Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Zadanie 4a – klasifikácia

Dokumentácia

Peter Smreček

AIS ID: 103130

E-mail: [xsmrecek@stuba.sk](mailto:xsmrecek@stuba.sk)

Predmet: UI

Deň a čas cvičenia: Utorok 16:00

Semester: ZS 20/21

Ročník: 2.

Obsah

[1. Zadanie 2](#_Toc58697661)

[2. Opis algoritmu 2](#_Toc58697662)

[2.1. Reprezentácia údajov problému 2](#_Toc58697663)

[2.2. Generovanie nových bodov 2](#_Toc58697664)

[2.3. Klasifikácia a pridanie nových bodov do plochy 2](#_Toc58697665)

[2.4. Klasifikácia celej plochy 3](#_Toc58697666)

[2.5. Vizualizácia 3](#_Toc58697667)

[3. Testovanie 3](#_Toc58697668)

[3.1. k = 1 4](#_Toc58697669)

[3.2. k = 3 5](#_Toc58697670)

[3.3. k = 7 6](#_Toc58697671)

[3.4. k = 15 7](#_Toc58697672)

[3.5. Zhodnotenie testovania 8](#_Toc58697673)

[4. Používateľské rozhranie a ovládanie programu 9](#_Toc58697674)

[5. Záver 9](#_Toc58697675)

# Zadanie

V 2D priestore so súradnicami X a Y v intervaloch od -5000 do +5000 sa nachádzajú body patriace 4 triedam. Na začiatku je v priestore 5 bodov z každej triedy, teda dokopy 20 bodov.

Úlohou je naprogramovať klasifikátor novo vkladaných bodov, ktorý priradí bodu triedu a vloží ho do 2D poľa. Postupne sa generuje 5000 bodov z každej triedy a vkladajú sa v poradí, že nikdy po sebe nenasledujú 2 body rovnakej triedy. Návratovú hodnotu klasifikátora porovnávame s pôvodnou triedou, s akou bol bod vygenerovaný.

Klasifikátor používa kNN algoritmus, pričom parameter k bude 1, 3, 7 alebo 15 a pôvodná vygenerovaná sada 20000 bodov bude pre každý experiment rovnaká.

Prázdne miesta v 2D ploche vyfarbíme celú podľa klasifikátora.

# Opis algoritmu

Program som implementoval v programovacom jazyku Python s použitím štandardných balíkov a s prevzatým kódom implementácie KD stromu, ktorý sa používa na hľadanie k susedov pri finálnom vyfarbovaní celej plochy.

## Reprezentácia údajov problému

Plochu reprezentujem maticou o veľkosti 10001 x 10001 bodov. Matica obsahuje na každom políčku číslo z rozsahu 0-4. 0 symbolizuje, že daný bod je biely, 1 symbolizuje, že je červený, 2 zelený, 3 modrý a 4 fialový. Do matice je na začiatku vložených 20 bodov daných zadaním.

## Generovanie nových bodov

Postupne generujem 4 polia o veľkosti 5000 bodov každé. Tieto polia následne spájam do jedného poľa o veľkosti 20000 súradníc s tým, že súradnice sa v ňom striedajú v poradí červená, zelená, modrá, fialová. Všetky súradnice sú jedinečné.

## Klasifikácia a pridanie nových bodov do plochy

Každý z 20000 vygenerovaných bodov je vložený do plochy funkciou vytvor\_testovaciu\_sadu. V tejto funkcií je každý bod klasifikovaný funkciou klasifikator\_testovacich\_bodov, ktorá vráti triedu novo vkladaného bodu na základe najpočetnejšej farby z farieb k bodov v jeho okolí. Následne sa vyhodnotí, či sa klasifikátorom určená trieda zhoduje s triedou určenou pri generovaní tohto bodu. Klasifikácia prebieha bruteforce hľadaním najbližších bodov.

## Klasifikácia celej plochy

Pri klasifikácií celej plochy sa používa funkcia klasifikator\_zvysnych\_bodov, ktorá na hľadanie k najbližších susedov nepoužíva bruteforce metódu, ale prevzatú implementáciu KD stromu. Implementáciu KD stromu som prevzal z <https://github.com/Vectorized/Python-KD-Tree/blob/master/kdtree.py>.

Táto funkcia je volaná funkciou vyfarbi\_mapu, ktorá ju volá pre každý v poradí niekoľký bod ktorý je zadaný argumentom. Nevyfarbuje sa teda matica o veľkosti 10001 x 10001 bodov, ale napríklad iba matica o veľkosti 400 x 400 bodov, ak sa vyfarbuje len každý 25. bod v poradí.

Obe klasifikácie používajú vlastnú funkciu pre výpočet euklidovskej vzdialenosti.

## Vizualizácia

Na vizualizáciu používam 2 rôzne funkcie. Pre vizualizáciu poľa vložených a oklasifikovaných súradníc používam funkciu vizualizuj\_pole, ktorá vytvorí plochu jednotlivých bodov.

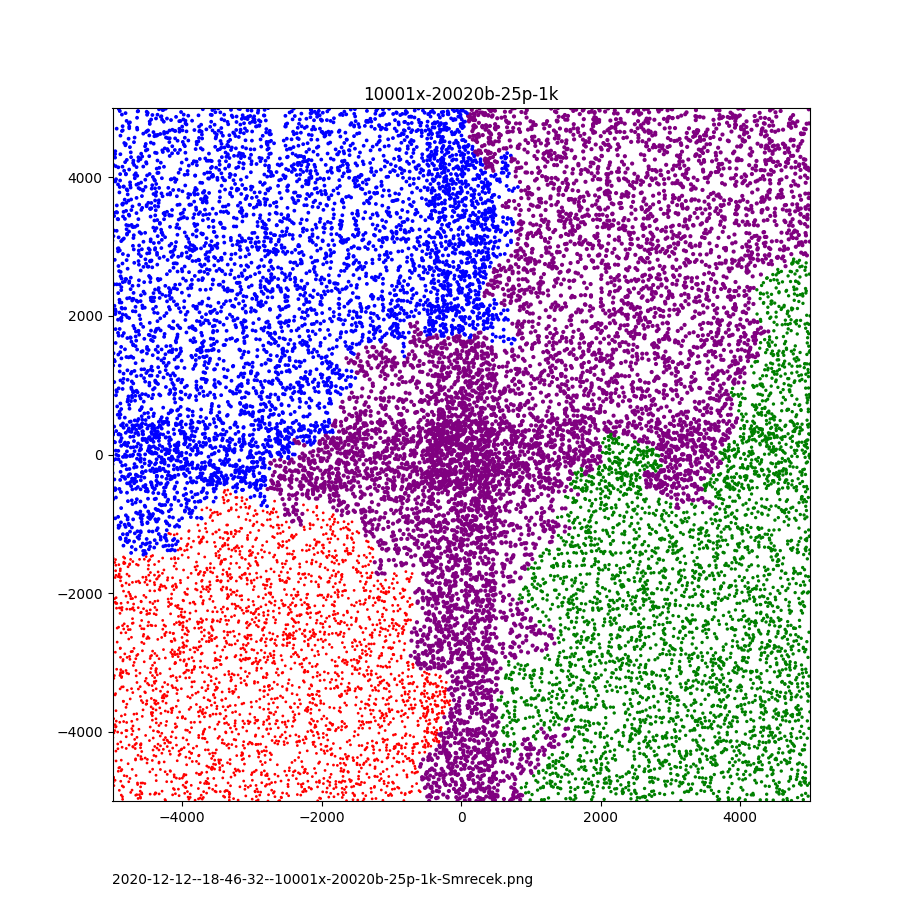
Pre vizualizáciu celej ofarbenej matice používam funkciu vizualizuj\_maticu, ktorá zoberie ako vstupný argument maticu, 2D pole kde na každom indexe sa nachádza číslo 0-4, čo symbolizuje farbu daného políčka.

Funkcie podľa vstupného argumentu buď obrázok ukážu, alebo uložia. Obrázok je označený dátumom a časom vytvorenia s charakteristikou testu pozostávajúcou z rozmeru matice, počtu testovacích bodov, každý koľký bod sa vizualizuje pri vizualizácií celej zafarbenej plochy, zvolená hodnota k a prípadne úspešnosť klasifikovania bodov v percentách. Pri vizualizácií testovacích bodov sa vizualizujú vždy všetky body. Iba pri vizualizácií celej plochy sa vizualizuje každý niekoľký bod.

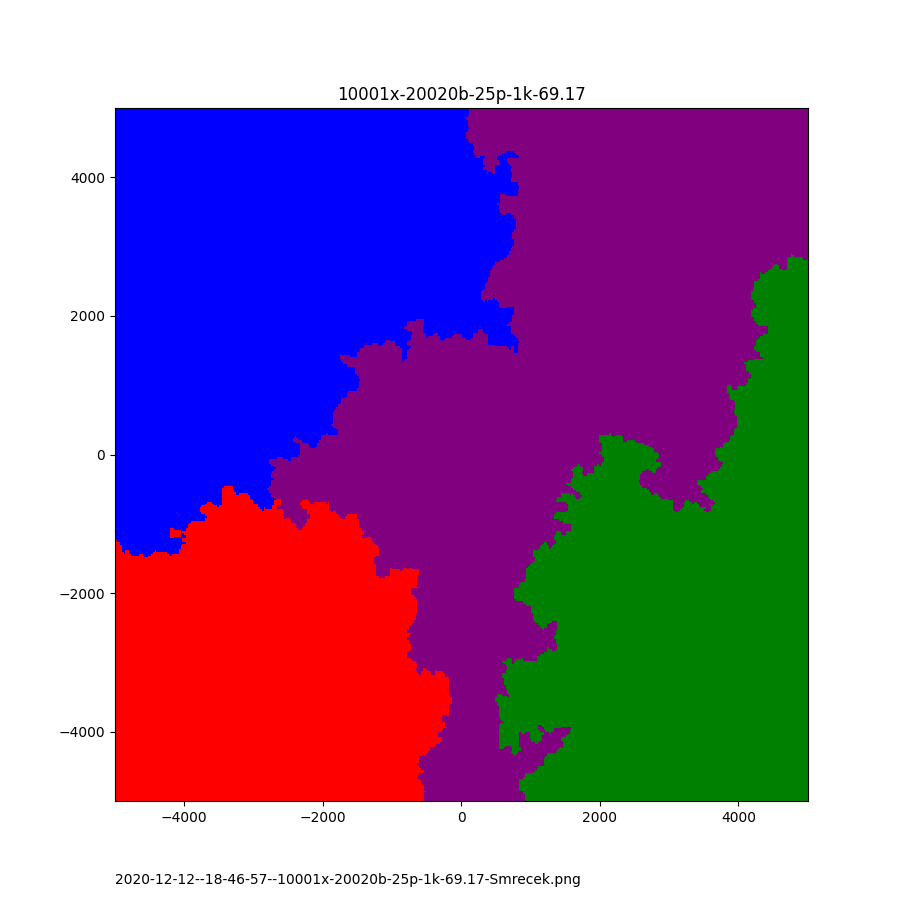
# Testovanie

Podľa požiadaviek zadania uvádzam 4 testovacie scenáre pre rôznu hodnotu k pre kNN algoritmus. Pre každý test uvádzam vizualizáciu testovacích bodov a vizualizáciu plochy, ktorú tieto body ohraničujú. Spoločne pre všetky testy uvádzam zhodnotenie testovania.

## k = 1

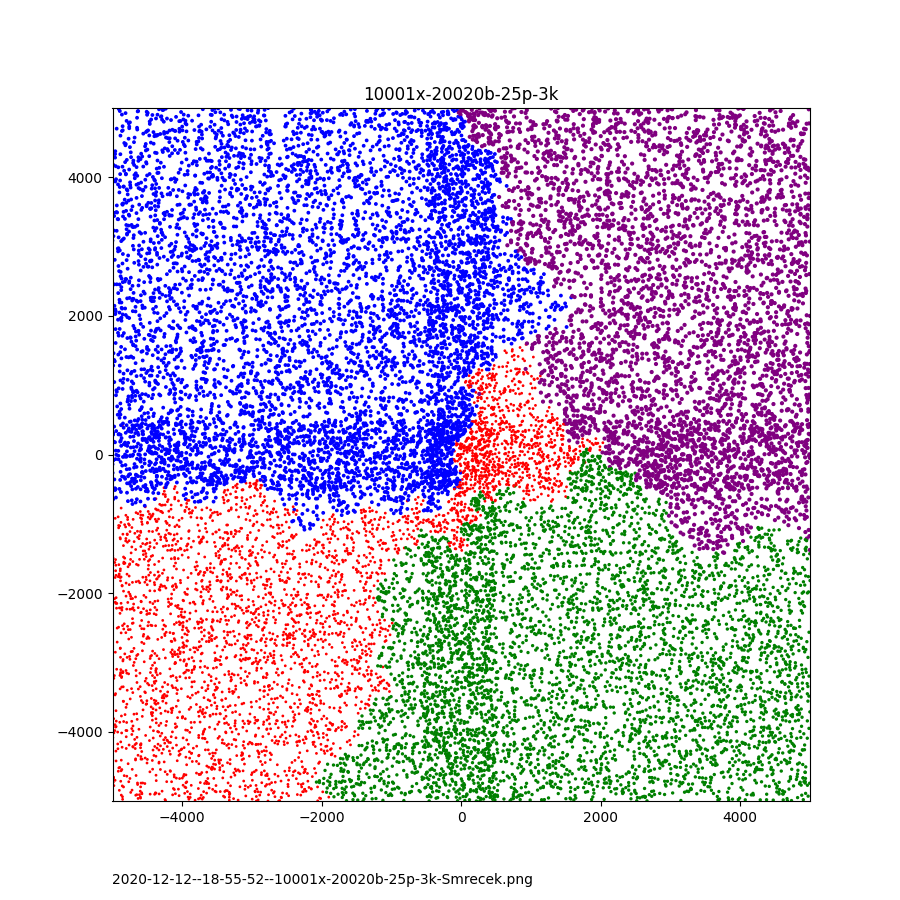


Vizualizácia testovacích bodov:

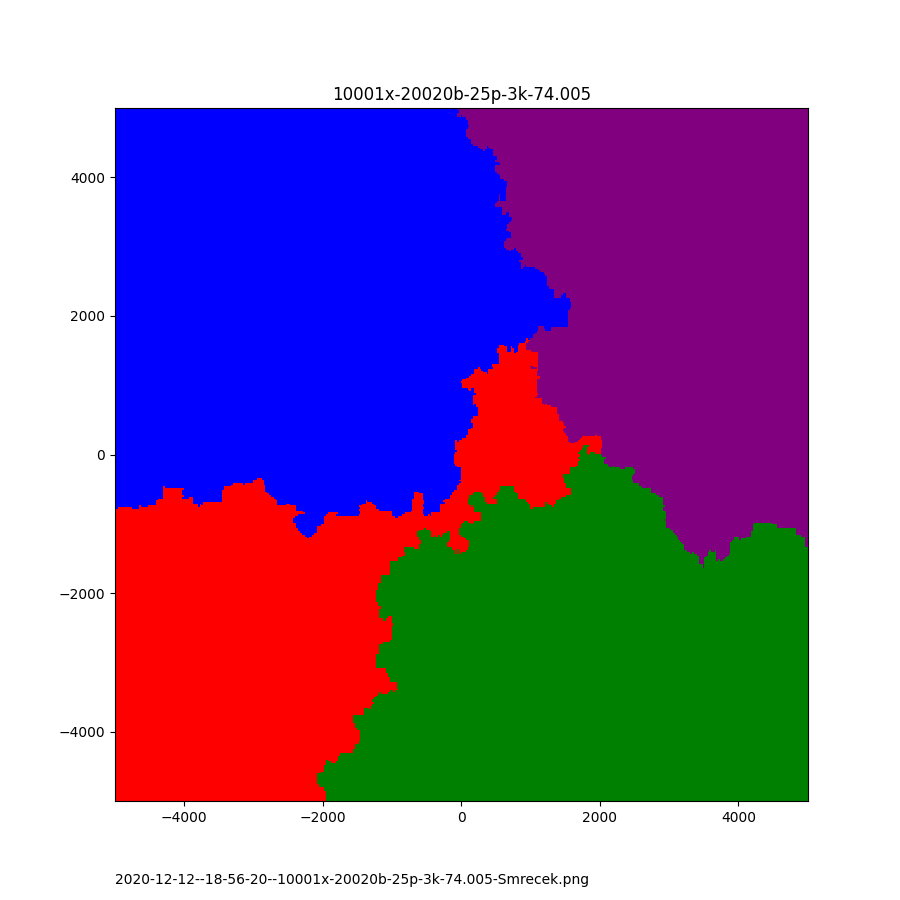


Vizualizácia celej zafarbenej plochy:

## k = 3



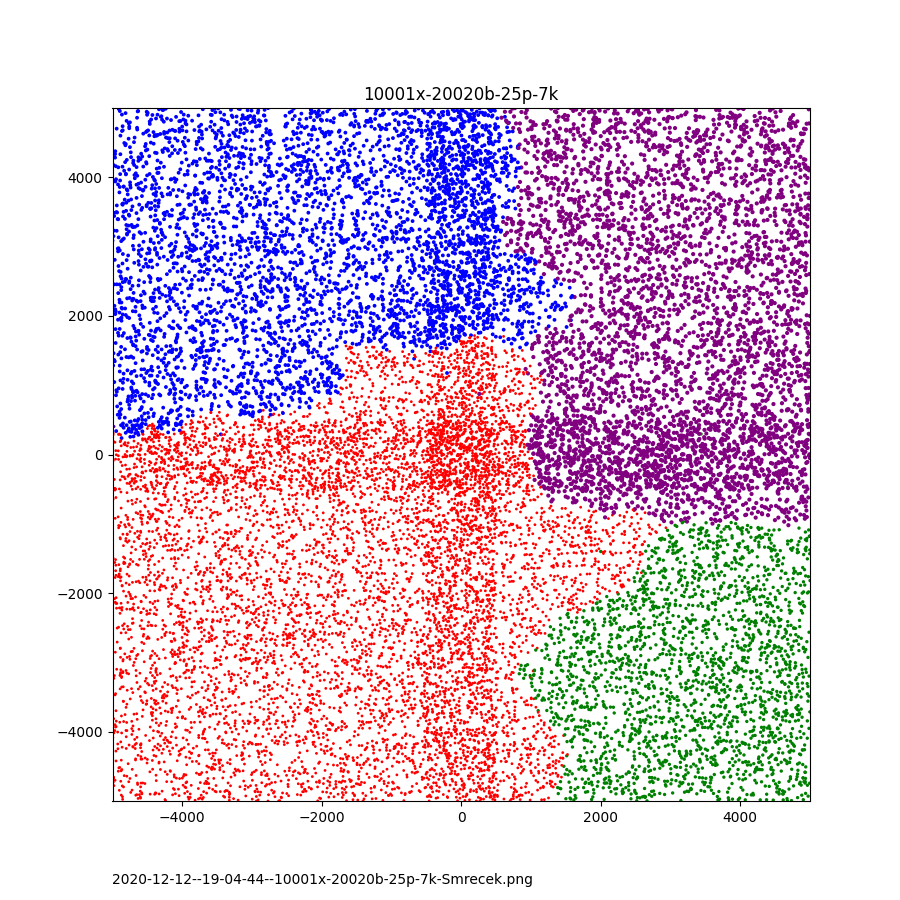
Vizualizácia testovacích bodov:

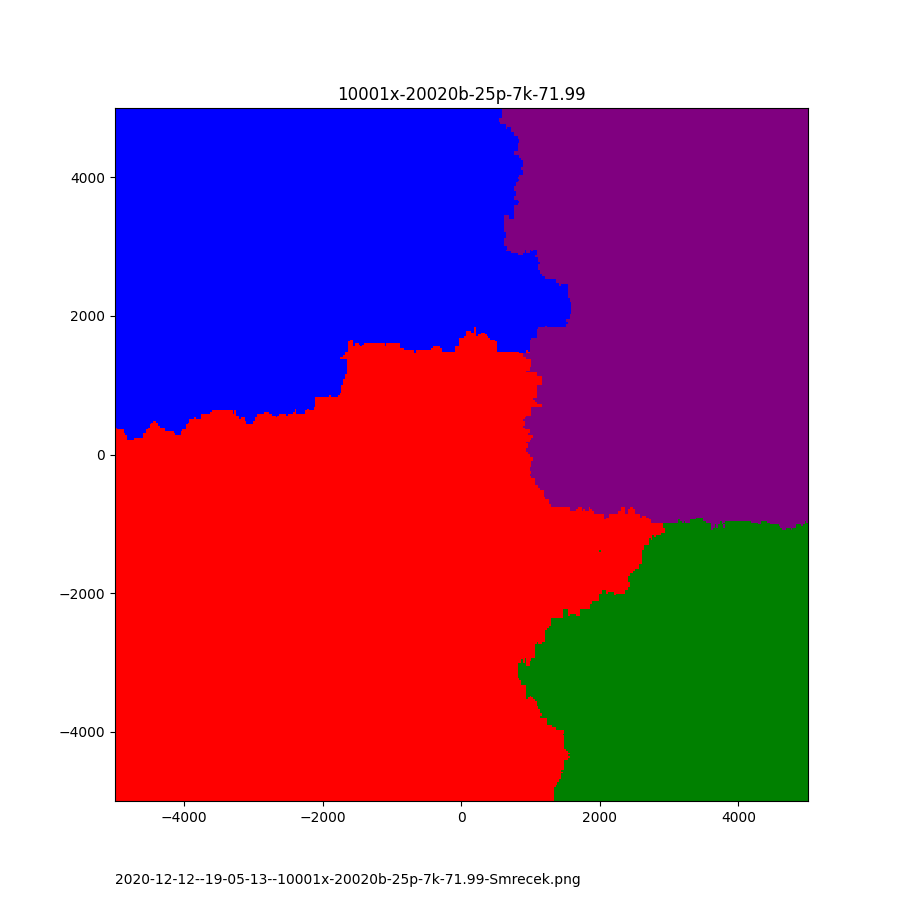


Vizualizácia celej zafarbenej plochy:

## k = 7

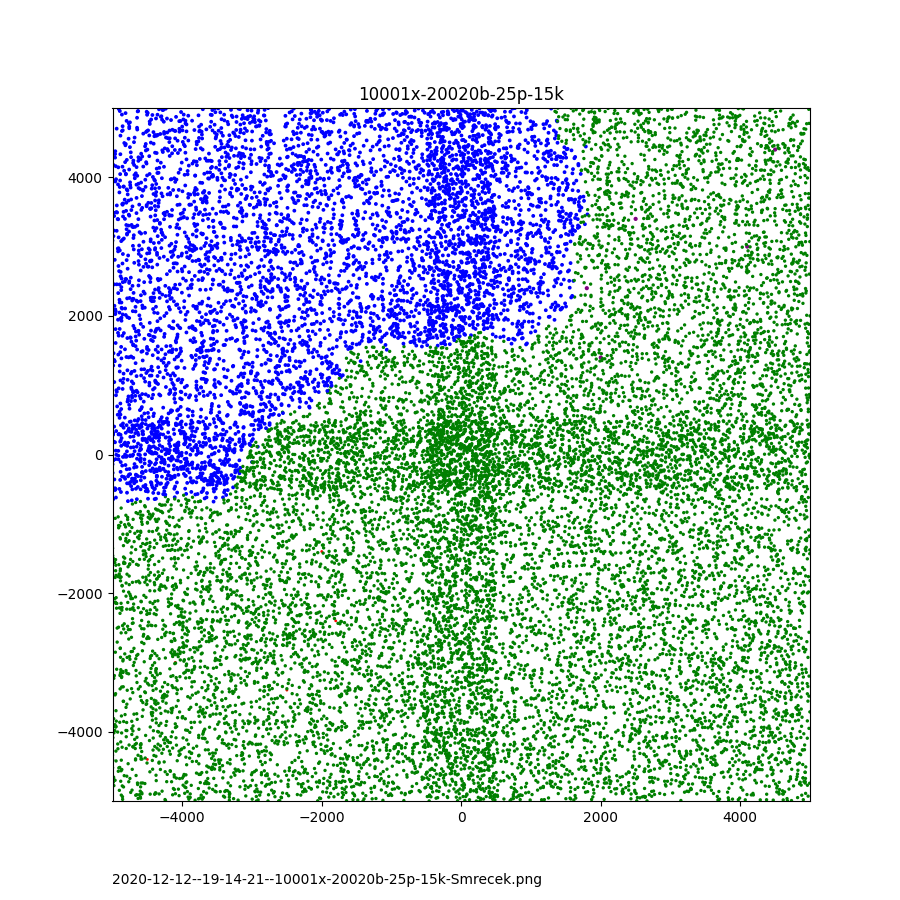
Vizualizácia testovacích bodov:



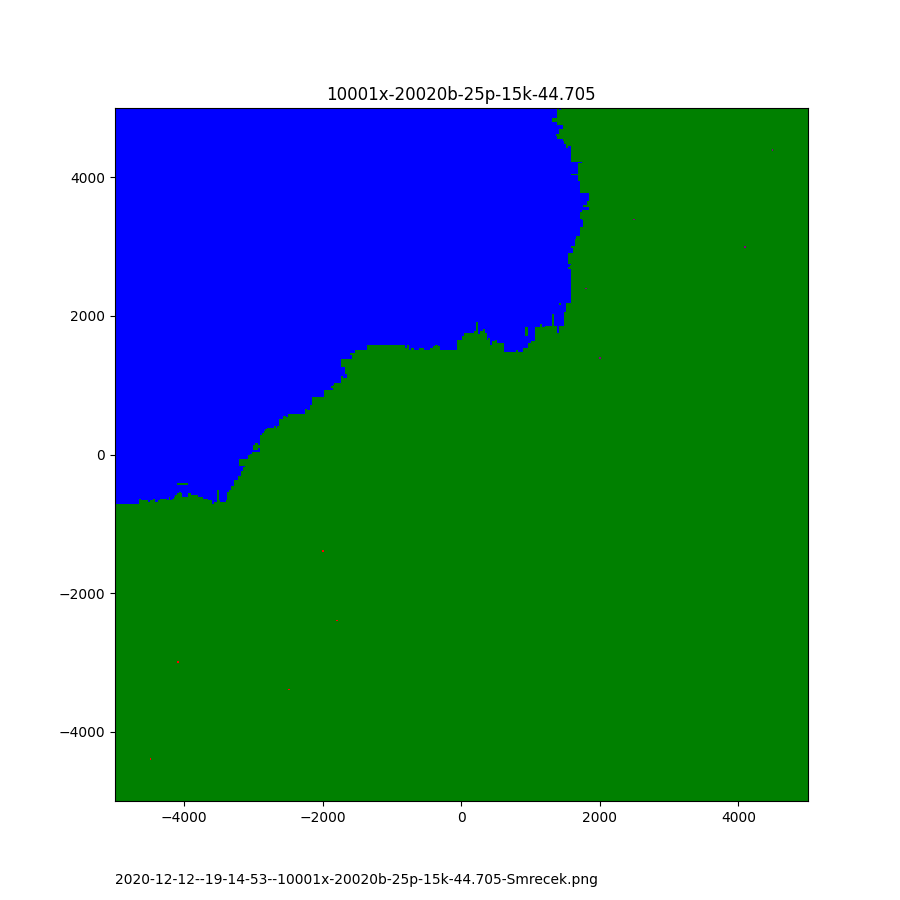
Vizualizácia celej zafarbenej plochy:

## k = 15

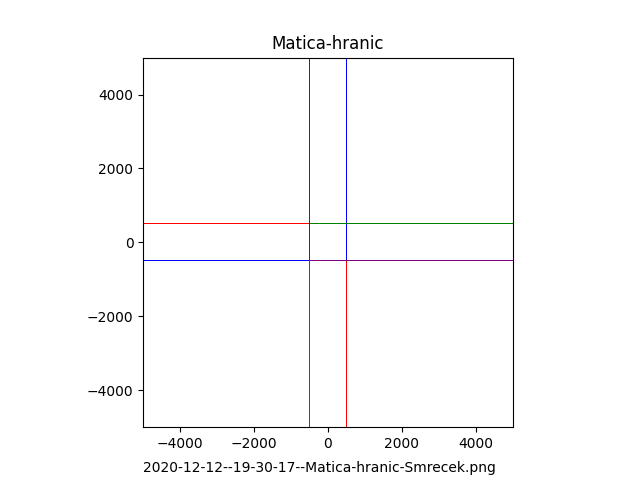
Vizualizácia testovacích bodov:



Vizualizácia celej zafarbenej plochy:



## Zhodnotenie testovania

Na obrázku nižšie sú zobrazené hranice plôch tried dané zadaním.

Testovanie ukázalo, že pri k = 1 mal klasifikátor úspešnosť 69,17%, pri k = 3 bola úspešnosť 74,01%, pri k = 7 bola 71,99% a pri k = 15 bola 44,71%. Percentuálne úspešnosti ukazujú, koľko bodov z 20000 bolo klasifikovaných rovnakou triedou, s akou boli vygenerované.

Môžeme vidieť, že pre k = 1 a pre k = 7 dosahuje klasifikátor podobné úspešnosti. Pri k = 3 dosahuje klasifikátor takmer trojštvrtinovú úspešnosť a pri k = 15 klasifikátor nedosahuje ani polovičnú úspešnosť.

Ako sme videli v prípade k = 1, že fialová výrazne zasiahla do plochy zelenej a červenej, bolo to spôsobené práve jednoprvkovou skupinou susedov, ktorých algoritmus zohľadňoval pri výbere farby nového bodu.

Pri k = 3 sme videli, že výsledky najviac zo všetkých pokusov korešpondovali s plochami tried určenými zadaním. Takýto výsledok sme dosiahli práve správnym zvolením hodnoty k.

Pri k = 7 sa dominantnou stala červená trieda a úspešnosť mierne klesla. Bolo to spôsobené príliš veľkou zvolenou hodnotou k.

Pri k = 15 body červenej a fialovej triedy zmizli takmer úplne. Na vizualizácií celej plochy ich nevidno vôbec. Klasifikátor klasifikoval nové body ako modré a zelené, pričom takmer úplne opomenul fialovú a červenú. Toto bolo spôsobené prehnane vysokou hodnotou k.

Z toho vyplýva, že pri klasifikácií je potrebné zvoliť také k, ktoré nie je ani príliš malé, ale také, ktoré nie je ani príliš veľké. V tomto prípade, z množiny 1, 3, 7, 15, bolo najlepšie možné zvolené k = 3.

# Používateľské rozhranie a ovládanie programu

Po spustení programu si používateľ vyberie jednu z troch možností. Možnosť a pre spustenie testovania, možnosť b pre vykreslenie hraníc do mapy a uloženie takejto matice do súboru, možnosť c pre vizualizáciu matice zo súboru.

Po zvolení možnosti a sú od používateľa pýtané hodnoty s ktorými má program pracovať. Teda k pre kNN algoritmus, počet bodov triedy, každý koľký bod si praje používateľ vizualizovať pri vyfarbovaní celej plochy a či si používateľ želá uložiť takto vygenerovanú maticu do súboru. Ak používateľ zvolí, že si praje maticu uložiť, bude vyzvaný na zadanie cesty k priečinku, kde má byť matica uložená. Následne prebehne program ktorý robí výpisy do konzoly a vygeneruje 2 obrázky, teda vizualizáciu testovacích bodov a vizualizáciu celej zafarbenej plochy. Ak si používateľ prial uložiť takto vygenerovanú maticu, tá sa uloží na zvolenú adresu. Po vykonaní všetkých úkonov program končí. Matice uložené z možnosti a sú uložené kvôli niekoľkominútovému vytváraniu matíc a prípadnému opätovnému spusteniu vizualizácie bodov a vyfarbenia nad tou istou maticou. Táto opätovná vizualizácia nie je v kóde priamo implementovaná ale je možné ju vyskladať z už existujúcich funkcií.

Po zvolení možnosti b je používateľ vyzvaný zadať adresu na uloženie matice a program následne vygeneruje a uloží do súboru maticu s vyfarbeným ohraničením tried.

Po zvolení možnosti c je používateľ vyzvaný zadať cestu k uloženej matici a program následne maticu otvorí a vizualizuje funkciou vizualizuj\_maticu. Táto funkcia je v programe používaná na vizualizáciu celých plôch, preto pre samotnú maticu bodov uloženú z možnosti a nedosiahne tak pekné výsledky. Táto možnosť je primárne určená pre matice získané z možnosti b. Grafy v časti Testovanie boli vygenerované možnosťou a s výnimkou grafu hraníc, ktorý ako jediný bol vygenerovaný možnosťou c.

# Záver

V programovacom jazyku Python som implementoval klasifikáciu bodov v 2D poli. Každá funkcia programu obsahuje svoj vlastný komentár, preto komentáre k funkciám programu v tejto dokumentácií neuvádzam.

Klasifikácia prebiehala na základe kNN algoritmu pre plochu o veľkosti 10001 x 10001. Pôvodných bodov bolo 20, ku ktorým som oklasifikoval a pridal ďalších 20000 bodov. Následne som vizualizoval týchto 20020 vložených bodov na grafe a vizualizoval som aj plochu, ktorú tieto body ohraničujú.

Otestoval som program pre hodnoty k rovné 1, 3, 7 a 15 a zhodnotil výstupy. Najlepšie výsledku dosahoval program pre k = 3.

Z výsledkov testovania vyplýva, že pri klasifikácií je potrebné zvoliť také k, ktoré nie je ani príliš malé, ale také, ktoré nie je ani príliš veľké.