VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

# METODY MĚŘENÍ ÚSPĚŠNOSTI DOLOVÁNÍ DAT

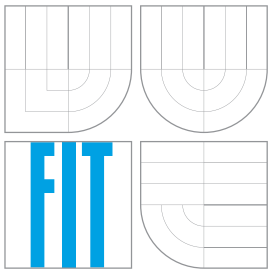
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAN TRUNKÁT

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

# METODY MĚŘENÍ ÚSPĚŠNOSTI DOLOVÁNÍ DAT

SUCCESS RATE MEASURE METHODS IN DATA MINING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAN TRUNKÁT

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR BARTÍK, Ph.D.

BRNO 2013

# Abstrakt

Bakalářská práce je zaměřená na měření úspěšnosti dolování dat v oblasti shlukovaní. Seznamuje se základními pojmy, vlastnostmi dolování dat a hlavně se shlukovou analýzou dat. V rámci práce byl vytvořen program, který implementuje metody měření úspěšnosti. V závěru práce jsou uvedeny výsledky úspěšnosti shlukování.



# Abstract

The Bachelor thesis is aimed at success rate measure methods in data mining in the area of clustering. It introduces the basic concepts, features of data mining and especially the cluster analysis. This work includes program, which implements methods of measuring success. In conclusion, they are given results of clustering success.



# Klíčová slova

dolování dat, získávání znalostí z databází, shlukování, shluková analýza, shluk, shlukovací metody

# Keywords

data mining, knowledge Discovery in Databases, clustering, cluster analysis, cluster, clustering methods


# Citace

Jan Trunkát: Metody měření úspěšnosti dolování dat, bakalářská práce, Brno, FIT VUT v Brně, 2013

# Metody měření úspěšnosti dolování dat

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vladimíra Bartíka, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jan Trunkát
14. května 2013

## Poděkování

Chtěl bych poděkovat panu Ing. Vladimíru Bartíkovi, Ph.D. za poskytnutí odborné pomoci při konzultacích, čím přispěl k vypracování této práce.

# Obsah

# Úvod

Množství dat, která se nahromadila za dobu používání výpočetní techniky zejména se vznikem databázových systému a používáním jich, je obrovské. Hledání nějakých souvislostí mezi daty v takovém množství a jejich využití je bez pomocných nástrojů téměř nemožné. Zde jako pomocný nástroj se objevil relativně mladý, ale rychle se rozvíjející obor *dolování dat*, který je schopný pomocí speciálních algoritmů automaticky objevovat v datech strategické informace. V dnešním světě se s tímto oborem potkáváme denně a ani o tom nevíme, ať už jde o nákup v obchodě nebo telefonování. Všechna tato data si společnosti ukládají pro analýzu, kde z této analýzy potom dělají většinu manažerských rozhodnutí. Tato práce je zaměřená hlavně na oblast *shlukování* a měření úspěšností algoritmů v oblasti shlukování. V Kapitole 1 jsou vysvětleny základní pojmy tohoto oboru, vysvětlen proces jak najít nějakou souvislost mezi daty, předzpracování dat, aby dolování bylo úspěšné a další věci potřebné získání souvislostí. V Kapitole 2 jsou popsány základní metody pro analýzu dat. Pro shlukovácí metody je vyhrazená samostatná Kapitola 3, která popisuje co vlastně shlukování je, jak se měří podobnost objektů mezi různými typy dat, jaké máme metody shlukové analýzy a na jakém základě měříme úspěšnost shlukovacích algoritmů. Shlukováním na konkrétních datech se práce zabývá v Kapitole 4. Konkrétní data se zpracovávají v program RapidMiner, což je nástroj pro *dolování dat*, kde je na data použita shlukovací metoda. V závěru této kapitoly je malé porovnání úspěšnosti shlukování. Kapitola 5 se zabývá implementací programu, který implementuje metody pro měření úspěšnosti shlukovácích algoritmů. V poslední Kapitole 6 se nachází zhodnocení dosažených výsledků.

# Kapitola 1

# Dolování dat

Kapitola vysvětluje základní pojmy a vlastnosti dolování dat.

## 1.1 Co je dolování dat

Nejdřív je třeba uvést názvosloví na pravou míru. Pojem *dolování dat (angl. data mining - DM)* může být chápan dvěma způsoby jednak jako jedno z alternativních jmen používaných pro přesnější název *získávání znalostí z databází (angl. Knowledge Discovery in Databases - KDD)* nebo jako jeden krok z procesu získávání znalostí. Definice KDD podle Fayyada [9] zní, že jde o netriviální získávání implicitních, skrytých, dříve neznámých a potenciálně užitečných informací/znalostí z dat.

Z definice vyplývají vlastnosti, které charakterizují zajímavost a to jsou:

- *netriviálnost*, kde informace nejde získat jednoduchým dotazem (např.: SQL dotaz nad databází),
- *skrytost*, kde informace nejde na první pohled vidět,
- *dříve neznáma znalost* má-li být informace zajímavá musí být pro nás nová,
- *potenciální užitečnost*, kde získaná informace, má význam pro další rozhodování.

Pojem *dolování dat* se dnes používá častěji než *získávání znalostí z databází*, proto v této práci bude využíván pojem *dolování dat*, ale jak bude dále ukázáno, je to jen jeden krok v procesu získávání znalostí. Dolování dat prostupuje mnoha disciplínami jako je např. statistika, informatika, umělá inteligence nebo databáze.

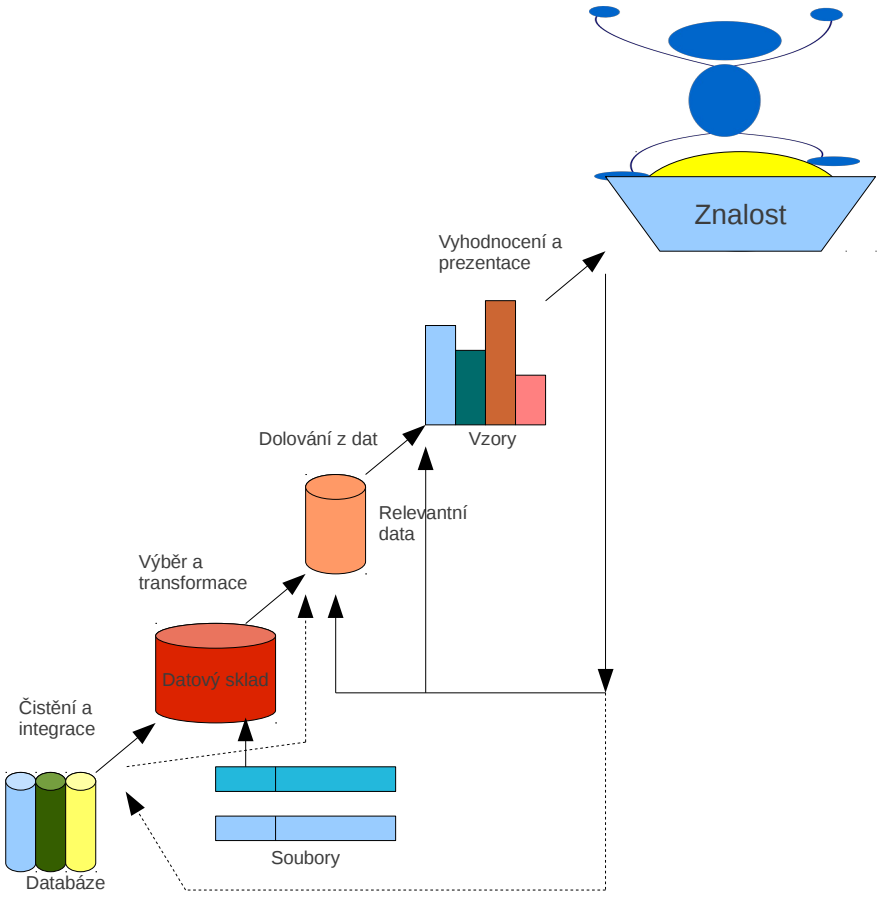
## 1.2 Proces získávání znalostí

Proces získávání znalostí je zobrazen v obrázku 1.1 a skládá se z následujících iterativních kroků [3]:

1. *Čištění dat*
2. *Integrace dat*
3. *Výběr dat*
4. *Transformace dat*

5. *Dolování dat* - aplikace určité metody a konkrétního algoritmu extrahovat z dat vzory
6. *Hodnocení vzorů* - identifikace zajímavých vzorů
7. *Prezentace znalostí* - prezentace výsledků dolování uživateli využitím technik vizualizace a reprezentace znalostí

Aby dolování a nalezení znalosti v datech bylo úspěšné je třeba samotná data předzpracovat. Pro předzpracování dat slouží kroky 1-4 v procesu získávání znalostí a jsou vysvětleny v podkapitole 1.4.




Obrázek 1.1: Proces získávání znalostí inspirováno z [3]

## 1.3 Druhy datových zdrojů pro dolování

Dolování může být použito na jakémkoliv druhu dat a to jak uložených v nějakých úložištích tak i transientních, jako jsou proudy dat. Zde jsou uvedeny nejčastější zdroje:

- *relační databáze* - kolekce tabulek, z nichž každá má unikátní název,
- *datové sklady* - jsou to úložiště informací/dat z několika různých zdrojů,

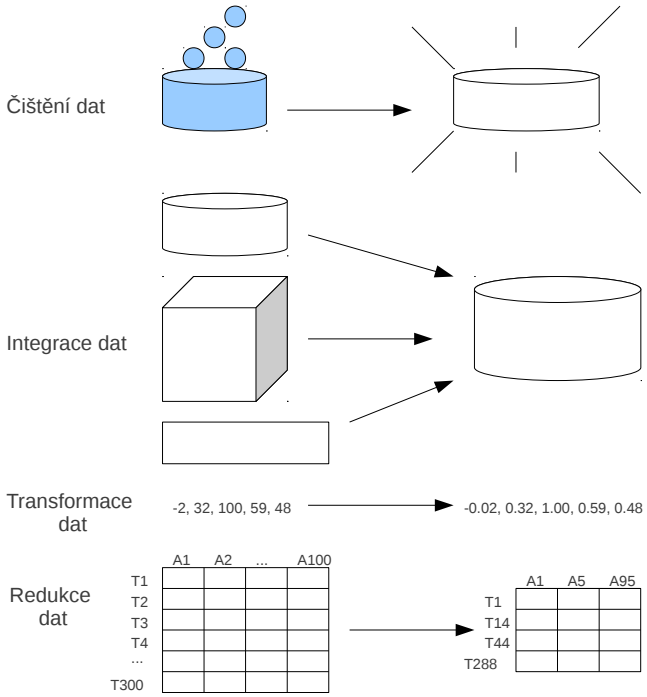
- *transakční databáze* - databáze obsahuje množinu transakcí, kde každá z nich obsahuje množinu položek.

## 1.4 Předzpracování dat

Předzpracování dat je důležitá součást procesu získávání znalostí, protože pokud nejsou kvalitní data, nebudou ani kvalitní výsledky samotného dolování. Data v reálném světě nikdy nejsou dokonalá, protože pochází z různých zdrojů. Projevy nekvality dat:

- *nekompletnost*
- *obsahující šum*
- *nekonzistentnost*
- *chybnost*
- *nejednoznačnost*

Všechny tyto projevy nekvality jdou odstranit díky úlohám předzpracováním dat, kterými jsou: *čistění dat*, *integrace dat*, *výběr dat* a *transformace dat*. Názorně jsou zobrazeny úlohy v Obrázku 1.2.




Obrázek 1.2: Předzpracování dat převzato z [3]

- **Čištění dat** - odstranění šumu a nekonzistence dat
- **Integrace dat** - integrace dat, které jsou z různých zdrojů ⇒ sjednocení dat

- **Výběr dat** - vybrat data, která jsou pro naše řešení vhodná
- **Transformace dat** - zmenšení objemu dat (např.: sumarizace, normalizace)

## 1.5 Typy atributů

V závislosti na vlastnostech oboru hodnot rozlišujeme tři základní typy: [4]

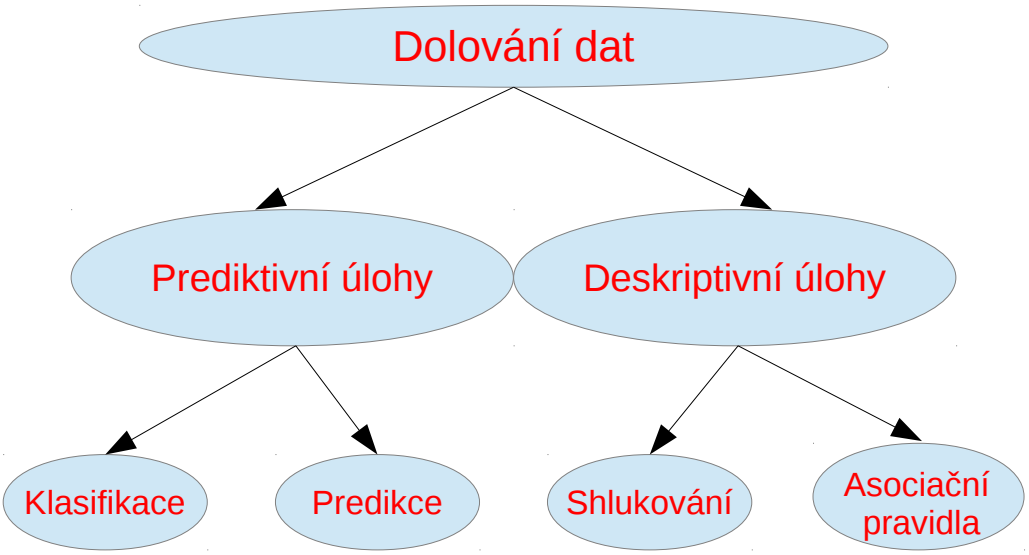
- *kvantitativní (numerický, spojitý, intervalový) atribut* - numerické hodnoty, obor hodnot je „nekonečný/velký“, je definováno uspořádání, např.: věk nebo váha
- *kategorický atribut* - diskrétní hodnoty, obor hodnot je většinou „malý“, není definováno uspořádání. např.: barva nebo město

  speciálním případem je *binární atribut*, který má jen dvě hodnoty např.: pohlaví
- *ordinální atribut* je kategorický atribut s tím, že má definované pořadí na oboru hodnot

# Kapitola 2

# Metody dolování dat

I když předzpracování dat je velmi důležité tak jádrem celého procesu získávání znalostí je správný výběr a použití dolovacích metod. Metody mohou být podle podobné charakteristiky rozděleny do dvou základních skupin jak je zobrazeno na Obrázku 2.1. [6]




Obrázek 2.1: Typy dolovacích úloh inspirace z [6]

- *Prediktivní skupina* - předpovídají budoucí hodnoty dat na základě doposud získaných hodnot dat. Mezi prediktivní úlohy zejména zapadá *klasifikace* a *predikce*.
- *Deskriptivní skupina* - cílem těchto metod je analyzovat data a zkoumat nové vzájemné vztahy, které ještě doposud nebyly objeveny a jsou potencionálně užitečné. Typická metoda pro deskriptivní úlohy je *shlukování*.

Níže jsou popsány základní a používané metody.

## 2.1 Klasifikace

Cílem klasifikace je roztřídit data do skupin podle jejich vlastností. Klasifikace je třífázový proces, kdy se data rozdělí na dvě množiny: [3]

- *1.Fáze* nebo-li *fáze učení* - jsou vybrány vzorky dat do množiny, která se nazývá *trénovací*. Na základě *trénovací množiny* klasifikátor vytvoří *klasifikační model*. Podle vytvořených pravidel se konkrétní objekty budou zařazovat do daných skupin. Abychom mohli vytvořit *trénovací množinu*, musíme vědět do jakých skupin data patří.
- *2.Fáze* nebo-li *testovací fáze* - opět jsou vybrány vzorky dat do množiny, ale tentokrát *testovací*. Vybrané vzorky by měly být nezávislé na datech, které byly vybrány do *trénovací množiny*. Pomocí *klasifikačního modelu* vytvořeného v první fázi jsou data zařazována do tříd. Jestli dochází ke správnému rozřazování do skupin můžeme procentuálně zjistit díky známosti skutečné třídy a podle výsledného rozřazení do skupin pomocí klasifikátoru. Když přesnost klasifikátoru bude považována za přijatelnou, tak v budoucnu může být klasifikátor použit na data, kde nebudeme vědět zařazení do skupin.
- *3.Fáze* je fáze užití, kdy aplikujeme model na data.

V následujících podkapitolách je základní výčet typů klasifikačních modelů.

### 2.1.1 Rozhodovací stromy

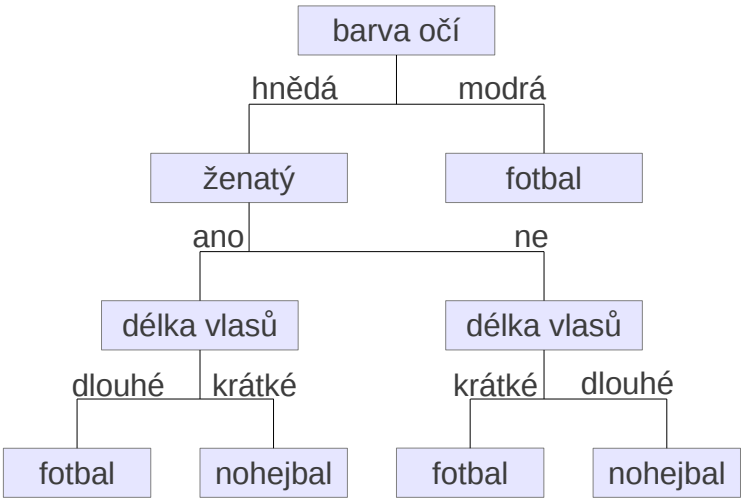
Stromy jsou známé a oblíbené struktury. Při jejich tvorbě se postupuje metodou *rozděl a panuj*. Trénovací data se postupně rozdělují na menší a menší podmnožiny (uzly stromu) tak, aby v podmnožinách stromu byly příklady jedné skupiny. Stromem se postupuje od kořene až k spodním listům čili shora-dolů. Obecný algoritmus pro tvorbu rozhodovacího stromu se nazývá *top down induction of decision trees* (TDIDT) a jeho algoritmus: [7]

1. zvol jeden atribut jako kořen dílčího stromu,
2. rozděl data v tomto uzlu na podmnožiny podle hodnot zvoleného atributu a přidej uzel pro každou podmnožinu,
3. existuje-li uzel, pro který nepatří všechna data do téže třídy, pro tento uzel opakuj postup od bodu 1, jinak skonči.

Pro ukázku vytvoření jednoduchého stromu je vybrán příklad z [5], kde studenti si mají vybrat mezi fotbalem nebo nohejbalem. Ukázka trénovací množiny je v následující tabulce 2.2.Jedna z možností vytvoření stromu je na Obrázku 2.2. Z rozhodovacího stromu je patrné, že všichni modroocí hrají fotbal. Pro hnědooké studenty je kritický faktor jestli je ženatý/vdaná. Jestli ano, tak dlouhovlasí hrají fotbal a krátkovlasí hrají nohejbal. Jestli nejsou, tak krátkovlasí hrají fotbal a dlouhovlasí nohejbal.

| barva očí | ženatý/vdaná | pohlaví | délka vlasů | sport |
|---|---|---|---|---|
| hnědá | ano | muž | dlouhé | fotbal |
| modrá | ano | muž | krátké | fotbal |
| hnědá | ano | muž | dlouhé | fotbal |
| hnědá | ne | žena | dlouhé | nohejbal |
| hnědá | ne | žena | dlouhé | nohejbal |
| modrá | ne | muž | krátké | fotbal |
| hnědá | ne | žena | dlouhé | nohejbal |
| hnědá | ne | muž | krátké | fotbal |
| hnědá | ano | žena | krátké | nohejbal |
| hnědá | ne | žena | dlouhé | nohejbal |
| modrá | ne | muž | dlouhé | fotbal |
| modrá | ne | muž | krátké | fotbal |

Tabulka 2.1: Trénovací množina




Obrázek 2.2: Rozhodovací strom

### 2.1.2 Bayesovská klasifikace

Metoda vychází z Bayesovy věty o podmíněných pravděpodobnostech 2.1.

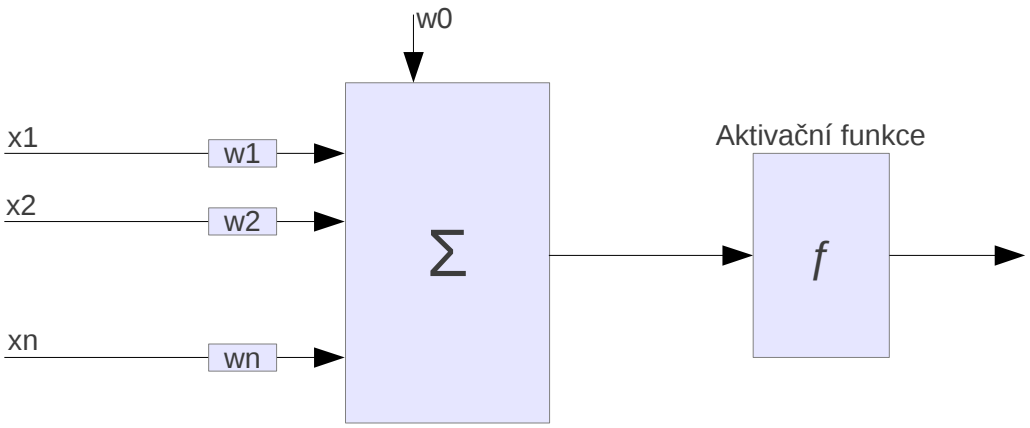
$$P(H \mid E) = \frac{P(E \mid H)P(H)}{P(E)}, [7] \tag{2.1}$$

kde vyjadřujeme pravděpodobnost hypotézy H, pokud pozorujeme evidenci E. Hypotéza nemusí být stažená jen k jedné evidenci a také hypotéz může být více. *Naivní bayesovský klasifikátor* vychází z předpokladu, že jednotlivé evidence $E_1$,...,$E_k$ jsou podmíněně nezávislé při platnosti hypotézy H a vychází ze vzorce 2.2.

$$P(H \mid E_1, ..., E_k) = \frac{P(E_1, ..., E_k \mid H)P(H)}{P(E_1, ..., E_k)}, [7] \tag{2.2}$$

### 2.1.3 Neuronové sítě

Skládají se z umělých neuronů, které mají inspiraci v biologickém neuronu. Propojená soustava biologických neuronů vede signály a každý neuron v soustavě na ně reaguje. Umělé neurony fungují stejným způsobem, kdy neuron přijímá několik signálu „na vstupu“ a jakmile, suma signálů překročí daný práh, tak se aktivuje. Základní schéma je zobrazeno na obrázku 2.3.




Obrázek 2.3: Schéma umělého neuronu podle [7]

Vstupem jsou signály označené $x_1, x_2, ..., x_n$. Každý signál $x_j$ je násoben váhou $w_j$, tento součet vstupuje do součtového členu, kde je vytvořen vážený součet: [7]

$$\text{SUM} = w * x = \sum_{j=1}^{n} w_j * x_j \tag{2.3}$$

Když vážený součet přesáhne práh $w_0$, je výstup roven 1, jinak 0, tzn. vstupy jsou tedy libovolná čísla, výstupy zůstávají dvouhodnotové.

## 2.2 Predikce

Metody na rozdíl od klasifikace předpovídají spojitou hodnotu atributu. Metody vycházející z predikce jsou např. různé varianty *regrese*. *Regrese* se používá k odhadu budoucích hodnot na základě minulých hodnot pomocí osázení bodů na křivku.

### 2.2.1 Lineární regrese

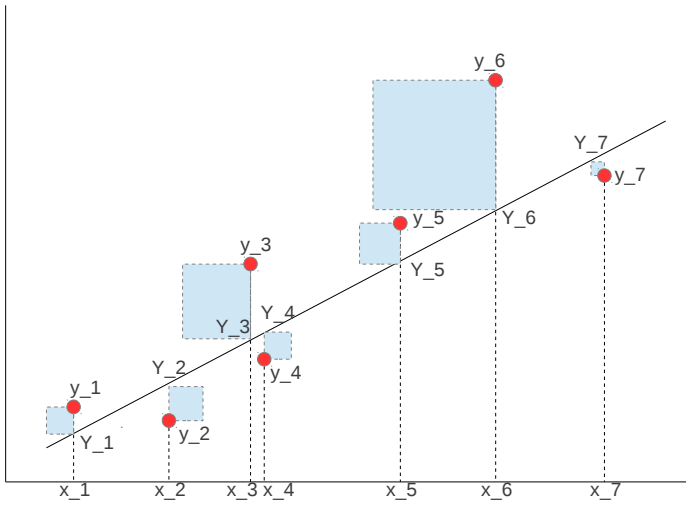
Metoda předpokládá lineární vztah mezi vstupními daty a výstupními daty, tzn. data ve tvaru $(x_1, y_1)$, $(x_2, y_2)$,...,$(x_n, y_n)$, kde $x_l$ jsou vstupní data a $y_l$ výstupní data, jejichž hodnota se bude předpovídat pro nová data, pro všechna $l = 1,...,s$. Hodnota je aproximována přímkou o rovnici 2.6: [4]

$$Y = aX + b, \tag{2.4}$$

kde:

- $x_1, x_2, ..., x_n$ - jsou dané vstupní data
- $y_1, y_2, ..., y_n$ - jsou skutečné výstupní data
- $Y_1, Y_2, ..., Y_n$ - jsou hodnoty po dosazení vstupního parametru do rovnice přímky

Nejznámější metodou pro určení koeficientů $a$ a $b$ je *metoda nejmenších čtverců*, která je založena na principu, že součet druhých mocnin rozdílu skutečné hodnoty $y_l$ a aproximované hodnoty $Y_l$ je minimální viz. Obrázek 2.4.




Obrázek 2.4: Schéma jednoduché lineární regrese inspirováno z [4]

Pro *metodu nejmenších čtverců* tedy můžeme koeficienty $a$ a $b$ dopočítat podle vzorců 2.5:

$$a = \frac{\sum_{l=1}^{n}(x_l - \overline{x})(y_l - \overline{y})}{\sum_{l=1}^{n}(x_l - \overline{x})^2}; \qquad b = \overline{y} - a\overline{x}, \tag{2.5}$$

kde $\overline{x}$ je aritmetický průměr všech hodnot $x$ a $\overline{y}$ je aritmetický průměr všech hodnot $y$.

## 2.3 Asociační pravidla a frekventované vzory

Jsou založeny na pravděpodobnosti výskytu stejných objektů u sebe. Mezi nejznámější příklad patří analýza nákupního košíku, kde sledujeme výskyt stejných věcí v jednom nákupu. Tento výskyt potom vede k určení *frekventovaných vzorů*. Pravidla pracují se syntaxí *IF-THEN* a zajímá nás, kolik příkladů splňuje předpoklad a kolik závěr pravidla, kolik případů splňuje předpoklad i závěr současně atd., tedy zajímá nás, jak pro pravidlo [7]

$$Ant \Longrightarrow Suc, \tag{2.6}$$

| | **Suc** | **¬Suc** | $\sum$ |
|---|---|---|---|
| **Ant** | a | b | r |
| **¬Ant** | c | d | s |
| $\sum$ | k | l | n |

Tabulka 2.2: Kontingenční tabulka

kde *Ant* (předpoklad, levá strana pravidla, antecedent) i *Suc* (závěr, pravá strana pravidla, sukcedent) jsou kombinace kategorií, vypadá příslušná kontingenční tabulka. Pro $n$ příkladů je její podoba v tabulce 2.2:

kde:

- $n(Ant \wedge Suc) = a$ je počet objektů pokrytých současně předpokladem i závěrem,
- $n(Ant \wedge \neg Suc) = b$ je počet objektů pokrytých předpokladem a nepokrytých závěrem,
- $n(\neg Ant \wedge Suc) = c$ je počet příkladů nepokrytých předpokladem ale pokrytých závěrem,
- $n(\neg Ant \wedge \neg Suc) = d$ je počet příkladů nepokrytých ani předpokladem ani závěrem,
- $n(Ant) = a + b = r$,
- $n(\neg Ant) = c + d = s$,
- $n(Suc) = a + c = k$,
- $n(\neg Suc) = b + d = l$,
- $n = a + b + c + d$.

Z těchto čísel (místo o počtu objektů pokrytých kombinací se někdy mluví o četnosti resp. frekvenci kombinace) můžeme počítat různé charakteristik pravidel a kvantitativně tak hodnotit nalezené znalosti. Mezi nejznámější metodu pro hledání pravidel je *algoritmus apriori*.

### 2.3.1 Algoritmus apriori

Algoritmus vyhledává často se opakující množiny položek.
**Algoritmus apriori:**

1. do $L_1$ přiřaď všechny kategorie, které dosahují alespoň požadované četnosti
2. polož k = 2
3. dokud $L_{k-1} \neq \oslash$
   - 3.1. pomocí funkce *apriori-gen* vygeneruj na základě $L_{k-1}$ množinu kandidátů $C_k$
   - 3.2. do $L_k$ zařaď ty kombinace z $C_k$, které dosáhly alespoň požadovanou četnost
   - 3.3. zvětši počítadlo $k$

**Funkce apriori-gen($L_k$):**

1. pro všechny dvojice kombinací $Comb_p$, $Comb_q$ z $L_{k-1}$

   1.1. pokud $Comb_p$ a $Comb_q$ se shodují v $k$ - 2 kategoriích, přidej $Comb_p \cap Comb_q$ do $C_k$

2. pro každou kombinaci $Comb$ z $C_k$

   2.2. pokud některá z jejich podkombinací délky $k$ - 1 není obsažena v $L_{k-1}$, odstraň $Comb$ z $C_k$

## 2.4 Shlukování

Této analýze je věnována celá Kapitola 3.

# Kapitola 3

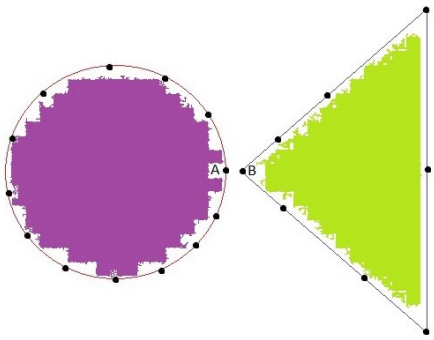
# Shluková analýza dat

Shlukování je analýza, která rozděluje objekty do shluků (angl. cluster). Rozdělování probíhá na základě podobnosti objektů a to především tak, aby dva objekty stejného shluku si byly více podobné, než dva objekty z různých shluků. Každý objekt musí být charakterizován pomocí svých vlastností, aby mohla být změřena *podobnost objektů* (např.: rostlina délkou a šířkou listů).Z hlediska strojového učení oproti klasifikaci jde o učení bez učitele tzn. předem nevíme do jakých shluků objekty zařadit a ani kolik bude shluků. Informace pro tuto kapitoly byly převážné čerpány z [3], [8], [2]. Měření podobnosti objektů je dvojího způsobu: *konvenční metoda* a *konceptuální metoda*. *Konvenční metoda* shlukování určuje shluky podle měření podobnosti (3.1), která je pouze funkcí z vlastností hodnot objektů (jejich atributů):

$$podobnost(A, B) \quad = \quad f(vlastnosti(A), vlastnosti(B)), \tag{3.1}$$

kde A a B je označení dvou objektů. *Konceptuální metoda* shlukování, kde shluky jsou založeny na *konceptuální soudržnosti*, která je funkcí nejen v vlastností hodnot objektů (jejich atributů), ale také jiných faktorech: *popisný jazyk* $L$, který systém používá pro popis tříd (skupin) objektů, a *okolí* $E$, což je množina sousedících vzorů:

$$\text{konceptuální soudržnost}(A, B) \quad = \quad f(vlastnosti(A), vlastnosti(B), L, E). \tag{3.2}$$




Obrázek 3.1: Rozdíl mezi konvenční metodou a konceptuální metodou [1]

V Obrázku 3.1 je uveden příklad, kde objekty jsou zobrazeny v dvourozměrném poli. Z bodů jsou vytvořeny obrazce trojúhelníku a kruhu. Kdyby šlo o konvenční metodu body A a B by byly ve stejném shluku, protože jejich vzdálenost je nejmenší. Avšak konceptuální soudržnost těchto shluků je malá, neboť patří do konfigurací reprezentující různé koncepty tudíž nebudou ve stejném shluku [1].

## 3.1 Typy dat

Data se převážně nachází v datových strukturách v podobě matice. Matice je definovaná objektem (záznamem) a jeho vlastnostmi (atributy) různých typů. Nejběžnější algoritmy používají jednu z následujících datových struktur:

- **Datová matice**: Tato matice reprezentuje $n$ objektů (např.: rostlin) pomocí $p$ proměnných (vlastností, atributů, např.: délka a šířka listů). Struktura je formou relační tabulky, nebo matice $n * p$:

$$\begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix}$$

- **Podobnostní matice**: Tato matice obsahuje vzdálenosti pro všechny dvojice objektů reprezentovaných jako $n * n$ tabulka:

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix},$$

  kde $d(i,j)$ je měřený rozdíl nebo odlišnost mezi objekty $i$ a $j$. Obecně je $d(i,j)$ nezáporné číslo, jehož hodnota se blíží k 0, jestliže objekty $i$ a $j$ jsou si velmi podobné. Se zmenšující se podobností objektů se hodnota $d(i,j)$ zvětšuje. Vzhledem k $d(i,j) = d(j,i)$ a $d(i,i) = 0$ dostáváme uvedenou trojúhelníkovou matici.

Abychom mohli změřit podobnost objektů, musíme vědět jakých hodnot proměnné nabývají. Proměnné můžou být *intervalové, ordinální, binární, nominální nebo poměrové*. Pro tyto typy proměnné je několik způsobů jak určit podobnost (vzdálenost) objektů a jsou popsány v následujících kapitolách.

### 3.1.1 Intervalové proměnné

Jsou takové proměnné, kdy pro dvě hodnoty můžeme vypočítat rozdíl mezi nimi např.: výška nebo teplota vzduchu. Podobnost objektů u intervalových proměnných se měří pomocí vzdálenostní funkce, kde mezi nejznámější patří: *Euklidovská, Manhattanská, Minkovského vzdálenost*. Problémem u těchto proměnných je, že jejich jednotky hodnot mohou být měněny, což vede k jiným výsledným rozdělení objektů do shluků, proto je dobré provádět

standardizaci jednotekt tzn. aby byly převedeny na bezjednotkové proměnné. Jak se může provést standardizace je zobrazeno ve vzorečcích:

1. Vypočítáme střední odchylku $s_f$:

$$s_f = \frac{1}{n}(\mid x_{1f} - m_f \mid + \mid x_{2f} - m_f \mid + \cdots + \mid x_{nf} - m_f \mid), \tag{3.3}$$

   kde $x_{1f}$,..., $x_{nf}$, je $n$ hodnot proměnné $f$ a $m_f$ je střední hodnota $f$.

2. Vypočítáme z-transformaci (standardizované hodnot):

$$z_{if} = \frac{x_{if} - m_f}{s_f} \tag{3.4}$$

Standardizace nemusí být vždy prospěšná pro všechny aplikace, proto záleží na uživateli jestli ji bude provádět nebo ne. Následně určujeme podobnost objektů pomocí vzdálenostních funkcí, pro které platí:

- $d(i, j) \geqq 0$ - vzdálenost je nezáporné číslo
- $d(i, i) = 0$ - vzdálenost objektu k sobě samému je 0
- $d(i, j) = d(j, i)$ - vzdálenost je symetrická funkce
- $d(i, j) \leqq d(i, h) + d(h, j)$ - vzdálenost mezi objekty i a j v prostoru nesmí být větší než součet vzdáleností objektů i a j od objektu h (trojúhelníková nerovnost)

**Nejznámější vzdálenostní funkce:**

- *Euklidovská vzdálenost ($D_E$)*

$$D_E(i, j) = \sqrt{\mid x_{i1} - x_{j1} \mid^2 + \mid x_{i2} - x_{j2} \mid^2 + \cdots + \mid x_{ip} - x_{jp} \mid^2}, \tag{3.5}$$

- *Manhattanská vzdálenost ($D_B$)*

$$D_B(i, j) = \mid x_{i1} - x_{j1} \mid + \mid x_{i2} - x_{j2} \mid + \cdots + \mid x_{ip} - x_{jp} \mid, \tag{3.6}$$

- *Minkovského vzdálenost ($D_M$)*

$$D_M(i, j) = \sqrt[q]{\mid x_{i1} - x_{j1} \mid^q + \mid x_{i2} - x_{j2} \mid^q + \cdots + \mid x_{ip} - x_{jp} \mid^q}, \tag{3.7}$$

### 3.1.2 Binární proměnné

Jsou proměnné, které mohou nabývat jen dvou hodnot: 0 a 1. Ku příkladu může být, že člověk vlastní auto, když vlastní auto tak je ohodnocen jako 1, ostatní 0. Máme dva typy binárních proměnných:

- *Symetrické* - oba stavy jsou stejně pravděpodobné, mají stejnou hodnot a váhu (např.: pohlaví). Pro měření podobnosti se využívá koeficient shody:

$$d(i,j) \quad = \quad \frac{r+s}{q+r+s+t}, \tag{3.8}$$

kde $r$ je počet proměnných, které mají hodnotu 1 pro objekt $i$ a hodnotu 0 pro objekt $j$, $s$ je počet proměnných, které mají hodnotu 0 pro objekt $i$ a hodnotu 1 pro objekt $j$, $q$ je počet proměnných, které mají hodnotu 1 pro oba objekty $i$ a $j$, $t$ je počet proměnných, které mají hodnotu 0 pro oba objekty $i$ a $j$.

- *Asymetrické* - obě hodnoty nejsou stejně významné (např.: test na nemoc, kdy pozitivní výsledek je důležitější než negativní). Pro stanovení podobnosti objektů se používá Jaccardův koeficient:

$$d(i,j) \quad = \quad \frac{r+s}{q+r+s}. \tag{3.9}$$

### 3.1.3 Nominální proměnné

Jsou zobecněné binární proměnné, mohou nabývat více hodnot a hodnoty nejsou číselné. Hodnoty jsou ale předem definovány a jejich počet omezen, Příkladem nominálních hodnot může být kraj(Moravskoslezský, Jihomoravský, ....). Podobnost se určíme podle koeficientu prosté shody:

$$d(i,j) \quad = \quad \frac{p-m}{p}, \tag{3.10}$$

kde $m$ je počet shod (počet proměnných, které mají stejnou hodnotu pro objekty $i$ a $j$) a $p$ je celkový počet proměnných.

### 3.1.4 Ordinální proměnné

Jsou podobné jako nominální, ale navíc u jejich hodnot je stanovené pořadí. Příkladem může být stupeň dosaženého vzdělání: základní, středoškolské nebo vysokoškolské, kde vysokoškolské má největší hodnotu. Podobnost objektů se určuje podobně jako u intervalových proměnných. Hodnotám se musí přiřadit číselná hodnota. Jestliže proměnná $f$ může nabývat $M_f$ hodnot, potom se dá přiřadit hodnoty 1,...,$M_f$. Potom jestliže pro objekt $i$ má proměnná $f$ hodnotu $x_{if}$, můžeme tuto hodnotu nahradit příslušnou číselnou hodnotou $r_{if} \in \{1, ...., M_f\}$. Ještě je nutné transformovat data, aby měla stejnou váhu, protože ordinální proměnné mohou mít různý počet stavů. Transformace se provede na základě:

$$z_{if} \quad = \quad \frac{r_{if}-1}{M_f-1}. \tag{3.11}$$

Transformace převede všechny hodnoty na interval (0,1). Z hodnot $z_{if}$ už určíme vzdálenost pomocí nějaké vzdáleností funkce.

### 3.1.5 Poměrové proměnné

Mají všechny znaky nominální (lze porovnat), ordinální (lze stanovit pořadí), intervalové (můžeme určit rozdíl) a přidává navíc poměr, kolikrát je jedna hodnota větší (resp. menší) než druhá. Hodnotami jsou pouze kladná čísla. Příkladem může být počet obyvatel města. Podobnost jde změřit třemi způsoby:

- zpracovávat je jako intervalové proměnné, může dojít ke zkreslení výsledků,
- aplikovat logaritmickou transformaci, výsledné hodnoty potom zpracovávat jako intervalové proměnné, hodnotu $x_{if}$ proměnné $f$ můžeme transformovat jako:

$$y_{if} = \log(x_{if}), \tag{3.12}$$

- zpracovávat $x_{if}$ jako ordinální data a jejich rozsah zpracovat jako intervalové proměnné.

## 3.2 Metody shlukové analýzy

Cílem metod je seskupit objekty do shluků nebo vytvořit hierarchii shluků objektů. Základní rozdělení tradičních metod můžeme dělit jako:

- *metody rozkladu a*
- *metody hierarchické.*

Existují i novější přístupy, které rozšiřují počet metod pro shlukování a jsou to např.:

- *metody založené na mřížce,*
- *metody založené na modelu,*
- *metody založené na hustotě.*

V následujících podkapitolách jsou vysvětleny metody jak fungují a popsány některé jejich algoritmy, které vyžívají.

### 3.2.1 Metody rozkladu

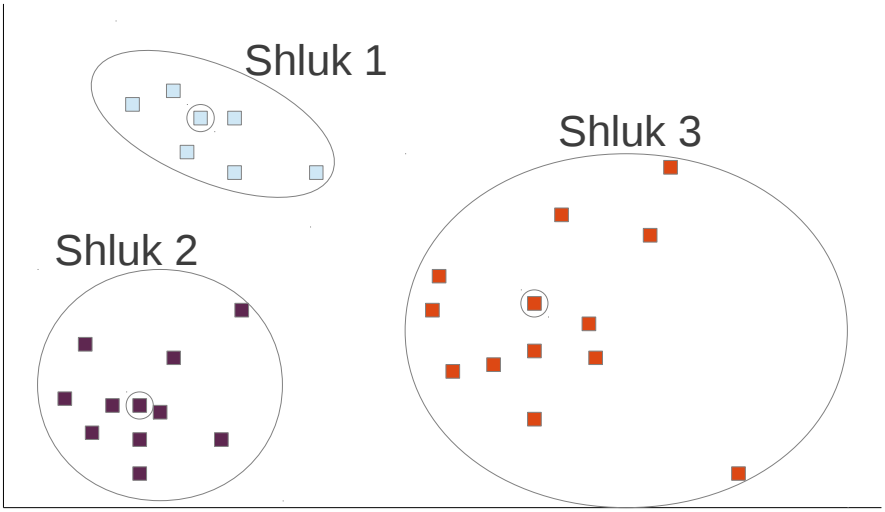
Metody roztřiďují objekty do předem určeného počtu shluků, přičemž každý shluk musí obsahovat alespoň jeden objekt a každý objekt patří pouze do jednoho shluku. Vztah můžeme vyjádřit následovně:

$$\text{počet shluků} \leqq \text{počet objektů}. \tag{3.13}$$

Při tomto typu shlukování je vytvářen konkrétní počet shluků. Nejprve se náhodně vyberou objekty, které budou dočasně tvořit střed shluků. Ostatní objekty se následně podle podobnosti objektů (vzdálenostní funkce) přiřadí k nejbližším středům. Tento děj se děje iterativně, kdy dochází vždy k přepočtu středů, které nejlépe vystihují třídu, do té doby dokud podobnost objektů uvnitř jednoho shluku je největší. Pro nalezení středů se používají různé heuristiky:

- *metoda založená na centrálním bodu (metoda k-průměrů, k-means method)*
- *metoda založená na reprezentujícím objektu (k-medoids method).*

Jedná se o metody pro pevné shlukování, kde problémem je stanovení počtu shluků, do kterých se budou přiřazovat objekty. Jak může vypadat takové rozdělení objektů do shluků je zobrazeno na Obrázku 3.2, kde kostičky v kroužku představují výsledné středové objekty shluků.




Obrázek 3.2: Ukázka rozdělení shluků

### Metoda založená na centrálním bodu (metoda k-průměrů, k-means method)

*K-means metoda* je jedna z nejpoužívanějších a nejstarších metod pro rozdělování objektů do tříd. Jde o metodu, která vychází z počátečního rozdělení do $k$ shluků ($k$ musí zadat uživatel). Shluky jsou vytvářeny pomocí dočasných fiktivních centrálních bodů, které tvoří dočasný střed shluků. Poté se objekty rozdělují do shluků pomocí jejich vzdáleností od centrálních bodů. Po rozdělení všech objektů se spočítají nové centrální body pro již vytvořené shluky a znova se spočítají vzdálenosti objektů od centrálních bodů, jestliže objekt má nyní blíž k jinému shluku reprezentovaným novým centrálním bodem tak se přesune do nového shluku. V praxi se většinou cyklus ukončí pokud už nedojde k žádnému přesunu objektu.

### Metoda založená na reprezentujícím objektu (k-medoids method)

*K-medoids metoda* taky vychází z počátečního rozdělení do $k$ shluků. Rozdílem je už výběr centrálního bodu. Centrální bod již není fiktivní, ale je zjištěn *medoid*, což je konkrétní objekt z dat. Počáteční *medoid* se vybírá objekt takový, aby součet vzdáleností jednotlivých objektů ve shluku od toho vybraného objektu byl minimální. Metoda pracuje dále už stejně jako *metoda K-means* dokud dochází ke zlepšování kvality shluků.

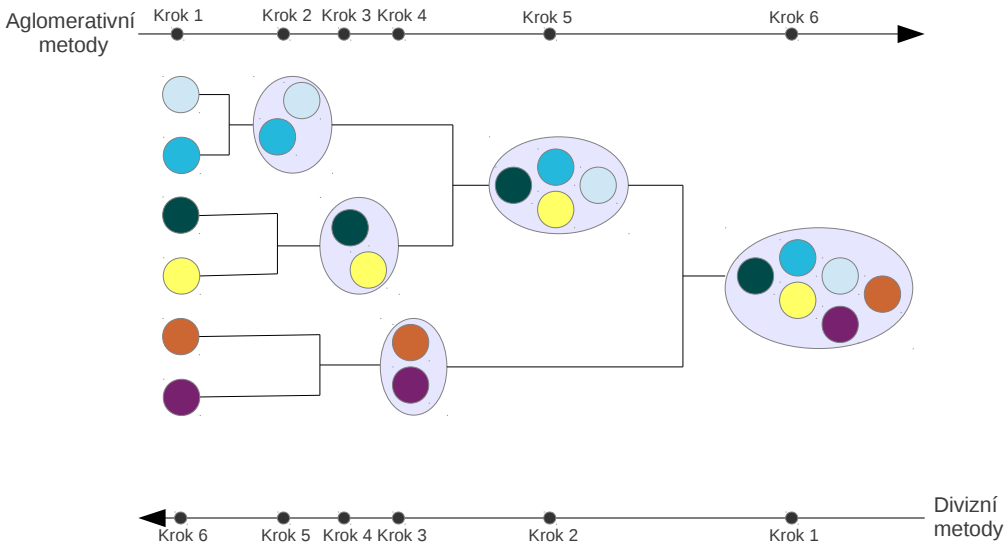
### 3.2.2 Metody hierarchické

Výsledkem metod je vytvoření nebo rozklad hierarchie skupin objektů => vzniká strom shluků. Můžeme je rozdělit do dvou základních skupin:

- *aglomerativní metody (shlukující hierarchické metody, metody shlukování zdola-nahoru)*
- *divizní metody (rozdělující hierarchické metody, metody shlukování shora-dolů).*

#### Aglomerativní metody (shlukující hierarchické metody, metody shlukování zdola-nahoru)

Každý objekt na začátku tvoří shluk. Následně postupuje po krocích, kdy vždy spojí dva nejpodobnější shluky. Pokračuje tak dlouho dokud nevznikne jeden velký shluk všech objektů. Stromovému diagramu, který vzniká tímto slučováním, se říká *dendogram* a je zobrazený na Obrázku 3.3.




Obrázek 3.3: Dendrogram

#### Divizní metody (rozdělující hierarchické metody, metody shlukování shora-dolů)

Fungují přesně naopak, kdy všechny objekty jsou přiřazeny do jednoho shluku a postupně jsou rozdělovány na menší shluky dokud každý objekt netvoří samostatný shluk nebo nedosáhne požadovaného rozdělování. Postup rozkladu je zobrazen na Obrázku 3.3.

### 3.2.3 Metody založené na hustotě

Vytvářejí shluky na základě četnosti objektů v prostoru. Za shluk se považují místa v prostoru s velkou četností objektů oddělené místy s malou četností objektů. Objekty, které se nacházejí v místech s malou četností, se považují za šum. Metody tímto způsobem umožňují

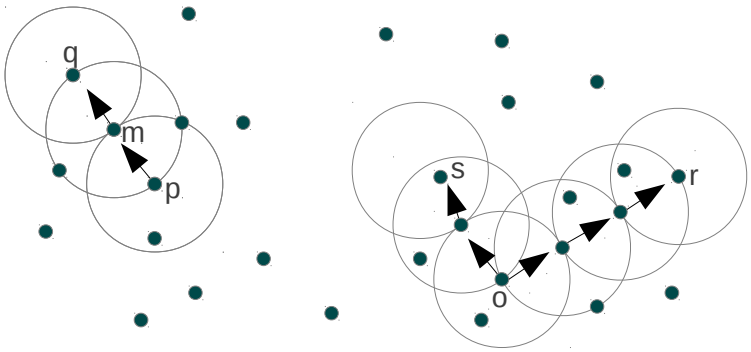
hledat shluky různých tvarů, odstraňují šum a odlehlé hodnoty. Metody založené na hustotě: DENCLUE nebo DBSCAN.

### DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Metoda pracuje s pojmem okolí bodu o poloměru $\varepsilon$, tedy tzv. $\varepsilon$-okolí. Jestliže $\varepsilon$-okolí bodu (objektu) obsahuje stanovený *minimální počet objektů*, je objekt označen za centrum shluku. Na Obrázku 3.4 je zobrazené hledání shluků pomocí tohoto algoritmu. Minimální počet objektů = 3.

- Z označených bodů $m$, $p$, $o$ a $r$ jsou jádrové objekty, protože každý z nich ve svém $\varepsilon$-okolí obsahuje alespoň tři objekty.
- $q$ je *přímo dosažitelné na základě hustoty* z $m$ a $m$ je *přímo dosažitelné na základě hustoty* z $p$ a naopak.
- $q$ je *(nepřímo) dosažitelné na základě hustoty* z $p$, protože $q$ je *přímo dosažitelné na základě hustoty* z $m$ a $m$ je *přímo dosažitelné na základě hustoty* z $p$. Objekt $p$ je *nepřímo dosažitelné na základě hustoty* z $q$, protože $q$ není jádrový objekt. Obdobně objekty $r$ a $s$ jsou *dosažitelné na základě hustoty* z $o$ a $o$ *dosažitelné na základě hustoty* z $r$.
- $o$, $r$ *a* $s$ jsou *spojené na základě hustoty*

DBSCAN nejprve projde celou databázi a zkontroluje $\varepsilon$-okolí každého objektu. Jestliže $\varepsilon$-okolí některého objektu obsahuje *minimální počet objektů*, potom je vytvořen shluk, jehož jádro tvoří objekt $p$. Metoda potom iterativně hledá objekty, které jsou přímo dosažitelné na základě hustoty z jader shluků, což může vést ke spojování shluků. Proces končí, když k žádnému shluku nemůže být připojen žádný další objekt. Problémem tohoto algoritmu je správné zvolení parametrů $\varepsilon$ a *minimální počet objektů*.




Obrázek 3.4: Dosažitelnost na základě hustoty a spojení na základě hustoty inspirováno z [3]

### 3.2.4 Metody založené na mřížce

Rozdělují prostor s objekty na konečný počet buněk, které tvoří mřížku. Na mřížce probíhá veškeré shlukování. Výhoda tohoto přístupu je nízká časová náročnost, která závisí na počtu buněk v mřížce. Metody založené na mřížce: STING nebo WaveCluster.

### 3.2.5 Metody založené na modelech

Předpokládají pro každý shluk nějaký matematický model, ke kterému hledají nejlepší přiřazení dat. Metody založené na modelech: Expectation-Maximization.

#### Expectation-Maximization

Na metodu se může pohlížet jako na rozšíření metody k-means, kde místo toho, aby přiřazoval objekty do vyhrazených shluků, tak na základě pravděpodobnosti (váhy) určuje příslušnost objektu ke shluku (neexistují pevné hranice mezi shluky jako u k-means). Tomuto kroku se říká *expectation step*. Dalším krokem je *maximization step*, kde tyto pravděpodobnosti se využívají pro výpočet parametrů nových reprezentujících objektů shluků. Parametry nových objektů slouží k určení distribučních funkcí, aby pravděpodobností rozložení co nejvíce odpovídalo zpracovávané datové množině.

## 3.3 Metody měření úspěšnosti se známým zařazením do skupin

Metody porovnávají výsledky s předpokládaným zařazením do shluků. Symbolem C je označena struktura, která je výsledkem nějakého shlukovacího algoritmu. Symbolem P je označena předpokládaná (známá) struktura. Pro porovnání výsledků se známým zařazením do skupin můžeme využít:

- *Kritéria založená na mírách podobností* nebo
- *Kritéria založená na konfuzní matici.*

### 3.3.1 Kritéria založená na mírách podobností

Následně jsou zavedeny symboly odpovídající počtům párů:

- $a =$ počet páru objektu ve stejném shluku v $\mathbf{C}$ i $\mathbf{P}$,
- $b =$ počet páru ve stejném shluku v $\mathbf{C}$, ale v různých shlucích v $\mathbf{P}$,
- $c =$ počet páru ve stejném shluku v $\mathbf{P}$, ale v různých shlucích v $\mathbf{C}$,
- $d =$ počet páru v různých shlucích v obou strukturách $\mathbf{C}$ i $\mathbf{P}$,
- $m_1 = a + b$, počet párů patřících do stejného shluku v $\mathbf{C}$,
- $m_2 = a + c$, počet párů patřících do stejného shluku v $\mathbf{P}$,
- $M = a + b + c + d$, celkový počet možných párů objektů.

Výsledky indexů nabývají hodnot od 0 do 1. Čím je hodnota vyšší, tzn. blíží se k 1, tím je souhlas mezi strukturami vyšší. Hodnota 1 představuje úplně stejné zařazení objektů do shluků v obou strukturách. Hodnota 0 potom představuje odlišné zařazení objektů do shluků.

#### Randův index (koeficient)

Je určen jako podíl párů ve stejném shluku v P i C nebo v rozdílných shlucích v P i C na celkový počet párů: [8]

$$R = \frac{a+d}{M}. \tag{3.14}$$

#### Jaccardův koeficient

Je určen jako podíl počtu objektů, které jsou ve stejném shluku v obou strukturách P i C, a počtu párů, které jsou ve stejných shlucích alespoň v jedné ze struktur: [8]

$$J = \frac{a}{a+b+c}. \tag{3.15}$$

#### Folkesův-Mallowsův index

Je geometrický průměr dvou relativních četností. První z nich je určen jako podíl počtu objektů, které jsou ve stejném shluku v obou strukturách P i C, a počtu párů patřících do stejného shluku v C, druhá relativní četnost je určena jako podíl počtu objektů, které jsou ve stejném shluku v obou strukturách P i C, a počtu párů patřících do stejného shluku v P: [8]

$$FM = \sqrt{\frac{a}{a+b}\frac{a}{a+c}} = \frac{a}{\sqrt{m_1 m_2}}. \tag{3.16}$$

### 3.3.2 Kritéria založená na konfuzní matici

Počet shluků v C i P se předpokládá stejný. Shluky uspořádáme, aby první shluk v C a první shluk v P měli nejvíce stejných objektů. Toto provedeme i pro ostatní shluky. Počet stejných objektů ve shlucích se zapíše na diagonálu *konfuzní matice* a jsou označeny $n_{hh}$. Hodnocení metody lze provádět podle *míry nesouladu*.

#### Míra nesouladu

Je určena jako: [8]

$$MD = \frac{n - \sum_{h=1}^{k} n_{hh}}{n}, \tag{3.17}$$

kde $n$ je počet objektů. Když by struktury byly stejné, potom by hodnota byla 0. Čím více jsou struktury rozdílné, tím víc se jejich hodnota blíží k 1 $\Rightarrow$ vysoká míra nesouladu. Vysoká míra nesouladu indikuje špatné zvolení shlukovací metody.

### Koeficient přesnosti

Hodnotí vytvořené shluky podle vzorce: [8]

$$P_{hh'} = \frac{n_{hh'}}{n_h}, \tag{3.18}$$

kde $n_{hh'}$ je počet společných objektů dvou shluků (jeden z předpokládané struktury P a druhý ze získané struktury C) a $n_h$ je počet objektů shluku ze struktury C.
Koeficient nabývá hodnot od 0 do 1. Když by všechny objekty shluku $C_h$ byly současně objekty určitého shluku $P_{h'}$, potom by pro jeden shluk $P_{h'}$ byla výsledkem hodnota 1 a pro ostatní 0 a šlo by se o optimální stav.

### Koeficient úplnosti

Je podobný *koeficientu přesnosti*, ale oproti němu počítá s počtem objektu ve struktuře P, tedy $n_{h'}$ je počet objektů shluku ze struktury P: [8]

$$R_{hh'} = \frac{n_{hh'}}{n_h}, \tag{3.19}$$

Koeficient nabývá hodnot od 0 do 1. Pokud by všechny objekty shluku $P_{h'}$ byly současně objekty určitého shluku $C_h$, potom by pro jeden shluk $P_{h'}$ byla výsledkem hodnota 1 a pro ostatní 0 a šlo by se o optimální stav.

# Kapitola 4

# Dolování konkrétních dat

Programů, které jsou využívány pro získávání znalostí, je spousta. Pro tuto práci je využíván program RapidMiner verze 5.3. RapidMiner je open-source dostupný na adrese: *http://rapid-i.com/*.

## 4.1 Vstupní data

Pro shlukování je vybrán data set *Iris*, který je i součastí RapidMineru. *Iris* je druh rostliny Kosatec a v datech se vyskytují jeho tři poddruhy: *iris-setosa*, *iris-versicolor* a *iris-virginica*. V data setu se nachází 150 objektů z toho každý poddruh po 50 objektech. Každý objekt má čtyři atributy: délku okvětního lístku (angl. petal length), šířku okvětního lístku (angl. petal width), délku kališního lístku (angl. sepal length) a šířku kališního lístku (angl. sepal width). Ukázka vstupních dat v programu RapidMiner je zobrazena na Obrázku 4.1.
Za povšimnutí stojí hodnota *label*, podle které jsou data klasifikována. Podle hodnoty label jsme schopni odhadnout, že naše předpokládána (známá) struktura se skládá ze 3 shluků a každý shluk má 50 objektů.

## 4.2 Sestavení schéma v RapidMineru

Prostředí RapidMiner po spuštění je zobrazeno na Obrázku 4.2, kde v zeleném rámečku se přepíná mezi dvěma hlavními částmi programu a to jsou:

- *design perspektive*, což je návrhový mód, který slouží k volbě a nastavení samotnému dolování na vybraná data a
- *result perspektive*, což slouží k prohlížení výsledků po dolování na datech.

### Design perspektive

Návrhový mód je zobrazen na Obrázku 4.2, kde v levé sloupci se nachází operátory (Operators) a vzorky dat (Repositories), uprostřed (okno Process), kde sestavujeme jak bude celý proces probíhat tzn. načtení dat, předzpracování, výběr metod, případný export do externího souboru. Sloupec vpravo slouží k nastavení parametrů operátorů.

ExampleSet (150 examples, 2 special attributes, 4 regular attributes)

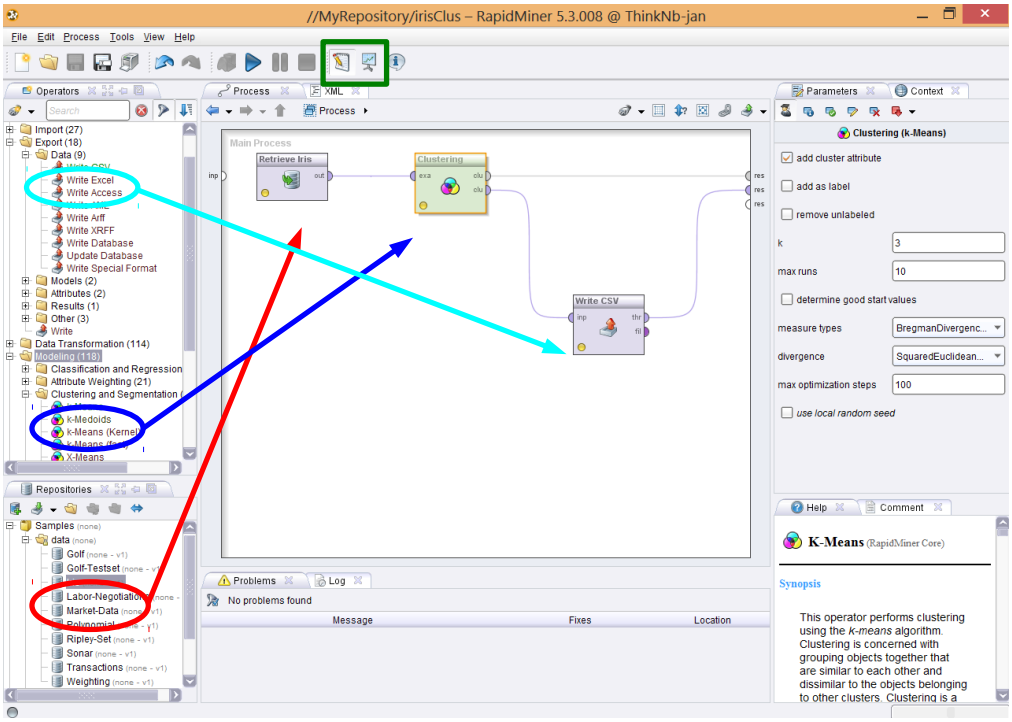
| Row No. | id | label | a1 | a2 | a3 | a4 |
|---|---|---|---|---|---|---|
| 37 | id_37 | Iris-setosa | 5.500 | 3.500 | 1.300 | 0.200 |
| 38 | id_38 | Iris-setosa | 4.900 | 3.100 | 1.500 | 0.100 |
| 39 | id_39 | Iris-setosa | 4.400 | 3 | 1.300 | 0.200 |
| 40 | id_40 | Iris-setosa | 5.100 | 3.400 | 1.500 | 0.200 |
| 41 | id_41 | Iris-setosa | 5 | 3.500 | 1.300 | 0.300 |
| 42 | id_42 | Iris-setosa | 4.500 | 2.300 | 1.300 | 0.300 |
| 43 | id_43 | Iris-setosa | 4.400 | 3.200 | 1.300 | 0.200 |
| 44 | id_44 | Iris-setosa | 5 | 3.500 | 1.600 | 0.600 |
| 45 | id_45 | Iris-setosa | 5.100 | 3.800 | 1.900 | 0.400 |
| 46 | id_46 | Iris-setosa | 4.800 | 3 | 1.400 | 0.300 |
| 47 | id_47 | Iris-setosa | 5.100 | 3.800 | 1.600 | 0.200 |
| 48 | id_48 | Iris-setosa | 4.600 | 3.200 | 1.400 | 0.200 |
| 49 | id_49 | Iris-setosa | 5.300 | 3.700 | 1.500 | 0.200 |
| 50 | id_50 | Iris-setosa | 5 | 3.300 | 1.400 | 0.200 |
| 51 | id_51 | Iris-versicolc | 7 | 3.200 | 4.700 | 1.400 |
| 52 | id_52 | Iris-versicolc | 6.400 | 3.200 | 4.500 | 1.500 |
| 53 | id_53 | Iris-versicolc | 6.900 | 3.100 | 4.900 | 1.500 |
| 54 | id_54 | Iris-versicolc | 5.500 | 2.300 | 4 | 1.300 |
| 55 | id_55 | Iris-versicolc | 6.500 | 2.800 | 4.600 | 1.500 |
| 56 | id_56 | Iris-versicolc | 5.700 | 2.800 | 4.500 | 1.300 |
| 57 | id_57 | Iris-versicolc | 6.300 | 3.300 | 4.700 | 1.600 |
| 58 | id_58 | Iris-versicolc | 4.900 | 2.400 | 3.300 | 1 |
| 59 | id_59 | Iris-versicolc | 6.600 | 2.900 | 4.600 | 1.300 |
| 60 | id_60 | Iris-versicolc | 5.200 | 2.700 | 3.900 | 1.400 |
| 61 | id_61 | Iris-versicolc | 5 | 2 | 3.500 | 1 |
| 62 | id_62 | Iris-versicolc | 5.900 | 3 | 4.200 | 1.500 |
| 63 | id_63 | Iris-versicolc | 6 | 2.200 | 4 | 1 |
| 64 | id_64 | Iris-versicolc | 6.100 | 2.900 | 4.700 | 1.400 |
| 65 | id_65 | Iris-versicolc | 5.600 | 2.900 | 3.600 | 1.300 |
| 66 | id_66 | Iris-versicolc | 6.700 | 3.100 | 4.400 | 1.400 |
| 67 | id_67 | Iris-versicolc | 5.600 | 3 | 4.500 | 1.500 |

Obrázek 4.1: Ukázka vstupních dat Iris

### Proces dolování konkrétních dat

Pro shlukování je vybrán data set *Iris*, který je popsán výše a je součástí RapidMiner. V levém sloupci v Repositories vybereme složku Samples → data → soubor *Iris* přetáhneme do Main Process viz. červenná šipka v Obrázku 4.2. Tím máme naimportované data pro modelování. Nyní je třeba zvolit jaký proces modelování chceme na data použít. Tato práce je zaměřená na shlukování tak vybereme jednu z metod pro shlukování a to základní metodu *k-means*. V levém sloupci v Operators zvolíme Modeling → Clustering and Segmentation → k-Means přetáhneme do Main process, modrá šipka v Obrázku 4.2. Pokud se nám operátory sami nespojí je třeba je spojit manuálně a to, že data Iris přivedeme na vstup k-Means viz. Obrázek 4.2. Kliknutím na operátor Clustering v Main Processu nastavujeme parametry shlukování a to hlavně do kolika $k$ shluků se má zařazovat a zvolení metody podobnosti objektů. V našem případě zvolíme za $k$ číslo 3, protože předpokládáme takový počet výsledných shluků viz. 4. Jeden výstup z Clustering navedeme na res (result) v pravé části Main Processu, druhý výstup nám bude sloužit na export dat. Pro náš implementovaný program je důležité data po shlukování vyexportovat z RapidMineru. Implementovaný program načítá .csv soubory, proto zvolíme v Operators Export → Data → Write CSV, jak je vyobrazuje tyrkysová šipka v Obrázku 4.2. Tomuto souboru musíme přiřadit cestu kam se má uložit a tu nastavíme při kliknutí na operátor Write CSV v Main Process v pravém sloupci parametrů a nastavíme oddělovací znak na ;. Abychom se mohli podívat na vy-

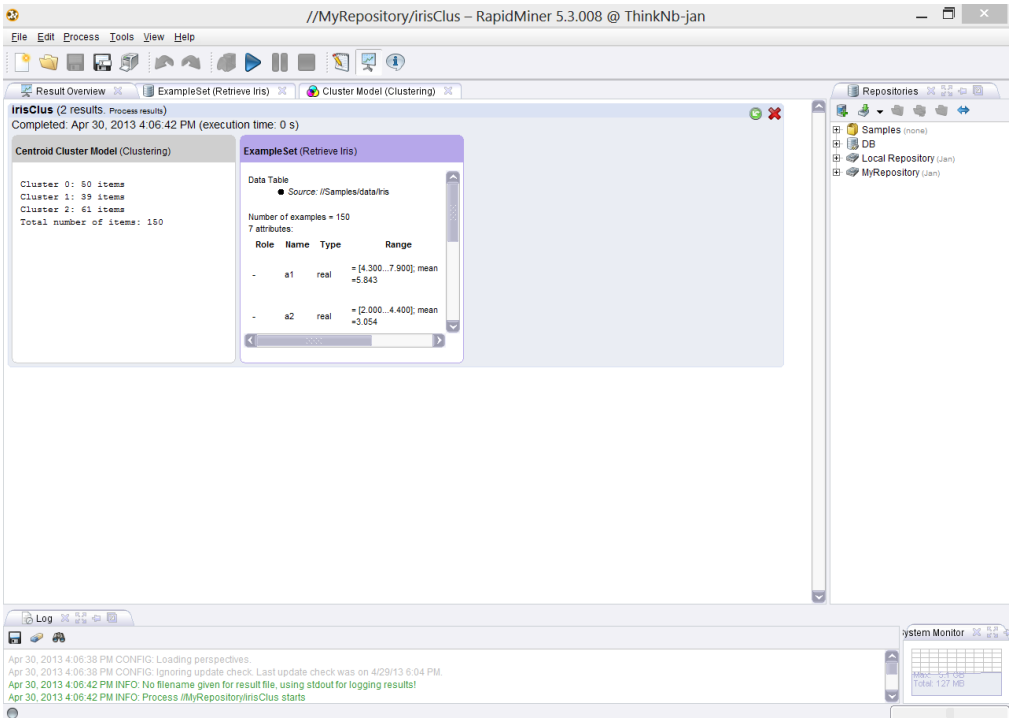
dolované data v RapidMineru, tak je třeba ještě také Write CSV napojit na res (result). Takhle sestavený proces nehlásí chyby tudíž jej můžeme spustit a to kliknutím na modrý trojúhelník nahoře v nabídce.




Obrázek 4.2: Dolování konkrétních dat

### Result perspektive

Po úspěšném dolování nás RapidMiner přepne do okna *result perspektive*, kde si můžeme prohlédnou výsledky, které jsou na Obrázku 4.3, kde vidíme vytvoření třech shluků z různými počty objektů, což znamená menší rozdíl oproti vstupním datům 4. Když se přepneme na druhou záložku *ExampleSet (Retrieve Iris)*, tak ta nám ukazuje do jakých shluků byly objekty rozděleny. Ukázka výsledných dat je zobrazena v Obrázku 4.4, která se nám i vyexportovala do nastaveného souboru. Ve třetí záložce *Cluster model (Clustering)* vidíme počty objektů ve shlucích, které objekty patří do jakého shluku, jaké hodnoty byly zvoleny pro centra jednotlivých atributů objektů, na základě kterých byly rozděleny do shluků a můžeme si tady vykreslovat různé grafy viz. Obrázek 4.5, kde jsou počty objektů ve shlucích a graficky znázorněné shluky, které čím blíž jsou sobě tím víc jsou si podobné.

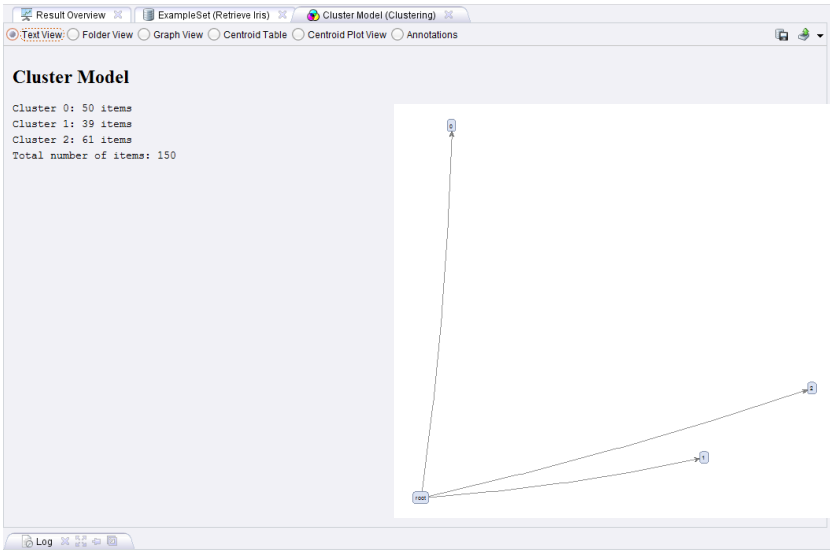



Obrázek 4.3: Přehled výsledků

ExampleSet (150 examples, 3 special attributes, 4 regular attributes)

| Row No. | id | label | cluster | a1 | a2 | a3 | a4 |
|---|---|---|---|---|---|---|---|
| 37 | id_37 | Iris-setosa | cluster_0 | 5.500 | 3.500 | 1.300 | 0.200 |
| 38 | id_38 | Iris-setosa | cluster_0 | 4.900 | 3.100 | 1.500 | 0.100 |
| 39 | id_39 | Iris-setosa | cluster_0 | 4.400 | 3 | 1.300 | 0.200 |
| 40 | id_40 | Iris-setosa | cluster_0 | 5.100 | 3.400 | 1.500 | 0.200 |
| 41 | id_41 | Iris-setosa | cluster_0 | 5 | 3.500 | 1.300 | 0.300 |
| 42 | id_42 | Iris-setosa | cluster_0 | 4.500 | 2.300 | 1.300 | 0.300 |
| 43 | id_43 | Iris-setosa | cluster_0 | 4.400 | 3.200 | 1.300 | 0.200 |
| 44 | id_44 | Iris-setosa | cluster_0 | 5 | 3.500 | 1.600 | 0.600 |
| 45 | id_45 | Iris-setosa | cluster_0 | 5.100 | 3.800 | 1.900 | 0.400 |
| 46 | id_46 | Iris-setosa | cluster_0 | 4.800 | 3 | 1.400 | 0.300 |
| 47 | id_47 | Iris-setosa | cluster_0 | 5.100 | 3.800 | 1.600 | 0.200 |
| 48 | id_48 | Iris-setosa | cluster_0 | 4.600 | 3.200 | 1.400 | 0.200 |
| 49 | id_49 | Iris-setosa | cluster_0 | 5.300 | 3.700 | 1.500 | 0.200 |
| 50 | id_50 | Iris-setosa | cluster_0 | 5 | 3.300 | 1.400 | 0.200 |
| 51 | id_51 | Iris-versicolc | cluster_1 | 7 | 3.200 | 4.700 | 1.400 |
| 52 | id_52 | Iris-versicolc | cluster_2 | 6.400 | 3.200 | 4.500 | 1.500 |
| 53 | id_53 | Iris-versicolc | cluster_1 | 6.900 | 3.100 | 4.900 | 1.500 |
| 54 | id_54 | Iris-versicolc | cluster_2 | 5.500 | 2.300 | 4 | 1.300 |
| 55 | id_55 | Iris-versicolc | cluster_2 | 6.500 | 2.800 | 4.600 | 1.500 |
| 56 | id_56 | Iris-versicolc | cluster_2 | 5.700 | 2.800 | 4.500 | 1.300 |
| 57 | id_57 | Iris-versicolc | cluster_2 | 6.300 | 3.300 | 4.700 | 1.600 |
| 58 | id_58 | Iris-versicolc | cluster_2 | 4.900 | 2.400 | 3.300 | 1 |
| 59 | id_59 | Iris-versicolc | cluster_2 | 6.600 | 2.900 | 4.600 | 1.300 |
| 60 | id_60 | Iris-versicolc | cluster_2 | 5.200 | 2.700 | 3.900 | 1.400 |
| 61 | id_61 | Iris-versicolc | cluster_2 | 5 | 2 | 3.500 | 1 |
| 62 | id_62 | Iris-versicolc | cluster_2 | 5.900 | 3 | 4.200 | 1.500 |
| 63 | id_63 | Iris-versicolc | cluster_2 | 6 | 2.200 | 4 | 1 |
| 64 | id_64 | Iris-versicolc | cluster_2 | 6.100 | 2.900 | 4.700 | 1.400 |
| 65 | id_65 | Iris-versicolc | cluster_2 | 5.600 | 2.900 | 3.600 | 1.300 |
| 66 | id_66 | Iris-versicolc | cluster_2 | 6.700 | 3.100 | 4.400 | 1.400 |
| 67 | id_67 | Iris-versicolc | cluster_2 | 5.600 | 3 | 4.500 | 1.500 |

Obrázek 4.4: Ukázka výstupních dat




Obrázek 4.5: Rozdělení do shluků a graf rozdělení na základě podobnosti shluků

# Kapitola 5

# Implementovaný program

Kapitola popisuje návrh a implementaci programu pro měření úspěšnosti shlukování.

## 5.1 Návrh aplikace

Aplikace je realizována jako konzolová, podporuje tedy pouze textový režim. Program vyžaduje zadání tří povinných parametrů. Prvním povinným parametrem (-p) je jméno souboru po klasifikaci. Druhým povinným parametrem (-c) je jméno souboru po shlukování. Třetím parametrem (-m) se volí metoda měření úspěšnosti - [R] pro Randův index nebo [M] pro míru nesouladu dle konfuzní matice. Program nejprve načte oba soubory. První soubor je výstupní soubor po klasifikaci (v dalším textu jen soubor [P]). Druhý soubor je výstupní soubor po shlukování (v dalším textu soubor [C]). Oba soubory jsou ve formátu .csv. Jako delimitační znak je použit znak „;“ (středník). Pro měření úspěšnosti jsou v aplikaci implementovány dvě metody. První metoda je výpočet Randova indexu - parametr [R], druhá metoda je výpočet míry nesouladu založený na konfuzní matici - parametr [M]. Po načtení souborů [P] a [C] je dle zvolené metody proveden výpočet. Výpis výsledku výpočtu je realizován na standardní výstup (stdout).

## 5.2 Popis aplikace

Program *measure* provádí měření úspěšnosti. Výsledek výpočtu je realizován na standardní výstup (stdout). Aplikace je vypracována ve vyšším programovacím jazyce C. Aplikace je rozdělena do čtyř modulů.

### 5.2.1 Modul main

*main.c* - hlavní program
Hlavní program nejprve načte parametry příkazového řádku. Pro načtení parametrů z příkazového řádku je použita funkce *getopt*. Po načtení parametrů jsou načtena data ze souborů [P] a [C]. Poté je proveden výpočet dle zvolené metody.

### 5.2.2 Modul readobj

*readobj.c* - zdrojový kód pro načtení souborů [P] a [C] do dynamicky alokovaných struktur
*readobj.h* - hlavičkový soubor, obsahuje definici konstant, datových struktur a deklarace funkcí

Modul implementuje funkce pro načtení vstupních hodnot ze souboru typu .csv. Data jsou načtena do dynamicky alokovaných struktur *structP* a *structC*. Po načtení dat je provedena kontrola počtu objektů v obou strukturách. Počet objektů v obou strukturách musí být stejný. Následně je provedena kontrola konzistence dat v obou strukturách - objekty ve shlucích po klasifikaci, které jsou v obou strukturách (id a label) musí mít stejný obsah.

### 5.2.3 Modul measuremod

*measuremod.c* - zdrojový kód pro výpočet míry nesouladu založený na konfuzní matici
*measuremod.h* - hlavičkový soubor, obsahuje definici konstant, datových struktur a deklarace funkcí
Modul implementuje funkce pro výpočet míry nesouladu založený na konfuzní matici - parametr [M]. Výpočet míry nesouladu viz. 3.3.2. Z důvodu potřeby vyhodnocení a přesunu velkých objemů dat neprovádí implementovaný algoritmus pro sestavení konfuzní matice postupné uspořádání shluků dle požadavku míry nesouladu. Místo toho sestaví konfuzní matici ze shluků postupně, jak jsou uvedeny ve strukturách P a C. Vlastní uspořádání počtů objektů tak, aby potřebná data byla na diagonále matice, je následně provedeno uspořádáním konfuzní matice. Konfuzní matice je uspořádána tak, že řádky odpovídají struktuře P a sloupce struktuře C. Nejprve jsou setříděny sloupce. Na první sloupec je přemístěn sloupec s nejvyšší hodnotou shodných objektů (bez ohledu na kterém řádku se tato hodnota vyskytuje). Na druhý sloupec je přemístěn sloupec s druhou nejvyšší hodnotou shodných objektů. Takto jsou setříděny všechny ostatní sloupce.

### 5.2.4 Modul measureri

*measureri.c* - zdrojový kód pro výpočet Randova indexu
*measureri.h* - hlavičkový soubor, obsahuje definici konstant, datových struktur a deklarace funkcí
Modul implementuje funkci pro výpočet Randova indexu, který měří úspěšnost (podobnost struktur P a C) pomocí kriteria založeného na míře podobnosti - parametr [R]. Výpočet míry nesouladu viz. 3.3.1.

## 5.3 Validace Randova Index

Ověření zda je správně naimplementovaná metoda *Randův index*. Mějme vstupní strukturu v následujícím tvaru:
**Předpokládaná struktura P**

- **0.shluk** - objekty s id (1, 2, 3, 4, 5),
- **1.shluk** - objekty s id (6, 7, 8, 9, 10),
- **2.shluk** - objekty s id (11, 12, 13, 14, 15),

kde ve struktuře máme 3 shluky po 5 objektech. Vstupní struktura je pouze ukázková (vymyšlená), ale dostačuje k ověření implementace. Na tuto strukturu je použit nějaký shlukovací algoritmus a jeho výstup a rozdělení objektů do shluků je následující:
**Získaná struktura C**

- **0.shluk** - objekty s id (1, 2, 3, 4, 5),

- **1.shluk** - objekty s id (6, 7, 8, 9, 10),
- **2.shluk** - objekty s id (11, 12, 13, 14, 15).

*Randův index* počítá s počty párů objektů ve shlucích viz. 3.3.1. Tedy máme celkem 15 objektů s různým zařazením do shluků. Počet párů je množství kombinací *k-té* třídy z $n$ prvků bez opakování. V našem případě se jedná o kombinaci 2-hé třídy ze všech objektů $n$. Příslušný počet kombinací vypočteme za vzorce (n nad k), v našem případě (n nad 2) což je:

$$\text{počet párů} = \frac{n!}{(n-k)! * k!} = \frac{n!}{(n-2)! * 2!} = \frac{n * (n-1)}{2}, \tag{5.1}$$

kde pro naše čísla rovnice vypadá následovně

$$\text{počet párů} = \frac{15 * (15-1)}{2} = 105, \tag{5.2}$$

tedy počet všech možných párů $M = a + b + c + c = 105$. Následně dopočítáme ostatní symboly, které jsou definovány v 3.3.1.

- $a$ - (1,2)(1,3)(1,4)(1,5)(2,3)(2,4)(2,5)(3,4)(3,5)(4,5)(6,7)(11,12)(11,15)(12,15)(8,9)(8,10)(9,10)(13,14) = 18 párů,
- $b$ - (6,11)(6,12)(6,15)(7,11)(7,12)(7,15)(8,13)(8,14)(9,13)(9,14)(10,13)(10,14) = 12 párů,
- $c$ - (6,8)(6,9)(6,10)(7,8)(7,9)(7,10)(11,13)(11,14)(12,13)(12,14)(13,15)(14,15) = 12 párů,
- $d$ - z najdeme zbytek párů d = M - a + b + c = 105 - 18 + 12 + 12 = 63 párů,
- $m_1$ - a + b = 18 + 12 = 30 párů,
- $m_2$ - a + c = 18 + 12 = 30 párů.

Po vypočítání všech symbolů dosadíme do rovnice 3.14:

$$MD = \frac{a+d}{M} = \frac{18+63}{105} = 0,77143. \tag{5.3}$$

**Výstup programu:**

*a: 18 b: 12 c: 12 d: 63*
*m1: 30 m2: 30 m: 105*
*Randuv index: 0.77143*

Jak je vidět tak výstupní hodnoty z programu (*Measure RI (Randuv Index): a = 18, b = 12, c = 12, d = 63, m1 = 30, m2 = 30, m = 105, Randuv index = 0.77143*) souhlasí s hodnotami vypočítanými matematicky, tímto je ověřena správnost metody *Randův index*. Ověření metody *Míra nesouladu* je zobrazeno v kapitole 6.

# Kapitola 6

# Výsledky

V této kapitole jsou zobrazeny výsledky měření úspěšnosti dolování dat v oblasti shlukování. Na data data set *Iris* je použito několik shlukovacích algoritmů v prostředí RapidMiner, kterými jsou:

- *k-means*
- *k-medoids*
- *x-means*
- *DBSCAN*
- *Expectation-Maximization*
- *random clustering*

Předpokládaná struktura **P** pro porovnání úspěšnosti shlukování pro všechny aplikovaní algoritmy je v následujícím tvaru:

- **0.shluk**: 50 objektů typu *iris-setosa*,
- **1.shluk**: 50 objektů typu *iris-versicolor*,
- **2.shluk**: 50 objektů typu *iris-virginica*.

Pro všechny algoritmy je zjišťovaná úspěšnost shlukování pomocí *kritérii založených na mírách podobností viz. 3.3.1*, kde je implementován *Randův index*. Pro některé algoritmy je ještě zjišťovaná úspěšnost shlukování pomocí *kritérii založených na konfuzní matici viz. 3.3.2*, kde je implementována *Míra nesouladu*. *Míra nesouladu* nemůže být použita na všechny algoritmy, protože podmínkou měření úspěchu je stejný počet shluků jak v předpokládané struktuře tak ve vzniklé struktuře po použití shlukovacího algoritmu.

## 6.1 Výsledky shlukování metodou k-means

Rozmístění objektů do shluků a vytvoření struktury **C** viz. 3.3 je následující:

- **0.shluk**: 50 objektů typu *iris-setosa*,

- **1.shluk**: 39 objektů z toho 3 objety typu *iris-versicolor* a 36 objektů typu *iris-virginica*,
- **2.shluk**: 61 objektů z toho 47 objektů typu *iris-versicolor* a 14 objektů typu *iris-virginica*.

Z vytvořené struktury **C** a předpokládané struktury **P** můžeme jednoduše dopočítat úspěšnost shlukování pomocí konfuzní matice 3.3.2. Sestavíme konfuzní matici:

| | | $P$ | | |
|---|---|---|---|---|
| | $n_{hh}$ | $0.shluk$ | $2.shluk$ | $1.shluk$ |
| $C$ | $0.shluk$ | 50 | 0 | 0 |
| | $1.shluk$ | 0 | 47 | 3 |
| | $2.shluk$ | 0 | 14 | 36 |

Na diagonále se nachází počet společných objektů ve shlucích, které dosadíme do rovnice *Míry nesouladu* 6.4:

$$MD = \frac{150 - (50 + 47 + 36)}{150} = 0.11\overline{3} \tag{6.1}$$

**Výstup programu pro Míru nesouladu:**

*Konfuzni matice:*
*Prvek 50; Prvek 0; Prvek 0;*
*Prvek 0; Prvek 47; Prvek 3;*
*Prvek 0; Prvek 14; Prvek 36;*
*Suma Nhh 133*
*Mira nesouladu dle konfuzni matice: 0.11333*

**Výstup programu pro Randův index:**

*a: 3030 b: 766 c: 645 d: 6734*
*m1: 3796 m2: 3675 m: 11175*
*Randuv index: 0.87374*

Z rovnice plyne nízká míra nesouladu $= 0.11\overline{3}$ a vyšší podobnost struktur dle *Randova indexu* $= 0.87374$, což vede k závěru, že zvolená shlukovací metoda je použitelná. Zároveň jde zde ověřena funkčnost metody *Míra nesouladu*, kdy matematicky vypočítané výsledky sedí s výsledky programovými.

## 6.2 Výsledky shlukování metodou k-medoids

Rozmístění do shluků a vytvoření struktury **C** je následující:

- **0.shluk**: 85 objektů z toho 45 objektů typu *iris-versicolor* a 50 objektů typu *iris-virginica*,
- **1.shluk**: 50 objektů typu *iris-setosa*,
- **2.shluk**: 15 objektů typu *iris-vesicolor*.

Sestavíme konfuzní matici:

| | | $P$ | | |
|---|---|---|---|---|
| | $n_{hh}$ | $2.shluk$ | $0.shluk$ | $1.shluk$ |
| $C$ | $0.shluk$ | 50 | 0 | 0 |
| | $1.shluk$ | 0 | 50 | 0 |
| | $2.shluk$ | 0 | 35 | 15 |

Dosadíme do rovnice *Míry nesouladu* 6.4:

$$MD = \frac{150 - (50 + 50 + 15)}{150} = 0.2\overline{3} \tag{6.2}$$

**Výstup programu pro Míru nesouladu:**

*Konfuzni matice:*
*Prvek 50; Prvek 0; Prvek 0;*
*Prvek 0; Prvek 50; Prvek 0;*
*Prvek 0; Prvek 35; Prvek 15;*
*Suma Nhh 115*
*Mira nesouladu dle konfuzni matice: 0.23333*

**Výstup programu pro Randův index:**

*a: 3150 b: 1750 c: 525 d: 5750*
*m1: 4900 m2: 3675 m: 11175*
*Randuv index: 0.79642*

Výsledek je trochu horší než u k-means, Míra nesouladu = 0.23333 a Randův index = 0.79642, ale i tak je metoda celkem úspěšná.

## 6.3 Výsledky shlukování metodou x-means

Rozmístění do shluků a vytvoření struktury **C** je následující:

- **0.shluk**: 27 objektů typu *iris-setosa*,
- **1.shluk**: 23 objektů typu *iris-setosa*,
- **2.shluk**: 39 objektů z toho 26 objektů typu *iris-versicolor* a 13 objektů typu *iris-virginica*,
- **3.shluk**: 25 objektů z toho 24 objektů typu *iris-versicolor* a 1 objekt typu *iris-virginica*,
- **4.shluk**: 12 objektů typu *iris-virginica*,
- **5.shluk**: 24 objektů typu *iris-virginica*.

Zde nesestavíme konfuzní matici, protože není splněn předpoklad stejného počtu shluků v obou strukturách.

**Výstup programu pro Randův index:**

*a: 1625 b: 362 c: 2050 d: 7138*

*m1: 1987 m2: 3675 m: 11175*
*Randuv index: 0.78416*

Metoda už se pohybuje na hranici optimálního úspěchu. Randův index = 0.78416.

## 6.4 Výsledky shlukování DBSCAN

Rozmístění do shluků a vytvoření struktury **C** je následující:

- **0.shluk**: 0 objektů,
- **1.shluk**: 50 objektů typu *iris-setosa*,
- **2.shluk**: 100 objektů z toho 50 objektů typu *iris-versicolor* a 50 objektů typu *iris-virginica*.

Při vygenerování .csv souboru jsou vytvořeny jen dva shluky, tudíž nemůžeme vytvořit konfuzní matici.
**Výstup programu pro Randův index:**
*a: 3675 b: 2500 c: 0 d: 5000*
*m1: 6175 m2: 3675 m: 11175*
*Randuv index: 0.77629*

Metoda už se pomalu vzdaluje od optimálního úspěchu. Randův index = 0.78416.

## 6.5 Výsledky shlukování metodou Expectation-Maximization

Rozmístění do shluků a vytvoření struktury **C** je následující:

- **0.shluk**: 50 objektů typu *iris-setosa*,
- **1.shluk**: 55 objektů z toho 5 objektů typu *iris-versicolor* a 50 objektů typu *iris-virginica*,
- **2.shluk**: 45 objektů typu *iris-vesicolor*.

Sestavíme konfuzní matici:

| | | $P$ | | |
|---|---|---|---|---|
| | $n_{hh}$ | $0.shluk$ | $2.shluk$ | $1.shluk$ |
| $C$ | $0.shluk$ | 50 | 0 | 0 |
| | $1.shluk$ | 0 | 50 | 0 |
| | $2.shluk$ | 0 | 5 | 45 |

Dosadíme do rovnice *Míry nesouladu* 6.4:

$$MD = \frac{150 - (50 + 50 + 45)}{150} = 0.0\overline{3} \tag{6.3}$$

**Výstup programu pro Míru nesouladu:**
*Konfuzni matice:*

*Prvek 50; Prvek 0; Prvek 0;*
*Prvek 0; Prvek 50; Prvek 0;*
*Prvek 0; Prvek 5; Prvek 45;*
*Suma Nhh 145*
*Mira nesouladu dle konfuzni matice: 0.03333*

**Výstup programu pro Randův index:**
*a: 3450 b: 250 c: 225 d: 7250*
*m1: 3700 m2: 3675 m: 11175*
*Randuv index: 0.95749*

Metoda dosahuje největší úspěšnosti tzn. struktura po aplikaci shlukovacího algoritmu *Expectation-Maximization* je nejpodobnější s předpokládanou strukturou. Míra nesouladu = 0.03333 a Randův index = 0.95749.

## 6.6 Výsledky shlukování metodou random clustering

Rozmístění do shluků a vytvoření struktury **C** je následující:

- **0.shluk**: 46 objektů z toho 10 objektů *iris-setosa*, 17 objektů typu *iris-versicolor* a 19 objektů typu *iris-virginica*,
- **1.shluk**: 55 objektů z toho 21 objektů *iris-setosa*, 16 objektů typu *iris-versicolor* a 18 objektů typu *iris-virginica*,
- **2.shluk**: 49 objektů z toho 19 objektů *iris-setosa*, 17 objektů typu *iris-versicolor* a 13 objektů typu *iris-virginica*.

Sestavíme konfuzní matici:

| | | $P$ | | |
|---|---|---|---|---|
| | $n_{hh}$ | $0.shluk$ | $2.shluk$ | $1.shluk$ |
| $C$ | $1.shluk$ | 21 | 18 | 0 |
| | $0.shluk$ | 10 | 19 | 17 |
| | $2.shluk$ | 0 | 5 | 17 |

Dosadíme do rovnice *Míry nesouladu* 6.4:

$$MD = \frac{150 - (21 + 19 + 17)}{150} = 0.62 \tag{6.4}$$

**Výstup programu pro Míru nesouladu:**
*Konfuzni matice:*
*Prvek 21; Prvek 10; Prvek 19;*
*Prvek 18; Prvek 19; Prvek 13;*
*Prvek 16; Prvek 17; Prvek 17;*
*Suma Nhh 57*
*Mira nesouladu dle konfuzni matice: 0.62000*

**Výstup programu pro Randův index:**

*a: 1220 b: 2476 c: 2455 d: 5024*
*m1: 3696 m2: 3675 m: 11175*
*Randuv index: 0.55875*

Metoda dosahuje nejhorších výsledků ze všech metod, protože jde o úplně náhodné rozdělení do shluků. Míra nesouladu = 0.62000 a Randův index = 0.55875.

## 6.7 Vyhodnocení úspěšnosti shlukovacích metod

Použité shlukovací metody jsou docela úspěšné při rozdělování objektů do shluků. Nejpřesnějšího výsledku dosahuje metoda *Expectation-Maximization*, která při *Míře nesouladu* zařadila špatně jen „5“ objektů a *Randův index* se přibližoval k velké shodě s číslem 1, což značí velký souhlas mezi předpokládanou a získanou strukturou. Druhou nejúspěšnější metodou je metoda *k-means*, která dosahuje taky uspokojivých výsledků. Metody *DBSCAN, k-medoid a x-means* dosahují poměrně stejných výsledků, u kterých by se dalo říct, že jsou ještě použitelné. Nejhoršího výsledku vůbec dosahuje metoda *random clustering*, které pokud můžeme se raději vyhněme.

## 6.8 Vyhodnocení metod měřících úspěšnost

Pro měření úspěšnosti jsou implementovány dvě metody *Míra nesouladu* a *Randův index*. Jak pracují již bylo vysvětleno v kapitole 3.3. *Míra nesouladu* jde lehce matematicky dopočítat a ve většině případů dosahuje větší shody struktur než metoda *Randův index*. *Randův index* porovnává páry objektů, což může vést k přesnějším výsledkům. Navíc metodu *Míry nesouladu* můžeme vyhodnocovat jen tehdy pokud je počet shluků ve strukturách stejný, *Randův index* můžeme použít vždy.

# Kapitola 7

# Závěr

Cílem této práce je implementace programu pro měření úspěšnosti dolování dat v oblasti shlukování a odzkoušení funkčnosti na vhodném vzorku dat. Práce seznamuje se základními pojmy a vlastnostmi dolování dat a hlouběji se shlukovou analýzou. Praktická část je rozdělena na dvě části. První část je zaměřena na dolování dat, kde na data je použito několik shlukovacích algoritmů v prostředí RapidMiner. Výsledky po shlukování jsou z programu RapidMiner vyexportovány ve formátu *.csv*. Druhá část je zaměřena na implementaci programu a zhodnocení dosažených výsledků. Vyexportované data jsou navedeny na vstup implementovaného programu, který implementuje dvě metody pro měření úspěšnosti shlukovacích algoritmů *Randův index* a *Miru nesouladu*. Obě metody nabývají hodnot v rozmezí od 0 do 1, kde pro *Míru nesouladu* platí čím je hodnota bližší 0 tím jsou si struktury podobnější, u *Randova indexu* je to naopak. Výsledky použitých shlukovacích algoritmů jsou uvedeny v kapitole 6. Algoritmy vykazovaly dobrou úspěšnost, ale z velké části díky kvalitním datům. Cílem práce tedy bylo implementovat program měřící podobnost předpokládané (P) a získané (C) struktury. Toto je i největší přínos práce, kdy můžeme změřit úspěšnost shlukovacích algoritmů (metod) a jejich rozřazování objektů do shluků. Pokračování práce by mohlo vést k implementaci dalších metod pro měření úspěšnosti nebo převod programu do grafického (uživatelsky příjemnějšího) režimu.

# Literatura


[1] C. R. Rao, E. J. Wegman a J. L. Solka: *Handbook of Statistics 24: Data Mining and Data Visualization*. Amsterdam: Elsevier, 2005, ISBN 0-444-51141-5, 800 s.

[2] H. Řezanková, D. Húsek a V. Snášel: *Shluková analýza dat*. Praha: Professional Publishing, druhé vydání, 2009, ISBN 978-80-86946-81-8, 218 s.

[3] J. Han a M. Kamber: *Data mining*. San Francisco: Morgan Kaufmann Publishers, druhé vydání, 2006, ISBN 1-55860-901-6, 770 s.

[4] J. Zendulka, V. Bartík, R. Lukáš a I. Rudolfová: *Studijní opora: Získávání znalostí z databází*. FIT VUT v Brně, 2009.
URL https://wis.fit.vutbr.cz/FIT/st/course-filesst.php/course/ZZN-IT/texts/ZZN.pdf

[5] M. Bramer: *Principles of data mining*. London: Springer, 2007, ISBN 1-84628-765-0, 343 s.

[6] M. Dunham: *Data Mining: Introductory and Advanced Topics*. New Jersey: Prentice-Hall, 2003, ISBN 0-13-088892-3, 315 s.

[7] P. Berka: *Dobývání znalostí z databází*. Praha: Academia, 2003, ISBN 80-200-1062-9, 366 s.

[8] T. Löster: *Hodnocení výsledků metod shlukové analýzy*. Disertační práce, VŠE v Praze - Fakulta informatiky a statistiky, 2011.

[9] U. Fayyad, G. Piatetsky-Shapiro a P. Smyth: From data mining to knowledge discovery in databases. *American Association of Artificial Intelligence*, ročník 17, č. 3, 1996.
URL http://www.aaai.org/ojs/index.php/aimagazine/article/view/1230/1131

# Příloha A

# Obsah CD

| | |
|---|---|
| \src | zdrojové soubory programu se vstupními daty |
| \doc | zdrojové soubory písemné zprávy |
| \bin | spustitelný soubor |
| \programy | obsahuje instalace volitelných programů RapidMiner, Code::Blocks |
| manual.pdf | manuál ke spuštění programu |
| BPxtrunk00.pdf | písemná zpráva |
| readme.txt | nápověda k obsahu CD |