SLOVENSKÁ TECHNICKÁ UNIVERZITA

Fakulta informatiky a informačných technológií

v Bratislave

Umelá inteligencia

Zadanie 4

Adam Tomčala

Prednášajúci: Ing. Lukáš Kohútka, PhD.

Cvičiaci: Ing. Boris Slíž

Čas cvičení: Streda 15:0

AIS ID: 103 164

Adam Tomčala

Definovanie problému:

Máme 2D priestor, ktorý má rozmery X a Y, v intervaloch od -5000 do +5000. Tento 2D priestor vyplňte 20 bodmi, pričom každý bod má náhodne zvolenú polohu pomocou súradníc X a Y. Každý bod má unikátne súradnice (t.j. nemalo by byť viacej bodov na presne tom istom mieste).

Po vygenerovaní 20 náhodných bodov vygenerujte ďalších 20000 bodov, avšak tieto body nebudú generované úplne náhodne, ale nasledovným spôsobom:

1. Náhodne vyberte jeden zo **všetkých** doteraz vytvorených bodov v 2D priestore. Ak je bod príliš blízko okraju, tak zredukujete príslušný interval v nasledujúcich dvoch krokoch.
2. Vygenerujte náhodné číslo X_offset v intervale od -100 do +100
3. Vygenerujte náhodné číslo Y_offset v intervale od -100 do +100
4. Pridajte nový bod do 2D priestoru, ktorý bude mať súradnice ako náhodne vybraný bod v kroku 1, pričom tieto súradnice budú posunuté o X_offset a Y_offset

Vašou úlohou je naprogramovať zhlukovač pre 2D priestor, ktorý zanalyzuje 2D priestor so všetkými jeho bodmi a rozdelí tento priestor na k zhlukov (klastrov). Implementujte rôzne verzie zhlukovača, konkrétne týmito algoritmami:

- **k-means**, kde stred je **centroid**
- **k-means**, kde stred je **medoid**
- **aglomeratívne zhlukovanie**, kde stred je **centroid**
- **divízne zhlukovanie**, kde stred je centroid

Vyhodnocujte úspešnosť/chybovosť vášho zhlukovača. Za úspešný zhlukovač považujeme taký, v ktorom žiaden klaster nemá priemernú vzdialenosť bodov od stredu viac ako 500.

Vizualizácia: pre každý z týchto experimentov vykreslite výslednú 2D plochu tak, že označkujete (napr. vyfarbíte, očíslujete, zakrúžkujete) výsledné klastre.

Dokumentácia musí obsahovať opis konkrétne použitých algoritmov a reprezentácie údajov. V závere zhodnoťte dosiahnuté výsledky ich porovnaním.

AIS ID: 103 164

Adam Tomčala

Základné informácie o programe:

- Na implementáciu tohto zadanie som použil jazyk Python 3.9.
- V mojom programe som použil nasledujúce knižnice:

```
import random
import math
import matplotlib.pyplot as plot
import time
```

- Program je konzolová aplikácia, ktorá vizualizuje dáta pomocou knižnice matplotlib.
- Používateľ zadá číslo testu, ktorý sa má vykonať.
- Najskôr sa vytvorí požadovaný počet bodov a spustia sa pre tieto body jednotlivé algoritmy.

Generovanie bodov:

- Ako prvé sa mi náhodne vygeneruje 20 bodov zo zadanej plochy (-5000 - 5000).
- Následne vytváram x bodov (podľa zadaného testu) a to tak, že si vyberiem jeden bod už z vytvorených bodov a vytvorím náhodný bod v jeho blízkosti (offset -> -100 – 100).
- Pokračujem až kým nevytvorím požadovaný počet bodov.
- Na vytvorenie prvých inicializačných bodov mi slúži funkcia ***prve_body()***.

```
def prve_body(pocet):
    """
    Funkcia vytvara random body na ploche
    :param pocet:   Pocet, kolko bodov treba vytvorit.
    :return:        Body
    """
    body = []
    for i in range(pocet):

        while True:
            x = random.randrange(hranice[0], hranice[1] + 1)
            y = random.randrange(hranice[0], hranice[1] + 1)

            if [x, y] not in body:
                body.append([x, y])
                break

    return body
```

- Funkcia, ktorá v mojom programe vygeneruje požadovaný počet bodov sa nazýva ***vygeneruj_body()***. Ak sa náhodný už vytvorený bod nachádza príliš blízko okraju plochy, tak sa prepočíta offset, aby novovzniknutý bod ležal v danej ploche.

- Body sa nesmú opakovať.
- Hranice plochy a offsety mám zadefinované ako globálne premenné:

```
hranice = [-5000, 5000]
offset = [-100, 100]
```

- Funkcia na generovanie bodov a upravovanie offsetov:

```
def vygeneruj_body(pocet, body):
    """..."""
    for i in range(pocet):
        random_bod = body[random.randrange(0, len(body))]
        offsetx1, offsetx2 = offset[0], offset[1]
        offsety1, offsety2 = offset[0], offset[1]

        # prepocitanie offsetu, ak je bod za hranicou
        if offsetx1 + random_bod[0] < hranice[0]:
            offsetx1 += abs(offsetx1 + random_bod[0]) % hranice[1]

        if offsetx2 + random_bod[0] > hranice[1]:
            offsetx2 -= (offsetx2 + random_bod[0]) % hranice[1]

        if offsety1 + random_bod[1] < hranice[0]:
            offsety1 += abs(offsety1 + random_bod[1]) % hranice[1]

        if offsety2 + random_bod[1] > hranice[1]:
            offsety2 -= (offsety2 + random_bod[1]) % hranice[1]

        # Overovanie ci bod uz existuje
        while True:
            novy_bod = [0, 0]
            novy_bod[0] = random_bod[0] + random.randrange(offsetx1, offsetx2 + 1)
            novy_bod[1] = random_bod[1] + random.randrange(offsety1, offsety2 + 1)
            if novy_bod not in body:
                body.append(novy_bod)
                break

    return body
```

- Pre algoritmus k-means si osobitne vygenerujem počiatočné body, ktorých počet bude rovnaký ako počet klastrov. Vytváram si ich preto, aby algoritmus k-means začínal s rovnakými centrami zhlukov pre **centroid** aj **medoid.** Na toto mi v mojom programe slúži funkcia ***vytvor_pociatocne_body().***

Reprezentácia klastrov a ich centier:

- Samotný bod reprezentujem ako list **[x, y]**, kde **x** je x-ová súradnica bodu a **y** je y-ová súradnica bodu.
- Centrá zhklukov si udržujem v **liste**, je to v podstate list bodov:
  - [[x1, y1], [x2, y2], [x3, y3], ...]
- Klastre si udržujem taktiež v **liste** a to následovne:
  - Všetky klastre si udržujem v liste, ktorý obsahuje všetky klastre.
  - Každý klaster má body, ktoré do neho patria.
  - Znázornenie:

[

[[x1, y1], [x2, y2]]

[[x3, y3], [x4, y4], [x5, y5], ...]

[[x6, y6]]

[[x7, y7], [x8, y8], [x9, y9], [x10, y10], [x11, y11], [x12, y12], ...]

[[x13, y13], [x14, y14]]

...

...

]

Priradenie bodov do klastrov:

- V mojom programe mi na to priradenie jednotlivých bodov do klastrov slúži funkcia ***zafarbi_body().*** Túto funkciu využívam pri k-means algoritme a pri divizívnom algoritme.
- Funkcia zisťuje vzdialenosti jednotlivých bodov od všetkých centier zhlukov. Bod sa priradí do zhluku, podľa najmenšej vzdialenosti od centra daného zhluku.
- Na výpočet eulerovskej vzdialenosti jednotlivých bodov mi slúži funkcia ***zisti_vzdialenost().***
- Pri aglomeratívnom zhlukovaní priraďujem body inak, popis bude uvedený pri opise algoritmu.

Prepočet centier zhlukov:

- **Centroid:**
    - Centorid je neexistujúci bod, ktorého x-ové a y-ové súradnice sa vypočítajú ako priemerné x a y všetkých bodov v klastri.
    - V mojom programe mi na to slúži funkcia ***prepocitaj_centroidy()***
- **Medoid:**
    - Medoid je jeden bod z bodov klastra, ktorý má najmenšiu celkovú vzdialenosť od všetkých bodov v klastri.
    - V mojom programe mi na to slúži funkcia ***prepocitaj_medoidy()***.

Popisy jednotlivých algoritmov:

- **K-means (cetroid, medoid):**
    - Pri k-means algoritme si najskôr vytvorím požiadočné body ako prvotné centrá zhlukov.
    - Následne všetky body priradím do klastrov podľa toho, ku ktorému centru je daný bod najbližšie.
    - Po vytvorení nových klastrov opäť prerátavám centrá zhlukov (buď ako centroid alebo ako medoid).
    - Opäť priradím body do klastrov podľa vzdialenosti od jednotlivých centier.
    - Toto vykonávam dovtedy, dokým sa všetky novovytvorené centrá nebudú rovnať všetkým predchádzajúcim.
    - V mojom programe na toto slúži funkcia ***k_means()***.
- **Divízne zhlukovanie (centroid):**
    - Pri divíznom zhlukovaní sú na začiatku všetky body v jednom zhluku.
    - Tento prvotný zhluk všetkých bodov rozdelím pomocou k-means algoritmu na 2 zhluky. Potrebujem si vytvoriť dve centrá, pretože rozdeľujem zhluk na dva. Prvé centrum si vyberiem ako náhodný bod z daných bodov a druhé centrum zvolím bod, ktorý je najďalej od prvého centra. Na toto mi slúži funkcia ***najdi_najvzdialenejsi()***.
    - Následne body rozdelím do týchto 2 zhlukov.
    - Po rozdelení všetkých bodov do 2 zhlukov vytváram ďalšie zhluky. Ďalej idem deliť ten zhluk, ktorý má najväčšiu celkovú vzdialenosť bodov zhluku od ich centra. Na toto mi slúži funkcia ***najdi_najhorsi_klaster().***

  - Keď nájdem najhorší zhluk, opäť ho delím pomocou k-means algoritmu na 2 zhluky.
  - Toto opakujem až kým nemám požadovaný počet zhlukov.

- **Aglomeratívne zhlukovanie (centroid):**
  - Na začiatku algoritmu sa berie každý bod ako samostatný zhluk.
  - Vytvorím si list zhlukov a list centier zhlukov (na začiatku budú rovnaké, každý bod bude aj svojím centrom).
  - Následne si vytvorím maticu vzdialeností medzi jednotlivými cetrami (vytváram si iba trojuholníkovú maticu, pretože sú zbytočné duplicitné informácie).
  - Spájam tie zhluky, ktoré sú k sebe najbližšie. Na toto mi slúži funkcia ***najdi_najmensiu cestu().*** Funkcia prehľadáva maticu vzdialeností a vráti mi indexi centier.
  - Pomocou týchto indexov zlúčim dané body klastrov do jedného. Prepočítam nový centroid tohto zhluku. Následne odstránim z listu klastrov klastre, ktoré už neexistujú a pridám novovzniknutý klaster. Toto urobim aj pre list centier.
  - Následne aktualizujem maticu vzdialeností. Vymažem vzdialenosti, ktoré už neexistujú a naopak pridám nové.
  - Toto opakujem až kým nedostanem požadovaný počet klastrov.

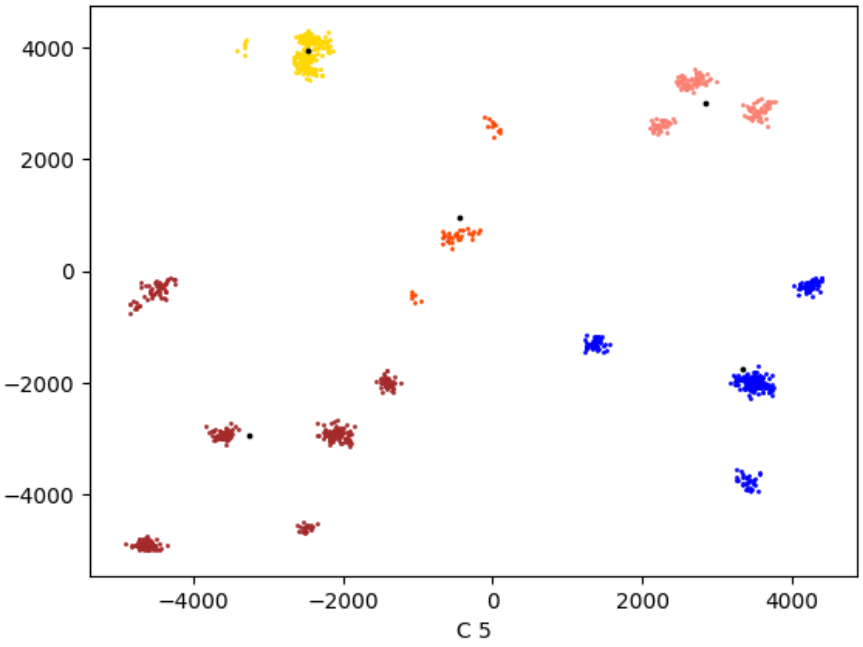
Testovanie:

- V mojom testovaní som testoval všetky na algoritmy na rovnakých bodoch.
- Testoval som následujúce možnosti:
  - 1 000 bodov + 20 inicializačných a 5 zhlukov
  - 2 500 bodov + 20 inicializačných a 10 zhlukov
  - 10 000 bodov + 20 inicializačných a 15 zhlukov
  - 20 000 bodov + 20 inicializačných a 20 zhlukov
- Pri všetkých možnostiach som pozoroval časy jednotlivých algoritmov, ich úspešnosť a následne som riešenie aj vizualizoval.
- Pri vizualizácií sú čierne bodky centrá jednotlivých zhlukov.

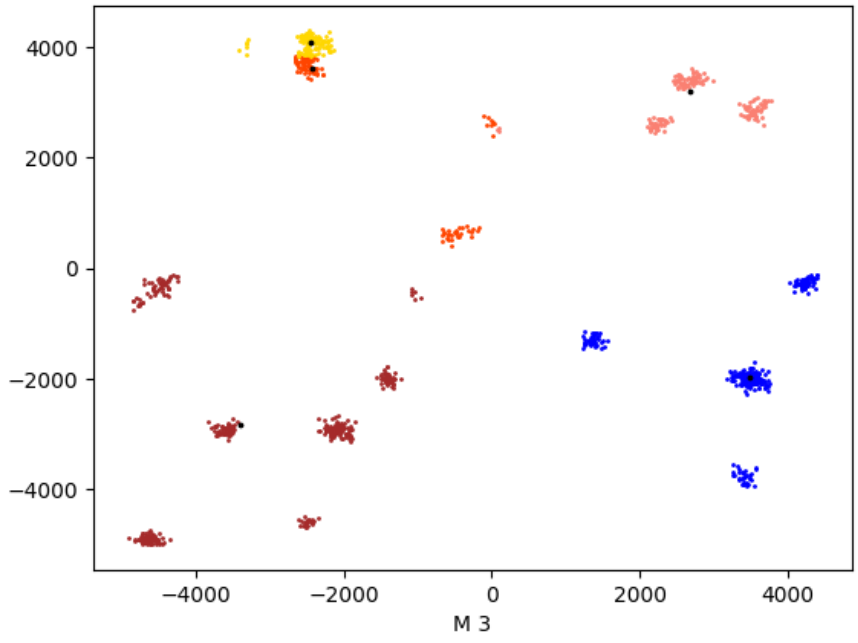
Vizualizácia jednotlivých testov (rovnaké body):

1 000 bodov a 5 zhlukov:

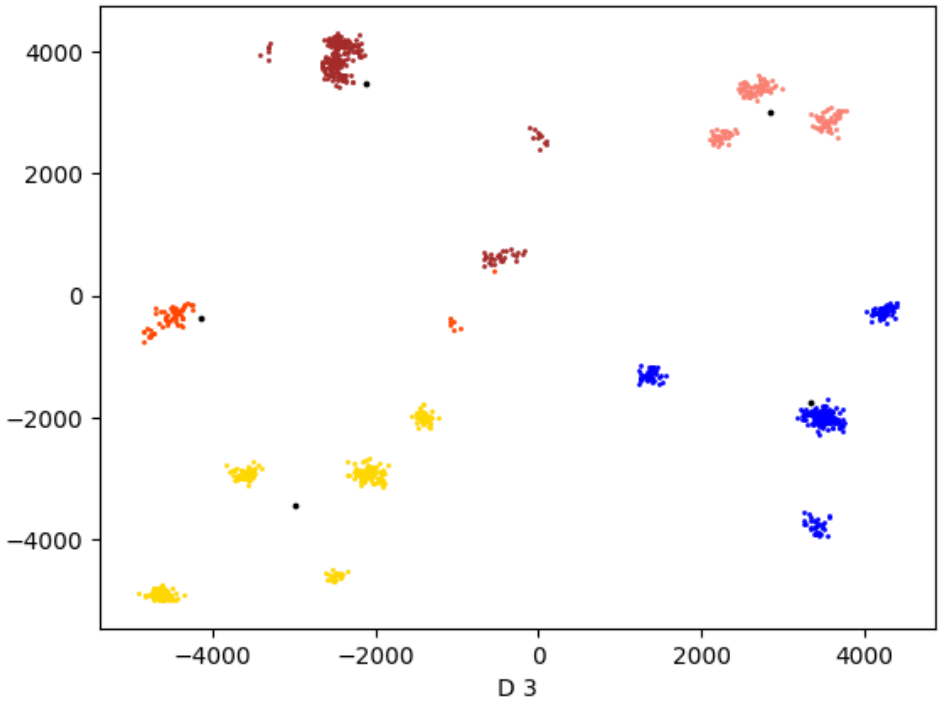
- **K-means (centroid):**



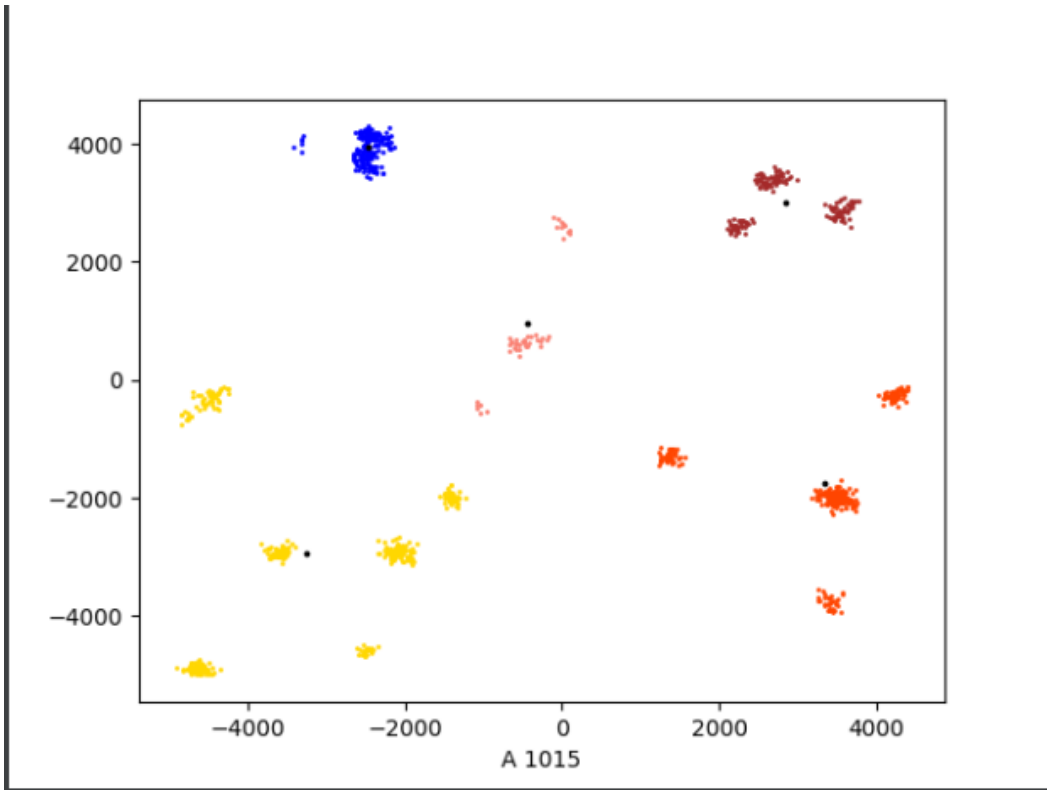
- **K-means (medoid):**



- **Divízne zhlukovanie (centroid):**



- **Aglomeratívne zhlukovanie (centroid):**

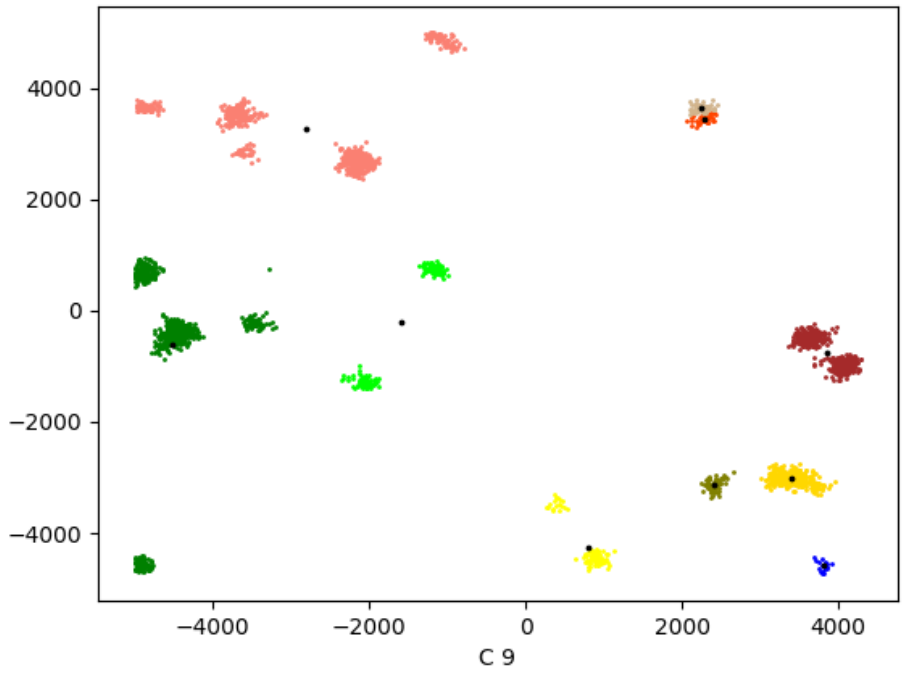


AIS ID: 103 164

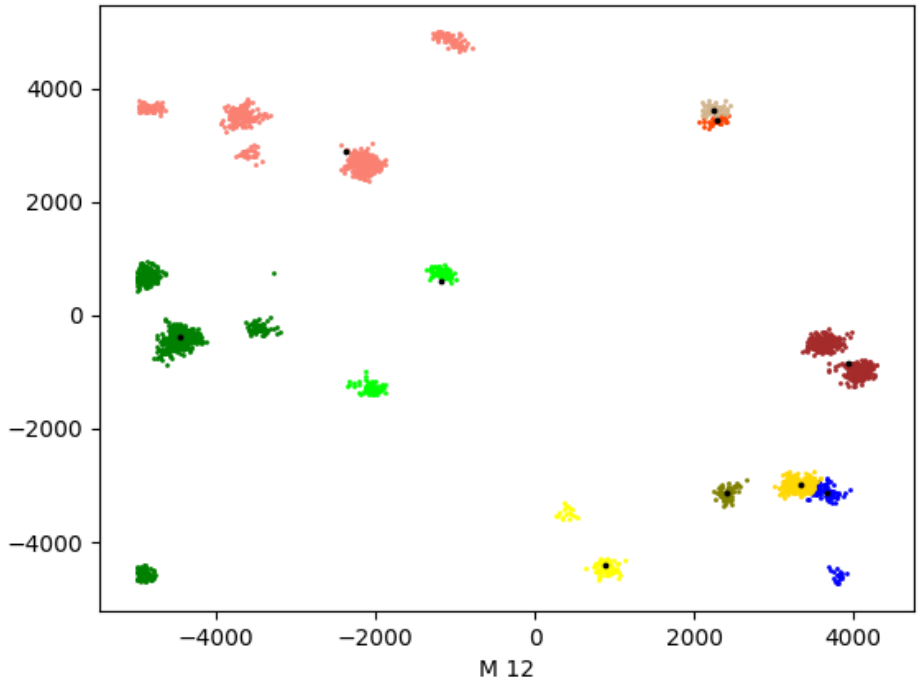
Adam Tomčala

2 500 bodov a 10 zhlukov:

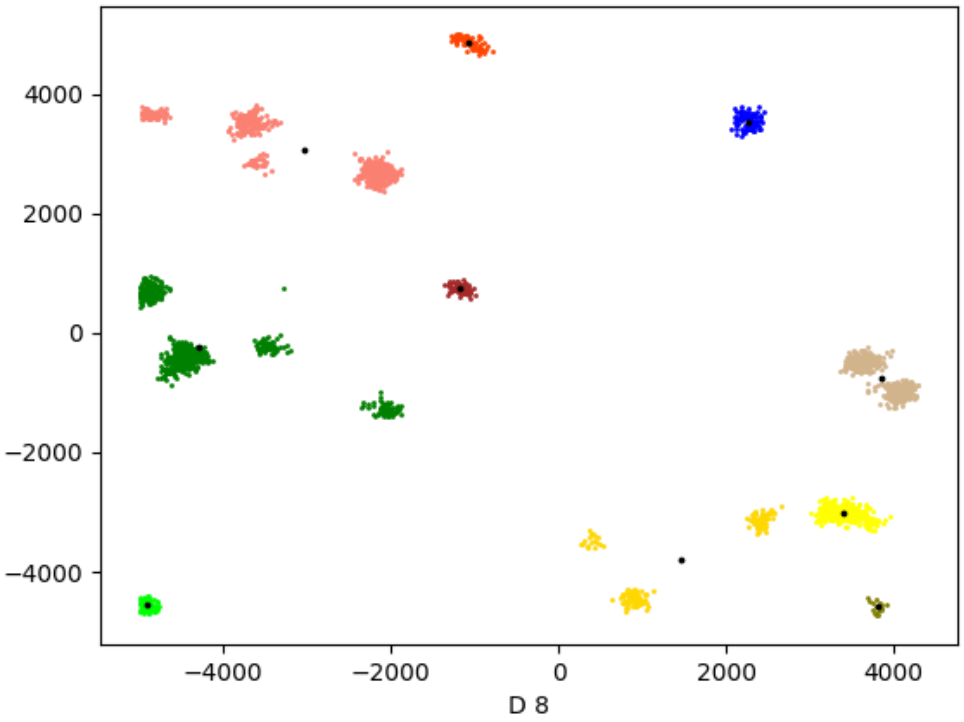
- **K-means (centroid):**



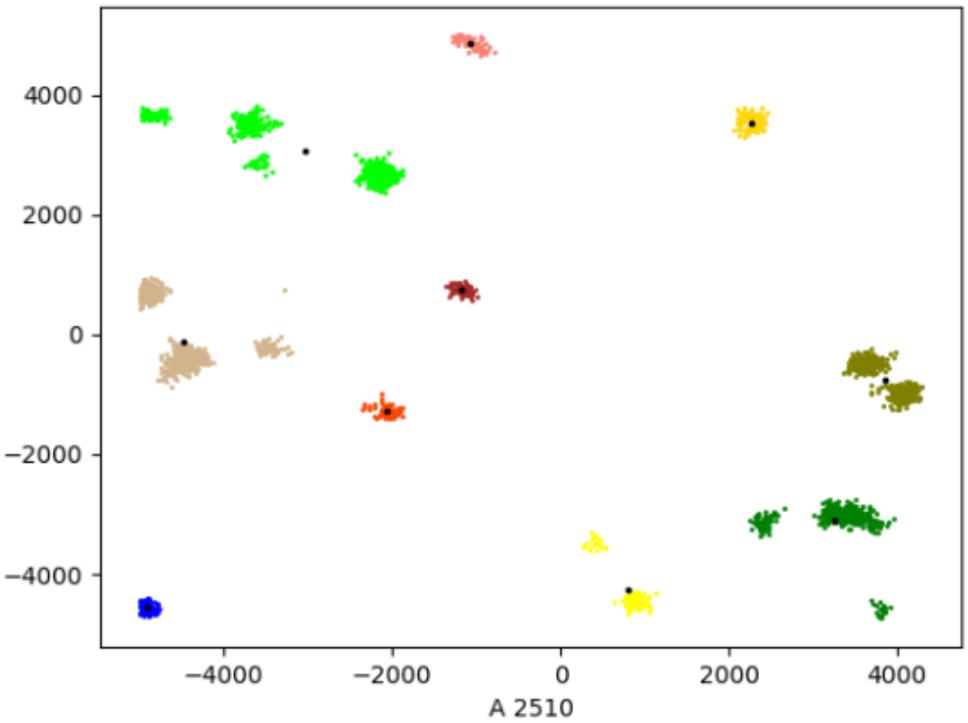
- **K-means (medoid):**



- **Divízne zhlukovanie (centroid):**



- **Aglomeratívne zhlukovanie (centroid):**

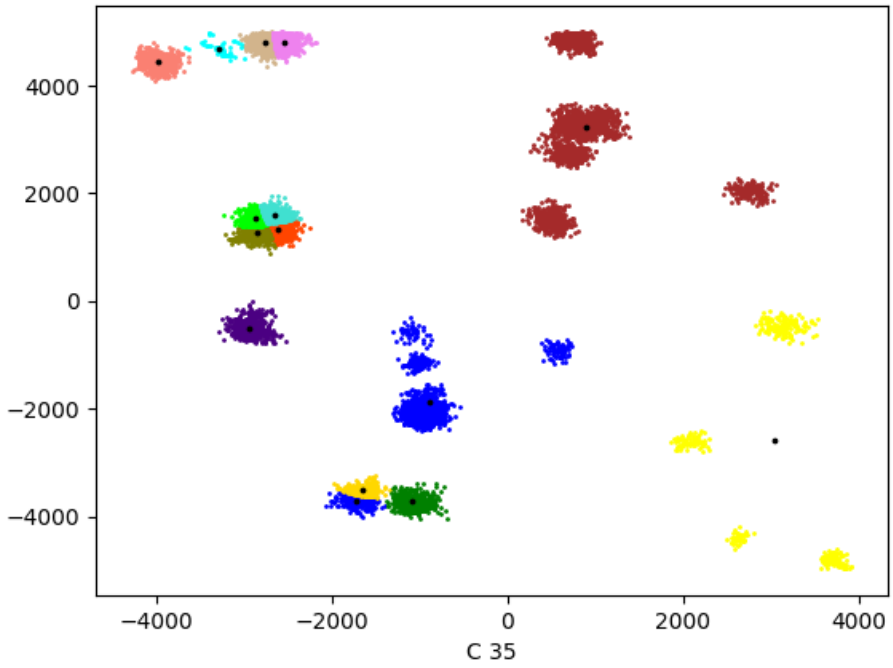


AIS ID: 103 164

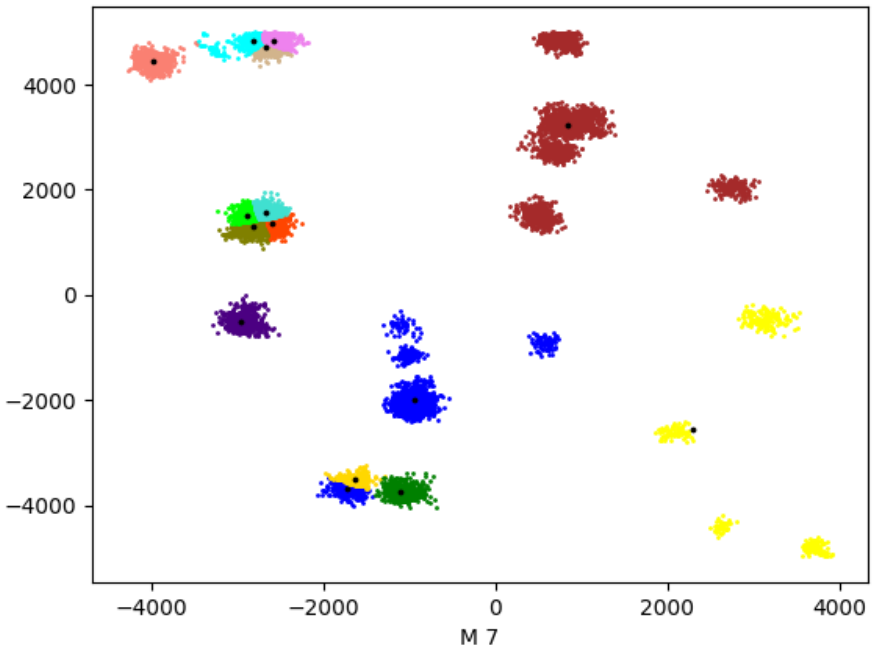
Adam Tomčala

10 000 bodov a 15 zhlukov:

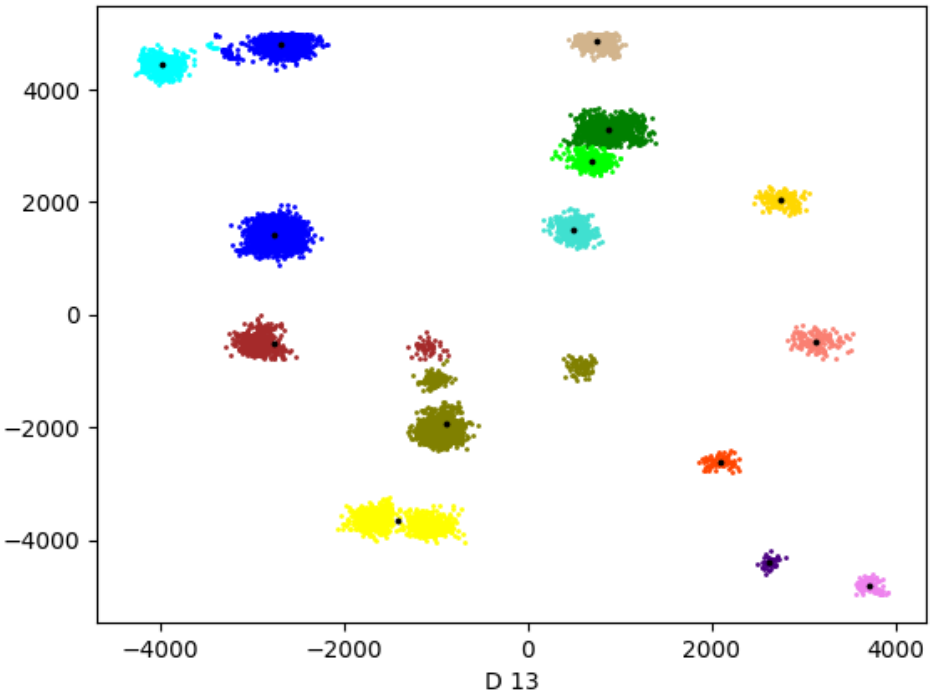
- **K-means (centroid):**



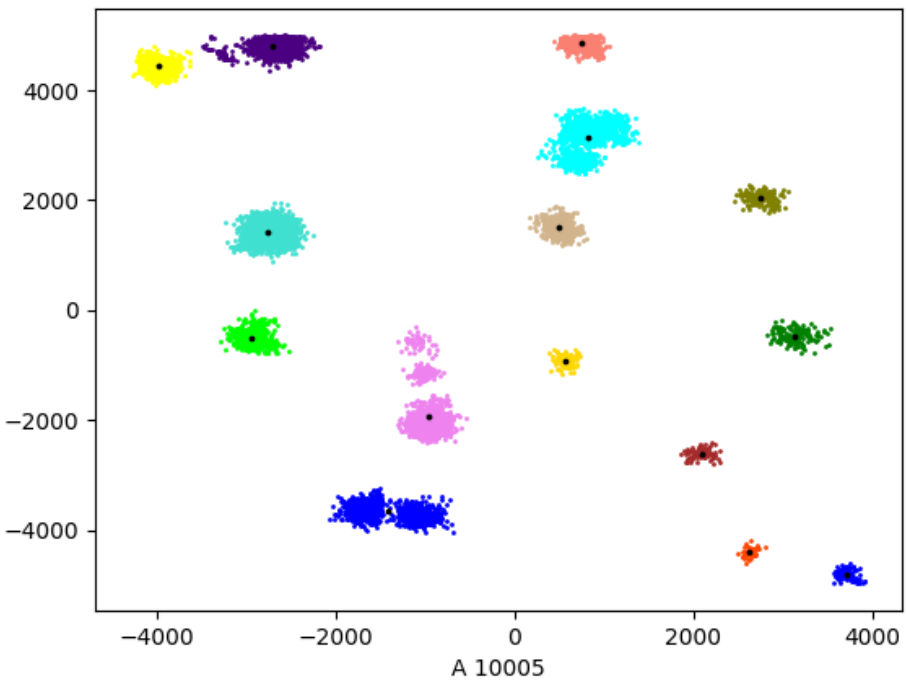
- **K-means (medoid):**



- **Divízne zhlukovanie (centroid):**



- **Aglomeratívne zhlukovanie (centroid):**

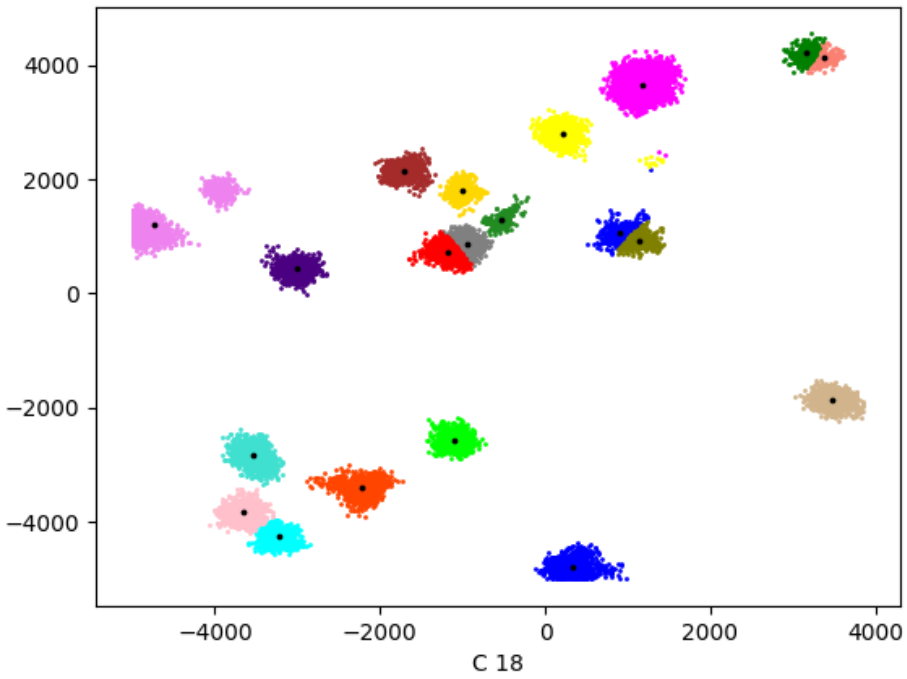


AIS ID: 103 164

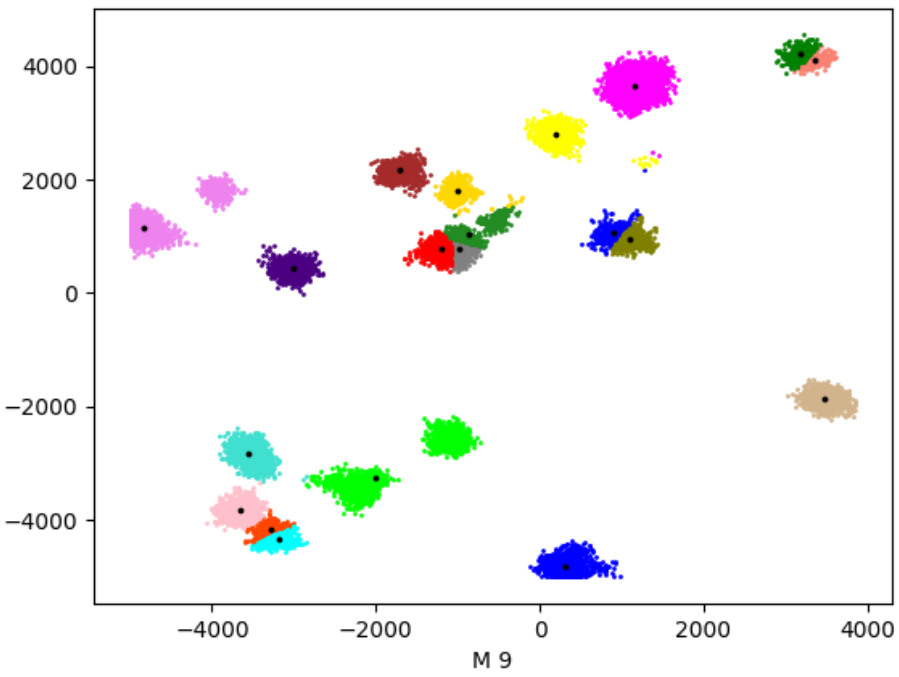
Adam Tomčala

20 000 bodov a 20 zhlukov:

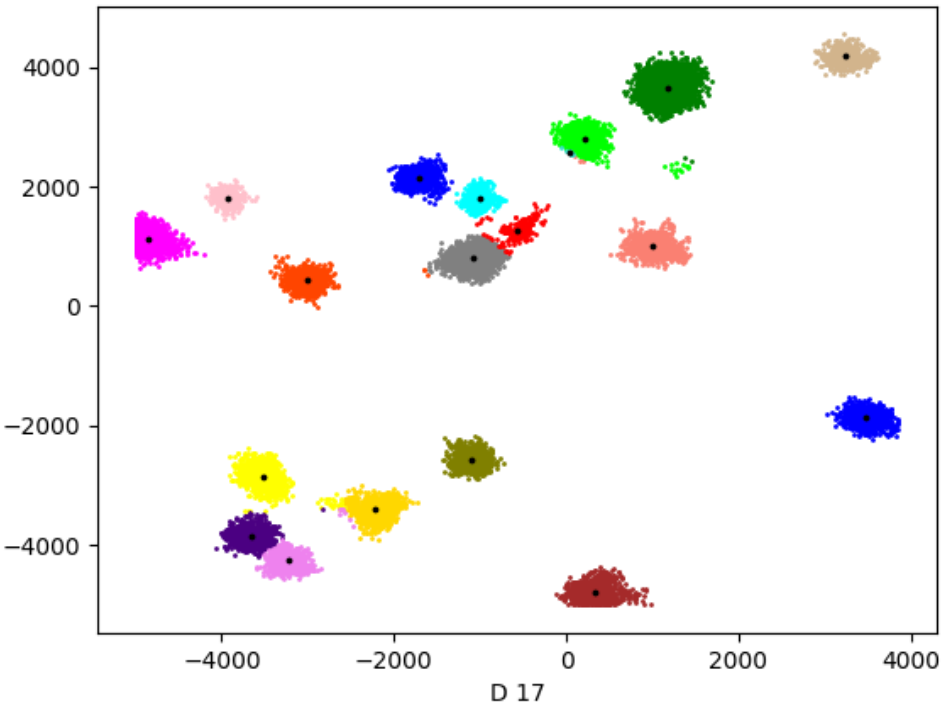
- **K-means (centroid):**



- **K-means (medoid):**



- **Divízne zhlukovanie (centroid):**



- **Aglomeratívne zhlukovanie (centroid):**
    - Pre tento počet bodov sa mi nepodarilo urobiť vizualizáciu. (Kvôli času)

Zhodnotenie testovania:

- Rýchlosť a úspešnosť algoritmov som testoval tak, že som spustil 10-krát každý algoritmus pre rôzne body a zistil priemer.
- Hodnoty su zobrazené v následujúcich tabuľkách:

- **1 000 bodov a 5 zhlukov (časy v sekundách):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
0,40	0,86	0,53	5,59
0,38	0,89	0,52	5,16
0,31	0,99	0,54	4,83
0,30	1,25	0,55	4,51
0,28	0,94	0,52	4,36
0,29	2,09	0,51	5,69
0,30	0,84	0,52	4,60
0,37	1,02	0,51	5,54
0,30	0,82	0,63	4,49
0,29	0,99	0,53	4,62
0,322	**1,069**	**0,536**	**4,939**

- **1 000 bodov a zhlukov (úspešnosť zhlukovača v %):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
44	45	28	44
16	41	14	23
31	55	31	15
26	50	28	21
33	57	22	47
44	69	45	27
19	53	27	31
40	61	19	29
16	52	18	16
24	58	22	37
29,3	**54,1**	**25,4**	**29**

- **2 500 bodov a 10 zhlukov (časy v sekundách):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
0,43	4,374	1,32	120,38
0,47	6,23	1,29	112,58
0,43	4,72	1,37	126,69
0,55	12,12	1,31	126,49
0,64	3,37	1,51	109,96
0,69	6,66	1,39	103,80
0,76	4,61	1,37	111,44
0,6	8,02	1,46	104,56
0,49	7,5	1,33	101,13
0,53	5,01	1,32	126,49
0,559	**6,261**	**1,367**	**114,352**

- **2 500 bodov a 10 zhlukov (úspešnosť zhlukovača v %):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
69	85	85	50
60	74	63	63
75	82	74	72
50	65	68	60
55	66	58	62
64	73	71	63
65	77	79	75
66	77	61	61
77	80	62	72
79	82	80	74
66	**76,1**	**70,1**	**65,2**

- **10 000 bodov a 15 zhlukov (časy v sekundách):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
1,90	69,61	3,17	Pre tento algoritmus som vykonával iba 1 test, pretože jeho trvanie bolo **7846,26 s.**
1,69	112,33	3,10	
3,89	71,83	3,18	
3,29	90,87	3,4	
3,48	63,68	3,17	
2,04	67,53	3,24	
2,58	58,71	3,22	
1,75	88,79	3,67	
2,37	51,2	3,3	
2,88	100,82	3,18	
2,587	**77,537**	**3,263**	

- **10 000 bodov a 15 zhlukov (úspešnosť zhlukovača v %):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
63	85	8z9	Úspešnosť tohto testu bola **96%.**
81	85	94	
75	87	99	
78	81	94	
86	89	93	
67	74	90	
79	88	84	
89	90	87	
82	88	92	
86	90	85	
78,6	**85,7**	**90,7**	

- **20 000 bodov a 20 zhlukov (časy v sekundách):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
8,19	230,96	7,41	**Test som nedokončil**
7,36	297,61	7,40	
14,57	340,21	7,43	
10,52	337,33	8,42	
18,89	173,42	6,34	
11,81	391,59	5,93	
12,94	511,83	6,71	
9,58	248,69	7,28	
10,91	298,56	8,12	
9,89	312,25	7,43	
11,466	**314,245**	**7,247**	

- **20 000 bodov a 20 zhlukov (úspešnosť zhlukovača v %):**

K-means (centorid)	K-means (medoid)	Divízne	Aglomeratívne
95	95	99	**Test som nedokončil**
98	93	98	
87	93	79	
90	84	96	
90	90	98	
85	86	99	
77	90	99	
82	89	96	
86	93	97	
91	92	95	
88,1	**90,5**	**95,6**	

Celkové zhodnotenie:

- Testovaním som zistil, ktorý z algoritmov je najrýchlejší. Najrýchlejšie algoritmy boli **k-means (centroid)** a **divízne zhlukovanie.** Najpomalší algoritmus bol **aglomeratívne zhlukovanie** (pre 20 000 bodov a 20 zhlukov som program spustil a bežal 9 hodín bez toho, aby som sa dostal k výsledku, preto neviem určiť jeho úspešnosť a priemerné trvanie).
- Čo sa týka uspešnosti zhlukovačov najúspešnejšie bolo **divízne a k-means (medoid).** Došiel som k záveru, že úspešnosť zhlukovačov nie je dobrá pri menšom počte bodov a klastrov, pretože plocha , na ktorej generujem body je dosť veľká a body generujem náhodne. Je preto veľká pravdepodobnosť (hlavne pri nižšom počte bodv), že vzdialenosť bodov od jeho centra bude väčšia ako **500.**
- Pri vyšších počtoch bodoch úspešnosť zhlukovačov rástla.

AIS ID: 103 164

Adam Tomčala