FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
SLOVENSKÁ TECHNICKÁ UNIVERZITA
Ilkovičova 2, 842 16 Bratislava 4

2021/2022

Umelá inteligencia

Zadanie č.4 – Klastrovanie

Cvičiaci: Ing. Ivan Kapustík
Čas cvičení: Štvrtok 12:00 – 13:50

Vypracovala: Monika Zjavková
AIS ID: 105345

Obsah

1. Zadanie – 4b

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.

2. Opis riešenia

Program začína vygenerovaním náhodných zhlukov 20 000 bodov, ktoré sú uložené v zozname. Každý bod je reprezentovaný ako dvojzložkový zoznam v tvare [x, y]. Klastre, kde sú následne organizované body sú ukladané do slovníka, kde kľúč je stredom daného klastra a hodnoty sú všetky body, ktoré sú k nemu priradené.

Body sú po priradení vykresľované pomocou importovanej knižnice matplotlib.pyplot, pri čom každý klaster je vyfarbený inou farbou.

2.1. K-means

Riešenie bolo realizované aj pre medoid a centroid jednou funkciu, ktorý sa líši len v hľadaní nových stredov. Na začiatku je určená K hodnota. Je vygenerovaná náhodne od 5-13, toto číslo predstavuje počet stredov a klastrov, ktoré v programe vzniknú.

K- počet stredov je na začiatku vybraných náhode zo všetkých bodov a statné body sú k nim priraďované podľa toho, ku ktorému z týchto stredov sú najbližšie. Pre každý bod sa teda prejdú všetky stredy a vypočíta sa vzdialenosť.

Pre rozdelené body v klastroch sú vypočítané nové stredy. Pre centroid to znamená, že keďže stredom nemusí byť jeden z bodov, vypočíta sa iba priemerná x a y súradnica, tie sú potom uložené ako nový stred.

Ak je stredom medoid, vo funkcii určenej na vypočítanie stredu sa prejdú všetky body v danom klastri a pre každý sa vypočíta priemerná vzdialenosť od ostatných. Bod s najmenšou priemernou vzdialenosťou je zvolený ako nový stred.

Keď sú vypočítané nové stredy, cyklus sa opakuje znovu, takže ku každému novému stredu sú odznova priraďované body podľa ich vzdialenosti ako v prvom kroku po vygenerovaní náhodných stredov.

Cyklus skončí vtedy, keď novo vypočítané klastre sú rovnaké ako predchádzajúce.

2.2. Aglomeratívne zhlukovanie

Aglomeratívne zhlukovanie pracuje na základe „bottom-up“ stratégii. Na začiatku každý bod predstavuje jeden klaster a je vypočítaná matica vzdialeností medzi týmito klastrami. Následne sa vyberie z matice minimum, ktoré predstavuje najmenšiu vzdialenosť v matici a indexy stĺpca a riadku predstavujú klastre.

Vybrané body sú vymazané, rovnako aj aj riadky a stĺpce na vybraných indexoch, čím sa veľkosť matice zmenšuje. Pridá sa však riadok a stĺpec s novými vzdialenosťami. Táto hodnota predstavuje vzdialenosť nového stredu, vypočítaného z pôvodných bodov, a všetkých zvyšných bodov, ktoré ostali.

Podľa toho, či už jeden z bodov bol priradený ku inému klastru sa vytvorí nový. Ak ešte ani jeden nebol v klastri, vytvorí sa nový, kde sa v ňom budú nachádzať iba tieto dva body. V prípade, že už z jedného z bodov bol vytvorený klaster, ten druhý je do neho pridaný a je aktualizovaný stred. Ak sú z oboch už vytvorené klastre, znamená to, že treba ich spojiť, takže sa aktualizuje stred a body k nemu priradené predstavujú spojené zoznamy z pôvodných klastrov.

Cyklus pokračuje pri čom, veľkosť matice sa zmenšuje zakaždým o 1 a pokračuje dovtedy, kým minimálna vzdialenosť je menšia ako 500.

2.3. Divizívne zhlukovanie

Program funguje na opačnom princípe ako aglomeratívne zhlukovanie, čiže všetky body sú na začiatku pridelené k jednému stredu, ktorým je centroid, teda je vypočítaný ako priemerná súradnica x a y.

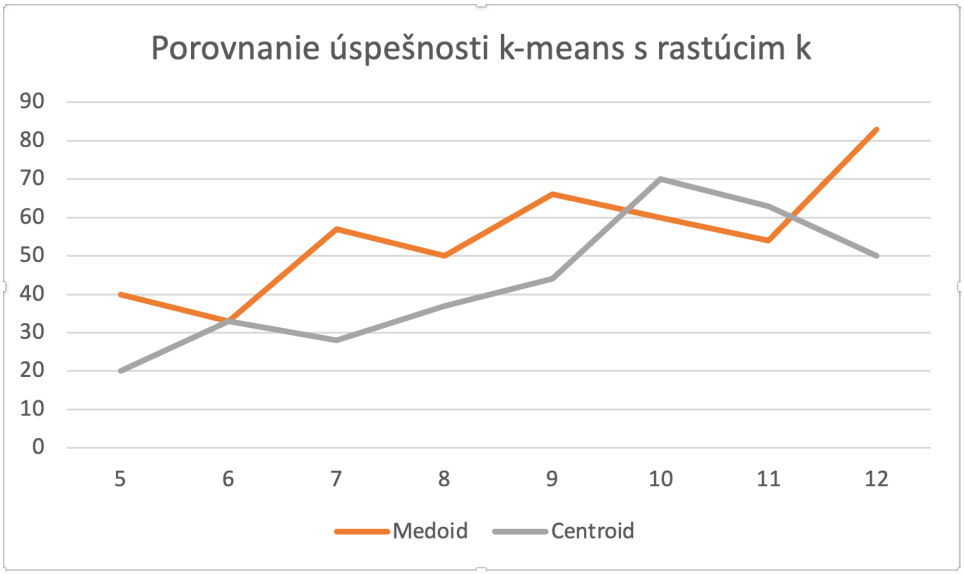
Najväčší klaster sa vždy rozkladá na 2 a to na rovnakom princípe ako k-means algoritmus, kde k je 2, čiže sa vyberú náhodne 2 body, ktoré predstavujú stredy nových klastrov. Potom sa k ním priraďujú body a upravujú stredy, kým nie sú pôvodné stredy rovnaké a tým pádom aj klastre rovnaké ako predchádzajúce.

Nové dva klastre sú potom vložené na miesto pôvodného veľkého. Ďalší klaster na rozloženie je vybratý pomocou vypočítania priemernej vzdialenosti bodov v klastri od ich stredu. Klaster s najväčšou priemernou vzdialenosťou je zvolený na rozdelenie.

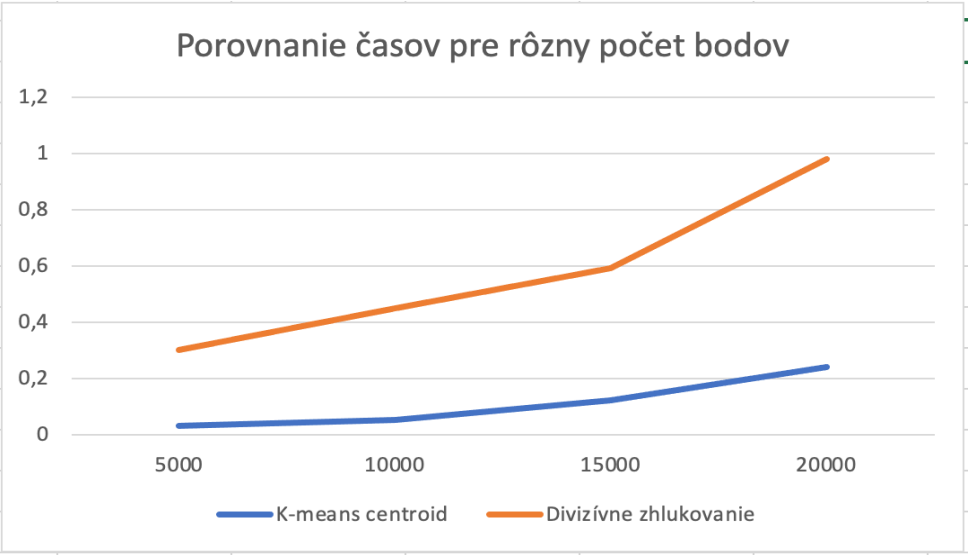
Rozdeľovanie končí, keď priemerná vzdialenosť od stredu už nie je viac menšia ako 500.

3. Zhodnotenie riešenia a testovanie

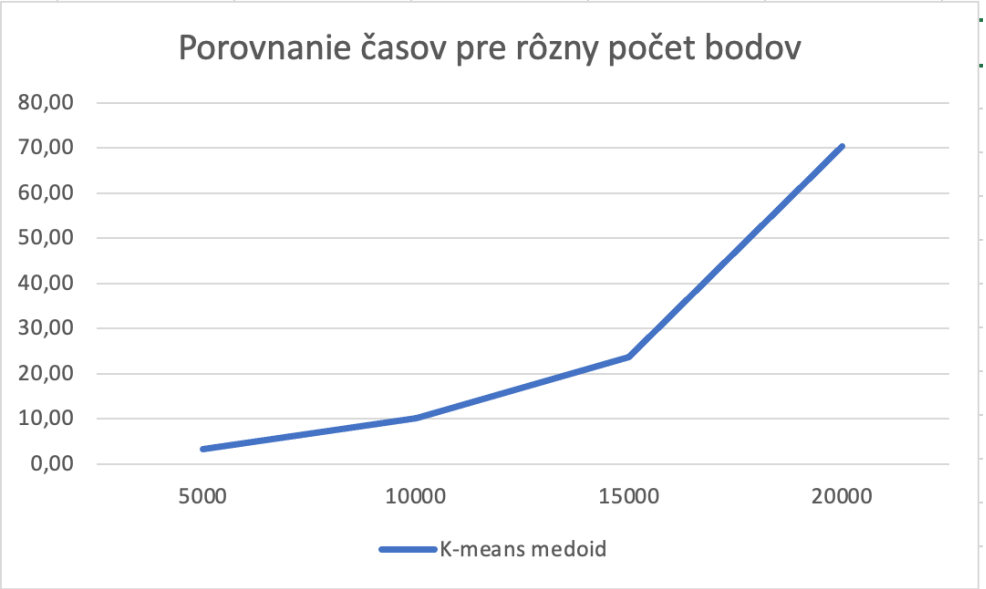
Najúspešnejšie algoritmy pre roztriedenie bodov do klastrov sú aglomeratívne a divizívne zhlukovanie, keďže sú naprogramované, aby skončili pri priemernej vzdialenosti 500. K-means bolo úspešné približne v 50% klastrov. Hodnota môže byť rozdielna aj pre rovnaký počet klastrov a počet bodov, keďže stredy sú vyberané náhodne a kvôli tomu môžu vznikať nerovnomerne veľké klastre. Riešenie so stredom ako medoid bolo však úspešnejšie ako centroid, keďže stredom je skutočný bod.



Najrýchlejší bol algoritmus k-means, kde je stred centroid, následne divizívne, keďže v oboch pri počítaní stredu stačí prejsť zoznam bodov iba raz a riešenie našlo za niekoľko sekúnd pri 20000 bodov. K-means s medoidom trvalo dlhšie, lebo bolo potrebné použiť vnorený cyklus na nájdenie stredu.

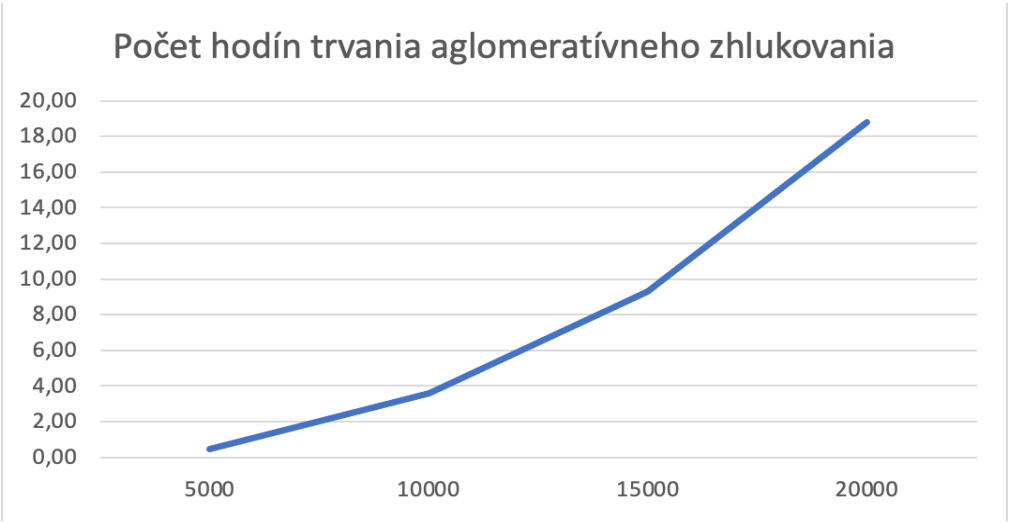


V grafe pre čas výpočtu k-means s medoidom je vidieť, že prechádzanie bodov v 2 cykloch program spomalilo a pri 20000 bodoch, trvalo riešenie približne 1,1 minúty



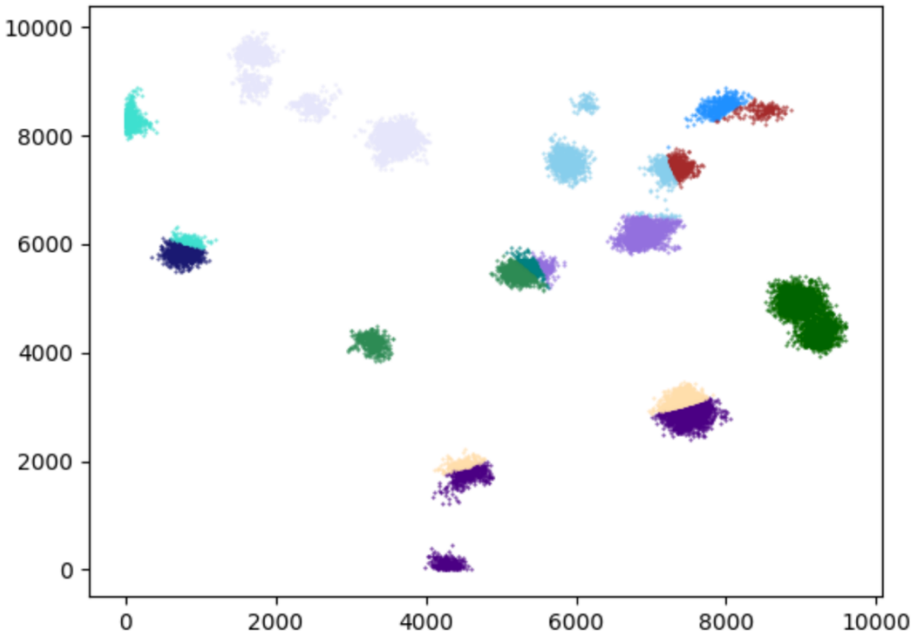
Najdlhšie trvalo aglomeratívne, kde sa pracuje s maticou počet bodov^2 pre určovanie najmenšej vzdialenosti. Najviac času tam teda zabralo naplnenie matice. Na optimalizáciu bola použitá importovaná knižnica numpy, kde sa vďaka numpy arrayom podarilo znížiť čas behu programu viac ako o polovicu. Hlavne kvôli tomu, že si array udržiava informáciu o minime a nebolo teda potrebné zakaždým prechádzať celú maticu. Rovnako aj vkladanie a vymazávanie je rýchlejšie a efektívnejšie kvôli špecifickým funkciam na prácu s maticou. Problém predstavovala aj prostredie Pycharm, pretože nedokázalo väčšinou pracovať viacej ako s 5% CPU.

Riešenie teda pre 20 000 bodov trvalo viac ako 19 hodín.

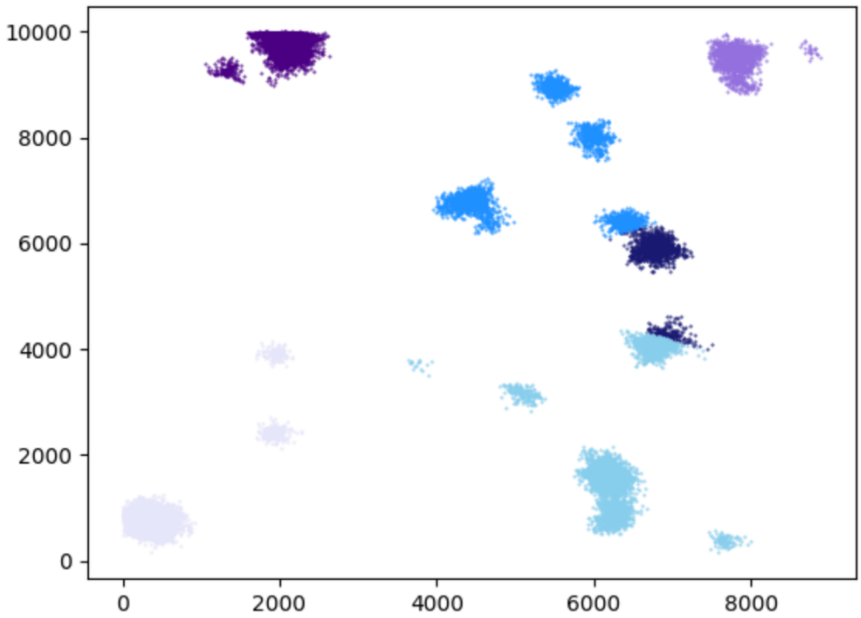


4. Výsledky

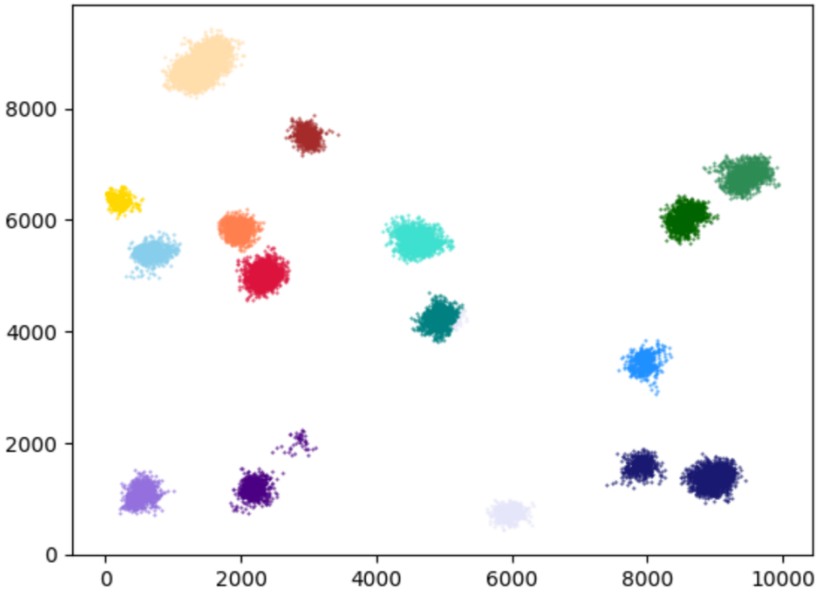
4.1. K-means – medoid



4.2. K-means – centroid



4.3. Aglomeratívne zhlukovanie



4.4. Divizívne zhlukovanie