Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Tibor Vanek

Zadanie 4a – **Klasifikácia**

Predmet: Umelá Inteligencia
Čas cvičenia: Štvrtok 14:00
Cvičiaci: Ing. Ivan Kapustík

Špecifikácia zadania

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.

Opis riešenia

Zadanie som vypracoval v *Python 3.10* a použil som IDE *PyCharm*. V programe používam knižnice `random`, `math`, `matplotlib` a `time`. Interval zo zadania a súradnice bodov si premapovávam z [-5000, 5000] na [0, 10000] pre jednoduchosť vykonania, ale pri vizualizácií sú zobrazené hodnoty podľa zadania.

V programe mám 1 bod reprezentovaný cez triedu `Bod`, ktorá má vlastnosti:

- `x` –> x-ová súradnica bodu
- `y` –> y-ová súradnica bodu
- `color` –> farba bodu

Inštancie tejto triedy sa vkladajú do 4 listov (1 pre každú možnosť k).

Na reprezentáciu 2D priestoru ešte vytváram 2-rozmerné pole (mapa) veľkosti 10 000 x 10 000, do ktorého vkladám informáciu, ktorý bod je vyfarbený. Táto mapa slúži hlavne na zaručenie toho, že sa **nebudú** vyskytovať pri generovaní **duplikáty** bodov.
Na začiatok si inicializujem **mapu**, aby boli všade prázdne miesta, t. j. nuly. Toto inicializovanie trvá určitý čas, ktorý ale **nezapočítavam** do dĺžky trvania algoritmu. V jednom cykle **zároveň generujem aj klasifikujem** body s náhodnými hodnotami x a y tak, že sa striedajú farby – vygeneruje sa *R* bod, potom *G*, *B* a *P*. Po vygenerovaní sa každý bod klasifikuje pre všetky 4 *k* hodnoty a vypočíta sa pre ne chybovosť. Pred generovaním bodu sa kontroluje pravdepodobnosť 1-100%. Ak sa náhodné číslo vygeneruje ako 100, nastala 1% šanca a bod sa vygeneruje do celého priestoru a následne sa klasifikuje. Inak sa vygeneruje do svojho určeného rozmedzia v mape.
Klasifikácia bodov prebieha po generovaní naraz pre všetky *k*. Každá hodnota *k* má list, plný `Bod`ov. Funkcia `classify()` vypočíta **euklidovskú vzdialenosť** vkladaného bodu **od všetkých ostatných bodov v mape** a uloží tieto vzdialenosti a ich farby do listu. Tento list zoraďujem pomocou vlastnej implementácie **k-smallest-values** algoritmu. Takto vyberiem k bodov s najmenšími vzdialenosťami a vrátim najčastejšie sa vyskytujúcu hodnotu (toto môže spôsobovať menšiu presnosť klasifikátora, lebo ak je rovnaký počet bodov 2 farieb, nenastáva náhodný výber ale vyberie sa farba, ktorá dosiahla maximum ako prvá).
Chybovosť sa po klasifikovaní určí pre všetky *k* a vloží sa do listu. Po skončení generovania a klasifikovania bodov sa body uložené v 4 jednotlivých listoch (1 pre každé *k*) vizualizujú pomocou knižnice *matplotlib* a vypíše sa **uplynutý čas** vykonania algoritmu a **úspešnosť klasifikátora** pre každé *k*.

Používateľské postredie

```
Map initialized, starting clock
>500 points generated
done, time elapsed: 12.3247 s ...plotting
2500 bodov  vygenerovanych, 625 per color
pocet chyb:
K=1: 691 -> 72 % uspesnost
K=3: 679 -> 73 % uspesnost
K=7: 658 -> 74 % uspesnost
K=15: 1280 -> 49 % uspesnost
END
```

Pre zmenu počtu bodov:

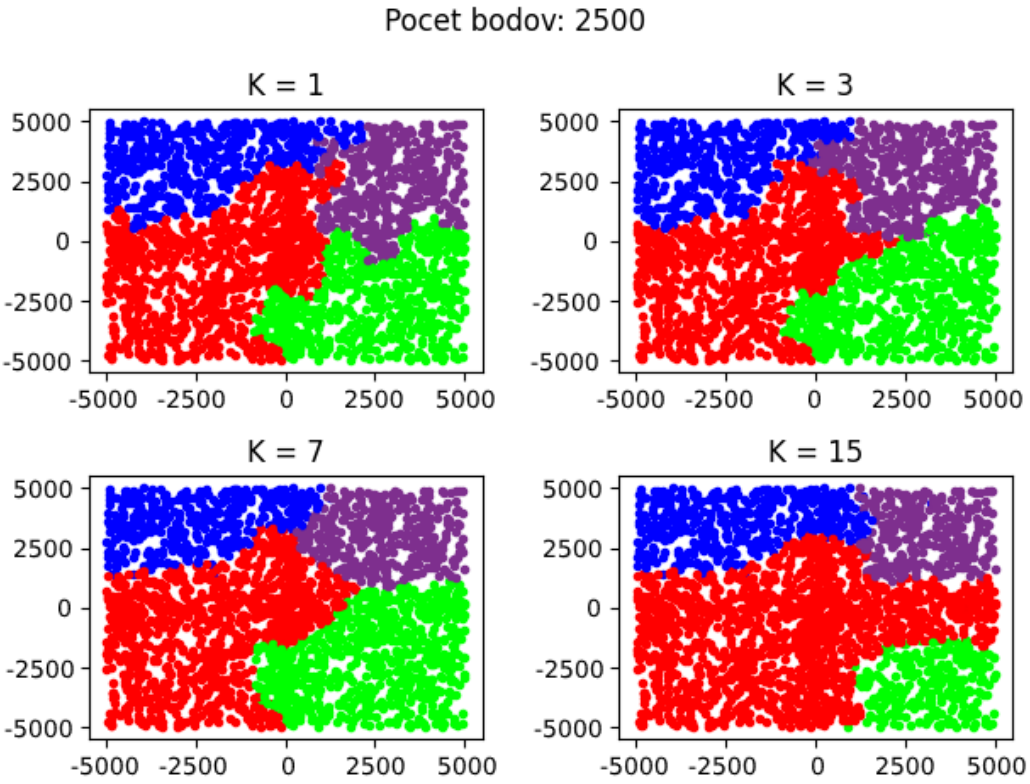
```
158     points_per_color = 625
```

(zobrazuje počet bodov pre 1 farbu, pre všetky farby je treba toto číslo vynásobiť štyrmi)

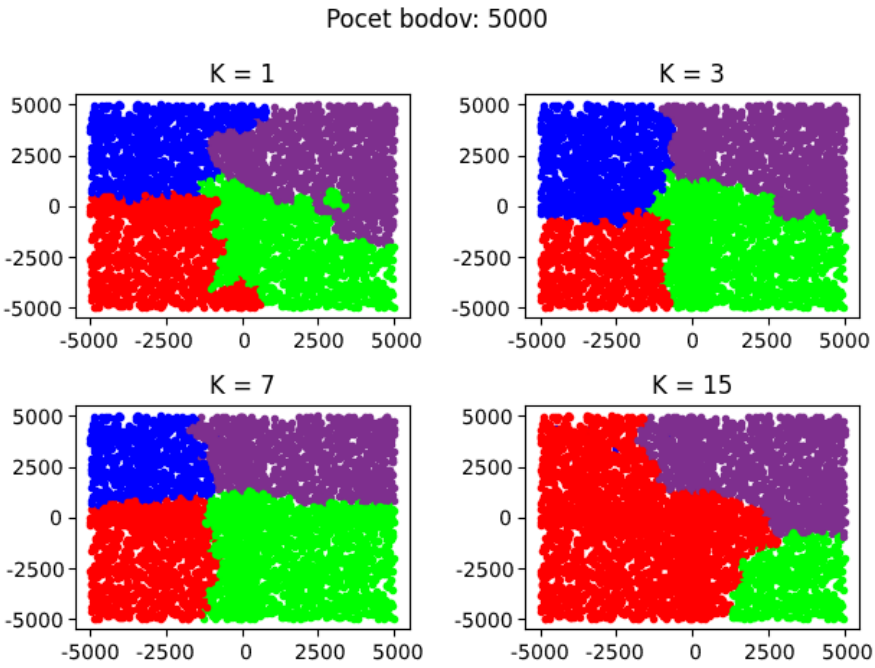
Testovanie a zhodnotenie

Program som testoval pre viaceré hodnoty, najviac 20 000 bodov (spolu). Tento výsledok trval cca 19 minút na vykreslenie.

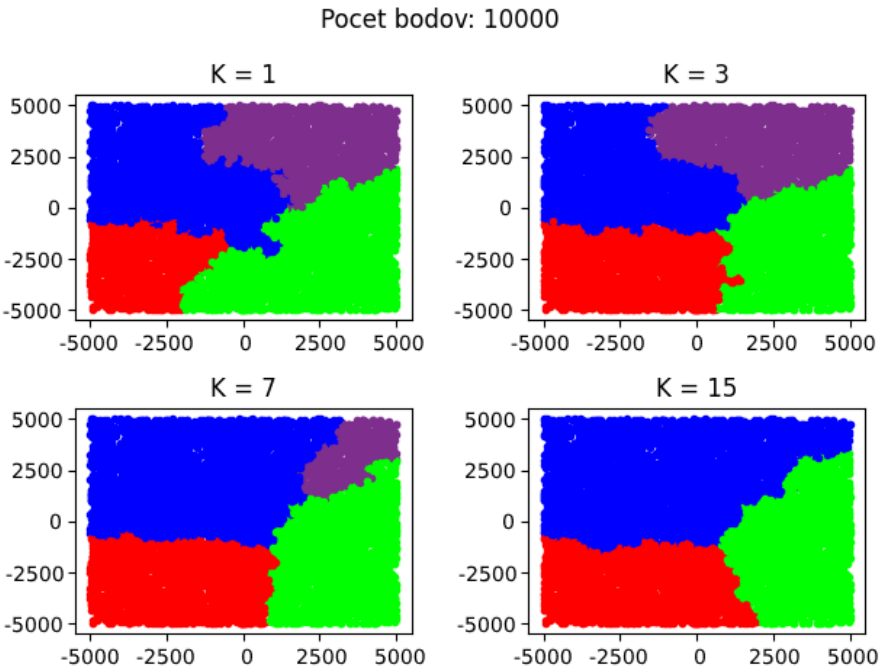
Pár vizualizovaných testov:



čas algoritmu: 12.4 sekúnd
počet chýb:
K=1: 659 -> 74 % úspešnosť
K=3: 630 -> 75 % úspešnosť
K=7: 697 -> 72 % úspešnosť
K=15: 970 -> 61 % úspešnosť

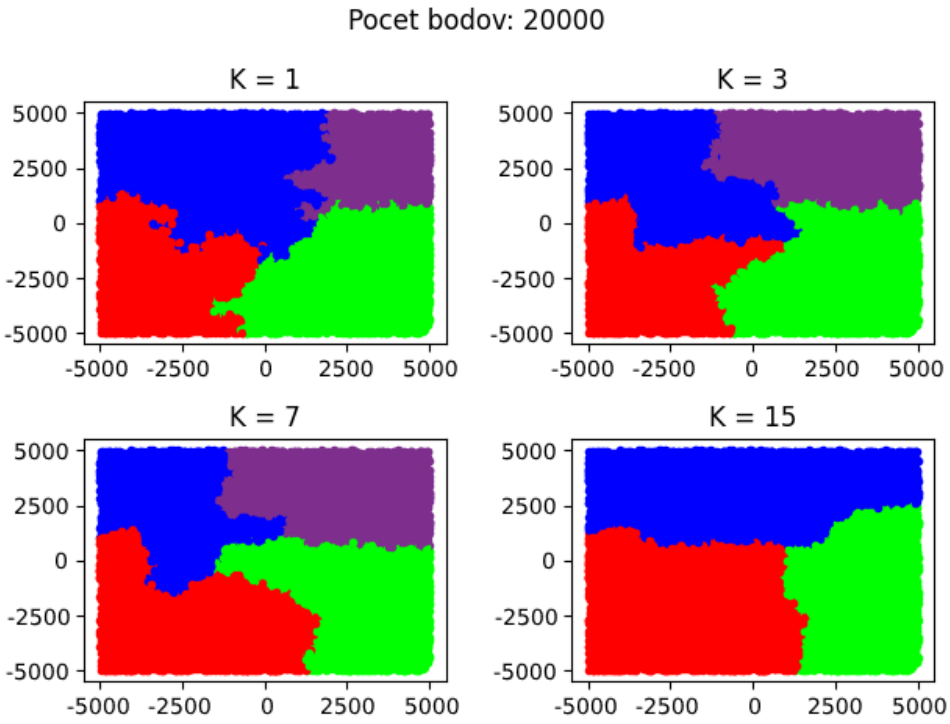


čas algoritmu: 58 sekúnd
počet chýb:
K=1: 1184 -> 76 % úspešnosť
K=3: 1158 -> 77 % úspešnosť
K=7: 1299 -> 74 % úspešnosť
K=15: 2122 -> 57 % úspešnosť



čas algoritmu: 4 min, 46 sekúnd
počet chýb:
K=1: 2861 -> 71 % úspešnosť

K=3: 2731 -> 73 % úspešnosť
K=7: 3230 -> 67 % úspešnosť
K=15: 4045 -> 59 % úspešnosť



čas algoritmu: 19 minút 36 sekúnd
počet chýb:
K=1: 5369 -> 73 % úspešnosť
K=3: 4892 -> 75 % úspešnosť
K=7: 5237 -> 74 % úspešnosť
K=15: 7936 -> 60 % úspešnosť

Zhodnotenie

Celková úspešnosť algoritmu sa pohybuje v priemere okolo 70%. Pri k = 15 býva najmenšia úspešnosť kvôli hľadaniu susedov (farby sa prekryjú).
Efektivita algoritmu by však mohla byť lepšia - kvôli narastajúcej časovej náročnosti s počtom bodov na klasifikovanie. Toto je spôsobené prehľadávaním listu *zafarbene_body*, ktorý sa prehľadáva celý a pridaním každého bodu sa zväčšuje. Možné zlepšenia by moli zahŕňať napríklad rozdelenie mapy na viac častí, aby sa nemuseli porovnávať vzdialenosti všetkých bodov (s týmto som však pri mojej implementácii mal problémy, tak som to nezahrnul v riešení).