Fedor Viest
ID: 103177

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológii

Ilkovičova 2, 842 16, Bratislava

UI Zadanie 4 – Klasifikácia KNN algoritmus – Zadanie 4a

Fedor Viest

Termín odovzdania: 13.12.2021

Prednášajúci: Ing. Lukáš Kohútka, PhD.

Cvičiaci: Ing. Boris Slíž

Fedor Viest
ID: 103177

Obsah

Fedor Viest
ID: 103177

Zadanie

V zadaní bolo úlohou spraviť klasifikátor bodov s využitím KNN algoritmu. Tento algoritmus spočíva v tom, že sa zoberie **N** najbližších susedov, na základe ktorých sa pridelí kategória vygenerovanému bodu. Na začiatku je potrebné vygenerovať 5000 bodov z každej kategórie podľa zadaných požiadaviek:

- 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

Body sú klasifikované funkciou **classify()**, pričom počet susedov, s ktorým algoritmus pracuje je 1, 3, 7, alebo 15.

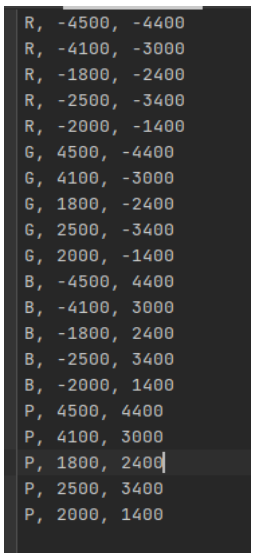
Nakoniec program vykreslí tieto body do grafu a vypíše úspešnosť klasifikátora s časom.

Fedor Viest
ID: 103177

Implementácia

Load_coords()

Na začiatku sa počiatočné súradnice bodov spolu s ich kategóriou načítajú zo súboru do poľa. Súbor **coords.txt** vyzerá nasledovne:



Generate_points()

V tejto funkcii program vygeneruje **5000** bodov z každej kategórie, pričom 99% z bodov bolo vygenerovaných podľa zadaných pravidiel uvedených v zadaní a 1% bolo vygenerovaných na celej ploche.

Classify()

V tejto funkcii program klasifikuje body podľa KNN algoritmu. Vo funkcii je jeden hlavný cyklus, ktorý prechádza po vygenerovaných bodoch. V tomto cykle je ďalší cyklus, ktorý prechádza po už určených bodoch. V tomto cykle program počíta vzdialenosť od vygenerovaného bodu k aktuálnemu bodu z určených bodov. Následne program zoradí tieto vzdialenosti vzostupne a vyberie **N** (podľa susedov) prvých prvkov z tohto poľa. V ďalšom cykle program počíta výskyt jednotlivých farieb.

Tieto hodnoty vloží do poľa a znova zoradí, tentoraz zostupne. Prvý prvok z tohto poľa sa nastaví ako nová kategória pre daný bod. Následne sa porovná novo vygenerovaná farba s pôvodnou a ak sa rovnajú, zvýši sa počítadlo o 1.

Print_plots()

Na vykresľovanie používam knižnicu matplotlib a funkciu **scatter().** Funkcia prechádza po všetkých bodoch a ukladá x, y a farby do jednotlivých polí.

Reprezentácia údajov

Body sú reprezentované v poli nasledovne:

Bod(x -> int, y -> int, farba -> string)

Fedor Viest
ID: 103177

Testovanie

Program je testovaný pri vygenerovaní 20 000 bodov a na počte susedov 1, 3, 7, 15.

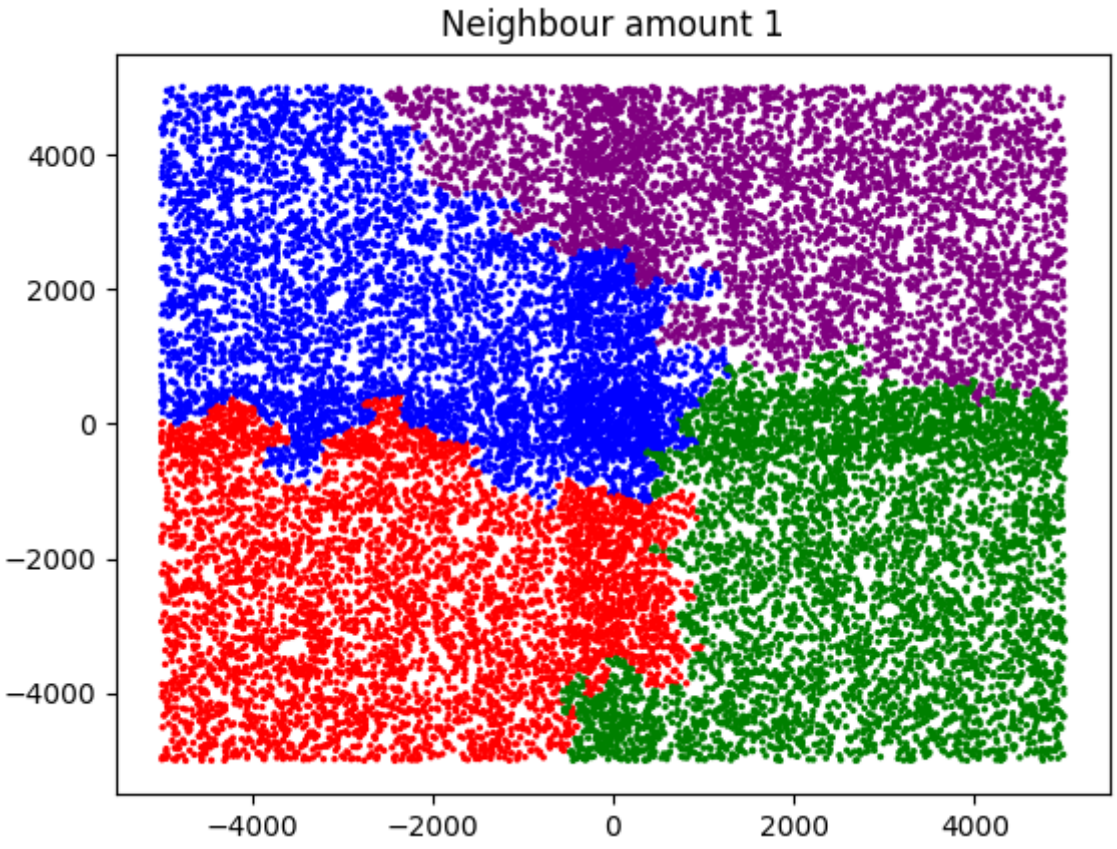
V programe sa testuje:

- čas, za ktorý klasifikuje a vykreslí body
- percentuálna úspešnosť klasifikátora oproti vygenerovaným bodom

Testovanie je ukončené vizualizáciou klasifikácie bodov.

Testovanie na počte susedov **1**:

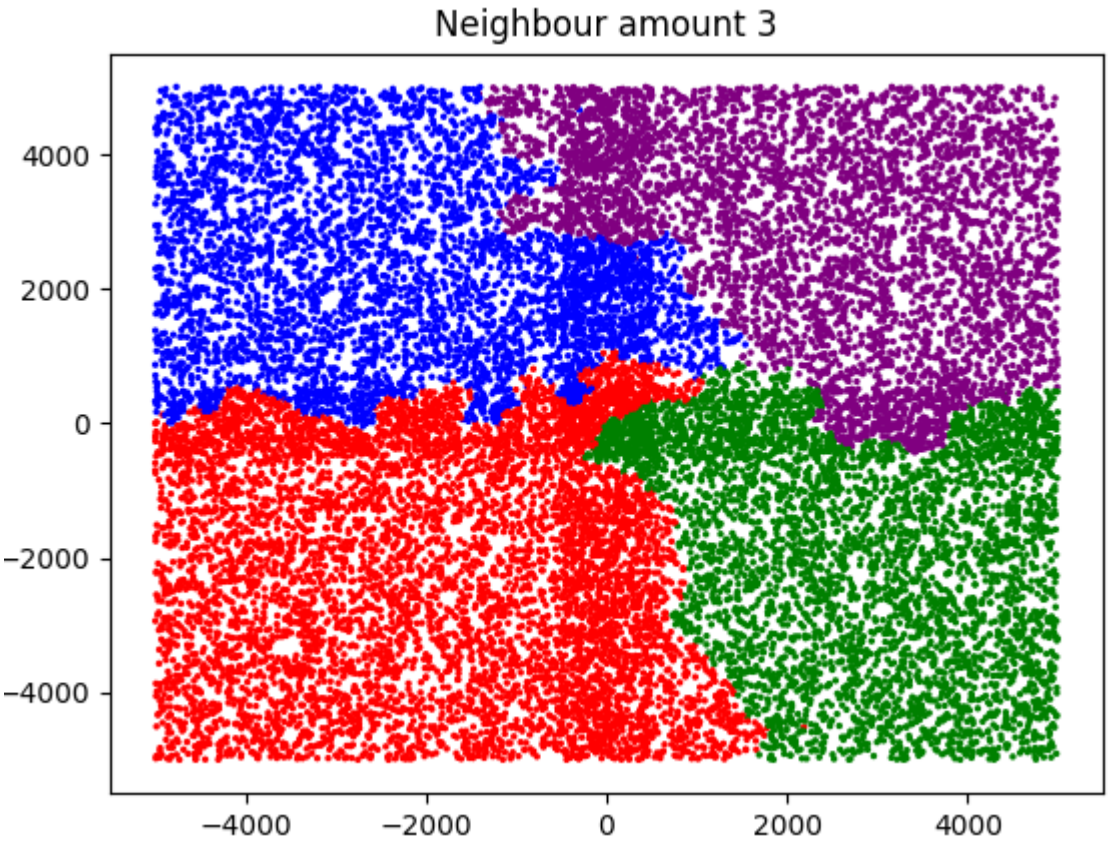
```
Classification with 1 neighbours
Took 424 seconds
Correctly classified points according to classifier: 76.95%
```



Fedor Viest
ID: 103177

Testovanie na počte susedov **3**:

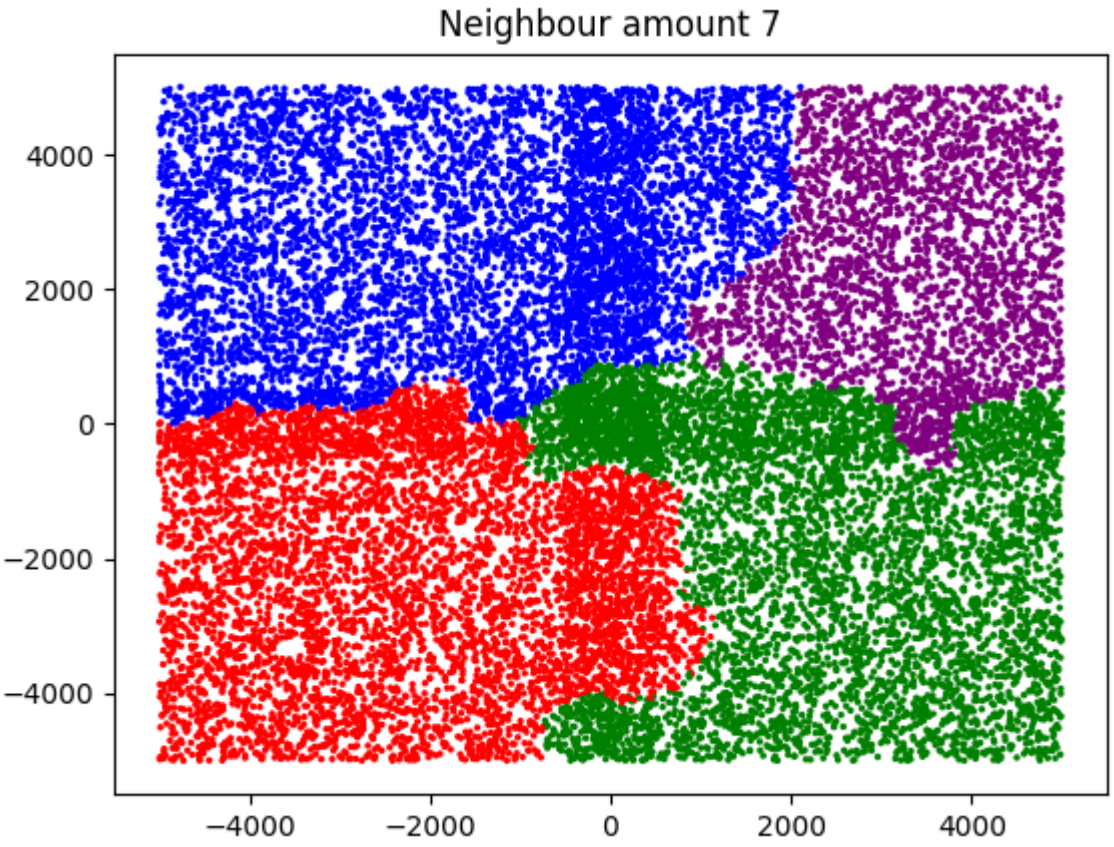
```
Classification with 3 neighbours
Took 453 seconds
Correctly classified points according to classifier: 77.52%
```



Fedor Viest
ID: 103177

Testovanie na počte susedov **7**:

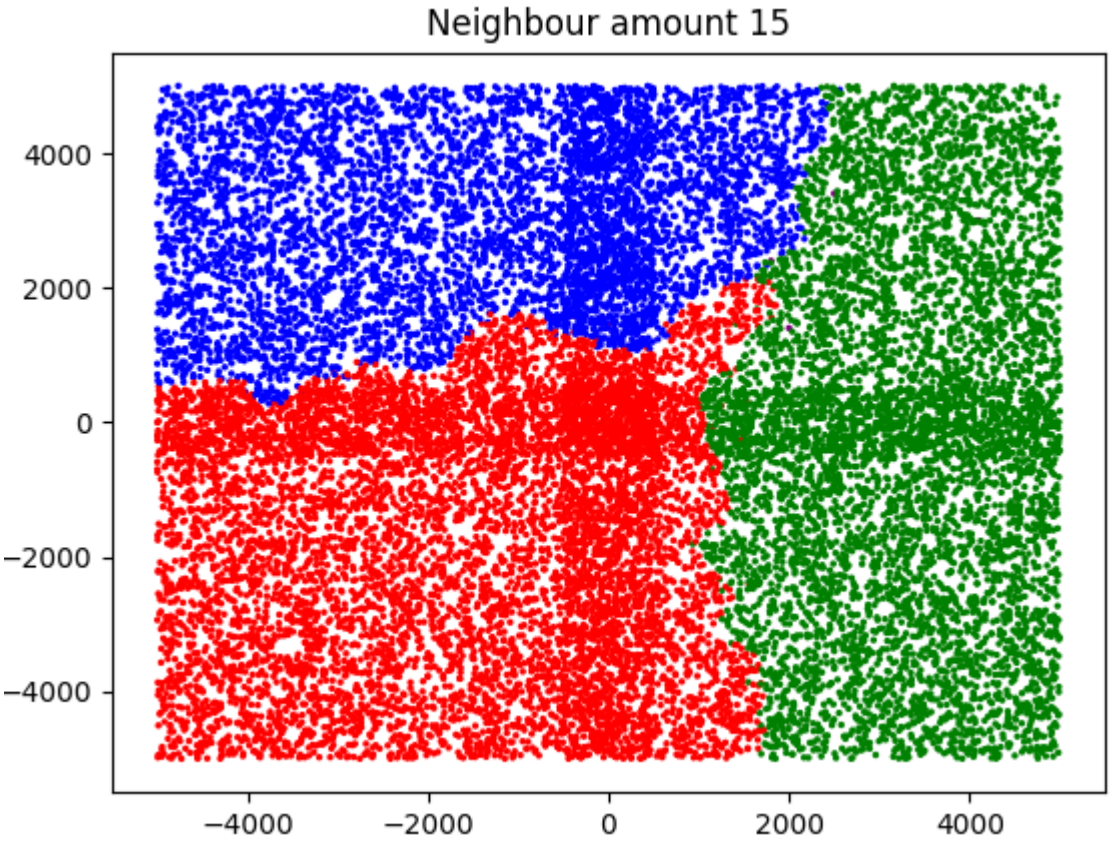
```
Classification with 7 neighbours
Took 454 seconds
Correctly classified points according to classifier: 75.69%
```



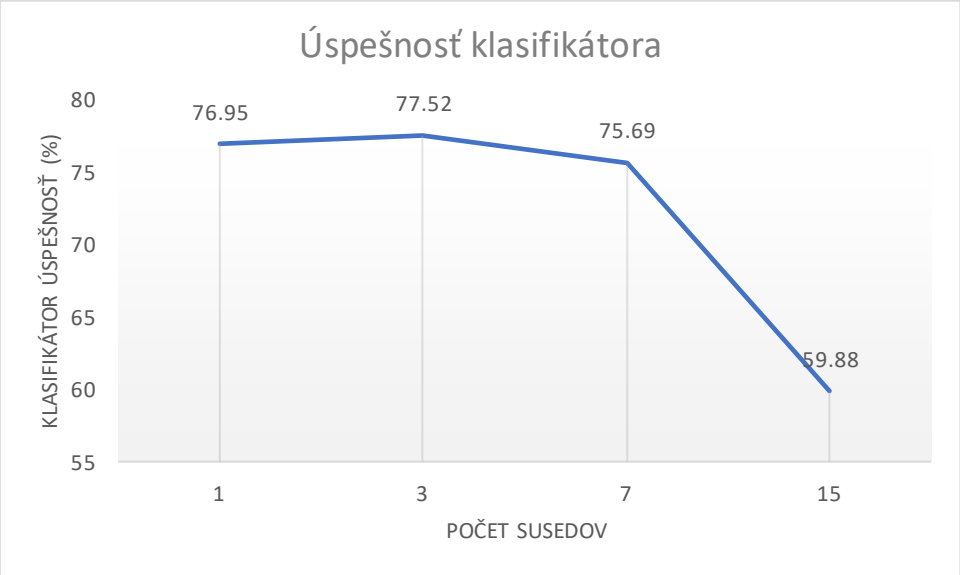
Fedor Viest
ID: 103177

Testovanie na počte susedov **15**:

```
Classification with 15 neighbours
Took 473 seconds
Correctly classified points according to classifier: 59.88%
```



Fedor Viest
ID: 103177



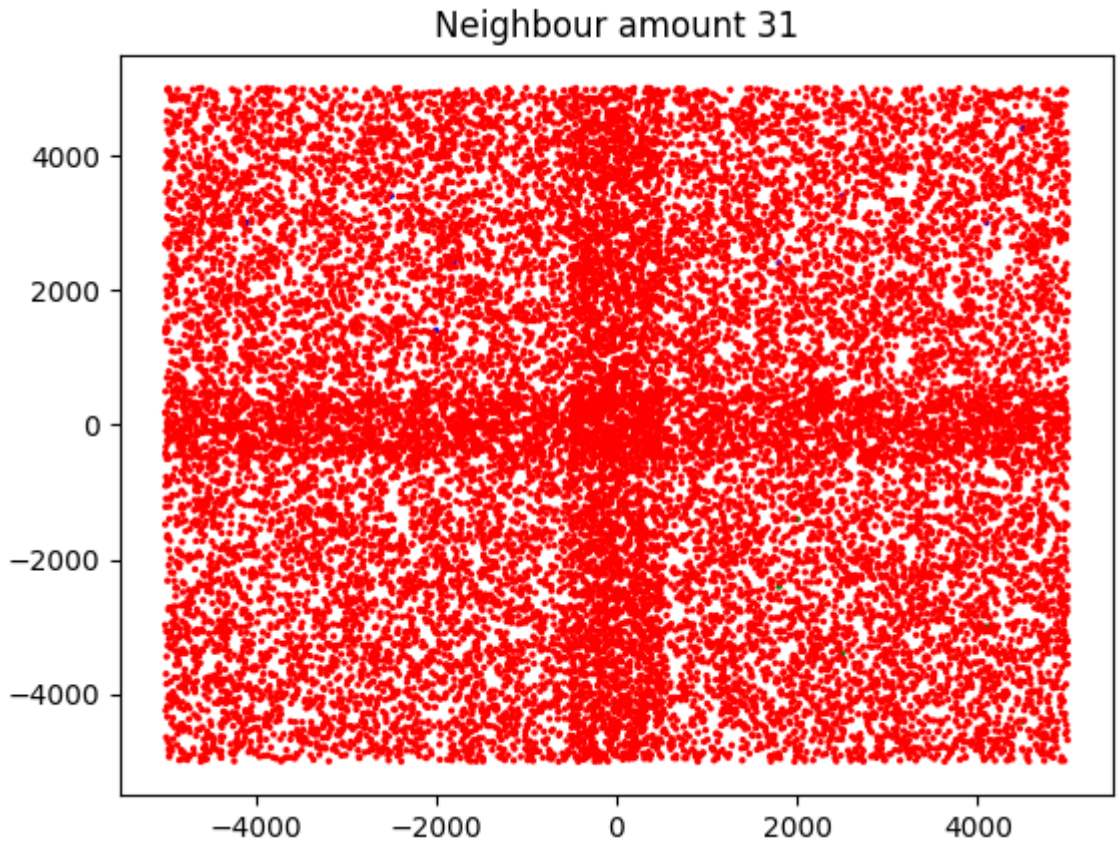
Ako je možné z tohto ale aj predošlých grafov vidno, najlepšie výsledky dosahujú počty susedov 1, 3 a 7. Pri počte susedov 15 sa už stratila jedna z farieb a preto je úspešnosť klasifikátora nižšia.

Záver testovania

Pri vygenerovaní 15 susedov bola úspešnosť klasifikátora zlá, pretože algoritmus už bral veľmi vzdialené body, naopak výsledky môžu byť nie úplne správne pri generovaní málo susedov, napríklad 1 suseda. V tomto prípade sú priemerne najlepšie prípady 3 a 7 susedia.

Na potvrdenie tvrdenia, že generovanie príliš mnoho susedov dáva zlé výsledky, som skúsil spustiť algoritmus pre 31 susedov. V tomto prípade je v grafe prítomná len 1 farba, všetky ostatné okrem počiatočných 20 bolo zmenených na červené.

Fedor Viest
ID: 103177



Systémové špecifikácie

Operačný systém: Windows 10

IDE: Pycharm 2021.2.3

Knižnice: matplotlib, random, copy, sys, time

Python: verzia 3.8.3

Fedor Viest
ID: 103177

Zhodnotenie

Zadanie by sa dalo optimalizovať napríklad rozdelením grafu do viacerých políčok, čo by malo za dôsledok, že by nebolo potrebné prechádzať všetky už klasifikované body, ale iba body v štvorčekoch, ktoré sa nachádzajú okolo daného bodu. Ďalšou optimalizáciou by bolo využitie multiprocesov, kde by sa program pre každé **K** vykonával v inom procese, čo by síce spomalilo vykonanie jedného testu, ale signifikantne zrýchlilo pri všetkých testoch. Napríklad v mojom prípade jeden test trval cca 450 sekúnd (čo je spolu cca 30 minút). V prípade, že by som využil multiprocesy, by vykonanie všetkých testov trvalo najviac 15 minút.