7/12/2021

Klasifikácia a zhlukovanie

Umelá inteligencia – zadanie 4

Samuel Hetteš, ID: 110968
STU FIIT 2021/2022

OBSAH

Strojové učenia sa
Samuel Hetteš, ID: 110968

ZADANIE 4A – KLASIFIKÁCIA

Máme 2D priestor, ktorý má rozmery X a Y, v intervaloch od -5000 do +5000. V tomto priestore sa môžu nachádzať body, pričom každý bod má určenú polohu pomocou súradníc X a Y. Každý bod má unikátne súradnice (t.j. nemalo by byť viacej bodov na presne tom istom mieste). Každý bod patrí do jednej zo 4 tried, pričom tieto triedy sú: red (R), green (G), blue (B) a purple (P). Na začiatku sa v priestore nachádza 5 bodov pre každú triedu (dokopy teda 20 bodov). Súradnice počiatočných bodov sú:

- R: [-4500, -4400], [-4100, -3000], [-1800, -2400], [-2500, -3400] a [-2000, -1400]
- G: [+4500, -4400], [+4100, -3000], [+1800, -2400], [+2500, -3400] a [+2000, -1400]
- B: [-4500, +4400], [-4100, +3000], [-1800, +2400], [-2500, +3400] a [-2000, +1400]
- P: [+4500, +4400], [+4100, +3000], [+1800, +2400], [+2500, +3400] a [+2000, +1400]

Vašou úlohou je naprogramovať klasifikátor pre nové body – v podobe funkcie classify(int X, int Y, int k), ktorá klasifikuje nový bod so súradnicami X a Y, pridá tento bod do nášho 2D priestoru a vráti triedu, ktorú pridelila pre tento bod. Na klasifikáciu použite k-NN algoritmus, pričom k môže byť 1, 3, 7 alebo 15.

Na demonštráciu Vášho klasifikátora vytvorte testovacie prostredie, v rámci ktorého budete postupne generovať nové body a klasifikovať ich (volaním funkcie classify). Celkovo vygenerujte 20000 nových bodov (5000 z každej triedy). Súradnice nových bodov generujte náhodne, pričom nový bod by mal mať zakaždým inú triedu (dva body vygenerované po sebe by nemali byť rovnakej triedy):

- R body by mali byť generované s 99% pravdepodobnosťou s X < +500 a Y < +500
- G body by mali byť generované s 99% pravdepodobnosťou s X > -500 a Y < +500
- B body by mali byť generované s 99% pravdepodobnosťou s X < +500 a Y > -500
- P body by mali byť generované s 99% pravdepodobnosťou s X > -500 a Y > -500

  (Zvyšné jedno percento bodov je generované v celom priestore.)

Návratovú hodnotu funkcie classify porovnávajte s triedou vygenerovaného bodu. **Na základe týchto porovnaní vyhodnoťte úspešnosť** Vášho klasifikátora pre daný experiment.

Experiment vykonajte 4-krát, pričom zakaždým Váš klasifikátor použije iný parameter k (pre k = 1, 3, 7 alebo 15) a vygenerované body budú pre každý experiment rovnaké.

Vizualizácia: pre každý z týchto experimentov vykreslite výslednú 2D plochu tak, že vyfarbíte túto plochu celú. Prázdne miesta v 2D ploche vyfarbite podľa Vášho klasifikátora.

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

Poznámka 1: Je vhodné využiť nejaké optimalizácie na zredukovanie zložitosti, napríklad pre hľadanie k najbližších bodov si rozdelíme plochu na viaceré menšie štvorce, do ktorých umiestňujeme body s príslušnými súradnicami, aby sme nemuseli vždy porovnávať všetky body, ale len body vo štvorci, kde sa nachádza aktuálny bod a susedných štvorcoch.
Je tiež možné využiť algoritmus na hľadanie k najmenších hodnôt.

Poznámka 2: Úlohu je možné riešiť aj pomocou neurónovej siete, pričom je vhodné použiť nejaký framework (napríklad PyTorch).

OPIS RIEŠENIA

Implementačné prostredie

Daný problém som sa rozhodol riešiť v programovacom jazyku Python, konkrétne využívam verziu Python 3.9. Prostredie v ktorom som programoval je PyCharm Community Edition 2021.2.2.

Spôsob programovania

Program je naprogramovaný čisto procedulárnym programovaním. Objektovo-orientované aspekty sa nevyužívajú.

Zoznam knižníc

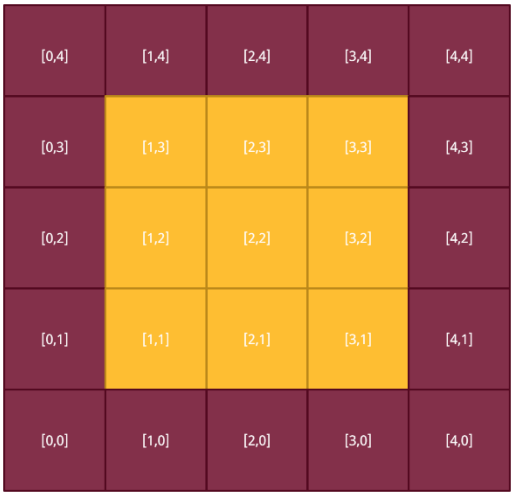
- *math* – počítanie euklidovej vzdialenosti
- *random* – generovanie náhodných bodov
- *matplotlib* – plotovanie grafov
- *numpy* – hľadanie k najmenších vzdialeností
- *timeit* – timer pre meranie celkového času vykonávania
- *collections* – Counter pre hľadanie najčastejšie sa vyskytujúcej triedy

Stavba programu a zoznam funkcií

- OPTIMALIZÁCIA

Predtým, než vysvetlím samotné riešenie, opíšem akým spôsobom som optimalizoval vyhľadávanie. Keďže hranice našej mapy sú -5000 až 5000, tak z toho vyplýva, že mapa je rozmerov 10000x10000. Preto som rozložil túto mapu na štvorce o veľkosti 100x100, teda dokopy 10000 štvorcov. Vytvoril som 4 polia štvorcov pre každý z klasifikátorov o veľkosti 100x100 a v každom konkrétnom štvorci sa potom ďalej nachádza list bodov, ktoré doň patria. Pri hľadaní najbližších bodov sa potom neprehľadáva celá mapa, ale len konkrétny štvorec a štvorce v prvej vrstve okolo neho. Ak by ani to nestačilo tak sa prehľadajú ďalšie vrstvy a tak ďalej. V podstate sa akoby prehľadávajú vrstvy terča, pričom začíname v strede. K štvorcu v ktorom sa nachádza daný bod sa pridáva aj vrstva okolo neho z toho dôvodu, že ak by daný bod bol na hranici štvorca, tak susedný štvorec by mohol mať niektoré body bližšie. Spočiatku však štvorce neobsahujú dostatočný počet hodnôt, aby bola táto optimalizácia výhodná, preto pre prvých 1000 bodov sa prehľadáva celý priestor. Až po vygenerovaní 1000 bodov sa začnú prehľadávať štvorce.

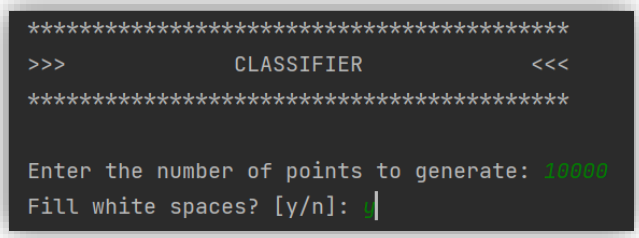
Postup prehľadávania štvorcov:



Ďalej opíšem funkcie a ich význam.

- FUNKCIA MAIN()

Main je prvá funkcia, ktorá sa spustí pri zapnutí programu. Na začiatku si program od používateľa vypýta vstupné informácie: počet generovaných bodov a či sa majú vypĺňať biele miesta v mape (doklasifikácia bodov, ktoré neboli vygenerované).



Potom sa vytvorí pole *sample (*počiatočná vzorka bodov), polia štvorcov pre každý z klasifikátorov a polia pre prvých 1000 bodov. Ďalej sa zabezpečí pridanie všetkých počiatočných bodov do štvorcov. Potom sa začne vykonávať cyklus generovania bodov. Body sa generujú s triedami v poradí Red, Blue, Green a Purple. Generovanie je rovnaké ako opisuje zadanie. Po vygenerovaní bodu sa zavolá funkcia *classify* pre každú hodnotu *k*, teda pre hodnoty 1, 3, 7 a 15. Funkcia si pre každú hodnotu uchová či bol bod klasifikovaný správne. Na záver tento vygenerovaný bod pridá do pola *sample* s triedou, ktorá ho generovala a rovnako aj do polí štvorcov a polí pre 1000 bodov, ak sme ešte nepresiahli túto hranicu, teraz sa však body pridávajú s triedami, ktoré vrátili klasifikátory. Pokiaľ sa takýto bod už vo vzorke nachádza, nepridá sa nič. Po skončení tohto cyklu vypíše počet správne a nesprávne klasifikovaných bodov, úspešnosť klasifikácie a celkový čas vykonávania. Rovnako ešte zavolá funkciu pre vyplotovanie diagramov.

- FUNKCIA EUCLID_DIST(A, B)

Táto funkcia len zabezpečí vypočítanie a vrátenie euklidovej vzdialenosti medzi bodmi A a B.

- FUNKCIA ADD_TO_SQUARES(POINT, SQUARES)

Táto funkcia pridáva daný bod do pola štvorcov. Zabezpečí vypočítanie indexov štvorca, do ktorého daný bod spadá a priradí ho.

- FUNKCIA CLASSIFY(X, Y, K, POINTS, SQUARES_SET)

Táto funkcia zabezpečí klasifikáciu daného bodu do triedy. Parametre x a y sú súradnice bodu, k je počet susedov, points je pole všetkých bodov a squares_set je bool hodnota, ktorá ak je nastavená, tak značí, že pole points je rozdelené do štvorcov. Na počiatku sa overí či je nastavený parameter *squares_set*, ak áno funkcia vypočíta indexy štvorca, do ktorého daný bod spadá, získa tento štvorec a vrstvu okolo neho, a prehľadá všetky body, ktoré sa v nich nachádzajú. V prípade, že počet nájdených bodov je menší ako hodnota *k*, tak prehľadáva ďalšie vrstvy až pokým nenájde dostatočný počet bodov. Ak parameter nie je nastavený, tak prehľadáva celý priestor. Pokiaľ je hodnota $k = 1$ a našiel sa iba jeden bod, tak vráti triedu tohto bodu. V opačnom prípade funkcia vyberie *k* najmenších hodnôt. Ďalej spočíta výskyt

tried týchto bodov. V prípade, že sa nájde majorita tak vráti túto triedu. Inak pre triedy s rovnakým výskytom spočíta súčet vzdialeností ich bodov a vráti tú, ktorej súčet vzdialeností je najmenší. V prípade, že aj súčet vzdialeností je rovnaký, tak vráti prvú nájdenú triedu funkciou *min*.

- FUNKCIA PLOT_SCATTER(K, SQUARES, FILL_SPACES)

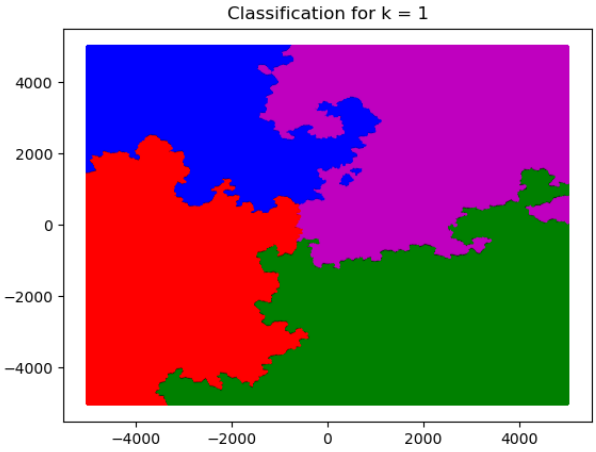
Táto funkcia zabezpečuje vyplotovanie bodových diagramov, ktoré sa uložia ako obrázok do priečinka, v ktorom je program spúšťaný. Tieto bodové diagramy obsahujú body zafarbené na základe triedy, do ktorej patria. Táto funkcia má parameter *fill_spaces*, ktorý ak je nastavený, tak táto funkcia ešte volá funkciu *classify* pre body, ktoré sa v mape nenachádzajú, aby v diagrame nevznikli biele miesta.

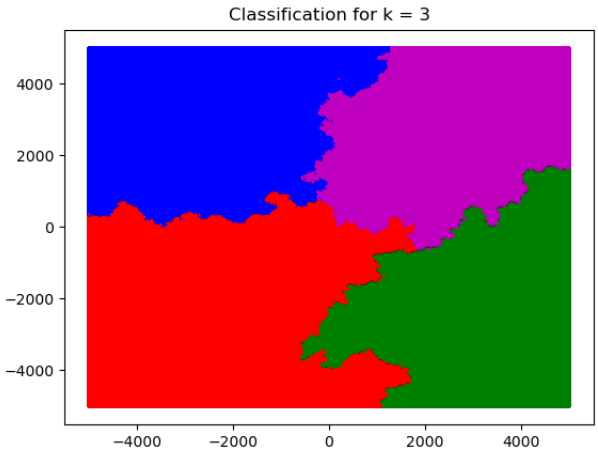
TESTOVANIE

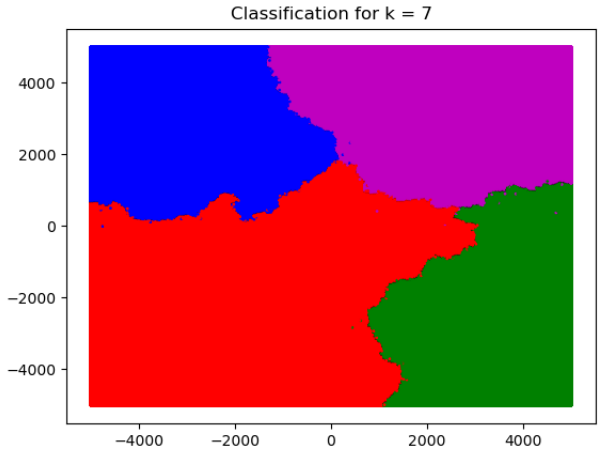
Aby som mohol zhodnotiť výsledky klasifikátorov, vykonal som 3 merania, pričom vždy sa generovalo 20000 bodov.

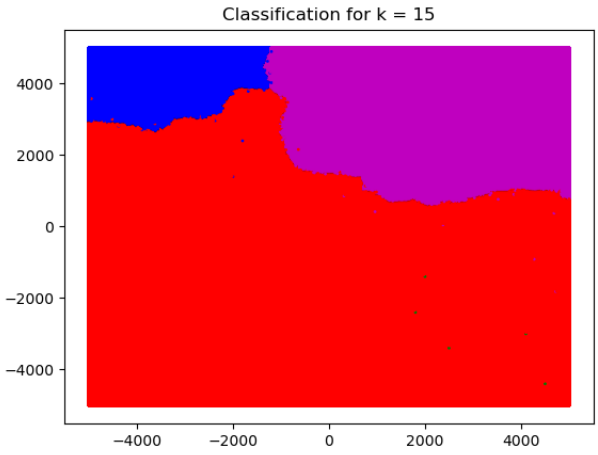
1. meranie

```
Points generated: 20000
Total time taken: 13.59 sec
----------------------------------------
>>> Results for k = 1
Classified correctly: 14380
Classified incorrectly: 5620
Success rate: 71.9%
----------------------------------------
>>> Results for k = 3
Classified correctly: 15558
Classified incorrectly: 4442
Success rate: 77.79%
----------------------------------------
>>> Results for k = 7
Classified correctly: 14606
Classified incorrectly: 5394
Success rate: 73.03%
----------------------------------------
>>> Results for k = 15
Classified correctly: 9666
Classified incorrectly: 10334
Success rate: 48.33%
```



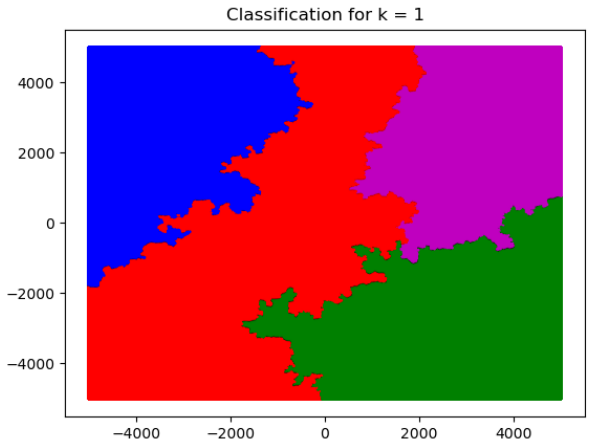


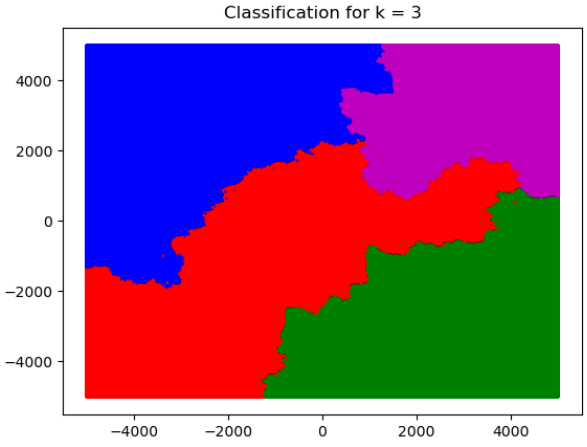


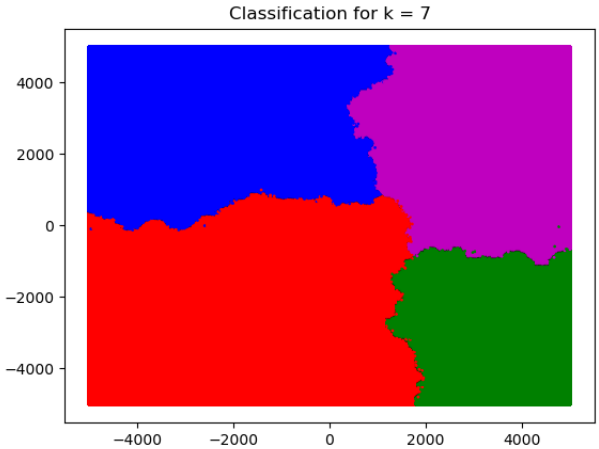


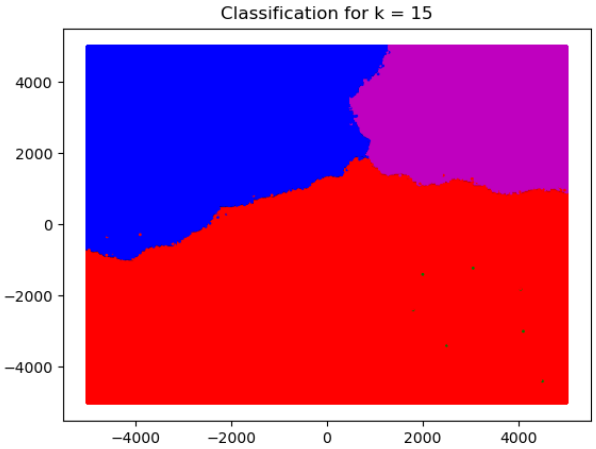
2. meranie

```
Points generated: 20000
Total time taken: 14.09 sec
----------------------------------------
>>> Results for k = 1
Classified correctly: 13713
Classified incorrectly: 6287
Success rate: 68.56%
----------------------------------------
>>> Results for k = 3
Classified correctly: 13891
Classified incorrectly: 6109
Success rate: 69.45%
----------------------------------------
>>> Results for k = 7
Classified correctly: 14719
Classified incorrectly: 5281
Success rate: 73.59%
----------------------------------------
>>> Results for k = 15
Classified correctly: 11427
Classified incorrectly: 8573
Success rate: 57.13%
```



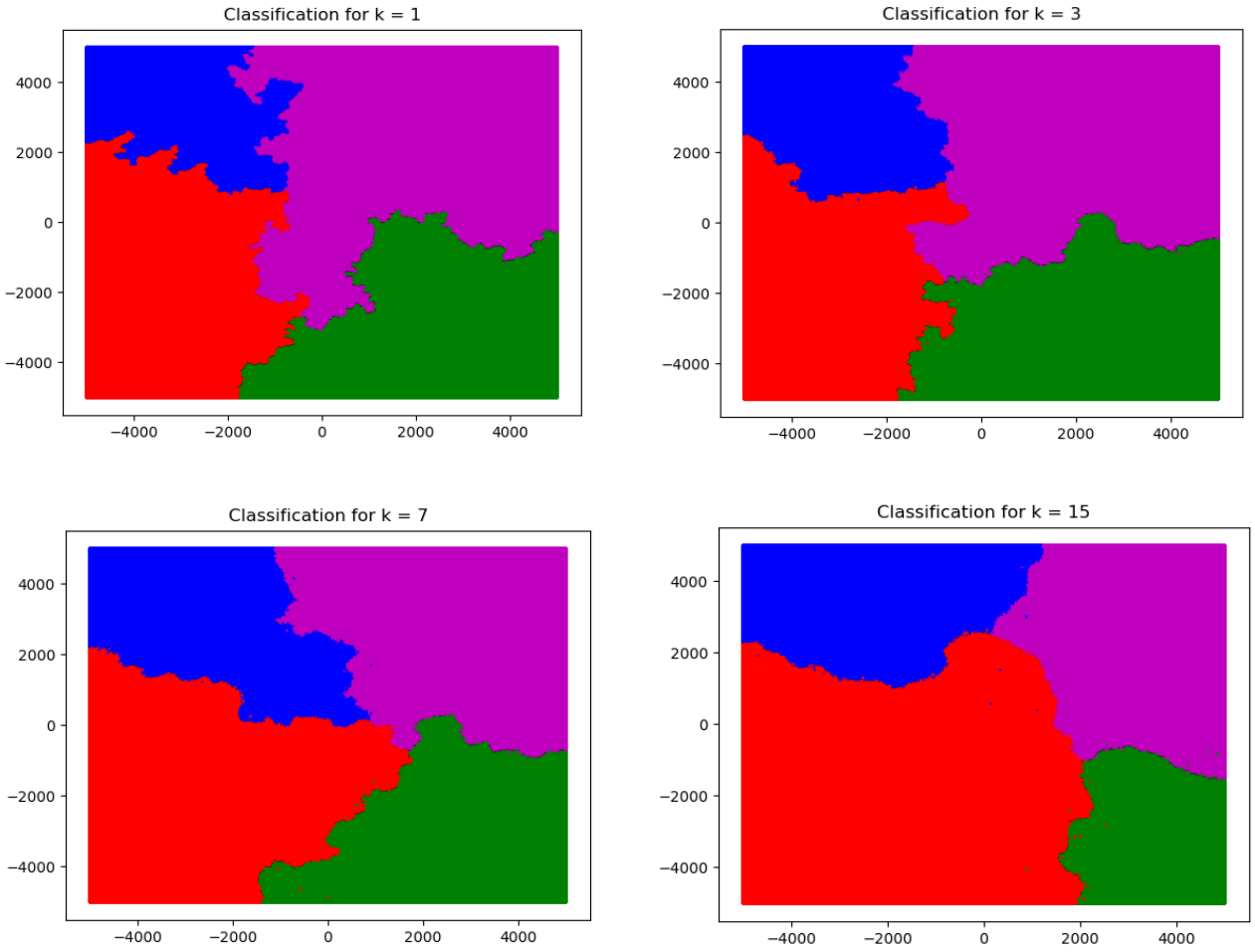






3. meranie

```
Points generated: 20000
Total time taken: 14.42 sec
----------------------------------------
>>> Results for k = 1
Classified correctly: 14159
Classified incorrectly: 5841
Success rate: 70.8%
----------------------------------------
>>> Results for k = 3
Classified correctly: 14577
Classified incorrectly: 5423
Success rate: 72.89%
----------------------------------------
>>> Results for k = 7
Classified correctly: 15056
Classified incorrectly: 4944
Success rate: 75.28%
----------------------------------------
>>> Results for k = 15
Classified correctly: 13406
Classified incorrectly: 6594
Success rate: 67.03%
```



Zhodnotenie výsledkov

- KLASIFIKÁTOR, K – 1

Úspešnosť tohto klasifikátora vo väčšine prípadov zaostávala za najúspešnejším klasifikátorom o 2-5%. Keď sa však bližšie pozrieme na rozloženie tried v mapách, tak si môžeme všimnúť, že v každej z nich sa niektorá z tried začne rozrastať do prostredia inej triedy a vznikajú výbežky. To je zapríčinené práve tým, že sa berie ohľad iba na najbližšieho suseda. Týmpádom sa pozdĺž hraníc nevie dobre vyhodnotiť, do ktorej triedy daný bod patrí. Hranice sa začnú deformovať a sú veľmi kostrbaté. Najväčší dopad to malo pri meraní číslo 2, kde sa červená trieda rozrástla cez celý stred mapy.

- KLASIFIKÁTOR, K - 3

Úspešnosť tohto klasifikátora bola vo väčšine prípadov veľmi dobrá (takmer najlepšia) a v prvom meraní dokonca najlepšia. Keď sa pozrieme na mapu rozloženia tried, tak môžeme vidieť pokrok oproti klasifikátoru 1. Na mape síce môžeme vidieť pár výbežkov do prostredia iných tried, ale celkovo sú hranice menej kostrbaté, pretože pri vyhodnocovaní bodov na hraniciach sa už prizerá

na viacero bodov ako iba jeden. Najväčší výbežok bol opäť pri meraní číslo 2, kde pravdepodobne došlo k nešťastnému generovaniu bodov.

- KLASIFIKÁTOR, K – 7

Úspešnosť tohto klasifikátora bola takmer vo všetkých prípadoch najlepšia. Zo samotnej mapy môžeme pekne vidieť, že výbežky už takmer zmizli a ak aj sú, tak sú minimálne. Hranice sú veľmi pekné a prostredie tried je pekne rozlíšiteľné. Dokonca nemal problémy ani pri meraní číslo 2, kde predchádzajúce klasifikátory zlyhávali. Počet bodov, na ktorý sa prihliada v tomto prípade, sa zdá byť ideálny pre klasifikáciu daného bodu. Aj keď klasifikátor, k – 3 dosahoval tiež veľmi dobré výsledky, mapa tohto klasifikátoru je krajšia a lepšie interpretovateľná, preto by som toto riešenie označil ako celkovo najlepšie.

- KLASIFIKÁTOR, K – 15

Úspešnosť tohto klasifikátora bola vo všetkých prípadoch najhoršia. V prvom a druhom meraní môžeme pekne vidieť, ako červená trieda naprosto prerástla zelenú triedu a takmer aj ďalšie. V poslednom meraní síce dosiahol veľmi slušný výsledok, ale to bude pravdepodobne zapríčinené dobrým generovaním bodov. V tomto prípade sa prizerá už na príliš veľa bodov a ak sa v malom priestore vytvorí dostatočná majorita, tak sa postupne začne veľmi rýchlo rozrastať. Toto riešenie je veľmi náchylné na začiatok, pretože ak sa na začiatku začne klasifikovať zle, tak celé riešenie zlyháva. Toto riešenie je s istotou celkovo najhoršie.

ZÁVER

Na záver by som rád poskytol zhodnotenie tohto zadania. Klasifikácia je síce veľmi zaujímavá, ale myslím, že v tomto prípade sme začínali s príliš malým počtom bodov v testovacej vzorke. Ak by sme mali väčší počet bodov, tak výsledky klasifikátorov budú určite presnejšie. Ak však zámerom tohto zadania bolo poukázať práve na chybné dáta, ktoré klasifikátory vytvárajú, tak to splnilo svoj účel.