
**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**
**BRNO UNIVERSITY OF TECHNOLOGY**

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**
**FACULTY OF INFORMATION TECHNOLOGY**

**ÚSTAV INFORMAČNÍCH SYSTÉMŮ**
**DEPARTMENT OF INFORMATION SYSTEMS**


# APLIKACE PRO DETEKCI ODLEHLÝCH HODNOT

**APPLICATION FOR OUTLIER DETECTION**

**DIPLOMOVÁ PRÁCE**
**MASTER'S THESIS**


**AUTOR PRÁCE** **FRANTIŠEK SILADI**
**AUTHOR**

**VEDOUCÍ PRÁCE** **Ing. IVANA BURGETOVÁ, Ph.D.**
**SUPERVISOR**



**BRNO 2022**

# Abstrakt

Cieľom tejto práce je preštudovať metódy na detekciu odľahlých hodnôt a vytvoriť aplikáciu, ktorá pomocou jednotlivých metód a kombináciou metód, dokáže tieto hodnoty správne identifikovať. Ďalším cieľom aplikácie je jednoducho a prehľadne zobrazovať výsledky detekcie odľahlých hodnôt a následne ich vhodne vizualizovať v 2D alebo 3D priestore. Súčasťou práce sú aj experimenty na zvolených dátových sadách, ktoré sú prispôsobené na detekciu odľahlých hodnôt a na automaticky generovaných sadách. Aplikácia a experimenty boli tvorené v jazyku Python a na vytvorenie grafického užívateľského rozhrania bolo použité Qt pre Python.



# Abstract

The goal of this work is to investigate methods for outlier detection and to create an application which is able to correctly identify these values using individual methods or combinations of outlier detection methods. Another goal of the application is to clearly display the results of outlier detection and then visualize them in 2D or 3D space. The work also includes experiments on selected data sets, which are adapted to outlier detection and on automatically generated sets. The application and experiments were created in Python and Qt for Python was used to create the graphical user interface.



# Klíčová slova

detekcia ohľahlých hodnôt, aplikácia, dolovanie dát, experimenty, Python

# Keywords

outliers detection, application, data mining, experiments, Python


# Citace

SILADI, František. *Aplikace pro detekci odlehlých hodnot.* Brno, 2022. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Ivana Burgetová, Ph.D.

# Aplikace pro detekci odlehlých hodnot

## Prohlášení

Prohlašuji, že jsem tuto diplmovovú prácu vypracoval samostatně pod vedením pani Ing. Ivany Burgetové Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
František Siladi
17. května 2022

## Poděkování

Týmto by som chcel poďakovať vedúcej mojej diplomovej práce, Ing. Ivane Burgetovej Ph.D., za odborné rady, podporu a správne nasmerovanie pri riešení tejto práce. Ďalej by som rád vyjadril poďakovanie priateľke, rodine a kamarátom, ktorý ma podporovali pri štúdiu a tvorbe tejto diplmovej práce.

# Obsah

# Kapitola 1

# Úvod

Detekcia odľahlých hodnôt je jednou z hlavných úloh dolovania údajov. Vo všeobecnosti sa to chápe ako identifikácia údajov, záznamov alebo položiek, ktoré sa výrazne odlišujú od zvyšných dát, teda nemajú normálne správanie.

Detekcia odľahlých hodnôt má mnoho rôznych použití v odvetviach ako je bankovníctvo, zdravotníctvo alebo v IT priemysle. V bankovníctve sa používa napríklad na identifikáciu podozrivých transakcií na karte, či už môže ísť o veľké vklady peňazí alebo podozrivé prevody. V zdravotníctve môže ísť napríklad o diagnostiku ochorenia na základe anomálii z krvných vzoriek a v IT sa zase môže použiť pri prevádzke serverov, v prípade že ne začnú prichádzať nejaké nezvyčajné dotazy. Toto je len zlomok prípadov, kedy sa dajú aplikovať metódy na detekciu odľahlých hodnôt a v mnohých prípadoch môžu ušetriť veľké množstvo peňazí, prípadne správna detekcia môže aj zachrániť životy.

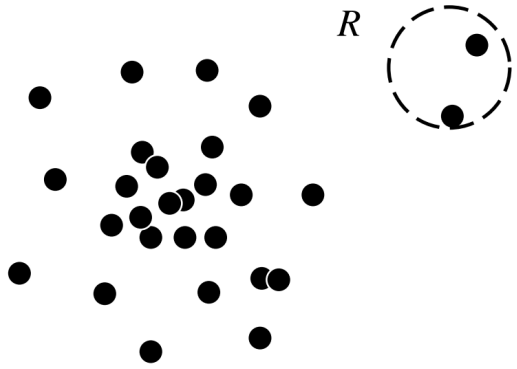
Cieľom tejto práce je preštudovať metódy na detekciu odľahlých hodnôt a vytvoriť aplikáciu, ktorá pomocou jednotlivých metód a kombináciou metód, dokáže tieto hodnoty správne identifikovať. Okrem identifikácie správnych údajov, by mala ponúknuť užívateľovi prehľadné rozhranie na nahrávanie ľubovoľných datasetov, možnosť prispôsobovať si jednotlivé metódy detekcie pomocou parametrov, prehľadné zobrazenie výsledkov detekcie a následnú vizualizáciu jednotlivých dát spoločne so štatistickými údajmi detekcie.

Práca je rozdelená do 6 kapitol, pričom kapitola 2 popisuje všeobecne rôzne typy a prístupy k detekcii odľahlých hodnôt a následne popisuje jednotlivé metódy, ktoré boli použité v aplikácii a pri experimentovaní. V kapitole 3 sú popísane dátové sady, ktoré boli použité pri vývoji aplikácie a na ktorých boli tvorené experimenty. Kapitola 4 obsahuje návrh aplikácie a jednotlive technológie, pomocou ktorých aplikácia vznikala. V ďalšej kapitole, kapitole 5, sú stručne zhrnuté najdôležitejšie časti implementácie a obrázky ako vyzerá UI aplikácie. V kapitole 6 sa nachádza popis experimentov s metódami na detekciu odľahlých hodnôt a ich výsledky.

# Kapitola 2

# Odľahlé hodnoty

Na začiatok si povedzme, čo sú to odľahlé hodnoty. Ide o hodnoty, ktoré sa významne líšia od ostatných objektov, ako keby boli generované iným mechanizmom[11]. Dátové objekty, ktoré nie sú odľahlé, sa môžu označovať ako normálne alebo očakávané údaje. Naopak odľahlé hodnoty sa môžu označovať ako abnormálne údaje. Na obrázku 2.1 sa väčšina objektov riadi Gaussovským rozložením. Avšak, objekty v oblasti R sa výrazne líšia od ostatných. Je nepravdepodobné, žeby sa riadili rovnakým rozložením ako ostatné objekty v tomto súbore, preto tieto objekty môžeme považovať za odľahlé hodnoty v tomto súbore hodnôt.




Obrázek 2.1: Dáta s odľahlým regiónom R[11].

## 2.1 Typy odľahlých hodnôt

Vo všeobecnosti môžu byť odľahlé hodnoty klasifikované do 3 kategórii, a to globálne odľahlé hodnoty, kontextové odľahlé hodnoty a kolektívne odľahlé hodnoty. V tejto sekcii budú jednotlivé kategórie popísane podrobnejšie. Informácie v tejto časti pochádzajú z [1] a [11].

### 2.1.1 Globálne odľahlé hodnoty

V danom súbore dát je dátový objekt globálnou odľahlou hodnotou ak sa výrazne odchyľuje od ostatných dát v súbore. Globálne odľahlé hodnoty sa niekedy nazývajú aj bodové anomálie a ide o najjednoduchší typ odľahlých hodnôt. Väčšina metód na detekciu odľahlých

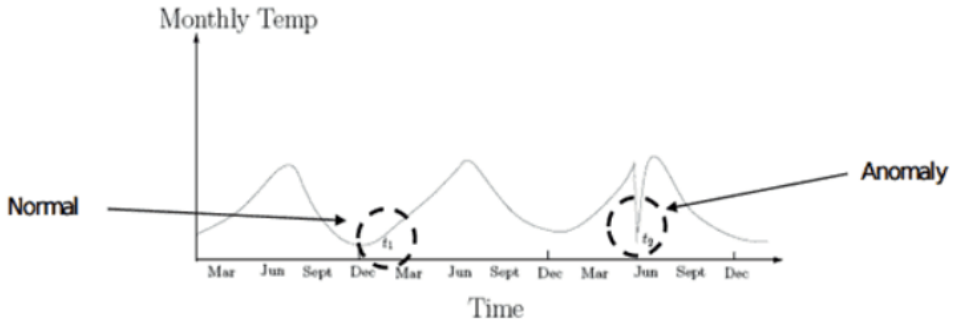
hodnôt je zameraná práve na detekciu týchto globálnych. Opäť uvažujme o dátovej sade z obrázku 2.1, v oblasti R sa nachádzajú odľahlé hodnoty, ktoré sú práve globálnymi odľahlými hodnotami. Na odhalenie globálnych odľahlých hodnôt je rozhodujúce nájsť vhodné meranie odchýlky vzhľadom na danú úlohu. Navrhujú sa rôzne merania a na ich základe sa metódy detekcie odľahlých hodnôt rozdeľujú na rôzne kategórie.

### 2.1.2 Kontextové odľahlé hodnoty

V danom súbore objektov je dátový objekt kontextovou odľahlou hodnotou, ak sa výrazne odchyľuje od ostatných vzhľadom na konkrétny kontext objektu. Kontextové odľahlé hodnoty sú tiež známe aj ako podmienené odľahlé hodnoty, pretože sú podmienené vybraným kontextom. Práve kvôli tomu musí byť kontext špecifikovaný v rámci definície úlohy. Všeobecne platí, že pri detekcii kontextových odľahlých hodnôt sa atribúty dátových objektov delia do dvoch skupín:

- Kontextové atribúty - tieto atribúty definujú kontext objektu. V prípade, že skúmame teploty, tak tieto atribúty môžu byť dátum a miesto. Pretože keď máme napríklad teplotu 28 stupňov tak je rozdiel či to je v Toronte v zime alebo v Káhire v lete.
- Behaviorálne atribúty - tieto atribúty definujú vlastnosti objektu a používajú sa na vyhodnotenie toho, či je objekt v kontexte, do ktorého patrí, odľahlý. V príklade s teplotami môžu byť týmito atribútmi teplota, vlhkosť alebo tlak.

Na rozdiel od globálnej detekcie odľahlých hodnôt sa pri kontextovej detekcii odľahlých hodnôt zisťuje, či dátový objekt závisí nielen na behaviorálnych atribútoch, ale aj kontextových. Na obrázku 2.2 môžeme vidieť, že hodnota sa považuje za odľahlú podľa mesiaca v ktorom bola zachytená. Globálne odľahlé hodnoty môžeme považovať za špeciálny prípad kontextových odľahlých hodnôt, pri ktorých množina kontextových atribútov je prázdna. Kontextová analýza odľahlých hodnôt poskytuje používateľovi flexibilitu v tom, že je možné skúmať odľahlé hodnoty v rôznych kontextoch, ktoré môžu byť v mnohých aplikáciach veľmi žiadúce.

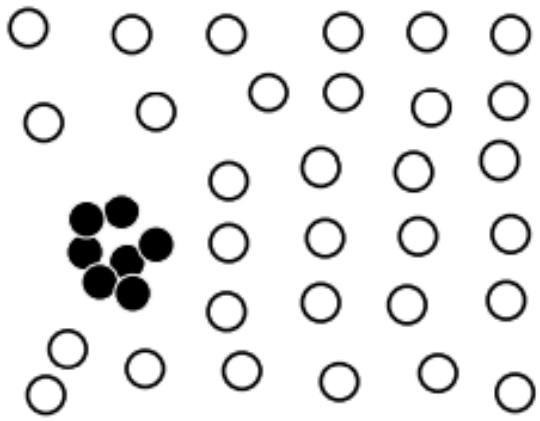



Obrázek 2.2: Príklad kontextovej odľahlej hodnoty [1].

### 2.1.3 Kolektívne odľahlé hodnoty

Vzhľadom na súbor údajov, tvorí podmnožina objektov kolektívny odľahlý bod, ak sa tá množina ako celok, výrazne odchyľuje od celého súboru dat. Dôležité je, že jednotlivé objekty

z odľahlej množiny nemusia byť samé o sebe odľahlé. Ako môžeme vidieť na obrázku 2.3, tak čierne objekty ako celok tvoria kolektívnu odľahlú hodnotu pretože hustota týchto objektov je oveľa vyššia ako hustota ostatných objektov v súbore dát. Avšak každý čierny objekt jednotlivo nie je odľahlou hodnotou vzhľadom na celý súbor dát. Na rozdiel od globálnej



Obrázek 2.3: Príklad kolektívnych odľahlých hodnôt[11].

alebo kontextovej detekcie odľahlých hodnôt máme pri kolektívnej detekcii brať do úvahy nie len správanie jednotlivých objektov, ale aj správanie skupín objektov. Preto na detekciu kolektívnych odľahlých hodnôt potrebujeme znalosti o vlastnostiach medzi dátovými objektami, ako sú vzdialenosti alebo podobnosti medzi nimi. Súhrnne možno povedať, že súbor dát môže mať viacero typov odľahlých hodnôt. Okrem toho môže objekt patriť k viac ako jednému typu odľahlých hodnôt. Globálna detekcia odľahlých hodnôt je najjednoduchšia. Kontextová detekcia vyžaduje informácie o pozadí na určenie kontextových atribútov. Kolektívna detekcia vyžaduje informácie o vlastnostiach vzťahov medzi objektami s cieľom nájsť skupinu odľahlých hodnôt.

## 2.2 Spôsoby detekcie odľahlých hodnôt

V literatúre aj v praxi je mnoho metód na detekciu odľahlých hodnôt. V knihe [11] sa uvádzajú dva spôsoby kategorizácie metód detekcie odľahlých hodnôt. Prvou kategorizáciou je podľa toho, či vzorky objektov na analýzu majú odborne priradené, či ide alebo nejde o odľahlé hodnoty, a tieto hodnoty môžu byť použite na vytvorenie modelu detekcie odľahlých hodnôt. Druhým spôsobom kategorizácie je rozdelenie metód do skupín podla spôsobu detekcie odľahlých hodnôt. V tejto sekcii sa budeme venovať prvej kategorizácii a to rozdelenie na učenie z učiteľom, bez učiteľa a na kombináciu týchto spôsobov. Informácie v tejto kapitole pochádzajú z knihy [11].

### Metódy učenia s učiteľom

V prípade ak máme k dispozícii označené dáta, teda vieme, ktoré vzorky sú odľahlé a ktoré nie, tak detekcia odľahlých hodnôt môže byť braná ako klasifikačný problém. Úlohou je naučiť klasifikátor rozpoznávať odľahlé hodnoty. Dátová sada je rozdelená na trénovaciu a testovaciu časť. Aj keď mnoho klasifikačných metód môže byť použitých, tak detekcia odľahlých hodnôt obsahuje nasledujúce problémy:

- Objekty delíme typicky na 2 triedy (normálne objekty a odľahlé objekty), avšak tieto triedy sú nevyvážené. Množstvo odľahlých objektov je typicky menšie ako množstvo normálnych objektov. Nedostatok odľahlých objektov v trénovacej sade sa môže vyriešiť pridaním tzv. umelých odľahlých hodnôt alebo zdvojením už existujúcich odľahlých hodnôt

- V mnohých aplikáciach na detekciu odľahlých hodnôt je potrebné zachytiť čo najviac odľahlých hodnôt. Je lepšie mať zachytených viacej odľahlých hodnôt, aj keď nesprávne, ako keby sa stalo, že nejaká odľahlá hodnota nebude správne identifikovaná.

Metódy učenia s učiteľom na detekciu odľahlých hodnôt musia byť dôkladne pripravené na to ako majú trénovať a ako majú interpretovať klasifikačný pomer vzhľadom k tomu, že odľahlých hodnôt je oveľa menej ako bežných hodnôt.

### Metódy učenia bez učiteľa

V niektorých prípadoch nie je k dispozícii značenie, či ide o normálny objekt alebo odľahlý. Metódy učenia bez učiteľom vytvárajú základný predpoklad, že normálne objekty sú nejako zoskupené. Inými slovami tieto metódy očakávajú, že odľahlé hodnoty porušujú nejaký vzor častejšie ako normálne objekty. Normálne objekty nemusia mať všetky rovnaké vlastnosti, ale môžu byť zoskupené do skupín s podobnými vlastnosťami, avšak odľahlé hodnoty sú ďaleko od týchto skupín.

Mnoho zhlukovacích metód môže byť použitých na detekciu odľahlých hodnôt bez učiteľa. Hlavná myšlienka je nájsť najprv zhluk a potom dáta ktoré do neho nepatria označiť ako odľahlé. Avšak vznikajú tu dva problémy. Prvý, že dáta ktoré nepatria do zhluku tak môžu byť šum. Druhý, nájsť zhluky môže byť často vysoko náročne, keďže sa predpokladá, že normálnych objektov je oveľa viacej ako odľahlých. Súčasné metódy na detekciu odľahlých hodnôt ponúkajú možnosti ako nájsť odľahlé hodnoty priamo bez nutnosti presne nájsť zhluky v normálnych dátach.

### Kombinácia metód učenia z učiteľom a učenia bez učiteľa

V mnohých aplikáciach je síce možne získať niekoľko označených príkladov, ale tento počet je často malý. Môžeme sa stretnúť s prípadmi keď je označená len malá množina normálnych a/alebo odľahlých hodnôt a väčšina je neoznačená. Metódy, ktoré pracujú s takými dátami sa musia zaoberať problémami, že niektoré dáta sú označenie a iné nie. Napríklad ak sú niektoré normálne dáta označené, tak ich môžeme použiť a podľa nich označiť dáta, ktoré sú v ich blízkosti alebo sú im veľmi podobné, aby sme natrénovali model pre nie odľahlé objekty. Tento model môže byť následne využitý na detekciu odľahlých hodnôt, teda hodnôt, ktoré nevyhovujú tomuto modelu. Ak máme druhý prípad a máme len malú množinu odľahlých objektov označenú, tak vytvárať model je neefektívne, pretože je nepravdepodobné, že týchto pár odľahlých hodnôt bude reprezentovať celú množinu odľahlých hodnôt. Na zlepšenie kvality nám pomôžu modely naučené bez učiteľa.

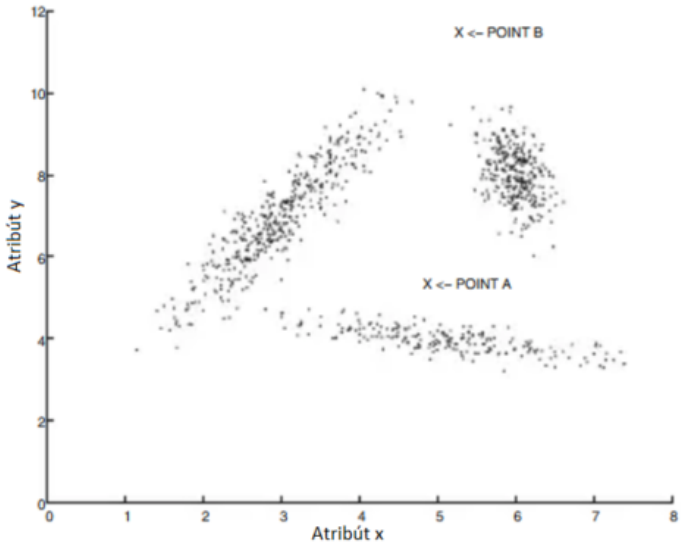
## 2.3 Metódy detekcie odľahlých hodnôt

V tejto kapitole je popísaný druhý spôsob kategorizácie metód na detekciu odľahlých hodnôt. Informácie o rozdelení pochádzajú z knihy [2].

## 2.4 Pravdepodobnostné modely na detekciu odľahlých hodnôt

Prvé metódy na detekciu odľahlých hodnôt boli založené na pravdepodobnostných a štatistických modeloch a pochádzajú z devätnásteho storočia. Tieto metódy boli navrhnuté dávno pred nástupom a popularizáciou počítačovej technológie, a preto boli navrhnuté bez väčšieho zamerania na praktické otázky, ako je reprezentácia údajov alebo výpočtová efektivita. Napriek tomu sú základné matematické modely mimoriadne užitočné a nakoniec boli prispôsobené rôznym výpočtovým scenárom. Informácie o pravdepodobnostných modeloch pochádzajú z knihy [2].

Obľúbenou formou štatistického modelovania v detekcii odľahlých hodnôt je zisťovanie extrémnych jednorozmerných hodnôt. V takýchto prípadoch je žiaduce určiť hodnoty objektov na chvoste jednorozmerného rozdelenia spolu s príslušnou úrovňou štatistickej dôležitosti. Hoci extrémne jednorozmerné hodnoty patria do veľmi špecifickej kategórie odľahlých hodnôt, majú početné uplatnenie. Napríklad prakticky všetky algoritmy na detekciu odľahlých hodnôt používajú číselné skóre na meranie odľahlosti dátových bodov a finálnym krokom v týchto algoritmoch je určenie extrémnych hodnôt z týchto skóre. Identifikácia štatisticky významných extrémnych hodnôt pomáha pri premene skóre odľahlosti jednotlivých hodnôt na binárne označenia. Preto aj v prípade, že modelovanie extrémnych




Obrázek 2.4: Zobrazenie rozdielu medzi viacrozmernými extrémnymi hodnotami a odľahlými hodnotami [2].

hodnôt nie je možné vykonať na pôvodných dátach, schopnosť efektívne určiť extrémne hodnoty tvorí jadro všetkých algoritmov detekcie odľahlých hodnôt.

Modelovanie extrémnych hodnôt sa dá ľahko rozšíriť aj na viacrozmerné údaje. Dátové body, ktoré ležia na pareto-extrémov[6] údajov sa označujú ako viacrozmerné extrémne hodnoty. Príklad na obrázku 2.4, dátový bod „B“ je viacrozmerná extrémna hodnota. Na druhej strane, dátový bod „A“ je odľahlou hodnotou, ale nie viacrozmernou extrémnou hodnotou. Metódy viacrozmernej analýzy extrémnych hodnôt sa niekedy používajú aj na všeobecnú analýzu odľahlých hodnôt. Tieto techniky môžu niekedy fungovať prekvapivo dobre v reálnych aplikáciách detekcie odľahlých hodnôt, hoci nie sú určené ako všeobecné metódy

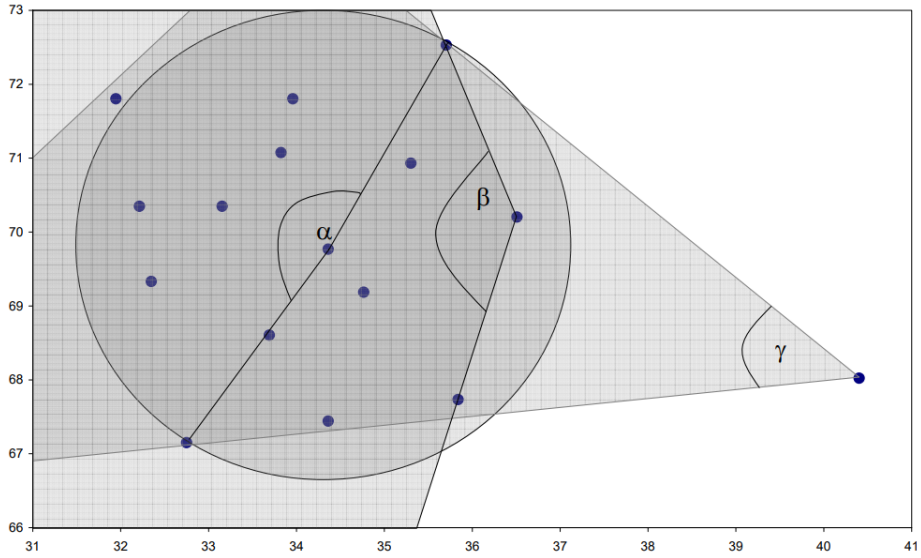
detekcie odľahlých hodnôt. Nevýhodou použitia takýchto metód vo všeobecnosti je, že dátové body ako „A“ na obrázku 2.4 sú takýmito metódami prehliadané. Napriek tejto zjavnej nevýhode by sa takéto metódy nemali v reálnych aplikáciách ignorovať. V mnohých prípadoch sa takéto techniky môžu pridať ako jedna alebo viacero zložiek kombinovaných metód s cieľom zvýšiť ich presnosť.

Pravdepodobnostné modelovanie je možné použiť aj na zistenie všeobecných odľahlých hodnôt mimo extrémnych hodnôt. Napríklad na obrázku 2.4 je možné modelovať súbor dát ako zmes troch Gaussových zložiek, a teda objaviť odľahlé hodnoty „A“ aj „B“. Modely zmesí možno považovať za pravdepodobnostné verzie zhlukovacích algoritmov, ktoré objavujú odľahlé hodnoty ako vedľajší produkt. Významnou výhodou týchto metód je, že sa dajú ľahko generalizovať na rôzne formáty dát alebo dokonca na zmiešané typy atribútov, akonáhle sa definuje generatívny model pre dané objekty. Väčšina pravdepodobnostných modelov predpokladá konkrétny tvar základného rozdelenia pre každú zložku zmesi (napr. Gaussovo) na modelovanie normálnych vzorov dátových bodov. Následne sa parametre tohto modelu naučia tak, aby pozorované dáta mali maximálnu vierohodnosť, že boli vygenerované daným modelom. Dátové body, ktoré majú neobvykle nízku pravdepodobnosť, že budú vygenerované modelom, sa identifikujú ako odľahlé hodnoty.

### 2.4.1 ABOD: Angle-Based Outlier Detection

Dolovanie objektov vo vysoko rozmerných dátach vyžaduje rôzne prístupy k hľadaniu vzorov. Je odporúčané používať nie len vzdialenosti medzi bodmi vo vektorom priestore, ale predovšetkým smery vektorov vzdialenosti. Jednou z metód ktorá sa venuje tomuto štýlu výpočtu tak je metóda ABOD. Informácie o tejto metóde vychádzajú z [17].

Porovnávanie uhlov medzi dvojicami vektorov vzdialenosti pomáha rozlíšiť medzi podobnými bodmi a bodmi, ktoré sú odľahlé. Uvažujme jednoduchý súbor dát, ako je znázornené na obrázku 2.5. Pre bod v zhluku sa uhol, medzi rozdielovými vektormi (vektory zo skúmaného bodu k iným dvom bodom v dátovej sade), značne líšia. Rozptyl uhlov sa zmenši pre body na hranici zhluku, Avšak aj tu je rozptyl stále relatívne vysoký v porovnaní s rozptylom uhlov pre skutočné odľahlé hodnoty. Tu budu uhly k väčšine párov bodov malé, pretože väčšina bodov je zoskupená v niektorých smeroch. Zodpovedajúce spektrá pre tieto tri typy bodov sú znázornené pre vzorový súbor údajov na obrázku 2.6. Ako ukazuje graf, spektrum uhlov k dvojiciam bodov zostáva pomerne malé, zatiaľ čo rozptyl uhlov je väčší pre hraničné body zhluku a veľmi vysoký pre vnútorné body zhluku. V dôsledku týchto úvah môže faktor odľahlosti založený na zhlukoch (angle-based outlier factor - ABOF), na ktorom je založená metóda ABOD, opisovať divergenciu smerov objektov. Ak je spektrum pozorovaných uhlov pre daný bod široké, tzn. bod bude obklopený inými bodmi vo všetkých možných smeroch, čo znamená, že bod je umiestnený vo vnútri zhluku. Ak je spektrum pozorovaných uhlov pre bod pomerne malé ostatné body budú umiestnené len v určitých smeroch, tak to znamená, že bod je umiestnený mimo priestor bodov, ktoré tvoria zhluk. Aby sme priradili hodnotu ABOF ľubovoľnému objektu v databáze $D$, vypočítamé skalárny súčin rozdielových vektorov ľubovoľnej trojice bodov (t.j. dotazovaný bod $\overrightarrow{A} \in D$ a všetkých dvojíc $(\overrightarrow{B}, \overrightarrow{C})$) normalizovaného kvadratickým súčinom dĺžok rozdielových vektorov, tzn., že uhol ma menšiu váhu ak sú odpovedajúce body vzdialené ďalej od neho. Týmto váhovým koeficientom vzdialenosť predsa len ovplyvňuje hodnotu, ale len z malej časti. Napriek tomu je toto váženie odchýlky dôležité. Rozptyl tejto hodnoty vo všetkých dvojiciach pre dotazovaný bod $\overrightarrow{A}$ predstavuje faktor odľahlosti založený na uhle (ABOF) bodu $\overrightarrow{A}$. Formálne:

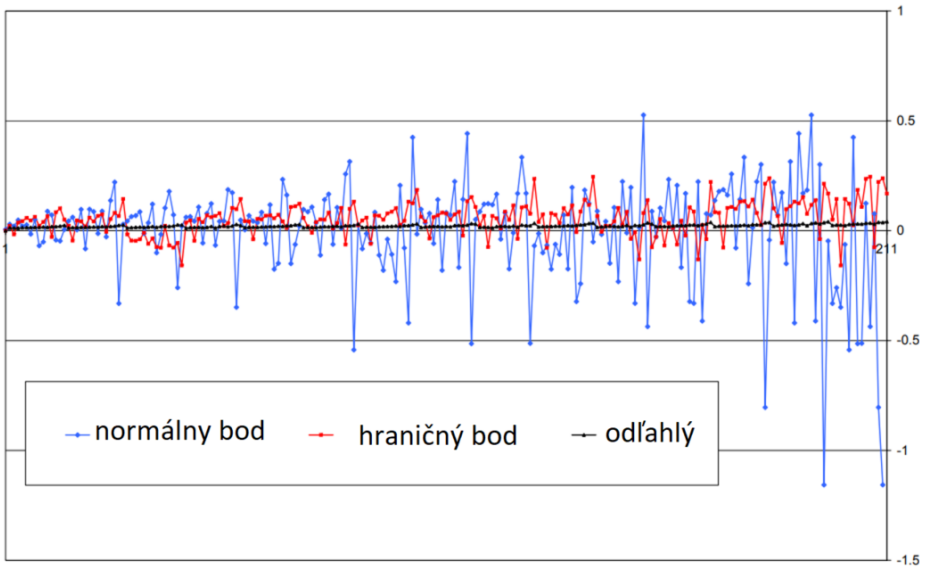



Obrázek 2.5: Ilustrácia myšlienky detekcie na základne uhlov[17].

**Definícia 2.4.1** (ABOF)**.** Máme databázu $D \subseteq \mathbb{R}^d$, bod $\overrightarrow{A} \in D$ a funkciu $||.|| : \mathbb{R}^d \to \mathbb{R}_0^+$. Skalárny súčin je označený ako $\langle ., . \rangle : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Pre dva body $\overrightarrow{B}, \overrightarrow{C} \in D, \overrightarrow{BC}$ označuje rozdielový vektor $\overrightarrow{C} - \overrightarrow{B}$.

Odľahlý faktor založený na uhloch $ABOF(\overrightarrow{A})$ je rozptyl cez uhly medzi rozdielovými vektormi bodu $A$ ku všetkým párom bodov v $D$ vážených vzdialenosťou bodov:

$$ABOF(\overrightarrow{A}) = VAR_{\overrightarrow{B},\overrightarrow{C} \in D} \left( \frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot |\overline{AC}\|^2} \right) \tag{2.1}$$




Obrázek 2.6: Zobrazenie spektier pre rozdielne typy bodov[17].

Pre každú trojicu bodov $(\overrightarrow{A}, \overrightarrow{B}, \overrightarrow{C})$ platí, že sú tieto tri body vzájomne odlišné. To znamená, že namiesto $\overrightarrow{B} \in D$ a $\overrightarrow{C} \in D$ definícia presnejšie znie ako $\overrightarrow{B} \in D \setminus \{\overrightarrow{A}\}$ a

$\overrightarrow{C} \in D \setminus \{\overrightarrow{A}, \overrightarrow{B}\}$. Ušetrené to je v prospech čitateľnosti. Algoritmus ABOF priradí každému bodu v databáze faktor odľahlosti ABOF založený na uhle a ako výsledok vráti zoznam bodov zoradený podľa ich ABOF. Uvažujme opäť vzorový súbor údajov na obrázku 2.5. Zoradenie týchto bodov, ktoré poskytuje ABOD, je vyznačené na obrázku 2.7a. V tomto príklade je najvyššie umiestnený bod (na obrázku označený ako 1) jednoznačne najodľahlejším bodom. Ďalšie poradie je obsadené hraničnými bodmi zhluku. Najnižšie hodnoty sú priradené vnútorným bodom zhluku. Keďže vzdialenosť sa zohľadňuje len ako váha hlavného kritéria, rozptylu uhlov, ABOD dokáže stručne odhaliť odľahlé hodnoty aj vo vysokorozmerných údajoch, kde LOF a iné prístupy založené čisto na vzdialenosti zhoršujú presnosť. Okrem toho, ako je znázornené vyššie, ABOD umožňuje aj odlišné hodnotenie hraničných bodov v porovnaní s vnútornými bodmi zhluku. To nie je možné pri väčšine ostatných modelov odľahlých hodnôt.

Väčšina modelov detekcie odľahlých hodnôt vyžaduje, aby používateľ špecifikoval parametre, ktoré sú pre výsledok prístupu rozhodujúce. V prípade prístupov bez učenia sú takéto požiadavky vždy nevýhodou. Veľkou výhodou ABOD je teda to, že je úplne bez parametrov. ABOD za behu získava vysvetlenie, prečo sa bod považuje za odľahlý. Vektor rozdielu k najpodobnejšiemu objektu v najbližšej skupine bodov poskytuje divergenciu kvantitatívne pre každý atribút, a teda vysvetľuje, prečo (t. j. v ktorých atribútoch o koľko) je bod odľahlý. V prípade príkladu na obrázku 2.5 je vysvetlením pre najvyššie umiestnený odľahlý bod to, že sa od najbližšieho bodu najbližšej skupiny je príliš ďaleko a preto má hodnota ABOF vysokú váhu, ako je znázornené na obrázku 2.7b.

Problém základného prístupu ABOD je zrejmý: keďže pre každý bod sa musia zohľadniť všetky dvojice bodov, časová zložitosť je $O(n^3)$, čo nie je atraktívne v porovnaní napr. s LOF, ktorý je v $O(n^2 \cdot k)$. V tomto smere vznikol vylepšený algoritmus fastABOD. Tento aproximačný algoritmus, aproximuje ABOF na základe vzorky databázy. Najvhodnejšie je použiť dvojice bodov s najsilnejšou váhou rozptylu, napr. dvojice medzi $k$-najbližšími susedmi. Na túto aproximáciu by sa mohla použiť aj náhodná množina k ľubovoľných dátových bodov, avšak najbližší susedia majú najväčšie váhy v ABOF. Použitie najbližších susedov by teda mohlo viesť k lepšej aproximácii, najmä v súboroch údajov s nízkou dimenziou, kde je vzdialenosť významnejšia.

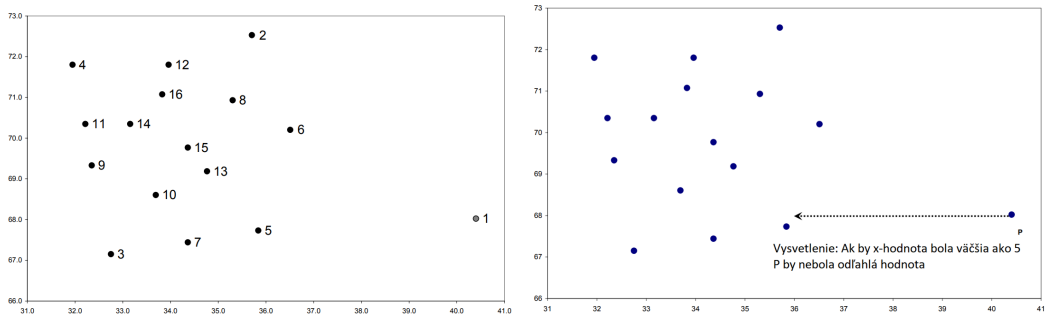
$$approxABOF_k(\overrightarrow{A}) = VAR_{\overrightarrow{B},\overrightarrow{C} \in \mathbb{N}_k(\overrightarrow{A})} \left( \frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot |\overline{AC}\|^2} \right) \tag{2.2}$$

Výsledkom tejto aproximácie je zrýchlenie o jeden rád. Výsledný algoritmus FastABOD je v $O(n^2 + n \cdot k^2)$. Vďaka tomu je FastABOD vhodný pre súbory údajov pozostávajúce z mnohých bodov. Kvalita aproximácie však závisí od počtu k najbližších susedov a kvality výberu najbližších susedov. Táto kvalita sa zvyčajne zhoršuje s rastúcou dimenziou údajov.

### 2.4.2 COPOD: Copula-Based Outlier Detection

Väčšina metód vyžaduje výber a ladenie parametrov (napríklad počet zhlukov). Metóda COPOD sa v tomto líši, keďže dokáže relatívne dobre určiť odľahlé hodnoty bez toho, aby bolo nutné zadávať rôzne parametre. Je založená na empirických kumulatívnych distribučných funkciách (ECDF) a nezahŕňa žiadne učenie ani stochastický tréning. Taktiež dokáže jednoducho spracovať súbory údajov veľkým množstvom atribútov a taktiež s veľkým počtom údajov. Popis metódy vychádza z [19].

Základom metódy sú kopule, ide o funkcie, ktoré umožňujú oddeliť okrajové distribúcie od štruktúry závislosti danej viacrozmernej distribúcie. Formálne, $d$-variančná ko-




(a) Hodnotenie bodov vo vzorových dátach podľa ABOF.

(b) Vysvetlenie rozhodnutia o odľahlej hodnote.

Obrázek 2.7: Ukážka detekcie odľahlých hodnôt metódou ABOD[17].

pula, $C : [0,1]^d \longrightarrow [0,1]$, je kumulatívna distribučná funkcia (CDF) náhodného vektoru $(U_1, U_2, ..., U_d)$ s uniformným rozložením:

$$C_U(u) = \mathbb{P}(U_1 \leq u_1, ..., U_d \leq u_d) \tag{2.3}$$

kde $P(U_j \leq u_j) = u_j$ pre $j \in 1, ..., d$ a $u_j \in [0,1]$.

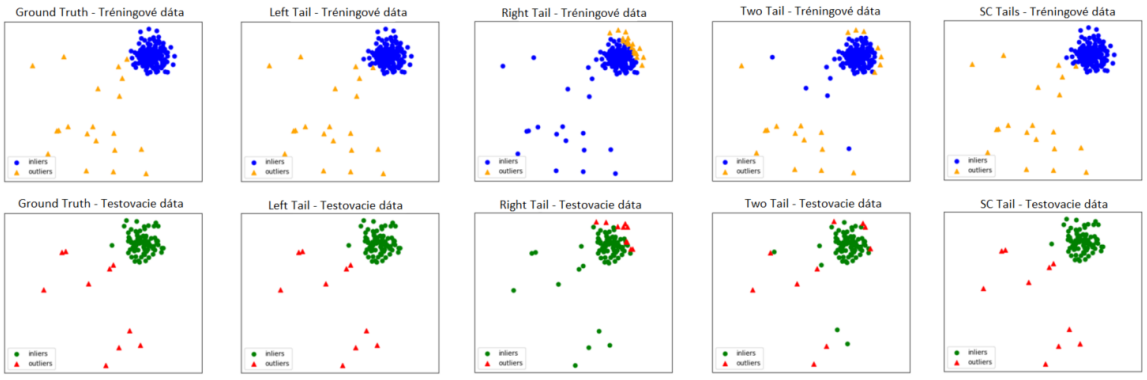
Je dobre známe, že uniformné rozloženie môže byť zmenené na akékoľvek požadované rozloženie pomocou inverzného vzorkovania

$$X_j = F_j^{-1}(U_j) \sim F_j \tag{2.4}$$

Taktiež pre ľubovoľnú náhodnú premennú $(X_1, ..., X_d)$ so spojením distribučných funkcii $F(x_1, ..., x_d)$ a marginálnym rozložením $F_1, ..., F_d$ existuje kopula

$$F(x) = C(F_1(x_1), ..., F_d(x_d)) \tag{2.5}$$

Inými slovami, kopula dovoľuje opísať prienik distribúcii $(X_1, ..., X_d)$ iba pomocou ich hraníc. To zaručuje, že každá dimenzia môže byť modelovaná oddelene.




Obrázek 2.8: Príklad ako rôzne *tail* pravdepodobnosti ovplyvňujú výsledok[19].

Detekcia odľahlých hodnôt pomocou COPOD je trojstupňový proces. Najprv vypočítame empirické CDF na základe údajov $X = [X_{1,i}, ..., X_{d,i}], i = 1, ..., n$. Po druhé, používame empirické CDF na vytvorenie funkcie empirickej kopule. Nakoniec použijeme empirickú kopulu na aproximáciu tzv. *tail* pravdepodobnosti pomocou rovnice 2.5. Algoritmus prebieha

tak, že sa pre každú dimenziu $d$ vypočíta tzv. *left tail* ECDFs pomocou

$$\hat{F}_d(x) = \frac{1}{n}\sum_{1}^{n}\mathbb{P}(X_i \leq x) \tag{2.6}$$

následne sa vypočíta tzv. *right tail* ECDFs

$$\hat{F}_d(x) = \frac{1}{n}\sum_{1}^{n}\mathbb{P}(-X_i \leq -x) \tag{2.7}$$

a nakoniec koeficient špicatosti

$$b_i = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x_i})^3}{\left(\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x_i})^2}\right)^3} \tag{2.8}$$

Následne sa pre všetky dáta sa vypočíta pozorovania empirickej kupoly.

$$\hat{U}_{d,i} = \hat{F}_d(x) \tag{2.9}$$

$$\hat{V}_{d,i} = \hat{F}_d(x) \tag{2.10}$$

$$\hat{W}_{d,i} = \hat{U}_{d,i} \text{ ak } b_d < 0 \text{ ináč } \hat{V}_{d,i} \tag{2.11}$$

a následne sa vypočítajú *tail* pravdepodobnosti pre $X_i$ a to následovne

$$p_l = \sum_{j=1}^{d}\log(\hat{U}_{j,i}) \tag{2.12}$$

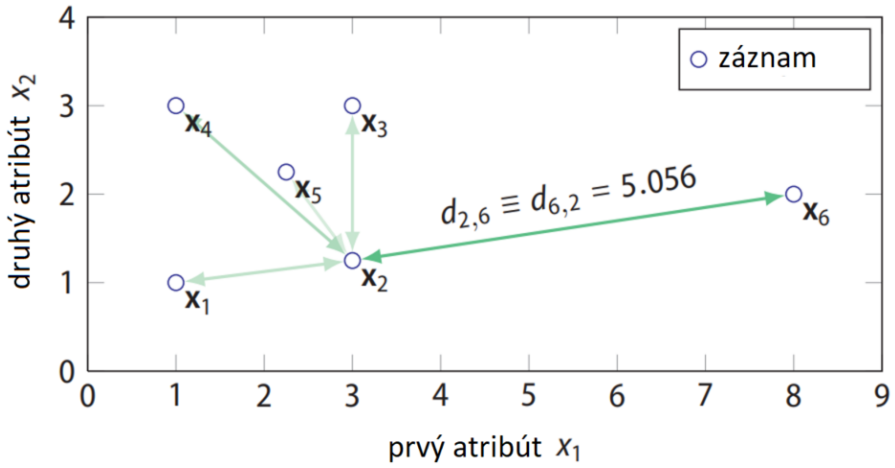
$$p_r = \sum_{j=1}^{d}\log(\hat{V}_{j,i}) \tag{2.13}$$

$$p_s = \sum_{j=1}^{d}\log(\hat{W}_{j,i}) \tag{2.14}$$

Z týchto troch hodnôt sa vyberie najvyššia hodnota a ta je vrátená ako skóre odľahlosti pre jednotlivé dáta. Skóre odľahlosti môže byť v rozmedzí $(0, \infty)$ a nehovorí, či ide alebo nejde o odľahlú hodnotu ale uvádza relatívnu mieru odľahlosti v porovnaní z ostatnými prvkami v datasete.

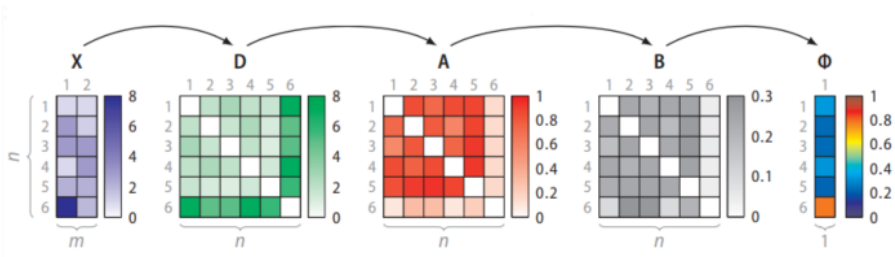
### 2.4.3 SOS: Stochastic Outlier Selection

SOS je algoritmus detekcie odľahlých hodnôt bez učiteľa, ktorý berie ako vstup maticu atribútov alebo maticu odlišnosti a ako výstup vracia pravdepodobnosť odľahlosti pre každý bod. Podrobne bol popísaný v [16]. Intuitívne sa dátový bod považuje za odľahlý, ak ostatné dátové body s ním nemajú dostatočnú príbuznosť. Majme dáta na obrázku 2.9. Tie sú zobrazené v matici atribútov $X$ a následne sa transformuje na maticu odlišnosti $D$ pomocou Euklidovskej vzdialenosti (prípadne ľubovoľnej inej vzdialenostnej funkcie). Pomocou matice odlišnosti $D$, SOS vypočíta maticu príbuznosti $A$, maticu pravdepodobnosti väzieb $B$ a nakoniec vektor pravdepodobnosti $\Phi$, viz. obrázok 2.10.




Obrázek 2.9: Príklad datasetu pre vysvetlenie funkčnosti SOS[16].

Použitie konceptu príbuznosti je inšpirované z *t-Distributed Stochastic Neighbour Embedding (t-SNE)*[20], čo je nelineárna technika redukcie dimenzionality. Oba algoritmy využívajú koncept príbuznosti na kvantifikáciu vzťahu medzi dátovými bodmi. t-SNE ho využíva na zachovanie lokálnej štruktúry vysoko-rozmerného súboru údajov a SOS ho využíva na výber odľahlých hodnôt. Príbuznosť určitého dátového bodu s iným dátovým bodov klesá Gaussovským rozložením vzhľadom na ich rozdielnosť.



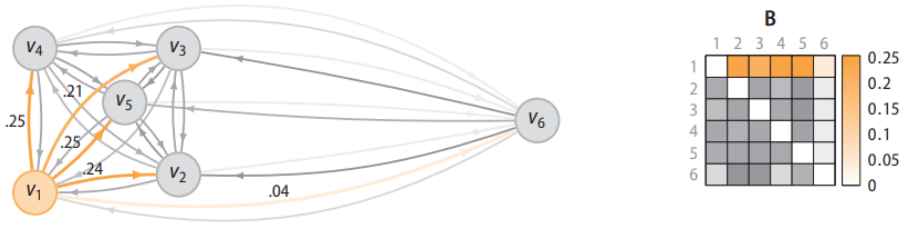

Obrázek 2.10: Kroky SOS algoritmu zo vstupu ku výstupu[16].

Ku každému dátovému bodu je priradený rozptyl. Ten závisí na hustote susedov v okolí daného bodu. Vyššia hustota znamená nižší rozptyl. V skutočnosti je rozptyl nastavený tak, že každý dátový bod ma v skutočnosti rovnaký počet susedov. Toto číslo je riadené jediným parametrom SOS nazývaným perplexity. Tento parameter môžeme interpretovať ako v kNN, sekcia 2.6.3, metóde počet susedov. Rozdiel je v tom, že v SOS nie je binárnou vlastnosťou, ale pravdepodobnostnou. Obrázok 2.11 znázorňuje pravdepodobnostné väzby dátového bodu $x_1$ z ďalšími piatimi dátovými bodmi.

Matica pravdepodobnosti väzby je len matica príbuznosti, takže súčet riadkov je rovný jednej. Na získanie odľahlej pravdepodobnosti dátového bodu vypočítame spoločnú pravdepodobnosť, že ostatné dátové body sa na ňu nenaviažu. Pravdepodobnosť, že dátový bod $x_i$ patrí k odľahlým hodnotám vypočítame ako

$$p(x_i \in C_O) = \prod_{j \neq i} (1 - b_{ji}) \tag{2.15}$$

kde $C_O$ predstavuje triedu odľahlých hodnôt, $X$ je dataset obsahujúci $n$ prvkov a $b_{ji}$ je normalizovaná rozdielnosť, čo je pravdepodobnosť, že $x_i$ si vyberie $x_j$ ako suseda. Táto jed-

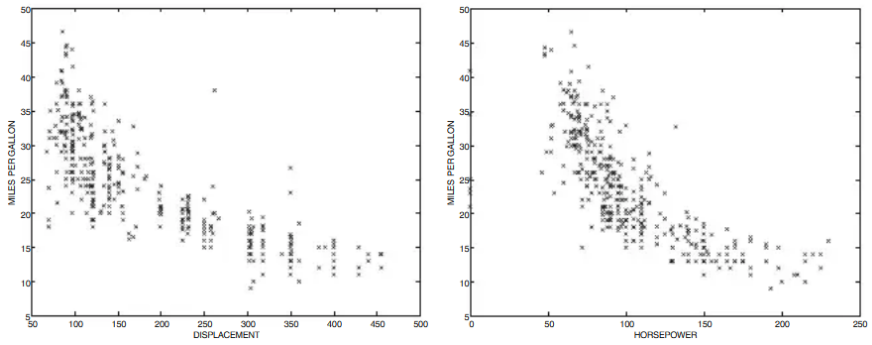



Obrázek 2.11: Zobrazenie pravdepodobnosti väzieb pre bod $v_1$[16].

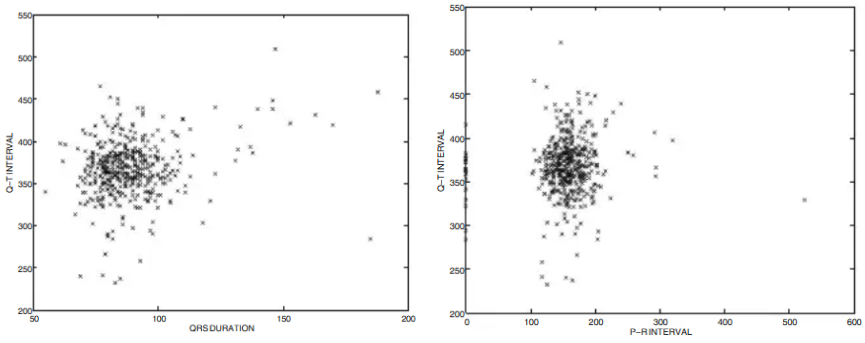
noduchá rovnica zodpovedá už spomínanej intuícii SOS: dátový bod sa považuje za odľahlý, ak s ním ostatné dátové body nemajú dostatočnú príbuznosť.

## 2.5 Lineárne modely na detekciu odľahlých hodnôt

Atribúty v reálnych údajoch sú zvyčajne vysoko korelované. Tieto závislosti poskytujú schopnosť predpovedať hodnoty atribútov na základe predchádzajúcich dát. Odľahlé hodnoty sú totiž hodnoty, ktoré sa odchyľujú od očakávaných (alebo predpovedaných) hodnôt na základe konkrétneho modelu. Lineárne modely sa zameriavajú na využívanie vzájomných závislostí atribútov na dosiahnutie správnej detekcie. V klasickej štatistickej literatúre sa tento proces označuje ako regresné modelovanie. Informácie o lineárnych modeloch pochádzajú z knihy [2].



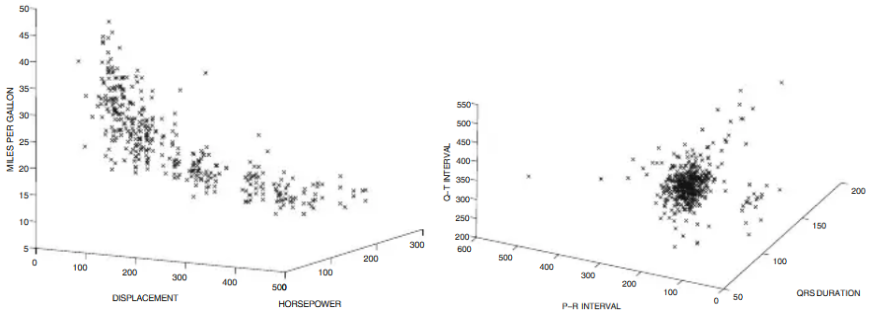

(a) Autompg - Závislosť 1. (b) Autompg - Závislosť 2.




(c) Arrythmia - Závislosť 1. (d) Arrythmia - Závislosť 2.

Obrázek 2.12: Efektívnosť lineárneho predpokladu je závislá od súboru údajov[2].

Regresné modelovanie je parametrická forma korelačnej analýzy. Táto analýza môže mať dve formy. Prvá forma predpovedá závislé premenné na základe ostatných nezávislých premenných a je vhodná pre použitie na komplexných typoch údajov ako sú časove rady. Druhá forma zhrnie celé údaje vo forme latentných premenných (sú odvodené z iných pozorovaných premenných), napríklad metóda hlavných komponentov, popísaná v sekcii 2.5.1, a je užitočnejšia pre viacrozmerný typ údajov.




(a) Autompg - 3D závislosť. (b) Arrythmia - 3D závislosť.

Obrázek 2.13: Efektívnosť lineárneho predpokladu je závislá od súboru údajov[2].

Hlavným predpokladom v lineárnych modeloch je, že (normálne) údaje sú vložené do menej rozmerného podpriestoru. Dátové body, ktoré prirodzene nezodpovedajú tomuto modelu, sa preto považujú za odľahlé hodnoty. V prípade metód založených na blízkosti, o ktorých hovoríme v sekcii 2.6, je cieľom určiť špecifické oblasti priestoru, v ktorých sa odľahlé body správajú veľmi odlišne od ostatných bodov. Na druhej strane, pri lineárnych metódach je cieľom nájsť podpriestory nižšej dimenzie, v ktorých sa odľahlé body správajú veľmi odlišne od ostatných bodov. Toto možno považovať za ortogonálny pohľad na metódy založené na zhlukov alebo metódy najbližšieho suseda, ktoré sa snažia zhrnúť údaje horizontálne (t. j. v riadkoch alebo hodnotách údajov), a nie vertikálne (t. j. v stĺpcoch alebo dimenziách).

Predpoklad lineárnych korelácií je základnou vlastnosťou, ktorú používajú lineárne modely. To môže, ale nemusí byť pravda pre konkrétny súbor údajov, čo bude mať zásadný vplyv na efektivitu modelovania. Na vysvetlenie tohto bodu použijeme Autompg a Arytmia dátové súbory z UCI Machine Learning Repository[23]. Prvý súbor údajov obsahuje atribúty popisujúce rôzne merania automobilov a zodpovedajúci počet najazdených kilometrov (mpg). Druhý súbor údajov obsahuje atribúty odvodené z údajov elektrokardiogramu (EKG) ľudských pacientov.

Na obrázkoch 2.12a a 2.12b je znázornená závislosť „Počet najazdených kilometrov na galón“ na každom z atribútov „objem“ a „konských síl“ pre atribúty Autompg dát. Je zrejmé, že tieto atribúty sú vysoko korelované. Hoci je v tomto konkrétnom súbore údajov prítomná aj značná úroveň šumu, lineárna závislosť medzi atribútmi je ľahko viditeľná. V prípade tohto súboru údajov možno v skutočnosti ukázať, že v rámci zvyšujúcej sa dimenzionality (výberom väčšieho počtu atribútov zo súboru údajov) možno údaje vyrovnať pozdĺž oveľa menej dimenzionálnych rovín. Je to zrejmé aj z trojrozmerného grafu na obrázku 2.13a. Na druhej strane, keď sa rôzne pohľady pozdĺž troch meraných rozmerov Arytmia (obrázky 2.12c, 2.12d a 2.13b), je zrejmé, že údaje sa rozdeľujú do dvoch zhlukov, z ktorých jeden je o niečo väčší ako druhý. Okrem toho je pomerne ťažké začleniť tento typ rozdelenia údajov do menej rozmerného podpriestoru. Tento súbor údajov je vhodnejší na analýzu založenú na blízkosti. V tomto konkrétnom prípade je zrejmé, že lineárny model

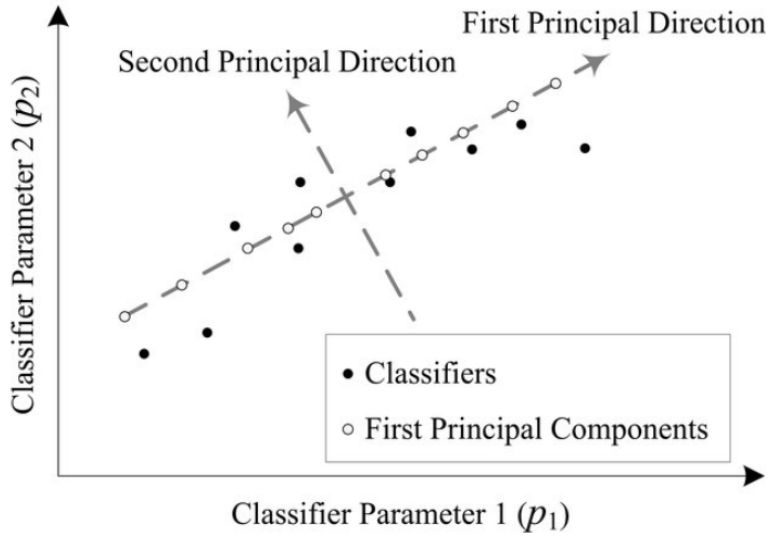
je vhodnejší pre súbory údajov, v ktorých sa údaje prirodzene zarovnávajú pozdĺž menej rozmerných hyperplôch.

### 2.5.1 PCA: Principal Component Analysis

PCA sa často používa na zmenšenie dimenzie údajov z čo najmenšou stratou informácie, aby sa dáta dali ľahko preskúmať a ďalej analyzovať. Hlavné komponenty(principal components) sú konkrétne lineárne kombinácie $p$ náhodných premenných $X_1, X_2, ..., X_p$ s tromi dôležitými vlastnosťami:

1. sú nekorelované,
2. prvý hlavný komponent ma najväčší rozptyl, druhý hlavný komponent ma druhý najväčší rozptyl atď,
3. celková variabilita všetkých hlavných komponentov spolu sa rovná celkovej variabilite pôvodných premenných $X_1, X_2, ..., X_p$

Tie môžeme ľahko získať z vlastnej analýzy (eigenanalysis) kovariančnej matice alebo korelačnej matice $X_1, X_2, ..., X_p$. Hlavné komponenty z kovariančnej matice a korelačnej matice zvyčajne nie sú rovnaké. Ak sú niektoré premenné v oveľa väčšom rozsahu ako ostatné, potom dostanú veľké váhy. Z toho dôvodu, ak sú premenné merané na stupniciach s rôznymi rozsahmi je lepšie vykonať PCA na korelačnej matici. V klasifikátore odľahlých hodnôt založenom na hlavných komponentoch PCC (principal component classifier)[28] sa predpokladá, že anomálie sú kvalitatívne odlišné od normálnych prípadov. Na vytvorenie detekčného algoritmu vykonáme PCA na korelačnej matici normálnych hodnôt. PCA je matematická technika, teda sa nevyžaduje, aby boli údaje v nejakom rozdelení, aby mohla byť metóda použitá. Metóda je však užitočná, len ak sú prvky korelované.




Obrázek 2.14: Príklad analýzy hlavných komponentov v 2D priestore[10].

Vzhľadom k tomu, že odľahlé hodnoty môžu priniesť veľké zvýšenie rozptylu, kovariancie a korelácie, je dôležite, aby trénovanie bolo bez odľahlých hodnôt aby sa správne určilo detekčné kritérium. Preto má význam začať PCA s robustným odhadom korelačnej matice.

Jednou z jednoduchých metód na získanie robustného odhadu je viacrozmerné orezávanie. Najprv použijeme Mahalanovisobu metriku na identifikáciu $100\gamma\%$ extrémnych pozorovaní, ktoré sa majú orezať. Počnúc konvenčnými odhadmi $\overline{x}$ a $S$ sa vypočíta vzdialenosť $d_i^2$ pre každé pozorovanie $x_i (i = 1, 2, ..., n)$. Pre dané $\gamma$ (v článku [28] nastavená na hodnotu 0,005) sa odstránia pozorovania odpovedajúce $\gamma * n$ najväčších hodnôt z $d_i^2 (i = 1, 2, ..., n)$. Zo zvyšných pozorovaní sa vypočítajú nové orezané odhady $\overline{x}$ a $S$ pre strednú hodnotu a kovariančnú maticu. Robustný odhad korelačnej matice sa získa pomocou prvkov $S$. Proces orezávania sa môže opakovať, aby sa zabezpečilo, že odhady $\overline{x}$ a $S$ sú odolné voči odľahlým hodnotám. Pokiaľ počet pozorovaní, ktoré zostanú po orezaní prekročí $p$ (dimenzia vektora $\overline{x}$ ), odhad $S$ určený pomocou viacrozmerného orezávania bude positive-definite (symetrický a všetky eigenvalues budú pozitívne).

Tento robustný postup mimochodom robí PCC vhodným na detekciu odľahlých hodnôt bez učiteľa. Nemôžeme očakávať, že trénovacie údaje budú vždy pozostávať len z normálnych prípadov. Niektoré podozrivé údaje alebo narušenia môžu byť v súbore údajov skryté. Aby však detekcia anomálií fungovala, predpokladáme, že počet normálnych prípadov musí byť oveľa väčší ako počet anomálií. Preto by sa pomocou vyššie opísaného postupu orezávania anomálie zachytili a odstránili z trénovacej množiny údajov. PCC pozostáva z dvoch funkcií pre skóre hlavných komponentov:

- Prvá funkcia je z hlavných komponentov $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i}$. Kde $y_i$ označuje hlavné komponenty, $q$ označuje počet hlavných komponent a $\lambda_i$ označuje rozptyl pre $i$-tu hlavnú komponentu. Táto funkcia, ktorá sa používa v literatúre, slúži na detekciu extrémnych pozorovaní s veľkými hodnotami niektorých pôvodných atribútov. Počet hlavných komponentov sa určuje na základe množstva variability v trénovacich údajoch, ktoré pripadá na tieto komponenty. Na základe experimentov je navrhované použiť $q$ ako počet hlavných komponentov, ktoré dokážu vysvetliť približne 50 percent celkovej variability štandardizovaných prvkov.

- Druhá funkcia je z vedľajších komponentov $\sum_{i=p-r+1}^{q} \frac{y_i^2}{\lambda_i}$. Kde $y_i$ označuje hlavné komponenty, $q$ označuje počet hlavných komponent, $\lambda_i$ označuje rozptyl pre $i$-tu hlavnú komponentu, $r$ označuje vedľajšie komponenty. Ak atribúty pôvodného datasetu sú nekorelované, tak každá hlavná komponenta z korelačnej matice má vlastné hodnoty (eigenvalues) rovné jednej. Na druhej strane, keď sú niektoré prvky skutočne korelované, eigenvalues niektorých vedľajších zložiek budú nulové. Preto $r$ vedľajších komponentov používaných v PCC sú tie komponenty, ktorých rozptyly alebo eigenvalues sú menšie ako 0.20, čo by znamenalo určité lineárne závislosti medzi prvkami. Použitie hodnoty 0.20 pre vlastné hodnoty možno odôvodniť aj na základe koncepcie koeficientu determinácie v regresnej analýze.

V PCC sa na klasifikáciu pozorovania $x$ najprv vypočíta skóre hlavného komponentu $x$, pre ktorý sa má určiť trieda. Nech $c_1$ a $c_2$ sú prahové hodnoty odľahlých hodnôt také, aby klasifikátor vytvoril určitú mieru falošných poplachov, $x$ sa klasifikuje takto.

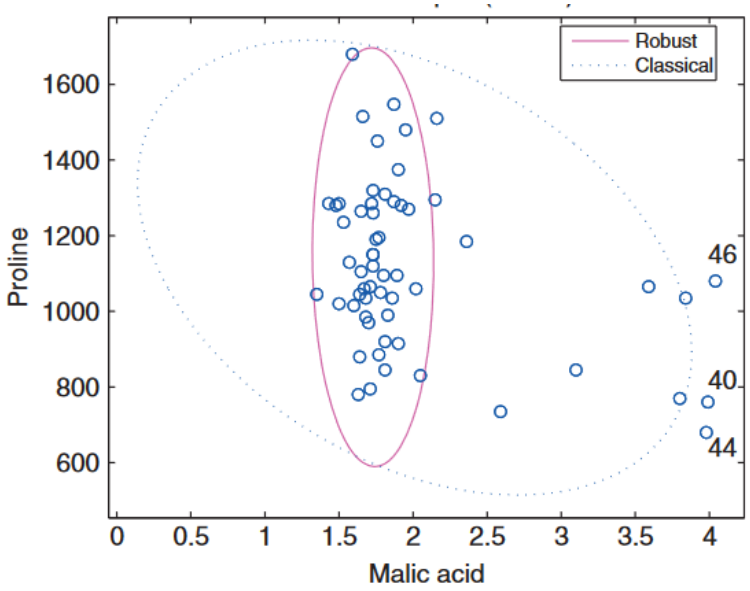
- Klasifikuj $x$ ako normálnu hodnotu ak $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} \leq c_1$ a $\sum_{i=p-r+1}^{q} \frac{y_i^2}{\lambda_i} \leq c_2$.

- Klasifikuj $x$ ako odľahlú hodnotu ak $\sum_{i=1}^{q} \frac{y_i^2}{\lambda_i} > c_1$ a $\sum_{i=p-r+1}^{q} \frac{y_i^2}{\lambda_i} > c_2$.

### 2.5.2 MCD: Minimum Covariance Determinant

MCD je metóda pre odhad priemeru a kovariančnej matice spôsobom, ktorý sa snaží minimalizovať vplyv odľahlých hodnôt. Presnú hodnotu MCD nie je možné zistiť, s výnimkou malých vzoriek alebo triviálnych prípadov. Takže algoritmus použitý na odhad MCD bude estimátorom. Kompletné informácie o algoritme pochádzajú z [15] a [12]. Algoritmus v prípade viacerých zhlukov nebude určovať štartovací bod ako náhodnú podvzorku dát. Dôvod, prečo je dôležité mať pre robustné zhlukovanie iný ako náhodný počiatočný bod je ten, že náhodný začiatok často vedieť k vzniku tvarov, ktoré sú vhodnejšie pre metriky, ktoré spracúvajú a vyhodnocujú celý dataset ako pre metriky, ktoré tvoria viacej zhlukov. Aj pri náhodných vzorkách z iba $g \times (p+1)$ bodmi, kde $g$ je počet zhlukov a $p$ je dimenzia, je veľmi nepravdepodobné, že náhodný štartovací bod rozdeliť body do $g$ zhlukov. Resp. z počiatočného bodu, ktorý odráža celú metriku údajov, je ťažké rozdeliť body do správnych $g$ zhlukov.

Zhlukovací algoritmus použitý v tej metóde je bližšie popísaný v článku [27]. Táto metóda predpokladá len to, že zhluky sú eliptické. Keďže tvar zhluku sa odhaduje z priradených bodov, vyžaduje sa, aby $p+1$ bolo priradených ku každému z hlavných zhlukov. Táto metóda však umožňuje aj nepriradenie bodov, takže by sa ľahko mohol povoliť zhluk menší ako $p+1$ a ten by bol priradený do skupiny odľahlých hodnôt.



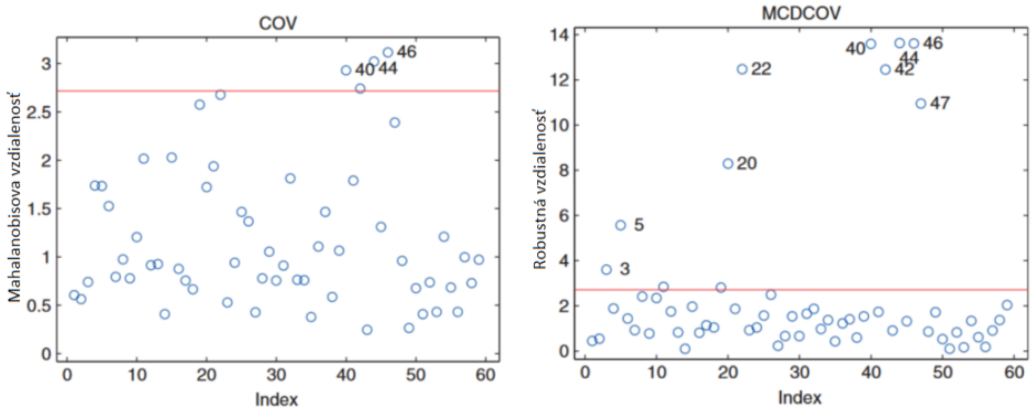

Obrázek 2.15: Zobrazenie príkladu použitia klasickej a robustnej tolerančnej elipsy[15].

Dôležitou častou tohto algoritmu je aj výpočet vzdialenosti. Aby sme mohli detekovať odľahlé hodnoty tak je potrebné vypočítať vzdialenosť pre každý bod od centra zhluku. Meranie kvadratickej vzdialenosti z body $X$ do miesta $Y$ s tvarom $S$ vo viacrozmernom prostredí sa dá určiť nasledujúcim spôsobom:

$$d_S^2(X,Y) = (X-Y)^T S^{-1}(X-Y) \tag{2.16}$$

Táto kvadratická forma sa často nazýva aj Mahalanobisova kvadratická vzdialenosť (MSD). Avšak tento výpočet nie je dostačujúco a preto sa používa robustná vzdialenosť, ktorá využíva odhady priemeru a kovariančnej matice a vyzerá následovne:

$$d_S^2(X,Y) = (X-\overline{X}_j^*)^(TS_j^*)^{-1}(X-\overline{X}_j^*) \tag{2.17}$$




(a) Mahalanobisova vzdialenosť. (b) robustná vzdialenosť.

Obrázek 2.16: Mahalanobisova vzdialenosť a robustná vzdialenosť[15].

kde $S_j^*$ a $\overline{X}_j^*$ sú odhady priemeru a kovariančnej matice MCD pre zhluk $j$. Majme dataset z obrázku 2.15, ktorý zobrazuje závislosť prolínu (aminokyselina) a jablčnej kyseliny pri vínach. Z tohto datasetu bola vypočítaná, pre jednotlivé body, Mahalanobisova a robustná vzdialenosť. Ako môžeme vidieť na obrázku 2.16, tak robustná vzdialenosť zoberie ako odľahlé hodnoty aj prvky, ktoré by neboli pre Mahalanobisovu vzdialenosť odľahlé, ale na obrázku 2.15 je jasne vidieť, že od hlavného zhluku sú ďalej ako zvyšné hodnoty.

Jadro algoritmu MCD je následovné:

- Nech $H_1$ je podmnožina bodov $h$.
- Nájdi $\overline{X}_{H_1}$ a $S_{H_1}$. (Ak $det(S_{H_1}) = 0$ potom pridávaj body do podmnožiny dokým $det(S_{H_1}) > 0$)
- Vypočítaj vzdialenosť $d^2_{S_{H_1}}(x_i, \overline{X}_{H_1}) = d^2_{H_i}(i)$ a zoraď ich pre všetky permutácie $\pi$ tak, že $d^2_{H_i}(\pi(1)) \leq d^2_{H_i}(\pi(2)) \leq ... \leq d^2_{H_i}(\pi(n))$
- $H_2 := \{\pi(1), \pi(2), ..., \pi(h)\}$

Pre každý dataset, je úplný postup výpočtu MCD pre každý zhluk následovný:

1. Použitie zhlukovacieho algoritmu na nájdenie počiatočného rozloženia zhlukov.
2. Výpočet priemeru a kovariančnej matice pre každý zhluk (každý bod patrí naviac do jedného zhluku).
3. Vypočítaj MSD pre každý zhkluk na základe naposledy vypočítaného priemeru a kovariančnej matice.
4. Priraď každý bod ku zhluku pre ktorý má najmenšiu MSD, čim sa určí veľkosť zhluku ($n_j$) na základe počtov bodov, ktoré sú najbližšie k danému zhluku.
5. Pre každý zhluk vyber „polovičnú vzorku“ ($h_j = \lfloor (n_j + p + 1)/2 \rfloor$) z bodov z kroku 4, ktoré majú najmenšie MSD.
6. Pre každý zhluk vypočítaj priemer a kovariančnú maticu aktuálnej polovičnej vzorky.
7. Opakuj kroky 4-7 dokým sa „polovičná vzorka“ nebude meniť

8. Reportuj odhady.

Pre každý zhluk, MCD bude posledná polovica vzoriek z kroku 6. Pre každý zhluk $j$ vypočítame robustnú vzdialenosť $d^2_{S^*_j}(x_i, \overline{X}^*_j)$, kde $S^*_j$ a $\overline{X}^*_j$ sú odhady priemeru a kovariančnej matice MCD pre zhluk $j$. Pre body $x_i$, ktoré sú odľahlé, tak robustná vzdialenosť bude veľká pre všetky $j$ a pre tie ktoré nie sú odľahlé nebude vzdialenosť veľká pre konkrétne $j$.

### 2.5.3 OCSVM: One-Class Support Vector Machines

Na rozdiel od tradičných SVM sa jednotriedne SVM snažia naučiť rozhodovaciu hranicu, ktorá dosahuje maximálne oddelenie medzi normálnymi a odľahlými bodmi. Je zaujímavé, že toto bola pôvodná myšlienka, z ktorej vznikli tradičné SVM. Tejto myšlienke bránila neschopnosť naučiť sa nelineárne rozhodovacie hranice, ako aj neschopnosť zohľadniť odľahlé hodnoty. Oba tieto problémy sa vyriešili zavedením jadier a začlenením mäkkých hraníc. OCSVM používa implicitnú transformačnú funkciu $\phi(\cdot)$ definovanú jadrom na premietnutie údajov do priestoru vyššej dimenzie. Algoritmus sa potom naučí rozhodovaciu hranicu (hyperplochu), ktorá oddeľuje väčšinu údajov. Len malá časť dátových bodov leží na druhej strane rozhodovacej hranice: tieto dátové body sa považujú za odľahlé hodnoty. Popis algoritmu OCSVM na detekciu odľahlých hodnôt pochádza z [3].

Existenciu takejto rozhodovacej hranice garantuje najmä Gaussovo jadro. Pozorovaním, že všetky položky jadra sú nezáporné, možno usúdiť, že všetky údaje v priestore jadra ležia v rovnakom kvadrante. Vďaka tomu je Gaussovské jadro vhodné na prácu s akýmkoľvek ľubovoľným súborom dát. Nech je funkcia $g(\cdot)$ definovaná takto:

$$g(x) = w^T \phi(x) - \rho \tag{2.18}$$

kde $w$ je vektor kolmý na rozhodovaciu hranicu a $\rho$ je bias. Potom rovnica 2.19 ukazuje rozhodovaciu funkciu, ktorú používa OCSVM na identifikáciu normálnych bodov. Funkcia vráti kladnú hodnotu pre normálne body, v prípade odľahlých bodov vráti zápornú hodnotu:

$$f(x) = sgn(g(x)). \tag{2.19}$$

OCSVM sa tradične používa v prostredí s čiastočným dohľadom(semi-supervised). Výstupom algoritmu je binárne označenie vyjadrujúce, či ide o normálny bod alebo o odľahlý bod.

Rovnica 2.20 zobrazuje primárnu úlohu OCSVM:

$$min_{w,\xi,\rho} \frac{\|w\|^2}{2} - \rho + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i \tag{2.20}$$

pre $w^T \phi(x_i) \geq \rho - \xi_i, \xi_i \geq 0$, kde $\xi_i$ je vedľajšia premenná (slack variable) pre bod $i$, ktorá mu umožňuje ležať na druhej strane rozhodovacej hranice, $n$ je veľkosť množiny trénovacích údajov a $\nu$ je parameter regularizácie.

Odpočet od teoretického k matematickému cieľu môže byť vyjadrený vzdialenosťou k rozhodovacej hranici. Tá je definovaná ako:

$$g(x) = 0. \tag{2.21}$$

V tomto kontexte môže byť vzdialenosť k rozhodovacej hranici vypočítaná ako:

$$d(x) = \frac{g(x)}{\|w\|} \tag{2.22}$$

Vzdialenosť, ktorú sa algoritmus snaží optimalizovať, sa teda dá získať zapojením počiatku súradnicové systému do rovnice poskytujúcej $\frac{\rho}{\|w\|}$. To môže byť tiež napísane ako minimalizácia $\frac{\|w\|^2}{2} - \rho$. Druhou časťou primárneho cieľa je minimalizácia vedľajších premenných $\xi_i$ pre všetky body. $\nu$ je parameter regulácie a predstavuje hornú hranicu podielu odľahlých hodnôt a dolnú hranicu počtu podporných vektorov (support vectors). Zmena $\nu$ riadi kompromis medzi $\xi$ a $\rho$.

Pri detekcii odľahlých hodnôt je cieľom priradiť najodľahlejším bodom čo najvyššie skóre odľahlosti. Rovnica 2.23 zobrazuje spôsob výpočtu takéhoto skóre. $g_{max}$ je maximálna vzdialenosť medzi bodmi súboru objektov a rozhodovacou hranicou. Skóre je škálované podľa tejto vzdialenosti tak, že body, ktoré ležia na rozhodovacej hranici, budú mať skóre odľahlosti rovné 1. Ak budú mať objekty skóre odľahlosti vyššie ako 1, tak ide o potenciálne odľahlé hodnoty.

$$f(x) = \frac{g_{max} - g(x)}{g_{max}}. \tag{2.23}$$

Obrázok 2.17 zobrazuje príklad výslednej rozhodovacej hranice v súbore údajov s odľahlými hodnotami. Rozhodovacia hranica je posunutá smerom k odľahlým, čiernym, bodom a v tomto príklade sú navyše vektormi podpory (support vectors). Odľahlé hodnoty sú teda hlavnými prispievateľmi k tvaru rozhodovacej hranice. Zatiaľ čo posunutie hranice nemusí mať veľký vplyv na celkové poradie bodov pri použití rovnice 2.23, tak tvar rozhodovacej hranice áno. Na prekonanie tohto problému vznikli dva prístupy s názvami Robust One-SVM a Eta One-class SVM, ktoré riešia problém, že odľahlé hodnoty významne prispievajú k tvaru rozhodovacie hranice. Obe prístupy sú inšpirované prácou vykonanou s cieľom urobiť tradičné SVM odolnejšie voči šumu v súbore trénovacích údajov. Popis týchto prístupov je v [3].

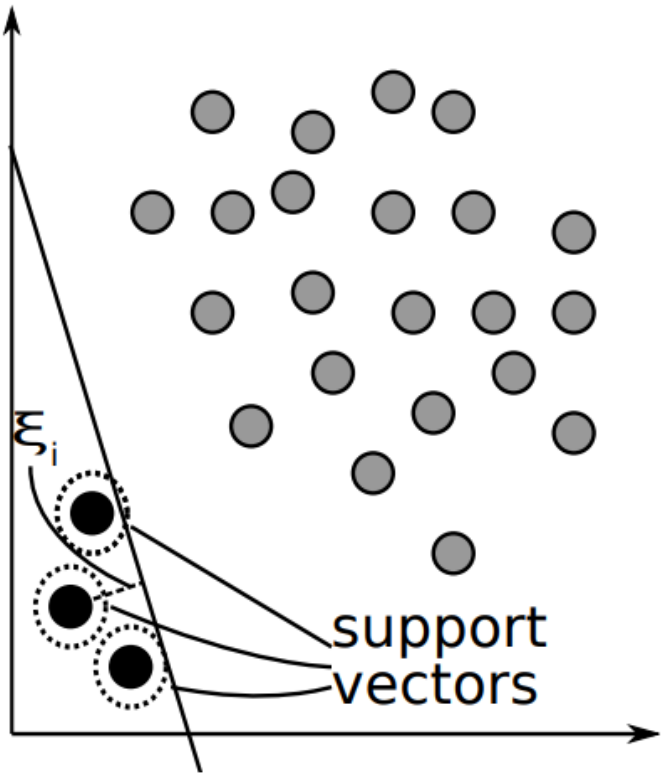
## 2.6 Metódy založené na blízkosti

Metódy založené na blízkosti definujú dátový bod ako odľahlý, ak jeho najbližší susedia sú v dátovom priestore od neho vzdialené, to znamená, že blízkosť objektu k jeho susedom sa výrazne odlišuje od blízkosti väčšiny objektov k ich susedom. Blízkosť dátového bodu sa môže definovať rôznymi spôsobmi, ktoré sa od seba jemne líšia, ale sú si dostatočne podobné na to, aby si zaslúžili jednotné spracovanie v rámci jednej kapitoly. Najbežnejšie spôsoby definovania blízkosti pre analýzu odľahlých hodnôt sú metódy založené na zhlukoch, metódy založené na vzdialenosti a metódy založené na hustote. Informácie v tejto sekcii pochádzajú z [2].

Je zrejmé, že všetky tieto techniky spolu úzko súvisia, pretože sú založené na určitom pojme blízkosti (alebo podobnosti). Hlavný rozdiel je v tom ako sa tá blízkosť definuje. Tieto rôzne spôsoby detekcie odľahlých hodnôt môžu mať rôzne výhody a nevýhody. V mnohých prípadoch sa rozdiely medzi nimi stierajú, keď sa skóre vypočítavá pomocou viacej ako jednej z týchto metód.

### Metódy založené na zhlukoch

Na výpočet skóre odľahlého bodu sa používa to, či sa nachádza v nejakom zhluku, vzdialenosť od ostatných zhlukov, veľkosť najbližšieho zhluku alebo kombinácia týchto spôsobov. Problém zhlukovania ma komplementárny vzťah k problému detekcie odľahlých hodnôt, v ktorom body buď patria do zhlukov, alebo by sa mali považovať za odľahlé. Medzi metódy založené na zhlukoch patrí napríklad metóda CBLOF popísaná v sekcii 2.6.2

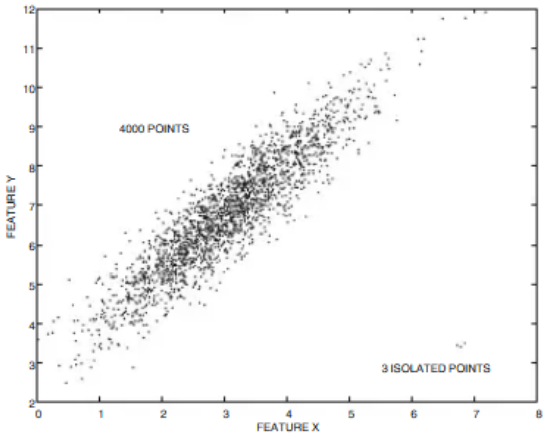



Obrázek 2.17: Príklad rozhodovacej hranice naučenej s OCSVM[3].

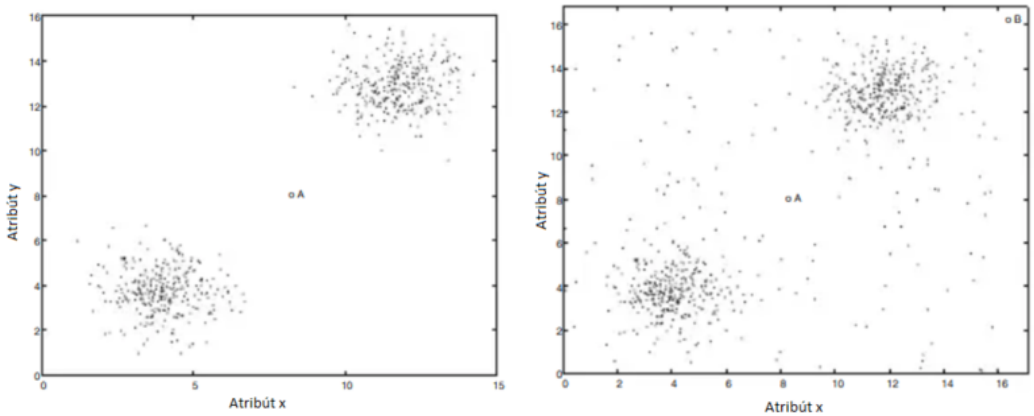
Medzi zhlukovaním a detekciou odľahlých hodnôt existuje dobre známy komplementárny vzťah. Zjednodušený pohľad by bol taký, že každý dátový bod je buď členom zhluku alebo odľahlou hodnotou. Pri zhlukovaní je cieľom rozdeliť body do podmnožín, zatiaľ čo pri detekcii odľahlých hodnôt je cieľom identifikovať body, ktoré sa nezdajú byť súčasťou týchto podmnožín. Niektoré zhlukovacie hlásia odľahlé hodnoty ako vedľajší produkt svojej analýzy. Je dôležité pochopiť, že považovať za odľahlé hodnoty všetky objekty, ktoré nepatria do zhluku nie je správnym riešením. Napríklad dátový bod, ktorý sa nachádza na okraji veľkého zhluku, je veľmi odlišný od bodu, ktorý je úplne izolovaný od všetkých ostatných zhlukov. Okrem toho niekedy sa môžu považovať aj veľmi malé zhluky za odľahlé. Preto sa pri použití zhlukovania na detekciu odľahlých hodnôt používa na výpočet skóre odĺahlosti diferencovanejší prístup (ako iba, či patrí alebo nepatrí do zhluku). Dôležitou výhodou metód zhlukovania je, že sú relatívne rýchle v porovnaní s populárnejšími metódami založenými na vzdialenosti. Tieto metódy majú časť behu, ktorý je kvadratický vzhľadom na dimenzionalitu dát. Na druhú stranu, hlavnou nevýhodou týchto metód je, že v menších súboroch údajov nemusia vždy poskytovať poznatky na požadovanej úrovni. Vo všeobecnosti sú metódy na detekciu odľahlých hodnôt lepšie ak pracujú s pôvodnými dátami a nie z agregovanými ako sú zhlukové centroidy.

### Metódy založené na vzdialenosti

Ďalšou podmnožinou metód založených na blízkosti sú metódy založené na vzdialenosti. Tieto metódy na určenie blízkosti používajú vzdialenosť dátového bodu od jeho $k$-najbližších susedov. Dátové body s veľkou vzdialenosťou $k$-najbližších susedov sa definujú ako odľa-




Obrázek 2.18: Príklad odľahlej hodnoty založenej na zhlukoch[2].




(a) Bez šumu. (b) So šumom.

Obrázek 2.19: Príklad množiny bodov bez šumu a so šumom[2].

hlé. Algoritmy založené na vzdialenosti zvyčajne vykonávajú analýzu s oveľa podrobnejšou granularitou ako metódy založené na zhlukoch a na hustote. Na druhej strane, táto väčšia granularita je často spojená so značnými výpočtovými nákladmi. Medzi najtypickejší prípad metódy založenej na vzdialonsti môžeme považovať metódu kNN, popísanú v sekcii 2.6.3.

Najjednoduchším príkladom je prípad, keď sa ako skóre odľahlej hodnoty uvádza vzdialenosť $k$-najbližších susedov bodu. Vzhľadom na jednoduchosť tejto definície je často jednoduché zovšeobecniť túto techniku na iné typy údajov. Hoci sa zameriavame na viacrozmerné číselné údaje, takéto metódy boli zovšeobecnené takmer na všetky ostatné oblasti, ako sú kategorické údaje, textové údaje, údaje o časových radoch a sekvenčné údaje. Metódy založené na vzdialenostiach pracujú s prirodzeným predpokladom, že vzdialenosti $k$-najbližších susedov odľahlých dátových bodov sú oveľa väčšie ako vzdialenosti normálnych dátových bodov. Hlavný rozdiel medzi metódami zhlukovania a metódami založenými na vzdialenostiach je v granularite analytického procesu. Metódy založené na vzdialenostiach majú vo všeobecnosti vyššiu granularitu analýzy v porovnaní s metódami založenými na zhlukovaní. Táto vlastnosť metód založených na vzdialenostiach môže umožniť rafinovanejšiu schopnosť rozlišovať medzi slabými a silnými odľahlými hodnotami v zašumených súboroch údajov. Napríklad v prípade obrázka 2.19 algoritmus založený na zhlukovaní nebude schopný ľahko rozlíšiť

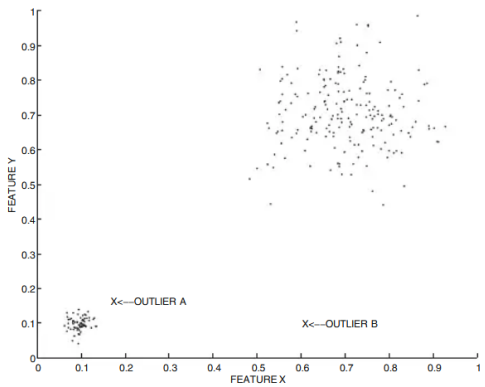
medzi šumom a anomáliami. Je to preto, že vzdialenosť k najbližšiemu centru zhluku pre dátový bod „A“ zostane na obrázkoch 2.19a a 2.19b rovnaká. Na druhej strane, algoritmus $k$-najbližších susedov, sekcia 2.6.3, bude rozlišovať medzi týmito situáciami, pretože šumové dátové body budú zahrnuté medzi vyhodnotenia vzdialenosti. Samozrejme, metódy založené na zhlukovaní je možné modifikovať aj tak, aby zahŕňali efekty šumu. V týchto prípadoch oba prístupy konvergujú k veľmi podobným schémam. Je to preto, lebo tieto dva typy metód sú úzko prepojené. Metódy založené na vzdialenostiach sú tiež schopné identifikovať izolované zhluky úzko prepojených odľahlých hodnôt. Napríklad na identifikáciu malého (anomálneho) zhluku, ktorý obsahuje $k_0$ dátových bodov, je potrebné v algoritme $k$-najbližších susedov použiť hodnotu $k \geq k_0$. Hoci takéto anomálie možno identifikovať aj metódami zhlukovania nastavením prahovej hodnoty počtu bodov v každom zhluku, takéto body môžu niekedy vstúpiť do zhlukov a skresliť príslušné centroidy zhlukov. To môže nepriaznivo ovplyvniť proces bodovania odľahlých hodnôt.

Najvšeobecnejší výstup metód založených na vzdialenostiach je vo forme skóre. Ak sa však vyžaduje skóre odľahlých hodnôt každého dátového bodu, potom algoritmus vyžaduje operácie presne úmerné $N^2$. V binárnej rozhodovacej verzii identifikácie, či je dátový bod odľahlý, je možné použiť rôzne typy orezávacích a indexovacích štruktúr na podstatné urýchlenie prístupu.

### Metódy založené na hustote

Na určenie lokálnej hustoty sa používa počet ďalších bodov v rámci špecifikovanej lokálnej oblasti (oblasť mriežky alebo oblasť založená na vzdialenosti) dátového bodu. Tieto hodnoty lokálnej hustoty sa môžu previesť na skóre odľahlosti. Na odhad hustoty sa môžu použiť aj iné metódy založené na jadre alebo štatistické metódy. Hlavným rozdielom medzi metódami zhlukovania a metódami založenými na hustote je, že metódy zhlukovania rozdeľujú dátové body do skupín, zatiaľ čo metódy založené na hustote rozdeľujú dátový priestor.

Metódy založené na hustote využívajú počet bodov v určitých oblastiach priestoru na definovanie odľahlých hodnôt. Sú veľmi úzko späté so zhlukovaním a metódami založenými na vzdialenostiach. V skutočnosti sa niektoré z týchto algoritmov dajú ľahšie považovať za metódy založené na zhlukovaní alebo vzdialenostiach, v závislosti od toho, ako sú prezentované. Je to preto, že pojmy vzdialenosti, zhlukovanie a hustota sú úzko prepojené a vzájomne závislé. Mnohé z týchto algoritmov zisťujú odľahlé hodnoty citlivé na lokalitu. V tejto časti sa rozoberajú lokálne aj globálne algoritmy. Aby sme pochopili túto špeci-




Obrázek 2.20: Vplyv lokálnej hustoty na odľahlé hodnoty[2].

fickú problematiku, zoberme si špecifický príklad súboru údajov s meniacou sa hustotou na obrázku 2.20. Obrázok obsahuje dve odľahlé hodnoty označené „A“ a „B“. Okrem toho obsahuje dva zhluky, z ktorých jeden je oveľa redší ako druhý. Je zrejmé, že odľahlé miesto „A“ nemôže byť objavené algoritmom založeným na vzdialenosti, pokiaľ algoritmus nepoužije menší prah vzdialenosti. Ak sa však použije menšia prahová hodnota vzdialenosti, potom môže byť mnoho dátových bodov v redšom zhluku nesprávne vyhlásených za odľahlé hodnoty. To tiež znamená, že poradie vrátené algoritmom založeným na vzdialenosti je nesprávne, ak existuje významná heterogenita v lokálnych rozdeleniach bodov.

### 2.6.1 LOF: Local Outlier Factor

V prvom rade ide o algoritmus, ktorý hľadá odľahlé hodnoty na základe lokálnej odchýlky daného objektu vzhľadom na svojich susedov. Informácie o algoritme pochádzajú z článku [24]. Najprv zavediem parameter $k$, čo je počet susedov, ktorých výpočet LOF berie do úvahy. LOF je výpočet, ktorý sa pozerá na susedov určitého bodu, aby zistil jeho hustotu a neskôr ju porovnal s hustotou ostatných bodov v dátovej sade. Použitie správneho čísla $k$ nie je priamočiare. Zatiaľ čo malé $k$ má viac lokálne zameranie, t. j. pozerá len na blízke body, je chybnejšie, keď má v údajoch veľa šumu. Veľké $k$ však môže minúť lokálne odľahlé hodnoty. Ako prvý pojem si zavedieme $k$-vzdialenosť objektu $p$ a tiež $k$-vzdialenosť okolia objektu $p$. Pre ľubovolné kladné celé číslo $k$ je $k-distance(p)$ definovaná ako vzdialenosť $d(p, o)$ medzi $p$ a objektom $o \in D$ následovne:

- pre najmenej $k$ objektov $o' \in D - \{p\}$ platí, že $d(p, o') \leq d(p, o)$ a
- pre najviac $k-1$ objektov $o' \in D - \{p\}$ platí, že $d(p, o') < d(p, o)$.

Máme $k$-vzdialenosť objektu $p$, potom $k$-vzdialenosť okolia objektu $p$ obsahuje každý objekt ktorého vzdialenosť od $p$ nie je väčšia ako $k$-vzdialenosť, formálne definovaná ako:

$$N_{k\text{-}distance}(p) = \{q \in D - \{q\} | d(p, q) \leq k\text{-}distance(p)\}. \tag{2.24}$$

Objekty $q$ sa nazývajú $k$-najbližších susedov objektu $p$.

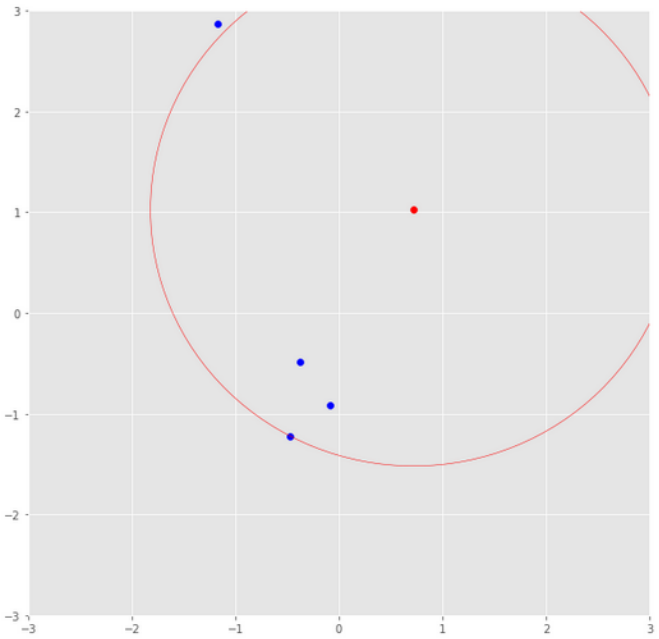
Ďalší pojem, ktorý nás zaujíma je dosiahnuteľná vzdialenosť (reachability distance) objektu $p$ vzhľadom na objekt $o$. Nech $k$ je prirodzené číslo, potom dosiahnuteľná vzdialenosť objektu $p$ vzhľadom na objekt $o$ je definovaná ako

$$reach\text{-}dist_k(p, o) = max\{k\text{-}distance(o), d(p.o)\} \tag{2.25}$$

Obrázok 2.22 ilustruje myšlienku dosiahnuteľnej vzdialenosti s $k = 4$. Intuitívne, ak je objekt $p$ ďaleko od objektu $o$ (napr. $p_2$ na obrázku), potom dosiahnuteľná vzdialenosť medzi týmito dvoma objektami je jednoducho ich skutočná vzdialenosť. Ak sú však „dostatočne“ blízko (napr. $p_1$ na obrázku), skutočná vzdialenosť sa nahradí $k$-vzdialenosťou objektu $o$. Dôvodom je, že štatistické fluktuácie $d(p, o)$ pre všetky $p$ blízko k $o$ možno výrazne znížiť. Sila tohto vyhladzovacieho efektu môže byť riadená parametrom $k$. Čím vyššia je hodnota $k$, tým sú vzdialenosti dosiahnuteľné pre objekty v rovnakom okolí podobné.

Zatiaľ sme definovali $k\text{-}vzdialenos(p)$ a $reach\text{-}dist_k(p)$ pre akékoľvek kladné celé číslo $k$. Ale na účely definovania odľahlých hodnôt sa zameriavame na špecifickú inštanciu $k$, ktorá nás spája späť so zhlukovaním na základe hustoty. V typickom zhlukovacom algoritme založenom na hustote existujú dva parametre, ktoré definujú pojem hustoty:

- parameter MinPts špecifikujúci minimálny počet objektov;




Obrázek 2.21: Zobrazenie $k$-vzdialenosti pre červený bod ak k=3[24].

- parameter určujúci objem (vynásobenie rozsahov každej dimenzie).

Tieto dva parametre určujú prah hustoty pre fungovanie zhlukovacích algoritmov. To znamená, že objekty alebo oblasti sú spojené, ak ich hustota susedstva prekročí daný prah hustoty. Na detekciu odľahlých hodnôt založených na hustote je však potrebné porovnávať hustoty rôznych množín objektov, čo znamená, že hustotu množín objektov musíme určovať dynamicky. Preto ponechávame MinPts ako jediný parameter a hodnoty $reach\text{-}dist_{MinPts}(p, o)$ pre $o \int N_{MinPts}(p)$ používame ako mieru objemu na určenie hustoty v okolí objektu $p$.

Ďalej je nutné si zadefinovať lokálnu dosiahnuteĺnú hustotu objektu $p$ a to tak, že

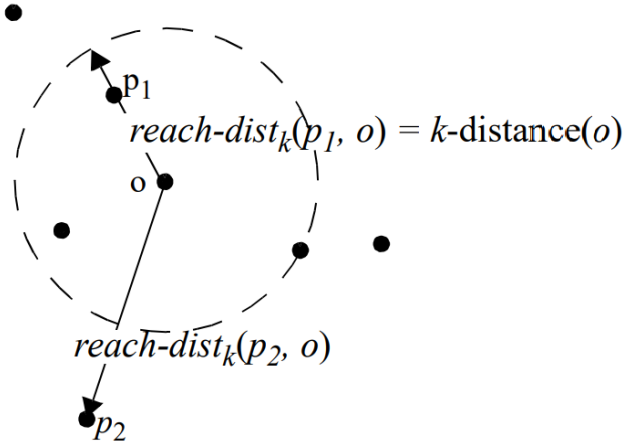
$$lrd_{MinPts}(p) = 1/\left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)}{|N_{MinPts}(p)|}\right) \tag{2.26}$$

Intuitívne je hustota lokálnej dosiahnuteľnosti objektu $p$ inverzná k priemernej vzdialenosti dosiahnuteľnosti na základe $MinPts$-najbližších susedov $p$. Lokálna hustota môže byť $\infty$, ak sú všetky vzdialenosti dosiahnuteľnosti v súčte 0. K tomu môže dôjsť pre objekt $p$, ak existujú aspoň $MinPts$ objektov, odlišné od $p$, ale zdieľajúce rovnaké priestorové súradnice, t. j. najmenej $MinPts$ duplikátov $p$ v množine údajov. Pre jednoduchosť nebudeme tento prípad riešiť explicitne, ale jednoducho predpokladáme, že neexistujú žiadne duplikáty. Aby sme sa vyrovnali s duplikátmi, môžeme náš pojem susedstva založiť na $k$-rozdielnej vzdialenosti, definovanej analogicky ako $k$-vzdialenosť, s dodatočnou požiadavkou, aby existovalo aspoň $k$ objektov s rôznymi priestorovými súradnicami.

A nakoniec LOF pre objekt $p$ je definovaný ako:

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|} \tag{2.27}$$

Odľahlý faktor objektu $p$ zachytáva mieru, do akej nazývame $p$ odľahlou hodnotou. Je to priemer pomeru hustoty lokálnej dosiahnuteľnosti $p$ a hustoty $p$ s $MinPts$-najbližšími




Obrázek 2.22: Zobrazenie $reach\text{-}dist_k(p_1, o)$ a $reach\text{-}dist_k(p_2, o)$ s $k = 4$ [8].

susedmi. Je ľahké vidieť, že čím nižšia je hustota lokálnej dosiahnuteľnosti $p$, a čím vyššie sú hustoty lokálnej dosiahnuteľnosti $MinPts$-najbližších susedov $p$ , tým vyššia je hodnota LOF pre $p$.

### 2.6.2 CBLOF: Clustering-Based Local Outlier Factor

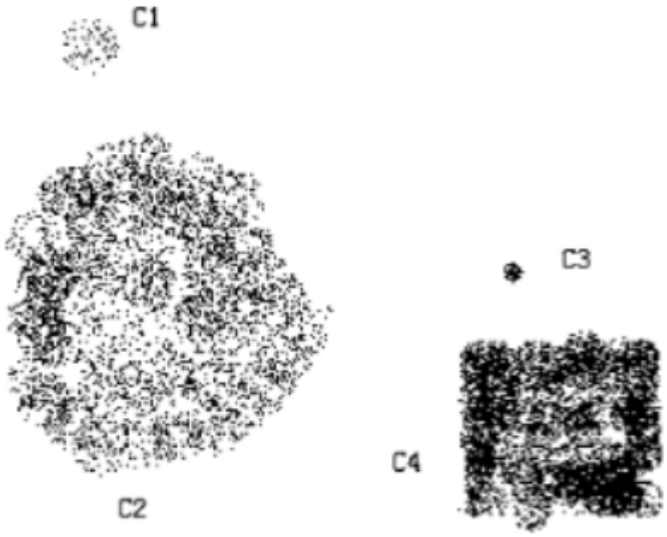
V tejto časti predstavíme definíciu odľahlej hodnoty pochádzajúcej z [13]: Clustering-Based Local Outlier Factor-CBLOF. Pred formálnou definíciou novej definície odľahlých hodnôt najprv uvedieme príklad na ilustráciu našich základných myšlienok, vzhľadom na 2D súbor údajov na obrázku 2.23. Na tomto obrázku sú štyri zhluky, C1, C2, C3 a C4. Je zrejmé, že dátové body v C1 a C3 by sa mali považovať za odľahlé hodnoty a mali by sa zachytiť navrhovanými definíciami. Intuitívne nazývame dátové body v C1 a C3 odľahlými hodnotami, pretože nepatria do zhluku C2 a C4. Preto je rozumné definovať odľahlé hodnoty z pohľadu zhlukov a identifikovať tie dátové body, ktoré neležia v žiadnom z veľkých zhlukoch, ako odľahlé hodnoty. V tomto príklade je počet dátových bodov v C2 a C4 dominantný. Na identifikáciu odľahlej hodnoty priradíme každému objektu faktor odľahlosti, konkrétne CBLOF, ktorý sa meria veľkosťou zhluku, do ktorého objekt patrí, ako aj vzdialenosťou medzi objektom a jeho najbližším zhlukom (ak objekt leží v malom zhluku). Algoritmus zhlukovania, ktorý sa použije na rozdelenie súboru údajov do neprekrývajúcich sa zhlukov, môžeme voliť volne. Jedinou požiadavkou na vybraný algoritmus zhlukovania je, že by mal mať schopnosť produkovať dobré výsledky zhlukovania.

Kritický problém, ktorý je potrebné vyriešiť pred definovaním CBLOF, je to, ako zistiť, či je zhluk veľký alebo malý. Tento problém je diskutovaný v definícii 2.6.1.

**Definícia 2.6.1** (Veľký a malý zhluk)**.** Predpokladajme $C = \{C_1, C_2, ..., C_k\}$ je množina zhlukov v poradí $|C_1| \geq |C_2| \geq ... \geq |C_k|$. Vzhľadom na numerické parametre $\alpha$, určuje od koľkých bodov sa zhluk považuje za veľký, a $\beta$, ktorá určuje, koľkokrát je veľký zhluk väčší ako malý zhluk, definujeme $b$ ako hranicu pre veľký a malý zhluk, ak jedno z nasledujúcich tvrdení platí:

$$(|C_1| + |C_2| + ... + |C_b| \geq |D|^*\alpha) \tag{2.28}$$

$$|C_b|/|C_{b+1}| \geq \beta \tag{2.29}$$




Obrázek 2.23: Príklad 2D datasetu obsahujúci 4 zhluky[13].

Potom, množina veľkých zhlukov je definovaná ako $LC = \{C_i | i \leq b\}$ a množina malých zhlukov ako $SC = \{C_j | j > b\}$.

Definícia 2.6.1 udáva spôsob na rozlíšenie veľkých a malých zhlukov. Vzorec 2.28 zohľadňuje skutočnosť, že väčšina dátových bodov v súbore údajov nie je odľahlá. Preto by sa zhluky, ktoré obsahujú veľkú časť dátových bodov, mali považovať za veľké zhluky. Napríklad, ak je $\alpha$ nastavená na 90 %, máme v úmysle považovať zhluky obsahujúce 90 % dátových bodov za veľké zhluky. Vzorec 2.29 zohľadňuje skutočnosť, že veľké a malé zhluky by mali mať výrazné rozdiely vo veľkosti. Napríklad, ak nastavíme $\beta$ na 5, veľkosť akéhokoľvek zhluku v $LC$ je najmenej päťkrát väčšia ako veľkosť zhluku v $SC$.

**Definícia 2.6.2** (Clustering-Based Local Outlier Factor)**.** Predpokladajme $C = \{C_1, C_2, ..., C_k\}$ je množina zhlukov v poradí $|C_1| \geq |C_2| \geq ... \geq |C_k|$ a významy premenných $\alpha, \beta, b, LC$ a $SC$ sú rovnaké ako v definícii 2.6.1. Pre každý záznam $t$ je CBLOF definovaný ako:

$$CBLOF(t) = \begin{cases} |C_i|^* min(distance(t, C_j)) & \text{kde } t \in C_i, C_i \in SC \text{ a } C_j \in LC \text{ pre } j \in \langle 1, b \rangle \\ |C_i|^* distance(t, C_j) & \text{kde } t \in C_i \text{ a } C_i \in LC \end{cases} \tag{2.30}$$

Z definície 2.6.2 je CBLOF nejakého záznamu, určený veľkosťou jeho zhluku a vzdialenosťou medzi záznamom a jeho najbližším zhlukom (ak tento záznam leží v malom zhluku) alebo vzdialenosťou medzi záznamom a zhlukom, do ktorého patrí (ak tento záznam patrí do veľkého zhluku), čo dáva dôležitosť lokálnemu správaniu údajov. Na výpočet vzdialenosti medzi záznamom a zhlukom stačí použiť mieru podobnosti používanú v zhlukovacom algoritme.

### 2.6.3 k-Nearest Neighbor

Metóda $k$-Nearest Neighbor (kNN) vychádza z rovnako pomenovanej metódy, ktorá bola určená na klasifikáciu alebo regresiu. Popis metódy vychádza z [4] a [25]. Majme označenie

$D^k(p)$, ktoré označuje vzdialenosť bodu $p$ od jeho $k^{th}$ najbližšieho suseda. Body zoraďujeme na základe ich $D^k(p)$ vzdialenosti, čo vedie k nasledujúcej definícii pre odľahlé hodnoty:

**Definícia 2.6.3.** Daný je vstupný súbor údajov s $N$ prvkami, parameter $n$, označujúci celkový počet odľahlých hodnôt o ktoré máme záujem, parameter $k$, označujúci počet susedov a bod $p$ je odľahlá hodnota, ak nie je viacej ako $n-1$ iných bodov $p'$ pre ktoré platí, že $D^k(p') > D^k(p)$.

Inými slovami, ak zoradíme body podľa ich vzdialenosti $D^k(p)$, tak horných $n$ bodov je považovaných za odľahlé hodnoty. Na vypočítanie vzdialenosti medzi bodmi je možné použiť ľubovoľnú metriku vzdialenosti ako sú manhattanovská alebo euklidovská vzdialenosť. Alternatívne, môžeme použiť nečíselné metódy na meranie vzdialenosti, napríklad pri dokumentoch.

Na základe definície 2.6.3 pre odľahlé hodnoty, je možné ich zoradiť na základe ich $D^k(p)$. Odľahlé hodnoty s vyšším $D^k(p)$ majú málo bodov okolo seba a tým pádom ide o „silnejšie“ odľahlé hodnoty.

Nakoniec je nutné dodať, že existujú varianty tohto algoritmu ako sú Average kNN a Median kNN. Na rozdiel od klasického kNN, ktoré si vracia vzdialenosť len od $k$-najbližšieho suseda a to vracia ako skóre odľahlosti, tak *Average kNN* berie $k$-najbližších susedov a ako skóre odľahlosti vracia priemer vzdialenosti. *Median kNN* funguje podobne, len miesto priemeru používa hodnotu mediánu $k$-najbližších susedov.

## 2.7 Kombinované metódy detekcie odľahlých hodnôt

Kombinovaná analýza je populárna metóda používaná na zlepšenie presnosti rôznych dolovacích algoritmov. Kombinované metódy spájajú výstupy viacerých algoritmov alebo základných detektorov s cieľom vytvoriť unifikovaný výstup. Základná myšlienka tohto prístupu spočíva v tom, že niektoré algoritmy budú lepšie na určitej podmnožine bodov, zatiaľ čo iné algoritmy budú lepšie na iných podmnožinách bodov. Kombinácia metód však často dokáže fungovať robustnejšie v celom rozsahu, pretože dokáže kombinovať výstupy viacerých algoritmov. Medzi základné algoritmy, ktoré reprezentujú kombinované metódy patrí Isolation Forest, popísaný v 2.7.1 a Feature Bagging, popísaný v 2.7.2. Informácie v tejto sekcii pochádzajú z knihy[2].

Kombinovaná analýza sa často používa v rôznych aplikáciách dolovania dát a strojového učenia, ako je zhlukovanie, klasifikácia a analýza odľahlých hodnôt. Okrem toho sa používa aj na rôzne typy údajov, ako sú viacrozmerné údaje, údaje v časových radoch alebo údaje s hodnoteniami. Všadeprítomnosť týchto metód v rôznych problémových oblastiach a typoch údajov je výsledkom ich relatívnej úspešnosti v rôznych prostrediach.

Pri vytváraní kombinácie metód na detekciu odľahlých hodnôt sú dve kľúčové voľby:

- Výber základného detektoru: Výber základného detektoru je jedným z prvých krokov tvorby kombinovaných metód. Tento základný detektor môže byť úplne odlišný od ostatných použitých detektorov, môže mať iné nastavenie parametrov alebo môže používať upravené základne dáta.

- Metodika normalizácie a kombinácie skóre: Rôzne detektory môžu vytvárať skóre na rôznych stupniciach. Napríklad priemerný detektor $k$-najbližších susedov vytvorí hrubé skóre vzdialenosti, zatiaľ čo algoritmus LOF vytvorí normalizovanú hodnotu.

Okrem toho, hoci všeobecnou konvenciou je vypisovať väčšie skóre pre odľahlé hodnoty, niektoré detektory používajú opačnú konvenciu a vypisujú menšie skóre pre odľahlé hodnoty. Preto je žiaduce previesť skóre z rôznych detektorov na normalizované hodnoty, ktoré možno zmysluplne kombinovať. Po normalizácii skóre je dôležitou otázkou výber kombinačnej funkcie použitej na výpočet skóre súboru bodov. Medzi najčastejšie voľby patria priemerovanie a maximalizácia kombinačných funkcií.

Väčšina detektorov odľahlých hodnôt je navrhnutá tak, aby pracovala s jedným alebo viacerými základnými detektormi a využívala opakovanú aplikáciu týchto základných detektorov s cieľom generovať skóre so zlepšenou odchýlkou alebo rozptylom. Kombinované metódy detekcie odľahlých hodnôt môžu byť kategorizované do dvoch skupín:

- Metódy zamerané na model - V tomto prípade rôzne základné detektory súboru zodpovedajú rôznym modelom (algoritmom), rôznym nastaveniam parametrov toho istého modelu alebo rôznym náhodným inštanciám toho istého modelu.
- Metódy zamerane na dáta - V tomto prípade rôzne základné detektory súboru zodpovedajú aplikácii toho istého modelu na rôzne derivácie údajov. Tieto rôzne derivácie môžu byť vytvorené výberom vzoriek z údajov, výberom náhodných projekcií z údajov, vážením dátových bodov, vážením rozmerov alebo pridaním šumu do údajov.

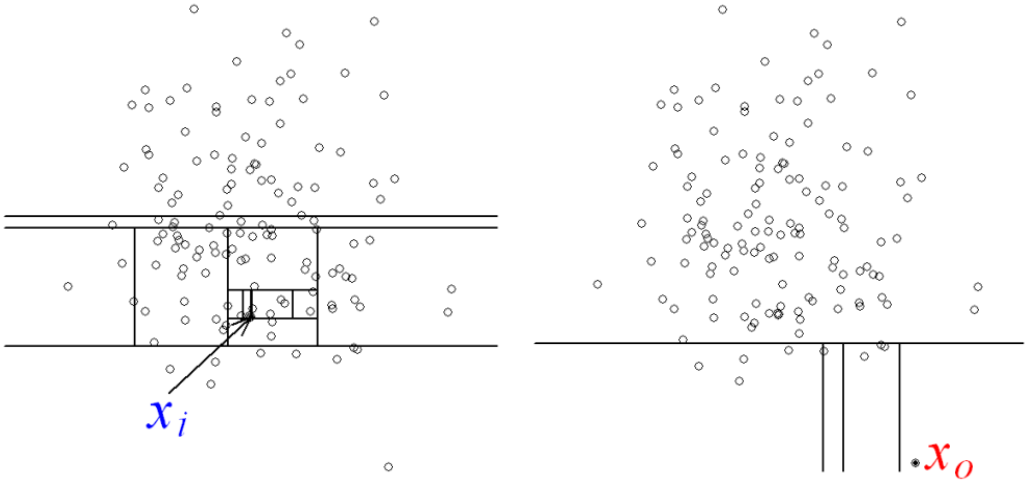
Avšak je tu aj iná cesta ako kategorizovať kombinované metódy na základe úrovne závislosti medzi jednotlivými metódami. Táto metodika kategorizácie detektorov je následovna:

- Nezávislé kombinácie - V tomto prípade sa jednotlivé komponenty základného detektora vykonávajú nezávisle od seba. Výsledky z rôznych detektorov sa potom skombinujú. Do tejto kategórie patrí väčšina existujúcich metód.
- Sekvenčné kombinácie - V tomto prípade sa základné detektory súboru vykonávajú jeden po druhom s cieľom vytvoriť postupne vylepšené modely. Príkladom takéhoto algoritmu je algoritmus, v ktorom sa z údajov postupne odstraňujú odľahlé hodnoty s cieľom vytvoriť lepší model normálnych údajov. Buď sa uvádzajú výsledky z finálneho vykonania súboru, alebo sa výsledky z rôznych komponentov skombinujú do jednotnej predpovede.

Nezávislé súbory môžu byť modelovo alebo dátovo orientované; podobne sekvenčné súbory môžu byť modelovo alebo dátovo orientované. Bolo navrhnutých veľké množstvo modelov, ktoré patria do týchto rôznych kategórií. Sekvenčné súbory však bývajú menej preskúmané ako iné typy modelov. Čiastočne je to preto, že sekvenčné súbory sú vo svojej podstate určené pre úlohy, ako je klasifikácia, v ktorých je k dispozícii základná pravda (ground-truth), ktorá sa môže použiť pre zlepšenie metód pri ďalšom vykonávaní. Ďalej budú predstavené dve konkrétne metódy a to Isolation Forest a Feature Bagging.

### 2.7.1 Isolation Forest

Ďalšou metódou ktorú si predstavíme bude metóda Isolation Forest (izolačné lesy) - iForest, informácie v tejto kapitole pochádzajú z [21]. Táto metóda vytvára súbor izolačných stromov pre daný súbor objektov a za odľahlé považuje tie objekty, ktoré majú krátku priemernú dĺžku ciest v stromoch. Izolačný strom predstavuje delenie priestoru na dve časti a pokračuje v delení dovtedy, kým sa v danej časti nenachádza iba jeden bod. Keďže odľahlých hodnôt




(a) Izolovanie bodu $x_i$. (b) Izolovanie bodu $x_o$.

Obrázek 2.24: Príklad pre zobrazenie izolovania dvoch bodov [21].

je málo a sú rôzne od ostatných, tak sú náchylnejšie na izoláciu. V náhodnom strome vyvolanom údajmi sa rozdelenie inštancií opakuje rekurzívne, kým sa neizolujú všetky inštancie. Toto náhodné rozdelenie vytvára viditeľné kratšie cesty pre odľahlé hodnoty, pretože:

- menší počet prípadov odľahlých hodnôt vedie k menšiemu počtu oddelení – kratšie cesty v stromovej štruktúre a
- výskyty s rozlíšiteľnými hodnotami atribútov sú s väčšou pravdepodobnosťou oddelené v prvých rozdeleniach.

Preto, keď les náhodných stromov spoločne vytvára kratšie dĺžky ciest pre niektoré konkrétne body, potom je veľmi pravdepodobné, že ide o odľahlé hodnoty.

Aby sme demonštrovali myšlienku, že anomálie sú náchylnejšie na izoláciu pri náhodnom rozdelení, ilustrujeme príklad na obrázkoch 2.24a a 2.24b na vizualizáciu náhodného rozdelenia normálneho bodu oproti odľahlému. Je možné vidieť, že normálny bod $x_i$ vo všeobecnosti vyžaduje na izolovanie viacej oddelení. Opak platí pre odľahlý bod $x_o$, ktorý vo všeobecnosti vyžaduje na izolovanie menej oddelení. V tomto príklade sa delenia generujú náhodným výberom atribútu a následným náhodným výberom delenej hodnoty medzi maximálnou a minimálnou hodnotou vybratého atribútu. Keďže rekurzívne rozdelenie môže byť reprezentované stromovou štruktúrou, počet oddelení potrebných na izoláciu bodu je ekvivalentný dĺžke cesty od koreňového uzla po koncový uzol. V tomto príklade je dĺžka cesty $x_i$ väčšia ako dĺžka cesty $x_o$.

Keďže každé delenie je generované náhodne, jednotlivé stromy sa generujú s rôznymi sadami oddelení. Spriemerujeme dĺžky ciest na viacerých stromoch, aby sme našli očakávanú dĺžku cesty. Obrázok 2.25 ukazuje, že priemerné dĺžky ciest $x_0$ a $x_i$ sa ustáľujú, keď sa zvyšuje počet stromov. Pri použití 1 000 stromov sa priemerné dĺžky ciest $x_o$ a $x_i$ ustáľujú na hodnotách 4,02 a 12,82. Ukazuje to, že odľahlé hodnoty majú dĺžky ciest kratšie ako normálne prípady.

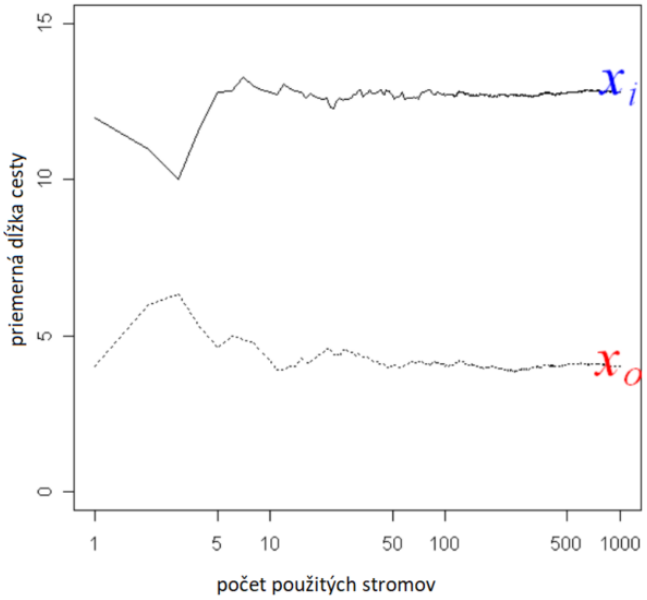
**Definícia 2.7.1** (Isolation Tree)**.** Nech $T$ je uzol izolačného stromu. Potom $T$ je buď koncový uzol bez potomka alebo vnútorný uzol s jedným testom a práve dvoma potomkami,

uzlami $(T_l, T_r)$. Test pozostáva z atribútu $q$ a deliacej hodnoty $p$ takej, že test $q < p$ rozdelí dáta na $T_l$ a $T_r$.

Majme vzorku údajov $X = x_1, ..., x_n$ z $n$ výskytov z množstvom atribútov $d$. Aby sme vytvorili izolovačný strom (isolation tree), rekurzívne rozdeľujeme $X$ náhodným výberom atribútu $q$ a deliacej hodnoty $p$, kým buď:

- strom nedosiahne výškový limit,
- $|X| = 1$ alebo
- všetky údaje v $|X|$ majú rovnaké hodnoty.

Izolovaný strom je správny binárny strom, kde každý uzol v strome má presne nula alebo dva dcérske uzly. Za predpokladu, že všetky inštancie sú odlišné, každá inštancia je izolovaná od koncového uzla, keď izolovaný strom úplne vyrastie, v takom prípade je počet koncových uzlov $n$ a počet vnútorných uzlov je $n - 1$; celkový počet uzlov izolačného stromu je $2n - 1$.




Obrázek 2.25: Zobrazenie priemernej dĺžky cesty pre body $x_i$ a $x_o$ [21].

Úlohou detekcie odľahlých hodnôt je poskytnúť poradie, ktoré odráža stupeň odľahlosti. Jedným zo spôsobov detekcie odľahlých hodnôt je teda triedenie údajových bodov podľa dĺžky ich cesty alebo skóre odľahlosti; a anomálie sú body, ktoré sú zoradené na začiatku zoznamu. Dĺžku cesty a skóre odľahlosti definujeme nasledovne.

**Definícia 2.7.2** (Dĺžka cesty)**.** Dĺžka cesty $h(x)$ údaju $x$ sa meria počtom hrán prechádzajúcich z koncového uzla $x$ do koreňového uzla v izolovanom strome.

Pre každú metódu detekcie odľahlých hodnôt je dôležite poskytnúť skóre odľahlosti. Ťažkosť pri odvodení takéhoto skóre z $h(x)$ je v tom, že zatiaľ čo maximálna možná výška izolovaného stromu rastie rádovo $n$, priemerná výška rastie rádovo $logn$. Normalizácia $h(x)$ ani jedným spôsobom nie je možná, keďže buď nie je ohraničená, alebo sa nedá priamo porovnávať.

Keďže izolované stromy majú rovnakú štruktúru ako binárne vyhľadávacie stromy, tak dohad priemernej hodnoty $h(x)$ môže byť vypočítaný ako

$$c(n) = 2H(n-1) - (2(n-1)/n) \tag{2.31}$$

kde $H(i)$ je číslo ktoré môže byť vypočítané ako $ln(i) + 0.5772156649$(Eulerova konštanta). Ak $c(n)$ je priemer z $h(x)$ pre $n$, potom môže byť použité na normalizáciu $h(x)$. Skóre odľahlosti $s$ pre bod $x$ je definované ako:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{2.32}$$

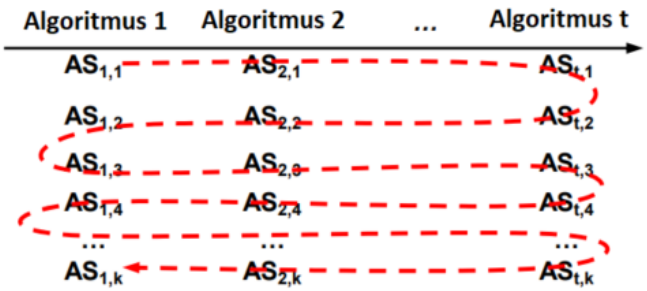
kde $E(h(x))$ je priemer z $h(x)$ cez všetky varianty izolačných stromov. Použitím skóre odľahlosti $s$ sme schopný tvrdiť, že:

- ak body vracajú $s$ veľmi blízke k 1, potom môžeme tvrdiť, že určite ide o anomálie,
- ak body majú $s$ oveľa menšie ako 0,5, potom je bezpečné tvrdiť, že ide o normálnu hodnotu a
- ak všetky body vracajú hodnotu $s$ približne rovnú 0,5 potom celý dataset nevykazuje žiadne výrazné odľahlé hodnoty.

### 2.7.2 Feature Bagging

V tejto sekcii si opíšeme metódu detekcie odľahlých hodnôt, ktorá kombinuje výsledky detekcie odľahlých hodnôt aplikovaných na rôzne podmnožiny atribútov z pôvodnej dátovej sady, informácie pochádzajú z [18]. Metódy, ktoré sa môžu použiť v podstate nie sú nijak obmedzené, avšak najlepšie výsledky sa dostávajú, ak sú použité metódy založené na hustote.

Predstavíme dve techniky na kombináciu algoritmov detekcie odľahlých hodnôt, ide o techniku Breadth-First a techniku kumulatívneho súčtu. Každá procedúra kombinovania techník detekcie odľahlých hodnôt prebieha v sérii $T$ kôl, aj keď tieto kolá môžu prebiehať paralelne pre rýchlejšiu realizáciu. V každom kole $t$ je volaný algoritmus detekcie odľahlých hodnôt a prezentovaný s inou sadou atribútov $F_t$, ktoré sa používajú pri výpočte vzdialenosti. Súbor atribútov $F_t$ je náhodne vybraný z pôvodného súboru dát tak, že počet atribútov v $F_t$ je tiež náhodne vybraný medzi $\lfloor d/2 \rfloor$ a $(d-1)$, kde $d$ je počet atribútov v pôvodnom súbore údajov. Keď je zvolený počet atribútov $N_t$ v $F_t$, atribúty $N_t$ sa náhodne vyberú bez výmeny z pôvodnej sady atribútov. Následne bude použitý algo-




Obrázek 2.26: Zobrazenie práce Breadth-First pre kombináciu skóre odľahlých hodnôt [18].

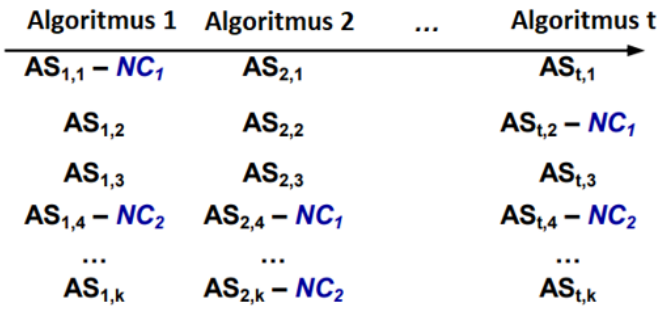
ritmus na detekciu odľahlých hodnôt $O_t$ na každú podmnožiny atribútov $F_t$. Výsledkom

každého algoritmu detekcie odľahlých hodnôt je rôzny vektor skóre odľahlosti $AS_t$, ktorý odráža pravdepodobnosť, že každý dátový záznam zo súboru údajov $S$ je odľahlou hodnotou. Napríklad, ak $AS_t(i) > AS_t(j)$, dátový záznam $x_i$ má vyššiu pravdepodobnosť, že bude odľahlý ako dátový záznam $x_j$. Na konci postupu, po $T$ kolách, existuje $T$ vektorov skóre odľahlosti, z ktorých každý zodpovedá jednému algoritmu detekcie odľahlých hodnôt. Funkcia $COMBINE(AS_t)$, kde $t = 1, ..., T$, sa potom použije na spojenie týchto vektorov skóre odľahlosti $AS_t$ do jedinečného vektora skóre odľahlosti $AS_{FINAL}$, ktorý sa nakoniec používa na označenie konečnej pravdepodobnosti, že bude dátový bod odľahlý.

Problém kombinovania vektorov odľahlého skóre je, že neexistuje označenie, ktoré by pomohlo pochopiť, aké relevantné sú výsledky jednotlivých algoritmov a zároveň poradie výsledkov z jednotlivých algoritmov je dôležité v procese kombinovania, pretože dáva predstavu o relevantnosti výsledku. Vysvetlíme si dva varianty funkcie $COMBINE$, ktorá integruje výstupy viacerých algoritmov detekcie odľahlých hodnôt.

Prvá metóda kombinovania, ktorú si predstavíme, Breadth-First, najprv zoradí všetky odľahlé detekčné vektory $AS_t$ do zoradených vektorov $SAS_t$ a vracia indexy $Ind_t$, ktoré prepájajú zoradené vektory a originálne dáta. Napríklad $Ind_t(1) = k$ znamená, že v $t$-tom vektore skóre detekcie odľahlej hodnoty $AS_t$ má dátový záznam $x_k$ najvyššie skóre odľahlosti $AS_t(k)$. Na obrázku 2.26 teda $AS_{1,1}$ zodpovedá dátovému záznamu, ktorý je podľa Algoritmu 1 hodnotený ako najpravdepodobnejšia odľahlá hodnota, $AS_{1,2}$ zodpovedá dátovému záznamu, ktorý je podľa Algoritmu 1 hodnotený ako druhý najpravdepodobnejší, atď.

Po zoradení všetkých odľahlých vektorov skóre $AS_t$, prístup Breadth-First jednoducho vezme dátové záznamy s najvyšším skóre anomálií zo všetkých odľahlých detekčných algoritmov (skóre $AS_{1,1}$, $AS_{2,1}$, $AS_{3,1}$, ..., $AS_{t,1}$ na obrázku 2.26) a vloží ich indexy do vektora $Ind_{FINAL}$, potom vezme dátové záznamy s druhým najvyšším skóre odľahlosti (skóre $AS_{1,2}$, $AS_{2,2}$, $AS_{3,2}$, ..., $AS_{t,2}$ na obrázku 2.26) a pripojí ich indexy na koniec vektora $Ind_{FINAL}$ a tak ďalej. Ak je index aktuálneho dátového záznamu už vo vektore $Ind_{FINAL}$, znova sa nepridáva. Na konci metódy Breadth-First index $Ind_{FINAL}$ obsahuje indexy dátových záznamov, ktoré sú zoradené podľa ich pravdepodobnosti, že sú odľahlé, a vektor $AS_{FINAL}$ obsahuje tieto pravdepodobnosti.




Obrázek 2.27: Zobrazenie práce Cumulative Sum pre kombináciu skóre odľahlých hodnôt [18].

Konečné výsledky metódy Breadth-First sú vo všeobecnosti citlivé na poradie algoritmov detekcie odľahlých hodnôt. Rozdiely sú však malé, pretože variácie sa môžu vyskytnúť iba v rámci $T$ hodnotení ($T$ je vo všeobecnosti oveľa menšie ako celkový počet dátových

záznamov), keďže pri každom $i$-tom prechode prechádzame indexmi $T$ pre dátové záznamy zaradené na $i$-tom mieste vo vektore detekcie odľahlých hodnôt.

Druhý variant funkcie $COMBINE$, označený ako prístup kumulatívneho súčtu. Táto kombinovaná metóda najprv vytvorí konečný vektor skóre odľahlých hodnôt $AS_{FINAL}$ sčítaním všetkých vektorov odľahlých hodnôt $AS_t$ zo všetkých $T$ iterácií, potom zoradí vektor $AS_{FINAL}$ a nakoniec identifikuje záznamy údajov s najvyšším skóre ako odľahlé hodnoty. Napríklad dátový záznam $NC_1$ na obrázku 2.27 môže byť hodnotený ako prvá odľahlá hodnota podľa Algoritmu 1, hodnotená ako štvrtá podľa Algoritmu 2, ..., a hodnotená ako druhá podľa Algoritmu $t$. V prístupe kumulatívneho súčtu spočítame všetky skóre, ktoré zodpovedajú dátovému záznamu $NC1$, menovite skóre $AS_{1,1}$, $AS_{2,4}$, ..., a $AS_{t,2}$, a potom zoradíme všetky dátové záznamy $NC_i, i = 1, ..., m$ podľa nanovo vypočítaného skóre. Jednou z výhod tejto metódy je, že odľahlá hodnota, ktorá je detekovaná jediným algoritmom, môže mať veľmi veľké odľahlé skóre a po vykonaní všetkých sčítaní môže mať stále dostatočne veľké konečné odľahlé skóre na to, aby bolo možné zistiť, že ide o odľahlú hodnotu. Táto skutočnosť je mimoriadne dôležitá v scenároch, v ktorých sú odľahlé hodnoty viditeľné len v niekoľkých dimenziách, pretože v takom prípade stačí vybrať relevantné vlastnosti len v malom počte opakovaní, vypočítať vysoké skóre odľahlých hodnôt pre tieto podmnožiny prvkov a tým spôsobiť, že tieto odľahlé hodnoty sú v konečnom skóre vysoko hodnotené.

## 2.8 Metriky

Dôležitou súčasťou detekcie odľahlých hodnôt je vyhodnotenie správnosti detekcie. To môže byť robené rôznymi spôsobmi. Avšak aby bolo možné jednotlivé metriky použiť tak je nutné, aby bol dataset anotovaný, to znamená, aby mal hodnoty, či ide alebo nejde o odľahlú hodnotu. V tejto sekcii si predstavíme niektoré základne metriky na zistenie účinnosti detekcie.

### 2.8.1 Matica zámen

Matica zámen (confusion matrix) je populárna metóda pri riešení klasifikačných problémov. Ako sa píše v [5], tak môže byť použitá ako na binárnu klasifikáciu, tak aj na viac-triednu klasifikáciu. Pri probléme detekcie odľahlých hodnôt sa používa ako pre binárny klasifikátor. Príklad matice zámen pre binárnu klasifikáciu môžeme vidieť v tabuľke 2.1.

Matica zámen pozostáva zo štyroch základných charakteristík (hodnôt), ktoré sú použité na definovanie metrík klasifikátoru. Tieto štyri hodnoty sú:

- TP (True Positive): TP reprezentuje počet pozitívnych príkladov klasifikovaných správne
- TN (True Negative): TN reprezentuje počet negatívnych príkladov klasifikovaných správne
- FP (False Positive): FP reprezentuje počet negatívnych príkladov klasifikovaných ako pozitívne
- FN (False Negative): FN reprezentuje počet pozitívnych príkladov klasifikovaných ako negatívne

Následne je možne z matice zámen odvodiť štyri metriky pre hodnotenie výkonu algoritmu. Ide o precision, accuracy, recall a f1 skóre, popis týchto metód pochádza z [29].

| | | Predpovedané | |
|---|---|---|---|
| | | Pozitívne | Negatívne |
| Skutočné | Pozitívne | $TP$ | $FN$ |
| | Negatívne | $FP$ | $TN$ |

Tabulka 2.1: Príklad matice zámen pre dve triedy.

Accuracy je reprezentovaná ako pomer dobre klasifikovaných bodov (TP+TN) ku celkovému množstvu bodov v dátovej sade.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \tag{2.33}$$

Precision metódy reprezentuje pomer správne klasifikovaných bodov z pozitívnej triedy (TP) ku všetkých bodom, ktoré boli klasifikované do pozitívnej triedy (TP+FP).

$$Precision = \frac{(TP)}{(TP + FP)} \tag{2.34}$$

Recall je metrika, ktorá definuje pomer správne klasifikovaných pozitívnych bodov (TP) ku všetkým bodov ktoré by mali byť pozitívne(TP+FN). Recall sa tiež niekedy nazýva aj sensitivity.

$$Recall = \frac{(TP)}{(TP + FN)} \tag{2.35}$$

Poslednou metrikou, ktorú môžeme odvodiť z matice zámen je f1 skóre. To vyjadruje rovnováhu medzi precision a recall.

$$F1score = \frac{(2 \times Precision \times Recall)}{(Precision + Recall)} \tag{2.36}$$

### 2.8.2 Precision@n

Detekcie odľahlých hodnôt je veľmi špecifický prípad klasifikácie, keďže jedna trieda sa vo vzorke dát nachádza veľakrát a druha podstatne menej. Práve tu menšiu triedu je nutné rozpoznať správne a už je len na užívateľovi, či chce aby sa rozpoznalo viacej dát ale môžu byť medzi nimi nepresné odhady alebo menej a tie musí trafiť všetky. Ako sa píše v [9], tak vo všeobecnosti používateľov metód detekcie odľahlých hodnôt zaujíma mala podmnožina pozostávajúca z objektov s najvyšším hodnotením skóre odľahlosti. Pre tie prípady, kde je vopred stanovený počet odľahlých kandidátov n, tak najjednoduchším meradlom je presnosť pri n (označujeme $P@n$), definovaná, [9], ako podiel správnych výsledkov v top n dátach podľa skóre odľahlosti. Pre databázu $DB$ o veľkosti $N$, pozostávajúcu z odľahlých hodnôt $O \subset DB$ a normálnych hodnôt $I \subseteq DB(DB = O \cup I)$ môže byť $P@n$ formálne zapísané ako:

$$P@n = \frac{|\{o \in O | rank(o) \leq n\}|}{n}. \tag{2.37}$$

Pri použití $P@n$ na posúdenie všeobecnej výkonnosti metódy hodnotenia odľahlých hodnôt nie je jasné, ako vhodne zvoliť parameter $n$. Nastavením ho na počet odľahlých hodnôt v ground truth, $n = |O|$, sa získa hodnotenie precision. Vždy, keď počet odľahlých hodnôt $n = |O|$ je veľmi nízky v porovnaní s $N$, tak hodnota $P@n$ môže byť príliš nízka a ako taká nie veľmi informatívna. Na druhej strane, keď $n = |O|$ je relatívne veľké (rovnakého rádu

ako $N$), možno získať veľmi vysoké hodnoty $P@n$, jednoducho vďaka relatívne malému počtu dostupných normálnych hodnôt, čo opäť nie je veľmi informatívne. Na uľahčenie porovnávania výsledkov metód s rôznym počtom odľahlých hodnôt je doporučené, aby sa $n$ nastavilo na hodnotu podla rovnice 2.38, ktorá je bližšie popísaná v [14].

$$n = \frac{Index - ExpectedIndex}{MaximumIndex - ExpectedIndex} \tag{2.38}$$

Avšak táto hodnota môže dobre určiť, či horné percento odľahlých hodnôt, ktoré detekoval algoritmus ako najodľahlejšie, sú skutočne odľahlé, keďže práve tie najodľahlejšie hodnoty je zvyčajne najdôležitejšie určiť správne.

# Kapitola 3

# Dátové sady

V tejto sekcii sa nachádza popis dátových sad, ktoré boli použité v rámci testovania metód na detekciu odľahlých hodnôt. Datasety pochádzajú z UCI repozitára dátových sad pre strojové učenie [23], z ktorej boli následne prevzaté a spracované v Outlier detection datasets (ODDS) [26] a v článku [9], ktorý sa zaoberal porovnávaním metód detekcie odľahlých hodnôt bez učiteľa.

O každom datasete je uvedený krátky popis, ako vznikol, počet a typ atribútov, počet inštancii. V ODDS je doplnená hodnota, či ide o odľahlú hodnotu alebo nie. V článku zase dátové sady podrobili preproccesingu, kde z každého datasetu získaného z UCI vzišli nové datasety v 4 hlavných kategóriach. Každý dataset nemusí obsahovať všetky kategórie. V každom datasete pomocou kNN, ktoré je popísane v 2.6.3, získali odľahlé hodnoty.

Ďalej budú popísané zdroje datasetov a jednotlivé datasety, ktoré môžu byť využité pri testovaní a experimentovaní s rôznymi metódami na detekciu odľahlých hodnôt. Budú obsahovať krátky popis, zdroj odkiaľ pochádzajú a parametre. Všetky informácie, ktoré budú uvedené v sekciach 3.1 a 3.2 pochádzajú zo zdrojov [26], resp. [9].

## 3.1 Datasety z ODDS

ODDS poskytuje prístup k širokej kolekcii datasetov na detekciu odľahlých hodnôt. Datasety sú rozdelené do skupín a každý dataset obsahuje informáciu o počte a spôsobe určenia odľahlých hodnôt. Jednotlivé odľahlé hodnoty sú označené číslicou 1 a tie ktoré nie sú odľahlé číslicou 0.

- Lymphography dataset - originálny dataset pochádza z inštitútu onkológie v Ľubľane a bol určený na klasifikáciu. Ide o dataset, ktorý obsahuje štyri triedy, ale dve z nich sú pomerne malé (obsahujú 2, resp. 4 záznamy). Preto sú tieto dve triedy spojené a považované za odľahlé hodnoty v porovnaní s ďalšími dvoma väčšími triedami (obsahujú 80, resp. 61 záznamov). Celkový počet záznamov v tomto datasete je 148, z toho je za odľahlé považovaných 6 záznamov, čo predstavuje 4.1%. Každý záznam ma 18 dimenzii a všetky hodnoty sú zapísané numerickými hodnotami.

- Arrhythmia dataset - originálny dataset pochádza z UCI repozitára. Pôvodne určený na klasifikáciu, kde cieľom bolo rozlišovať medzi prítomnosťou a absenciou srdcovej arytmie a následné zaradenie do jednej zo 16 skupín. Trieda 1 sa vzťahovala na normálne prípady, triedy 2 až 15 sa týkali rôznych druhov arytmie a trieda 16 obsahovala ostatné, nezaradené dáta. Pôvodne obsahoval 279 atribútov,z ktorých bolo 5 katego-

rických a tie boli odstránené. Za odľahlé hodnoty sú považované najmenšie triedy a to konkrétne triedy 3, 4, 5, 7, 8, 9, ,14 , 15. Zvyšok tried je označený ako normálne dáta. Celkovo obsahuje 452 záznamov z ktorých 14,6% je odľahlých hodnôt.

- Ionosphere dataset - originálny dataset pochádza z UCI repozitára. Pôvodný dataset slúžil na binárnu klasifikáciu a uchováva údaje pozbierané radarom. „Dobré“ radarové hodnoty sú tie, ktoré vykazujú dôkaz o nejakom type štruktúry v ionosfére a „zlé“ sú tie všetky ostatné. Za normálne hodnoty sú považované „dobré“ hodnoty a „zlé“ sú považované za odľahlé. Pôvodný dataset obsahuje 34 atribútov, avšak jeden atribút bol odstránený, keďže obsahoval iba nulové hodnoty a nemal by vplyv na detekciu. Celkovo obsahuje 351 záznamov z 35% podielom odĺahlých hondôt.

- Mnist dataset - originálny dataset obsahuje ručne písane čísla a pozostáva zo 60 000 trénovacích príkladov a 10 000 testovacích. Čísla boli normalizované veľkosťou a vycentrované do stredu. Ako normálne hodnoty sa zoberie trieda čísla 0 a ako odľahlá hodnota bude trieda čísla 6, z ktorej sa zoberie podstatne menšia časť. Z celkového množstva 784 atribútov sa vybralo náhodne 100 z nich. Dataset obsahuje celkovo 7603 záznamov z toho 9,2% sú odľahlé hodnoty.

- Satimage-2 dataset - dataset vychádza z datasetu Statlog (Landsat Satellite) dataset z UCI repozitára. Dataset obsahuje spektrálne hodnoty pixelov zo satelitných snímiek. Pôvodne obsahuje 7 tried, avšak pre detekciu odľahlých hodnôt bola trieda 2 zmenšená a označená ako odľahlá, zatiaľ čo ostatné triedy boli spojené a označené ako normálne hodnoty. Použitý dataset tak obsahuje 5803 záznamov, 36 atribútov, ktoré sú v rozmedzí 0-255 a obsahuje 1,22% odľahlých hodnôt.

- Vowels dataset - pôvodný dataset pochádza z UCI. Rôzny rečníci vyslovovali japonsku samohlásku a následne bol zvuk spracovaný. Pre detekciu odľahlých hodnôt budeme každý rámec považovať za jeden bod, zatiaľ čo v pôvodnom datasete sa ako bod považuje blok rámcov. Trieda (rečník) 1 je zmenšená na 50 prvkov a považuje sa za odľahlú. Normálne hodnoty sú považované triedy 6,7 a 8. Zvyšné triedy sa neberú do úvahy. Dataset obsahuje 1456 záznamov, 12 atribútov pre každý záznam a 3,4% odľahlých hodnôt.

- Shuttle dataset - data pochádzajú z univerzity v Austrálii, konkrétne z fakulty informatiky a dataset bol pôvodne určený na klasifikáciu do viacerých tried. Pri detekcii odľahlých hodnôt je trénovací a testovací dataset spojený a keďže 80% dát patrí do triedy 1, tak prvky z tried 2, 3, 5, 6, 7 sú považované za odľahlé. Prvky z triedy 4 boli vyradené. Dataset obsahuje 49097 záznamov, má 9 numerických atribútov a 3511 odľahlých hodnôt, čo je 7% z celkového množstva záznamov.

## 3.2 Datasety z článku

Tento repozitár obsahuje datasety získane z UCI repozitára. Na jednotlivé datasety bola aplikovaná kNN metóda na detekciu odľahlých hodnôt a počet odľahlých hodnôt v jednotlivých datasetoch bol určený podľa skóre odľahlosti, kedy sa zobralo percento najodľahlejších hodnôt. Každý dataset sa v repozitáry nachádza v rôznych variantách podľa aplikovaného predspracovania a tie varianty sú: normalizované dáta bez duplikátov, normalizované dáta s duplikátmi, nenormalizované dáta bez duplikátov, nenormalizované dáta s duplikátmi. Každý dataset nemusí obsahovať všetky varianty.

Pre každý dataset je k dispozícii porovnanie 12 základných metód na detekciu odľahlých hodnôt z rôznymi variantami parametrov a zobrazené v prehľadných tabuľkách a grafoch. Ďalej boli datasety rozdelené na trénovacie a testovacie dáta. Ak dataset obsahuje dostatok dat, tak to bolo prevedené viackrát a vznikli rôzne kombinácie trénovacích a testovacích dat.

- Cardiotocography Data Set - ide o dataset zaoberajúci sa srdcovými chorobami pôvodne učený na klasifikáciu. Pôvodný dataset obsahuje 2126 záznamov, avšak v upravenom datasete sa ich nachádza 1688. Dokopy je k dispozícii 41 verzii datasetu a každých 10 z nich obsahuje postupne 2%, 5%, 10%, 20% odľahlých hodnôt a posledný ich obsahuje 22%. Datasety obsahujú 23 reálnych atribútov a záznamy sú klasifikované do 3 tried: normálne, podozrivé alebo patologické. Normálni pacienti sú považovaný normálnu hodnotu, zvyšné dve triedy sú považované za odľahlé hodnoty.

- Waveform dataset - ide o generované dáta pôvodne určené na klasifikáciu. Obsahuje 3 triedy. Každá trieda vznikla kombináciou 2 z 3 základných vĺn. Oproti pôvodnému datasetu, ktorý obsahoval 5000 záznamov, tak v upravenom je jedna trieda podvzorkovaná na hodnotu 100 a tá je považovaná za odľahlú triedu. Teda upravený dataset obsahuje 3443 záznamov a každý záznam má 21 numerických atribútov. K dispozícii je 20 datasetov, ktoré vznikli z pôvodného datasetu z UCI, 10 z nich je normovaných a 10 je nenormovaných.

- Internet Advertisements Data Set - dataset reprezentuje súbor možných reklám na internetových stránkach. Funkcia zakóduje geometriu obrázka a tiež frázy vyskytujúce sa v URL, URL obrázka, alternatívny text obrázka a ďalšie informácie súvisiace s daným obrázkom. Pôvodnou úlohou bolo predpovedať, či ide o reklamu alebo o obrázok, ktorý nie je reklamou. V upravenom datasete sú reklamy považované za odľahlé hodnoty. Dataset má 3264 záznamov, z nich 454 je odľahlých a každý záznam ma 1555 atribútov. K dispozícii je 30 datasetov, ktoré sú rozdelené na 10 podla počtu odľahlých hodnôt, prvých 10 ma 2% odľahlých hodnôt, ďalších 10 ich ma 5% a posledných 10 majú po 10% odľahlých hodnôt.

- Stamps dataset - tento dataset nie je z UCI repozitáru ale bol použitý v [22]. Dáta reprezentujú tlačené alebo oskenované pečiatky a pravé, atramentové pečiatky. Vlastnosti sú založené na farbách a tlačových vlastnostiach pečiatok. Falošné, teda skenované alebo tlačené, sú považované za odľahlé hodnoty. Dataset obsahuje 340 záznamov, z nich 31 je považovaných za odľahlé. Každý záznam pozostáva z 9 atribútov. K dispozícii je 20 variant datasetov z 2%, resp. 10% odľahlých hodnôt.

- Spambase dataset - ide o klasifikáciu Emailov na spam alebo tzv. nonspam. Koncept spamu je rôznorodý: reklamy na produkty/webové stránky, schémy rýchleho zarábania peňazí, reťazové listy.... Zbierka emailov pochádza zo spoločnosti HP a ich vedúceho spracovania pošty, prípadne od jednotlivcov, ktorí poslali spam. Dataset obsahuje 4601 záznamov, z toho 1813 je považovaných za spam, čo je považované za odľahlú hodnotu. Každý záznam má 57 atribútov. K dispozícii sú rôzne varianty datasetov a s rôznym množstvo odľahlých hodnôt.

## 3.3 Datasety z priemyselnej siete

Ide o 6 dátových súborov, ktoré sa líšia dĺžkou (255, 576 alebo 1152 normálnych hodnôt) a rôznym počtom odľahlých hodnôt (1% alebo 5%). Dva súbory, ktoré majú označenie „-fm-“, tak obsahujú jeden zhluk normálnych hodnôt. Ostatné štyri súbory obsahujú po 2 zhluky normálnych hodnôt.

Každý objekt predstavuje päť minútové okno chodu priemyslovej siete (IEC104) a obsahuje 3 atribúty. Prvý atribút popisuje počet prenesených paketov v danom časovom okne, druhý a tretí atribút označuje tiež počet prenesených paketov, ale tentokrát sú to pakety prenesené do/od určitého času od predchádzajúceho paketu.

# Kapitola 4

# Návrh aplikácie

Táto kapitola popisuje návrh aplikácie na detekciu odľahlých hodnôt. To znamená, že popisuje použité technológie a prípady použitia aplikácie. V sekcii 4.1 je napísaná neformálna špecifikácie aplikácie, v sekcii 4.2 je rozpis použitých technológii a v sekcii 4.3 sú rozpísané niektoré prípady použitia aplikácie.

## 4.1 Neformálna špecifikácia

Cieľom aplikácie je prehľadne zobrazovať výsledky rôznych metód na detekciu odľahlých hodnôt a zároveň ponúknuť pre užívateľa kombináciu metód, ktorá by mala vylepšiť výkonnosť detekcie. Vstupom aplikácie je dátová sada pozostávajúca z numerických atribútov. Užívateľ ju nahraje do aplikácie v odpovedajúcom formáte a to bud bez pravdivostných hodnôt (Ground Truth) alebo s nimi. Užívateľ sa tiež rozhodne, či chce dáta normalizovať, prípadne na koľko desatinných miest chce ich mat zaokrúhlené. Po nahraní dátovej sady sa zobrazí tabuľka, kde bude vidno hodnoty pre jednotlive atribúty, prípadne hodnoty, či ide o odľahlé hodnoty. Následne užívateľ môže aplikovať jednu z vybraných metód na dátovú sadu. Môže sa rozhodnúť, či chce dataset rozdeliť na trénovaciu časť a testovaciu. Táto možnosť je však možná iba ak zadal údaje s ground truth. Jednotlivé metódy môžu byť aplikované v jednoduchom stave, bez použitia špeciálnych parametrov, alebo v pokročilom stave, kde už užívateľ môže vybrať konkrétne parametre. Takisto môže užívateľ použiť kombináciu metód podľa priemeru. Ak sa tak rozhodne, tak mu vybehnú možnosti, kde si zvolí počet metód, prípadne určí aké váhy majú mať výsledky jednotlivých detektorov. Následne si už len klasicky vyberie metódy, ktoré chce pridať do kombinácie a po pridaní poslednej mu aplikácie dovolí aplikovať metódu. Po použití metódy sa v tabuľke vedľa dátovej sady zobrazí stĺpec reprezentujúci výstup metódy na detekciu odľahlých hodnôt. Jednotlivé riadky sa zafarbia podľa toho, či ide o odľahlú hodnotu alebo nie. Ak dátová sada obsahovala ground truth, tak sú zvýraznené riadky kde sa táto hodnota líši od výstupu metódy.

Po dokončení detekcie, užívateľ môže s týmito dátami pracovať. Môže si ich zoradiť podľa jednotlivých atribútov. Môže zobraziť, ktoré atribúty sú ako dôležité pre detekciu danou metódou. Taktiež si môže vybrať 2 alebo 3 atribúty a následne si zobraziť body v podpriestore daných atribútov. V týchto grafoch si užívateľ môže vybrať a zvýrazniť body, ktoré ho zaujímajú. Súčasťou je aj zobrazenie reportu danej detekcie, kde sú zobrazené hodnoty ako presnosť alebo matica zámen.

## 4.2 Použité technológie

Hlavné jadro aplikácie sa bude vytvárať v jazyku Python. V ňom sa bude hlavne pracovať s knižnicami ako je NumPy, SciPy, Pandas, Sklearn, Dalex a hlavne s PyOD, ktorá ponúka dostatočné množstvo metód na detekciu odľahlých hodnôt. Vzhľadom k tomu, že ma ísť o grafickú aplikáciu, tak pre tvorbu frontendu bolo použité QT verzie 5 pre python.

### 4.2.1 NumPy

Numpy[1] je knižnica pre Python, ktorá je používaná pre prácu z poľami. Tiež ma funkcie na prácu v oblasti lineárnej algebry, Fourierovej tranformácie a matíc. V porovnaní zo zoznamami v Pythone, tak pracuje oveľa rýchlejšie a poskytuje veľa podporných metód na prácu z poľami. Ide o jednu z najzákladnejších knižníc, ktorú využívajú aj mnohé iné knižnice.

### 4.2.2 SciPy

SciPy[2] je knižnica vedeckých výpočtov, ktorá používa NumPy. Poskytuje funkcie pre optimalizáciu, štatistiku a spracovanie signálov. V tejto práci bol použitý balíček `io`, ktorý ponúka funkcie na prácu zo súbormi. Konkrétne bola použitá metóda loadmat na načítanie vstupných súborov.

### 4.2.3 Pandas

Pandas[3] je balík pre programovací jazyk Python, ktorý sa najčastejšie používa pre prácu s dátami a úlohy spojené so strojovým učením. Tiež vychádza s balíka NumPy. Hlavným využitím knižnice v tejto úlohe je použitie dátového typu Dataframe, v ktorom sú uchovávané dáta po nahraní do aplikácie a následne spracovávané. Ide o dvojrozmernú dátovú štruktúru, ktorá môže obsahovať stĺpce rôznych typov, hlavičky pre jednotlivé osy môžu byť pomenované alebo je možné robiť aritmetické operácie cez jednotlivé osy.

### 4.2.4 Sklearn

Knižnica sklearn[4] sa zameriava na strojové učenie. Obsahuje množstvo metód pre klasifikáciu, regresiu a zhlukovanie a je navhrnutá na prácu s ostatnými knižnicami Pythonu ako NumPy alebo SciPy. Obsahuje tiež metódy na predspracovanie údajov alebo metriky na výpočet presnosti, prípadne matice zámen. Taktiež niektoré metódy klasifikácie tvoria základ odpovedajúcich metód v knižnici PyOD.

### 4.2.5 PyQtGraph

PyQtGraph[5] je grafická a GUI knižnica pre Python postavená na PyQt a NumPy. Je určená na použitie v matematických/vedeckých/inžinierskych aplikáciach. Napriek tomu, že je celá napísaná v Pythone, tak je veľmi rýchla, vďaka svojmu dobrému využitie `NumPy` na prácu s číslami a frameworku Qt GraphicsView na rýchle zobrazenie. V aplikácii je použitá na zobrazenie hodnôt v 2D a 3D grafe, ich farebné oddelenie a následný pohyb po grafe.

[1] https://numpy.org/
[2] https://scipy.org/
[3] https://pandas.pydata.org/
[4] https://scikit-learn.org/stable/
[5] https://www.pyqtgraph.org/

### 4.2.6 Dalex

Balíček dalex[7] prechádza akýkoľvek model vytvorený metódami strojového učenia, v mojom prípade prechádza model vytvorený knižnicou PyOD, a pomáha preskúmať a vysvetliť jeho správanie. Medzi konkrétne modely, ktoré dokáže spracovať ešte patria sklearn, keras, pandas alebo tensorflow. Hlavný objekt `Explainer` vytvára obal okolo prediktívneho modelu. Zabalené modely je potom možné preskúmať a porovnať so zbierkou vysvetlení na úrovni modelu a na úrovni predpovedí. V tejto aplikácii bol použitý na zistenie dôležitosti jednotlivých atribútov po použití metódy detekcie odľahlých hodnôt.

### 4.2.7 PyQt

PyQt[6] spája medziplatformový framework Qt C++ s jazykom Python, ide o GUI modul. Qt je viac než len súprava nástrojov GUI, a preto obsahuje abstrakcie sieťových soketov alebo vlákien spolu s Unicode, SQL, databázami, SVG, OpenGL, XML, operačným prehliadačom, servisným systémom a obrovským množstvom miniaplikácií GUI. Princíp, na ktorom trieda Qt funguje, súvisí s mechanizmom slotov zodpovedným za ponúkanie komunikácie medzi položkami s cieľom jednoducho navrhnúť opätovne použiteľné softvérové komponenty. Qt tiež prichádza s Qt Designerom, nástrojom, ktorý funguje ako grafické používateľské rozhranie. PyQt dokáže navrhnúť kód Python z Qt Designer a zároveň pridať nové ovládacie prvky GUI.

Tento framework bol použitý na tvorbu graficke užívateľského rozhrania pre vytváranú aplikáciu. Hlavné okno aplikácie je tvorené pomocou `QMainWindow`triedy do ktorej sú pridávané ďalšie widgety, prvky frameworku, ako sú `QPushButton`, `QToolBar`, `QTextField` alebo `QCombobox`. Tieto widgety môžu byť rozmiestnené v hlavnom okne priamo v triede `QMainWindow` alebo pomocou Qt Designeru sa vytvorí statické UI a to sa potom importuje do danej triedy.

### 4.2.8 PyOD

Ide o komplexnú a škálovateľnú súpravu nástrojov jazyka Python na detekciu odľahlých objektov v dátach s viacerými atribútmi. Obsahuje viacej ako 30 rôznych detekčných algoritmov. Používa sa v rôznych akademických výskumoch a komerčných produktoch. Je tiež uznávaný komunitou strojového učenia s rôznymi špecializovanými príspevkami.

PyOD ponúka jednotné API, podrobnú dokumentáciu a interaktívne príklady rôznych algoritmov, pokročilé modely klasických modelov zo sklearn, ale aj novších modelov a algoritmov. Taktiež ponúka optimalizovaný výkon a paralelizáciu a je kompatibilný s Pythonom 2 a 3.

#### Implementované algoritmy

Jednotlivé algoritmy, ktoré ponúka knižnica PyOD, sú rozdelené podľa typu na pravdepodobnostné modely (2.4), lineárne modely (2.5), modely založené na blízkosti (2.6) a kombinované metódy (2.7). Z týchto jednotlivých typov boli následne vybrané a použité v aplikácii konkrétne metódy:

1. Pravdepodobnostné modely
    - COPOD (Copula-Based Outlier Detection)

[6] https://doc.qt.io/qtforpython/

- ABOD (Angle-Based Outlier Detection)
- SOS (Stochastic Outlier Selection)

2. Lineárne modely
   - PCA (Principal Component Analysis)
   - OCSVM (One-Class Support Vector Machines)
   - MCD (Minimum Covariance Determinant)
3. Modely založené na blízkosti
   - LOF (Local Outlier Factor)
   - CBLOF (Clustering-Based Local Outlier Factor)
   - kNN (k Nearest Neighbors)
4. Kombinované modely
   - IForest (Isolation Forest)
   - FB (Feature Bagging)

## 4.3 Prípady použitia aplikácie

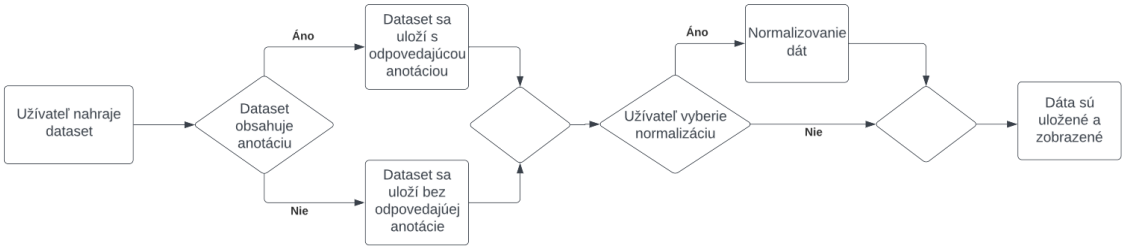
Jednou z hlavných časti aplikácie je interakcia s užívateľom. Aplikácia by mala byť jednoduchá a prehľadná, zároveň by však užívateľovi mala zobraziť všetky relevantné informácie k detekcii odľahlých hodnôt. Na obrázku 4.3 môžeme vidieť graf návrhu prípadov použitia aplikácie.

## 4.4 Spracovanie vstupných dát

V tejto sekcii bude popísaný návrh spracovania vstupného datasetu po nahraní užívateľom do aplikácie.

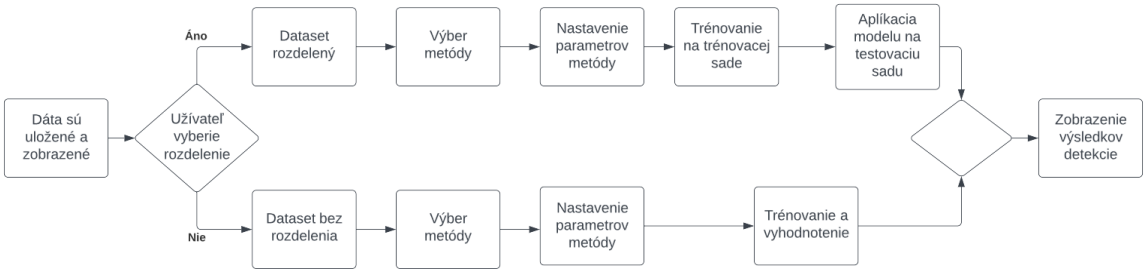
Predtým ako je možné používať aplikáciu na detekciu odľahlých hodnôt, tak je nutné aby užívateľ nahral dataset, na ktorom chce vybranú metódu použiť. Zadaný dataset môže byť vo formáte .csv, .xlsx alebo formáte .mat. Užívateľ si ho môže vybrať z prieskumníka súborov, ktorý je otvorený pomocou widgetu QFileDialog z pyQt knižnice. Ako môžeme vidieť v schéme na obrázku 4.1, tak pri nahrávaní datasetu sa musí užívateľ rozhodnúť, či chce nahrať dataset obsahujúci hodnoty ground truth, teda, či je anotovaný, alebo bude bez nich. Ďalšou možnosťou, ktorú bude mať užívateľ pri nahrávaní datasetu, je výber, či chce vstupný dataset normalizovať a ak sa tak rozhodne, tak dáta budú prevedené do rozsahu hodnôt 0 až 1. Na normalizovanie je použitá metóda standardizer z balíčka utility knižnice PyOD. Následne sa užívateľovi zobrazia hodnoty z datasetu v prehľadnej tabuľke. Dáta, ktoré užívateľ takto nahraje sa uložia do premennej typu Dataframe, ktorú poskytuje knižnica Pandas.

Po nahraní datasetu môže užívateľ prejsť na detekciu odľahlých hodnôt. Ako môžeme vidieť na blokovej schéme, obrázok 4.2, tak ak je dátová sada uložene a zobrazená, tak užívateľ má možnosť si vybrať, či chce dataset rozdeliť na trénovaciu a testovaciu časť alebo, či chce aplikovať trénovanie na pôvodnom datasete. Ak sa rozhodne aplikovať rozdelenie, tak pomocou widgetu QSlider z PyQt knižnice, bude môcť rozdeliť dataset na dve časti

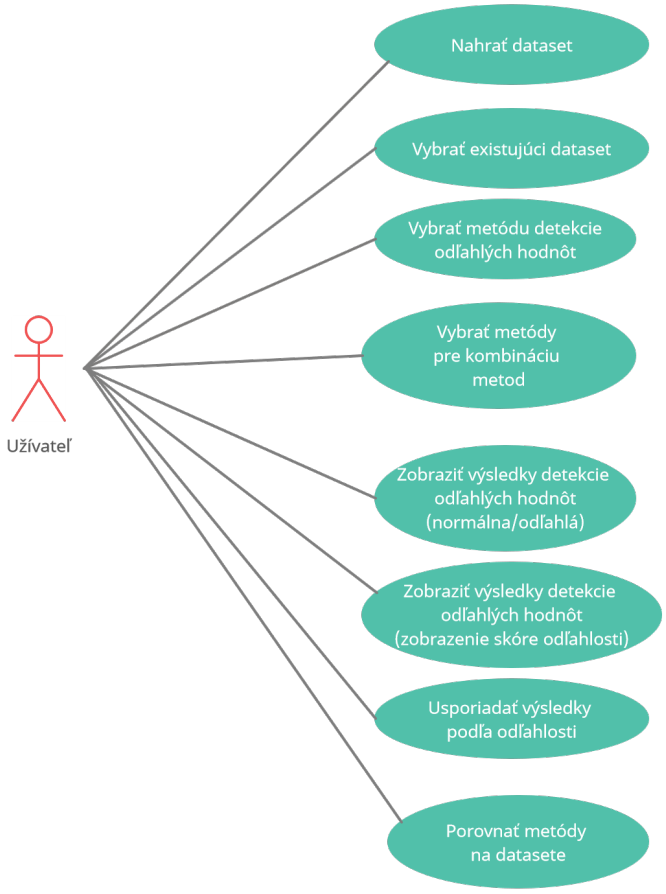



Obrázek 4.1: Návrh nahrávania datasetu.

a to v pomere, ktorom si určí na danom slideri. Ak sa rozhodne, že dataset nerozdeľuje, tak slider nie je pre neho viditeľný. Následne si užívateľ vyberie z QComboboxu jednu z dostupných metód, prípadne kombináciu dostupných metód. V predvolenom stave sú metódy k dispozícii s predvolenými parametrami. Ak sa užívateľ rozhodne, že ich chce zmeniť a prispôsobiť si danú metódu, tak zaklikne QCheckbox a zobrazia sa mu pokročilé nastavenia jednotlivých parametrov metódy s popisom, čo znamenajú, prípadne, aké hodnoty môžu obsahovať. Formulár pokročilých nastavení metód je tvorený widgetmi z PyQt ako napríklad QCombobox, QSlider, QPushButton alebo QTextField. Informácie o jednotlivých parametroch sa zobrazia pomocou QTooltipu. Následne je aplikovaná vybraná metóda na trenovaciu sadu. Ak bol dataset rozdelený tak model sa použije aj na testovaciu sadu, podľa toho, čo sa naučil. Následne sú výsledky zobrazené, kde odľahlých hodnotám bolo priradené číslo 1 a normálnym číslo 0. Dané hodnoty boli doplnené do zobrazenej tabuľky do posledného stĺpca a jednotlivé riadky boli podľa tejto hodnoty zafarbené na zeleno, resp. červeno. Ak dátová sada obsahoval hodnoty ground truth, tak jednotlivé riadky obsahuju svetlejšiu farbu, ak sa výstup z modelu nerovná hodnote ground truth .




Obrázek 4.2: Návrh aplikovania detekcie odľahlých hodnôt na nahraný dateset.




Obrázek 4.3: Diagram prípadu použitia aplikácie.

# Kapitola 5

# Implementácia

V tejto kapitole sú popísané detaily implementácie pre najdôležitejšie časti aplikácie. Bude obsahovať podrobnejší popis využitia knižníc popísaných v 4.2.

## 5.1 Grafické užívateľské rozhranie

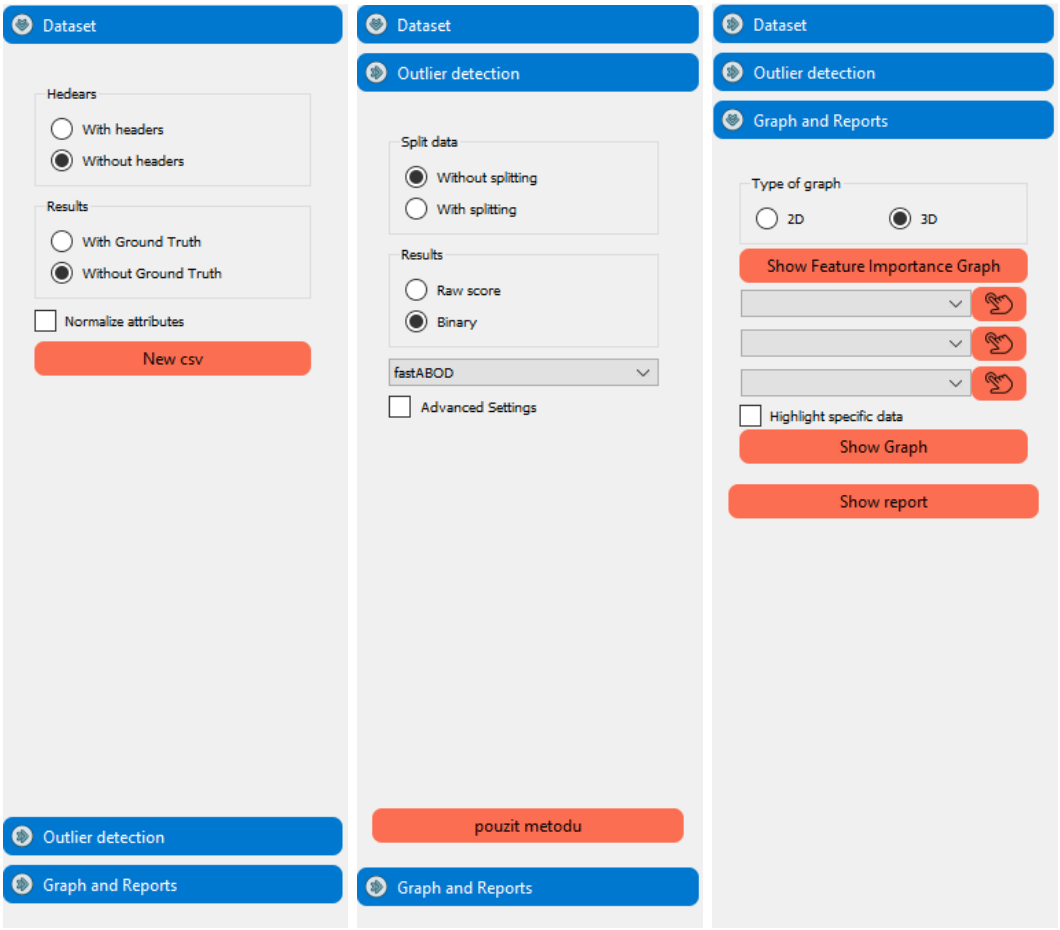
Cieľom bolo vytvoriť aplikáciu, ktorá bude užívateľský prístupná a bude prehľadne zobrazovať výsledky detekcie odľahlých hodnôt užívateľovi. Grafické rozhranie je implementované pomocou knižnice PyQT 5. PyQt vychádza z Qt, čo je multiplatformový software pre tvorbu grafického rozhrania. Za pomoci nástroja Qt Designer, čo je nástroj na navrhovanie a vytváranie grafického používateľského rozhrania, bolo vytvorených niekoľko základných obrazoviek. Čo sa týka štýlovania obrazoviek, tak na to bolo použité CSS, avšak iba vo verzii CSS2, keďže vyššiu verziu PyQt nepodporuje. V jednotlivých podsekciách budú vysvetlené hlavné časti aplikácie a následne v prílohe B je možné vidieť screenshoty celej aplikácie.

### 5.1.1 Hlavná stránka

Hlavná obrazovka sa nachádza v súbore main_ui.ui a je reprezentovaná triedou `MainWindow`. Hlavná obrazovka sa skladá z dvoch kľúčových časti. V ľavej časti je vytvorené menu pomocou komponentu `QToolBox`, ktoré pozostáva z troch stránok, obrázok 5.1. Na prvej stránke je vytvorený formulár pre nahranie dátovej sady, na druhej stránke je formulár na aplikovanie zvolenej metódy detekcie odľahlých hodnôt a tretia stránka obsahuje formulár na vytvorenie grafu a zobrazenie štatistík detekcie odľahlých hodnôt. V druhej hlavnej časti, centrálnej časti, je pomocou komponenty `QTableWidget` vytvorená tabuľka, ktorá sa po nahraní dátovej sady zaplní a po aplikovaní detekcie je v nej možné vidieť jednotlivé odľahlé hodnoty. Ak je zvolená detekcia s rozdelením datasetu na trénovaciu a testovaciu časť, tak sa zobrazia dve tabuľky, kde v ľavej je práve trénovacia sada a v pravej testovacia.

### 5.1.2 Detekcia

Po tom čo užívateľ nahraje dataset, tak sa môže presunúť na záložku detekcie odľahlých hodnôt. Tu si najprv vyberie, či chce dataset rozdeliť na dve časti alebo nie, prípadne v akom pomere ho chce rozdeliť. Takisto si vyberie, či chce zobraziť výsledky v binárnej forme alebo výsledky ako skóre odľahlosti. Následne z `QComboboxu` si vyberie metódu detekcie odľahlých hodnôt z knižnice PyOD. Keď si vyberie metódu, tak má možnosť ju použiť v predvolenej podobe, alebo zaklikne `QCheckbox` a sa mu zobrazia pokročilejšie nastavenia parametrov



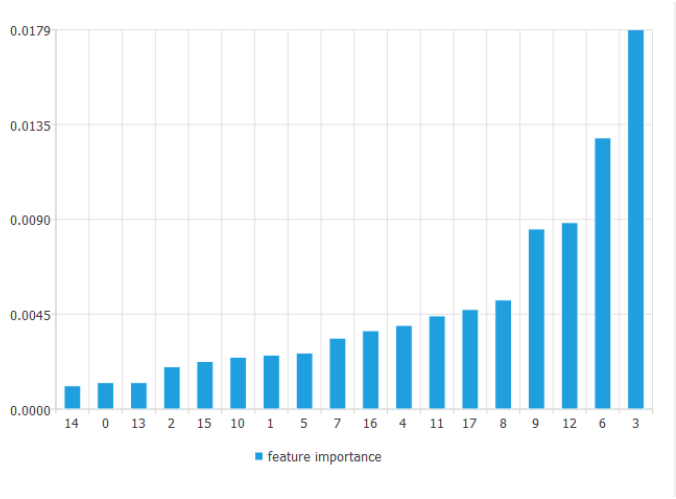

(a) Nahranie datasetu. (b) Detekcia. (c) Graf a štatistiky.

Obrázek 5.1: Zobrazenie menu stránok aplikácie na detekciu odľahlých hodnôt.

danej metódy. K dispozícii je 12 metód popísaných v kapitole 2 a potom špeciálny výber kombinácie hodnôt. Ak si užívateľ vyberie túto možnosť tak sa mu zobrazia možnosti, kde si môže vybrať typ kombinácie, počet koľko metód chce kombinovať a pokiaľ si vyberie možnosť kombinácie priemerom, tak môže jednotlivým metódam priradiť aj váhy. Takto si užívateľ môže skúšať rôzne varianty metód pre zlepšenie detekcie.

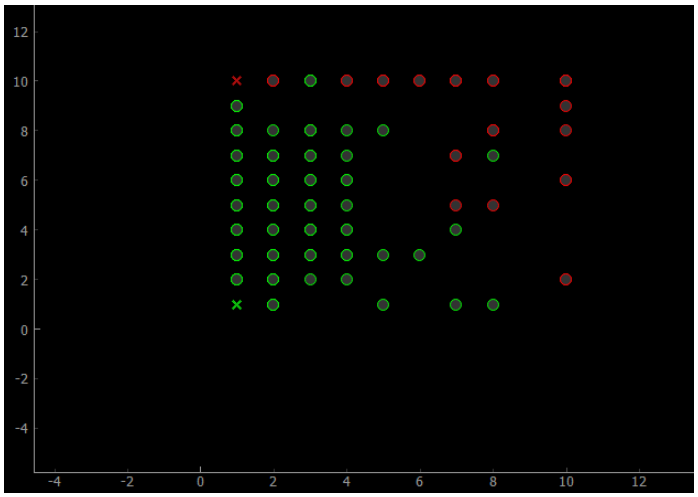
### 5.1.3 Grafy

Jednou z kľúčových časti aplikácie je zobrazenie dátovej sady v grafickom formáte. Po nahraní datasetu a použití metódy na detekciu odľahlých hodnôt, tak si užívateľ môže zobraziť 2D alebo 3D graf. Atribúty, ktoré si chce zobraziť, tak si môže ľubovoľne zvoliť, či už kliknutím na daný stĺpec v tabuľke alebo výberom z `QComboboxu`. Vzhľadom k tomu, že dátová sada môže obsahovať väčšie množstvo atribútov, z ktorých niektoré môžu výraznejšie ovplyvňovať rozhodovanie o odľahlých hodnotách ako iné, tak bolo umožnené užívateľovi zobraziť si dôležitosť jednotlivých atribútov, viz obrázok 5.2. Toto zobrazenie je naprogramované v súbore featureImportance.py a na zobrazenie histogramu sa používajú komponenty z balíčku `PyQt5.QtChart` ako `QChart`, `QBarSet`,`QBarSeries` alebo `QChartView`.



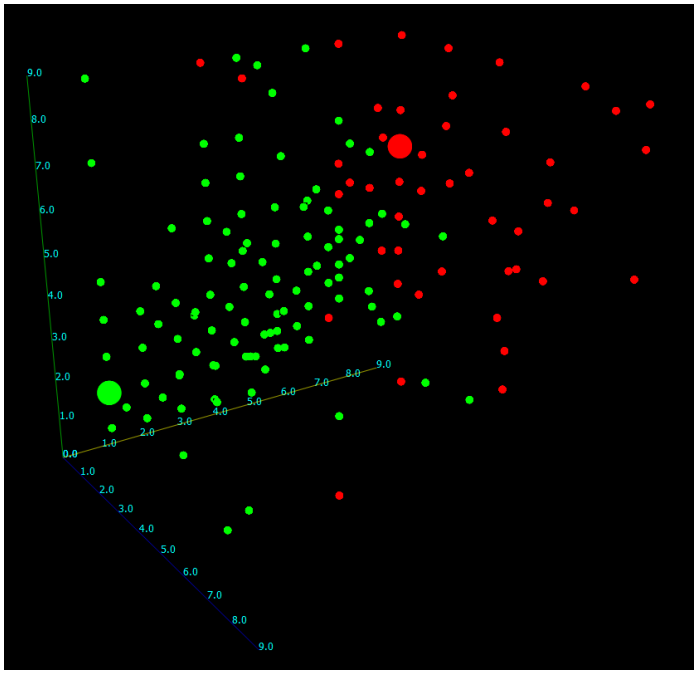

Obrázek 5.2: Príklad grafu dôležitosti atribútov pre atribúty 0-17 pre dataset lympho a metódu Isolation Forest.

Samotné zobrazenie grafu môže byť s dvoma alebo troma atribútmi a zobrazenie grafu je v súbore graphViewModel.py. V oboch grafoch sa môže užívateľ pomocou kurzoru na myši pohybovať, priblížovať a v 3D grafe môže aj meniť polohu kamery v 3D priestore. Ak ide o 2D graf, tak je použitý `PlotWidget` z `pyqtgraph.opengl`, kde sú jednotlivé údaje zobrazene buď červeným alebo zeleným kruhom, podľa toho, či ide o odľahlú hodnotu alebo nie. Ak si užívateľ zvoli údaje ktoré chce zvýrazniť, tak tie majú miesto krúžku krížik, viz. obrázok 5.3. V prípade 3D grafu je opäť použitý balíček `pyqtgraph.opengl`, avšak




Obrázek 5.3: Príklad 2D grafu zo zvýraznenými bodmi.

tentoraz sú body zobrazené pomocou `GLScatterPlotItem`. Opäť sú body oddelene farebne, avšak na rozdiel od 2D grafu, kde zvýraznenie bodov je robene pomocou krížikov, tu sú zvýraznené body zväčšené, viz. obrázok 5.4. Toto rozhodnutie vzišlo z toho, že bol použitý iný typ grafu oproti 2D zobrazeniu, kde oba grafy nepodporujú rovnakú úpravu bodov. Pre 3D graf však bolo potrebné prispôsobiť jednotlivé osy, Kód pre tieto osy sa nachádza v súbore customAxis.py




Obrázek 5.4: Príklad 3D grafu zo zvýraznenými bodmi.

### 5.1.4 Štatistiky detekcie

Užívateľ si môže jednotlivé štatistiky detekcie zobraziť iba v prípade, keď zadal dátovú sadu s odpovedajúcimi hodnotami Ground Truth. Zobrazenie výsledkov sa v aplikácii nachádza v záložke Graph and Reports a po kliknutí na tlačidlo sa užívateľovi zobrazia dve tabuľky. Tabuľka vľavo obsahuje hodnoty precision, recall, f1 score a accrucacy, tabuľka vpravo obsahuje maticu zámen a hodnotu precision@n, všetky tieto metriky boli popísane v sekcii 2.8. Pokiaľ ide o zdrojové súbory, tak zobrazenie tabuliek a reportu sa nachádza v súbore reportViewModel.py. Na získanie hodnôt boli použité metódy `classification_report`, `confusion_matrix` a `top_k_accuracy_score` z balíčku `metrics` knižnice sklearn.

# Kapitola 6

# Experimenty

V tejto časti si porovnáme výkonnosť jednotlivých metód na detekciu odľahlých hodnôt a porovnáme si ju naprieč rôznymi datasetmi popísanými v 3 a nad generovanými dátami. Na experimentovanie boli, tak ako v aplikácii, použité implementácie metód z knižnice PyOD, ktoré sú popísané v kapitole 2. Jednotlivé metódy boli použité z predvolenými parametrami, pokiaľ nebude napísané inak. Na experimenty nebola použitá priamo vyvíjaná aplikácia vzhľadom k tomu, že aplikácia sa viacej sústredi na analýzu a zobrazenie výsledkov jednotlivých metód užívateľovi a nie na masívne spúšťanie rôznych metód. Zdrojové kódy pre jednotlivé experimenty môžeme nájsť v súboroch experimentsWithOdds.py, popísané v sekcii 6.1, experimentsWithgenerated.py, popísané v sekcii 6.2 a experimentsWithRealData.py, popísane v sekcii 6.3. Výsledky jednotlivých metód môžu slúžiť užívateľovi ako východisko pre používanie jednotlivých metód v aplikácii alebo výsledky pre jednotlivé metódy mu môžu dať dobrú predstavu, ktoré metódy by mohol použiť pri kombinácii a ktoré nedodávajú dobré výsledky.

Testovanie prebiehalo na notebooku s operačným systémom Windows 10 v programe Spyder. Notebook bol vybavený procesorom Intel Core i7 9.generácie s 16GB pamäťou RAM.

Jednotlivé experimenty prebiehali v desiatich iteráciach. Každá dátová sada bola rozdelená na trénovaciu časť a testovaciu časť v pomere 60% ku 40%. Na trénovacej sade sa natrénoval model, ktorý bol následne aplikovaný na testovaciu sadu. Aby sa zaistilo, aby sa v každej iterácie nepracovalo s rovnakým datasetom, tak rozdelenie vždy prebiehalo náhodne a teda detekcia bola aplikovaná vždy na mierne iný dataset a výsledky neboli vždy rovnaké. Následne boli výsledky jednotlivých iterácii spriemerované a uložené do tabuľky.

## 6.1 Existujúce experimenty

Knižnica PyOD ponúka na svojich stránkach experimenty z jednotlivými metódami. Tie sa nachádzajú v súbore benchmark.py. V porovnaní s mojimi experimenty, však bola použitá iná varianta označenia odľahlých hodnôt, keďže u mňa vychádzam z natrénovaného modelu a následne používam natrénovanú hranicu na určenie, či ide alebo nejde o odľahlú hodnotu. V tomto existujúcom riešení natrénujú model a následne na testovacích dátach vyhodnotia odľahlé hodnoty a vrátia skóre odľahlosti. Potom sa neriadia rozhodovacou hranicou, ale spočítajú koľko odľahlých hodnôt sa má nachádzať v danom testovacom vzorku a to percento dát s najvyšším skóre odľahlosti označia ako odľahlé a následne vyhodnotia precision. Z toho

| Dáta | ABOD | COPOD | SOS | PCA | MCD | OCSVM | LOF | CBLOF | KNN | FB | IForest |
|---|---|---|---|---|---|---|---|---|---|---|---|
| arrhythmia | 0,354 | 0,452 | 0,332 | 0,386 | 0,352 | 0,404 | 0,357 | 0,374 | 0,408 | 0,415 | **0,457** |
| ionosphere | 0,837 | 0,592 | 0,641 | 0,626 | **0,869** | 0,737 | 0,73 | 0,764 | 0,853 | 0,756 | 0,658 |
| letter | 0,371 | 0,033 | 0,422 | 0,087 | 0,183 | 0,143 | **0,391** | 0,212 | 0,324 | 0,375 | 0,09 |
| lympho | 0,357 | 0,858 | 0,367 | 0,625 | 0,469 | 0,715 | 0,708 | 0,56 | 0,55 | 0,692 | **0,917** |
| mnist | 0,358 | 0,24 | 0,149 | 0,39 | 0,265 | 0,369 | 0,344 | 0,391 | **0,429** | 0,341 | 0,32 |
| satimage-2 | 0,179 | 0,739 | 0,045 | 0,842 | 0,64 | **0,929** | 0,065 | 0,92 | 0,376 | 0,079 | 0,881 |
| shuttle | 0,193 | 0,952 |  | 0,95 | 0,743 | 0,955 | 0,143 | 0,253 | 0,219 | 0,084 | **0,953** |
| vowels | 0,486 | 0,004 | 0,18 | 0,177 | 0,046 | 0,294 | 0,343 | 0,363 | **0,514** | 0,339 | 0,17 |

Tabulka 6.1: Namerané hodnoty precision mojich experimentov.

| Dáta | ABOD | COPOD | SOS | PCA | MCD | OCSVM | LOF | CBLOF | KNN | FB | IForest |
|---|---|---|---|---|---|---|---|---|---|---|---|
| arrhythmia | 0,3808 |  |  | 0,4613 | 0,3995 | 0,4614 | 0,4334 | 0,4539 | 0,4464 | 0,4230 | **0,4961** |
| ionosphere | 0,8442 |  |  | 0,5729 | **0,8806** | 0,7000 | 0,706 | 0,6088 | 0,8602 | 0,7056 | 0,6369 |
| letter | 0,3801 |  |  | 0,0875 | 0,1933 | 0,1510 | 0,3641 | 0,0749 | 0,3312 | **0,3642** | 0,1003 |
| lympho | 0,4483 |  |  | 0,7517 | 0,5183 | 0,7517 | 0,7517 | 0,7517 | 0,7517 | 0,7517 | **0,9267** |
| mnist | 0,3555 |  |  | 0,3846 | 0,3462 | 0,3962 | 0,3343 | 0,3348 | **0,4204** | 0,3299 | 0,3135 |
| satimage-2 | 0,2130 |  |  | 0,8041 | 0,6481 | **0,9356** | 0,0555 | 0,8846 | 0,3809 | 0,0555 | 0,8754 |
| shuttle | 0,1977 |  |  | 0,9501 | 0,7506 | 0,9542 | 0,1424 | 0,2943 | 0,2184 | 0,0695 | **0,9546** |
| vowels | 0,5710 |  |  | 0,1364 | 0,2186 | 0,2791 | 0,3551 | 0,0831 | **0,5093** | 0,3224 | 0,1875 |

Tabulka 6.2: Výsledky hodnoty precision referenčných experimentov z PyOD.

vyplýva, že výsledky sa môžu jemne líšiť, ale nie nejak závažne. Niekedy ukazuje jedna metrika lepšie výsledky, inokedy zase druhá.

V tabuľkách 6.1 a 6.2 môžeme vidieť porovnanie nameraných výsledkov. Zvýraznené výsledky označujú výsledky, ktoré sú najlepšie pre danú dátovú sadu. Metódy SOS a COPOD sa nenachádzajú v referenčnom riešení preto tabuľka neobsahuje žiadne hodnoty. V tabuľke s mojimi nameranými hodnotami je prázdne políčko pri metóde SOS a dátovej sade *shuttle*. Pri tomto datasete sa nepodarilo spraviť ani jedno meranie, keďže celé vykonávanie programu zamrzlo.

Môžeme si všimnúť, že hodnoty sa vo väčšine prípadov relatívne zhodujú. Väčšie rozdiely si však môžeme všimnúť u metódy CBLOF a dátových sadách *ionosphere*, *letter*, *lympho* a *vowels*. Pri datasetoch *ionosphere*, *letter* a *vowels* vyšli moje experimenty o poznanie lepšie, rozdiel môže byť spôsobený jemným rozdielom vo vyhodnocovaní odľahlých hodnôt. Naopak pri datasete *lympho* vyšiel môj výsledok o poznanie menší a pri bližšom pohľade na výsledky jednotlivých iterácii v priečinku experiments_results
PrecisionForOdds je vidieť, že až vo dvoch iteráciach vyšiel výsledok precision rovný 0, čo výrazne ťahá priemer smerom dole. Pri datasete *lympho* je taktiež vyšší rozdiel v metóde kNN a to až 20%. Posledný väčší rozdiel je vidno pri metóde MCD a datasete *vowels*. Taktiež si možno všimnúť, že metóda Feature Bagging, ktorá v predvolenom nastavení používa metódu LOF, má s metódou LOF podobné výsledky. Teda ak LOF samostatne pracuje dobre, napríklad dátová sada *ionosphere*, tak aj FB má obdobne dobré výsledky. Avšak ak LOF nedokáže správne nájsť odľahlé hodnoty, tak FB nedokáže vylepšiť jej presnosť.

## 6.2 Experimenty na umelých dátach

Ďalším druhom experimentov, ktoré dané metódy absolvovali, boli experimenty na umelo vytvorených dátach. K vytvoreniu dát poslúžila metóda `generate_data` z PyOD knižnice, ktorá ako parametre berie počet trénovacích vzoriek, testovacích vzoriek, počet atribútov a

| Dáta | Počet vzoriek | Počet atribútov | Percento odľahlých hodnôt |
|---|---|---|---|
| generated data 1 | 6000 | 5 | 1 |
| generated data 2 | 6000 | 5 | 5 |
| generated data 3 | 6000 | 5 | 10 |
| generated data 4 | 6000 | 10 | 1 |
| generated data 5 | 6000 | 10 | 5 |
| generated data 6 | 6000 | 10 | 10 |

Tabulka 6.3: Popis generovaných dát.

percento odľahlých hodnôt. Pre účely experimentovania bolo vytvorených šesť variant dátových sad s rôznym percentom odľahlých hodnôt a rôznym počtom atribútov. Podrobnejší popis dátových sád je v tabuľke 6.3.

Tak ako pri reálnych dátach tak experimentovanie prebiehalo v desiatich iteráciach pre každú metódu. Rozdiel bol však v tom, že tu sa v každej iterácii vygenerovali nové dáta podľa zadaného počtu odľahlých hodnôt a počtu atribútov a tie následne boli normalizované. Následne boli výsledky z jednotlivých cykloch spriemerované a vyšla výsledná hodnota precision pre jednotlivé metódy. Výsledky experimentov sú v tabuľke 6.4.

| Dáta | ABOD | COPOD | SOS | PCA | MCD | OCSVM | LOF | CBLOF | KNN | FB | IForest |
|---|---|---|---|---|---|---|---|---|---|---|---|
| generated data 1 | 0,87 | 0,87 | 0 | 0,9 | 0,8 | 0,79 | 0,19 | **0,95** | 0,79 | 0,34 | 0,89 |
| generated data 2 | 0,87 | 0,95 | 0,054 | 0,894 | 0,848 | 0,784 | 0,022 | 0,984 | **0,986** | 0,106 | 0,998 |
| generated data 3 | 0,795 | 0,873 | 0,088 | 0,85 | 0,9 | 0,8 | 0,03 | **0,99** | 0,825 | 0,069 | 0,795 |
| generated data 4 | **1** | **1** | 0 | 0,7 | 0,9 | 1 | 0,04 | **1** | 0,8 | 0,15 | 0,9 |
| generated data 5 | **0,994** | 0,988 | 0 | 1 | 0,7 | 0,5 | 0,008 | 1 | 0,9 | 0,036 | 0,9 |
| generated data 6 | 0,962 | 0,898 | 0,027 | 0,7 | 0,868 | **0,909** | 0,008 | 0,9 | 0,9 | 0,026 | 0,9 |

Tabulka 6.4: Výsledky metriky precision na generovaných dátach.

Môžeme si všimnúť, že oproti reálnym dátam tak sa v týchto generovaných pohybujeme v rádovo vyšších hodnotách úspešnosti takmer všetkých metód. Najhoršie sa v tomto javí metóda SOS, ktorá nedokázala prekročiť hranicu 10% ani v jednom zo šiestich generovaných datasetov. V prípade metódy Feature Bagging, tak sa opäť potýkame z veľkou nepresnosťou vzhľadom k tomu, že základná metóda LOF, ktorú FB používa, tak ma veľmi slabé výsledky. Preto som sa rozhodol pridať tu ešte dva varianty FB, keď jedna z nich bude mať ako základnú metódu KNN a druhá PCA. Výsledky si môžeme všimnúť v tabuľke 6.5.

| Dáta | FB(LOF) | FB(KNN) | FB(PCA) |
|---|---|---|---|
| generated data 1 | 0,34 | 0,84 | 0.68 |
| generated data 2 | 0,106 | 0,89 | 0.894 |
| generated data 3 | 0,069 | 0,9 | 0,797 |
| generated data 4 | 0,15 | 0,78 | 0,9 |
| generated data 5 | 0,036 | 0,789 | 0,8 |
| generated data 6 | 0,026 | 1 | 0,781 |

Tabulka 6.5: Porovnanie precision hodnoty pre FB metódy z rôznymi základnými metódami.

## 6.3 Experimenty na dátach z priemyselnej siete

Posledná skupina experimentov prebehla na šiestich súboroch, ktoré pochádzajú z päťminutového okna fungovania priemyselnej siete (IEC104) a boli popísané v 3.3. Na rozdiel od reálnych dát z ODDS popísaných v 3.1, kde sú datasety určené na klasifikáciu a len nejakým kľúčom bolo určené, ktoré triedy sú odľahlé, tak to to je jasne dopredu dané. Vzhľadom k tomu, že súbory s dátami obsahujú malú vzorku dát, tak datasety nerozdeľujem na trénovaciu a testovaciu časť, ale metódy sú trénované na celej časti a na nej aj vyhodnotia výsledok. Vo výsledkoch detekcie neuvádzam hodnoty pre ABOD a SOS, keďže tieto metódy nedokázali spracovať vstupné údaje a buď vyhodili chybu alebo spadli.

| Dáta | COPOD | PCA | MCD | OCSVM | LOF | CBLOF | KNN | FB | IForest |
|---|---|---|---|---|---|---|---|---|---|
| IEC104-fm+tm-1day-1percent | 0 | 0,5 | 0,667 | 0,833 | **1** | 0,667 | **0,833** | 0,6 | **0,833** |
| IEC104-fm+tm-1day-5percent | 0,241 | 0,607 | 0,862 | 0,862 | **0,966** | 0,862 | 0,897 | 0,724 | 0,759 |
| IEC104-fm+tm-2day-1percent | 0,167 | 0,417 | **0,833** | 0,75 | 0 | **0,833** | **0,833** | 0 | 0,583 |
| IEC104-fm+tm-2day-5percent | 0,214 | 0,414 | 0,845 | 0,828 | 0,018 | **0,845** | 0,569 | 0,098 | 0,672 |
| IEC104-fm-1day-1percent | 0,667 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** |
| IEC104-fm-1day-5percent | 0,643 | 0,733 | **0,933** | **0,933** | **0,933** | **0,933** | 0,643 | 0 | 0,786 |

Tabulka 6.6: Výsledky experimentov na dátach z priemyselnej siete.

Vo výsledkoch, v tabuľke 6.6, si môžeme všimnúť, že najlepšie fungovala detekcia na dátovej sade `IEC104-fm-1day-1percent`, ktorá obsahovala len 1% odľahlých hodnôt a normálne hodnoty sa nachádzali v jednom zhluku. Najhoršie dopadla metóda COPOD, ktorá si vôbec nedokázala poradiť s dátami vo dvoch zhlukoch (súbormi označenými „fm+tm“).

# Kapitola 7

# Záver

Cieľom práce bolo vytvoriť aplikáciu na detekciu odľahlých hodnôt, ktorá užívateľovi ponúkne jednoduché užívateľské rozhranie, dovolí mu nahrávať dátové sady a následne ich analyzovať. Analýza sa vykonáva pomocou metód na detekciu odľahlých hodnôt alebo ich kombináciou. Následne aplikácia užívateľovi zobrazí výsledky detekcie spolu s pridanými štatistikami a taktiež dovolí užívateľovi vybrať dva alebo tri atribúty a v daných dimenziách mu zobrazí pôvodné dáta v 2D, resp 3D grafe.

Na začiatku som si preštudoval dostupné metódy na detekcie odľahlých hodnôt a vybral z nich podmnožinu, ktorá bola použitá v aplikácii. Taktiež som si vyhľadal niekoľko datasetov, ktoré boli pripravené na detekciu odľahlých hodnôt. V priebehu práce boli k tejto sade pridané aj datasety od vedúcej mojej práce, ktoré sú vhodné na detekciu odľahlých hodnôt. Na tejto množine datasetov a množine vybraných metód detekcie odľahlých hodnôt bola vykonaná sada experimentov a porovnaná s referenčnou sadou experimentov. Dosiahnuté výsledky môžu užívateľovi dať predstavu o výkonnosti jednotlivých metód a dopomôcť mu v rozhodovaní, ktoré metódy, prípadne aké váhy, im priradiť pri ich kombinácii metód v aplikácii.

Čo sa týka samotných výsledkov, tak pri generovaných dátach najhoršie dopadli metódy SOS a LOF, ktoré dosiahli veľmi slabé výsledky, rádovo v jednotkách percent. Ostatné metódy dosahovali priemerné skóre na úrovni približne 80% až 90%. Čo sa týka metód aplikovaných na ODDS datasety, tak tu boli výsledky jednotlivých metód o niečo vyrovnanejšie, metódy tu dosahovali skóre precision na úrovni 30% až 55% a najjednoduchšie bolo pre ne určiť odľahlé hodnoty v datasete *ionosphere* a najťažšie v datasete *vowels*. V prípade dát dodaných z priemyselnej siete, tak výsledky, ktoré dosahovali metódy COPOD a FB boli najhoršie, v priemere na úrovni 32%, resp. 40%. Ako najlepšie sa javili metódy MCD, OCSVM a CBLOF, ktoré dosahovali hodnoty precision na úrovni 85%. Z týchto hodnôt vyplýva, že je ťažké zovšeobecniť použitie jednej alebo viacerých metód na dataset z určitej oblasti a vždy je lepšie skúmať viacej možností.

Čo sa týka budúcej práce, tak by bolo vhodné zmeniť technológiu použitú na vytvorenie frontendu. Vzhľadom k tomu, že backend bol vytváraný v jazyku Python som sa od začiatku zameral aj na tvorbu GUI v tomto jazyku, čo môžem takto spätne považovať za chybu. Pri výbere technológie bolo vybrané PyQt, ktoré som zhodnotil ako najlepšiu možnosť z dostupných Python frameworkov, avšak postupom času som narazil na viacej problémov, ktoré by sa nevyskytovali v pokročilejších technológiách ako napríklad obmedzenia na použitie CSS vo verzii 2.

# Literatura


[1] Abhigyan. Different types of Outliers. *Medium*. 2020. Dostupné z: https://medium.com/analytics-vidhya/different-types-of-outliers-dd6363983744.

[2] Aggarwal, C. C. *Outlier Analysis*. 2nd. Springer Publishing Company, Incorporated, 2017. ISBN 978-3-319-47577-6.

[3] Amer, M., Goldstein, M. a Abdennadher, S. Enhancing one-class Support Vector Machines for unsupervised anomaly detection. In:. Srpen 2013, s. 8–15. DOI: 10.1145/2500853.2500857.

[4] Angiulli, F. a Pizzuti, C. Fast Outlier Detection in High Dimensional Spaces. In:. Srpen 2002, sv. 2431, s. 15–26. ISBN 978-3-540-44037-6.

[5] Arjaria, S. K., Rathore, A. S. a Cherian, J. S. Chapter 13 - Kidney disease prediction using a machine learning approach: A comparative and comprehensive analysis. In: N, P., Kautish, S. a Peng, S.-L., ed. *Demystifying Big Data, Machine Learning, and Deep Learning for Healthcare Analytics*. Academic Press, 2021, s. 307–333. DOI: https://doi.org/10.1016/B978-0-12-821633-0.00006-4. ISBN 978-0-12-821633-0. Dostupné z: https://www.sciencedirect.com/science/article/pii/B9780128216330000064.

[6] Arnold, B. C. Pareto Distribution. In: *Wiley StatsRef: Statistics Reference Online*. John Wiley & Sons, Ltd, 2015, s. 1–10. DOI: https://doi.org/10.1002/9781118445112.stat01100.pub2. ISBN 9781118445112. Dostupné z: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118445112.stat01100.pub2.

[7] Baniecki, H., Kretowicz, W., Piatyszek, P., Wisniewski, J. a Biecek, P. Dalex: Responsible Machine Learning with Interactive Explainability and Fairness in Python. *Journal of Machine Learning Research*. 2021, sv. 22, č. 214, s. 1–7. Dostupné z: http://jmlr.org/papers/v22/20-1473.html.

[8] Breunig, M., Kriegel, H.-P., Ng, R. a Sander, J. LOF: Identifying Density-Based Local Outliers. In:. červen 2000, sv. 29, s. 93–104. DOI: 10.1145/342009.335388.

[9] Campos, G., Zimek, A., Sander, J., Campello, R., Micenková, B. et al. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*. Červenec 2016, sv. 30. DOI: 10.1007/s10618-015-0444-8.

[10] Guo, Y., Jia, X. a Paull, D. Effective Sequential Classifier Training for SVM-Based Multitemporal Remote Sensing Image Classification. *IEEE Transactions on Image Processing*. Únor 2018, sv. 27, s. 3036 – 3048. DOI: 10.1109/TIP.2018.2808767.

[11] Han, J., Kamber, M. a Pei, J. *Data Mining: Concepts and Techniques*. 3rd. Morgan Kaufmann Publishers Inc., 2011. ISBN 978-0-12-381479-1.

[12] Hardin J., R. D. Outlier Detection in the Multiple Cluster Setting Using the Minimum Covariance Determinant Estimator. In:. 2004. Dostupné z: https://escholarship.org/uc/item/9pg389hg.

[13] He, Z., Xu, X. a Deng, S. Discovering cluster-based local outliers. *Pattern Recognition Letters*. 2003, sv. 24, č. 9, s. 1641–1650. DOI: https://doi.org/10.1016/S0167-8655(03)00003-5. ISSN 0167-8655. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0167865503000035.

[14] Hubert, L. J. a Arabie, P. Comparing partitions. *Journal of Classification*. 1985, sv. 2, s. 193–218.

[15] Hubert, M. a Debruyne, M. Minimum covariance determinant. *WIREs Computational Statistics*. 2010, sv. 2, č. 1, s. 36–43. DOI: https://doi.org/10.1002/wics.61. Dostupné z: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.61.

[16] Janssens, J. H. *Outlier Selection andOne-Class Classification*. 2013. Disertační práce. Tilburg University. Dostupné z: https://github.com/jeroenjanssens/phd-thesis/blob/master/jeroenjanssens-thesis.pdf.

[17] Kriegel, H.-P., Schubert, M. a Zimek, A. Angle-based outlier detection in high-dimensional data. In:. Srpen 2008, s. 444–452. DOI: 10.1145/1401890.1401946.

[18] Lazarevic, A. a Kumar, V. Feature bagging for outlier detection. In:. Leden 2005, sv. 21, s. 157–166. DOI: 10.1145/1081870.1081891.

[19] Li, Z., Zhao, Y., Botta, N., Ionescu, C. a Hu, X. COPOD: Copula-Based Outlier Detection. In:. Září 2020. DOI: 10.1109/ICDM50108.2020.00135.

[20] Linderman, G. C., Rachh, M., Hoskins, J. G., Steinerberger, S. a Kluger, Y. Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data. *Nature Methods*. Springer Science and Business Media LLC. feb 2019, sv. 16, č. 3, s. 243–245. Dostupné z: https://arxiv.org/abs/1712.09005.

[21] Liu, F. T., Ting, K. a Zhou, Z.-H. Isolation Forest. In:. Leden 2009, s. 413 – 422. DOI: 10.1109/ICDM.2008.17.

[22] Micenková, B., Beusekom, J. a Shafait, F. Stamp Verification for Automated Document Authentication. In:. Leden 2015, s. 117–129. ISBN 978-3-319-20124-5.

[23] Newman, D., Hettich, S., Blake, C. a Merz, C. *UCI Repository of machine learning databases*. 1998. Dostupné z: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[24] Phillip, W. Local Outlier Factor for Anomaly Detection. *WIREs Computational Statistics*. 2018. Dostupné z: https://towardsdatascience.com/local-outlier-factor-for-anomaly-detection-cc0c770d2ebe.

[25] Ramaswamy, S., Rastogi, R. a Shim, K. Efficient Algorithms for Mining Outliers from Large Data Sets. *SIGMOD Rec.* New York, NY, USA: Association for Computing Machinery. may 2000, sv. 29, č. 2, s. 427–438. DOI: 10.1145/335191.335437. ISSN 0163-5808. Dostupné z: https://doi.org/10.1145/335191.335437.

[26] Rayana, S. *ODDS Library*. 2016. Dostupné z: http://odds.cs.stonybrook.edu.

[27] Reiners T., W. D. A blackboard architecture applied to maximum likelihood clustering. In:. 2001. Dostupné z: https://escholarship.org/uc/item/9pg389hg.

[28] Shyu, M.-L., Chen, S.-C., Sarinnapakorn, K. a Chang, L. Principal Component-based Anomaly Detection Scheme. In:. Listopad 2005, sv. 9, s. 311–329. DOI: 10.1007/11539827_18. ISBN 3-540-28315-3.

[29] Singh, P., Singh, N., Singh, K. K. a Singh, A. Chapter 5 - Diagnosing of disease using machine learning. In: Singh, K. K., Elhoseny, M., Singh, A. a Elngar, A. A., ed. *Machine Learning and the Internet of Medical Things in Healthcare*. Academic Press, 2021, s. 89–111. DOI: https://doi.org/10.1016/B978-0-12-821229-5.00003-3. ISBN 978-0-12-821229-5. Dostupné z: https://www.sciencedirect.com/science/article/pii/B9780128212295000033.

# Příloha A

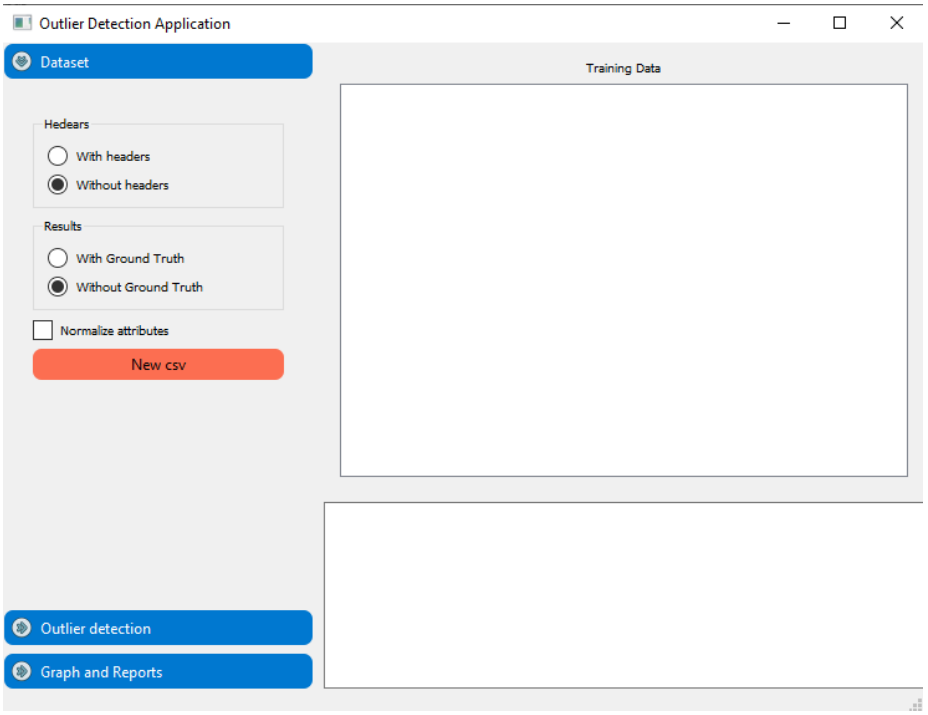
# Obsah DVD

Priložené DVD obsahuje:

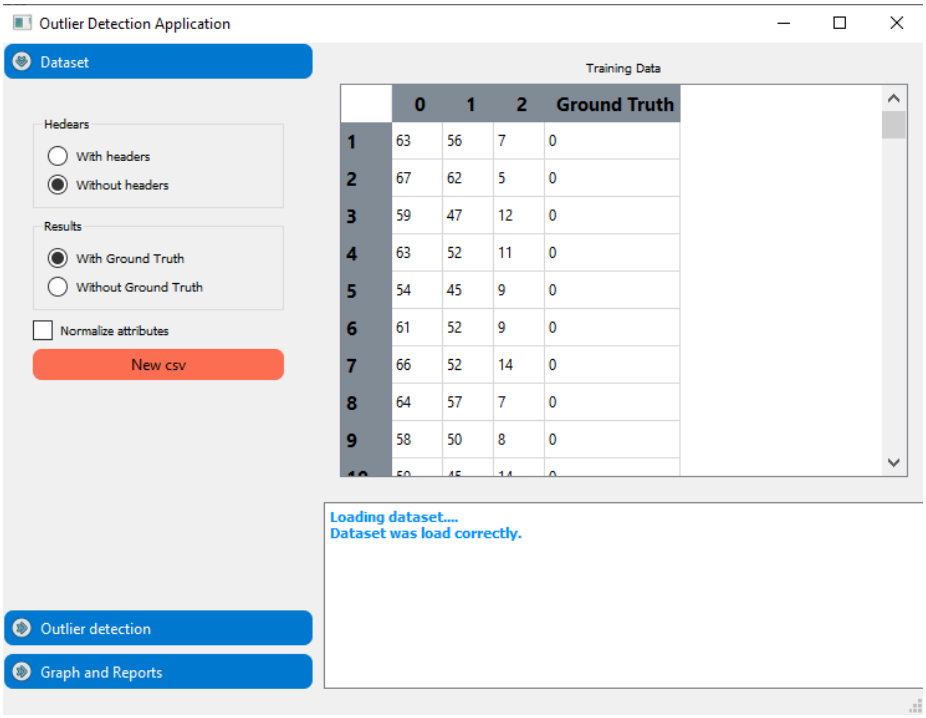
- *latex/* - zdrojové súbory tejto práce pre LaTeX
- *datasets/* - príklad použitých datasetov pri práci
- *experiments/* - zdrojové súbore, ktoré boli použité na tvorbu experimentov
- *experiments_results/PrecisionForIndustryData/* - výsledky experimentov pre jednotlivé iterácie na dátach z priemyselnej oblasti
- *experiments_results/PrecisionForGeneratedData/* - výsledky experimentov pre jednotlivé iterácie na generovaných dátach
- *experiments_results/PrecisionForOdds/* - výsledky experimentov pre jednotlivé iterácie na dátach z ODDS
- *icons/* - zložka z ikonami použitými v aplikácii
- *src/* - zdrojové súbory aplikácie

# Příloha B

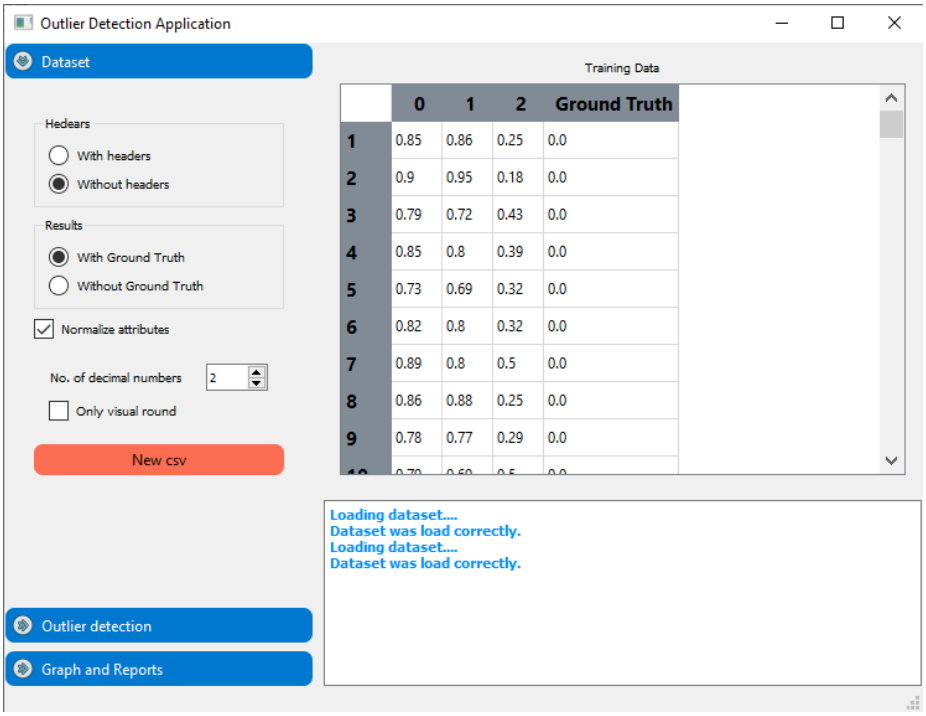
# Ukážky aplikácie



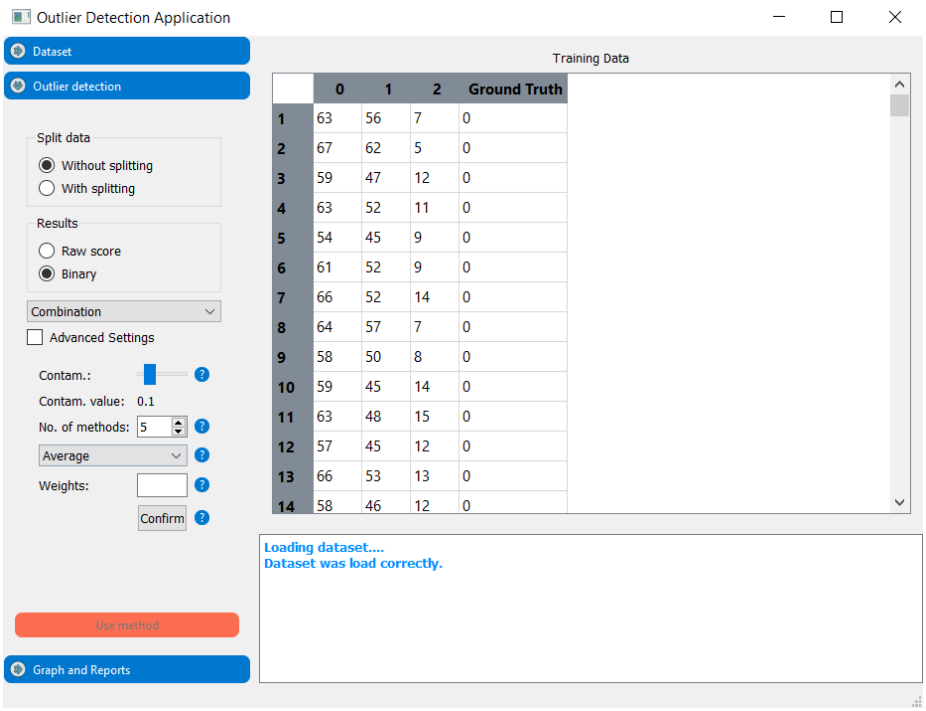

Obrázek B.1: Snímka úvodnej obrazovky aplikácie.



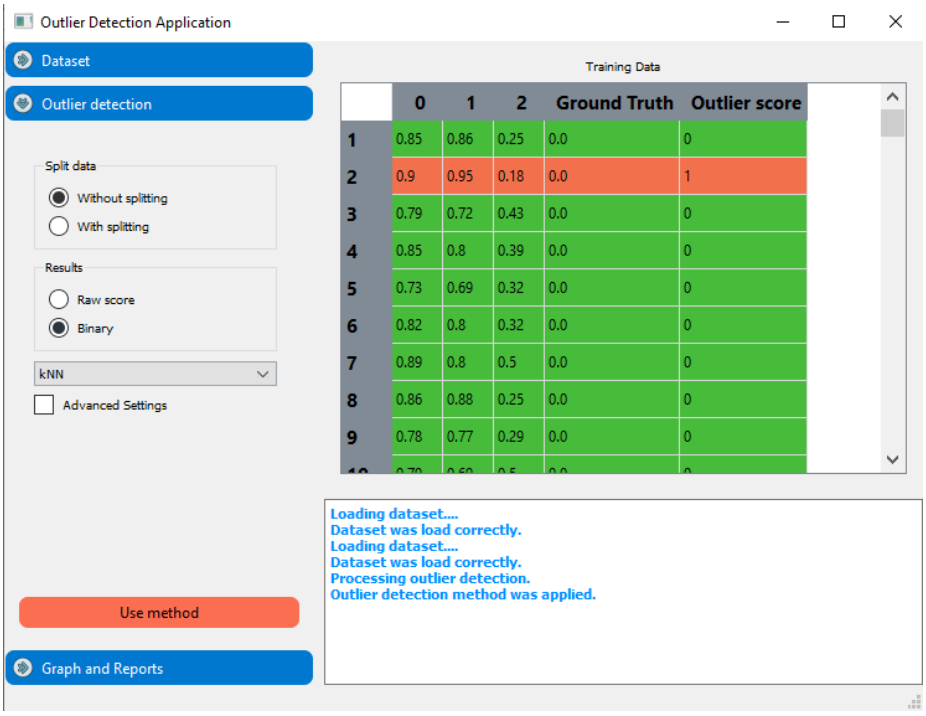

Obrázek B.2: Snímka aplikácie po nahraní anotovaného datasetu.



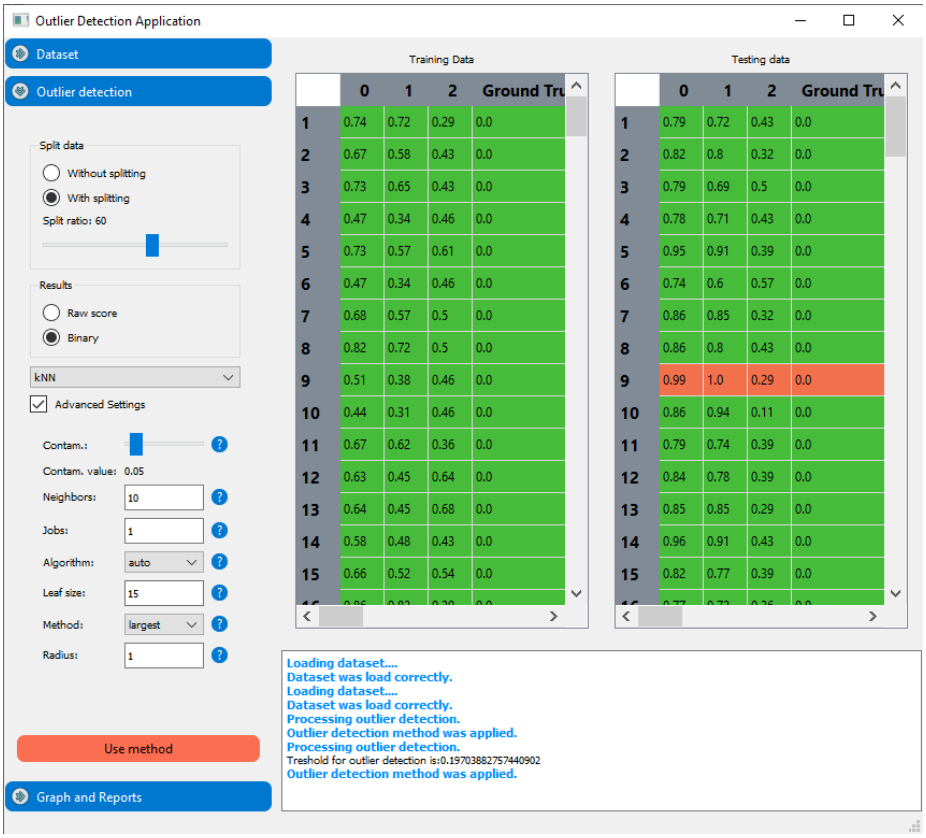

Obrázek B.3: Snímka aplikácie po nahraní anotovaného normalizovaného datasetu.



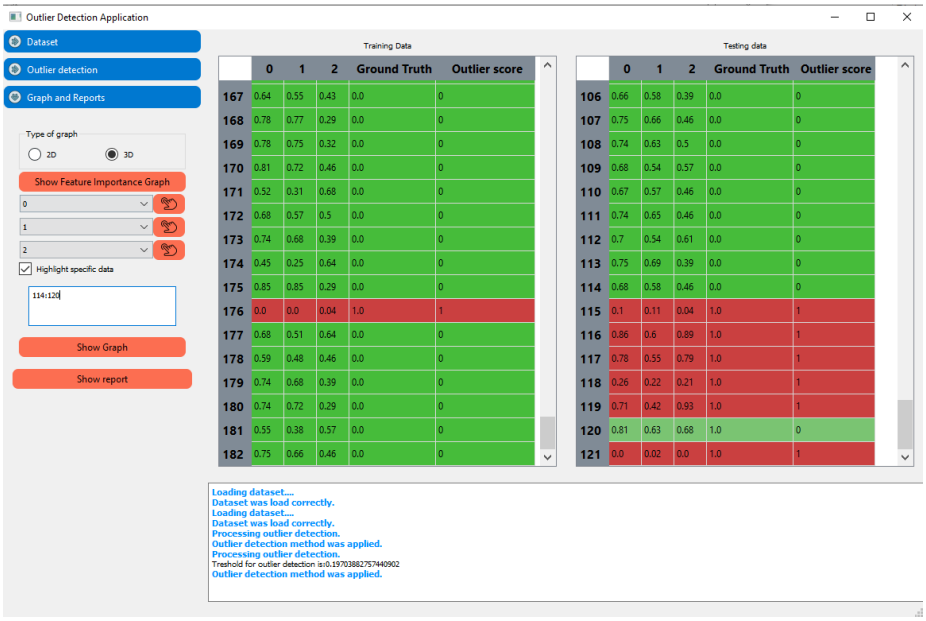

Obrázek B.4: Snímka aplikácie pred nastavením detekcie pomocou kombinácie metód.



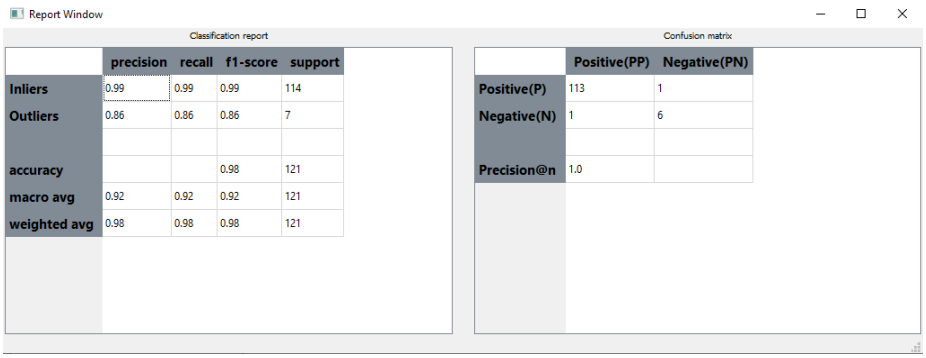

Obrázek B.5: Snímka aplikácie po použití metódy kNN na celú trenovaciu sadu.



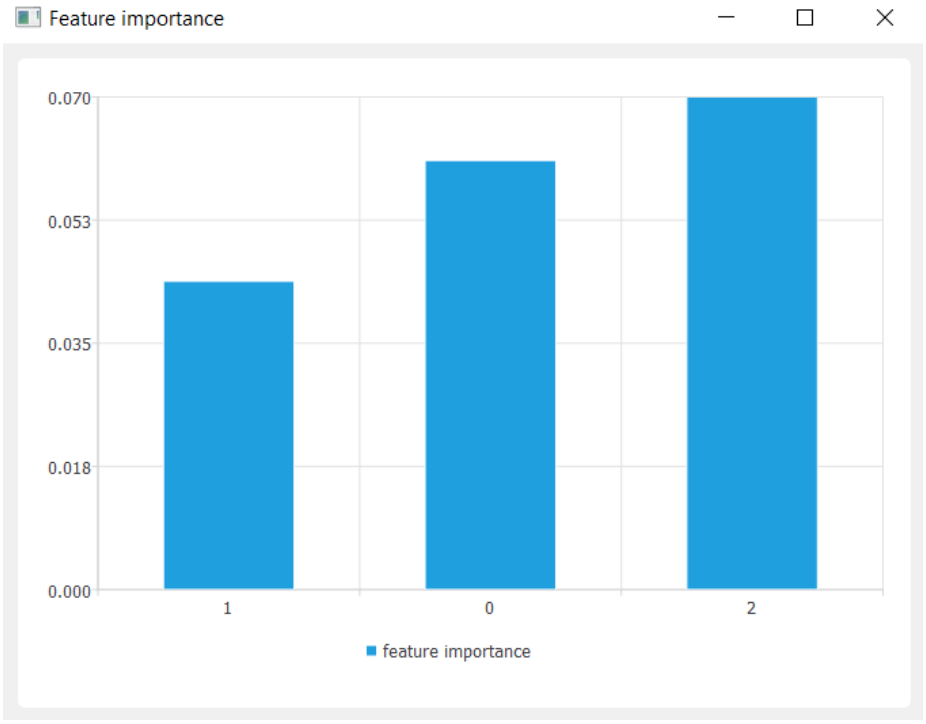

Obrázek B.6: Snímka aplikácie po rozdelení dátovej sady a použití metódy kNN.




Obrázek B.7: Snímka obrazovky predtým ako sa zobrazí 3D graf zo zvýraznenými hodnotami 114 až 120.




Obrázek B.8: Snímka obrazovky zobrazujúca report detekcie odľahlých hodnôt.




Obrázek B.9: Snímka obrazovky zobrazujúca dôležitosť jednotlivých atribútov.