Slovenská technická univerzita
Fakulta informatiky a informačných technológií

Roman Bitarovský

Klastrovanie

Obsah

Zadanie

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:

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.

Vygenerujte náhodné číslo X_offset v intervale od -100 do +100

Vygenerujte náhodné číslo Y_offset v intervale od -100 do +100

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.

Opis použitého algoritmu

Ako prvé sa vždy vygeneruje náhodných 20 unikátnych bodov. K týmto bodom sa následne prigeneruje ďalších 20 000 bodov, kde už nesledujeme či sú unikátne alebo nie. Po vytvorení všetkých počiatočných bodoch nasleduje ich kastrovanie podľa príslušného algoritmu.

k-means

Tento algoritmus roztriedi body do klastrov. Počet klastrov spolu s bodmi sú vstupy pre algoritmus. Algoritmus funguje takže prvých k stredov pre k klastrov vyberie náhodne. Následne pre každý klaster vykoná prepočítanie stredu. Prepočítanie stredu spočíva vo spriemerovaní súradníc všetkých bodov daného klasteru. Následne je vykonané znovu pridelenie bodov do klastrov na základe toho k akému stredu majú najkratšiu vzdialenosť. Toto prepočítanie stredov a znovu zaradenie bodov do klastrov nechávam vykonať 100 krát nakoľko sa mi to pri testovaní osvedčilo ako bohato postačujúce.
Čo sa týka výpočtu spomínanej vzdialenosti tak nato sa využíva Euklidovská vzdialenosť.

Spomínaný stred reprezentujeme dvomi spôsobmi a to:

- centroid (vychádza priamo z výpočtu, je abstaktný)
- medoid (je reálny bod, taký ku kt, je najkratší súčet vzdialeností od ostatných bodov)

V programu sú implementované obe tieto verzie a možnosť vybrať si máme prostredníctvom menu pri spustení programu.

Rozdiely v súradniciach stredu pre príslušný výber môžeme pozorovať na nasledovnom výpisku.

Ako príklad uvádzam prvú a 10tu iteráciu (seed 500):

CENTROID

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 1
Iter: 0
Cluster: 0, centroid: x:-843.1462264150944 y:-2205.08679245283
Cluster: 1, centroid: x:-682.7317073170732 y:-3336.2195121951218
Cluster: 2, centroid: x:1554.6786951501156 y:-620.5424364896074
Cluster: 3, centroid: x:-5266.924126172208 y:4047.752770673487
Cluster: 4, centroid: x:-3453.9107913669063 y:795.6762589928057
Cluster: 5, centroid: x:-122.2474645030426 y:-2353.73630831643
Cluster: 6, centroid: x:-5161.805970149254 y:3660.4378109452737
Cluster: 7, centroid: x:-5781.718146718146 y:3620.876447876448
Cluster: 8, centroid: x:-4254.3893805309735 y:613.2566371681415
Cluster: 9, centroid: x:-5490.210344827587 y:3407.658620689655
Cluster: 10, centroid: x:2510.397918731417 y:1958.4350842418237
Cluster: 11, centroid: x:-4895.450505050505 y:-4104.642424242425
Cluster: 12, centroid: x:-3356.845947396672 y:-4170.814814814815
Cluster: 13, centroid: x:723.5110663983903 y:-3238.714285714286
Cluster: 14, centroid: x:-2740.0576923076924 y:-3643.3543956043954
Cluster: 15, centroid: x:754.084375 y:-2063.061458333333
Cluster: 16, centroid: x:2857.0099140779907 y:-2309.203569068077
Cluster: 17, centroid: x:241.9493670886076 y:-4644.265031645569
Cluster: 18, centroid: x:-4671.928816466552 y:-4435.713550600343
Cluster: 19, centroid: x:120.7874251497006 y:-2694.0808383233534
```

MEDOID

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 2
Iter: 0
Cluster: 0, medoid: x:-710 y:-2530
Cluster: 1, medoid: x:-634 y:-3319
Cluster: 2, medoid: x:1585 y:-607
Cluster: 3, medoid: x:-5290 y:4026
Cluster: 4, medoid: x:-3604 y:922
Cluster: 5, medoid: x:-132 y:-2397
Cluster: 6, medoid: x:-5185 y:3674
Cluster: 7, medoid: x:-5736 y:3637
Cluster: 8, medoid: x:-4220 y:634
Cluster: 9, medoid: x:-5470 y:3451
Cluster: 10, medoid: x:2611 y:1842
Cluster: 11, medoid: x:-4874 y:-4121
Cluster: 12, medoid: x:-3340 y:-4181
Cluster: 13, medoid: x:738 y:-3225
Cluster: 14, medoid: x:-2800 y:-3646
Cluster: 15, medoid: x:717 y:-2108
Cluster: 16, medoid: x:2891 y:-2270
Cluster: 17, medoid: x:257 y:-4672
Cluster: 18, medoid: x:-4678 y:-4369
Cluster: 19, medoid: x:97 y:-2654
```

```
Iter: 9
Cluster: 0, centroid: x:-1363.4246575342509 y:-555.1712328767184
Cluster: 1, centroid: x:-825.8014323186806 y:-3249.7064646118843
Cluster: 2, centroid: x:1639.5445252288089 y:-335.7748458024368
Cluster: 3, centroid: x:-5148.380826911011 y:4145.104444679411
Cluster: 4, centroid: x:-3533.5807068174736 y:1122.9253929655363
Cluster: 5, centroid: x:-317.93075042091925 y:-2346.2929546361124
Cluster: 6, centroid: x:-5295.523278169726 y:3743.831794745333
Cluster: 7, centroid: x:-5678.727970493314 y:3882.913205035209
Cluster: 8, centroid: x:-3965.297763697692 y:631.547333761933
Cluster: 9, centroid: x:-5687.096511273333 y:3365.9220254913002
Cluster: 10, centroid: x:3246.455882349037 y:3459.9372548994534
Cluster: 11, centroid: x:-4536.201500537129 y:-4147.46409431767
Cluster: 12, centroid: x:-3459.2626939420966 y:-4290.857529567002
Cluster: 13, centroid: x:628.5603812591964 y:-3627.5973939041623
Cluster: 14, centroid: x:-2960.263602195609 y:-3817.2359337136436
Cluster: 15, centroid: x:871.2577671869084 y:-1774.3737568598008
Cluster: 16, centroid: x:2858.9801190009216 y:-2309.9522861619994
Cluster: 17, centroid: x:25.688144459683194 y:-5106.804182319784
Cluster: 18, centroid: x:-5043.652421650464 y:-4539.814814817163
Cluster: 19, centroid: x:-43.00509777907853 y:-2719.9684848817874
```

```
Iter: 9
Cluster: 0, medoid: x:-933 y:-2533
Cluster: 1, medoid: x:-742 y:-3507
Cluster: 2, medoid: x:1666 y:-444
Cluster: 3, medoid: x:-5154 y:4130
Cluster: 4, medoid: x:-3538 y:1105
Cluster: 5, medoid: x:-42 y:-2341
Cluster: 6, medoid: x:-5318 y:3772
Cluster: 7, medoid: x:-5643 y:3843
Cluster: 8, medoid: x:-3946 y:671
Cluster: 9, medoid: x:-5688 y:3389
Cluster: 10, medoid: x:3192 y:3949
Cluster: 11, medoid: x:-4553 y:-4148
Cluster: 12, medoid: x:-3449 y:-4263
Cluster: 13, medoid: x:589 y:-3528
Cluster: 14, medoid: x:-3004 y:-3845
Cluster: 15, medoid: x:751 y:-2053
Cluster: 16, medoid: x:2891 y:-2270
Cluster: 17, medoid: x:52 y:-5112
Cluster: 18, medoid: x:-5009 y:-4502
Cluster: 19, medoid: x:-342 y:-2816
```

Divízne zhlukovanie, kde stred je centroid

Tento algoritmus využíva už implementovaný k-means so stredom ako centroid a to tak, že na začiatku sa všetky body (akoby jeden klaster) rozdelia na dva klastre. Následne ak aktuálny počet klastrov nie je ten požadovaný vyberie sa z existujúcich klastrov ten, kt. má najväčšiu priemernú vzdialenosť bodov s stredu a ten sa zase pomocou k-means nechá rozdeliť na dva samostatné klastre. To sa opakuje až kým počet existujúcich klastrov nie je rovný požadovanému počtu klastrov.

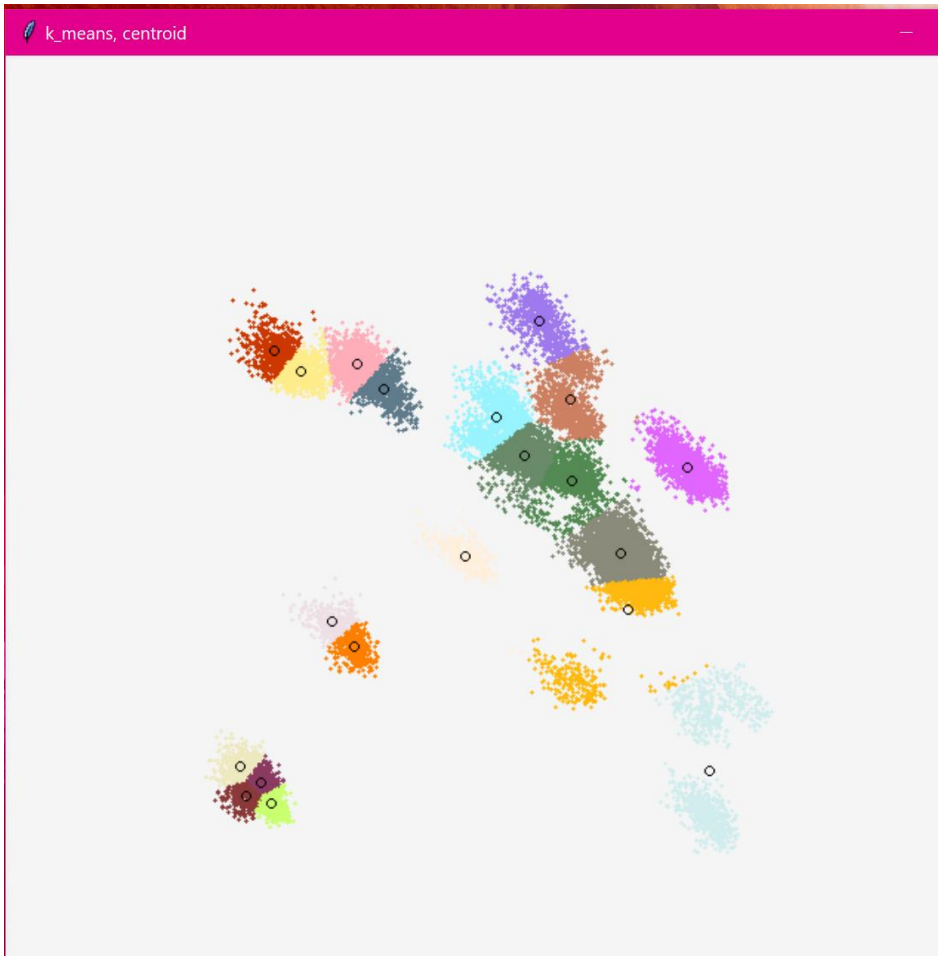
Aglomeratívne zhlukovanie, kde stred je centroid

Funguje naopak ako divízne zhlukovanie, začína tak že n bodov tvorí n klastrov. Následne sa hľadajú dva k sebe najbližšie klastre a tie sa zlúčia do jedného klastru. Toto sa opakuje až kým nedosiahneme požadovaný počet klastrov.

Testovanie

k-means, kde stred je centroid

Príklad vizualizácie pre 20 klastrov, 20k bodov, seed: 500.

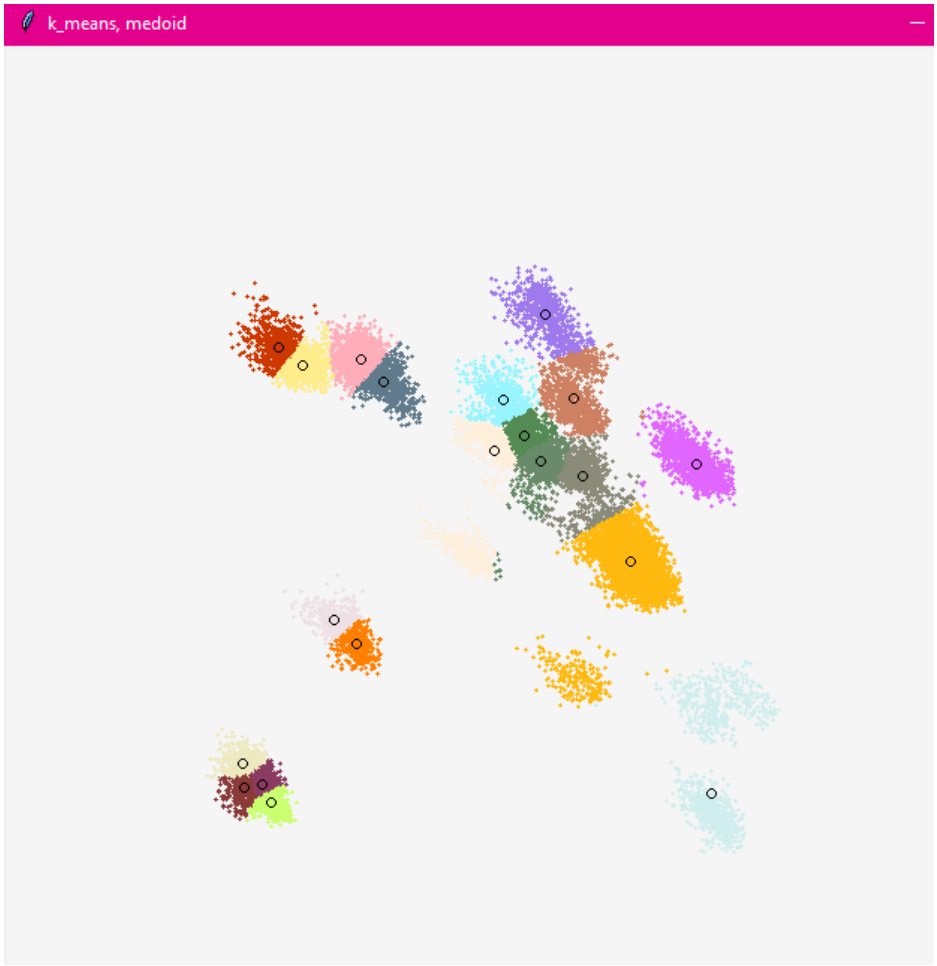


10 experimentov (20k bodov) z toho 1 - 100% úspešný, ostatné 100% úspešnosť nedosiahli

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 1
1: Pocet dobrych clusterov: 17, pocet zlych clusterov: 3, globalny priemer vzdianosti: 381.9143
2: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 367.3773
3: Pocet dobrych clusterov: 19, pocet zlych clusterov: 1, globalny priemer vzdianosti: 369.7262
4: Pocet dobrych clusterov: 16, pocet zlych clusterov: 4, globalny priemer vzdianosti: 440.5384
5: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 394.4092
6: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 391.5924
7: Pocet dobrych clusterov: 16, pocet zlych clusterov: 4, globalny priemer vzdianosti: 391.2313
8: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 336.2297
9: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 371.331
10: Pocet dobrych clusterov: 15, pocet zlych clusterov: 5, globalny priemer vzdianosti: 406.0193
Time: 0.8469 min
```

k-means, kde stred je centroid

Príklad vizualizácie pre 20 klastrov, 20k bodov, seed: 500.

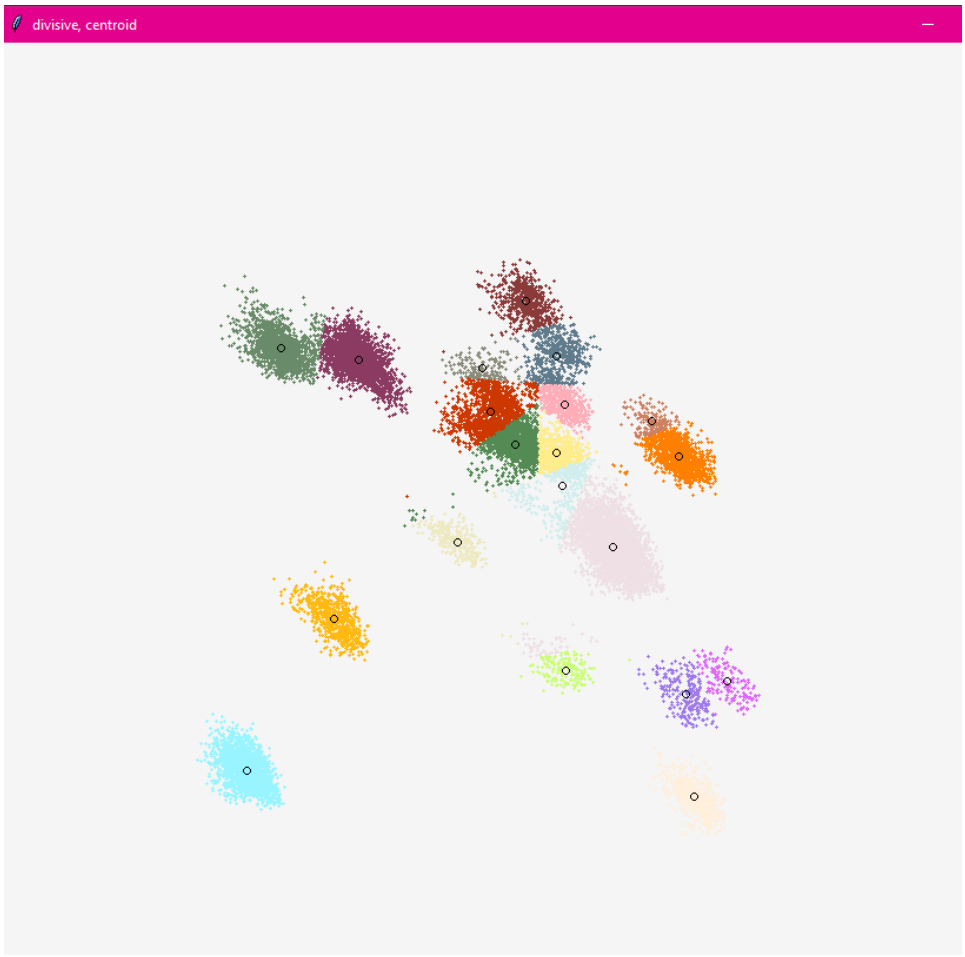


10 experimentov (20k bodov), žiadny nebol 100% úspešný

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 2
1: Pocet dobrych clusterov: 17, pocet zlych clusterov: 3, globalny priemer vzdianosti: 383.7972
2: Pocet dobrych clusterov: 19, pocet zlych clusterov: 1, globalny priemer vzdianosti: 361.6318
3: Pocet dobrych clusterov: 17, pocet zlych clusterov: 3, globalny priemer vzdianosti: 383.2699
4: Pocet dobrych clusterov: 15, pocet zlych clusterov: 5, globalny priemer vzdianosti: 438.7974
5: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 371.7435
6: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 371.4014
7: Pocet dobrych clusterov: 17, pocet zlych clusterov: 3, globalny priemer vzdianosti: 359.8749
8: Pocet dobrych clusterov: 17, pocet zlych clusterov: 3, globalny priemer vzdianosti: 354.7171
9: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 367.4242
10: Pocet dobrych clusterov: 15, pocet zlych clusterov: 5, globalny priemer vzdianosti: 358.6512
Time: 36.0939 min
```

Divízne zhlukovanie, kde stred je centroid

Príklad vizualizácie pre 20 klastrov, 20k bodov, seed: 500.



10 experimentov (20k bodov), všetky 100% úspešné

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 3
1: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 354.2443
2: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 371.3771
3: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 327.1129
4: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 370.3785
5: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 377.4436
6: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 351.5521
7: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 372.7273
8: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 338.6695
9: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 367.3377
10: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 356.0514
Time: 4.0941 min

Process finished with exit code 0
```

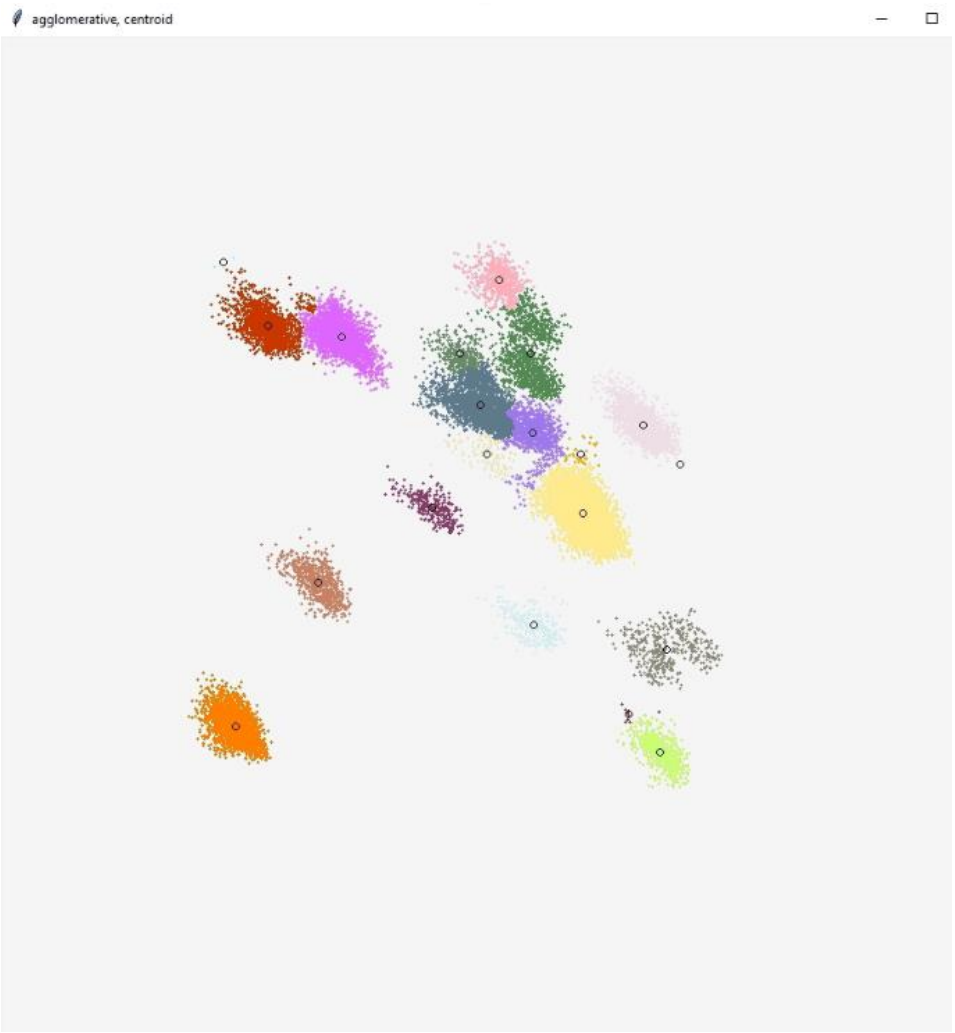
Aglomeratívne zhlukovanie, kde stred je centroid

20 klastrov, 15k bodov, seed: 500.

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 4
1: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 311.7865
Time: 523.9655 min
```

Príklad vizualizácie pre 20 klastrov, 20k bodov, seed: 500.

```
C:\Windows\py.exe
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 4
1: Pocet dobrych clusterov: 18, pocet zlych clusterov: 2, globalny priemer vzdianosti: 347.3019
Time: 1410.5932 min
```



10 experimentov (5k bodov) z toho 8 bolo 100% úspešné

```
Pycharm starting..
1) K-Means with centroid
2) K-Means with medoid
3) Divisive clustering
4) Agglomerative clustering
Your choice: 4
1: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 271.2145
2: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 290.6575
3: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 301.1951
4: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 291.6243
5: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 278.0839
6: Pocet dobrych clusterov: 19, pocet zlych clusterov: 1, globalny priemer vzdianosti: 297.9293
7: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 296.654
8: Pocet dobrych clusterov: 19, pocet zlych clusterov: 1, globalny priemer vzdianosti: 316.273
9: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 287.854
10: Pocet dobrych clusterov: 20, pocet zlych clusterov: 0, globalny priemer vzdianosti: 241.3762
Time: 171.2476 min
```

Zhodnotenie testovania

Z testovania hodnotím, že program funguje správne a dokáže vypočítať všetky prípady použitia pre 20000 bodov.

Ako najrýchlejší a najefektívnejší sa mi ukázal prístup divízneho zhlukovania. To dokáže za pomerne krátky čas priniesť veľmi úspešné výsledky.

Čo sa týka porovnania k-means podľa výberu stredu výsledky nie sú úplne jednoznačné avšak vzhľadom na čas konštatujem, že použitie cendroidu je oveľa lepšia cesta. Nedáva síce najlepšie výsledky to však vieme upraviť zvýšením parametru zodpovedajúcemu počtu koľko krát sa prepočítajú dané stredy. Toto by sme vedeli urobiť aj v prípade k-means medoid avšak z časového hľadiska je potom algoritmus veľmi náročný.

Čo sa týka aglomeratívneho zhlukovania tak to hodnotím ako najhoršie vzhľadom na jeho obrovskú časovú náročnosť. Dosahuje síce najlepšie hodnoty čo sa výsledku týka avšak pomer čas a výsledok viac nahráva divíznemu zhlukovaniu.

Zadania je spracované v jazyku python 3.9.5 v prostredí PyCharm a je spustiteľné ak máme funkčné všetky balíčky uvedené na začiatku kódu.