SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

ZADANIE 4 – Klastrovanie

Meno a priezvisko: Marko Stahovec

Dátum vypracovania: 29.11.2021

Cvičiaci: Ing. Boris Slíž

Meno: Marko Stahovec
ID: 110897

Meno: Marko Stahovec
ID: 110897

1. Zadanie

Našou úlohou bola implementácia niekoľkých **klastrovacích algoritmov na zhlukovanie dát v 2D priestore**. Klastrovanie je vo svojej podstate priradenie množiny pozorovaní do podmnožín tak, že pozorovania v rovnakej podmnožine sú istým spôsobom prepojené.

Identifikácia podobnosti sa ráta vzdialenosťou v priestore medzi dvoma bodmi, z čoho vyplýva, že s klesajúcou vzdialenosťou medzi dvojicou bodov rastie pravdepodobnosť, že daná dvojica bude patriť do jedného zhluku.

2. Opis algoritmov

Algoritmov na klastrovanie je niekoľko, no v tejto implementácii možno algoritmy deliť na **centrálne** a **hierarchické**. Centrálne algoritmy využívajú istú formu **centra** resp. **stredu**, ku ktorému sa priradzujú dáta, ktoré k nemu patria. Tento stred môže byť relatívny (centroid) alebo skutočný (medoid).

Centroid je bod, ktorý je vypočítaný ako priemer zo súradníc dát, ktoré k nemu patria, a teda výsledná hodnota jeho súradníc nemusí ukazovať na konkrétny záznam z dát. **Medoid** je vo svojej podstate najstrednejšia hodnota z daného zhluku, teda bod, od ktorého sú všetky ostatné body v rámci jedného zhluku vzdialené najmenej.

Hierarchické algoritmy môžu taktiež využívať istú formu centier, či už centroidov alebo medoidov, no tieto algoritmy pracujú primárne so vzdialenosťou medzi bodmi na globálnej úrovni, vďaka čomu berú v úvahu **distribúciu a hustotu rozptylu dát** na úkor časovej a pamäťovej náročnosti.

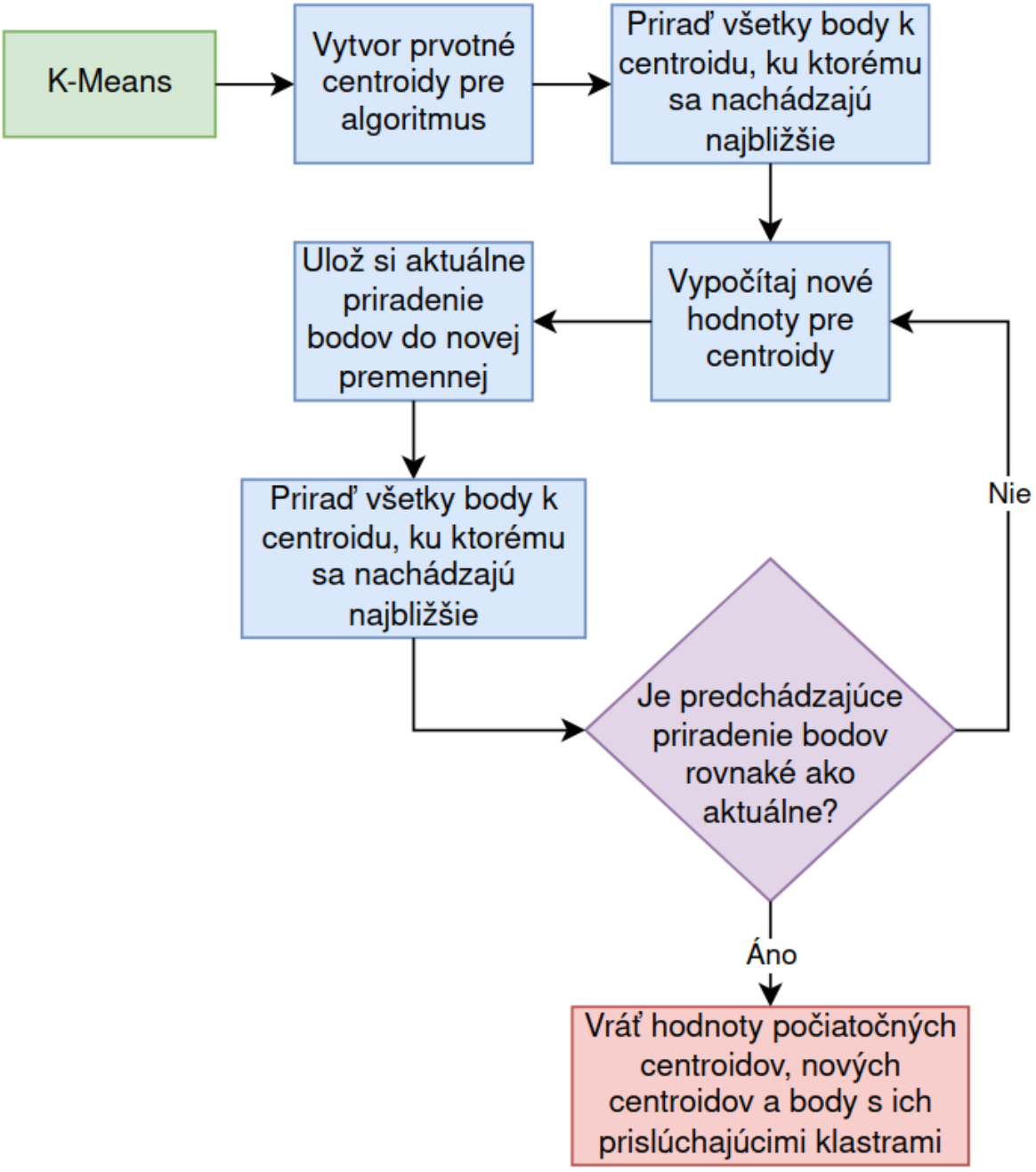
Cieľom klastrovacích algoritmov je stav, v ktorom sú všetky dáta identifikované a rozdelené do zhlukov, v ktorých sú len tie dáta, ktoré si sú navzájom podobné.

Algoritmy, ktoré bolo potrebné implementovať:

- **k-means**, kde stred je centroid
- k-means, kde stred je medoid, tzv. **k-medoid**
- **aglomeratívne zhlukovanie**, kde stred je centroid
- **divizívne zhlukovanie**, kde stred je medoid

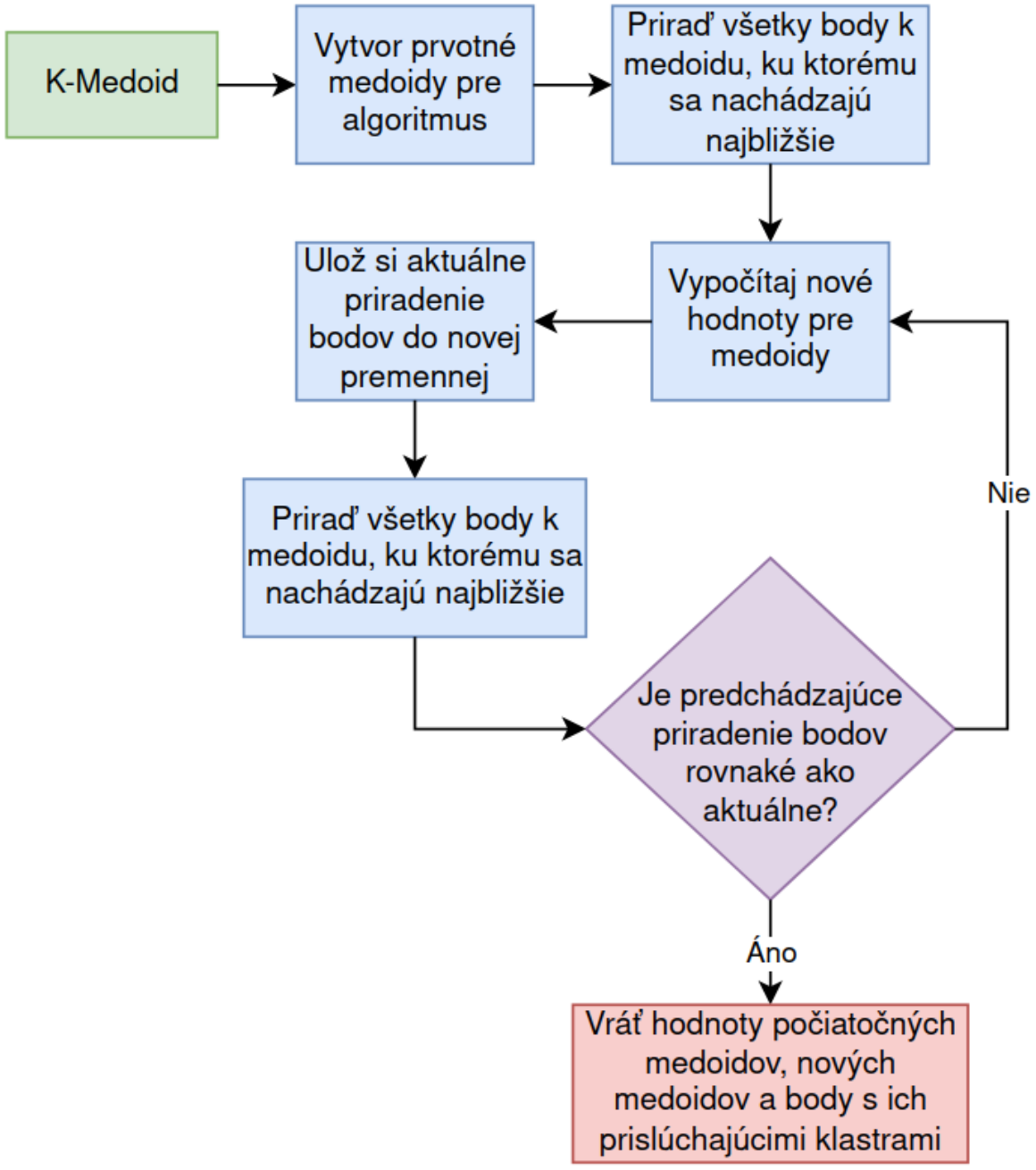
2.1 K-means

K-means je metóda vektorovej kvantizácie, ktorej cieľom je rozdeliť dáta do k zhlukov, v ktorých každé pozorovanie z dát patrí do zhluku s najbližším priemerom (centroid).
K-means si na začiatok zvolí **k náhodných centroidov**, ktorých súradnice sa budú počas behu algoritmu kontinuálne meniť. Následne sa všetky body priradia do jedného z týchto centrov podľa toho, ktorému centru sú najbližšie. Potom sa prechádza do cyklu, v ktorom sa vypočítavajú nové hodnoty centroidov tak, aby bol nový centroid ozajstným centroidom pre body, ktoré mu boli pridelené v kroku vyššie. To sa opakuje dovtedy, pokiaľ sa v dvoch cykloch nezmení ani jeden klaster.



2.2 K-medoid

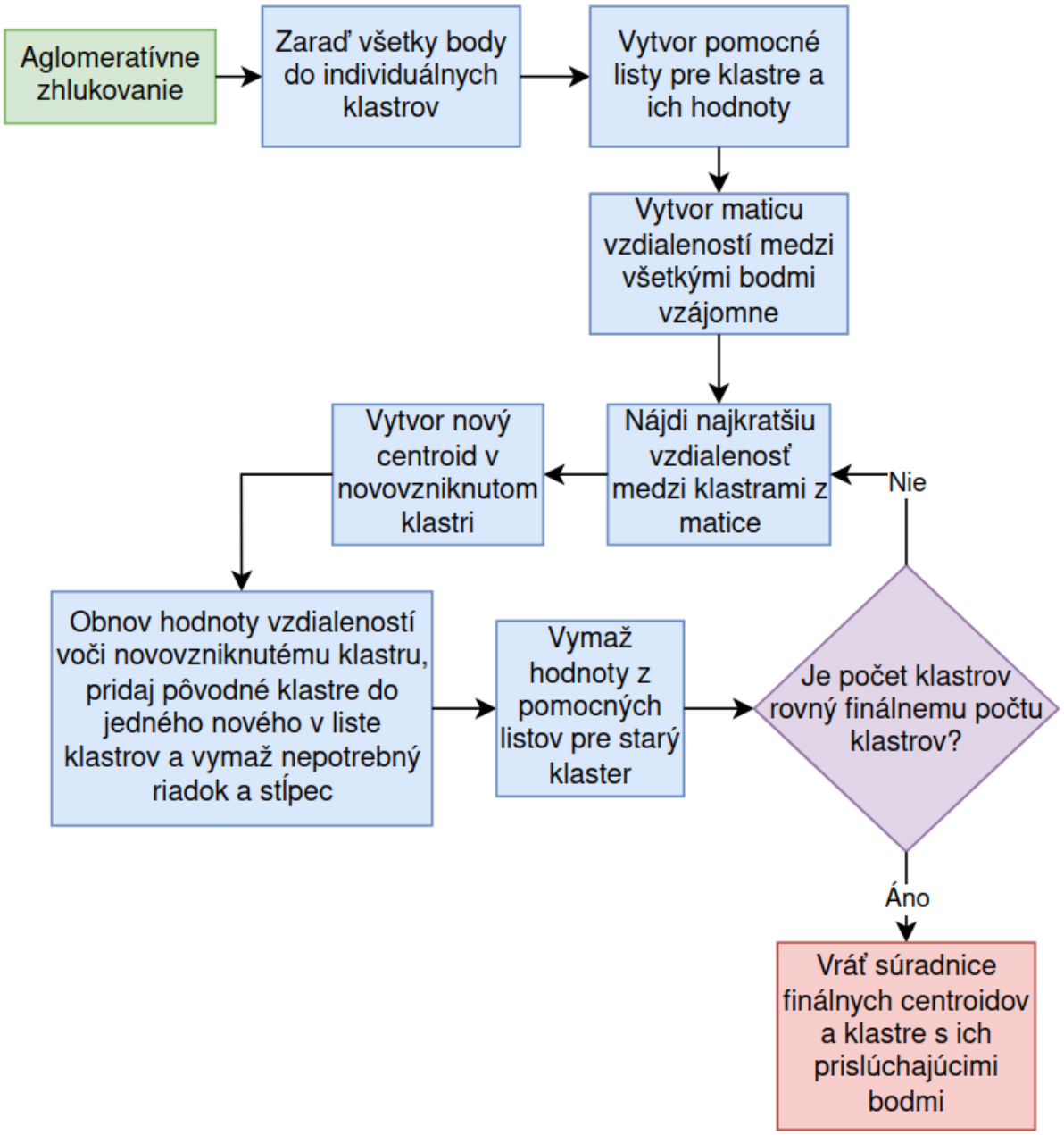
K-medoid je variácia vyššie spomenutého k-means algoritmu. Operuje rovnako s tým rozdielom, že sa body nepriradzujú centroidom, ale **medoidom**.
Medoid je definovaný ako objekt v danom zhluku, ktorého priemerná odlišnosť od všetkých objektov v zhluku je minimálna, to znamená, že je to **najviac centrálne umiestnený bod** v zhluku.



2.3 Aglomeratívne zhlukovanie

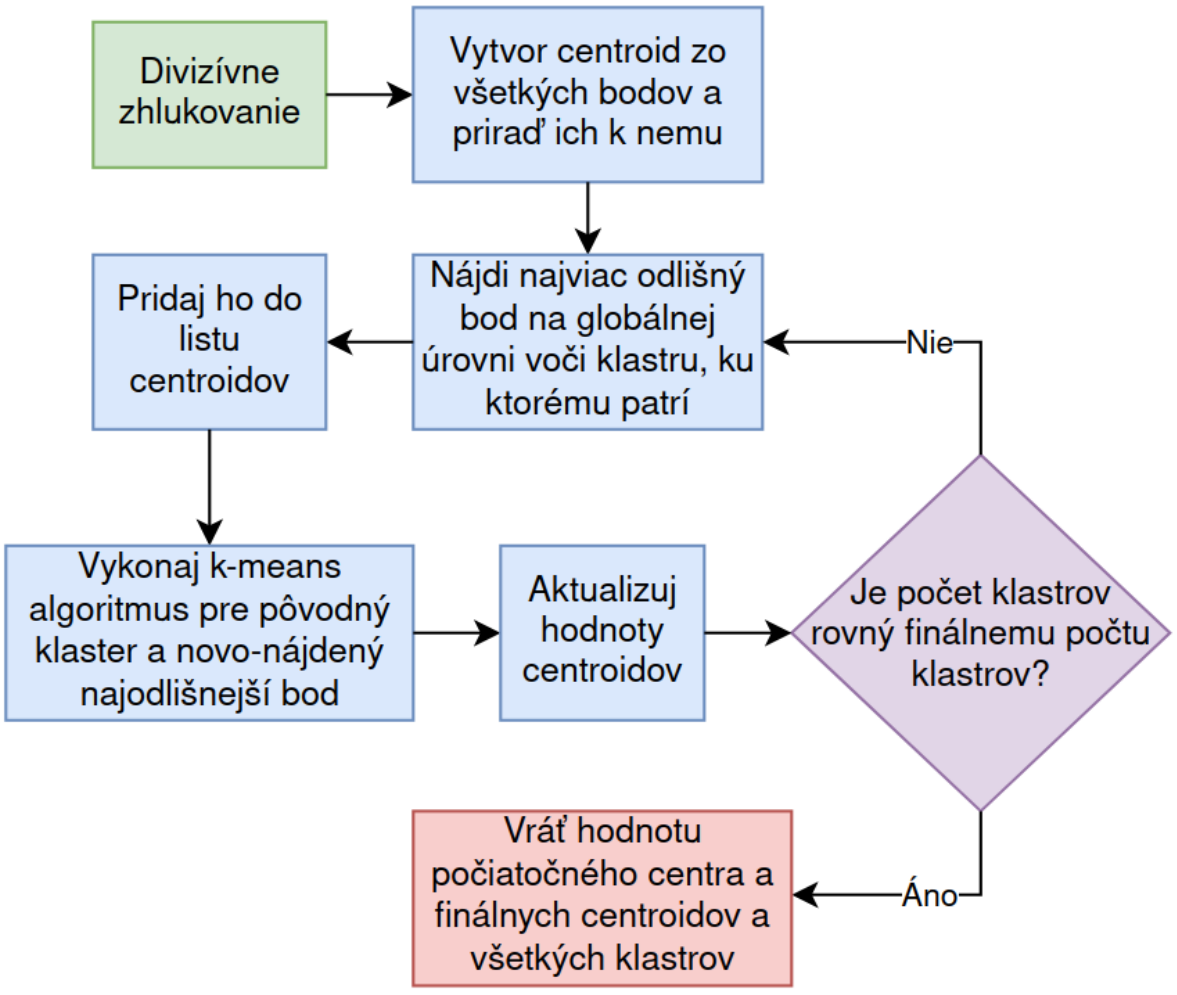
Aglomeratívne je tzv. “bottom-up” hierarchický algoritmus, ktorého počiatočná konfigurácia predstavuje stav, v ktorom **každý bod je samostatný klaster**. Tieto samostatné body sú zhlukované do klastrov alebo klastre do väčších klastrov dovtedy, pokiaľ nie je dosiahnutý želaný počet klastrov.
Na tento algoritmus som aplikoval niekoľko mnou navrhnutých optimalizácií. Keďže pre tento algoritmus je vhodné reprezentovať vzdialenosti medzi bodmi maticou

vzdialeností, najdôležitejšia z nich začína odtiaľ. Keďže je táto matica zrkadlová pozdĺž hlavnej diagonály, stačí prechádzať len jej polovicu napr. nad diagonálou, vďaka čomu je algoritmus 2x rýchlejší, čo sa mi potvrdilo aj na krátkych testoch.



2.4 Divizívne zhlukovanie

Divizívne zhlukovanie patrí taktiež do skupiny hierarchických. Ide o algoritmus, ktorý pracuje “top-bottom” resp. zhora nadol, čo znamená, že pracuje opačným smerom ako aglomeratívne zhlukovanie.
Tento algoritmus začína v stave, v ktorom **všetky body patria do jedného veľkého klastra**. Na začiatku každého cyklu sa určí bod, ktorý je najviac odlišný od ostatných v klastre, v ktorom existuje. Tento bod sa určí ako nový stred nového klastra s tým, že všetky zvyšné body sa priradia buď k tomuto novému stredu, alebo ostanú priradené k pôvodnému klastru podľa ich vzdialenosti k týmto centrom.



3. Základné funkcie a reprezentácia údajov

3.1 Základné funkcie

Medzi základné funkcie v programe patria placeholdery pre samotné algoritmy a pomocné výpočtové funkcie, ktoré sú zdieľané medzi nimi. Prvé menované:

`k_means(all_points, k_clusters, recalc_method)` -> funkcia, v ktorej sa vykonáva **k_means** a **k_medoid**. Rozdiel udáva argument recalc_method, ktorý určuje funkcionalitu, či ide o algoritmus k_means alebo k_medoid.

`agglomerative(all_points, k_clusters)` -> funkcia, ktorá vykonáva algoritmus **aglomeratívneho zhlukovania** na liste bodov v argumente all_points. Argument k_clusters udáva finálny počet klastrov, ktorý predstavuje hraničný bod pre vykonávanie algoritmu.

`divisive(all_points, k_clusters)` -> placeholder pre algoritmus **divizívneho zhlukovania**, ktorý má identické argumenty ako aglomeratívne zhlukovanie - all_points sú všetky dáta a k_clusters predstavuje želaný počet klastrov.

Dôležité pomocné funkcie:

`assign_to_clusters(centroids, points)` -> funkcia, ktorá **priradzuje body do klastrov**, ku ktorým sú najbližšie vzdialené. Väčšinou sa táto funkcia volá po vypočítaní nových hodnôt centroidov alebo medoidov.

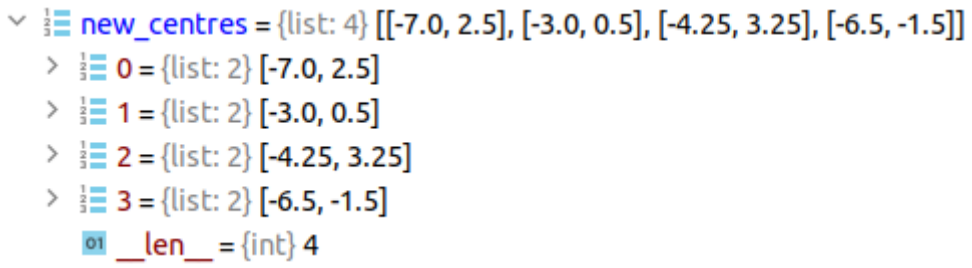
`calculate_centroids(points, clusters)` -> funkcia, ktorá počíta **nové hodnoty pre algoritmy,** ktoré využívajú **centroidy** ako ústredné hodnoty pre určovanie klastrov. Obdobne funguje aj funkcia `calculate_medoids(points, clusters)`.

`euclidian_distance(x, y)` -> jednoriadková funkcia na **výpočet vzdialenosti v priestore pre dvojicu bodov**, ktoré sú zadané v argumentoch.

3.2 Reprezentácia údajov

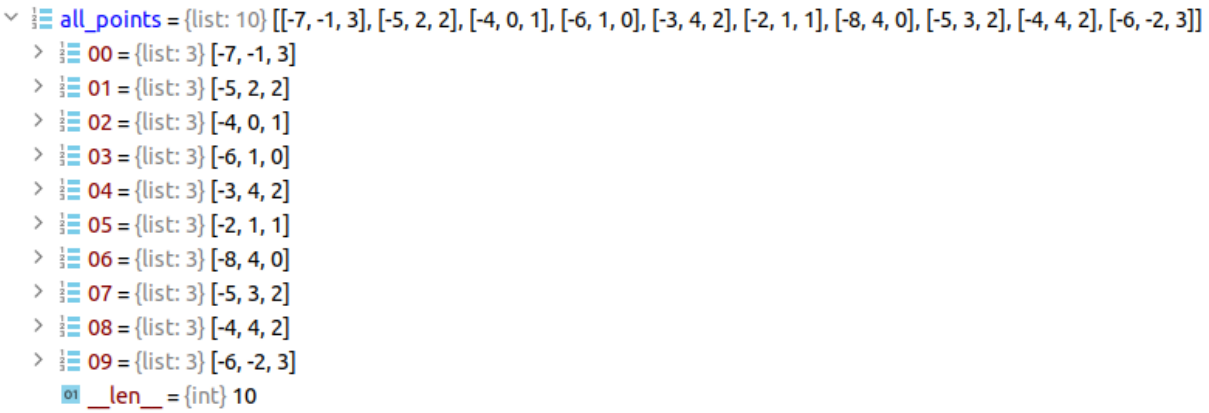
Táto implementácia ukladá informácie výhradne do dátového typu **list** v Pythone. Tie sa používajú na reprezentácie všetkých potrebných údajov ako napr. **súradnice bodov, výsledné súradnice aj s priradením do klastrov či súradnice samotných centroidov**. Vo väčšine prípadov ide o **dvojrozmerné listy**, ktoré sú vo svojej podstate listy listov, v ktorých sú uložené konkrétne súradnice.

Príklad reprezentácie centroidov:



Reprezentácia bodov ako dát sa od centroidov trochu odlišuje. Okrem x-ovej a y-ovej súradnice so sebou nesú aj **tretiu položku**, a to poradové číslo samotného klastra, do ktorého patria. Z toho vyplýva, že tieto čísla sú z intervalu od 0 po dĺžku listu centroidov/medoidov.

Príklad reprezentácie údajov:



V prípadoch delenia na subsety, ktoré predstavujú rôzne klastre, môže dočasne vzniknúť **3D list**, ktorý obsahuje listy listov. To je potrebné v prípade, ak **musia byť všetky body pohromade a zároveň rozdelené do klastrov v jednej štruktúre**.

4. Používateľské rozhranie

Používateľské rozhranie je jednoduché a samovysvetľujúce, orientované na zopár jednoduchých operácií v konzole. Skladá sa z **5 vstupných hodnôt**, ktoré definujú správanie sa programu. Výstup z programu je zobrazený knižnicami seaborn a matplotlib, ktorá vykreslí **farebné grafy podľa klastrov do 2x2 gridu**. Druhá časť výstupu je taktiež v konzole, kde sú vypísané všetky **priemerné vzdialenosti od stredov klastrov** pre všetky algoritmy. V testovaní tento výstup nebude zahrnutý, keďže by bol siahodlhý.
Konkrétne parametre pre tento program budú vysvetlené v ďalšej časti pri opise testovania.

Ukážka používateľského rozhrania na **vstupe**:

```
[INT] [INPUT] -- Number of generating points: 5
[INT] [INPUT] -- Number of all points: 750
[INT] [INPUT] -- Offset for generated points: 12
[INT] [INPUT] -- Dimension of space: 250
[INT] [INPUT] -- Number of clusters: 5
```

Ukážkový **výstup** z konzole hovoriaci o priemerných vzdialenostiach od centrov klastrov:

```
---------- K-Centroid ----------
Cluster 0 avg. distance: 14.061
Cluster 1 avg. distance: 68.044
Cluster 2 avg. distance: 10.033
Cluster 3 avg. distance: 14.284
Cluster 4 avg. distance: 14.556

---------- K-Medoid ----------
Cluster 0 avg. distance: 100.746
Cluster 1 avg. distance: 68.981
Cluster 2 avg. distance: 12.818
Cluster 3 avg. distance: 12.793
Cluster 4 avg. distance: 10.19

---------- Agglomerative Clustering ----------
Cluster 0 avg. distance: 18.062
Cluster 1 avg. distance: 11.49
Cluster 2 avg. distance: 10.033
Cluster 3 avg. distance: 10.328
Cluster 4 avg. distance: 14.061

---------- Divisive Clustering ----------
Cluster 0 avg. distance: 14.831
Cluster 1 avg. distance: 14.061
Cluster 2 avg. distance: 68.044
Cluster 3 avg. distance: 10.033
Cluster 4 avg. distance: 13.877
```

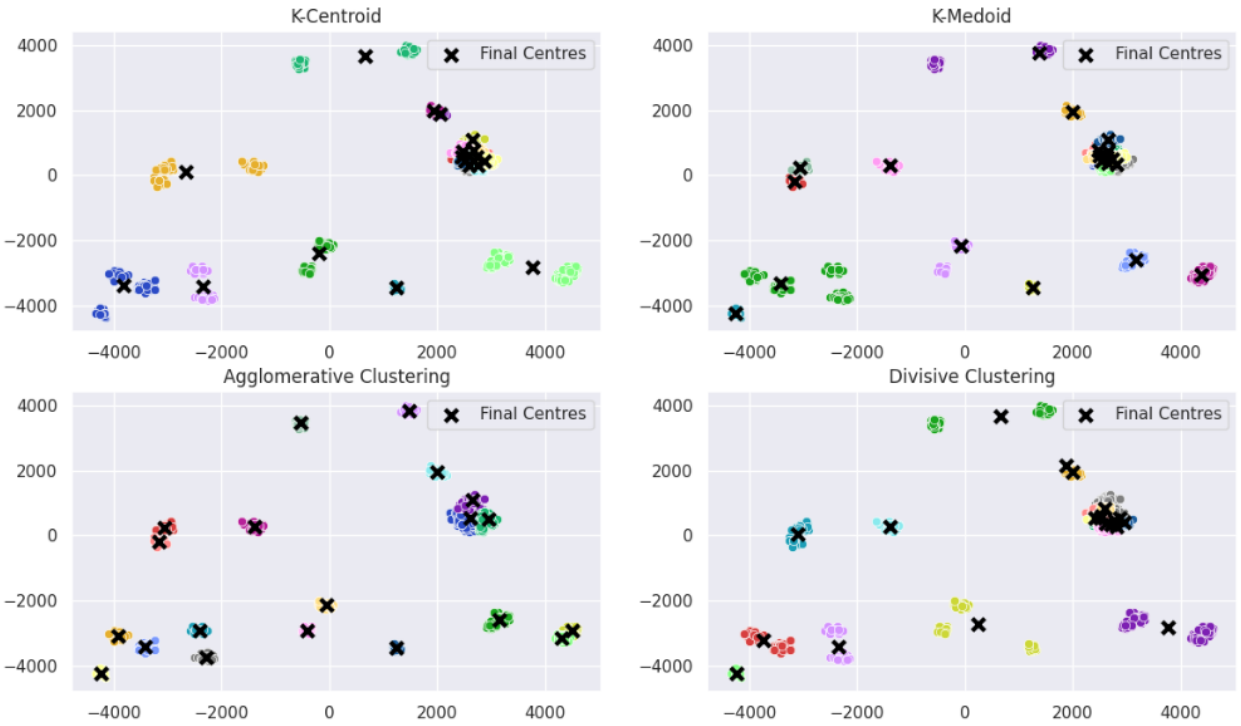
Príklad **výstupného grafu** z knižnice matplotlib a seaborn:

5. Testovanie

Testovanie prebehlo v 4 fázach, resp. pre 4 rôzne počty vstupných bodov. Pre všetky algoritmy bol použitý **rovnaký zoznam bodov**, aby mohli pracovať v rovnakých podmienkach a ich výstup bol porovnateľný a zmyselný.
Prvé dva testy, konkrétne pre počty bodov 2500 a 10000, sú zahrnuté len z informatívneho hľadiska. Testovanie na **15000** bodoch bolo vykonané z dôvodu, lebo môj počítač by **nezvládol rozbehnúť aglomeratívne zhlukovanie na 20000 bodoch**, resp. by to trvalo približne 2 dni. Preto bol vykonaný test na 15000 bodoch na demonštráciu funkčnosti aglomeratívneho zhlukovania a porovnania medzi všetkými algoritmami a zároveň bolo vykonané testovanie na 20000 bodoch pre zvyšné 3 algoritmy pre maximálny počet bodov.
Okrem počtu bodov sa vstupné parametre nijako nemenili, t.j. počet počiatočných generovacích bodov bol rovný 20, rozmery priestoru boli rovné -5000 až 5000, počet klastrov 20 a offset pre novo-vygenerované body bol stanovený na hodnotu 100.

Meno: Marko Stahovec
ID: 110897

5.1 N = 2500 bodov



K-Centroid:

- **úspešnosť: 17/20** (85%)
- najväčší priemer v klastri: 982.8
- indexy klastrov nad hraničnou hodnotou: [4, 8, **12**]

K-Medoid:

- **úspešnosť: 18/20** (90%)
- najväčší priemer v klastri: 852.399
- indexy klastrov nad hraničnou hodnotou: [3, **4**]

Aglomeratívne zhlukovanie:

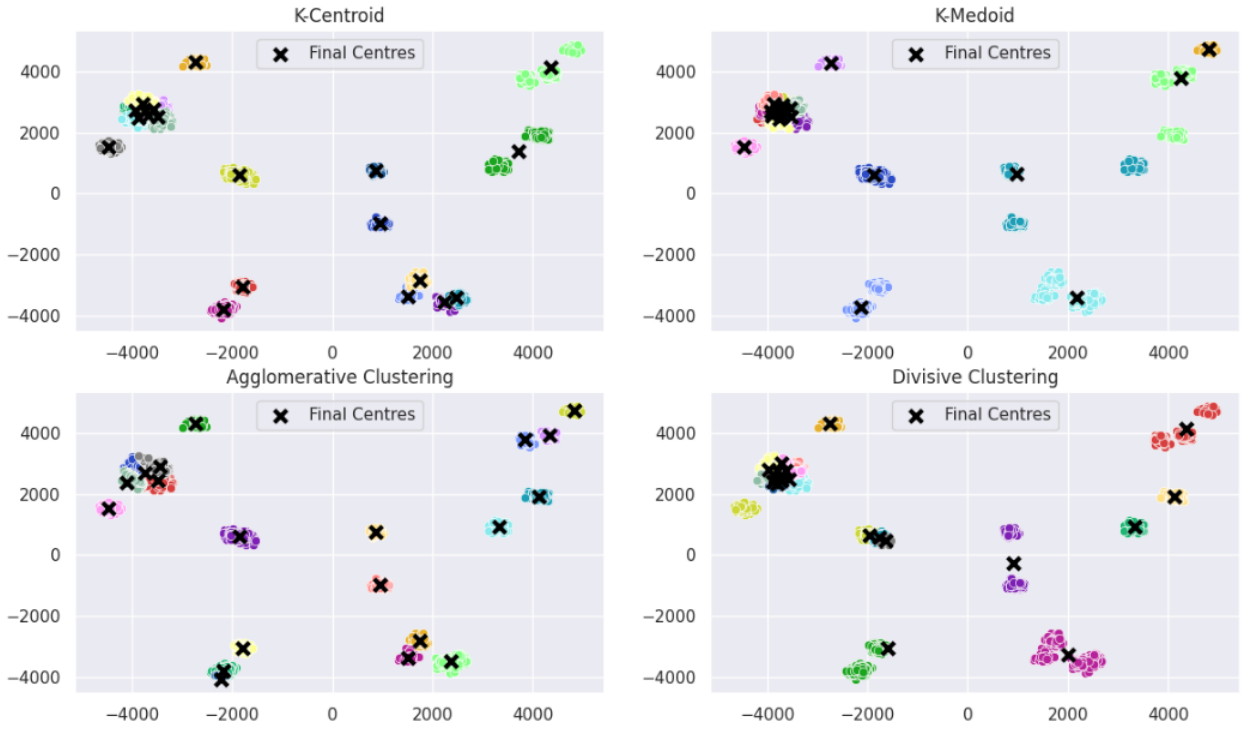
- **úspešnosť: 20/20** (100%)
- najväčší priemer v klastri: 188.462
- indexy klastrov nad hraničnou hodnotou: []

Divizívne zhlukovanie:

- **úspešnosť: 18/20** (90%)
- najväčší priemer v klastri: 857.032
- indexy klastrov nad hraničnou hodnotou: [3, **5**]

Meno: Marko Stahovec
ID: 110897

5.2 N = 10000 bodov



K-Centroid:

- **úspešnosť: 19/20** (95%)
- najväčší priemer v klastri: 637.537
- indexy klastrov nad hraničnou hodnotou: [**2**]

K-Medoid:

- **úspešnosť: 18/20** (90%)
- najväčší priemer v klastri: 1524.029
- indexy klastrov nad hraničnou hodnotou: [**7**, 12]

Aglomeratívne zhlukovanie:

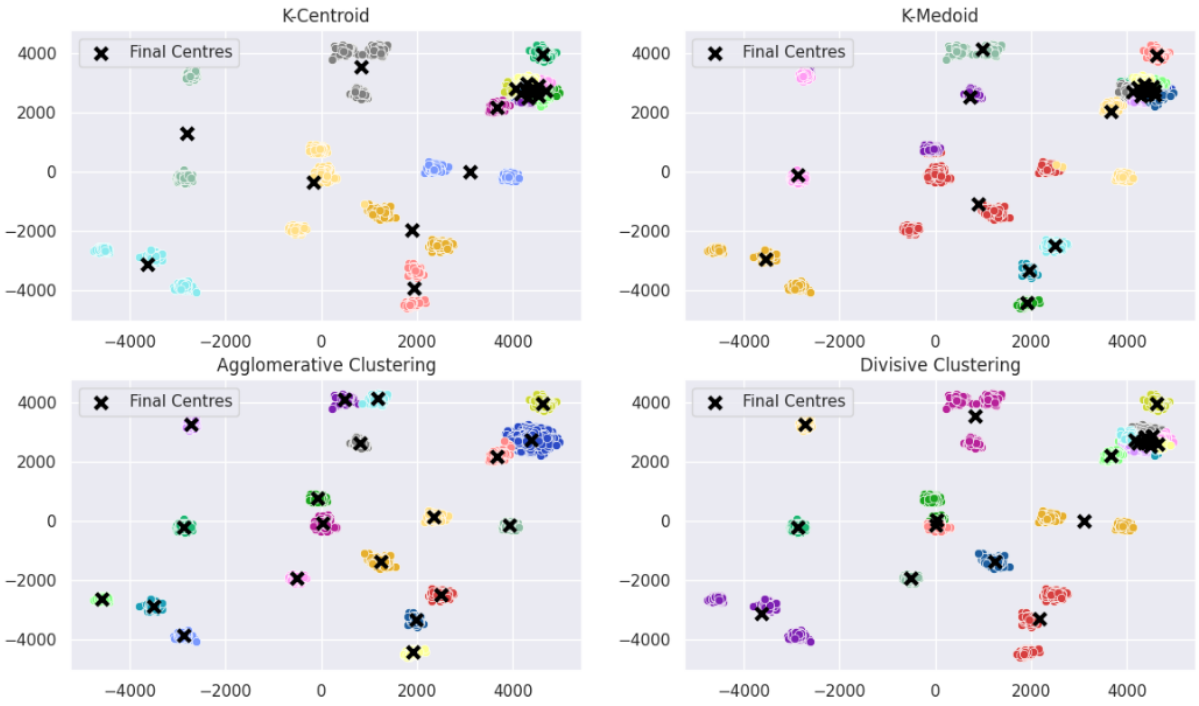
- **úspešnosť: 20/20** (100%)
- najväčší priemer v klastri: 219.545
- indexy klastrov nad hraničnou hodnotou: []

Divizívne zhlukovanie:

- **úspešnosť: 17/20** (85%)
- najväčší priemer v klastri: 1326.579
- indexy klastrov nad hraničnou hodnotou: [2, 3, **5**]

Meno: Marko Stahovec
ID: 110897

5.3 N = 15000 bodov



K-Centroid:

- **úspešnosť: 15/20** (75%)
- najväčší priemer v klastri: 1712.177
- indexy klastrov nad hraničnou hodnotou: [9, 10, 13, 15, **18**]

K-Medoid:

- **úspešnosť: 16/20** (80%)
- najväčší priemer v klastri: 1446.286
- indexy klastrov nad hraničnou hodnotou: [1, 3, 13, **16**]

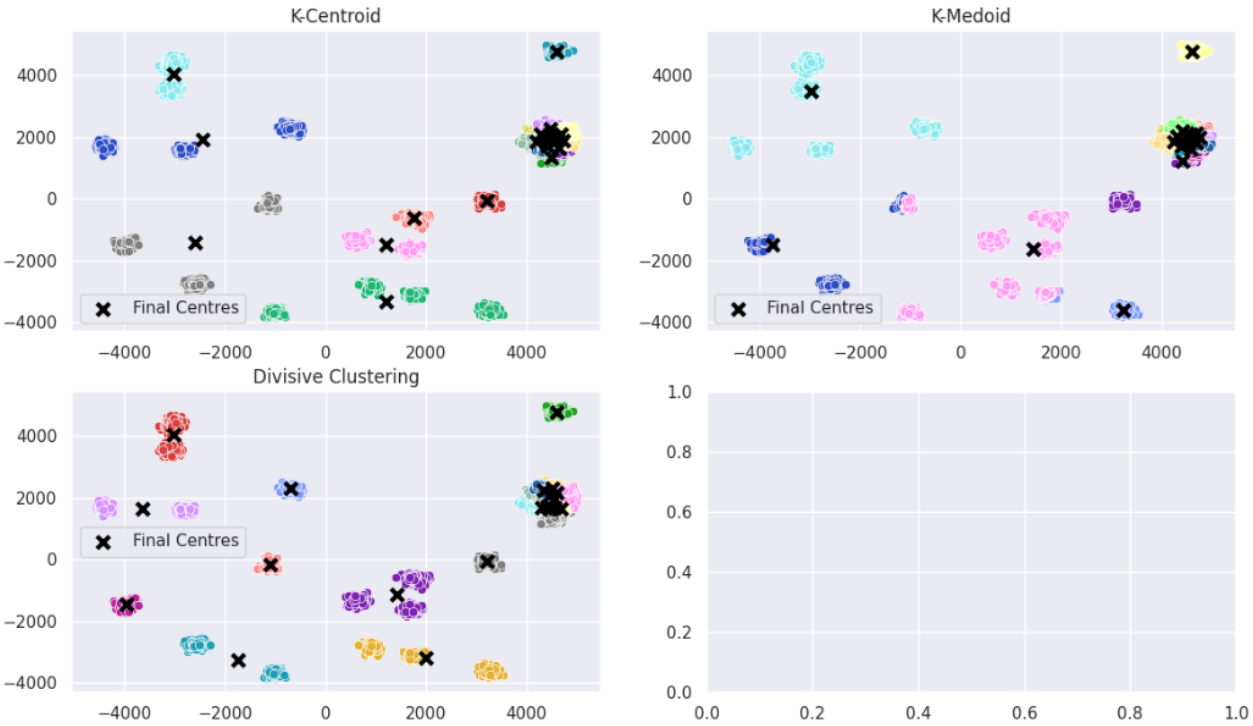
Aglomeratívne zhlukovanie:

- **úspešnosť: 20/20** (100%)
- najväčší priemer v klastri: 180.618
- indexy klastrov nad hraničnou hodnotou: []

Divizívne zhlukovanie:

- **úspešnosť: 16/20** (80%)
- najväčší priemer v klastri: 797.112
- indexy klastrov nad hraničnou hodnotou: [1, 3, **4**, 6]

5.4 N = 20000 bodov



K-Centroid:

- **úspešnosť: 17/20** (85%)
- najväčší priemer v klastri: 1545.619
- indexy klastrov nad hraničnou hodnotou: [0, 8, **9**]

K-Medoid:

- **úspešnosť: 16/20** (80%)
- najväčší priemer v klastri: 1511.418
- indexy klastrov nad hraničnou hodnotou: [0, 3, **15**, 16]

Divizívne zhlukovanie:

- **úspešnosť: 16/20** (80%)
- najväčší priemer v klastri: 1468.483
- indexy klastrov nad hraničnou hodnotou: [4, 7, **9**, 14]

6. Zhodnotenie a záver

Toto zadanie bolo zamerané na porovnanie rôznych klastrovacích algoritmov v rôznych podmienkach. Najúspešnejším algoritmom bolo **aglomeratívne zhlukovanie**, ktoré aj napriek signifikantne najhoršej časovej zložitosti dosahuje najlepšie výsledky. Aglomeratívne zhlukovanie by podľa všetkého bolo úspešné aj na 20000 bodoch.
Algoritmy K-Centroid a K-Medoid fungovali relatívne korektne, no ich veľkým problémom bol fakt, že **počiatočné centrá boli vyberané náhodne**. To by sa dalo vylepšiť algoritmom **K-means++**, no ten podľa zadania nebolo potrebné implementovať. Podľa testovania nie je možné zaručene potvrdiť, že počet bodov neovplyvňuje úspešnosť algoritmov.
Divizívne zhlukovanie dosahovalo obdobné výsledky ako K-. algoritmy s **nebadateľne lepšími výsledkami pri priemeroch nevyhovujúcich klastrov**, t.j. ak aj mal klaster priemernú vzdialenosť od stredu väčšiu ako 500, tak táto vzdialenosť bola zvýšená len minimálne. Myslím si, že jediným problémom pri divizívnom zhlukovaní je fakt, že má tendenciu **rozpoľovať väčšie zhluky** bodov a potreboval by ešte ďalšie heuristické pravidlo na lepšie výsledky.
Platí teda, že aj v tejto oblasti ide o tzv. trade-off medzi výpočtovou náročnosťou a úspešnosťou, a preto je ťažké určiť najlepší algoritmus. Každopádne je táto implementácia korektná, primerane optimalizovaná a správa sa tak, ako by sa od nej dalo očakávať.