Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Dokumentácia k tímovému projektu

# Inžinierske dielo

Vedúci tímu: Mgr. Martin Sabo, PhD., Ing. Marta Prnová, PhD.
Číslo tímu: 05
Kontakt: mikasa.fiit@gmail.com

# Obsah

# 1 Úvod

V našom projekte sa venujeme automatickému rozpoznávaniu spektier olejov zo vzoriek získaných z AMIS (Advanced Ion Mobility Spectrometer). V tomto dokumente opisujeme priebeh projektu. Definujeme globálne ciele, ale aj ciele pre jednotlivé semestre, celkový pohľad na systém, organizáciu v podobe modifikovaných WBS diagramov a diagramu monolitickej architektúry nášho systému. V krátkosti sú opísané funkcionality, ktoré aplikácia poskytuje.

# 2 Globálne ciele

Hlavným cieľom nášho projektu je vytvoriť systém na automatické rozpoznávanie spektier, ktorý využíva metódy strojového učenia a neurónových sietí na klasifikáciu olivových olejov. Následne chceme tieto výsledky prezentovať v aplikácii.

## 2.1 Ciele zimného semestra

Počas prvého semestra sme sa na systém pozreli z vyššej perspektívy. Oboznámili sme sa s dátami a možnými smermi, akými sme mohli ísť.

Za zimný semester sme pripravili viacero prototypov rôznych klasifikátorov. Dáta sme nie len spracovali do podoby .csv súborov, ale vytvorili sme aj obrázky, ktoré sme následne analyzovali.

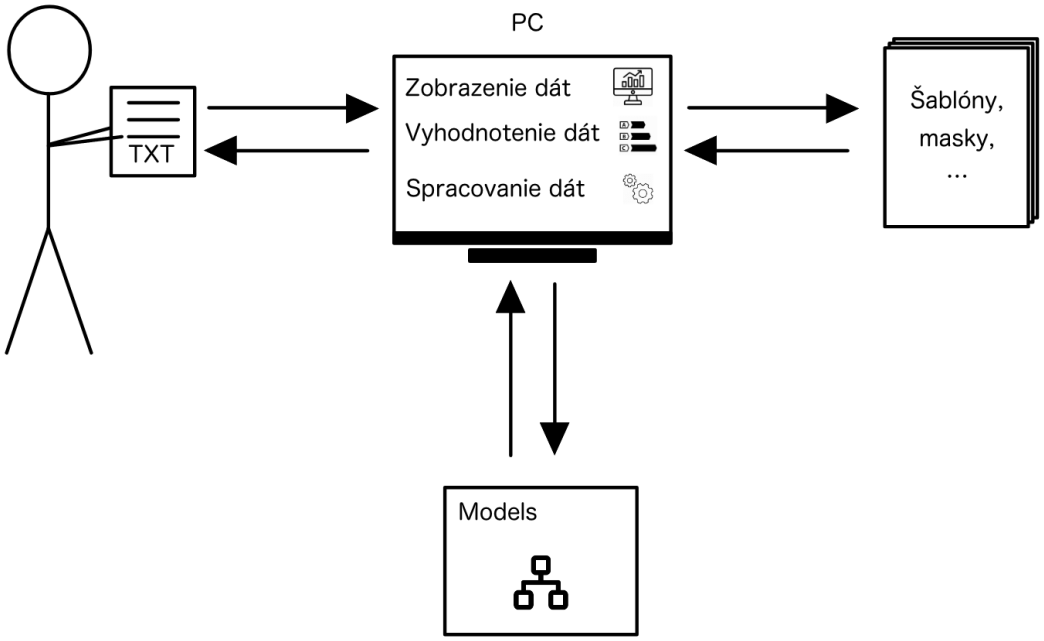
Hlavné ciele, ktoré chceme splniť do konca zimného semestra sú:

- Predspracovanie dát a tvorba čŕt na vylepšenie pôvodných atribútov.
- Vytvorenie prvotných tradičných klasifikátorov (decision tree, knn, naive bayes, svm a random forest).
- Vytvorenie prvotných neurónových sieti (mlp a Vgg16).
- Klasifikácia obrázkov vytvorených z dát.
- Spísanie požiadaviek a návrhu obrazoviek pre aplikáciu.

# 3 Celkový pohľad na systém

Na obrázku 4 vidieť monolitickú architektúru pripravovanej aplikácie. Aplikácia bude mať 3 hlavné funkcionality:

- spracovanie dát
- vyhodnotenie dát
- zobrazenie dát




*Obr. 1: Pohľad na zjednodušenú architektúru*

Vstupom pre aplikáciu sú primárne .txt súbory vygenerované spektrometrom, alebo .csv súbory predspracované aplikáciou.

Hlavnou funkcionalitou aplikácie bude klasifikácia olejov do troch tried EVOO, VOO a LOO. Budú použité tradičné klasifikátory KNN, SVM, Naive Bayes, Random forest, Decision tree a ďalšie. Z neurónových sietí budú využité Vgg16 a MLP. Ďalšími funkcionalitami bude aj vizualizácia jednotlivých olejov zo vstupných dát a príprava dát na klasifikáciu. V rámci prípravy dát na klasifikáciu budú spracované dáta do .csv súborov, budú vytvorené nové črty a nepotrebné dáta budú zredukované.

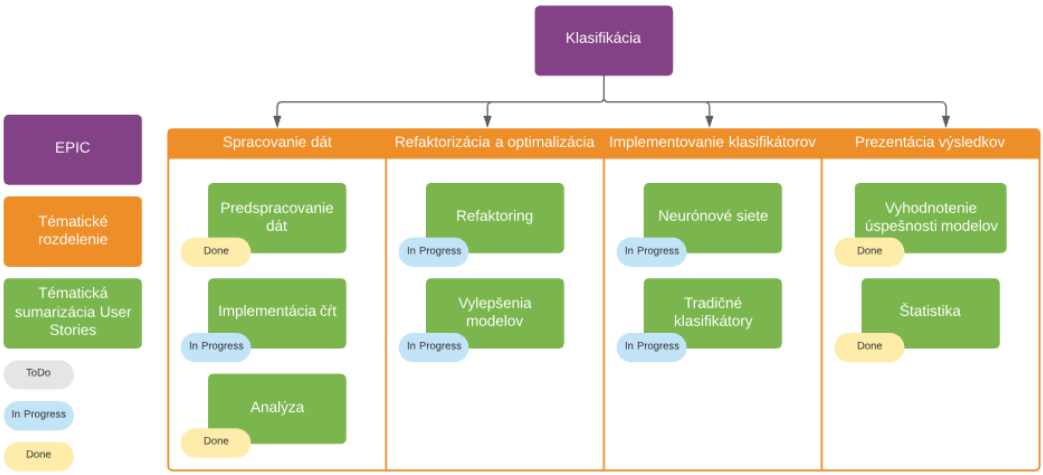
Súčasťou aplikácie budú natrénované modely vyššie spomenutých klasifikátorov, ale aj modely pripravené pre trénovanie na vlastných dátach.

Aplikácia bude obsahovať aj šablóny, masky a iné súbory. Šablóny a masky sú generované pre každú triedu oleja a reprezentujú jej vlastnosti. Šablóna predstavuje očakávané hodnoty a maska zabezpečuje odstránenie variabilných hodnôt v rámci triedy, čiže hodnôt, ktoré sú rôzne v rámci olejov danej triedy. Tieto súbory boli vytvorené na základe existujúcich olejov a sú potrebné pre vypočítanie črty diff.

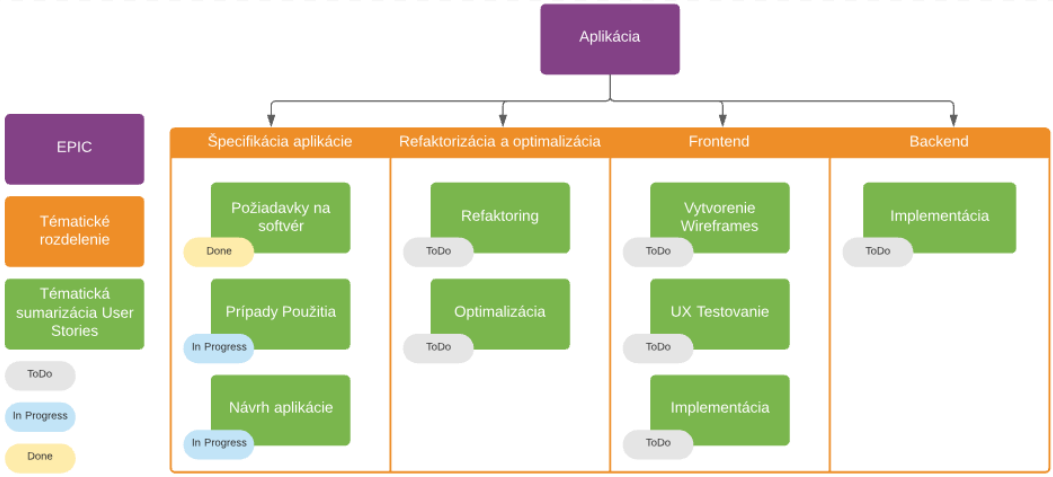
## 3.1 Modifikované WBS diagramy

WBS diagramy znázorňujú menšie komponenty projektu. Názvy komponentov sú na diagrame znázornené fialovou farbou. Tieto komponenty sú v Jire reprezentované ako Epic. Každý Epic má pridelené Story. Tie sme tematicky rozdelili. Tematické rozdelenie je na obrázkoch znázornené oranžovou farbou. Sumarizácia stories je znázornená zelenou farbou.

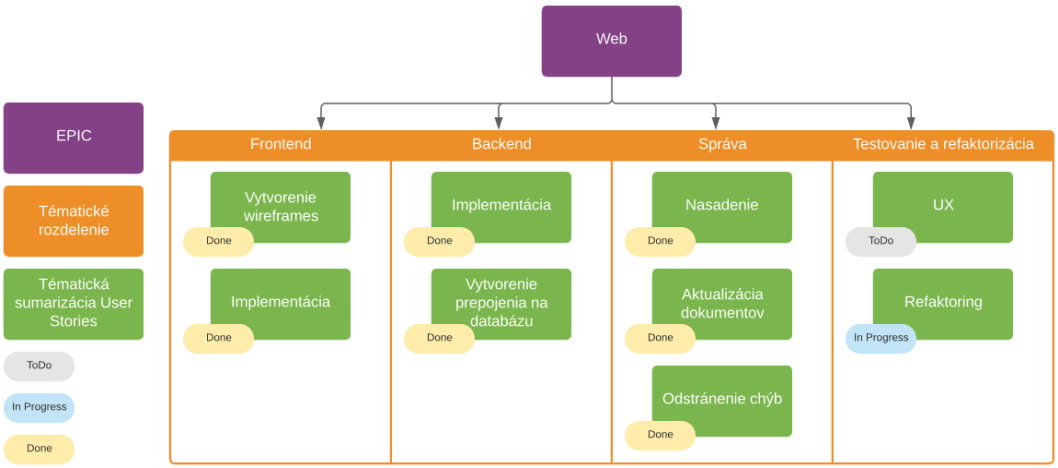
Stavy jednotlivých sumarizácií stories sú v ľavom dolnom rohu. Nezačatá Story je označená ako 'ToDo'. Začatá Story je označená ako 'In Progress' a dokončená ako 'Done'.




*Obr. 2: WBS diagram pre implementáciu klasifikátorov.*




*Obr. 3: WBS diagram pre implementáciu aplikácie.*




*Obr. 4: WBS diagram pre implementáciu webu.*

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Dokumentácia k tímovému projektu

# Riadenie projektu

Vedúci tímu: Mgr. Martin Sabo, PhD., Ing. Marta Prnová, PhD.
Číslo tímu: 05
Kontakt: mikasa.fiit@gmail.com

# Obsah

# 1 Úvod

Tento dokument opisuje postupy pri riadení tímu na projekte Automatické rozpoznávanie spektier.

V nasledujúcich kapitolách sú opísané role jednotlivých členov tímu a ich manažérska pozícia, z ktorej vyplývajú ich zodpovednosti na projekte, prehľad podielu práce na dokumentácii, opis spôsobov aplikácie manažmentov a zavádzania postupov v tíme, pokrok tímu na projekte počas jednotlivých šprintov a evidencia dodaných používateľských scenárov a globálna retrospektíva.

# 2 Role členov a podiel práce na dokumentácii

## 2.1 Role členov tímu

| Meno a Priezvisko | Manažér | Rola |
|---|---|---|
| Simona Klučková | Manažér dátovej analýzy | Data scientist |
| Maroš Kollár | Manažér vývoja pre neurónové siete | Data scientist |
| Jakub Kučečka | Manažér kvality kódu | Developer |
| Zuzana Popovcová | Manažér úloh | Software architect |
| Mária Rajníková | Manažér vývoja klasifikačných metód | Data scientist |
| Alena Valová | Manažér dokumentácie | Scrum master |

## 2.2 Podiel práce na dokumentácii

| Meno a Priezvisko | Percentuálny podiel |
|---|---:|
| Simona Klučková | 16% |
| Maroš Kollár | 17% |
| Jakub Kučečka | 16% |
| Zuzana Popovcová | 17% |
| Mária Rajníková | 16% |
| Alena Valová | 18% |

# 3 Aplikácie manažmentov

Pre aplikáciu manažmentov v tíme využívame metodológie. Ide o dokument, v ktorom si definujeme pravidlá a postupy a procesy v tíme. Zoznam metodík nájdete na stránke tímu.

Metodiky sú priebežne aktualizované a dopĺňané. V každej metodike sa preto nachádza história verzií, ktorá nesie informácie o vykonaných zmenách v dokumente.

## 3.1 Manažment komunikácie

Na komunikáciu v tíme využívame viacero komunikačných nástrojov. Hlavným nástrojom je Microsoft Teams. Tento nástroj je využívaný na komunikáciu členov tímu a ich stretnutia.

Najvyužívanejším kanálom Microsoft Teams je kanál *Stand-up*, ktorý používame namiesto daily stand-up.

Na komunikáciu s vedúcimi projektu (ďalej len PO) využívame viacero komunikačných nástrojov. Primárne však využívame Google Meet.

Všetky komunikačné nástroje a ich kanály, ktoré v tíme využívame nájdete v prílohe C.

## 3.2 Manažment vývoja

Hlavným programovacím jazykom pre vývoj projektu je jazyk Python. Odporúčaným vývojovým prostredím pre členov tímu je Pycharm.

Pre tvorbu kvalitného a dobre štrukturovaného kódu, sme si zadefinovali pravidlá vývoja v Metodike vývoja v Pythone. V rámci metodiky sme určili štruktúru pracovných adresárov projektu a základné pravidlá pre tvorbu jednotného a dobre štrukturovaného kódu.

Podobne popísané pravidlá vývoja v jazyku Python sú popísané v dokumente Metodika vývoja v Python-e. Tento dokument nájdete v prílohe D.

## 3.3 Manažment testovania

Dôležitou súčasťou vývoja je testovanie. Napomáha k doručeniu validných a nezávadných častí softvéru. Pre validáciu dodaných riešení sú využívané akceptačné testy a pre odhalenie chýb sú vytvárané jednotkové testy. Jednotkové testy budú vykonávané pomocou knižnice *pytest*.

Presnejšie zadefinované pravidlá použitia týchto testov a ich správa v projekte je zadefinovaná v dokumente Metodika testovania. Metodiku nájdete v prílohe E

## 3.4 Manažment splnenia úlohy

Pre zadefinovanie pravidiel, kedy je činnosť na úlohe dokončená sme vytvorili dokument Metodika splnenej úlohy, ktorý nájdete v prílohe F. V metodike sme si zadefinovali pravidlá, kedy bude úloha, používateľský príbeh a projekt považovaný za úplne dokončený.

## 3.5 Manažment práce s úlohami

Hlavným nástrojom pre manažment úloh na projekte je Jira. Tento nástroj využívame na plánovanie práce na jednotlivých šprintoch. Vytvárame v ňom úlohy, ktoré je potrebné realizovať na projekte. Všetky úlohy sú vytvárané v backlogu. Následne na spoločnom stretnutí ohodnocujeme úlohy bodmi a zaraďujeme ich podľa priority do šprintu.

Pre bližšie zadefinovanie stavov úloh, ich manipuláciu a spôsobu vytvárania bola vytvorená Metodika práce s úlohami. Celú metodiku nájdete v prílohe G.

## 3.6 Manažment verziovania programu

Pre umožnenie vytvárania verzii programu a paralelného pracovania na projekte používame GitHub.

Všetky pravidlá pre manipuláciu s týmto nástrojom, pomenovanie vetiev a potvrdzujúcich správ sú bližšie definované v dokumente Metodika verziovania programu. Celú metodiku nájdete v prílohe H.

# 4 Sumarizácie šprintov

## 4.1 Šprint č.1

| **Názov šprintu** | Starohutský vodopád |
|---|---|
| **Trvanie** | 14.10.2020 - 23.10.2020 |

Prvý šprint bol zameraný na zavádzanie procesov v tíme, vytvorenie reprezentačného webu a na urgenciu vedúceho (ďalej PO) bola do šprintov zaradená aj práca na klasifikátore. User stories zahrnuté do šprintu boli ohodnotené na 67 story points (ďalej SP).

Spoločne sme v šprinte zaviedli procesy prostredníctvom napísania metodík. Pravidlá v napísaných metodikách sa podarilo počas šprintu rýchlo integrovať s výnimkou metodiky vývoja v Pythone a metodiky verziovania programu. Integrácia pravidiel týchto metodík bola podmienená ukončením všetkých implementačných úloh. Počas šprintu sa podarilo úspešne vytvoriť a odovzdať prihlášku na TP-cup.

V rámci práce na projekte bolo do šprintu zaradené spracovanie dát a vytvorenie prvotného klasifikátora. Všetky úlohy na projekte boli zvládnuté výborne. PO bol s výsledkami prvotného klasifikátora spokojný a nemal žiadne výhrady.

Najviac SP bolo pridelených v šprinte na vytvorenie webovej stránky. Stránka bola úspešne vytvorená a nasadená.

Najväčší podiel bodov v šprinte získal Jakub Kučečka, ktorý vytváral webovú stránku a odviedol pri tom skvelú prácu.

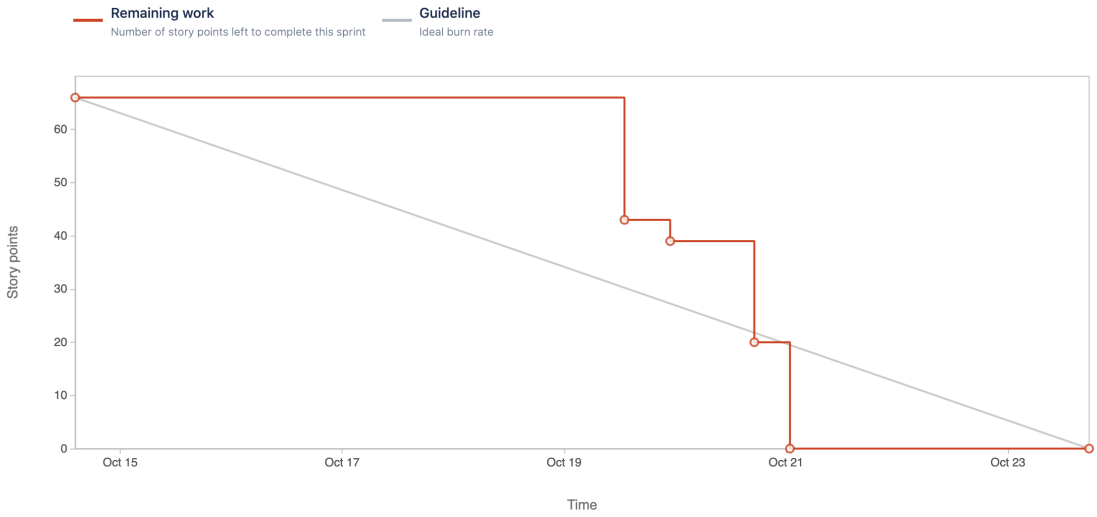
### User stories

| **Názov** | **SP** | **Schválená** |
|---|---|---|
| Ako produkt owner potrebujem rozoznať jednotlivé druhy olejov pomocou tradičných klasifikátorov | 21 | Áno |
| Ako tím, potrebujeme vytvoriť frontend pre prezentáciu tímu | 23 | Áno |
| Ako tím potrebujeme metodiky pre jasné definovanie pravidiel v tíme aby sme zefektívnili a zkvalitnili našu prácu | 19 | Áno |
| Ako product owner chcem aby sa tím zúčastnil súťaže TP cup pre reprezentáciu výsledkov projektu | 4 | Áno |

### Práca na projekte

| Meno a Priezvisko | Počet úloh | Percentuálny podiel na šprinte |
|---|---|---|
| Simona Klučková | 4 | 15% |
| Maroš Kollár | 5 | 15% |
| Jakub Kučečka | 1 | 23% |
| Zuzana Popovcová | 3 | 17% |
| Mária Rajníková | 3 | 15% |
| Alena Valová | 4 | 15% |

### Burndown chart




*Obr. 1: Burndown chart*

### Export úloh

Dokument s exportom úloh 1. šprintu.

## 4.2 Šprint č.2

| | |
|---|---|
| **Názov šprintu** | Vodopád Skok |
| **Trvanie** | 23.10.2020 - 6.11.2020 |

Šprint bol zameraný na vylepšovanie a rozširovanie počtu klasifikátorov, vytváranie obrazových dát a implementovanie administratívneho rozhrania pre webovú stránku. User stories zahrnuté do šprintu boli ohodnotené na 60 story points (ďalej SP).

Počas šprintu boli vytvorené 4 nové modely tradičných klasifikátorov a to Naive bayes, decision tree, KNN a SVM. Vytvoril sa prvý model neurónovej siete a návrh konvolučnej siete. Taktiež sme v šprinte pokračovali v implementácií nových čŕt. V šprinte sa podarilo vytvoriť aj obrazovú reprezentáciu dát. Všetky úlohy súvisiace s klasifikáciou na projekte boli zvládnuté perfektne.

K reprezentačnej stránke tímu bolo úspešne vytvorené a nasadené admin rozhranie, ktoré umožňuje jednoduché pridávanie nových dokumentov na webovú stránku.

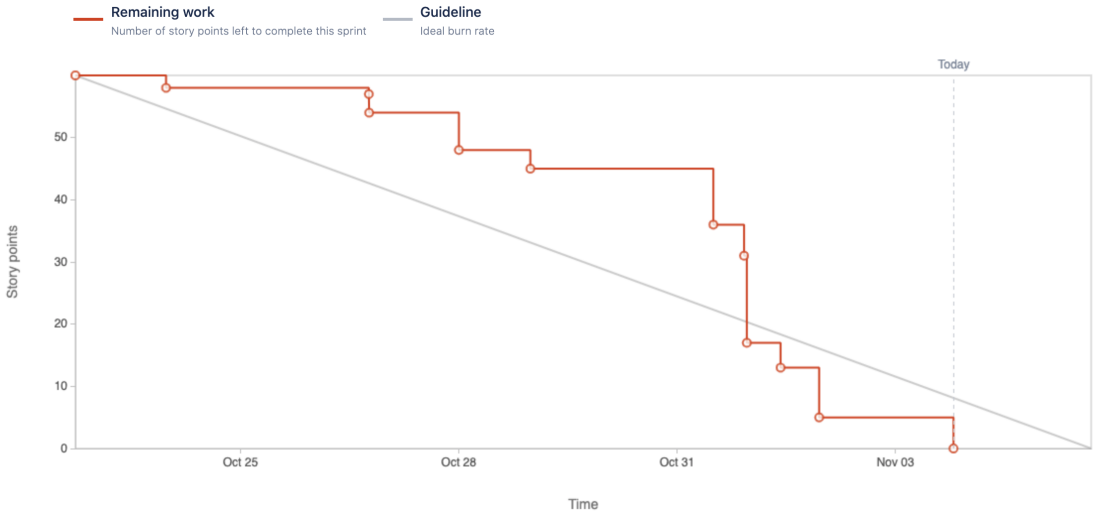
**User stories**

| **Názov** | **SP** | **Schválená** |
|---|---|---|
| Ako tím potrebujeme vytvoriť správy o riešení projektu pre úspešné absolvovanie predmetu | 3 | Áno |
| Ako tím potrebujeme zrefaktorovať kód z predošlých šprintov | 3 | Áno |
| Ako tím potrebujeme v 2.týždni šprintu zabezpečiť prípravu a dokumentáciu stretnutí | 5 | Áno |
| Ako product owner potrebujem implementovať neurónové siete | 8 | Áno |
| Ako product owner potrebujem implementovať tradičné klasifikátory | 4 | Áno |
| Ako product owner potrebujem dosiahnuť najlepšie výsledky pri klasifikácií, preto potrebujem rozšíriť zoznam čŕt | 14 | Áno |
| Ako tím potrebujeme v 1.týždni šprintu zabezpečiť prípravu a dokumentáciu stretnutí | 6 | Áno |
| Ako správca webovej domény potrebujem implementovať backend webu | 9 | Áno |

## Práca na projekte

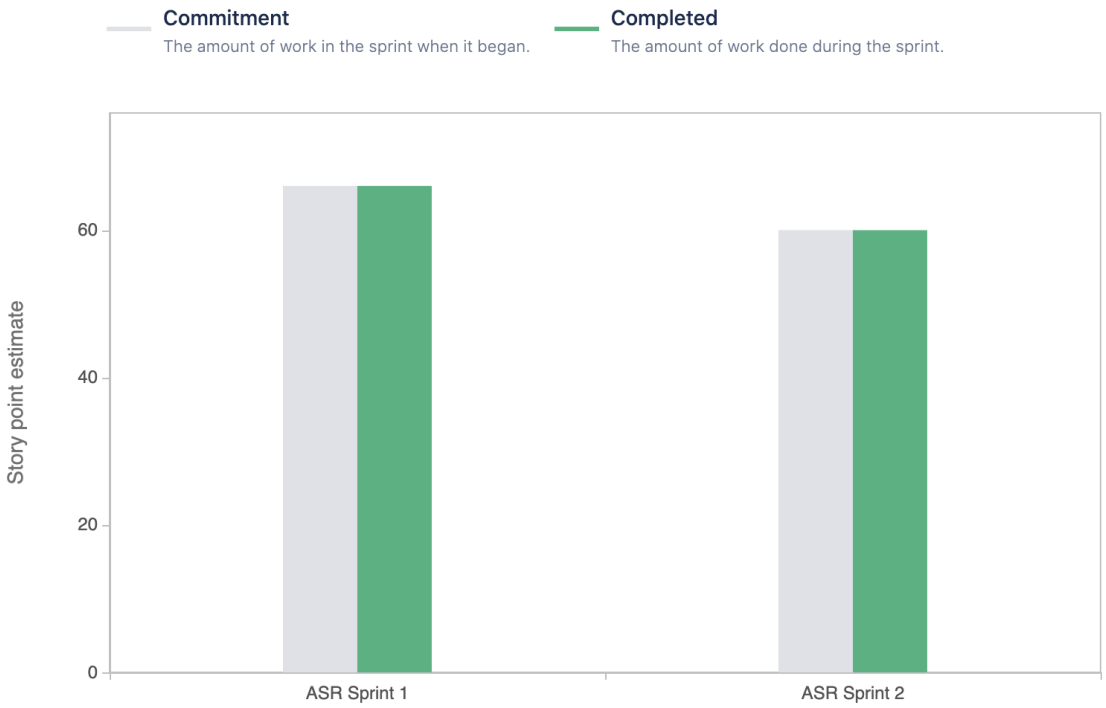
| Meno a Priezvisko | Počet úloh | Percentuálny podiel na šprinte |
|---|---|---|
| Simona Klučková | 6 | 17% |
| Maroš Kollár | 2 | 17% |
| Jakub Kucecka | 4 | 18% |
| Zuzana Popovcová | 3 | 17% |
| Mária Rajníková | 3 | 15% |
| Alena Valová | 6 | 17% |

## Burndown chart




*Obr. 2: Burndown chart*

### Velocity report




*Obr. 3: Velocity report*

### Export úloh

Dokument s exportom úloh 2. šprintu.

## 4.3 Šprint č.3

| **Názov šprintu** | Tomašovský vodopád |
|---|---|
| **Trvanie** | 6.11.2020 - 20.11.2020 |

Šprint bol zameraný na vytváranie dokumentácie k projektu a spisovanie požiadaviek na aplikáciu. Pokračovali sme vo vytváraní čŕt a ďalších klasifikátorov. User stories zahrnuté do šprintu boli ohodnotené na 64 story points (ďalej SP).

Počas šprintu boli vytvorené 3 dokumenty.

- **Dokumentácia k riadeniu** - ”big picture”riadenia projektu. Obsahom dokumentu je prevažne prehľad rolí členov v tíme a prehľad manažmentov.
- **Dokumentácia modulov systému** - Obsahom dokumentu je popis analýzy, návrhu, implementácie a testovania.
- **Inžinierske dielo** - ”big picture”cieľov semestra. Obsahom dokumentu sú globálne ciele semestra a celkový pohľad na systém.

V šprinte sme začali definovať požiadavky na systém a boli vytvorené a spísané hlavné prípady použitia aplikácie.

Pri klasifikácii sme pokračovali vo vytváraní nových čŕt, ako napríklad diferenčné mapy, median, std a ďalšie. Bola vytvorená a otestovaná prvá konvolučná sieť. Do šprintu sme zahrnuli aj analytické úlohy nad dátami a úlohu o vyhľadávaní existujúcich riešení pre klasifikáciu látok.

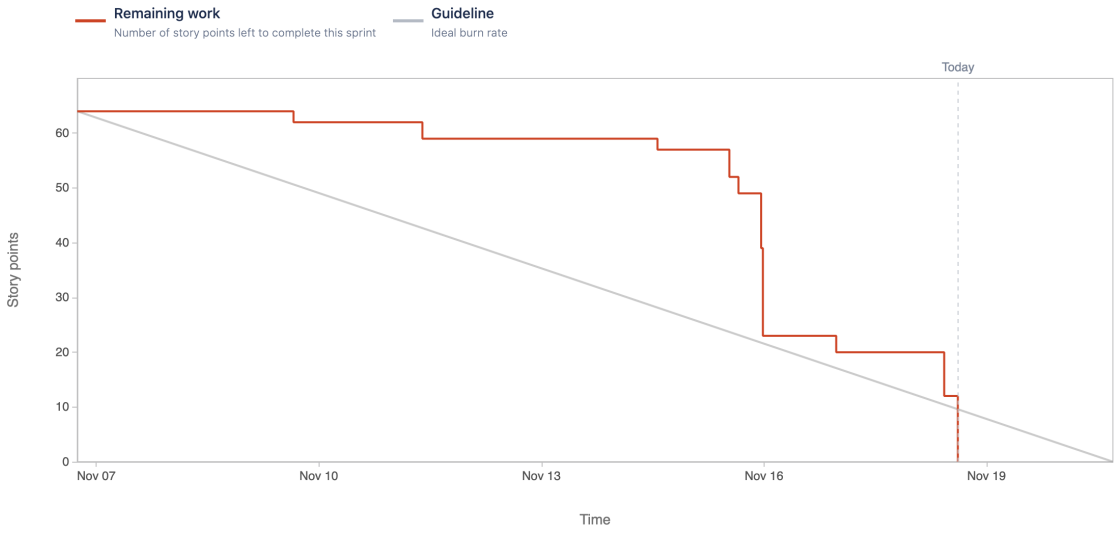
**User stories**

| **Názov** | **SP** | **Schválená** |
|---|---|---|
| Črty | 5 | Áno |
| Analýza dát | 8 | Áno |
| Konvolučná neurónová sieť | 3 | Áno |
| Predspracovanie dát | 7 | Áno |
| Diferenčná mapa | 10 | Áno |
| Vytvorenie dokumentu Inžinierske dielo | 3 | Áno |
| Vytvorenie dokumentu Modulov systému | 8 | Áno |
| Vytvorenie dokumentu Riadenie projektu | 2 | Áno |
| Zabezpečenie prípravy dokumentácie a stretnutí - 2. týždeň | 5 | Áno |
| Zabezpečenie prípravy dokumentácie a stretnutí - 1. týždeň | 3 | Áno |
| Špecifikácia aplikácie | 8 | Áno |

## Práca na projekte

| Meno a Priezvisko | Počet úloh | Percentuálny podiel na šprinte |
|---|---|---|
| Simona Klučková | 4 | 19% |
| Maroš Kollár | 4 | 17% |
| Jakub Kucecka | 4 | 12% |
| Zuzana Popovcová | 4 | 17% |
| Mária Rajníková | 4 | 19% |
| Alena Valová | 7 | 16% |

## Burndown chart




*Obr. 4: Burndown chart*

## Velocity report


*Obr. 5: Velocity report*

## Export úloh

Dokument s exportom úloh 3. šprintu.

# 5 Globálna retrospektíva

## 5.1 Retrospektíva šprintu č.1

Retrospektívy sa zúčastnili všetci členovia tímu.

| Meno a Priezvisko | Zúčastnil/a sa na stretnutí |
|---|---|
| Simona Klučková | Áno |
| Maroš Kollár | Áno |
| Jakub Kučečka | Áno |
| Zuzana Popovcová | Áno |
| Mária Rajníková | Áno |
| Alena Valová | Áno |

**Čo sme spravili dobre?**

- Začali sme s klasifikáciou pomerne skoro.
- Dokončili sme písanie metodík.
- Dali sme do poriadku git.
- Dobre sme zvládli komunikáciu v tíme.
- Super komunikácia s PO.
- Hotový web po prvom šprinte.
- Odovzdaná prihláška na TP cup.
- Podarilo sa nakonfigurovať server.

**Čo sme mohli urobiť lepšie?**

- Písanie úloh. Trvalo nám to moc dlho.
- Vkladanie odkazov na dokončený výstup práce, keď ju posúvame na review.
- Zaviesť stand-up, aby sme vedeli na čom, kedy a kto robí.
- Viac plánovať implementáciu a menej riešiť úlohy.
- Častejšie riešiť na stretnutiach implementačné detaily.
- Častejšie zapájať vedúcich do vývoja.

**Čo ideme zaviesť?**

- Zmeniť pomenovanie úloh zadaných používateľom. Aktuálny nadpis pôjde do opisu a nadpis bude podobný ako pri ostatných úlohách.
- Lepšia príprava na stretnutia, na ktorých sa plánuje šprint. Vopred si pripraviť úlohy, ktoré úlohy chcem do šprintu zaradiť.
- Vkladanie odkazov do úloh, ktoré sme dokončili.
- Zaviesť stand-up prostredníctvom komunikačného kanála vytvoreného na Microsoft teams.
- Každú 2. stredu, aspoň jeden krát za šprint sa budú na stretnutí riešiť spôsoby implementácie.
- Dohodnúť si vedúcimi stretnutie a spísať s nimi plán na 2 semestre.
- Zahrnúť do úloh prípravu prezentácii.

## 5.2 Retrospektíva šprintu č.2

Retrospektívy sa zúčastnili všetci členovia tímu.

| Meno a Priezvisko | Zúčastnil/a sa na stretnutí |
|---|---|
| Simona Klučková | Áno |
| Maroš Kollár | Áno |
| Jakub Kučečka | Áno |
| Zuzana Popovcová | Áno |
| Mária Rajníková | Áno |
| Alena Valová | Áno |

**Čo sme spravili dobre?**

- Zaviedli sme Standup a RRF.
- Zavedenie retrospektívy pomocou Trella.
- Plánovanie šprintov.
- Splnili sme všetky úlohy v dohodnutom čase.
- Zapracovali sme všetky pripomienky z minulej retrospektívy.
- Zapojenie vedúcich do tvorby príbehov.

- Konzultovanie obsahu dokumentácie.
- Zaviedli sme implementačné stretnutia.

**Čo sme mohli urobiť lepšie?**

- Viac SP pre tasky s testovaním.
- Viac farebných grafov.
- Nezabúdať na písanie standup.
- Prestať robiť okamžité implementácie. Viac analýzy.
- Menej dlhých a nezáživných stretnutí.
- Menej dokumentácie.

**Čo ideme zaviesť?**

- Začať označovať ľudí v skupine.
- Veci týkajúce malej skupiny ľudí riešiť v osobnom chate.
- Začať viac analyzovať.
- Zohľadňovať pri planning pokri úlohy, v ktorých sa programuje.
- Pomenovávanie šprintov podľa vodopádov. A pridanie obrázkov na web.
- Pridávanie nálepiek k nápadom, ktoré sa nám v trelle páčia a nepáčia.
- Do názvu kartičky v trelle napísať na začiatok svoje iniciály.

## 5.3 Retrospektíva šprintu č.3

Retrospektívy sa zúčastnili všetci členovia tímu.

| **Meno a Priezvisko** | **Zúčastnil/a sa na stretnutí** |
|---|---|
| Simona Klučková | Áno |
| Maroš Kollár | Áno |
| Jakub Kučečka | Áno |
| Zuzana Popovcová | Áno |
| Mária Rajníková | Áno |
| Alena Valová | Áno |

**Čo sme spravili dobre?**

- Máme progress na klasifikácii.
- Prezentácia výsledkov na sprint review.
- Práca pod stresom.
- Samostatné riešenie problémov na úlohách.
- Výborná komunikácia pri riešení problémov.
- Zvládnutie formátovania UC do latex.

**Čo sme mohli urobiť lepšie?**

- Potrebujeme viac ľudí na review.
- Tvorba backlogu.
- Lepšie plánovanie backlogu.
- Lepšie písanie opisu.
- Pre-planning a planning rozdeliť do dvoch stretnutí.
- Pridanie viac odkazov k úlohe (tuto to nájdeš...), s lepším vysvetlením.
- Lepší systém pre hodnotenie úloh.
- Nepísať správy v noci.
- Pri udeľovaní úloh rátať s rizikom.

**Čo ideme zaviesť?**

- Prideľovať k úlohám jednotlivo Akceptora.
- Pridať stĺpec Ready to Approve.
- Určiť ľudí zodpovedných za backlog epic.
  - Klasifikácia - Majka
  - Aplikácia - Jakub
  - Administrácia - Ala
- Nočný režim 23:00 - 10:00

# A Motivačný list

## A.1 Predstavenie tímu

Sme mladý a flexibilný tím pripravený prekonať každú výzvu. Voláme sa Mikasa po postave z anime a po značke výrobcu volejbalových lôpt.

Každý člen tímu sa užšie špecializuje na niečo iné, ale naše predstavy o tom, čomu sa plánujeme venovať sú približne rovnaké. Zaujímajú nás neurónové siete, dátová analýza a vývoj aplikácií a softvérov. Zoznam technológií, ktoré ovládame.

Spoločným cieľom je vytvárať produkty s vysokou kvalitou. Základom vysokej kvality je:

- analýza - každý člen tímu vypracoval už minimálne 3 analytické dokumenty.
- návrh - počas štúdia sme vytvorili rôzne návrhy softvérov v rôznych notáciach.
- program - záleží nám na kvalite výsledného produktu, a fungujúci program je jeho základom. Každý jeden z nás už napísal tisíce riadkov kódu v minimálne 4 rôznych jazykoch.
- praktickosť uplatniteľniteľnosť vo svete - všetky programy sú tvorené pre ľudí, a preto nám záleží na prínose do ich životov.

Vyššie uvedené body kvality vieme zabezpečiť aj vďaka:

- Sime, ktorá sa zaujíma o dátovú analýzu, strojové učenie a virtuálnu realitu. V rámci svojej bakalárskej práce získala cenu dekana za ohodnocovanie biometrických charakteristík na základe gest rukami, kde vytvorila vlastnú hru, pomocou ktorej získala dáta od používateľov a tie následne spracovala. Použila machine learning na zisťovanie možnosti identifikácie používateľa na základe gest rukami a faktu, či je možné tieto behaviorálne črty napodobniť a "ukradnúť".
- Marošovi, ktorý sa zaujíma o neurónové siete a na väčšine školských projektov a aj v rámci brigády programoval backend. Vo svojej bakalárskej práci na základe vizuálnych charakteristík určoval komplementárne páry produktov pomocou siamskej neurónovej siete pre ich možné využitie v rámci personalizovaného odporúčania.
- Jakubovi, ktorý sa zaujíma o tvorbu webových stránok a aplikácií. Taktiež je flexibilný a vie sa chytiť modelovania. Vo svojej bakalárskej práci Rekurentné

vzťahy v grafických objektoch, vysvetľoval fungovanie rekurentných vzťahov pre postupnosti triviálnym spôsobom na rekurentných krivkách. Tiež má skúsenosti z práce so správou core-backendu a scriptovacími jazykmi.

- Zuzke, ktorú najviac zo všetkého baví modelovanie softvéru. Je flexibilná a v tíme pomôže, kde sa dá. Má skúsenosti s tvorením webov a s neurónovými sieťami. V rámci bakalárskej práce sa Zuzka venovala spracovaním medicínskych dát pomocou konvolučných neurónových sietí.
- Majke, ktorá sa zaujíma o dátovú analýzu a strojové učenie, taktiež bola šikovnou výskumníčkou. V rámci svojej bakalárskej práce získala cenu dekana za vytvorenie modelu, ktorý dokáže rozoznať ukradnutie mobilného zariadenia od bežného používania. Využila pritom dáta zo senzorov zaznamenané vlastnou mobilnou aplikáciou.
- Ale, ktorá sa zaujíma tvorbu webových stránok, modelovanie softvéru a má skúsenosti aj s tvorbou hier. V jej bakalárskej práci Meranie a navrhovanie algoritmických zručností, pomocou jednoduchej hry zisťovala, ktoré metriky vplývajú na výpočtové myslenie. Z hry získavala dáta a následne ich spracovala.

## A.2 Motivácia: Automatické rozpoznávanie spektier (08)

Pri skúmaní súčasného informatického sveta a spôsobu, akým sa v posledných rokoch uberajú spoločnosti si v rámci nášho tímu myslíme, že spracovanie dát a strojové učenie sú oblasti informatiky, ktoré budú postupne ešte viac žiadané ako je tomu v súčasnosti.

Mnohí členovia nášho tímu by tak radi využili nadobudnuté vedomosti z predmetu Inteligentná Analýza Údajov alebo bakalárskych prác. Všetci by sme však pomocou práce na tomto projekte svoje vedomosti radi prehĺbili, prípadne sa naučili niečo úplne nové.

Téma Automatického rozpoznávania spektier nás zaujala najmä vďaka tomu, že sa primárne jedná o výskumne orientovanú tému. Súčasne však téma vytvára priestor spojiť výskum s praxou, na čom by sme sa radi podieľali.

Veríme, že aplikáciu umožňujúcu automatické rozpoznávanie spektier by prax uvítala, či už v rámci výrobných procesov alebo v rámci dodatočných špecializovaných rozborov.

Tému tímového projektu tak berieme ako možnosť získania nových vedomostí z oblasti dátovej vedy či skúseností z práce v tíme venujúcemu sa dátam a strojovému učeniu. Možnosť podieľať sa na tvorbe systému, ktorý by odborníkom ušetril množstvo času by však taktiež uspokojila našu chuť zjednodušovať prácu ľuďom z iných oblastí a vytvorilo skúsenosti pre nasledujúci profesný život.

Dúfame, že naše doterajšie skúsenosti so spracovaním dát a strojovým učením v kombinácií s našim zanietením pre túto oblasť informatiky z nás robia vhodných kandidátov pre prácu na tímovom projekte Automatického rozpoznávania spektier.

## A.3 Motivácia: Podporný informačný systém pre študijné oddelenie (19)

Téma Podporný informačný systém pre študijné oddelenie náš tím zaujala najmä z dôvodu, že sa jedná o komplexnú webovú aplikáciu. Ako sme už spomínali, náš tím má bohaté skúsenosti s vývojom webových aplikácií ako aj návrhom softvéru. Pri práci na tejto aplikácií by sme radi využili naše vedomosti z predmetu Webové technológie, ktorý absolvovali 3 členovia nášho tímu a tiež z predmetu Webové publikovanie. Taktiež sa môžeme spoľahnúť aj na Alu a Jakuba, ktorí majú v letnom semestri zapísaný predmet Vývoj webových aplikácií v prostredí cloudu. Zároveň by sme si veľmi radi vyskúšali nasadenie funkčnej aplikácie do používania.

Ďalším dôležitým faktorom pri výbere tejto témy je aj možnosť pomôcť nášmu študijnému oddeleniu. Študijné referentky sú veľmi podstatnou súčasťou bezproblémového fungovania fakulty, a preto majú často práce vyše hlavy. Študenti sa na nich obracajú s rôznymi otázkami a oni im na každú veľmi ochotne odpovedajú. Preto by sme im chceli uľahčiť ich prácu systémom, ktorý ponúkne odpovede na opakujúce sa otázky študentov a vybavenie rôznych požiadaviek, či už zo strany študentov ale aj iných zamestnancov fakulty.

Myslíme si, že práve náš tím má potenciál na to, aby zvládol prekonať všetky problémy, ktoré so sebou táto téma prinesie a vytvoriť tak funkčnú a používateľsky príjemnú aplikáciu.

## A.4 Motivácia: Korekcia dynamických vlastností virtuálnych modelov komponentov vozidiel (13)

Autonómne vozidlá sú jedným z najzaujímavejších smerov, do ktorého sa informatici môžu ponoriť. Čiastočná účasť na tomto projekte by bola neobyčajnou príležitosťou pre náš tím.

Celý tím sa primárne zaujíma o metódy machine learningu a ich prepojenie s praxou. Veľkú výhodu vidíme aj v spolupráci so Strojníckou fakultou s firmou Slovakia Ring a so zahraničnou firmou Siemens Belgium. Pre Slovensko je v budúcnosti kľúčová spolupráca na medzinárodných projektoch, a preto je dôležité vytvorenie "dobrej" vizitky počas práce na týchto projektoch, ktorú náš tím vie zabezpečiť.

Na tejto téme nás zaujali nasledujúce 2 smery:

- AI - Testovanie matematických modelov pomocou AI by nemuselo byť v budúcnosti využívané iba pri autonómnych vozidlách. Náš tím môže poskytnúť kvalitu odvedenej roboty, skúsenosti a 100% nasadenie počas celej doby projektu. Tím vo svojich doterajších prácach opakovane používa umelú inteligenciu, ktorej sa plánujeme venovať aj v profesijnom živote. Pracovanie na tomto projekte by bolo jedinečnou príležitosťou, získavať nových znalostí a rozšírenia rozhľadov pri práci s ML.

- Simulačné prostredia - simulačné prostredia sú pre našu budúcnosť kľúčové. Pre vykonávanie niektorých činností sa zavedením simulátorov ušetrí čas a náklady na testovanie.

Tešíme sa na spoločné stretnutie a prípadne na našu ďalšiu spoluprácu.

## A.5 Príloha A

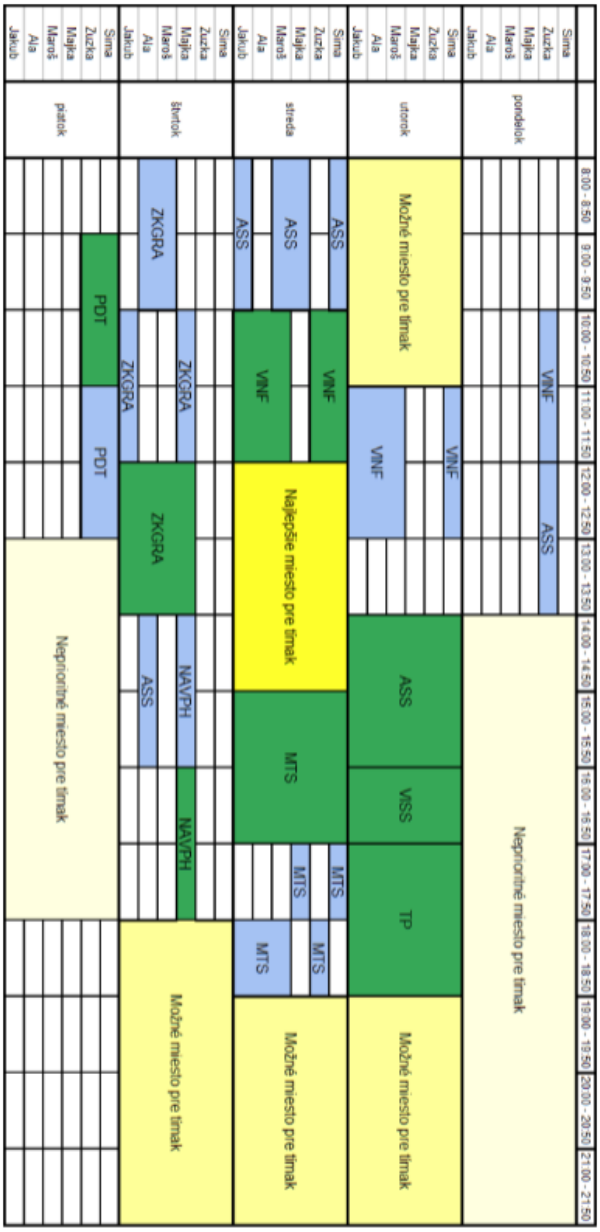
### Zoradenie tém podľa priority

| Priorita | Téma | Číslo témy |
|---|---|---|
| 1. | Automatické rozpoznávanie spektier [ARS] | 08 |
| 2. | Podporný informačný systém pre študijné oddelenie [CROSS-CHECK] | 19 |
| 3. | Korekcia dynamických vlastností virtuálnych modelov komponentov vozidiel [CarComponents] | 13 |
| 4. | VR laboratórium pre dištančné vzdelávanie [VRLab] | 17 |
| 5. | Educational and coworking driven orchestration portal [EDUCO] | 05 |
| 6. | Inteligentný informačný systém zameraný na projektový manažment a automatizáciu procesu verejného obstarávania [iPP] | 16 |
| 7. | FIFé International Cat Show [MIAOW] | 18 |
| 8. | Transformácia priestorov na bezpečné a inteligentné miesta na prácu [SmartSpace] | 06 |
| 9. | Vzdialené monitorovanie zdravotného stavu človeka pomocou E-Health | 09 |
| 10. | Safety panel a spätná analýza údajov pre vývoj autonómneho vozidla [avPANEL] | 12 |
| 11. | Webové IDE pre ASIC [ASICDE] | 02 |
| 12. | Platforma pre sledovanie dodávateľského reťazca s využitím technológie blockchain [S-Chain] | 14 |

*Tabuľka 1: Zoradenie tém podľa priority.*

## A.6 Príloha B

### Tímový rozvrh




*Obr. 6: Rozvrh. Miesta pre tímové stretnutia / spoločnú prácu sa bližšie špecifikujú podľa preferencií vedúceho.*

# B Prihláška na TP cup 2020

## B.1 Náš tím

Sme mladý a flexibilný tím pripravený prekonať každú výzvu. Voláme sa Mikasa po postave z anime a po značke výrobcu volejbalových lôpt. Sme ochotní na sebe pracovať a získavať nové skúsenosti z rôznych oblastí informatiky. Väčšina členov nášho tímu sa zaujíma o strojové učenie a neurónové siete. V oblasti umelej inteligencie máme nadobudnuté skúsenosti z predmetu Inteligentná analýza údajov, Umelá inteligencia a z bakalárskych prác. O kvalite našej práce svedčia aj 2 bakalárske práce, ktoré boli ocenené dekanom našej fakulty. Okrem toho máme skúsenosti s návrhom softvéru, vývojom aplikácií, či virtuálnou realitou.
Nebojíme sa výskumu rôznych oblastí informatiky, akou je napríklad automatické rozpoznávanie spektier. V rámci rôznych pracovných príležitostí sme nadobudli skúsenosti s prácou v tíme a s vývojom aplikácií, preto veríme, že sa v tíme dokážeme popasovať s akýmkoľvek problémom.

## B.2 Motivácia

Kvalita olivových olejov je posudzovaná primárne na základe chuti, vzhľadu a vône. V súčasnosti je hodnotenie kvality oleja regulované normami Európskej únie. Tieto nariadenia definujú 3 triedy olivového oleja, „extra panenský“ (EVOO), „panenský“ (VOO) a „lampantový“ (LOO). Každá vzorka oleja je klasifikovaná do 1 z týchto tried pomocou panelového testu. Experti v danej oblasti posudzujú oleje v rôznych kategóriách a prideľujú mu určité skóre. Aplikovaním štatistickej analýzy na skóre, ktoré bolo udelené účastníkmi testu sa vyhodnotí trieda oleja. Tento proces býva časovo náročný a drahý, preto sa hľadajú iné alternatívy vyhodnocovania kvality olivových olejov. Cieľom nášho projektu je umožniť rýchlejšiu analýzu oleja a vyhodnotenie bez nutnosti časovo a finančne náročnej manuálnej analýzy spektra expertom v danej oblasti.

## B.3 Náš projekt

Náplňou nášho projektu je automatické rozpoznávanie spektier, ktoré nám umožní analyzovať kvalitu vzoriek olivových olejov. Pre tento účel vytvárame aplikáciu, ktorá bude

využívať algoritmy strojového učenia a neurónových sietí. Využitím rôznych metód chceme dosiahnuť čo najvyššiu možnú presnosť pri klasifikovaní olivových olejov. Pomocou aplikácie bude možné zobraziť namerané vzorky vo forme diagramov, aby boli dáta zrozumiteľné aj pre ľudí, ktorí nie sú expertami v oblasti analýzy spektier.

## B.4 Ciele projektu

Hlavným cieľom nášho projektu je vytvoriť systém na automatické rozpoznávanie spektier, ktorý využíva metódy strojového učenia a neurónových sietí na klasifikáciu olivových olejov.
Navrhnutý systém by mal vedieť spracovávať spektrum, ktoré je výstupom merania meracieho zariadenia AMIS (Advanced Ion Mobility Spectrometer). Spektrum si môžeme predstaviť ako časovú sekvenciu meraní spektrálnych koeficientov uloženú vo formáte *.txt. Systém by mal vedieť nájsť rozličné črty pre olivové oleje, na základe ktorých bude môcť vyhodnotiť triedu oleja.
Na nami identifikované požiadavky sme navrhli nasledujúce riešenie:

- Poskytnuté dáta obsahujú vzorky 200 olejov, ktoré sú rozdelené do 3 tried. Každá vzorka oleja obsahuje viac ako 10 000 spektier uložených v samostatnom súbore. Z týchto súborov vytvoríme celistvú tabuľku, ktorá bude obsahovať všetky informácie o vzorkách.

- Každý olej je charakterizovaný nameranými hodnotami v spektrách. Zo spracovaných dát vypočítame črty, ktoré následne použijeme pri klasifikácií. V projekte sa zameriame na základné črty, akými sú priemer alebo maximum, taktiež vyhľadáme aj komplexnejšie črty. Pozrieme sa aj na hodnoty v špecifických spektrách, kedy môže nastať zmena oproti inej triede.

- V projekte sa zameriame na niekoľko algoritmov strojového učenia, ktoré budeme využívať na klasifikáciu olivových olejov. Chceme vyskúšať algoritmy, akými sú Náhodný les (Random Forest), KNN, SVM a neurónové siete. Natrénované modely vyhodnotíme a tie najúspešnejšie z nich použijeme vo výslednom produkte. Budeme pritom dohliadať na celkovú úspešnosť, ako aj úspešnosť pre jednotlivé triedy.

- Vytvoríme systém na rozpoznávanie tried olejov, pomocou ktorého bude môcť používateľ zobrazovať dáta vo forme diagramov a klasifikovať vzorku oleja. Klasifikačné modely a spôsoby vyhodnotenia si bude môcť používateľ prispôsobiť.

Našim cieľom je vytvoriť funkčnú a spoľahlivú aplikáciu, ktorá bude môcť byť v praxi využívaná, aby sa zabránilo zamieňaniu kvalitných a nekvalitných olejov, aby nedochádzalo k stratám ziskov výrobcov olivových olejov. Veríme, že kombináciou našich skúseností, nápadov a moderných technológií dokážeme vytvoriť presný a spoľahlivý produkt.

# C Metodika komunikácie

## C.1 Komunikačné nástroje

Metodika komunikácie definuje pravidlá komunikácie medzi jednotlivými členmi tímu. V tejto metodike sú predstavené komunikačné kanály, ktoré sa používajú v tíme, na to aby sa predchádzalo vzniku komunikačnému ruchu.

#### Google Meet

Nástroj Google Meet sa používa na pravidelné online stretnutia členov tímu s vedúcim tímu z dôvodu dištančnej výučby. Tieto stretnutia sú naplánované v rozvrhu členov tímu.

#### MsTeams

Nástroj MsTeams sa používa na pravidelné online stretnutia členov tímu.

#### Komunikačné kanály

- General - určený na komunikáciu všeobecných vecí a riešenie problémov v projekte §
- Classification - určený na komunikáciu o spracovávaní dát, vytváraní klasifikačných metód a vyhodnocovaní klasifikátorov
- Web - určený na komunikáciu k vytváraniu wireframov, frontendu a backendu, nasadzovania stránky
- Stand-up - kanál, ktorý nahrádza ranné stand-upy, členovia tímu sem napíšu čo idú robiť alebo aké problémy aktuálne riešia
- Ready for review - kanál, kde sa pridávajú hotové tasky pripravené na review

Členovia tímu poznajú zameranie jednotlivých komunikačných kanálov a využívajú ich podľa spomenutých pravidiel. Vzniknuté problémy rieši tím spoločne, so vzniknutými problémami sa člen tímu netrápi sám. Ak chce člen spomenúť v komunikácií iného člena tímu, označí ho pomocou @. Každý člen tímu môže vytvoriť nový kanál. Nový kanál

musí byť výstižne pomenovaný a musí byť predpoklad, že sa bude v budúcnosti využívať. Po vytvorení nového komunikačného kanálu ho daný člen pridá a primerane opíše v metodike.

### Jira

Komunikácia o úlohách (otázky, riešenie bugov, ...) sa rieši prostredníctvom pridávania komentárov k úlohám v nástroji Jira. Každý člen tímu môže pridať komentár k akejkoľvek úlohe. Komentáre musia byť stručné a týkať sa konkrétnej úlohy. V prípade, že je komentár príliš dlhý je vhodné využiť komunikačný kanál v násroji MsTeams.

### Email

Komunikácia tímu o projekte s ďalšími stranami prebieha prostredníctvom emailovej adresy mikasa.fiit@gmail.com. Každý člen tímu má k emailu prístup a je povinný ho pravidelne sledovať. V prípade, že člen tímu príjme novú správu, informuje o nej zvyšok tímu.

### Messenger

Messenger slúži na súkromnú komunikáciu členov tímu. Pomocou tohto komunikačného kanálu sa dohadujú stretnutia členov tímu. Členovia tímu môžu použiť tento kanál na upozornenie ostatných členov o dokončení úlohy.

# D Metodika vývoja v Pythone

## D.1 Manipulácia s projektom

Pre zefektívnenie a zachovanie rovnorodosti práce na projekte sú v tomto dokumente definované pravidlá manipulácie s projektom.

Odporúčaným IDE pre písanie programov v jazyku Python je Pycharm. Zvoleným jazykom pre vytváranie pomenovaní je anglický jazyk.

### Názvoslovie pre súbory a adresáre

Všetky priečinky budú pomenované:

- v anglickom jazyku
- s veľkým začiatočným písmenom

Súbory s koncovkou *.py* sa budú pomenovávať v závislosti od ich obsahu:

- **Triedy** - V prípade, že obsahom súboru bude definícia triedy, súbory sa pomenujú cez cammelCase, s výnimkou veľkého začiatočného písmena. Podobne ako súbory. Napríklad trieda, ktorá definuje príspevky pre používateľov sa bude volať *UserPost*.
- **Scripty** - Ak obsahom súboru nie sú triedy, súbory budú pomenované cez *lower_case*, t.j. všetky slová v názve budú začínať malým začiatočným písmenom a budú oddelené cez podčiarkovník. Napríklad, pre funkcie upravujúce reťazce, by bol názov súboru *string_modifier*.

#### Adresáre v projekte

Pre zefektívnenie, sprehľadnenie a zjednodušenie práce na projekte budú v hlavnom projekte existovať nasledujúce adresáre:

- **Preprocessing** - všetky súbory z predspracovania.
- **FeatureSelection** - funkcie pre vytváranie čŕt.
- **Neurons** - neurónové siete.
- **StandardClasification** - štandardné klasifikátory.
- **Data** - adresár obsahuje všetky dáta. Tento adresár obsahuje aj:

  - **Raw** - adresár obsahujúci predspracované dáta.
  - **Features** - adresár obsahujúci dáta s vypočítanými črtami, a adresáre s obrázkami a diffmapami
  - **Original** - adresár originálne poskytnuté dáta, ešte pred spracovaním.
- **Statistics** - adresár obsahujúci štatistické údaje, grafy a iné.
- **UnitTests** - adresár obsahujúci unit testy.
- **Models** - adresár obsahujúci klasifikačné modely.
  - **Neurons** - adresár obsahujúci modely neurónových sietí.
  - **StandardClasification** - adresár obsahujúci modely štandardných klasifikátorov.

## D.2 Písanie programov

### Hlavička dokumentu - Importy a globálne premenné

Všetky globálne importy sa budú nachádzať na začiatku súboru. Prvý import sa umiestni na riadok č.1. V prípade že sa za importom globálne premenné:

- nachádzajú - Nasledujú za importom 1 prázdny riadok.
- nenachádzajú - Nasledujú za importom 2 prázdne riadky.

Následne budú umiestnené deklarácie globálnych premenných. Názvy globálnych premenných budú napísané veľkými písmenami, oddelených podčiarkovníkom. Napríklad globálna premenná pre maximálnu hodnotu bude označená ako *MAX_VALUE*. Príklad je uvedený na obrázku č. 7

```
import csv

MAX_VALUE = 1000000


def function:
    # some code to add
```

*Obr. 7: Príklad deklarácie hlavičky súboru*

### Telo dokumentu

### Funkcie

Medzi funkciami budú 2 voľné riadky.
V prípade, ak sa jedná o spustiteľný súbor, je povinná funkcia main v tvare:
*if __name__ == "__main__":*

### Podmienky

Uprednostnenie *or* a *and* pred || a *&&*.
Pre podmienky *if, elif, else* bude blok kódu začínať vždy na novom riadku.

### Cykly

Blok kódu začínať vždy na novom riadku.

### Polia s kľúčovou hodnotou

Pri deklarácii polí s kľúčovou hodnotou sa odporúča nepridávať čiarku na koniec posledného riadku. Ukážka je uvedený na obrázku č. 8

```
thisdict =  {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

*Obr. 8: Príklad deklarácie polí s kľúčovou hodnotou*

### Reťazce

Deklarácie reťazcov budú primárne začínať a končiť dvojitou úvodzovkou ".

### Tvorba pomenovaní

### Črty

Názvy čŕt sa skladajú z dvoch častí, ktoré sú oddelené podčiarkovníkom. Prvá časť je názov funkcie, ktorá danú črtu ráta. Druhú časť tvorí stĺpec alebo viacero stĺpcov

(riadkov), nad ktorými sa daná črta ráta.
Príklad črty priemer všetkých hodnôt zo stĺpca v_1:

*mean_v_1*

Ak je v názve viac ako jeden stĺpec (riadok) oddeľujú sa pomocou pomlčky. V prípade názvov, kde sa nachádzajú viaceré stĺpce v kombinácii s riadkami píšu sa iba čísla daného stĺpca (riadku). Ak názov funckie tvorí viac slov spájajú sa pomocou pomlčky.
Príklad výpočtu maxima z bodu (v-s-n):

*max-point_1-7-10*

V prípade, že chceme pri nejakej funkcii vyrátať črtu nad všetkými stĺpcami použijeme ako názov stĺpca slovo „ALL“.
Príklad výpočtu minima nad všetkými stĺpcami:

*min_ALL*

### Premenné

Názvy premenných v programe sa budú písať cez lower_case. Názvy premenných by nemali obsahovať slovesá, preferované sú podstatné mená. Odporúča sa používať jednotné číslo, s malými výnimkami. Napríklad názov premennej, ktorá uchováva príspevky používateľov sa bude volať user_posts.

### Triedy

Triedy budú pomenované cez cammelCase s veľkým začiatočným písmenom. Napríklad trieda uchovávajúca príspevky používateľov sa bude volať *UserPosts*.

V prípade nedefinovaných pravidiel sa budeme spoliehať na štandardy PEP-8 a PEP-257. Pravidlá definované v tomto dokumente sú nadradené pravidlám týchto štandardov.

# E Metodika testovania

## E.1 Testovanie

Testovanie je dôležitou súčasťou vývoja, nakoľko pomáha validácií riešenia a identifikácií chýb. Preto by malo zastrešovať čo najväčšiu časť vytvoreného produktu.

Vzhľadom na druh produktu, ktorý vyvíjame a dátovú prácu s ním súvisiacu, sme sa rozhodli využiť najmä jednotkové testy a akceptačné testy.

### Jednotkové testy

Jednotkové testy budú vykonávané pomocou knižnice **pytest**. V rámci využívania tejto knižnice je preto nutné dodržať konvenciu pomenovávania.

- **Súbory:** názov súboru sa musí začínať slovom „test". Aj napriek metodike vývoja v Pythone sa budú súbory pomenovavávať cez *lower_case* nezávisle na tom, či obsahujú triedy alebo nie. Príklad: *test_features.py*
- **Triedy:** Podľa metodiky vývoja v Pythone sa triedy budú pomenovávať pomocou *CammelCase*. Prvé slovo v triede musí byť slovo „Test".
- **Metódy:** Metóda musí začať slovom „test". Dodržaná bude metodika vývoja v Pythone, čo znamená, že metódy budú písané ako lower_case.

Príklad jednotkového testu:

```
def test_mean_normal(self):
    value = add_two_numbers(5, 2)
    assert value == 7
```

### Nutnosť jednotkových testov

Jednotlivé komponenty/metódy budú testované jednotkovými testmi podľa uváženia osoby, ktorá daný komponent/metódu napísala, čím musí zvážiť nutnosť jej iného ako fyzického testovania.

Odporúča sa však mať jednotkovými testami pokrytú čo najväčšiu časť kódu, vďaka čomu si môžu vývojári byť neustále istí správnym fungovaním komponentov/metód.

Existencia jednotkových testov taktiež napomôže review nových častí kódu, vďaka čomu sa ušetrí čas na manuálnom testovaní.

Taktiež sa odporúča, aby jednotkový test k určitému komponentu/metóde napísala osoba, ktorá daný komponent/metódu vytvorila. Ušetrí sa tak čas potrebný pre naštudovanie komponentu/metódy iným členom tímu.

### Organizácia testov

Všetky súbory s testmi budú v špeciálnom adresári „*UnitTests*" z dôvodu ich jednoduchšej organizácie. Testy budú organizované do súborov a tried v závislosti od ich logickej organizácie.

Každá metóda v súboroch s testmi zobrazuje samostatný test.

### Spustenie jednotkových testov

Testovanie pomocou jednotkových testov sa bude spúšťať príkazom „*pytest -v*" pomocou konzoly z adresáru „*UnitTests*". Prepínač *-v* slúži pre výpis stavu testovania po jednotlivých testoch.

### Akceptačné testy

Na konci fázy tvorby softvéru, ktorý bude možné použiť pre používateľsky jednoduchšiu klasifikáciu vzoriek oleja sa vykoná akceptačný test.

Akceptačný test bude vytvorený na začiatku fázy tvorby softvéru. Test bude spísaný na základe dohody s product ownerom. Zadané budú možné vstupy a predpokladané výsledky k nim. Testované musia byť aj hraničné vstupy, ktorými sa overí schopnosť softvéru reagovať na chyby a zotaviť sa z chybových stavov.

Pre akceptovanie akceptačného testu je nutné bez výnimky splniť všetky dohodnuté body testu.

# F Metodika splnenej úlohy

## F.1 Definition of Done

V rámci vývoja softvéru viacerými členmi, je nutné si v rámci tímu dohodnúť jasné pravidlá, ktoré okrem iného označujú aj to, kedy je možné určitú činnosť považovať za dokončenú. Takéto pravidlá nazývame Definition of Done.

Definition of Done v rámci nášho tímu je prispôsobená podľa veľkosti celku, ktorý je potrebné označiť ako „dokončený". V rámci projektu teda rozlišujeme definíciu dokončenia pre:

- Task
- User Story
- Projekt

#### Definition of Done pre Task

Task je najmenšia jednotka, ktorú je možné považovať za dokončenú. Ako task sa označuje činnosť, ktorá je vykonateľná jednou osobou v rámci User story.

Task má jasne stanovený názov, ktorý označuje čo je nutné vykonať. Činnosť však lepšie opisuje popis tasku, ktorý je viac detailnejší ako samotný popis.

Pre považovanie tasku za dokončený je nutné:

- Vykonať popísanú činnosť v rámci kontextu projektu
- V prípade kódu:
  - Otestovať samotný kód (jednotkové testy / manuálne)
  - Otestovať správanie kódu v spolupráci s inými komponentmi
  - Opísať novovytvorený kód
- Prejsť si review tasku s iným členom tímu
- Zapracovať prípadné zmeny z review a opäť vykonať review

**Definition of Done pre User Story**

User Story je celok, ktorý má v rámci projektu priniesť product ownerovi určitú pridanú hodnotu. Je zložený z Taskov, ktoré presne popisujú činnosti, ktoré je nutné vykonať pre splnenie User Story.

Aby sme mohli User Story považovať za dokončené, je nutné:

- Považovať všetky Tasky súvisiace s User Story za dokončené
- Priniesť funkcionalitu opísanú v zadaní User Story
- Overiť funkcionalitu ako celok
- Zhodnotenie všetkých členov tímu o schválení User Story za dokončené

**Definition of Done pre Projekt**

Projekt je možné považovať za dokončený, keď:

- Dodá sa plánovaná funkcionalita
- Presnosť rozpoznávania oleja bude na najlepšej možnej hranici
- Tím zrealizuje všetky možné nápady zlepšovania výsledkov
- Nebude viac možné posunúť projekt na lepšiu úroveň

# G Metodika práce s úlohami

## G.1 Práca s úlohami

Pre zachovanie rovnorodosti práce, podporu spolupráce tímu na projekte a prevencii vzniku nezrovnalostí pri práci s úlohami na projekte sú v tomto dokumente definované pravidlá manipulácie s úlohami, spolu s definíciou typov úloh.

Zvoleným softvérom pre vytváranie a správu rôznych typov úloh je Jira. Zvoleným jazykom pre vytváranie úloh je jazyk slovenský, jazyk pre názvy stavov úloh je jazyk anglický.

#### Issue types

#### Epic

V projekte sa vyskytujú nasledovné epiky:

- **Administrácia** - združuje všetky administratívne úlohy a user stories. Patria sem všetky issue types, ktoré sa týkajú administratívnej časti projektu, ako napríklad TP cup, udržiavanie metodík a príprava na stretnutia.
- **Klasifikácia** - združuje všetky úlohy súvisiace s klasifikačnou časťou projektu. Napríklad rozpoznávanie druhov olejov a vytváranie nových čŕt.
- **Web** - združuje všetky úlohy súvisiace s webovou reprezentáciou tímu. Napríklad vytvorenie wireframov pre web, alebo vytvorenie frontendu. Tiež zastrešuje spravovanie servera.
- **Aplikácia** - združuje všetky úlohy súvisiace s aplikačnou časťou projektu. Napríklad špecifikácia požiadaviek, návrh prípadov použitia (Use case), vytvorenie aplikácie.

#### Používateľský príbeh

Používateľské príbehy, anglicky user stories, sa budú písať s vedúcimi projektu, v prípade, že sa tak nestane, musia byť príbehy akceptované vedúcimi. Podúlohy sú tvorené členmi tímu.

Do opisu príbehu sa doporučuje zahrnúť:

- **Úvod** - Ako KTO?, potrebuje ČO? a PREČO?
- **Analýza** - jednoduchú analýzu úlohy.
- **Podúlohy** - zoznam podúloh potrebných na splnenie príbehu.
- **Must be** - zoznam bodov, ktoré musia byť v rámci úlohy splnené pri dodaní, aby mohlo byť dodanie akceptované.

#### Podúloha

Podúlohy, anglicky subtasky, sú tvorené členmi tímu, na spoločných stretnutiach. Story pointy podúlohy sú prideľované na spoločných stretnutiach metódou planning poker.

Do opisu podúlohy sa doporučuje napísať:

- **Opis** - Stručný opis úlohy.
- **Vstupy a výstupy** - V prípade implementačných úloh sa môže nachádzať príklad vstupu a výstupu práce alebo pridanej funkcionality.
- **Testy** - Rozhodnutie, či je potrebné pre implementačnú úlohu vytvárať testy.
- **Kontrolovanie výstupu** - Zoznam bodov, ktoré treba skontrolovať, keď sa úloha nachádza v stave *ready for review*.

### Manipulácia s úlohami

#### Vytváranie úloh a presúvanie úloh do šprintu

Úlohy sa prednostne vytvárajú a presúvajú do šprintov na tímových stretnutiach. Pridávanie úloh do šprintu počas jeho priebehu sa neodporúča. Výnimkou je nahlásenie chýb v projekte.

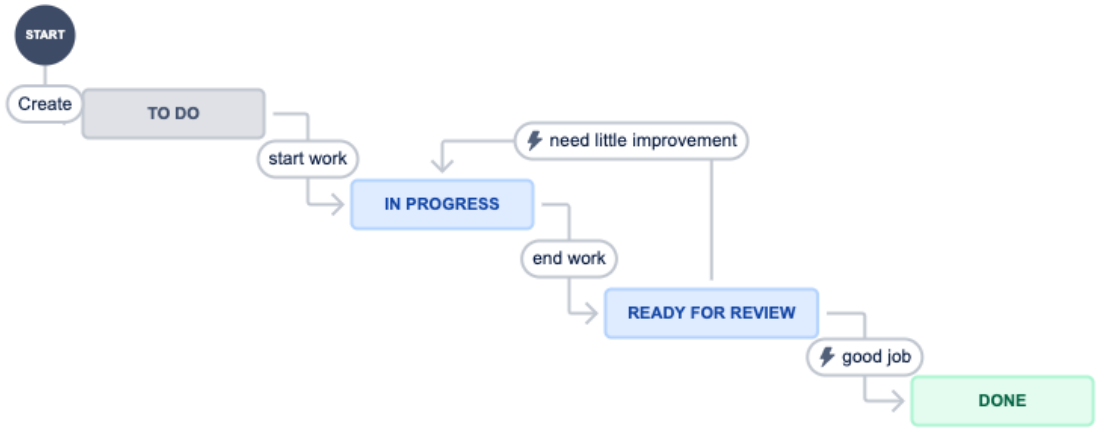
#### Stavy v úloh

Zoznam stavov úloh:

- **To Do** - Zoznam úloh, ktoré treba dokončiť počas šprintu.
- **In Progress** - Zoznam rozpracovaných úloh alebo úloh, na ktorých sa začína pracovať.

- **Ready For Review** - Zoznam dokončených úloh, ktoré sú pripravené na kontrolu. Úloha musí byť dokončená v zmysle pravidiel Metodiky splnenej úlohy. Kontrolu úloh vykonáva Jakub Kučečka.
- **Done** - Hotové úlohy, ktoré prešli kontrolou.

Presun medzi stavmi je povolený len v jednom smere, zľava doprava. Úlohy môžu presúvať ľubovoľní členovia tímu s výnimkou stavu *Ready For Review*. Z tohto stavu úlohy môžu presúvať aj spätne do stavu *In Progress* a môže ich presúvať len člen tímu zodpovedný za kontrolu.




*Obr. 9: Diagram prechodov medzi stavmi úloh*

# H Metodika verziovania programu

## H.1 Práca v nástroji GitHub

Pre zachovanie rovnorodosti práce, podporu spolupráce tímu na projekte a prevencii vzniku nezrovnalostí vo verziách programu na projekte sú v tomto dokumente definované pravidlá manipulácie s verziami projektu.

Zvoleným kolaboratívnym nástrojom na projekte je Github. Slovenský jazyk sa bude využívať pri vytváraní:

- potvrdzujúcich správ, (ďalej commit message)
- názvov vetiev patriacich úlohám

V anglickom jazyku budú pomenované vetvy master a dev.

#### Vetvy

#### Master

Hlavnou vetvou je vetva master. Do tejto vetvy sa budú pridávať len hotové funkcionality spustiteľného programu. V tejto vetve by sa **nesmie** nachádzať program s nedokončenou funkcionalitou a s chybami.

Nad touto vetvou **môže** existovať len vetva dev. Do tejto vetvy sa bude vykonávať merge vetvy dev a to **prednostne** na konci šprintu po odsúhlasení v tíme. Pridaná časť programu **musí** mať hodnotu pre zákazníka. Nedokončená funkcionalita sa do tejto vetvy nepridáva.

#### Dev

Vetva určená na vývoj je vetva dev. Do tejto vetvy sa budú pridávať len ukončené funkcionality z pridelených úloh. Ak si obsah úlohy žiada testovanie, nesmie byť do vetvy pridaná neotestovaná časť programu. Výnimkou tohto prípadu je scenár, kedy je dané testovanie súčasťou inej úlohy.

Nad touto vetvou **môžu** vznikať len vetvy, ktoré sú vytvorené za účelov splnenia pridelenej úlohy. Do vetvy dev sa môže vykonať merge len s vetvou, ktorej:

- zodpovedajúca úloha je splnená na 100%

- pridaný program zodpovedajúci úlohe prešiel kontrolou druhej strany.

Spájanie vetiev bude vykonávať Jakub Kučečka. Riešenie konfliktov počas spájania bude riešiť osoba, ktorej úloha bola pridelená.

### Vetvy úloh

Pre každú úlohu bude existovať vetva, ktorá sa vytvorí z vetvy dev tesne pred začatím programovania a zanikne po ukončení úlohy vykonaním merge do vetvy dev. Pre pod úlohy sa nemusí vytvárať vetva.

Vetvy budú nazývané podľa čísla úloh, priradené systémom Jira (bez diakritiky). Vzor:

ASR-1_vytvorenie_metodiky_verziovania_programu

Okolo pomlčky na nenachádzajú medzery. Za id úlohy nasleduje podčiarkovník a názov úlohy v slovenskom jazyku. Všetky začiatočné písmená sú malé a slová sú rozdelené podčiarkovníkom.

### Commit messages

Príklad commit message:

ASR-1 pridanie funkcionality ....

Začína sa id úlohy, nasleduje popis k pridanej funkcionalite. Odporúča sa podrobný popis.

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Dokumentácia k tímovému projektu

# Moduly systému

Vedúci tímu: Mgr. Martin Sabo, PhD., Ing. Marta Prnová, PhD.
Číslo tímu: 05
Kontakt: mikasa.fiit@gmail.com

# Obsah

# 1 Analýza

Táto kapitola je zameraná na analýzu problematiky. V prvej časti je vysvetlený problém klasifikácie olivových olejov, následne je opísaný prehľad existujúcich riešení. V ďalšej časti tejto kapitoly uvádzame funkcionálne a nefunkcionálne požiadavky na vyvíjaný softvér. V poslednej časti sú opísané a analyzované dáta pre trénovanie klasifikátorov poskytnuté vedúcim projektu (PO).

## 1.1 Úvod do problematiky

Olivové oleje sú posudzovaná primárne na základe chuti, vzhľadu a vône. Kvalita olivového oleja nezávisí len od využitia technologických postupov pri spracovaní olív. Na jeho kvalitu vplývajú aj prírodné faktory.

Významným faktorom, ktorý ovplyvňuje chuť oleja je druh pôdy. Pôda obsahuje rôzne zložky, ktorých výskyt a koncentrácia sa často líšia v závislosti od oblasti a nadmorskej výšky. Zloženie pôdy vo veľkej miere ovplyvňuje výslednú chuť a celkovú kvalitu olivového oleja.

Ďalším faktorom, ktorý ovplyvňuje kvalitu oleja je typ podnebia a s ním úzko súvisiace zrážky. Podnebie indukuje syntézu sekundárnych metabolitov, ktoré sa v rámci spektrometra analyzujú. Kvalita oleja závisí od koncentrácie látok. Napríklad, keď sa v pôde nachádza väčšia koncentrácia vápnika a podnebie je suchšie, budú sa produkovať látky s polyfenolyckými štruktúrami. Naopak, v prípade vlhkého podnebia a hornatého kraja, budú sa produkovať lipidické štruktúry.

V súčasnosti je hodnotenie kvality oleja regulované normami Európskej únie. Tieto nariadenia definujú 3 triedy olivového oleja, „extra panenský“ (EVOO), „panenský“ (VOO) a „lampantový“ (LOO).

Každá vzorka oleja je klasifikovaná do jednej z týchto tried pomocou panelového testu. Experti v danej oblasti posudzujú oleje v rôznych kategóriách a prideľujú mu určité skóre. Aplikovaním štatistickej analýzy na skóre, ktoré bolo udelené účastníkmi testu sa vyhodnotí trieda oleja. Tento proces býva časovo náročný a drahý. Z tohto dôvodu je potrebné sa v súčasnosti zamerať na modernejšiu a lacnejšiu metódu zisťovania kvality oleja. Preto sa kvalita zisťuje napríklad pomocou spektrometra.

Spektrometer je druh vedeckého prístroja, ktorý umožňuje skúmať prvkové chemické

zloženie látky či objektu na báze merania odrazeného svetla respektíve odrazenej vlnovej dĺžky svetla a jeho absorpcie alebo na základe merania vzniknutého svetla.

## 1.2 Existujúce riešenia

V článku [3] sa autori zaoberali falšovaním olivových olejov a rozpoznávaním pravosti oleja. Teda zisťovali, či sa jedná o čistý a kvalitný olej alebo bol zamenený napríklad za slnečnicový olej.

Autori navrhli dve metódy kontroly kvality oleja pomocou klasifikačných metód:

- 32 získaných vstupov je aplikovaných celých bez úprav priamo do klasifikátora
- 32 získaných vstupov je zredukovaných na 8 a tie sú aplikované do klasifikátora

### Dataset

Surové dáta z olejov boli zozbierané a zdigitalizované pomocou zariadenia e-nose (Cyranose 320), čo je imitačný nástroj čuchu, ktorý má 32 senzorov a z nich boli vygenerované trénovacie a testovacie dáta.

Bolo získaných 12 rôznych druhov olejov (z rôznych regiónov v Balikesir v Turecku) a pre každý druh bolo zozbieraných 10 rôznych vzoriek. Testovacie a trénovacie dáta týchto dvanástich druhov sa nemiešali.

Metódy mali rôzne vstupné dáta (Obrázok 1), pretože pri druhej metóde bol použitý algoritmus PCA, ktorý redukuje dimenzionalitu (počet atribútov).

| For the first method | | | | For the second method | | | |
|---|---|---|---|---|---|---|---|
| S1 | ⋯ | S32 | Class | svd_1 | ⋯ | svd_8 | Class |
| 0,001309 | ⋯ | 0,00054 | (1) | 0,035625215 | ⋯ | 0,046224595 | (1) |
| 0,001093 | ⋯ | 0,000626 | (2) | 0,039502122 | ⋯ | −0,007833031 | (2) |
| 0,001858 | ⋯ | 0,000817 | (3) | 0,063373384 | ⋯ | 0,09547233 | (3) |
| 0,001043 | ⋯ | 0,000465 | (4) | 0,033603573 | ⋯ | 0,07006505 | (4) |
| 0,000819 | ⋯ | 0,00032 | (5) | 0,026595428 | ⋯ | −0,01141755 | (5) |
| 0,000673 | ⋯ | 0,000177 | (6) | 0,020821755 | ⋯ | −0,019554513 | (6) |
| 0,001115 | ⋯ | 0,000454 | (7) | 0,03409242 | ⋯ | −0,146848992 | (7) |
| 0,001757 | ⋯ | 0,000852 | (8) | 0,050890223 | ⋯ | −0,00503703 | (8) |
| 0,007326 | ⋯ | 0,003501 | (9) | 0,248933465 | ⋯ | 0,008284441 | (9) |
| 0,002671 | ⋯ | 0,001396 | (10) | 0,087664315 | ⋯ | 0,043067494 | (10) |
| 0,00404 | ⋯ | 0,002006 | (11) | 0,131178594 | ⋯ | 0,00053172 | (11) |
| 0,003203 | ⋯ | 0,001444 | (12) | 0,105479212 | ⋯ | −0,034223378 | (12) |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |

*Obr. 1: Ukážka dát pre prvú metódu (vľavo) a pre druhú metódu (vpravo) [3]*

Zoznam použitých klasifikátorov:

- *k*-NN
- Naive bayes
- Decision Tree
- Support Vector Machine (SVM)
- Artificial Neural Networks (ANN)
- Linear Discriminate Analysis (LDA)

Autori v článku [3] uviedli aj nastavenia hyperparamterov daných klasifikátorov. Tieto nastavenia môžeme vyskúšať aj v našom projekte, keďže autori pomocou nich dosiahli výborné výsledky.

Nastavenia hyperparametrov *k*-NN:

- Measure type = Numerical Measures
- Numerical measure = Kernel Euclidean Distance
- Kernel type = Anova
- Kernel gamma = 0.5
- Kernel degree = 0.5

Nastavenia hyperparametrov Decision Tree:

- Criterion = gain ratio
- Maximal depth = 20
- Confidence = 0.25
- Minimal gain = 0.1
- Minimal leaf size = 1
- Minimal size for split = 4
- Number of prepruning = 3

Nastavenia hyperparametrov *k*-NN:

- SVM type = C-SVC
- Kernel type = linear
- C = 2.0
- Cache size = 80
- Epsilon = 0.001

## Výsledky

Na vyhodnocovanie úspešnosti klasifikátorov sa používala metrika accuracy.

Výsledky pre prvú metódu (Obrázok 2) sú v rozpätí 45% - 65%. Najlepší výsledok dosiahla neurónová sieť, ktorá má náskok takmer 10% nad najlepším tradičným klasifikátorom SVM. Ďalšie klasifikátory mali približne rovnakú úspešnosť.

| | Train | Test |
|---|---|---|
| Naïve Bayes | 65% | 45,83% |
| *K*-NN | 66,67% | 48,33% |
| LDA | 94,17% | 52,50% |
| Decision Tree | 98,33% | 52,50% |
| ANN | 71,67% | 65,83% |
| SVM | 74,17% | 56,67% |

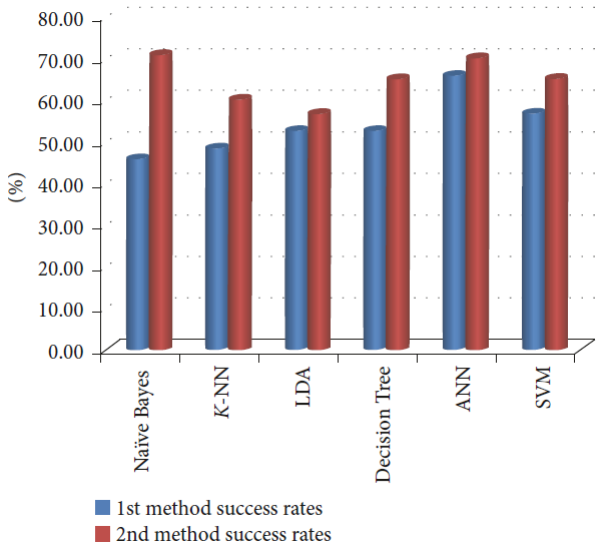
*Obr. 2: Vyhodnotenie prvej metódy pomocou metriky accuracy. [3]*

Výsledky pre druhú metódu (Obrázok 3) sú v rozpätí 56% - 70%. Najlepší výsledok dosiahol klasifikátor naivný bayes a hneď za ním bola neurónová sieť. Aj v tomto prípade je rozdiel medzi najlepším a najhorším klasifikátorom takmer 15%. Ďalšie klasifikátory mali približne rovnakú úspešnosť v danom rozpätí.

| | Train | Test |
|---|---|---|
| Naïve Bayes | 85,83% | 70,83% |
| *K*-NN | 70% | 60% |
| LDA | 75,83% | 56,67% |
| Decision Tree | 98,33% | 65% |
| ANN | 95% | 70% |
| SVM | 90% | 65% |

*Obr. 3: Vyhodnotenie druhej metódy pomocou metriky accuracy. [3]*

Pri porovnaní metód (Obrázok 4) môžeme vidieť, že druhá metóda bola oveľa úspešnejšia ako prvá. Teda aj pri zredukovanom počte dát klasifikátor dokázal správne určiť triedu. Najväčší rozdiel je vidieť pri klasikátory naivný bayes, kde sa úspešnosť zväčšila až o 25%.




*Obr. 4: Porovanie výsledkov metód. [3]*

V článku [2] sa autori zaoberali viacerými možnosťami zaradenia oleja pomocou klasifikačných metód. Teda vznikli tri hlavné body, ktorým sa venovali. Tie sú nasledovné:

- Zaradenie oleja do troch tried podľa kvality oleja.
- Zaradenie oleja do piatich tried podľa odrodu olív.
- Zaradenie oleja do troch tried podľa regiónu.

### Dataset

Bolo zozbieraných 193 vzoriek (5 rôznych druhov olív) z rôznych oblastí Grécka. Boli získavané spektrálne dáta vo forme 2001 vlnových dĺžok v rozmedzí od 20 do 700 nm. Taktiež boli zaznamenané charakteristiky oleja ako kyslosť a oxidačné ukazovatele – hodnota (PV), rovnako aj hodnota absorpcie K232 a K270, ktoré sú tradične spojené s kvalitou oleja.

Pri testovacích a trénovacích dátach bola použitá krížová validácia, konkrétne metóda *leave one out*.

### Použité klasifikátory

Zoznam použitých klasifikátorov:

- $k$-NN
- Linear discriminant analysis (LDA)
- Logistic regression (LR)
- Quadratic discriminant analysis (QDA)
- Artificial Neural Networks (ANN)

### Výsledky

Pri vyhodnotení (Obrázok 5) bola použitá metrika accuracy a vyhodnocovali sa dva body. Variety – odroda olív a Sens – kvalita oleja. V prípade odrody olív mali klasifikátory väčšiu úspešnosť ako pri kvalite. Avšak pri hodnotení kvality sú tieto výsledky dobré, nakoľko niekedy ani ľudia kontrolujúci kvalitu nemávajú úspešnosť 100%

| Method | Success | Criterion: Variety | | Criterion: Sens | | Software |
|---|---|---|---|---|---|---|
| | | Training set | Leave-one-out | Training set | Leave-one-out | |
| LDA | Number | 186 | 177 | 87 | 81 | SPSS |
| | [%] | 99.5 | 94.7 | 95.6 | 89.0 | |
| QDA | Number | 187 | 131 | 91 | 60 | SAS |
| | [%] | 100.0 | 70.1 | 100.0 | 65.9 | |
| KNN | Number | 186 | 185 | 85 | 80 | SAS |
| | [%] | 99.5 | 98.7 | 93.4 | 87.9 | |
| LR | Number | 187 | – | 91 | – | SPSS |
| | [%] | 100.0 | – | 100.0 | – | |
| ANN | Number | – | – | 91 | – | JMP |
| | [%] | – | – | 100.0 | – | |

*Obr. 5: Porovnanie výsledkov pre odrody a kvalitu. [2]*

### Zhodnotenie

Vďaka analýze článkov sme zistili, že pre získanie vysokej úspešnosti postačujú aj tradičné klasifikátory. Avšak neurónová sieť sa ukázala ako vhodný klasifikátor, keďže vždy patrila k skupine najlepších klasifikátorov. Taktiež sme zistili, že náš nápad vložiť všetky získané dáta do klasifikátora bez úprav a výberu čŕt nemá veľký zmysel. Zredukované dáta mali oveľa väčšiu úspešnosť ako neupravované dáta. Takisto môžeme vyskúšať nastavenie hyperparametrov a krížovú validáciu.

## 1.3 Požiadavky na softvér

V tejto časti analýzy sme opísali identifikované požiadavky na softvér. V časti Funkcionálne požiadavky sme identifikovali funkcionality, ktoré má systém vykonávať. V časti Nefunkcionálne požiadavky sme definovali vlastnosti, ktorú musí aplikácia spĺňať.

### Funkcionálne požiadavky

Používateľ môže do aplikácie vložiť 1 alebo viacero vzoriek oleja v jednom behu načítania súborov z priečinka, ktorý si zvolil. Tieto vzorky musia byť vo formáte .txt alebo .csv. Používateľovi sa po načítaní zobrazí zoznam súborov, ktoré načítal. V prípade, že sa rozhodne nejakú vzorku nepoužiť, môže tento súbor zo zoznamu vymazať. Ak sa rozhodne nevyhodnotiť načítané vzorky, môže ich všetky naraz vymazať. Pri načítavaní nových vzoriek bude možné zvoliť priečinok, z ktorého bude možné dáta do aplikácie načítať.

Používateľ môže v aplikácií zobraziť vzorku oleja ako obrázok. Obrázok je vygenerovaný aplikáciou a reprezentuje hodnoty v spektrách namerané v čase.

Používateľ môže vyhodnotiť načítaný súbor, teda vzorku oleja, zvoleným klasifikátorom. Klasifikátor si používateľ môže zvoliť z ponuky. Ponuka musí obsahovať predtrénované klasifikátory a možnosť vybrať vlastný natrénovaný model.

Používateľ má možnosť dotrénovať model, ktorý bol už vopred natrénovaný, na nových dátach. Taktiež môže natrénovať nový model na nových dátach. Pri trénovaní a dotrénovaní modelu nie je možné meniť hyperparametre jednotlivých klasifikátorov. Načítané dáta sú rozdelené na trénovaciu a validačnú vzorku, pomer rozdelenia dát je pevne stanovený, teda používateľ ho nemôže zmeniť. Po natrénovaní alebo dotrénovaní modelu sa zobrazí úspešnosť modelu. Aby nedošlo k strate dát, model bude po vyhodnotení dotrénovaný aj na dátach, ktoré boli označené ako validačné.

Používateľ môže uložiť natrénovaný model do priečinka, ktorý si vyberie. Ukladanie modelu je podstatnou súčasťou, aby mohol byť tento model použitý aj pri vyhodnotení nových dát.

Používateľ môže uložiť predpracované súbory vo formáte .csv, aby ich pri ďalšom načítaní nemusel predspracovávať znova.

Používateľ môže uložiť obrázok vo formáte .png do priečinka, ktorý si zvolí.

Používateľ si vie zvoliť klasifikátor, ktorý vie identifikovať extra virgin olivový olej s tým, že ostatné triedy vyhodnocuje ako 1 triedu.

#### Nefunkcionálne požiadavky

Používateľ vie do aplikácie načítať naraz maximálne 50 súborov.

V prípade, že používateľ chce vložiť súbor, ktorý je v inom formáte ako .txt alebo .csv, tak systém nezlyhá. Ak táto situácia nastane, systém upozorní používateľa, že sa snaží vložiť súbor s nepodporovaným formátom.

Aplikáciu je možné používať na počítačoch s operačným systémom Windows 7 a vyššie.

Frekvencia zlyhania musí byť menšia ako 0,5%. To znamená, že napríklad z 1000 načítaní nových súborov sa súbor nenačíta maximálne 5-krát, resp. načítavanie súboru zlyhá v 5 prípadoch.

Aplikácia nekomunikuje alebo nespolupracuje s inými softvérmi. Systém spracováva dáta namerané pomocou spektrometra, ktoré sú uložené v súbore. To znamená, že spektrometer uloží dáta do súboru v priečinku a aplikácia tieto dáta načíta, ale priamo s ním nekomunikuje.

Používateľ musí vidieť možnosť vkladania nových dát a päť načítaných súborov na jednej obrazovke. Pri výbere klasifikátora sa všetky možnosti musia nachádzať na jednej

## 1.4 Opis dát

Od zadávateľa projektu sme dostali dáta vo formáte .txt. Každý súbor reprezentuje 1 vzorku olivového oleja. Dáta sú uložené v 3 priečinkoch podľa triedy oleja (EVOO, VOO, LOO), ktorá bola priradená danej vzorke. Každý priečinok obsahuje ďalšie priečinky, z ktorých každý reprezentuje 1 vzorku oleja a v nich sú uložené súbory s nameranými hodnotami pre daný olej. Takýto súbor je pomenovaný Spektra.POSITIVE.txt. Každá vzorka oleja obsahuje záznamy o spektrách a hodnotách, ktoré boli namerané v týchto spektrách. Počet spektier je v jednotlivých súboroch variabilný, od 9149 do 10607 spektier. V jednotlivých spektrách sa zaznamenávajú:

- hodnoty intenzita driftového poľa (ang. drift_field_intensity),
- tlak (ang. pressure),

- teplota (ang. temperature),
- dĺžka driftovej trubice (ang. drift_tube_length),
- čas analýzy (ang. analysis_time),
- čas, kedy bola nameraná intenzita,
- intenzity namerané v danom čase.

# 2 Návrh

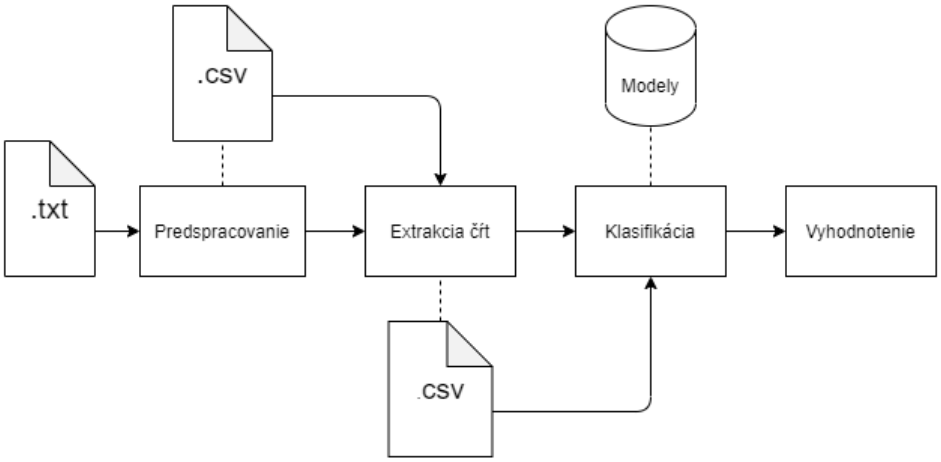
V tejto kapitole je opísané navrhované riešenie. Riešenie je rozdelené do dvoch častí.

V prvej časti Klasifikácia opisujeme navrhované moduly systému, spôsoby predspracovania dát, navrhované črty, trénovanie neurónových a tradičných klasifikátorov a spôsoby vyhodnocovania .

V druhej časti Aplikácia popisujeme návrh aplikácie, ako napríklad prípady použitia navrhovaného softvéru.

## 2.1 Návrh metódy

Náš systém identifikácie tried oleja je založený na 4 väčších moduloch (Predspracovanie, Extrakcia čŕt, Klasifikácia, Vyhodnotenie), ktoré sú znázornené na Obrázku 6. Vstupom sú surové dáta z .txt súboru. Tieto dáta vstupujú do modulu predspracovanie, čoho výstupom je .csv súbor. Tento súbor následne vstupuje do modulu extrakcia čŕt, kde sú vytvorené jednotlivé črty a tie sú uložené do nového súboru .csv. Novovzniknuté dáta sú popísané v časti Opis dát. Takto upravené dáta už vstupujú do klasifikácie, kde sa trénujú klasifikátory a vytvárajú modely. Následne sú modely vyhodnotené.




*Obr. 6: Návrh metódy pre klasifikáciu kvality oleja.*

## 2.2 Predspracovanie dát

V procese predspracovania dát sa bude konvertovať .txt súbor do .csv súboru, pre lepšiu manipuláciu s dátami. Vo formáte .csv bude spektrum reprezentované ako jeden riadok s príslušnými hodnotami.

V tejto časti budeme identifikovať chýbajúce hodnoty alebo chybné hodnoty a upravovať nesúlad v dátach. Taktiež sa vykoná normalizácia dát. Normalizácia transformuje dáta do normálneho rozdelenia, aby sme zmenšili rozptyl dát. Napríklad všetky hodnoty budú z rozsahu 0 – 1. Takéto rozdelenie požadujú napríklad klasifikátory *k*-NN a SVM, keďže počítajú so vzdialenosťou a podobnosťou medzi dátami.

### 2.2.1 Extrakcia čŕt

Po predspracovaní dát a dôkladnej analýze vzťahov medzi črtami v zozbieraných dátach, vyberieme črty, ktoré budú pre nás dôležité a pomôžu nám v identifikácii tried. Najprv budeme vyberať črty ako priemer, maximum, minimum, medián, štandardná odchýlka pre všetky stĺpce a neskôr sa doimplementujú aj zložitejšie črty.

Pri extrakcii sa najskôr vytvorí samostatná tabuľka so základnými údajmi a do nej sa budú postupne pridávať nové črty. Teda po vytvorení základnej tabuľky a pridaní čŕt sa nebudú musieť počítať všetky črty od začiatku, ak budeme chcieť pridať novú črtu.

### 2.2.2 Klasifikácia

Klasifikácia sa bude vykonávať pomocou piatich tradičných klasifikačných algoritmov a pomocou neurónových sietí. Tieto klasifikátory boli vybrané aj na základe podobných prác, kde dosiahli výborné výsledky v danej oblasti. Tieto práce sú popísané v časti Existujúce riešenia. Vybrané klasifikačné algoritmy sú:

- *k*-NN
- SVM
- Decision Tree
- Random Forest
- Naive Bayes
- Artificial Neural Networks

Pred trénovaním budeme deliť dáta na testovacie a trénovacie v určitých pomeroch a z nich vyberieme jeden najefektívnejší. Taktiež skúsime aplikovať krížovú validáciu.

Pri trénovaní klasifikátorov budeme ladiť hyperparametre a snažiť sa dosiahnuť čo najlepšie výsledky pomocou rôznych nastavení.

### 2.2.3 Vyhodnotenie

Pre každý klasifikátor vyhodnotíme ich skóre úspešnosti pomocou metrík na vyhodnocovanie úspešnosti kvalifikátorov. Pre každý klasifikátor vytvoríme obrázok matice zámen a taktiež vytvoríme jeden graf, na ktorom budú porovnané výsledky všetkých klasifikátorov.

V prípade, ak sú triedy nevyvážené, samotná správnosť (angl. *accuracy*) môže byť zavádzajúca. Preto sa používa viacero metrík, ktoré rôzne opisujú úspešnosť klasifikátora. Z matice zámen (Tabuľka 1), možno odvodiť viacero metrík (Tabuľka 2).

*Tabuľka 1: Matica zámen pre dve triedy. [1]*

| | Predikovaná pozitívna trieda | Predikovaná negatívna trieda |
|---|---|---|
| Pozitívna trieda | TP | FN |
| Negatívna trieda | FP | TN |

- TP – pravdivé pozitíva (angl. *true positive*), predstavujú počet správne označených vzoriek, ktoré do triedy patria [1].
- TN – pravdivé negatíva (angl. *true negative*), predstavujú počet správne označených vzoriek, ktoré do triedy nepatria [1].
- FP – falošné pozitíva (angl. *false positive*), predstavujú počet vzoriek, ktoré boli nesprávne klasifikované do triedy, do ktorej nepatria [1].
- FN – falošné negatíva (angl. *false negative*), predstavujú počet vzoriek, ktoré boli označené, že do triedy nepatria, ale do triedy patrili [1].

*Tabuľka 2: Metriky hodnotiace úspešnosť klasifikátorov odvodené z matice zámen. [1]*

| **Metrika** | **Vzorec** |
|---|---|
| Správnosť (acc) | $\frac{TP+TN}{TP+TN+FP+FN}$ |
| Presnosť (pre) | $\frac{TP}{TP+FP}$ |
| Úplnosť (rec) | $\frac{TP}{TP+FN}$ |
| F1 skóre | $\frac{2.pre.rec}{pre+rec}$ |

V našom prípade máme tri triedy (EVOO, LOO, VOO), čiže sa jedná o klasifikáciu do viacerých tried. V tomto prípade použijeme metriky pre vyhodnotenie úspešnosti klasifikátora pri viacerých triedach (počítame mikro- a makro-priemer metrík)

$$B_{macro}\frac{1}{q}\sum_{\lambda}^{q} B(TP_\lambda, FP_\lambda, TN_\lambda, FN_\lambda) \tag{1}$$

$$B_{micro} = B(\sum_{\lambda}^{q} TP_\lambda \sum_{\lambda}^{q} FP_\lambda \sum_{\lambda}^{q} TN_\lambda \sum_{\lambda}^{q} FN_\lambda) \tag{2}$$

- B – binárna vyhodnocovacia metrika ako napr. presnosť, úplnosť, $F_1$ skóre
- q – počet tried
- $\lambda$ – trieda

$B_{macro}$ vypočítava metriku zvlášť pre každú triedu a potom určí ich priemer. $B_{micro}$ agreguje výsledky všetkých tried na výpočet priemernej metriky [4].

## 2.3 Aplikácia

V tejto kapitole si zadefinujeme prípady použitia, ktoré boli vytvorené po konzultácii s PO, ale rovnako tak s celým tímom.

### 2.3.1 Prípady použitia

#### UC01 Načítanie dát

**Opis**
V tomto prípade použitia je potrebné vložiť dáta do aplikácie a tie budú postupne spracované do požadovaného .csv formátu. Tento prípad použitia sa môže opakovať, nakoľko vieme pridať viacero súborov.

**Predpoklady**
Používateľ má nainštalovanú a spustenú aplikáciu.

**Dôsledky**
Dáta sú pripravené na klasifikáciu a ďalšie operácie s nimi spojené.

**Hlavný scenár**

1. Používateľ v aplikácií zvolí možnosť načítať dáta.
2. Aplikácia zobrazí okno výberu súboru.
3. Používateľ vyberie vstupný súbor z adresára a potvrdí výber.
4. Aplikácia overí, či je to .txt alebo .csv súbor a overí, či je validný jeho obsah.
5. Aplikácia súbor .txt odošle na spracovanie a zobrazí hlášku, že to môže chvíľu trvať.
6. Po spracovaní dát aplikácia zobrazí hlášku, že dáta boli spracované. [Uloženie .csv súborov] [Vymazanie .csv súboru]
7. Aplikácia odblokuje ďalšie tlačidlá.
8. Prípad použitia končí.

| Od kroku | Po krok | Alternatívny scenár |
|---|---|---|
| 5 | 7 | AS01 Načítanie .csv súboru |

**Alternatívne scenáre**

AS01 Načítanie .csv súboru

1. Po kroku 4 UC01 Načítanie dát aplikácia potvrdí validne .csv.
2. Scenár pokračuje krokom 7 UC01 Načítanie dát.

| Od kroku | Po krok | Alternatívny scenár |
|---|---|---|
| 2 | - | AS02 Neschválenie .csv súboru |

AS02 Neschválenie .csv súboru

1. Po kroku 1 AS01 Načítanie .csv súboru zistíme, že .csv súbor nie je validný.
2. Aplikácia zobrazí hlášku, že súbor nie je validný.
3. Prípad použitia pokračuje krokom 2 UC01 Načítanie dát.

**UC02 Klasifikácia dát**

**Opis**

Prípad použitia je zameraný a vyhodnotenie predspracovaných dát.

**Predpoklady**

Aplikácia má predspracované dáta pomocou UC01 Načítanie dát a máme natrénované modely.

**Dôsledky**

Používateľ vie do akej triedy patria jeho vzorky.

**Hlavný scenár**

1. Používateľ zvolí možnosť klasifikácie dát.
2. Aplikácia zobrazí zoznam natrénovaných klasifikátorov.
3. Používateľ vyberie 1 alebo viacero klasifikátorov.

4. Aplikácia vyhodnotí vzorky dát zvolenými klasifikátormi.
5. Aplikácia zobrazí klikateľný zoznam vzoriek z dát. [Zobrazenie detailu vzorky] [Uloženie výsledku vyhodnotenia]
6. Prípad použitia končí.

#### UC03 Zobrazenie dát z obrázka

**Opis**
Prípad použitia má za úlohu transformovať textové dáta do podoby obrázkov.

**Predpoklady**
Aplikácia má predspracované dáta pomocou UC01 Načítanie dát.

**Dôsledky**
Vizualizácia dát pomocou obrázkov.

**Hlavný scenár**

1. Používateľ zvolí možnosť zobrazenia dát v obrázku.
2. Aplikácia odošle súbor na transformáciu na obrázok a zobrazí hlášku, že to môže chvíľu trvať.
3. Po transformácii aplikácia zobrazí klikateľný zoznam obrázkov z dát.
4. Aplikácia zobrazí detail vzorky. [Zobrazenie detailu obrázka][Uloženie obrázka]
5. Prípad použitia končí.

#### UC04 Trénovanie modelu

**Opis**
Prípad použitia má na starosť vytvoriť vlastné modely z predspracovaných dát.

**Predpoklady**
Aplikácia má predspracované dáta pomocou UC01 Načítanie dát.

**Dôsledky**

Používateľ má vytvorený vlastný model, na ktorom bude môcť klasifikovať nové údaje.

**Hlavný scenár**

1. Používateľ zvolí možnosť natrénovania si vlastného modelu.
2. Aplikácia zobrazí zoznam modelov, ktoré vieme natrénovať.
3. Používateľ zvolí modely, ktoré chce trénovať.
4. Aplikácia natrénuje modely.
5. Aplikácia zavolá UC05 Dotrénovanie modelov.
6. Aplikácia zobrazí klikateľný zoznam netrénovaných modelov. [Zobrazenie detailu netrénovaného modelu]
7. Prípad použitia končí.

| Od kroku | Po krok | Alternatívny scenár |
|---|---|---|
| 3 | 6 | AS01 Optimalizácia výberu |

**Alternatívne scenáre**

AS01 Optimalizácia výberu

1. Používateľ vyberie možnosť optimalizovať hyperparametre.
2. Aplikácia natrénuje viacero kombinácií hyperparametrov pre jeden model a vyhodnotí, ktorý je najlepší. Na ten zavolá UC05 Dotrénovanie modelov.
3. Prípad použitia pokračuje bodom 6 UC04 Trénovanie modelu.

**UC05 Dotrénovanie modelu**

**Opis**

Prípad použitia má na starosť dotrénovať existujúce modely z predspracovaných dát.

**Predpoklady**

Aplikácia má predspracované dáta pomocou UC01 Načítanie dát a natrénované modely

uložené v aplikácii.

**Dôsledky**
Používateľ má dotrénuje existujúci model s novými dátami. Klasifikátor by mal dosahovať presnejšie výsledky.

**Hlavný scenár**

1. Používateľ zvolí možnosť dotrénovania existujúceho modelu.
2. Aplikácia zobrazí zoznam modelov, ktoré chceme dotrénovať.
3. Používateľ zvolí modely, ktoré chce dotrénovať.
4. Aplikácia dotrénuje, uloží modely a zobrazí hlášku, že modely sú dotrénované. [Uloženie modelu]
5. Prípad použitia končí.

| Od kroku | Po krok | Alternatívny scenár |
|---|---|---|
| 1 | 4 | AS01 Dotrénovanie modelu aplikáciou |

**Alternatívne scenáre**
AS01 Dotrénovanie modelu aplikáciou

1. Aplikácia zvolí možnosť dotrénovania existujúceho modelu.
2. Aplikácia zvolí modely, ktoré chce dotrénovať.
3. Prípad použitia pokračuje bodom 4 UC05 Dotrénovanie modelu.

**UC06 Ukladanie súborov**

**Opis**
Prípad použitia má na starosť ukladanie súborov.

**Predpoklady**
Máme na výber súbory, ktoré vieme uložiť.

**Dôsledky**
Súbory sú uložené na lokálnom počítači.

**Body rozšírenia**
Uloženie csv súborov.
Uloženie obrázka.
Uloženie detailu.
Uloženie výsledku vyhodnotenia.
Uloženie modelu.

**Hlavný scenár**

1. Používateľ zvolí možnosť uloženia súborov.
2. Aplikácia otvorí okno s možnosťou výberu adresára, kde budú súbory uložené.
3. Používateľ vyberie adresár, kam majú byť uložené dáta.
4. Aplikácia súbory uloží.
5. Prípad použitia končí.

**UC07 Vymazanie súborov**

**Opis**
Používateľ viem vymazať súbory, ktoré pridal do aplikácie.

**Predpoklady**
Máme načítané súbory pomocou UC01 Načítanie dát, ktoré chceme odstrániť.

**Dôsledky**
Súbory nie sú súčasťou aplikácie.

**Body rozšírenia**
Vymazanie .csv súboru.

**Hlavný scenár**

1. Používateľ zvolí možnosť vymazať súbor.
2. Aplikácia sa opýta na potvrdenie rozhodnutia.
3. Používateľ potvrdí vymazanie.
4. Aplikácia vymaže súbory.
5. Prípad použitia končí.

**UC08 Zobrazenie detailu**

**Opis**
Prípad použitia zahŕňa zobrazenie detailu po vybratí položky zo zoznamu.

**Predpoklady**
Zobrazená zoznam prvkov, pre ktoré vieme zobraziť detail.

**Dôsledky**
Zobrazenie detailu prvku.

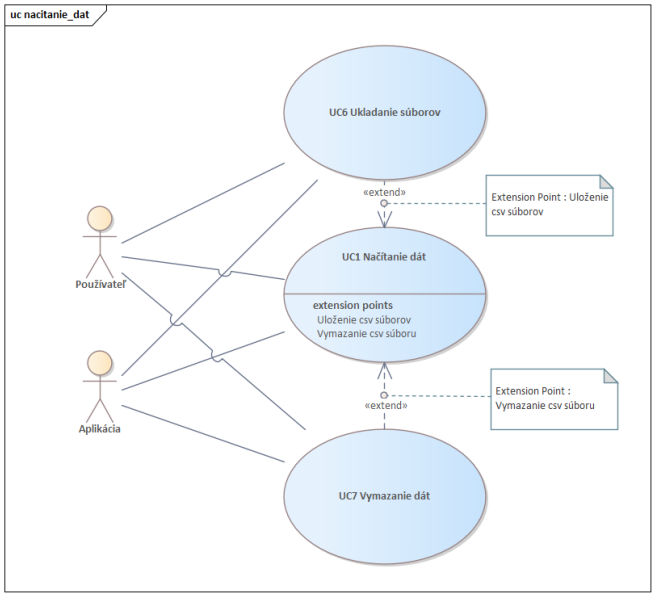
**Body rozšírenia**
Zobrazenie detailu vzorky.
Zobrazenie detailu obrázka.
Zobrazenie detailu netrénovaného modelu.

**Hlavný scenár**

1. Používateľ zvolí možnosť zobraziť detail.
2. Aplikácia zobrazí detail. [Uloženie detailu]
3. Prípad použitia končí.

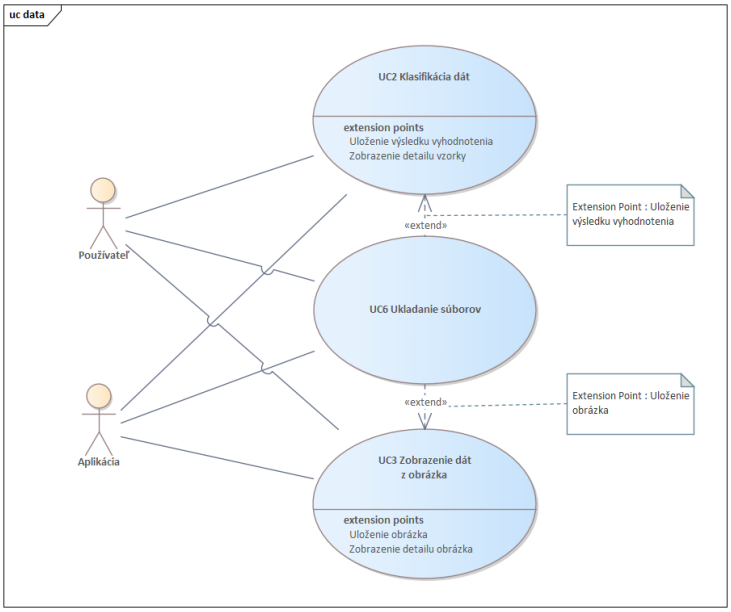
### 2.3.2 Závislosti medzi prípadmi použitia

Obrázok 7 reprezentuje načítavanie údajov do aplikácie. Súbory, ktoré sú načítané, môžu byť z aplikácie aj uložené a odstránené.



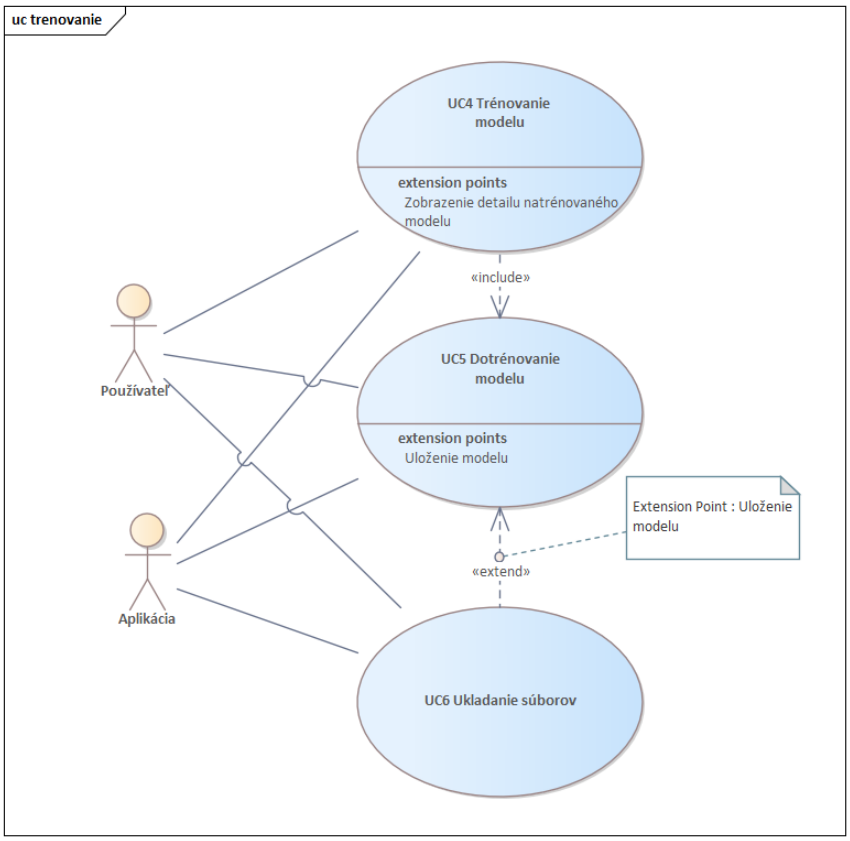

*Obr. 7: Diagram, ktorý ukazuje závislosť medzi prípadmi použitia pri načítavaní dát do aplikácie.*

Obrázok 8 reprezentuje klasifikovanie dát do jednotlivých tried. Taktiež je načítané dáta možné zobrazovať vo forme obrázka, ktorý je možné uložiť.



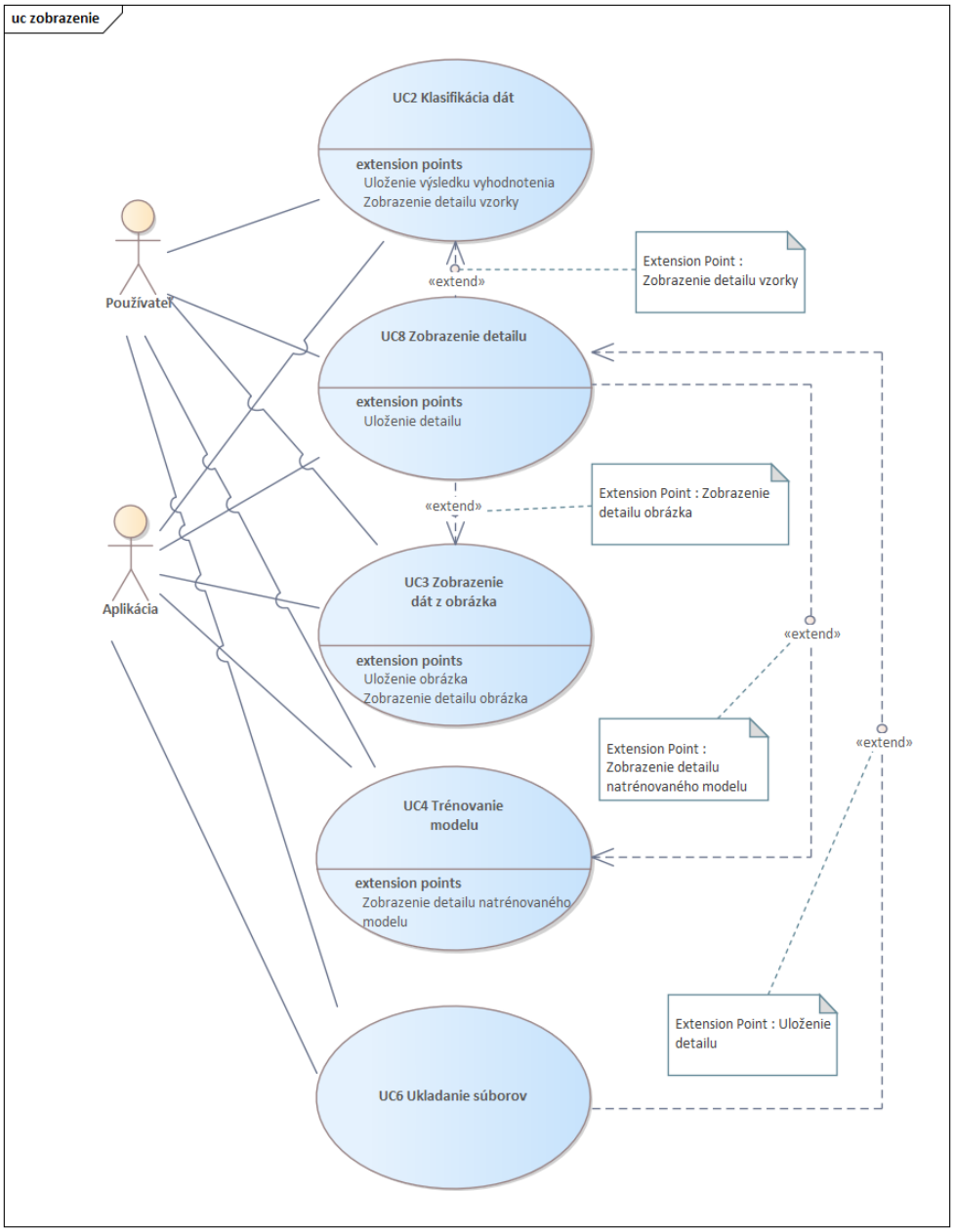

*Obr. 8: Diagram, ktorý ukazuje závislosť medzi prípadmi použitia, ktoré zodpovedajú za klasifikovanie dát a zobrazovanie obrázka z dát.*

Obrázok 9 reprezentuje možnosť vytvorenia nových modelov, ktoré vzniknú natrénovaním modelov na nových dátach. Zároveň je možné dotrénovať existujúce modely na nových dátach. V oboch prípadoch vzniknú nové modely.




*Obr. 9: Diagram, ktorý ukazuje závislosť medzi prípadmi použitia pri trénovaní klasifikačných metód.*

Obrázok 10 ukazuje, čo všetko je možné zobrazovať pomocou aplikácie.




*Obr. 10: Diagram, ktorý ukazuje závislosť medzi prípadmi použitia pri ukladaní dát v aplikácii.*

# 3 Implementácia

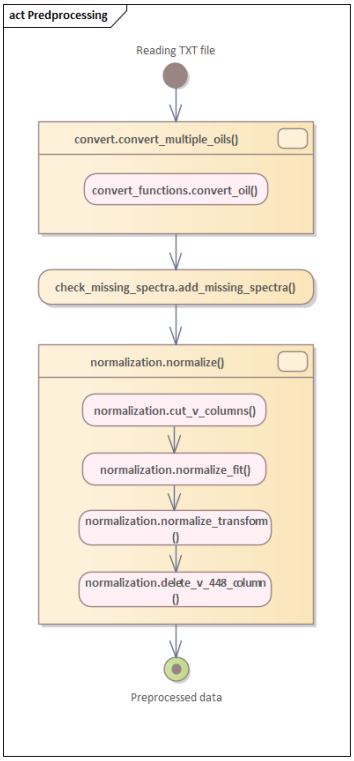
V tejto kapitole sú opísané postupy implementácie predspracovania, jednotlivých klasifikátorov a vyhodnocovania úspešnosti nami navrhnutých klasifikátorov. Zároveň sú v tejto kapitole opísané aj všetky súbory, ktoré sme vytvorili pri spracovávaní dát.

## 3.1 Opis implementácie

Dokumentácia implementácie je zverejnená v dokumente prílohe A. V tejto kapitole sa nachádzajú diagramy, ktoré zobrazujú jednotlivé kroky implementácie. Funkcie z diagramov sú bližšie popísané v spomínanej dokumentácii.

### 3.1.1 Predspracovanie dát

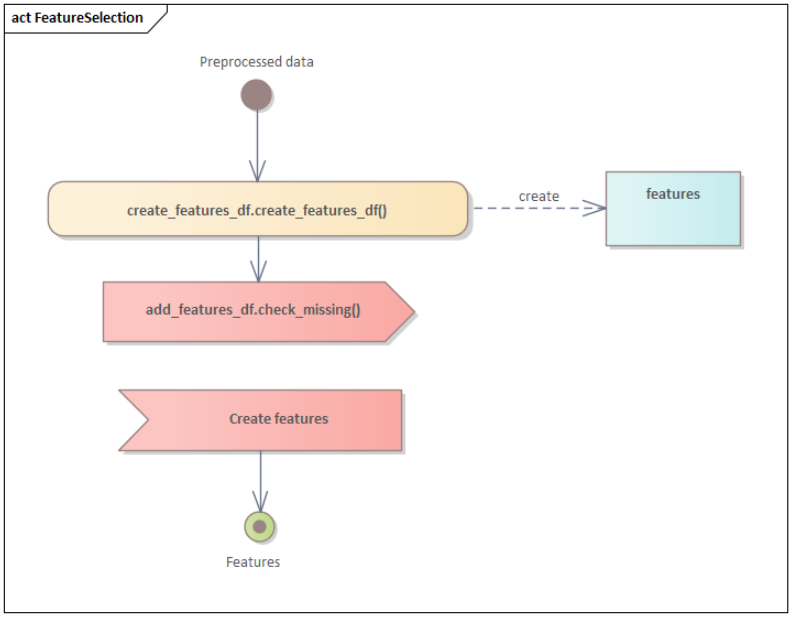
Obrázok 11 zobrazuje priebeh volania funkcií pri predspracovaní dát.




*Obr. 11: Ukážka priebehu predspracovania dát.*

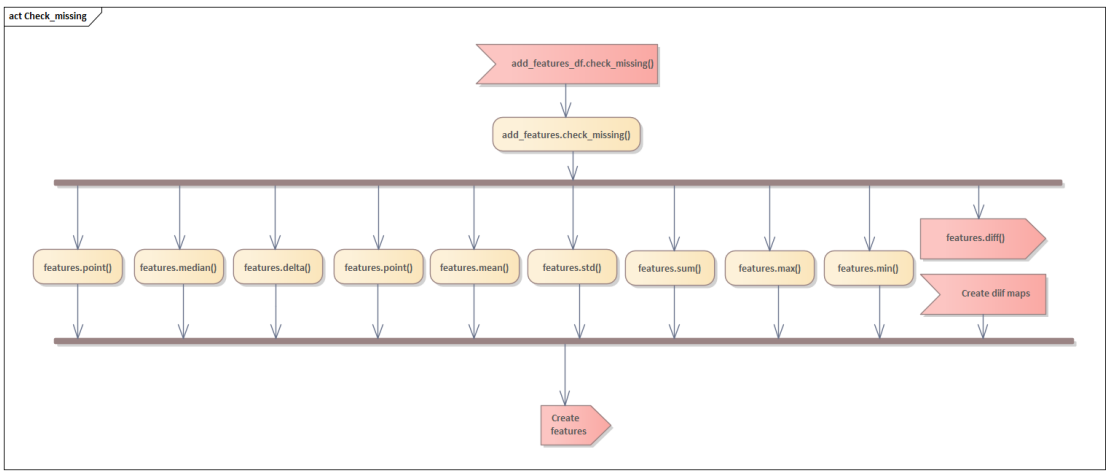
### 3.1.2 Črty

Obrázok 12 zobrazuje priebeh vytvárania čŕt z predspracovaných dát.



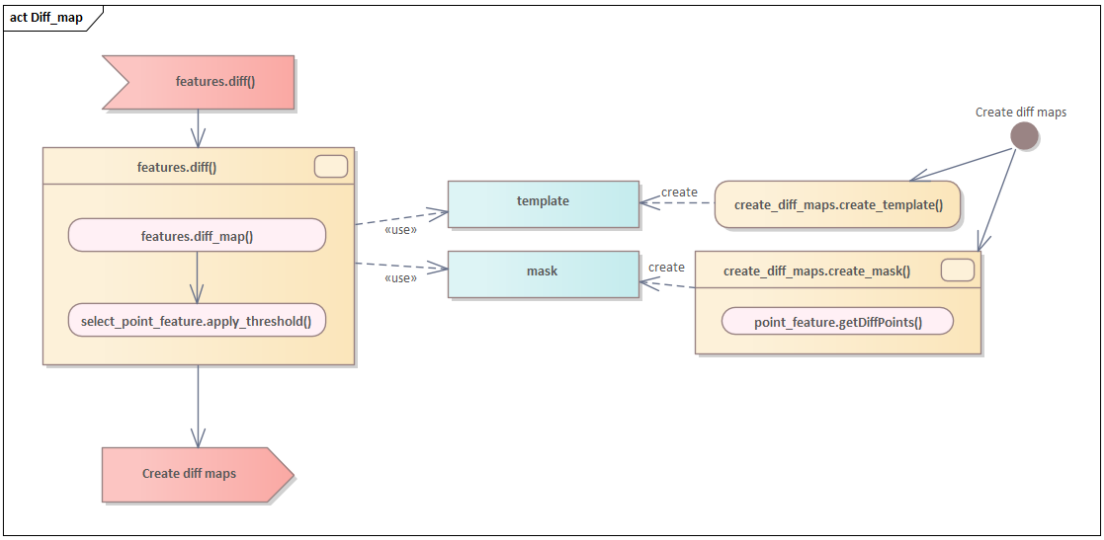

*Obr. 12: Ukážka priebehu vytvárania čŕt.*

Obrázok 13 zobrazuje kontrolu, či sú vytvorené všetky črty. V prípade, že črty chýbajú, zavolajú sa funkcie, ktoré chýbajúce črty vytvoria.



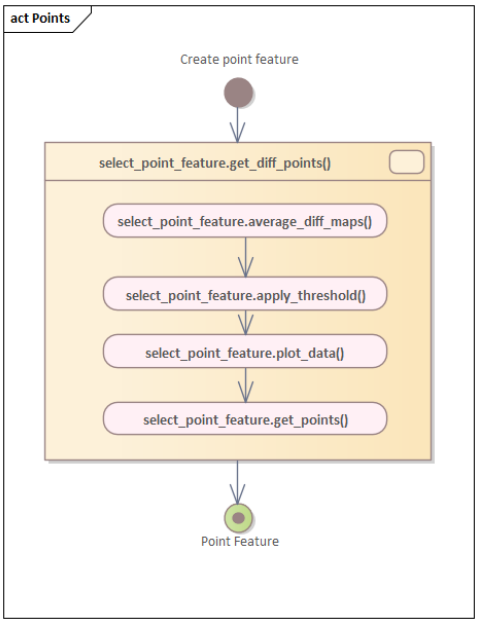

*Obr. 13: Ukážka vytvorenia chýbajúcich čŕt.*

Obrázok 14 zobrazuje priebeh vytvorenia diferenčnej mapy, ktorá je neskôr využitá ako črta pri klasifikácií.



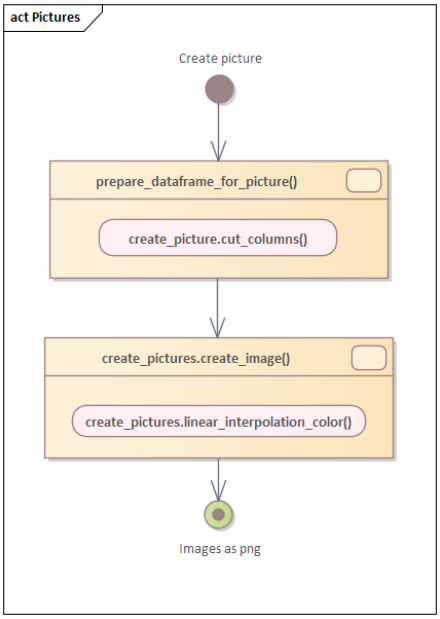

*Obr. 14: Ukážka vytvorenia diferenčnej mapy.*

Obrázok 15 zobrazuje priebeh vytvorenia črty Point.




*Obr. 15: Ukážka vytvorenia črty Point.*

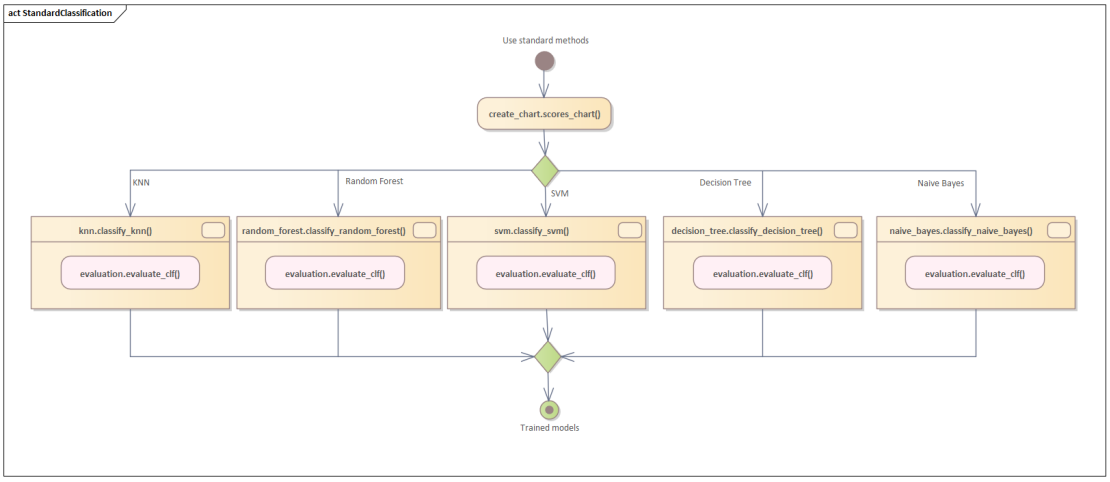
Obrázok 16 zobrazuje generovanie obrázkov z predspracovaných dát.




*Obr. 16: Ukážka generovania obrázkov z predspracovaných dát.*

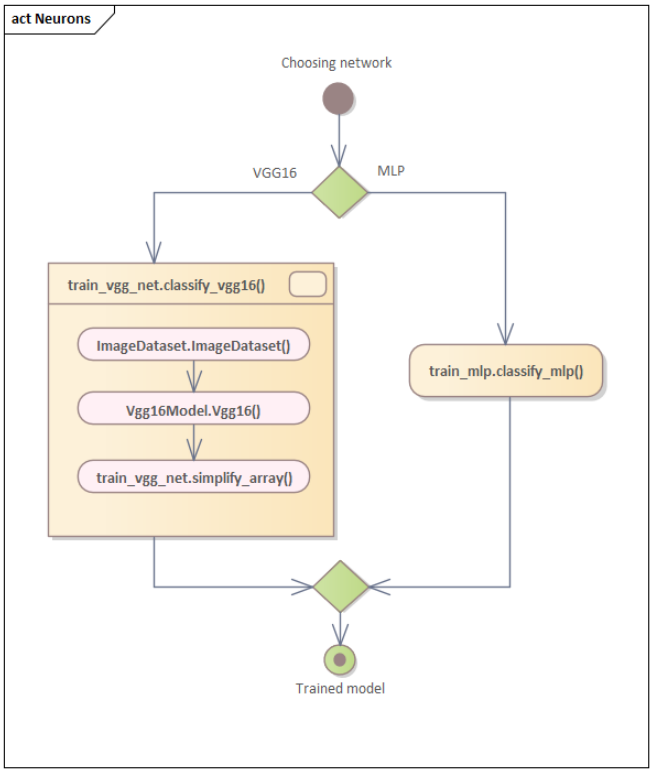
### 3.1.3 Klasifikácia

Obrázok 17 zobrazuje ako sú volané štandardné metódy klasifikácie počas trénovania modelov.




*Obr. 17: Ukážka trénovania štandardných klasifikátorov.*

Obrázok 18 zobrazuje ako sú volané trénované modely neurónových sietí.




*Obr. 18: Ukážka trénovania neurónových sietí.*

## 3.2 Opis dát

### Spracované dáta

Súbory, ktoré sme dostali na začiatku sme spracovali do 3 súborov – dataEVOO.csv, dataVOO.csv, dataLOO.csv. Tieto súbory obsahujú záznamy o všetkých olejoch z jednotlivých tried. Súbory obsahujú stĺpce class, oil_id, spectrum, drift_field_intensity, pressure, temperature, drift_tube_length, analysis_time, t_0 – t_448, v_0 – v_448:

- stĺpec class definuje triedu oleja,
- oil_id reprezentuje vzorku oleja,
- spectrum identifikuje číslo spektra, ktorého namerané hodnoty sú uložené v riadku,
- drift_field_intensity, pressure, temperature, drift_tube_length, analysis_time sú stĺpce, v ktorých sú zaznamenané hodnoty namerané pre jednotlivé spektrá olejov,

- t_0 – t_448 je 449 stĺpcov, v ktorých sú uložené časy, v ktorých boli namerané intenzity,
- v_0 – v_448 je 449 stĺpcov, v ktorých sú uložené namerané intenzity v spektrách.

| | class | oil_id | spectrum | drift_field_intensity | pressure | temperature | drift_tube_length | analysis_time | t_0 | t_1 | ... | v_439 | v_440 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | EVOO | 200427_174846 | 1 | 558.521 | 994.85 | 110.292 | 11.51 | 0.13 | 28.83505 | 27.20288 | ... | 0.037 | 0.031 |
| 1 | EVOO | 200427_174846 | 2 | 558.521 | 994.85 | 110.292 | 11.51 | 0.26 | 28.83505 | 27.20288 | ... | 0.036 | 0.032 |
| 2 | EVOO | 200427_174846 | 3 | 558.521 | 994.85 | 110.370 | 11.51 | 0.39 | 28.83505 | 27.20288 | ... | 0.036 | 0.033 |
| 3 | EVOO | 200427_174846 | 4 | 558.521 | 994.85 | 110.370 | 11.51 | 0.52 | 28.82917 | 27.19733 | ... | 0.038 | 0.031 |
| 4 | EVOO | 200427_174846 | 5 | 558.521 | 994.85 | 110.370 | 11.51 | 0.65 | 28.82917 | 27.19733 | ... | 0.038 | 0.030 |

5 rows × 906 columns

*Obr. 19: Ukážka súboru dataEVOO.csv.*

Riadok v každom súbore obsahuje namerané dáta z jedného spektra, preto sme pridali do súboru stĺpec oil_id, ktorý identifikuje o aký olej ide, pretože 1 vzorka oleja obsahuje dáta o viacerých spektrách. V súboroch sa nenachádzajú chýbajúce spektrá, teda každá vzorka oleja obsahuje všetky spektrá.

Súbory dataEVOO.csv, dataVOO.csv a dataLOO.csv sme spojili do 1 súboru a vytvorili sme dataAll.csv, ktorý má rovnaké stĺpce ako predošlé súbory.

V rámci testovania funkcionality softvéru sme vytvorili 3 testovacie súbory – data1.csv, data 3x1.csv, data3x5.csv. Všetky 3 súbory majú rovnaké stĺpce ako predošlé súbory. Súbor data1.csv obsahuje namerané hodnoty pre 1 olej, data3x1.csv obsahuje hodnoty pre 1 olej z každej triedy a data3x5.csv obsahuje hodnoty pre 5 olejov z každej triedy.

### Črty

Súbor featuresAll.csv obsahuje niekoľko čŕt, ktoré sme vytvorili počas prvých 2 šprintov. Črty boli vypočítané zo stĺpcov, ktoré sme identifikovali ako podstatné pre klasifikátor. Z daných stĺpcov sme postupne vypočítali:

- priemer pre každý olej,
- priemernú hodnotu pre každý stĺpec v danej vzorke oleja,
- maximum pre každý stĺpec v danej vzorke oleja,
- minimum pre každý stĺpec v danej vzorke oleja,
- medián pre každý stĺpec v danej vzorke oleja,

- štandardnú odchýlku pre každý stĺpec v danej vzorke oleja,
- deltu pre každý stĺpec v danej vzorke oleja (maximum - minimum),
- súčet hodnôt pre každý stĺpec v danej vzorke oleja.

V rámci čŕt sme vytvárali aj diferenčné mapy, ktoré reprezentujú rozdiel medzi 2 porovnávanými vzorkami oleja. V jednotlivých olejoch porovnávame rozdiel hodnôt vo vybraných stĺpcoch. Porovnávali sme 2 oleje z rovnakej triedy, to znamená, vytvorili sme 5 súborov pre každú triedu, ktoré porovnávajú 2 rôzne oleje. Zároveň sme porovnávali aj oleje z 2 rôznych tried (EVOO-VOO, EVOO-LOO, LOO-VOO). Opäť sme vytvorili 5 súborov pre porovnanie z každej dvojice.

Aby sme mohli vyskúšať konvolučné neurónové siete, vytvorili sme obrázky z vybraných stĺpcov. Tieto obrázky sú vo formáte .png a boli vytvorené pre všetky oleje z každej triedy. Obrázky majú veľkosť počet_spektrierX449 pixlov. Hodnoty sú reprezentované odtieňmi modrej a červene farby, podľa jej veľkosti. Modrá reprezentuje nízke hodnoty, červená vysoké, na prechod medzi farbami sme využili lineárnu interpoláciu.

# 4 Testovanie

V tejto kapitole sú stručne opísané testy. Na validáciu navrhnutého softvéru používame akceptačné testy a pre zamedzenie chýb v programe sú používané jednotkové testy (ďalej ako unit testy).

## 4.1 Unit testy

Hlavným účelom unit testov je odhaľovanie chýb vo vytvorených funkciách. Pri testovaní unit testami sa porovnáva očakávaný výsledok funkcie s vráteným výsledkom.

V rámci klasifikácie sú týmto spôsobom testované funkcie na spracovanie dát. Cieľom je odhaliť chyby a spraviť systém robustným voči neočakávaným vstupom.

## 4.2 Akceptačné testy

Aby sme aplikáciu mohli odovzdať product owner-ovi, potrebuje mu ukázať, že sme vytvorili požadovanú aplikáciu. Na túto prezentáciu použijeme akceptačné testy, ktoré sme si rozdelili do 2 kategórií.

Prvá kategória popisuje najjednoduchšie testy, ktoré je možné vykonať pozorovaním.

- Aplikácia nesmie obsahovať gramatické chyby.
- Dizajn obrazoviek vyzerá ako v návrhu.

V druhej kategórií sú zadefinované testy, ktoré overujú funkčnosť aplikácie.

- Používateľ vie vložiť súbor vo formáte .txt a .csv.
- Používateľ nevie vložiť iný súbor ako .txt alebo .csv.
- Používateľ dokáže načítať 50 súborov do aplikácie.
- Používateľ vie vymazať načítaný súbor.
- Používateľ vie zobraziť obrázok z načítaného súboru.
- Používateľ vie klasifikovať vzorku oleja so zvoleným klasifikátorom.
- Používateľ vie natrénovať nový model.
- Používateľ vie uložiť obrázok.
- Používateľ vie uložiť predspracované dáta.

- Používateľ vie uložiť natrénovaný model.
- Používateľ vie načítať vlastný model na trénovanie.
- Používateľ vie vyhodnotiť iba triedu EVOO
- Používateľ vie uložiť vyhodnotené dáta.

**Kroky a očakávaný výstup akceptačných testov**

Vloženie súborov .txt a .csv

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Vložte súbor test.csv do aplikácie.
4. **Očakávaný výstup** – Súbor test.txt a test.csv bude vložený do aplikácie.

Vloženie súboru s inou príponou ako .txt a .csv.

1. Otvorte si aplikáciu.
2. Vložte súbor test.pdf do aplikácie.
3. **Očakávaný výstup** – Súbor test.pdf nebude vložený do aplikácie. Aplikácia upozorní používateľa, že sa snaží vložiť súbor v nepodporovanom formáte.

Používateľ dokáže načítať 50 súborov do aplikácie.

1. Otvorte si aplikáciu.
2. Vložte všetky súbory z priečinka test50 do aplikácie.
3. **Očakávaný výstup** – úbory z priečinka budú načítané do aplikácie.

Používateľ vie vymazať načítaný súbor.

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Vymažte súbor test.txt.
4. **Očakávaný výstup** – Súbor test.txt bude odstránený z aplikácie.

Používateľ vie zobraziť obrázok z načítaného súboru.

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Zobrazte obrázok z dát načítaného súboru.
4. **Očakávaný výstup** – Zobrazí sa obrázok, ktorý reprezentuje načítané dáta.

Používateľ vie klasifikovať vzorku oleja so zvoleným klasifikátorom.

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Vyberte možnosť klasifikácie dát.
4. Vyberte si 1 z ponúkaných klasifikátorov.
5. Vyhodnoťte triedu oleja.
6. **Očakávaný výstup** – Používateľovi sa zobrazí predpokladaná trieda oleja.

Používateľ vie natrénovať nový model.

1. Otvorte si aplikáciu.
2. Vložte všetky súbory z priečinka test50 do aplikácie.
3. Vyberte možnosť natrénovania modelu.
4. Vyberte si 1 z ponúkaných klasifikátorov.
5. Natrénujte model.
6. **Očakávaný výstup** – Používateľovi sa zobrazí úspešnosť modelu, ktorý bol natrénovaný na nových dátach.

Používateľ vie uložiť obrázok.

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Zobrazte obrázok z dát načítaného súboru.
4. Uložte načítaný obrázok ako test_obrazok.png do priečinku test.

5. **Očakávaný výstup** – Uložený súbor test_obrazok.png v priečinku test.

Používateľ vie uložiť predspracované dáta.

1. Otvorte si aplikáciu.
2. Vložte súbor test.txt do aplikácie.
3. Vyberte možnosť predspracovania dát.
4. Uložte predspracované dáta ako súbor predspracovane.csv do priečinku test.
5. **Očakávaný výstup** – Uložený súbor predspracovane.csv v priečinku test.

Používateľ vie uložiť natrénovaný model.

1. Otvorte si aplikáciu.
2. Vložte všetky súbory z priečinka test50 do aplikácie.
3. Vyberte možnosť natrénovania modelu.
4. Vyberte si 1 z ponúkaných klasifikátorov.
5. Natrénujte model.
6. Uložte natrénovaný model ako test_model.joblib do súboru test.
7. **Očakávaný výstup** – Uložený súbor test_model.joblib v priečinku test.

Používateľ vie uložiť natrénovaný model.

1. Otvorte si aplikáciu.
2. Vložte všetky súbor z priečinka test50 do aplikácie.
3. Vyberte možnosť natrénovania modelu.
4. Vyberte možnosť vlastného klasifikátora.
5. Zvoľte klasifikátor test_model.joblib z priečinka test.
6. Vyhodnoťte dáta.
7. **Očakávaný výstup** – Načítaný klasifikátor dokáže vyhodnotiť triedu oleja.

Vyhodnotenie extra virgin olivového oleja.

1. Otvorte si aplikáciu.

2. Vložte súbor test.txt do aplikácie.
3. Vyberte možnosť vyhodnotenia triedy EVOO.
4. Vyberte si 1 z ponúkaných klasifikátorov.
5. Vyhodnoťte triedu oleja.
6. **Očakávaný výstup** – Používateľovi sa zobrazí predpokladaná trieda oleja, ktorá je buď EVOO alebo INÉ. Na výstupe nesmieme dostať tried VOO alebo LOO.

Vyhodnotenie extra virgin olivového oleja.

1. Otvorte si aplikáciu.
2. Vložte všetky súbory z priečinka test50 do aplikácie.
3. Vyberte možnosť klasifikácie dát.
4. Vyberte si 1 z ponúkaných klasifikátorov.
5. Vyhodnoťte triedy oleja.
6. Uložte vyhodnotené vzorky do súboru test_vyhodnotenie.csv, ktorý sa bude nachádzať v priečinku test.
7. **Očakávaný výstup** – Uložený súbor test_vyhodnotenie.csv v priečinku test. Dáta budú v súbore uložené vo forme „názov_súboru trieda“.

# Literatúra


[1] S. Agarwal. Data mining: Data mining concepts and techniques. In *2013 International Conference on Machine Intelligence and Research Advancement*, pages 203–207, 2013.

[2] Dasa Kruzlcova, Jan Mocak, Evagelos Katsoyannos, and Ernst Lankmayr. Classification and characterization of olive oils by uv-vis absorption spectrometry and sensorial analysis. *Journal of food and nutrition research*, 47:181–188, 01 2008.

[3] Emre Ordukaya and Bekir Karlik. Quality control of olive oils using machine learning and electronic nose. *Journal of Food Quality*, 2017:1–7, 10 2017.

[4] S. Wang and W. Shi. *Data Mining and Knowledge Discovery*, pages 49–58. 01 2011.

# A Dokumentácia implementácie

Dokumentácia systému automatického rozpoznávania spektier.