k^*$ i njihovim centrima $\mu_1^*, \ldots, \mu_k^*$. Razmotrimo odnos klastera π_j^* prema ostalim klasterima. Veličinom

$$D_j := \max_{s \neq j} \frac{\sigma_j + \sigma_s}{\|\mu_j^* - \mu_s^*\|_2}, \quad \sigma_j^2 := \frac{1}{|\pi_j^*|} \sum_{x \in \pi_j^*} ||\mu_j^* - x||_2^2,$$

je zadano najveće moguće preklapanje klastera π_j^* s nekim drugim klasterom. Veličina

$$\frac{1}{k}(D_1 + \ldots + D_k)$$

predstavlja još jednu mjeru interne kompaktosti i eksterne razdvojenosti klastera u particiji. Što je taj broj manji, klasteri su kompaktniji i bolje razdvojeni.
DB-indeks optimalne particije Π^* skupa $\mathcal{X}$ s klasterima $\pi_1^*, \ldots, \pi_k^*$ i njihovim centrima $\mu_1^*, \ldots, \mu_k^*$ definiran je na sljedeći način:

$$DB(k) := \frac{1}{k} \sum_{j=1}^{k} \max_{s \neq j} \frac{\sigma_j + \sigma_s}{\|\mu_j^* - \mu_s^*\|_2}, \quad \sigma_j^2 := \frac{1}{|\pi_j^*|} \sum_{x \in \pi_j^*} ||\mu_j^* - x||_2^2.$$

Particija s najmanjim DB-indeksom smatra se particijom s najprikladnijim brojem klastera [8].

3.3 Kriterij širine siluete

Kriterij širine siluete (engl. *Silhouette Width Criterion* (SWC)) definira se na sljedeći način:
Ako imamo optimalnu k-particiju $\Pi^* = \{\pi_1^*, \ldots, \pi_k^*\}$, tada se za svaki $x_i \in \mathcal{X} \cap \pi_r^*$ računaju brojevi

$$\alpha_{ir} = \frac{1}{|\pi_r^*|} \sum_{b \in \pi_r^*} d(x_i, b), \quad \beta_{ir} = \min_{q \neq r} \frac{1}{|\pi_q^*|} \sum_{b \in \pi_q^*} d(x_i, b)$$

i tada je odgovarajući SWC-indeks definiran s:

$$\mathrm{SWC}(k) = \frac{1}{m} \sum_{i=1}^{m} \frac{\beta_{ir} - \alpha_{ir}}{\max\{\alpha_{ir}, \beta_{ir}\}}.$$

Što je particija kompaktnija i klasteri bolje separirani, to je SWC-indeks veći. Particiju s najvećim SWC-indeksom smatramo particijom s najprikladnijim brojem klastera.

Često se zbog numeričke složenosti SWC-indeksa koristi Pojednostavljeni kriterij širine siluete (engl. *Simplified Silhouette Width Criterion* (SSC)). On umjesto prosječne vrijednosti koristi udaljenost od elementa $x_i \in \mathcal{X} \cap \pi_r^*$ do centara $\mu_1^*, \ldots, \mu_k^*$:

$$\alpha_{ir} = d(x_i, \mu_r^*), \quad \beta_{ir} = \min_{q \neq r} d(x_i, \mu_q^*)$$

i tada je odgovarajući SSC-indeks definiran s:

$$\mathrm{SSC}(k) = \frac{1}{m}\sum_{i=1}^{m}\frac{\beta_{ir}-\alpha_{ir}}{\max\{\alpha_{ir},\beta_{ir}\}}.$$

Particija s najvećim SSC-indeksom smatra se particijom s najprikladnijim brojem klastera [8].

Primjer 4. *Generirano je 1000 podataka u* $\mathbb{R}^2$. *Odredimo najprikladniji broj klastera primjenom prethodno spomenutih CH, DB i SWC-indeksa.*

Kod u Pythonu koji generira podatke, računa indekse i prikazuje odgovarajuće grafove:

```
import numpy as np
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import calinski_harabasz_score,
    davies_bouldin_score, silhouette_score
import matplotlib.pyplot as plt

X, y = make_blobs(n_samples=1000, centers=3, cluster_std=0.7,
    random_state=42)

num_clusters = range(2, 7)

ch_scores = []
db_scores = []
silhouette_scores = []

for k in num_clusters:
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(X)

    ch_score = calinski_harabasz_score(X, labels)
    ch_scores.append(ch_score)

    db_score = davies_bouldin_score(X, labels)
    db_scores.append(db_score)

    silhouette_avg = silhouette_score(X, labels)
    silhouette_scores.append(silhouette_avg)

    print(f'Broj klastera: {k}, CH-indeks: {ch_score:.2f}, DB-
    indeks: {db_score:.2f}, Silhouette indeks: {silhouette_avg:.2f}'
    )

plt.figure(figsize=(18, 6))

plt.subplot(1, 3, 1)
plt.plot(num_clusters, ch_scores, marker='o')
plt.title("Calinski-Harabasz indeks")
plt.xlabel("l")
plt.ylabel("CH(k)")
```

```
37
38 plt.subplot(1, 3, 2)
39 plt.plot(num_clusters, db_scores, marker='o', color='orange')
40 plt.title("Davies-Bouldin indeks")
41 plt.xlabel("k")
42 plt.ylabel("DB(k)")
43
44 plt.subplot(1, 3, 3)
45 plt.plot(num_clusters, silhouette_scores, marker='o', color='green'
       )
46 plt.title("Silhouette indeks")
47 plt.xlabel("k")
48 plt.ylabel("SWC(k)")
49
50 plt.tight_layout()
51 plt.show()
```

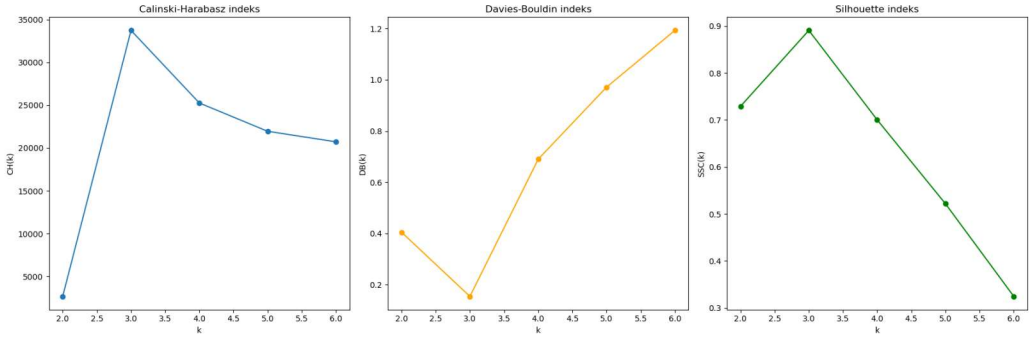
- podaci su raspoređeni u 3 klastera:

  Vrijednosti indeksa za $k = 2, 3, 4, 5, 6$ prikazani su u tablici 3.1. Sva tri indeksa ukazuju na to da se za $k = 3$ postiže particija s najprihvatljivijim brojem klastera.

	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
CH	2688.86	**33730.39**	25261.68	21956.82	20728.33
DB	0.40	**0.15**	0.69	0.97	1.19
SWC	0.73	**0.89**	0.70	0.52	0.32

Tablica 3.1: Vrijednosti indeksa za generirane podatke raspoređene u 3 klastera

Grafovi koji prikazuju ovisnost indeksa o broju k prikazani su na slici 3.1.



Slika 3.1: Indeksi za generirane podatke raspoređene u 3 klastera

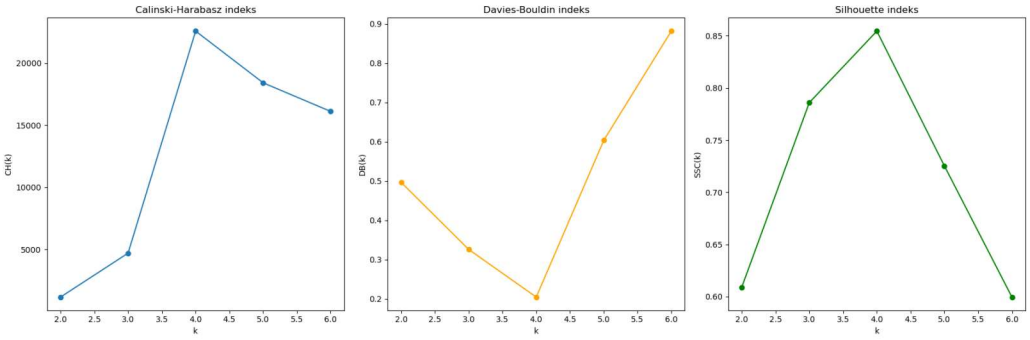
- podaci su raspoređeni u 4 klastera:

  Vrijednosti indeksa za $k = 2, 3, 4, 5, 6$ prikazani su u tablici 3.2. Sva tri indeksa ukazuju na to da se za $k = 4$ postiže particija s najprihvatljivijim brojem klastera.

	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
CH	1158.05	4699.21	**22585.78**	18418.12	16114.14
DB	0.50	0.33	**0.20**	0.60	0.88
SWC	0.61	0.79	**0.85**	0.73	0.60

Tablica 3.2: Vrijednosti indeksa za generirane podatke raspoređene u 4 klastera

Grafovi koji prikazuju ovisnost indeksa o broju k prikazani su na slici 3.2.



Slika 3.2: Indeksi za generirane podatke raspoređene u 4 klastera

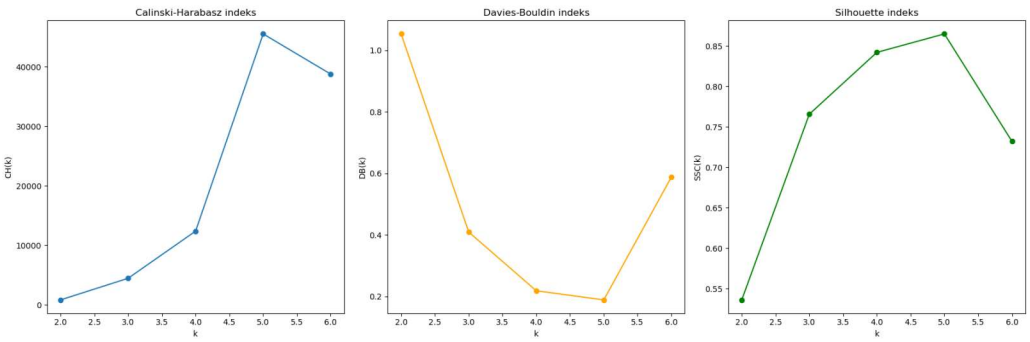
- podaci su raspoređeni u 5 klastera:

  Vrijednosti indeksa za $k = 2, 3, 4, 5, 6$ prikazani su u tablici 3.3. Sva tri indeksa ukazuju na to da se za $k = 5$ postiže particija s najprihvatljivijim brojem klastera.

	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
CH	822.95	4450.24	12378	**45499.68**	38748.64
DB	1.05	0.41	0.22	**0.19**	0.59
SWC	0.54	0.77	0.84	**0.86**	0.73

Tablica 3.3: Vrijednosti indeksa za generirane podatke raspoređene u 5 klastera

Grafovi koji prikazuju ovisnost indeksa o broju k prikazani su na slici 3.3.



Slika 3.3: Indeksi za generirane podatke raspoređene u 5 klastera

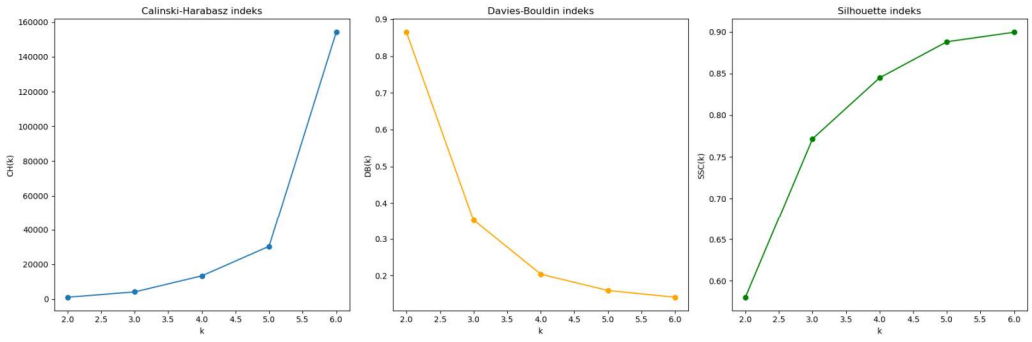
- podaci su raspoređeni u 6 klastera:

  Vrijednosti indeksa za $k = 2, 3, 4, 5, 6$ prikazani su u tablici 3.4. Sva tri indeksa ukazuju na to da se za $k = 6$ postiže particija s najprihvatljivijim brojem klastera.

	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
CH	1080.38	4145.30	13401.88	30393.13	**154147.42**
DB	0.86	0.35	0.20	0.16	**0.14**
SWC	0.58	0.77	0.84	0.89	**0.90**

Tablica 3.4: Vrijednosti indeksa za generirane podatke raspoređene u 6 klastera

Grafovi koji prikazuju ovisnost indeksa o broju k prikazani su na slici 3.4.



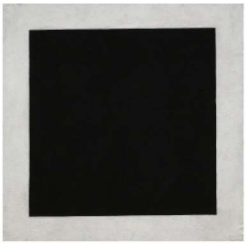
Slika 3.4: Indeksi za generirane podatke raspoređene u 6 klastera

4 | Segmentacija slike

Segmentacija slike važan je korak u mnogim algoritmima računalnog vida. Cilj segmentacije je podijeliti sliku u smislene regije. U idealnom slučaju, svaka regija u segmentiranoj slici trebala bi biti homogena obzirom na neke karakteristike ili značajke kao što su razina sive boje ili tekstura, a susjedne regije trebale bi imati značajno različite karakteristike ili značajke [3]. Glavni cilj segmentacije je pojednostaviti ili promijeniti prikaz slike na način da postane značajniji i lakše analiziran. Jedan od načina segmentiranja slika je putem klasteriranja. Koristi se k-means algoritam kako bi se grupirali pikseli sa sličnim karakteristikama (npr. boja, tekstura) u iste regije, tj. klastere. Također, možemo primijeniti i indekse opisane u prethodnom poglavlju koji nam u ovome slučaju govore o broju dominantnih nijansi na slici, tj. primjerenom broju klastera pri klasteriranju.

4.1 Segmentacija crno-bijele slike

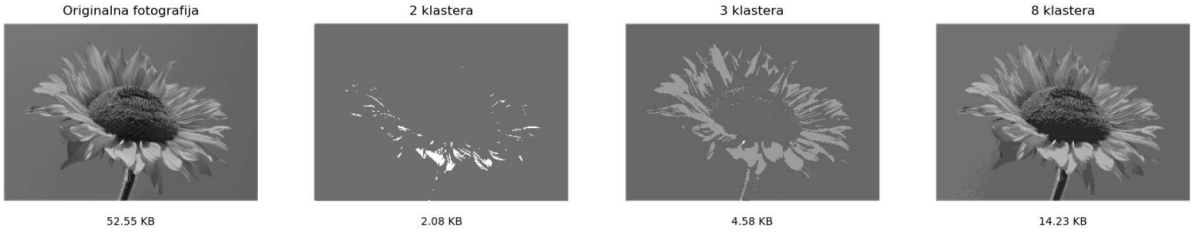
Segmentaciju crno-bijele slike shvaćamo kao klasteriranje u $\mathbb{R}^1$.
Crno-bijela slika, kao i svaka slika, sastoji se od piksela, a svaki piksel ima jednu vrijednost koja označava njegovu nijansu sive boje. Ova vrijednost varira od 0 (crna) do 255 (bijela) kada se koristi 8-bitna dubina (256 različitih nijansi sive). Dakle, svaki piksel se tretira kao jedna točka u $\mathbb{R}$ koja za vrijednost dobiva nijansu sive boje (gray level). Za segmentaciju može se koristiti k-means algoritam koji je ranije opisan.
Slika 4.1 prikazuje sliku "Crni kvadrat" umjetnika Kazimira Maljeviča. Pomoću navedene slike možemo uočiti najjednostavniji slučaj segmentacije, a to je segmentacija u dva klastera.



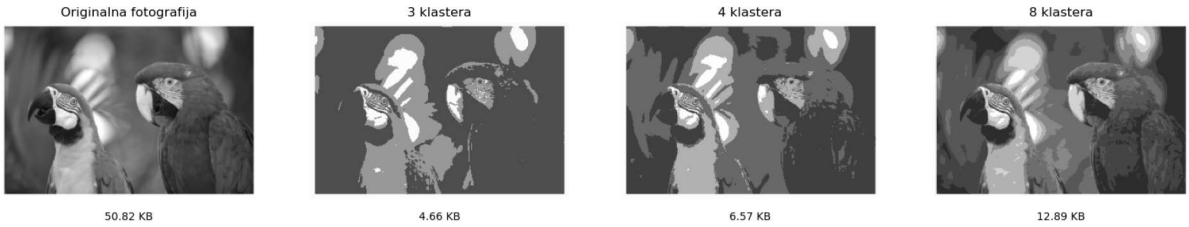
Slika 4.1: Crni kvadrat, $k = 2$

Ova slika (Slika 4.1) sastoji se isključivo od čiste crne boje unutar kvadrata i čiste bijele boje pozadine. U kontekstu klasteriranja, to znači da svaki piksel slike može

biti svrstan u jedan od dva klastera - crni ili bijeli. Ovu tvrdnju potvrđuju i izračunati indeksi (CH, DB i SWC) za primjereni broj klastera, tj. u ovom slučaju broj dominantnih nijansi, što je ovdje $k = 2$.
Slike 4.2 i 4.3 prikazuju segmentaciju 300×211 slike (Slika 4.2) i 300×204 (Slika 4.3) u 2, 3 i 8, odnosno 3, 4 i 8 klastera (nijansi). Ispod slika također pišu njihove veličine u KB, pri čemu je uočljiv efekt kompresije. U slučaju slike 4.2, imamo podatke $\mathcal{X} = \{x_i \in \mathbb{R} : i = 1, \ldots, 63300\}$, a za sliku 4.3 je $\mathcal{X} = \{x_i \in \mathbb{R} : i = 1, \ldots, 61200\}$. Redni broj (indeks) podataka x_i definira njegovu poziciju na slici. Za sliku 4.2 primjenom CH, DB i SWC indeksa dobivamo da je broj dominantnih nijansi, tj. broj primjerenih klastera jednak 3 (CH-indeks) i 2 (DB i SWC-indeks). S druge strane, za Sliku 4.3 dobivamo 4 (CH-indeks) i 3 (DB i SWC-indeks).



Slika 4.2: Segmentacija crno-bijele slike (primjer 1)



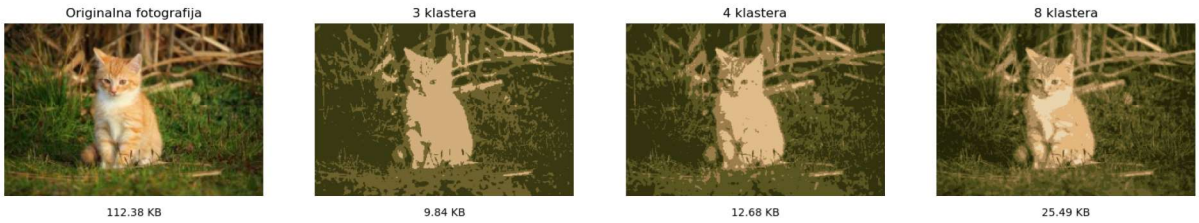
Slika 4.3: Segmentacija crno-bijele slike (primjer 2)

4.2 Segmentacija slike u boji

Segmentaciju slike u boji shvaćamo kao klasteriranje u $\mathbb{R}^3$.
Svaki piksel na slici predstavlja točku u trodimenzionalnom prostoru koji obuhvaća intenzitete crvene, plave i zelene komponente, a naš algoritam za segmentaciju tretira svaki piksel na slici kao zasebnu podatkovnu točku. Važno je napomenuti da ovaj prostor strogo gledano nije euklidski jer su intenziteti kanala ograničeni intervalom $[0, 1]$. Unatoč tome, možemo bez problema primijeniti k-means algoritam. Rezultat izvođenja k-means algoritma do konvergencije, za bilo koju određenu vrijednost k, prikazujemo tako da ponovno prikažemo sliku zamjenjujući svaki vektorski piksel s (R, G, B) intenzitetskom trojkom koja odgovara centru μ_k kojem je taj piksel dodijeljen. Algoritam za zadanu vrijednost k prikazuje sliku koristeći paletu od samo k boja [4].
Slike 4.4 i 4.5 prikazuju segmentaciju 300×332 slike (Slika 4.4) i 300×200 (Slika

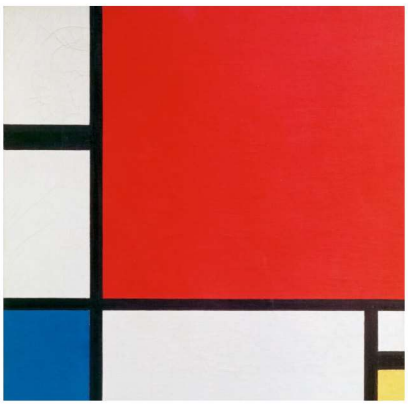
4.5) u 2, 5 i 8, odnosno 3, 4 i 8 klastera (nijansi). Za sliku 4.4 primjenom CH, DB i SWC indeksa dobivamo da je broj dominantnih nijansi, tj. broj primjerenih klastera jednak 5 (CH-indeks) i 2 (DB i SWC-indeks). S druge strane, za Sliku 4.5 dobivamo 4 (CH-indeks) i 3 (DB i SWC-indeks).

Slika 4.4: Segmentacija slike u boji (primjer 1)



Slika 4.5: Segmentacija slike u boji (primjer 2)

Još jedan primjer prikazan je na Slici 4.6 koja prikazuje sliku "Kompozicija s crvenom, crnom, žutom i plavom" umjetnika Pieta Mondriana. U ovom slučaju, primjereni broj klastera odgovara bojama koje se nalaze na slici - crvena, plava, žuta, crna i bijela, a to je 5. Svaki piksel na slici može se precizno svrstati u jedan od ovih klastera, budući da su boje jasno razdvojene i nema nijansi ili prijelaza između njih.



Slika 4.6: Kompozicija s crvenom, crnom, žutom i plavom, $k = 5$

Literatura

[1] C. C. Aggarwal, C. K. Reddy, *Data Clustering: Algorithms and Applications*, CRC Press, 2014.

[2] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers, 1981.

[3] J. C. Bezdek, J. Keller, R. Krisnapuram, N. R. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Springer, 1999.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[5] A. K. Jain, *Data Clustering: 50 Years Beyond K-Means*, Pattern Recognition Letters **31**(2010), 651–666.

[6] Y. Jung, H. Park, D. Z. Du, B. Drake, *A Decision Criterion for the Optimal Number of Clusters in Hierarchical Clustering*, Journal of Global Optimization **25**(2003), 91–111.

[7] F. Leisch, *A Toolbox for K-Centroids Cluster Analysis*, Computational Statistics & Data Analysis **51**(2006), 526–544.

[8] R. Scitovski, K. Sabo, *Klaster analiza i prepoznavanje geometrijskih objekata*, Sveučilište Josipa Jurja Strossmayera u Osijeku, Odjel za matematiku, Osijek, 2020.

Sažetak

U ovome završnom radu objašnjen je pojam klaster analize i primjena klaster analize na segmentaciju slike. Uvedeni su pojmovi nužni za shvaćanje osnovnih koncepata klasteriranja te je objašnjen k-means algoritam. Obrađeni su indeksi za određivanje primjerenog broja klastera te je na kraju prikazana segmentacija crnobijele slike i slike u boji. Potrebne funkcije za određivanje indeksa, segmentaciju i samo klasteriranje implementirane su u programskom jeziku Python.

Ključne riječi

klaster analiza, segmentacija slike, k-means algoritam, primjereni broj klastera, Python

Application of cluster analysis to image segmentation

Summary

In this bachelor thesis, the concept of cluster analysis and its application to image segmentation is explained. The terms necessary for understanding the basic concepts of clustering are introduced and the k-means algorithm is described. Indices for determining the most appropriate partition are discussed, and finally the segmentation of a black and white image and a color image is demonstrated. The necessary functions for determining the indices, for segmentation and for clustering itself are implemented in the Python programming language.

Keywords

cluster analysis, image segmentation, k-means algorithm, most appropriate partition, Python

Životopis

Rođena sam 2002. u Slavonskom Brodu. Pohađala sam Osnovnu školu "Antun Mihanović" od 2009. do 2017. godine. Gimnaziju "Matija Mesić", opći smjer, završila sam 2021. godine te iste godine upisujem sveučilišni preddiplomski studij Matematika i računarstvo na Odjelu za matematiku (sada Fakultet primijenjene matematike i informatike) Sveučilišta Josipa Jurja Strossmayera u Osijeku.