VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KLASIFIKACE WEBOVÝCH STRÁNEK

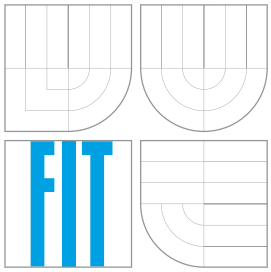
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ROMAN KOLÁŘ

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KLASIFIKACE WEBOVÝCH STRÁNEK

WEB PAGE CLASSIFICATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ROMAN KOLÁŘ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR BARTÍK, PhD.

BRNO 2008

Abstrakt

Práce se zabývá problematikou automatické klasifikace webových stránek s využitím asociačního klasifikátoru. Je představena klasifikace, jakožto jeden z oborů dolování znalostí z databází; zvláštní prostor je věnován klasifikaci textových dat. Jsou diskutovány různé metody klasifikace textových dokumentů se zdůrazněním výhod klasifikátorů využívajících pro rozhodování asociační pravidla. Cílem práce je pokusit se přizpůsobit vybranou klasifikační metodu pro relační data a navrhnout systém pro klasifikaci webových stránek podle vizuálních vlastností - rozložení jednotlivých oblastí na stránce, nikoliv podle čistého textového obsahu. K tomu je využitý asociační klasifikátor ARC-BC kombinující výhody známých klasifikačních metod.

Klíčová slova

klasifikace, klasifikátor, Web, dolování znalostí, asociační pravidlo, přesnost, data, diskretizace, kategorie, struktura, atribut, podpora, spolehlivost, text, interval

Abstract

This paper presents problem of automatic webpages classification using association rules based classifier. Classification problem is presented, as a one of datamining technique, in context of mining knowledges from text data. There are many text document classification methods presented with highlighting benefits of classification methods using association rules. The main goal of work is adjusting selected classification method for relation data and design draft of webpages classifier, which classifies pages with the aid of visual properties - independent section layout on the web page, not (only) by textual data. There is also ARC-BC classification method presented as a selected method and as one of intriguing classificators, that derives accuracy and understandableness benefits of all other methods.

Keywords

classification, classificator, Web, datamining, association rule, precission, data, discretization, category, structure, attribute, support, confidence, text, interval

Citace

Roman Kolář: Klasifikace webových stránek, diplomová práce, Brno, FIT VUT v Brně, 2008

Klasifikace webových stránek

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně.

.....
Roman Kolář
15. května 2008

Obsah

Kapitola 1

Úvod

Množství informací obsažených na Webu je obrovské a každým rokem se mohutně zvětšuje. Již v roce 2005 vyhledávač Google indexoval více než 8 miliard webových stránek a toto číslo se prudkým tempem zvyšuje. Čím více prostor dokumentů uložených na Webu roste, tím více roste také potřeba získávat z těchto dat specifické informace. *Data mining* poskytuje řadu technik použitelných pro vyhledání zajímavých vzorů v datech a pro vyjádření těchto vzorů jako smysluplných informací pro koncové uživatele. Jednou z technik dolování znalostí z dat také je klasifikace, která hraje důležitou roli v mnoha oblastech řízení a sběru informací.

Klasifikace webových stránek má za cíl automatizované rozpoznávání tématu, kterému se stránka věnuje, a může být například využita pro kategorizaci stránek do určitých tříd, či upřesnění vyhledávacích dotazů.

Významnou vlastností webových stránek je jejich semistrukturovanost. Holá textová část představuje nestrukturovanou část stránky, HTML značky a jejich obsah potom strukturovanou. Tato vlastnost webových stránek indikuje použití odlišných technik pro klasifikaci (a dolování dat obecně), než je tomu u obyčejných textových dokumentů, nebo plně strukturovaných relačních a transakčních dat.

Prozatím se většina výzkumů klasifikace webových stránek věnuje především klasifikaci podle textového, případně multimediálního obsahu, nebo podle struktury celého webu.

Tato práce se věnuje klasifikaci webových stránek jakožto procesu komplexního ohodnocení webové stránky podle více kriterií, než je jen pouhá analýza textového obsahem. Zaměřuje se přitom zejména na vizální vzhled, tedy rozmístění významných, logicky samostatných, oblastí na stránce.

V dřívějších výzkumech a pracích byla navržena řada klasifikačních metod, jako jsou rozhodovací stromy[18, 5], Bayesovské klasifikační metody[6, 23, 17], na pravidlech založená klasifikace [21, 14, 1, 13], či různé statistické přístupy. Tyto většinou vycházejí z přístupů používaných v jiných oblastech dolování znalostí a vedou získání reprezentativního vzorku jistých znalostí (pravidel) z trénovací množiny, které jsou následně použité pro klasifikaci nových dat. Jednou z nových metod je také metoda ARC-BC využívající asociační pravidla a dosahující při její nativní úloze klasifikace textových dat velice dobrých výsledků.

V následujících kapitolách bude představena problematika dolování znalostí z dokumentů se zaměřením na klasifikaci, samostatná část bude vyhrazena pro klasifikaci webových dokumentů s využitím asociačních pravidel(Kapitola 3). Vysvětlením základních pojmů klasifikace a asociačních pravidel a jejich zařazení do dolování znalostí jako celku se věnuje Kapitola 2. V Kapitole 3 bude mj. popsán a vysvětlen princip klasifikační metody ARC-BC a vyzdviženy výhody oproti ostatním klasifikačním metodám. Kapitola 5 pak představuje

hrubý návrh hypotetického klasifikačního systému webových stránek založeného na metodě ARC-BC a pracujícího s daty získanými analýzou vizuálních vlastnostostí stránky.

V implementační části práce bude kromě popisu základních tříd a balíčku programu prezentován postup při přetváření metody ARC-BC pro relační data. Závěrečná kapitola shrnuje práci jako celek, vysvětluje výsledky dosažené při experimentálních klasifikacích a otevírá diskusi pro možná navazující rozšíření a vylepšení.

Kapitola 2

Dolování znalostí z dat

Dolování znalostí z dat chápeme jako extrakci zajímavých vzorů z dat, které jsou předem neznámé, skryté a potenciálně užitečné. Data k extrakci mají ve většině případů velký objem, často se dolování provádí nad rozsáhlými datovými sklady, či produkčními databázemi.

Významnou vlastností vzorů, které z dat pomocí dolování získáváme, je fakt, že před počátkem dolování jsou v datech skryté, na první pohled nezjistitelné. Dolování pak představuje mocný nástroj k jejich získání, přestože se na celou řadu dolovacích technik může pohlížet jen jako na "pouhé" inteligentní statistické metody.

Dolování se uplatňuje všude tam, kde dochází k rozsáhlému sběru dat a je potřeba tato data jistým způsobem analyzovat. Typickými datovými zdroji pro dolování jsou finanční data, obchodní data, či data telekomunikačních společností. Za jeden z důležitých úkolů dolování je analýza nákupního košíku, která zkoumá nákupní zvyky zákazníků.

2.1 Dolování znalostí z textových dat

Dolování znalostí z textových dat (TM - text mining) je jednou z úloh dolování znalostí z dat. Požadavek na dolování textových dat souvisí mj. s obrovským rozmachem elektronicky uložených dokumentů - emailových zpráv, vědeckých článků, elektronických knih a webových stránek. Zatímco ostatní odvětví data miningu se zaměřují především na strukturovaná data jako jsou data relační, transakční, či data v datových skladech, TM se snaží získat informace nestrukturovaných, kde klasické metody DM selhávají a kde jsou vyžadované speciální metody a algoritmy.

V současné době existuje k text miningu mnoho různých přístupů a metod. Obecně dělíme přístupy TM podle vstupních dat, které zadáváme TM systému na: (1) **přístup založený na klíčových slovech**, kde jsou vstupnímy daty klíčová slova dokumentu, (2) **přístup založený na značkách**, kde je vstupem jistá množina značek (tags), a (3) **přístup založený na extrakci informací**, vstupem jsou zde sémantické informace, jako např. události, fakta. Přístup založený na extrakci informací je oproti ostatním dvěma progresivnější a může vést k nalezení významnějších znalostí v datech, ale vyžaduje sémantickou analýzu textového dokumentu. Z těchto tří základních přístupů časem vznikalo čím dál více úloh pro dolování textových dat jako jsou klasifikace dokumentů, asociační analýza, extrakce informací, různé druhy asociačních analýz, které se dnes běžně využívají při řešení běžných problémů (filtrování spamu, vyhledávání stránek ve webových vyhledávačích aj.).

Asociační analýza klíčových slov
Asociační analýza klíčových slov je analýza dokumentů snažící se nalézt množiny klíčových slov, či výrazů, které se vyskytují v textu ve větší frekvenci, než ostatní slova. Jako řada dalších analýz prováděných v dokumentu vyžaduje asociační analýza data vhodným způsobem předzpracovaná, např. jsou nalezeny kořeny všech slov a odstraněné spojky, předložky a další irelevantní slova (tzv. stop words). Po provedení asociační analýzy klíčových slov získáme kolekci záznamů {*documentId, setOfKeywords*}, která ke každému dokumentu označenému identifikátorem *documentId* přiřazuje množinu klíčových slov *setOfKeywords*. V souvislosti s klíčovými slovy proběhla řada výzkumů - např. využití klíčových slov pro dolování znalostí[8].

Klasifikace dokumentů
Klasifikace dokumentů je důležitou úlohou dolování znalostí. Existence velkého počtu online dokumentů vyžaduje automatickou organizaci dokumentů do kategorií podle daných kriterií. Kategorie, do kterých chceme dokument přiřadit, musí být předem známé (např. mějme třídy Automobilismus, Přírodní tématika, Vědecká zpráva, a dokument neznámého obsahu, jenž chceme klasifikovat). Klasifikace dokumentů se používá v mnoha aplikacích a existuje pro ni mnoho metod a algoritmů [15, 10]. Podrobněji se klasifikaci dokumentů věnuje část 2.1.

Shluková analýza
Shluková analýza dokumentů je významná pro organizaci dokumentů, o kterých nemáme žádné informace. Oproti klasifikaci se liší tím, že nejsou předem známé žádné třídy, do kterých by se dokumenty rozřazovaly. V průběhu analýzy se hledají v dokumentech zajímavé shluky (obdoba tříd u klasifikace) dat, které reprezentují množinu společných vlastností dokumentů.

Dolování znalostí z Webu
Dolování znalostí z webových stránek se snaží získat informace uložených ve webových stránkách. Cílem je stejně jako u běžných nestrukturovaných dokumentů získat důležité informace o obsahu webové dokumentu - např. jeho klasifikace do tříd, nalezení klíčových slov aj., přičemž se nehledí na strukturu vybraného webu, pouze na obsah. V tomto směru lze na webovou stránku pohlížet jako na textový dokument [7, 19] s případným rozšířením o multimediální prvky - obrázky.

Dolování znalostí ze struktury webového dokumentu
Dolování znalostí ze struktury webového dokumentu se také zaměřuje na webové stránky, ale snaží se zjistit informace nikoliv z textového obsahu dokumentu, nýbrž ze struktury webových stránek. Přitom se uplatňují různé způsoby, jako např. reprezentace webových stránek jako grafu [16], či klasifikace webových stránek podle analýzy struktury celého webu [9]. Další možností dolování znalostí ze struktury je analýza a sledování hypertextových odkazů stránek [4, 3, 2, 22].

2.2 Zdroje dat pro dolování znalostí

Principielně můžeme znalosti dolovat z jakéhokoliv uložiště informací. Mezi nejčastěji používané zdroje dat pro dolování potom patří zejména:

- relační databáze
- transakční databáze
- datové sklady
- ostatní (textové databáze, objektově orientované databáze...)

2.2.1 Relační databáze

Relační databáze je databáze založená na relačním modelu dat a relační algebře. Data jsou uspořádána do tabulek (*relací*), nad kterými jsou definovány přípustné operace. Sofware pro řízení databáze se obvykle nazývá *Relational Database Management System*(RDBMS). Jazykem pro definici dat(DDL) a manipulaci s daty (DML) je jazyk SQL, dotazovací strukturovaný jazyk. Relační databázový model sdružuje data do relací (tabulek), které obsahují n-tice (řádky). Tabulky (relace) tvoří základ relační databáze. Tabulka je struktura záznamů s pevně stanovenými položkami (sloupci tabulky - atributy). Každý sloupec má definován jednoznačný název, typ a rozsah - doménu. Záznam tabulky je v z matematického hlediska uspořádanou n-ticí (*tuple*) prvků. Pokud jsou v různých tabulkách sloupce stejného typu, pak tyto sloupce mohou vytvářet vazby mezi jednotlivými tabulkami. Tabulky se poté naplňují vlastním obsahem - konkrétními daty.

ID	category	att 1	att 2	att 3
1	Category 3	value 1.1	value 1.2	value 1.3
2	Category 5	value 2.1	value 2.2	value 2.3
3	Category 3	value 3.1	value 3.2	value 3.3
4	Category 2	value 4.1	value 4.2	value 4.3
5	Category 3	value 5.1	value 5.2	value 5.3
...	...	...	...	...
n	Category X	value n.1	value n.2	value n.3

Tabulka 2.1: Tabulka relační databáze.

2.2.2 Transakční databáze

Transakční databáze nejčastěji uchovávájí prodejní data pro obchodní účely. Klasickým použitím transakční databáze je databáze provedených nákupů v prodejně. Jednotlivé nákupy se ukládají ve formě transakce, kdy každá transakce obsahuje položky koupené v jednom nákupu.

Definice 2.2.3 Nechť T je množina transakcí (transakční databáze) a nechť $I = \{I_1, I_2, \ldots, I_m\}$ je množina položek. Každá transakce T v transakční databázi je množinou položek takovou, že $T \subseteq I$.

2.2.4 Datový sklad

Datový sklad je subjektivně orientovaný, integrovaný, časově proměnný, leč stálý soubor dat, který slouží pro podporu rozhodování. Datový sklad neuchovává data, která nejsou vhodná pro podporu rozhodování. Vzhledem k tomu, že do datového skladu vstupují data

z různých produkčních databází, je důležitá integrace a sjednocení dat. Toto integrování zahrnuje sjednocení stejných ukazatelů, sjednocení měřítek (například zda se budou informace o výdajích ukládat v korunách, nebo v tísících korunách atd. . .).
Všechna data v datovém skladu představují časový snímek dat z produkčních databází sejmutý v určitém ukamžiku. Datový sklad je aktualizován offline v určitých časových intervalech (měsíčně, čtvrtletně, ročně) a je rovněž analyzován odděleně od produkčních databází. Výhodou je, že nešetrný zásah do datového skladu neovlivní produkční databázi. Pro dolování jsou datové sklady nejlepším zdrojem - obsahují velké množství dat, které činí výsledky dolování relevantnější, než je tomu u "malých" produkčních databází.

2.3 Asociační pravidla

Asociační pravidla jsou jedny z nejčastěji dolovaných znalostí v datech a využívají se především při tzv. *analýze nákupního košíku*(market basket analysis). Tento proces slouží k analýze nákupních zvyků zákazníků hledáním asociací mezi položkami, které zákazníci vložili do svých nákupního košíku[11]. Nabyté znalosti mohou pomoci usnadnit provádění strategických kroků k cílené marketingové kampani. Uvažujme obchodní řetězec, kde datový specialista zjistil, že pokud si zákazník zakoupí mléko, potom si téměř vždy zakoupil také chléb. Pomineme-li triviálnost a obecnou znalost tohoto pravidla, může vedení umístit prodejní plochu mléka blízko k místu prodeje chleba, čímž jednak vyjde vstříc zákazníkům, kteří nebudou nuceni absolvovat složité cesty po supermarketu hledajíc požadované zboží, jednak mohou přimět ke koupi obou artiklů i ty, kteří původně před příchodem do obchodu jejich zakoupení neplánovali.

Formálně můžeme asociační pravidla definovat takto [11]: Nechť $I = \{I_1, I_2, \ldots, I_m\}$ je množina prvků. Nechť D je množina databázových transakcí, kde každá transakce T je množina prvků takových, že $T \subseteq I$. Každá transakce T je svázána s *identifikátorem transakce* nazývaným TID. Nechť A je množina prvků. Říkáme,že transakce T obsahuje A tehdy a jen tehdy, když $A \subseteq T$. Asociační pravidlo je implikace ve tvaru $A \Rightarrow B$, kde $A \subset I$, $B \subset I$ a $A \cap B = \emptyset$ a mají dvě základní charakteristiky - *podporu* a *spolehlivost* Asociační pravidlo má **podporu** (*support*) v D rovnu $s\%$ transakcí v D, které obsahují $X \cup Y$. **Spolehlivost** (*confidence*) pravidla udává, kolik $s\%$ transakcí v D, jež obsahují X, obsahuje také Y. Neformálně řečeno podpora udává, v kolika procentech transakcí T je obsažena množina prvků X asociačního pravidla; spolehlivost je hodnota říkající v kolika procentech transakcí kde se vyskytuje X se vyskytuje také Y[1]

$$\textit{mléko} \wedge \textit{rohlíky} \Rightarrow \textit{chléb}\ [\textit{supp 0.01, conf 0.8}]$$

Problém nalezení asociačních pravidel v datech sestává z generování pravidel, které mají spolehlivost a podporu vyšší než zadané prahové hodnoty. Taková pravidla nazýváme *silná asociační pravidla.*

Obecně mohou asociační pravidla sestávat z jakýkoliv výrazů, o který jsme schopni v konečném čase rozhodnout, zda jsou pravdivá či nikoliv. Základní typ pravidel je získáván z transakčních databází a typicky je výsledkem dříve zmiňované analýzy nákupního košíku. Transakční databáze zaznamenávají všechny provedené transakce (např. obchodní transakce), tedy informace o každém nákupu. Data v transakcích sestávají z jednoduchých boolovských atributů (které mohou nabývat pouze hodnot 0 a 1) ve stejné dimenzi. Pokud stále uvažujeme

[1] Jedná se defakto o podmíněnou pravděpodobnost $P(B|A)$.

analýzu nákupního košíku, tak typickým příkladem dimenze je **zakoupil**, kde atributy příslušné této dimenzi tvoří jednotlivé položky zboží. Pokud je hodnota atributu v transakci 1, potom zákazník příslušné zboží zakoupil; pokud je 0, potom jej nezakoupil.

2.3.1 Algoritmus pro generování silných asociačních pravidel

Výsledkem dolovacích algoritmů je množina všech **frekventovaných množin**, což je množina jistých prvků (položek) zdrojových dat. Frekventovaná množina, která obsahuje k prvků, se nazývá k-množina.

Jakmile jsou nalezeny všechny frekventované množiny z transakcí v databázi D, následuje generování silných asociačních pravidel (kde slovo *silný* vyjadřuje, že pravidlo splňuje požadavek na minimální podporu *support* i minimální spolehlivost *confidence*). To se provádí následujícím výpočtem spolehlivosti:

$$confidence(A \Rightarrow B) = \frac{podpora(A \cup B)}{podpora(A)},$$

kde *podpora(*$A \cup B$*)* je číslo, vyjadřující počet transakcí obsahujících množinu položek $A \cup B$ a *podpora(*A*)* počet transakcí obsahujících množinu položek A.

- Pro každou frekventovanou množinu L vygeneruj všechny neprázdné podmnožiny.
- Pro každou neprázdnou podmnožinu S, $S \in L$, vytvoř pravidlo $S \Rightarrow (L - S)$ právě tehdy, když $\frac{podpora(L)}{podpora(S)} \geq min\_supp$, kde *min_supp* je práh minimální podpory.

2.3.2 Algoritmy pro generování asociačních pravidel

Vývoj algoritmů pro generování asociačních pravidel přímo souvisí s již zmiňovanou analýzou nákupního košíku. Prvním použitým algoritmem vůbec byl algoritmus Apriori a jeho upravené varianty, které více či méně zvyšovaly účinnost algoritmu a snižovaly jeho obrovské paměťové nároky.

Algoritmus FP-Growth[12] přinesl oproti Apriori nebývale rychlé generování asociačních pravidel. Pracuje na principu uložení původních dat do kompaktní stromové struktury FP-Tree, čímž odpadá zdlouhavý proces generování a testování frekventovaných množin.

Apriori

Základním algoritmem pro získávání asociačních pravidel je algoritmus Apriori. Jedná se o jednoduchý algoritmus, který z frekventovaných n-množin generuje frekventované $(n+1)$-množiny, k čemuž využívá metodu prohledávání do šířky(breadth-first search).

V každém kroku algoritmu se vygenerují množiny prvků a testuje se, zda podpora těchto prvků je větší než minimální podpora. Množiny, které tímto testem úspěšně projdou, se pak stávají zdrojem pro generování množin obsahujících o jeden prvek více, než původní množina.

2.4 Klasifikace

Klasifikace, tedy přiřazování objektů reálného světa do určité kategorie, je přirozený proces, pomocí kterého si lidský mozek ujasňuje typické rysy o množině sobě podobných objektů,

přičemž tato míra podobnosti je čistě subjektivní a záleží na mnoha vlivech. Aniž bychom si to uvědomovali, klasifikace nám usnadňuje rychlé ohodnocení počitků bez nutnosti jejich kompletní analýzy. Malému dítěti nedělá problém dotknout se holou rukou vařícího se hrnce, neboť nedokáže kvalitně vyhodnotit, jaký bude mít kontakt lidské tkáně a rozžhaveného objektu důsledky. Až po tom, co se poprvé popálí, v jeho mozku se uloží příslušná informace, kterou bychom mohli vyjádřit například takto:

Pokud se z objektu na plotně kouří, potom je nebezpečný

V průběhu života následně dochází k postupnému "upřesňování" vytvořených pravidel na základě dalšího prožití identické (nebo podobné) situace. Další a další kontakty s hrncem na plotně vedou ke kvalitnějšímu vyhodnocení následujících situací a postupem času se mohlo pravidlo transformovat do podoby:

Pokud se objekt na plotně podobá hrnci, vaří se, kouří se z něj a pokud nemá dřevěné držadlo, je objekt nebezpečný.

Jiným příkladem klasifikace objektů může být určení bonity klienta bankovních institucí. Představme si sama sebe jako ředitele banky, který poskytuje klientům finanční úvěry. Problém je v tom, že určitá skupina klientů úvěr problémy se splácením splátek a vaším úkolem je rozhodnout, jak rozlišit ty klienty, kteří jsou bezproblémoví, a kterým není rizikové peníze půjčit, a ty, kteří naopak splácet nebudou.
Pokud máme k dispozici záznamy o např. 1000000 posledních žadatelů o úvěr, můžeme analýzou osobních informací (např. výše platu, věk) zařadit každého do jedné ze tříd {schopný splácet, neschopný splácet}. S využitím nabitých znalostí o minulých zákaznících pak můžeme nově příchozí klienty ohodnotit tak, že např. porovnáme jejich osobní informace s již analyzovanými informacemi. Informace o tam obrovském množství klientů jsou však velmi těžce zpracovatelné lidským mozkem, a tak je zcela logické, že se v průběhu let začaly vytvářet techniky a nástroje pro automatizovanou klasifikaci.

V této kapitola bude vysvětlen a diskutován význam klasifikace z hlediska pomoci při rozhodování. Zvláštní samostatná část bude věnovaná využití asociačních pravidel při klasifikaci.

U klasifikačních metod [3] sledujeme několik důležitých vlastností, které nám pomáhají je mezi sebou porovnávat a určovat oblasti vhodného použití(viz.[11]):

- **Stupeň přesnosti** klasifikátoru udává jak přesně dokáže klasifikátor ohodnotit nově příchozí vzorky a je určena procentuální úspěšnosti klasifikace.
- **Rychlost** klasifikátoru vyjadřuje výpočetní čas spojený s učením a testováním klasifikátoru.
- **Robustnost** je schopnost klasifikátoru vypořádat se i s poškozenými vstupními daty (zašuměná data, chybějící hodnoty).
- **Stabilita** vypovídá o tom, jak je klasifikátor schopný správné funkčnosti i na velkém množství dat.
- **Interpretovatelnost** udává stupeň srozumitelnosti klasifikátoru.

[3] Klasifikační metody jsou metody realizující mapovací funkcí klasifikátoru.

Výběr klasifikační metody závisí především na požadovaných vlastnostech. Pokud potřebujeme rychlý klasifikátor pro klasifikaci vzorků v reálném čase, bude nám záležet na její rychlosti a oželíme například interpretovatelnost.

2.4.1 Klasifikace jako metoda dolování znalostí

Klasifikace je proces zařazení objektu do určité třídy a sestává ze dvou fází[11]:

1. *Fáze trénování klasifikátoru*(Training phase)
2. *Fáze testování*(Testing phase)

Trénovací fáze

V *trénovací fázi* je z dat vytvořená tzv. trénovací množina - vyberou se vzorky dat, které budou reprezentovat klasifikátor. U těchto vybraných dat musíme přesně vědět, do které třídy jsou zařazena (tříd musí být předem známý konečný počet) [1] Trénovací množinu si můžeme představit jako dvojici $(X, Class)$, kde vektor $X = (x_1, x_2, x_3, \ldots, x_n)$ je vektor hodnot n atributů nějakého objektu a $Class$ je označení třídy, do které je objekt přiřazen. Úkolem klasifikátoru je pak naučit se funkci $y = f(X)$, která předpovídá třídu na základě znalosti vektoru X. Mapovací funkce může být reprezentovaná různými způsoby - např. formou *klasifikačních pravidel*, *rozhodovacích stromů*, různých *matematických vzorců* apod. Výstupem první fáze klasifikace je tedy jakási černá skříňka, které když na vstup přiložíme vektor X, tak na výstupu vrátí třídu $Class$, do které (s určitou pravděpodobností) objekt reprezentovaný X patří.

Testovací fáze

Ve druhé fázi klasifikace, kterou nazýváme *fází trénování* dochází k ověření vlastností klasifikátoru, především určení MR[2] a určení, do jaké míry se klasifikátor hodí pro řešení toho konkrétního problému.
Pro tento krok musíme opět vybrat vzorky dat se známou třídou, do které objekt patří - *testovací množinu*. Tyto vzorky by se měli lišit od vzorků použitých v první fázi klasifikace. Zatímco ve fáze trénování se klasifikátor jistým způsobem naučil "předpovídat", v této fázi se hodnotí míra kvality jeho předpovědí a na základě známosti tříd, do kterých vzorek patří, se určuje jeho MR.

2.4.2 Hodnocení klasifikačních metod

Pro ohodnocení klasifikátorů z hlediska kvality předpovědi existuje celá ředa metrik. Ty se hodí také v případě, že potřebujeme porovnat několik různých klasifikačních metod. Asi nejčastěji užívanou metrikou je *přesnost*, jenž udává poměr správně klasifikovaných dokumentů ku všem dokumentům v testovací množině. Dalé se často mluví o *chybovosti* klasifikátoru, která je definovaná jako $1-$ *přesnost*.

- p^+ (*true_positive*) = počet dokumentů klasifikovaných do správné třídy

[1] Trénovací fáze je příkladem učení s učitelem, kdy známe třídy, do kterých se budou trénovací vzorky dat přiřazovat.

[2] MR = *Misclassification Rate*, neboli pravděpodobnost špatné klasifikace. Čím nižší hodnoty nabývá, tím kvalitnější výsledky má klasifikátor.

	C_1	C_2
C_1	*true_positive*	*false_negative*
C_2	*false_positive*	*true_negative*

- n^- (*true_negative*) = počet dokumentů správně neklasifikovaných do třídy
- p^- (*false_positive*) = počet dokumentů klasifikovaných do chybné třídy
- n^+ (*false_negative*) = počet dokumentů chybně neklasifikovaných do třídy

Musí platit, že $p^+ + p^- + n^+ + n^- = N$, kde N je rovno celkovému počtu dokumentů v testovací množině.

Přesnost(*precision*)

Udává počet správně správně klasifikovaných dokumentů v poměru k počtu všech dokumentů, které byly klasifikované do jakékoliv třídy.

$$precision = \frac{p^+}{p^+ + n^+}$$

Z jiného pohledu lze na přesnot pohlížet jako na pravděpodobnost výběru správně klasifikovaného dokumentů z množiny všech klasifikovaných.

Úplnost(*recall*)

Je metrikou, která vyjadřuje pravděpodobnost, s jakou mezi je dokument správně klasifikován do příslušné kategorie.

$$recall = \frac{p^+}{p^+ + p^-}$$

Chybovost(*error rate*)

Je často využívaným měřítkem pro hodnocení klasifikačních metod. Udává poměr všech špatně klasifikovaných dokumentů ku všem dokumentům v testovací množině dokumentů. Kromě názvu *Error-rate* se v literatuře často můžeme setkat s pojmem *Misclassification Rate - MR*; jejich význam je v kontextu klasifikace totožný.

$$error\ rate = \frac{n^+ + p^-}{p^+ + p^- + n^- + n^+}$$

Obecně můžeme MR chápat jako pravděpodobnost, s jakou klasifikátor daný dokument D zařádí do špatné třídy.

Kromě těchto existuje celá řada dalších metrik, které dohromady tvoří mocný nástroj pro popis vlastností klasifikačních metod. Z dalších důležitých metrik se často používá také např. *fallout*, *f-measure*, *sensitivity*, *specificity* a jiné.

2.5 Tradiční klasifikační metody

2.5.1 Rozhodovací stromy

Rozhodovací strom je stromová struktura, v niž uzly nesou hodnotu určitého atributu z množiny atributů, hrany mezi uzly definují podmínku vztahující se k atributu výše položenému uzlu a listy udávájí třídu, do niž je vstupní vzorek klasifikován. Rozhodovací strom je grafickým vyjádřením *rozhodovacích pravidel*, v rámci klasifikace je možné na na rozhodovací strom a rozhodovací pravidla nahlížet jako na ekvivalentní modely pro klasifikaci.

U rozhodovacích stromů je důležité správné sestrojení samotného stromu, tedy postupném určení "nejvýznamnějších" atributů s nejvyšší rozhodovací schopností a rozdělení hodnot atributu.

Neuronové sítě

Oblíbeným klasifikačním modelem jsou umělé neuronové sítě, které simulují chování sítí neuronů lidského mozku. Základní jednotkou neuronové sítě je neuron se vstupy a výstupy.

Pro každý vstup x_i neuronu i je definována váha w_i a pro celý neuron bias Θ; transformací vypočítané sumy

$$\sum_{i=1}^{n} w_i x_i + \Theta$$

jistou aktivační funkcí získáme výstupní hodnotu neuronu, která může být v případě sítí neuronů šířena na vstupy jiných neuronů, případně může tvořit výstupní hodnotu klasifikace vzorku.

Učení pro klasifikaci neuronovými sítěmi spočívá ve správném nastavení vah w_i všech neuronů n_i a biasu Θ u všech neuronů sítě tak, aby výstupy koncových neuronů správně ohodnotily vzorky do příslušné třídy. Na počátku se hodnoty nastaví náhodně a postupným testováním se upravují do té doby, dokud nedosáhneme požadované přesnosti.

Nejčastěji používanou neuronovou sítí je sít *Backpropagation* tvořená neurony nazývanými *perceptron*. Neurony v síti Backpropagation jsou seskupeny do vrstev; rozlišujeme vstupní vrstvu, skryté vrsty a vrstvu výstupní. Vstupní vrstva je tvořena neurony přijímající vstupní hodnoty. Ve skrytých vsrtvách dochází k postupnému zpracování hodnot ze vstupní vrstvy a k šíření hodnot do vrstvy výstupní, jejíž neurony na výstupu určují výslednou hodnotu vstupního vzorku. Učení sítě Backpropagation je založená na zpětném šíření chyby, kdy se hodnota výstupní vrstvy porovnává s očekávanou hodnotou a podle toho jsou upraveny váhy neuronů od poslední vrstvy až po vrstvu vstupní.

Metoda k-sousedství (k-nearest neighbor)

Velice jednoduchou metodou pro klasifikaci textu je metoda *k-sousedství* pracující na principu, že dva sobě podobné dokumenty budou pravděpodobně zařazené do stejné třídy. Podobnost dokumentů se určuje na základě Eukleidovské vzdálenosti vektorů popisujících dokument. Z trénovacích dat je vybráno právě k vzorů, jejichž vzdálenost je nejmenší k právě klasifikovanému prvku. Klasifikovaný prvek je potom zařazen do té třídy, která je nejčetnější u těchto k vybraných prvků.

Kapitola 3

Klasifikace dokumentů založená na asociačních pravidlech

3.1 Klasifikace textových dokumentů

Klasifikace textových dat (TC - Text Classification) je úloha automatického třídění dokumentů do daných tříd (kategorií). Tato úloha spadá do oblasti *získávání informací*(Information Retrieval) a *strojového učení*(Machine Learning).

Typicky prvním krokem při klasifikaci textových dat je transformace dokumentu, který je ve většině případů reprezentován jako řetězec znaků, do podoby vhodné pro algoritmus klasifikační metody. Výzkumy posledních let poukázaly především na důležitost *stemmingu* slov - určení kořene slov. To vede k reprezentaci textu jako dvojice atribut-hodnota, kde u každého slova (slovního kořene) evidujeme počet výskytů v dokumentu. Jedním z problémů při stemmingu je velký počet irelevantních slov, proto se většinou používají různé techniky vedoucí k redukci počtu slov.

Obecně se při klasifikaci textu postupuje obdobně jako u klasifikace relačních dat - z trénovací množiny dat ze vytvoří klasifikační schema, podle kterého následně klasifikujeme další dokumenty. Zásadní rozdíl je ve skutečnosti, že relační data jsou plně strukturovaná, např. v n-tici { *slunečno, teplo, vlhko, bezvětří, procházka* } hodnota *slunečno* koresponduje s atributem *stavMračen*, hodnota *horko* s atributem *teplota* atd .... Cílem asociační analýzy je rozhodnout, jaká množina dvojic atribut-hodnota atributu má největší vliv na to, zda se půjde jistá osoba projít na procházku. Oproti tomu databáze dokumentů nejsou takto strukturované a klasické relační klasifikační metody, např. klasifikační stromy, zde nejsou efektivní.

3.1.1 Praktické využití

Filtrování textu

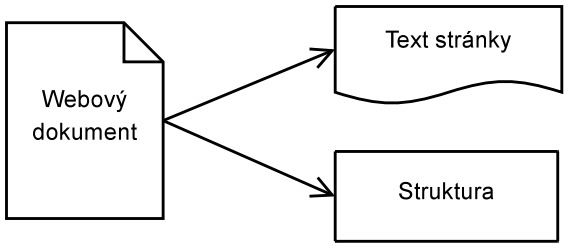
Filtrování textu je proces ohodnocení, či klasifikace příchozích dokumentů podle jejich obsahu a rozhodnutí, zda dokument je přípustný, či nikoliv. Typickými případy filtrovacího systému jsou filtr elektronické pošty, filtr nepřípustného obsahu, nebo filtr příspěvků do internetových diskuzí. Filtrovací systém může blokovat přijetí dokumentu, o který příjemce nemá zájem. Filtrování je případ binárního TC, kdy se provádí klasifikace dokumentů do dvou disjunktních kategorií - relevantní a irelevantní.

Organizace dokumentů

Potřeba organizovat dokumenty do kategorií je zde od počátku existence textových dokumentů vůbec. Krátký popis dokumentu (např. formou názvu souboru) je výhodný při prohlížení malého počtu dokumentů, při větším množství je vyhledání konkrétního dokumentu obtížné. Proto dokumenty organizujeme hierarchicky do kategorií, podkategorií atd. Např. v redakci novin může přijít požadavek na organizaci napsaných článků pro budoucí jednodušší vyhledávání. Možnými kategoriemi zde mohou být "Zprávy z domova", "Zahraniční zprávy", "Sportovní zprávy" aj.

3.2 Klasifikace webových stránek

World Wide Web představuje celosvětový obrovský distribuovaný zdroj informační centrum novinových zpráv, reklam, obchodních informací, učebních materiálů, a mnoha dalších informačních služeb. Mimo jiné Web poskytuje také bohatou dynamickou kolekci hyperlinkových odkazů, informací o přístupech na stránku, zátěží serverů, a dalších informací vhodných pro dolování znalostí.



Obrázek 3.1: Oddělení textu a struktury webového dokumentu

Složitost webových dokumentů(stránek) je daleko větší než u jakékoliv kolekce tradičních textových dokumentů. Webové stránky postrádají sjednocenou strukturu dokumentu, jako je např. název autora, obsah atd. . . a vyhledávání v nich je proto obtížné. Říká se, že 99% informací na Webu je pro 99% návštěvníků nepotřebných. Toto pravidlo zcela odpovídá skutečnosti, kdy je problém v závalů webových dokumentů problém najít ty, které odpovídají našim zájmům v dané oblasti. Klasifikace webových stránek je ale také v lecčems podobná klasifikaci obyčejných textových dokumentů. Kromě samotného textu je dalším možným zdrojem znalostí rozmístění prvků dokumentu, tj. rozložení a vlastnosti jednotlivých částí jako jsou navigační menu, reklamní banner, nebo tělo dokumentu. Je zřejmé, že např. webová stránka zpravodajského deníku bude mít jiné rozložení, než osobní stránka, či blog. V oblasti dolování znalostí segmentací probíhají výzkumy intenzivně až v poslední době, věnuje se jí např. [20].

Ideální klasifikátor při určování třídy, do které dokument patří, provádí klasifikaci dvakrát - jednak se provede klasifikace podle textového obsahu stránky, jednak se dokument klasifikuje podle rozložení částí dokumentů na stránce. Úkolem klasifikačního systému je pak výsledky obou kroků sjednotit a určit kategorii podle obou kriterií.

3.2.1 Klasifikace s využitím asociačních pravidel

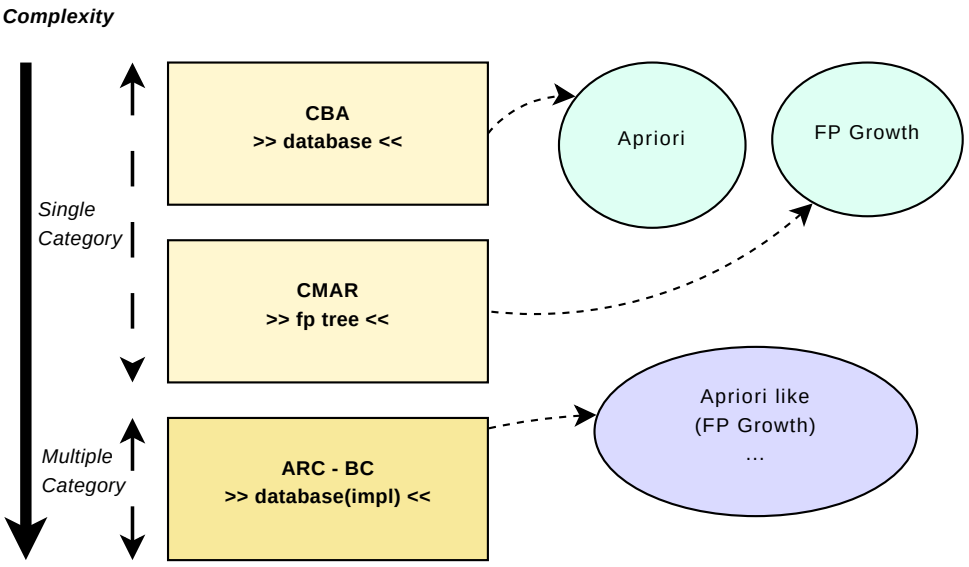
Metoda založená na asociačních pravidlech klasifikuje dokumenty na základě asociace často se vyskytujícího vzorku textu (slovo, slovní spojení) s třídou, která je tímto vzorkem reprezentovaná. Problém je, že v textu se vyskytuje mnoho často se opakujících vzorků (spojky, předložky), které o zařazení do třídy nemají žádný vliv; úkolem asociační klasifikátoru je tyto vzorky vyloučit a najít pouze vhodné vzorky textu.

V případě dat relačních se asociační klasifikátor od textového odlišuje zejména ve fázi dolování asociačních pravidel, kdy místo vzorků textu tvoří frekventované množiny dvojice atribut-hodnota.

Asociační klasifikátory pracují obecně v několika krocích. Nejprve musí data připravit pro metodu *dolování asociačních pravidel*. Nalezená pravidla se následně mohou seřadit podle kvality a právě na základě těchto pravidel klasifikátor určí, do které třídy dokument patří. Pro klasifikaci asociačními pravidly existuje celá řada metod.

Jednoduchá metoda CBA(Classification-Based Association)[14] provádí vícenásobné průchody daty a hledá asociační pravidla, přitom pracuje na principu podobném algoritmu Apriori. Nový dokument je zařazen do té třídy, která je pokrytá prvním pravidlem v seřazené množině získaných asociačních pravidel.

Metoda CMAR(Classification based od Multiple Association Rules)[13]je v mnohém podobná předchozí metodě, liší se však přístupem, jakým se hledají asociační pravidla, a jakým se vytváří samotný klasifikátor. Místo algoritmu Apriori je pro nalezení asociačních pravidel použitá varianta efektivního algoritmu *FP-growth*(Frequent Pattern-growth)[12], která v datové struktuře nazývané *FP-strom*(FP-tree) uchovává informace o všech frekventovaných množinách datového souboru. Pro uchování asociačních pravidel používá také stromovou strukturu, tzv. *CR-strom*(Classification Rule-tree). Metoda CMAR dosahuje při praktickém použití vyšší efektivity klasifikace a vyšší průměrné přesnosti klasifikace než algoritmus CBA[13].



Obrázek 3.2: Oddělení textu a struktury webového dokumentu

Nevýhodou výše zmíněných metod CBA a CMAR bylo, že klasifikovaly vzorky pouze do

jedné třídy. V praxi se s dokumenty náležícími výhradně do jedné kategorie setkáme velmi zřídka a výsledky klasifikace takového dokumentu pak mohou být značně zkreslené. Tento zásadní problém dal za vznik dvěma metodám určeným výhradně pro klasifikaci textových dat - metodám ARC-AC(Association Rule-based Classifier with All Categories) a ARC-BC(Association Rule-base Classifier By Category). Obě metody se liší pouze ve způsobu hledání asociačních pravidel. Zatímco ARC-AC hledá pravidla v celé trénovací množině dokumentů, ARC-BC nejprve rozdělí dokumenty do skupin podle kategorie, do které patří, a extrakci asociačních pravidel pak provádí zvlášť pro každou skupinu. Podle [24] je ARC-BC výhodný i v případě kategorií, do kterých spadá pouze malé procento dokumentů.

3.3 Metoda ARC-BC

Pro klasifikaci dokumentů vzniklo velké množství různých metod. Metoda ARC-BC[1] se snaží zkombinovat jejich přednosti (rychlost, interpretovatelnost) do klasifikační metody využívající asociační pravidla. Klasifikátor byl navržen s ohledem na dva hlavní problémy: (1) nalezení kvalitních reprezentativních asociační pravidel v textových datech generováním a prořezáváním; a (2) použití nalezených pravidel k vybudování textového klasifikátoru.

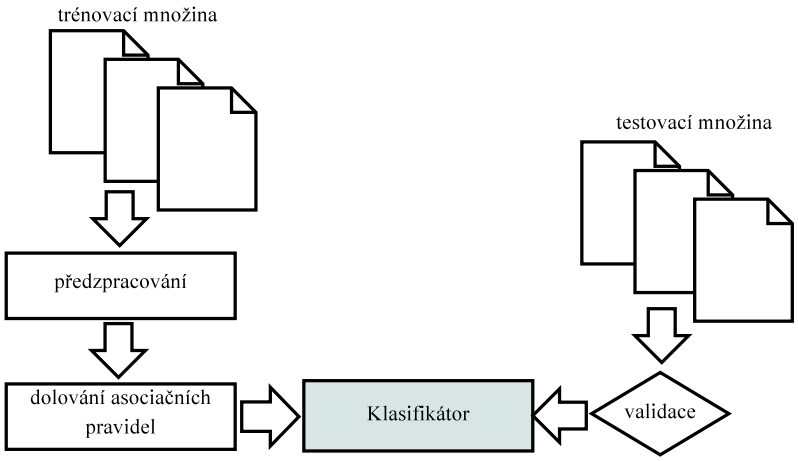
3.3.1 Popis metody

ARC-BC klasifikátor (Association Rule-based Classifier By Category) je klasickým asociačním klasifikátorem. Označení ARC-BC vyjadřuje hned několik důležitých informací o klasifikátoru. ARC říká, že klasifikátor pracuje na principu generování asociačních pravidel, BC potom upřesňuje, jakým způsobem se generují asociační pravidla v jednotlivých kategoriích. V případě ARC-BC se postupuje tak, že pokud dokument náleží do více jak jedné třídy, potom se vyskytuje ve vstupních datech tolikrát, do kolika tříd patří a dolování pravidel se provádí samostatně pro každou množinu dokumentů D_i, ve které jsou pouze dokumenty náležící do třídy c_i. Naproti tomu metoda ARC-AC (Association Rule-based Classifier with All Categories)[24] doluje asociační pravidla z kompletní množiny dokumentů. Problémem metody ARC-AC je, že obtížně zpracovává kategorie, do kterých spadá jen malý počet dokumentů, viz.[1].

Na vstup klasifikátoru předložíme kolekci dokumentů (obecně jakýchkoliv dat), po provedení řady kroků je nalezen klasifikační model. Prvním krokem v tomto netriviálním procesu je předzpracování vstupních dat[1]. Dalším krokem je vybudování asociačního klasifikátoru hledáním asociačních pravidel algoritmem Apriori.[2], Jakmile je vygenerovaná množina asociačních pravidel, důležitým krokem je použití prořezávacích technik vedoucích k redukci počtu pravidel. Fáze redukce pravidel je velice důležitá, neboť velké množství pravidel má zásadní vliv na rychlost klasifikátoru. Po prořezání pravidel je vytvořen asociační klasifikátor - jeho znalostní bázi tvoří prozeřaná asociační pravidla. V posledním kroku se vytvořenému klasifikátoru předkládá dokument ke klasifikaci a klasifikátor se snaží předpovědět do které třídy (resp. tříd) dokument náleží. Princip činnosti klasifikátoru založeného na generování asociačních pravidel je zobrazen na Obrázku 7.3

[1]Data mohou být v surové podobě zašuměná, neúplná, či duplicitní, pro správnou funkčnost klasifikátoru je potřeba tyto neduhy odstranit; viz např. [11]

[2]Algoritmus Apriori je jednoduchý algoritmus pro hledání asociačních pravidel, jeho nevýhodou je (v případě velkého množství dat) velká časová náročnost a nutnost mít stále aktivní přístup ke zdroji dat (např. databáze). Není problém nahradit algoritmus Apriori jiným, výkonnějším, algoritmem (např. algoritmem FPTree[11]), nicméně pro použití v klasifikátoru, kde k fázi trénování dochází pouze zřídka, není jeho použití nezbytné.



Obrázek 3.3: Data z trénovací množiny jsou předzpracovaná a jsou z nich vydolované asociační pravidla. Na základě pravidel je natrénován asociační klasifikátor. Odbdobným způsobem proběhne nalezení pravidel u testovací množiny dat; tato pravidla ale slouží k validaci klasifikátoru.

V dalších částech této kapitoly budou popsány jednotlivé fáze činnosti algoritmu, tj. fáze předzpracování dat, dolování asociačních pravidel, prořezání asociačních pravidel a fáze klasifikace nového dokumentu.

3.3.2 Předzpracování dat

Další z mnoha výhod metody spočívá ve snadném přizpůsobením se na různé zdroje dat - textová data, relační, transakční aj. Originální verze metody pracuje nad textovými daty a očekává dokumenty ve tvaru D_i={Cat_i, t_1, t_2, t_3,... t_n. Pro správnou funkčnost klasifikátoru je nutné data převést do této podoby, nebo modifikovat algoritmus pro dolování asociačních pravidel. Teoreticky je možné použít jakýkoliv algoritmus pro dolování asociačních pravidel, čímž se značně rozšiřují možnosti klasifikace.

3.3.3 Dolování asociačních pravidel

Pokud již máme připravena vstupní data, algoritmem Apriori se vygenerují asociační pravidla. V některých případech (velmi často) můžeme narazit na problém, že vygenerovaných pravidel je příliš velké množství, z toho řada může být pro klasifikaci zbytečná. Z těchto důvodů se generují pouze pravidla, které mají na pravé straně označení nějaké třídy c_i.

Algoritmus 1 ARC-BC Dolování asociačních pravidel v dokumentech

Vstup: Množina dokumentů D ve tvaru D_i=$\{Cat_i, t_1, t_2, t_3, \ldots t_n\}$, kde Cat_i je kategorie přiřazená dokumentu, t_n jsou vybrané výrazy; Minimální spolehlivost *minsupp*; Minimální spolehlivost *minconf*;

Výstup: Množina asociačních pravidel ve tvaru $t_1 \wedge t_2 \wedge t_3 \wedge \ldots \wedge t_n \Rightarrow Cat_i$ kde Cat_i je kategorie a t_j je nějaký výraz;

1: $C_1 \leftarrow$\{ Kandidáti na 1-frekventované množiny a jejich podpora \}
2: $F_1 \leftarrow$\{ Frekventované 1-množiny a jejich podpora \}
3: $i = 2$
4: **while** $F_{i-1} \neq \emptyset$ **do**
5: $p_{i_1} = i/2$
6: $p_{i_2} = i - p_{i_1}$
7: $C_i = F_{p_{i_1}} \bowtie F_{p_{i_2}}$
8: $C_i = C_i - \{c | sizeof(c) \neq i\}$
9: $F_i = \{c \in C_i | support(c) \geq minsupp\}$
10: **end while**
11: $M = \bigcup_i \{c \in F_i | i \geq 1\}$
12: $R = \emptyset$
13: **for all** frequent itemsets f **in** M **do**
14: najdi všechny d_x z D, které obsahují f, a vytvoř asociační pravidla $r_x : f \Rightarrow Cat_x$
15: **if** confidence(r_x) $\geq$ minconf **then**
16: $R = R \cup r_x$
17: **end if**
18: **end for**

V krocích (1 a 2) jsou nalezeni kandidáti na frekventované 1-množiny. V praxi to znamená nalézt takové výrazy, které jsou pravdivé v tolika dokumentech F, aby byla splněna podmínka minimální podpory *minsupp*. Vytváření i-frekventovaných množin probíhá spojením již vytvořených frekventovaných množin nižšího řádu (kroky $4 - 11$). Generování se opakuje tak dlouho, dokud v daném kroku i, $i \geq 2$ nejsou nalezené žádné frekventované i-množiny. Po nalezení všech frekventovaných množin $F_i | i \geq 1$ v množině dokumentů D se nalezne množina asociačních pravidel M_R, která obsahuje pravidla R ve tvaru $R : t_1 \wedge t_2 \wedge t_3 \wedge \ldots \wedge t_n \Rightarrow Cat_i$, viz. kroky $(12 - 16)$.

3.3.4 Prořezávání asociačních pravidel

Výstupem předchozí fáze metody byla sada asociačních pravidel, která jistým způsobem popisují textový dokument. Těchto pravidel může být v některých případech velké množství, což způsobuje různé komplikace při následné klasifikaci.

Jednak muže obrovském množství pravidel obsahovat šum vedoucí k chybám při klasifikaci, se zvyšuje doba klasifikace. Pro řešení problému velkého počtu pravidel se provádí *prořezáním asociačních pravidel* - odstraněním irelevantních a málo obecných pravidel. V případě, kdy požadujeme opravdu rychlé odezvy (online zpracování dotazů atd ...) je prořezání asociačních pravidel již více méně nutností, než možností k případnému urychlení. Před samotným prořezáním je výhodné určit pořadí, ve kterém se budou pravidla v množině zpracovávat podle Definice 3.3.5:

1. *health*=bad ∧ *finance*=bad ∧ *children*=0 ⇒ *Class*=looser (0.03, 0.85)
2. *health*=bad ∧ *finance*=perfect ∧ *children*=2 ⇒ *Class*=lucky (0.2, 0.95)
3. *health*=bad ⇒ *Class*=looser (0.4, 0.98)
4. *health*=good ∧ *finance*=bad ⇒ *Class*=healthy (0.23, 0.77)

Tabulka 3.1: Množina R asociačních pravidel nalezených v prvních fázi

Definice 3.3.5 *Mějme pravidla* $R1$*:* $T1 \Rightarrow C$ *a* $R2$*:* $T2 \Rightarrow C$*. Potom pravidlo* $R1$ *je obecnější než pravidlo* $R2$ *právě tehdy, když* $T1 \subseteq T2$*.*

Algoritmus pro prořezání pravidel je postaven na jednoduchém principu - projdi množinu všech asociačních pravidel, ponechej nejvíce obecná pravidla (s malým počtem výrazů na levé straně) s nejvyšší spolehlivostí, ostatní pravidla smaž. [1] navrhují algoritmus, který provádí ještě agresivnější redukci počtu asociačních pravidel, nicméně pro účely dolování např. vizuálních vlastností plně vystačuje postup podle Algoritmu 2.

Předpokládejme, že dolovací algoritmus s fáze 1 našel sadu asociačních pravidel, která je v Tabulce 1.1. Množina obsahuje čtveřici pravidel, která klasifikují člověka podle hodnot atributů *health*, *finance* a *children* do jedné ze tříd {looser, healthy, lucky}. V závorce za asociačními pravidly je hodnota podpory pravidla *support* a hodnota spolehlivosti asociačního pravidla *confidence*.

Tato pravidla jsou vstupem pro algoritmus prořezání pravidel, který zjistí, že pravidlo č. 3 svými atributy na levé straně pokrývá pravidlo 1 a 2, přičemž spolehlivost pravidla 3 je vyšší než spolehlivost 1. a 2. pravidla. Proto budou první dvě pravidla odstraněny. Odstranění pravidel je zcela logickým krokem - proč uchovávat pravidla, která mají nižší, příp. stejnou podporu (pravidlo $t_1 \Rightarrow c1$ má vždy vyšší, nebo stejnou podporu jako pravidlo $t_1 \wedge t_2 \Rightarrow c1$) a zároveň nemají vyšší spolehlivost?

Algoritmus 2 Prořezání asociačních pravidel

Vstup: Množina R asociačních pravidel získaných ve fází dolování asociačních pravidel

Výstup: Zredukovaná množina asociačních pravidel R' obsahující pravidla, která budou použita ve fázi klasifikace

1: Seřaď pravidla v R podle Definice 3.3.5
2: **for all** pravidla r **in** R **do**
3: najdi pravidla, která jsou více obecná, než r a odstraň ta, která mají nižší spolehlivost, než má r
4: **end for**

3.3.6 Klasifikace nového dokumentu

Množina asociačních pravidel, která prošla sítem prozeřání pravidel v předchozím kroku tvoří znalostní bázi klasifikátoru. Podle těchto pravidel se bude klasifikační algoritmus snažit předpovědět, do které třídy nově příchozí dokument patří.

Proces předpovědi probíhá tak, že klasifikátor prochází asociační pravidla ve znalostní bázi a zkouší, jestli levé strany pravidel (kde jsou výrazy) pokrývají nový dokument. V takovém případě se přiřadí s jistou pravděpodobností dokument do třídy, která je na pravé straně testovaného asociačního pravidla.

Obecně můžeme rozlišit dva typy klasifikace. Nejjednodušší způsob zařazuje nově příchozí dokument *právě* do jedné třídy; do té, která je podpořena nejvyšším součtem spolehlivostí asociačních pravidel pokrývajících dokument. Nevýhodou tohoto jednoduchého způsobu klasifikace je, že dokument může mít společné prvky s více třídami a klasifikátor vybere pouze nejvíce dominantní třídu.

Sofistikovanější způsob umožňuje dokument přiřadit do více tříd. Originální metoda prezentovaná v [1] k tomu využívá tzv. *dominantní faktor*. V této práci byla použita zjednodušená metodu, která rozdělí pravidla pokrývající dokument podle kategorie na pravé straně, a podle spolehlivosti pravidel pro každou třídu urči, jak velká je důvěra v to, že dokument patří právě do té konkrétní třídy.

Algoritmus 3 Klasifikace nového dokumentu

Vstup: Nový dokument o; asociativní klasifikátor (ARC); minimální spolehlivost (práh spolehlivosti) c

Výstup: Kategorie, ke kterým je nový dokument přiřazen;

1: $S \leftarrow \emptyset$
2: **for all** asociační pravidla r v ARC **do**
3: **if** $r \subset o$ **then**
4: proveď $cnt = cnt + 1$
5: **end if**
6: **if** $cnt == 1$ **then**
7: $frc \leftarrow$ spolehlivost pravidla r
8: přidej pravidlo r do množiny S
9: **else if** spolehlivost pravidla $r > frc - c$ **then**
10: přidej pravidlo r do množiny S
11: **else**
12: exit
13: **end if**
14: **end for**
15: rozděl množinu S do množin podle kategorií: $S_1, S_2, \ldots, S_n$
16: **for all** vytvořené množiny $S_1, S_2, \ldots, S_n$ **do**
17: spočítej součet spolehlivostí pravidel v množině S_k a vyděl tento součet počtem pravidel v S_k
18: **end for**
19: zařaď o do kategorií s důvěrou, která je udaná hodnotou vypočtenou v posledním kroku

Kapitola 4

Návrh systému pro klasifikaci webových stránek

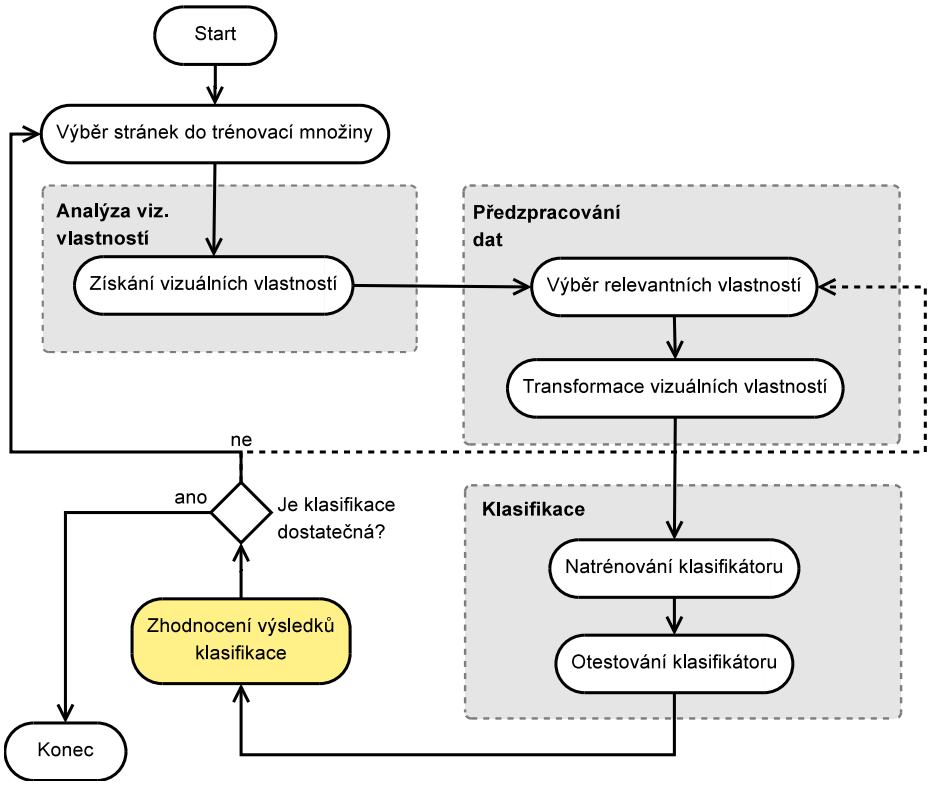
V předchozích kapitolách byl diskutován problém klasifikace dokumentů, včetně dokumentů na Webu, byly vysvětleny metody, které byly pro klasifikaci s postupem času navržené a následně otestované. Práce se zaměřuje na klasifikace webových stránek z pohledu návrhu komplexního klasifikátoru, tedy klasifikátoru, který by současně klasifikoval na základě textového charakteru dat a zároveň na základě vizuálních vlastností (rozložení stránky) ve formě dat relačních. Této úloze se zatím podrobnější výzkumy vyhýbaly, částečně se problematice klasifikace podle vzhledu stránek věnuje [20].

Cílem práce bude pokusit se o přizpůsobení textového klasifikační metody ARC-BC pro relační data. Metoda byla vybraná jednak s ohledem na dostatečnou přesnost klasifikace textových dat [1], jednak proto, že jsou výsledky metody ve formě asociačních pravidel snadno srozumitelné a přehledné. Dalším důvodem bylo to, že znalosti ve formě asociačních pravidel mají dobrý předpoklad být vhodným typem pro klasifikaci právě relačních dat.

4.1 Popis

V případě dobrých výsledků klasifikace výše zmíněného klasifikátoru by bylo možné jednoduchým způsobem sestrojit multikriteriální klasifikační systém pro klasifikaci webových stránek. Takový systém by sestával z následujících částí:

- **Systém pro extrakci vizuálních vlastností z webových stránek**
  Jedním z problémů klasifikace podle vizuálních vlastností je vydolování informací z webových stránek. Touto problematikou se zabývá Ing. Radek Burget PhD., který pro potřeby projektu poskytne data získaná vizuálním analyzátorem webových stránek. Analyzátor pracuje na principu detekce oblastí webové stránky.
- **Systém pro extrakci textových informací z webových stránek**
  Tato část systému by se starala o extrakci textových dat z webových stránek.
- **Klasifikátor ARC-BC**
  Samotná klasifikace připravených dat by byla jednodušše realizovaná ARC-BC klasifikátorem, který bude sestávat z všech příslušných částí - z části pro dolování asociačních pravidel, z části pro prořezání asociačních pravidel a z části pro klasifikaci nových dokumentů. Klasifikátor by ze vstupních textových dat a dat vizuálních vlastností klasifikoval stránku do příslušné třídy.



Obrázek 4.1: Systém pro klasifikaci webových stránek podle vizuálních vlastností

Obrázek 4.1 zachycuje princip činnosti klasifikačního systému (zde pouze vizuálních vlastností). Nejdříve dochází k výběru stránek, které budou reprezentovat trénovací množinu. Analyzátor vizuálních vlastností z nich potom vydoluje informace o vzhledu a pošle je do systému pro předzpracování dat. Zde se provede selekce relevantních vlastností a transformace dat do vhodné podoby. Klasifikátor použije tato data pro natrénování a otestování vlastností. Podle dosažených výsledků pak dojde buď k dalším pokusům s výběrem vlastností pro klasifikaci (příp. k výběru nové množiny stránek), nebo se proces klasifikace ukončí a klasifikátor bude připravený pro klasifikaci stránek.

4.2 Cíl projektu

Hlavní idea projektu tkví v centralizovaném pojetí klasifikace webových stránek. Je známo mnoho metod pro klasifikaci textových dat, stejně tak mnoho metod pro klasifikaci dat relačních. Cílem této práce je použít některou ze stavájících klasifikačních metody a pokusit se ji přizpůsobit pro relační data, čímž by bylo následně možné jej využít pro zpracovaní strukturované(semistukturované) i nestrukturované části webového dokumentu. Pro tento účel jsem zvolil metodu ARC-BC.

Je zřejmé, že v průběhu adaptace klasifikátoru na relační data bude potřeba modifikovat algoritmus metody ARC-BC a zabývat se problémy (jako např. diskretizace num. atributů), o kterých v případě dat textových není nutné uvažovat.

Sestrojená klasifikační metoda bude po implementační části podrobena důkladnému testování s cílem určit míru její použitelnosti na relačních datech. Pro testování budou mimo určených dat získaných analýzou oblastí stránek použité navíc další dva datové soubory.

Kapitola 5

Vstupní data

V této kapitola budou představeny vstupní datové soubory pro klasifikace. Hlavním zdrojem dat jsou data získané analýzou vizuálních vlastností stránek. Dašimi použitými daty, která jsou určena pro přímé porovnání vlastnostní klasifikační metody, jsou datové soubory NURSERY a ADULT.

Následující část práce se věnuje podrobnému popisu těchto datových souborů, zejména pak atributům záznamů a jejich možným hodnotám.

5.1 Popis dat ke klasifikaci

Data získaná analýzou oblastí jsou uložena v jedné tabulce relační databáze a odpadá tak práce s vytvářením dotazů pro spojování tabulek. Celkem je v datech sledováno 9 atributů oblastí webových stránek, atribut `category` u každého záznamu tabulky udává příslušný typ oblasti stránky.

`fontsize`	průměrná velikost písma v procentech, kde 100% je průměrná velikost písma v celém dokumentu
`weight`	převažující váha písma v oblasti (tučné nebo netučné)
`style`	převažující sklon písma v oblasti (normální nebo skloněné)
`aabove, abelow, aleft, aright`	počet oblastí vyskytujících se nad, pod, vlevo a vpravo od dané oblasti v rámci rodičovské oblasti
`tlength`	počet znaků textu v oblasti
`tdigits, tlower, tupper, tspaces`	počet číslic, malých a velkých písmen abecedy a mezer v textu
`textbtns`	průměrná světelnost (luminosity) textu
`bgbtns`	průměrná světelnost pozadí
`contrast`	průměrný rozdíl světelnosti textu a pozadí

Tabulka 5.1: Popis atributů oblastí testovacích dat ([Kunc, Burget])

Kategorie oblastí byly vytvořené ručně a logicky odpovídají nejčastěji se vyskytujícím

částem webových stránek.

`h1`	nadpis hlavního článku
`h2`	nadpis běžného článku
`h3`	nadpis aktuality nebo zprávy měnšího významu (upoutávky apod.)
`aktualita`	krátká zpráva nebo aktualita
`menu`	navigační oblast
`date`	datum publikování, obvykle i se jménem autora
`none`	ostatní neanotované oblasti

Tabulka 5.2: Vybrané třídy pro klasifikaci testovacích dat([Kunc, Burget])

5.1.1 Format ARFF a jeho převod

Datový formát ARFF(*Attribute-Relation File Format*) je formát textového souboru s daty uloženými ve formě tabulek (jak je tomu obdobně u relačních databází). Formát ARFF je určen zejména pro použití s dolovacím systémem Weka, který také umožňuje exportovat data z ARFF do CSV souboru. Z tohoto souboru jsou následně data do databáze vložena standartním příkazem jazyka SQL.

5.2 Popis dat testovací databáze NURSERY

Databáze NURSERY byla vytvořena z hierarchického rozhodovacího modelu sestrojeného jako zdroj dat pro ohodnocení žadatelů o místo v mateřské školce. Z důvodu velkého zájmu rodičů o umístění dítěte do mateřské školky a kapacitních omezení nebylo možné přijmout všechny žadatele a školka přijímala pouze některé předškoláky. A jak tomu většinou bývá, byla snaha o přijetí "nejvhodnějších" žadatelů, resp. nepřijetí potenciálně problematických dětí (např. z důvodů obavy z infekčních nemocí, neplacení školného atd...). O každém žadateli je v databází vedeno osm atributů a třída (C1 - C5) do které byl žadatel podle hodnot atributů zařazen. Celá databáze má podobu jedné tabulky se všemi potřebnými údaji.

Zvláštností databáze NURSERY je fakt, že domény atributů mají velmi malý počet prvků a navíc jsou tyto atributy nenumerického charakteru - jedná se o řetězce znaků. Proto není nutné provádět diskretizaci numerických atributů a výsledky klasifikace datového souboru NURSERY mohou být použity pro určení vlastností samotné klasifikační metody nezatížené možnými zkreslenými údaji diskretizačního algoritmu.

`parents`	vztahy mezi rodičem a dítětem {`usual, pretentious, great_pret`}
`has_nurs`	zázemí dítěte {`usual, pretentious, great_pret`}
`form`	rodinný stav {`completen, completed, incomplete, foster`}
`children`	počet dětí v rodině {`1, 2, 3, more`}
`housing`	stav bydlení rodiny dítěte{`convenient, less_conv, critical`}
`social`	sociální schopnosti {`non_pron, slightly_prob, problematic`}
`health`	zdravotní vztah dítěte{`recommended, priority, not_recom`}

Tabulka 5.3: Popis atributů datového souboru NURSERY.

Každé kombinaci atributů (data v tabulce zcela pokrývají množinu kombinací atributů) je přiřazena třída ohodnocení žadatele

`not_recom`	dítě s nejmenší šancí na přijetí}
`very_recom`	dítě doporučené na přijetí
`priority`	dítě prioritně doporučené na přijetí
`spec_prior`	dítě s nejvyšší prioritou doporučení

Tabulka 5.4: Kategorie datového souboru NURSERY

Prostor atributů tabulkou kompletně pokrývá kartezský součin domén atributů (s vyjímkou atributu *class*). Pro každou kombinaci hodnot atributů existuje v tabulce záznam, který jí přiřadí jednu z hodnot domény *class*.

5.3 Popis dat testovací databáze ADULT

Data v datovém soubor ADULT obsahují informací pořízené při sčítání obyvatel. Z původní databáze bylo po upravách vybráno několik významných atributů se zaměřením na velikost jejich platu v jednom roce; jako hraniční bod byla určena hodnota 50.000$ ročně.

Každý záznam v tabulce tak obsahoval navíc klasifikující atribut category, který nabývá hodnot z množiny $' <= 50', ' > 50k'$. Mezi sledovanými vlastnostmi nechybí např. pohlaví osoby, věk, vzdělání, rodinný stav apod.

Celkem tabulka s daty obsahuje 15 atributů, 6 z nich je celočíselného typu a je třeba na nich provést diskretizaci (narozdíl od datového souboru NURSERY s nulovým výskytem numerických atributů). Počet záznamů v tabulce je roven 32561.

Klasifikační třídy jsou v případě datového souboru Adult pouze dvě a říkají, zda má žadatel odpovídající jednomu záznamu v tabulce vyšší, či nižší příjem než 50.000$ ročně.

age	věk (číselný atribut)
work_class	{Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked}
fnlwgt	(číselný atribut)
education	nejvyšší dosažené vzdělání {Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool}
education-num	(číselný atribut)
marital-status	rodinný stav{Married-civ-spouse, Divorced, Never-married, Seperated, Widowed, Married-spouse-absent, Married-AF-spouse}
occupation	zaměstnání{Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-speciality, Handlers-cleaners, Machine-op-inpsct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces}
relationshop	{Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried}
race	{White, Asiac-Pac-Islander, Amer-Indian-Eskimo, Other, Black}
sex	{Male, Female}
capital_gain	(číselný atribut)
capital_loss	(číselný atribut)
hours-per-week	(číselný atribut)
native-country	{United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands}

Tabulka 5.5: Popis atributů datového souboru ADULT

<=50K$
>50K$

Tabulka 5.6: Kategorie datovéhou souboru ADULT

Kapitola 6

Implementace klasifikátoru

Klasifikátor byl implementovaný v programovacím jazyce JAVA 1.5 s využitím JDBC konektoru pro připojení k databázi MySQL. Z logického pohledu je projekt rozdělený do čtyř balíčků.

Hlavním balíčkem projektu je balíček `textclassifier`, ten pak obsahuje všechny ostatní podbalíčky `database, mining, ArcBC` a `discretization`.

- balíček `database` se stará o zajištění správné komunikace aplikace a databáze
- balíček `discretization` provádí diskretizaci dat a správu datových struktur spojených s diskretizací
- balíček `mining` obsahuje třídy struktur dolovacích algoritmů
- balíček `ArcBC` zastřešuje třídy klasifikačního algoritmu ARC-BC

V průběhu implementace bylo potřeba provést značnou část úprav klasifikačního algoritmu ARC-BC pro adaptaci z textových dat na data relační. Jedná se zejména o diskretizaci numerických hodnot. U textového klasifikátoru je diskretizace ireleventní, u řady numerických atributů je naopak důležitým prostředkem pro správný popis dat.

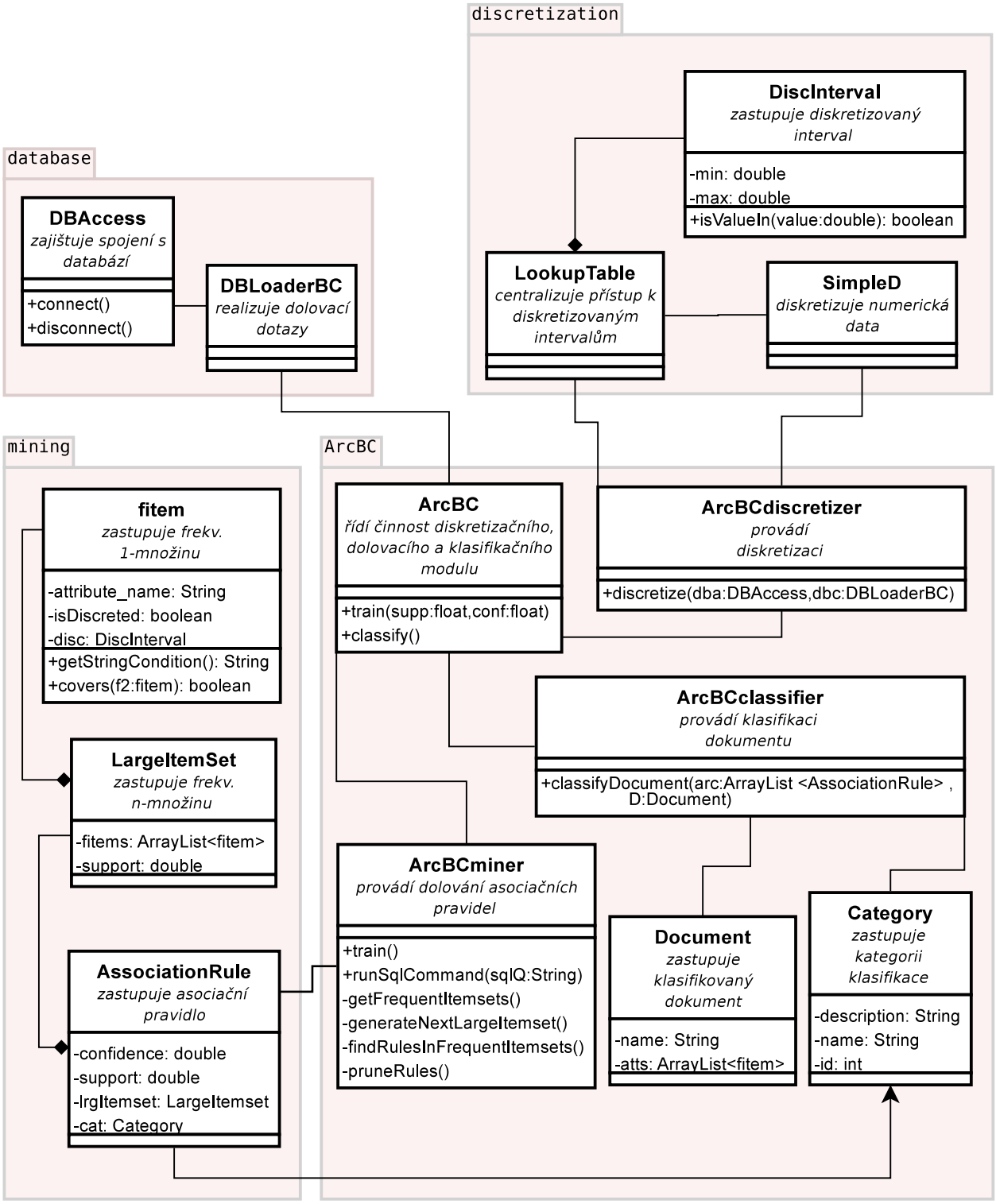
Stejně tak byly objeveny některé nejasnosti související s výpočtem spolehlivosti vydolovaných asociačních pravidel.

Tato kapitola se snaží představit a popsat průběh implementace systému a řešení problémů, které v souvislosti s přechodem textová data → relační data vznikaly.

6.1 Celkový pohled

Implementace systému byla zahájena vytvořením základních tříd balíčku `mining`, které byly potřebné pro správnou funkci klasifikátoru ARC-BC. Jednalo se zejména o třídy `fitem` a `LargeItemSet`. Poté proběhla implementace samotné metody ARC-BC. Kvůli poměrně velké škále funkcí nutných pro implementaci jsem rozdělil implementační proces do těchto fází:

1. Implementace dolování frekventovaných množin
2. Implementace generování a prořezávání asociačních pravidel
3. Implementace klasifikace dokumentů
4. Implementace diskretizace



Obrázek 6.1: Diagram tříd

6.1.1 Implementace dolování frekventovanách množin

Dolování frekventovaných množin v datech je časově nejnáročnější částí programu. Charakteristickým rysem této části je častá komunikace s datovým zdrojem přes přostředníka ve formě třídy `DBLoaderBC`.

Na prvním místě v seznamu postupných prací při dolování frekventovaných množin je získání seznamu atributů. Jejich výčet je uvedený v konkrétní implementaci rozhraní `DBLoaderBC` a určuje atributy, jejichž hodnoty budou použité pro ohodnocení dokumentu. Následující kroky jsou pak prováděné pro všechny kategorie zvlášť (dokumenty spadající do jedné kategorie jsou chápané jako oddělená trénovací množina).

V datech dokumentů kategorie Cat_i se hledají frekventované 1-množiny (prvků) nad vybranými atributy. Tato operace je realizována SQL dotazem, jež vybere všechny hodnoty atributů, které jsou v datech s vyšší než minimální podporou *min_supp*. Množina takto nalezených frekv. jednoprvkových množin je použita pro generování frekventovaných množin o vyšším počtu prvků. Frekv. 2-množiny vzniknou spojením spojením dvou frekv. 1-množin, frekv. 3-množiny spojením frekv. 2-množiny a frekv. 1-množiny atd... Vzhledem k tomu, že Java poskytuje pro reprezentaci množiny třídu `HashSet`, dají se tyto činnosti provádět jednoduchým použitím tradičních množinových operací. Testování na splnění minimální podpory je, stejně jako u frekv. 1-množin, realizované vhodným SQL dotazem.

6.1.2 Implementace generování asociačních pravidel

Fáze generování končí tehdy, když je nově nalezené množina frekv. n-množin prázdná. Poté nastupuje na řadu generování asociačních pravidel z frekventovaných množin. K tomu je použitý klasický postup na generování asociačních pravidel viz.[11]. Minimální spolehlivost asociačních pravidel je ověřovaná opět příslušným SQL dotazem: pro každou frekv. množinu se vytvoří dotaz na počet dokumentů splňujících podmínku vytvořenou "atribut = hodnota AND atribut = hodnota AND ...".

Přořezání asociačních pravidel se provádí průchodem seznamu asociačních pravidel a odstraňováním méně obecných pravidel podle postupu viz.[11] Výstupem fáze generování asociačních pravidel je seznam `ArrayList` asociačních pravidle `AssociationRule`.

6.1.3 Implementace klasifikace dokumentů

Přechozí fáze zakončila trénování klasifikátoru. Klasifikátor získal povědomí o datech a vytvořil seznam asociačních pravidel. Klasifikace neznámého dokumentu probíhá tak, že se prochází seznamem všech pravidel a zkouší se, zda pravidlo R pokrývá dokument D, tj. jestli hodnoty atributů dokumentu D odpovídají hodnotám frekv. 1-množin levé strany asociačního pravidla (v případě diskretizovaných atributů musí platit, že hodnota h_d atributu a_d dokumentu D musí ležet v intervalu I_d příslušného atributu a_r asociačního pravidla R).

Po otestování celé sady asociačních pravidel známe ta pravidla, která pokrývají dokument D. Následujícím krokem je rozdělení pravidel do skupin podle kategorie na pravé straně těchto pravidel. Pokud tak vznikne pouze jedna skupina $Group_{Cat_i}$, klasifikátor zařadil D jednoznačně do kategorie Cat_i. V případě více skupin klasifikátor provedl vícenásobnou klasifikaci a skupiny se seřadí podle průměru součtu spolehlivosti asociačních pravidel ve skupině.

Příklad
Pro prořezání zůstala tři asosiační pravidla

$$R_1 : A_1 \rightarrow Cat_5(conf = 0.9),$$
$$R_2 : A_2 \rightarrow Cat_8(conf = 0.85),$$
$$R_1 : A_3 \rightarrow Cat_5(conf = 0.7).$$

Jsou vytvořeny celkem 2 kategorie $Group_{Cat_5}$ a $Group_{Cat_8}$. Průměr spolehlivost pravidel skupiny Cat_5 $m_{Group_{Cat_5}} = (0.9 + 0.7)/2 = 0.8$, u druhé skupiny Cat_8 nabývá hodnoty $m_{Group_{Cat_8}} = 0.85$. Klasifikátor upřednostní skupinu Cat_8 a výsledkem klasifikace bude seřazený seznam ($Cat_8(0.85)$, $Cat_5(0.8)$).

6.1.4 Implementace diskretizace

V druhé části projektu byla doimplementovaná diskretizace.

Ne vždy je nutné diskretizovat všechny numerické atributy, nejdříve je třeba určit, které z nich mají podléhat diskretizaci. Nastavení se provádí úpravou návratové hodnoty metody `getDiscretedAttributes` v implementaci rohzraní `DBLoaderBC`. Pro každý takový atribut se pak získá seřezený seznam všech hodnot, který se jako parametr předa metodě třídy *SimpleD*, jenž zpracuje číselná data a vrátí seznam diskretizovaných intervalů. Tato akce proběhne pro všechny vybrané atributy.

Přidání diskretizace s sebou přineslo řadu změn a úprav stávajících struktur. Jedním z hlavních problémů bylo, jak "zapasovat" diskretizované intervaly do systému tak, aby se s nimi dalo transparentně pracovat jako s frekventovanými 1-množinami.

Jeden konkrétní diskretizovaný interval atributu A_i nahrazuje obecně jednu konkrétní hodnotu atributu A_i, tudiž na něj můžeme pohlížet jako na zvláštní případ výskytu objektu `fitem`. Do objektu `fitem` byla přidaná reference na objekt třídy `DiscInterval`. Pokud `fitem` zastupoval diskretizovaný interval, potom reference odkazovala právě na konkrétní objekt `DiscInterval`, v jiném případě nabývala hodnoty `null`.

Při vytváření frekventovaných n-množin se potom na diskretizovaný interval nahlíží stejně jako na frekventovanou 1-množinu.

6.2 Balíček `database`

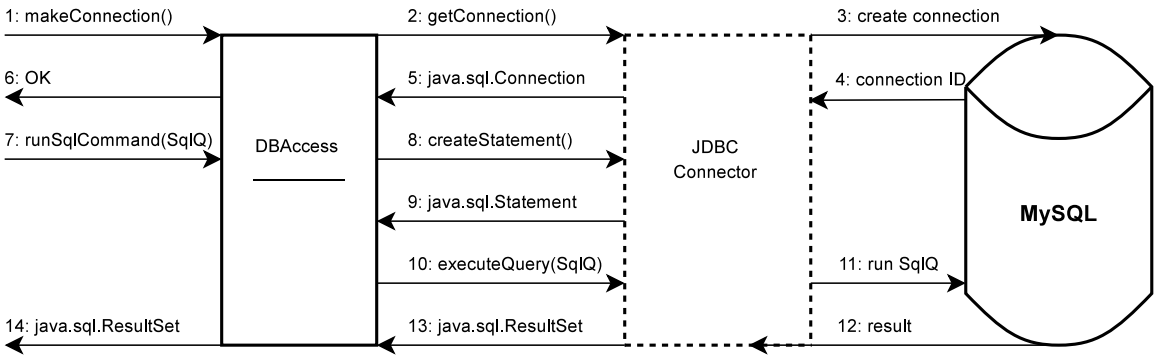
V balíčku database jsou třídy umožňující komunikující s databází. Třída DBAccess je spojující třídou a obsahuje metody pro návázání spojení s databází, realizuje dotazy a vrací výsledky dotazů.

Třída DBLoaderBC využívá spojení DBAccess pro pokládání dotazů pro potřeby algoritmu ARC-BC, jako jsou dotazy pro zjištění frekventovaných množin, dotazy ověřující spolehlivost frekventovaných množin atd... Komunikační kanál je vytvořen JDBC konektorem, který také udržuje spojení `Connection` se samotnou databází a provádí realizaci SQL dotazů reagováním na zavolání metody `executeQuery(String sqlQ)`.

6.2.1 Třída `DBAccess`

Jak již bylo řečeno, třída DBAccess je určená pro přístup k datovým zdrojům v databázi, k čemuž využívá stávající prostředky jazyka Java v podobě JDBC konektoru.

Důležité metody třídy:



Obrázek 6.2: Diagram komunikace mezi třídou DBAccess a databází

- `DBAccess(String server, String db, String user_name, String pass)` vytvoří nový objekt třídy DBAccess
- `boolean makeConnection()` vytvoří nové spojení s databází
- `ResultSet runSqlCommand(String sqlQ)` položí SQL dotaz na databázi s použitím stávajícího připojení; v případě chybějícího spojení je vyvolána vyjímka

6.2.2 Třída `DBLoaderBC`

Rozhraní `DBLoaderBC` slouží jako vzor k realizaci dotazů konkrétních úkolů dolovacího algoritmu metody ARC-BC. Třídá vytváří dotazy typu "*Urči v kolika kategoriích se frekventovaně vyskytuje dokument s vlastnostmi XY.*".

```
public interface DBLoaderBC {
  public String getCategories();
  ...
}
```

6.3 Balíček `ARC-BC`

V balíčku `ARCBC` jsou seskupené třídy klasifikační metody ARC-BC. Třída `ArcBC` je třídou, která řídí další tři podtřídy `ArcBCminer`, `ArcBCdiscretizer` a `ArcBCclassifier`.

6.3.1 Třída `ArcBC`

Konstruktor třídy `ArcBC` vyžaduje přístup k databázi a k třídě implementující rozhraní `DBLoaderBC`. Po vytvoření instance třídy můžeme volat metody `train`, resp. `classify` pro trénování klasifikátoru, resp. klasifikaci dokumentu.

Důležité metody třídy:

- `ArcBC(DBLoaderBC dbl, DBAccess dba)` konstruktor vytvoří novou instanci třídy
- `ArrayList<AssociationRule> train(float supp, float conf)` provede natrénování klasifikátoru

- `int classifyDocument(<AssociationRule> ARC, Document D, double conf_thresh` provede klasifikaci dokumentu

6.3.2 Třída `ArcBCdiscretizer`

Třída `ArcBCdiscretizer` slouží pro diskretizaci numerických dat. Vstupem pro modul diskretizéru jsou datové zdroje `DBAccess` a `DBLoaderBC`, výstupem je tabulka diskretizovaných intervalů `LookupTable`.

Důležité metody třídy:

- `LookUpTable discretize(DBAccess dba, DBLoaderBC dbl` diskretizuje numerické atributy

6.3.3 Třída `ArcBCminer`

Třída `ArcBCminer` implementuje dolovací modul klasifikátoru ARC-BC. Po správnou činnost vyžaduje množinu hodnot diskretizovaných atributů (výsledků činnosti třídy `ArcBCdiscretizer`. Třída obsahuje důležité metody pro dolování a generování frekventovaných množin, pro vytvoření asociačních pravidel z frekventovaných množin a také pro prořezání asociačních pravidel. Návratová hodnota metody `train` je seznam nalezených asociačních pravidel.

Důležité metody třídy:

- `ArrayList<AssociationRule> train(LookUpTable DiscTable, double min_supp, double min_conf` provádí dolování (*mining*) a prořezání(*pruning*) asociačních pravidel

6.3.4 Třída `ArcBCclassifier`

Klasifikáční část má na starosti třída `ArcBCclassifier`, jejíž hlavní funkci plní metoda `classify`. Návratovou hodnotou metody je seřazený seznam kategorií, do kterých byl dokument zařazen.

6.3.5 Třída `Category`

Pro reprezentaci kategorie dokumentů slouží objekty třídy `Category`. Jedná se o jednoduchou strukturu, která modeluje třídu za pomocí jednoznačného identifikátoru (`id`), názvu (`name`) a slovnímu popisu (`description`) kategorie C. Objekty třídy `Category` se vyskytují na pravé straně vydolovaných asociačních pravidel $R : A \rightarrow Cat_i$, které jsou reprezentovány objekty třídy `AssociationRule`.

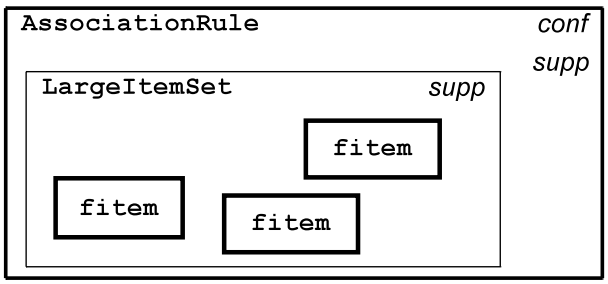
6.3.6 Třída `Document`

Třída `Document` reprezentuje dokumenty, které jsou vstupem do procesu klasifikace. Struktura objektu `Document` je tvořena především množinou dvojic atribut-hodnota; tato množina popisuje dokument a podle jejich prvků je prováděna klasifikace pokrýváním asociačnimi pravidly.

6.4 Balíček `mining`

V balíčku mining jsou umístěné třídy potřebné pro získání a uchování výsledků dolování z dat. Celkem obsahuje 3 třídy: `fitem`, `LargeItemSet` a `AssociationRule`. Obecně mohou

třídy balíčku sloužít pro dolování jakéhokoliv typu dat, pro účely klasifikace dokumentů jsou specializované pro dolování asociačních pravidel, která mají na pravé straně kategorii Cat_i.



Obrázek 6.3: Vztahy v balíčku mining

6.4.1 Třída `fitem`

Objekty `fitem` jsou odrazem frekventovaného prvku v datech. Jejich základní funkcí je uchovávat dvojice {atribut, hodnota}. Hodnota atributů může být různého typu(double, String, int), v případě diskretizovaného numerického atributu objekt uchovává informace o diskretizovaném intervalu, který zastupuje rozmezí příslušných hodnot.

6.4.2 Třída `LargeItemSet`

Objekty třídy `LargeItemSet` popisují frekventované n-množiny, které vznikají seskupením několika objektů třídy `fitem`. Tak mohou vznikat například frekventované 1-množiny (přidáním jednoho objektu `fitem`, ze kterých jsou generovány 2-množiny (a dále obecně i+1-množiny).

Nejdůležitější datovou strukturou třídy je množina `HashSet` objektů třídy `fitem` a také hodnota podpory *supp* frekventované množiny.

6.4.3 Třída `Association Rule`

Třída `Association Rule` zastupuje asociační pravidla nalezené v datech. Objekty třídy `Association Rule` uchovávají informace o spolehlivosti a podpoře asociačních pravidel $R : A \rightarrow Cat_i$ tvořených frekventovanou n-množinou výrazů A na levé straně a kategorií Cat_i na straně pravé.

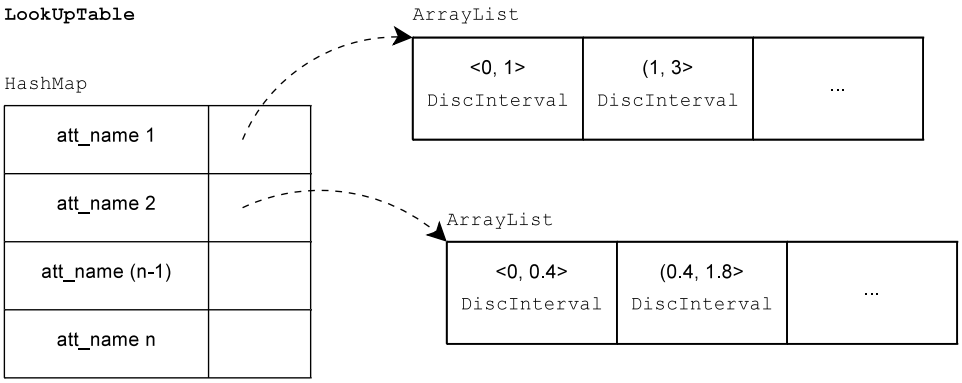
6.5 Balíček `discretization`

Třídy balíčku `discretization` slouží pro diskretizaci numerických atributů, a to metodou diskretizace do šířky. Diskretizační metoda do šířky je zastoupena třídou `simpleD`, která obsahuje potřebné metody pro samotný proces diskretizace. Diskretizované intervaly jsou reprezentované objekty třídy `discInterval`; objekty nesou informaci o velikosti a ohraničení intervalů a také metody pro manipulaci s těmito intervaly.

Po dokončení procesu diskretizace jsou výsledky ve formě kolekce diskretizovaných intervalů uložené v třídě lookupTable, která pak slouží jako centrální evidenční uzel pro práci s intervaly.

6.5.1 Třída `LookupTable`

Udržování konzistence intervalů všech diskretizovaných atributů v datech může být při velkém počtu atributů obtížné. Třída `LookUpTable` slouží právě pro jednoduchý centralizovaný přístup k diskretizovaným intervalům.



Obrázek 6.4: Uložení intervalů v tabulce

Informace jsou uložené v datové struktuře `HashMap`, která je v jazyce Java obdobou asociovaného pole. Tato `HashMap` asociuje jméno diskretizovaného atributu se seřazeným seznamem(`ArrayList`) diskretizovaných intervalů daného atributu.

6.5.2 Třída `SimpleD`

Třída `SimpleD` implementuje diskretizační metodu do šířky. Obsahuje veřejnou statickou metodu `process`, která pro zadanou numerickou řadu provede diskretizaci do určeného počtu intervalů s daným rozsahem.

Důležité metody třídy:

- `ArrayList<DiscInterval> process(double[] numbers, int ivals, int ivalsize)`

Po provedení diskretizace je vytvořený seznam všech nalezených intervalů `DiscInterval`, který se nastaví jako návratová hodnota metody.

6.5.3 Třída `DiscInterval`

Třída `DiscInterval` zastupuje jeden diskretizovaný interval atributu. Vyznačuje se především hodnotami určujícími počáteční a konečnou hodnotu intervalu a odkazem na atribut, se kterým je svázaný.

6.6 Načítání vstupních dat

Nastavení zdroje vstupních dat pro klasifikaci není v projektu řešeno "klasickým" způsobem - tj. konfiguračním souborem, nýbrž třídou implementující rozhraní. Výhodou takového řešení je to, že umožnuje pracovat s různorodými zdroji dat a poskytuje flexibilitu při změně vstupních dat. Bylo proto navrženo rozhraní `DBLoaderBC` definující sadu metod potřebnou pro zajištění všech potřebných dat ve fázi dolování asociačních pravidel algoritmu ARC-BC.

6.7 Problémy při implementaci

6.7.1 Výpočet spolehlivost asociačních pravidel

Jak bylo uvedeno v popisu metody ARC-BC, umožňuje tato provádět vícenásobnou klasifikaci, tj. zařadit dokument do více než jedné třídy. V dokumentu [1] nebylo možné jednoznačně zjistit popis výpočtu spolehlivosti *conf* asociačního pravidla R, což se v průběhu implementace ukázalo být vážným problémem.

Mějme asociační pravidlo $R : A \rightarrow B$, které má v kontextu klasifikace dokumentů tvar

$$t_1 \wedge t_2 \wedge t_3 \wedge \ldots \wedge t_n \rightarrow Cat,$$

kde t_i jsou jistá pravdivá tvrzení o dokumentu D a Cat kategorie, do které je dokument splňující všechna tvrzení na levé straně pravidla zařazený. Za této situace je spolehlivost pravidla R rovna

$$conf_R = \frac{supp(t_1 \wedge t_2 \wedge t_3 \wedge ... \wedge t_n)}{supp(t_1 \wedge t_2 \wedge t_3 \wedge \ldots \wedge t_n \wedge Cat)},$$

což nečiní problémy při variantě algoritmu dolujícího pravidla nad kompletní množinou testovacích dat a hodnota *conf* bude v rozmezí $0 \ldots 1$.

Metoda ARC-BC ovšem doluje pravidla po kategoriích (*By Category*), tedy pokládá každou kategorii trénovacích dat za samostatnou část a nakládá s ní jako s celkem. Výpočet, který pro předchozí variantu fungoval, nyní selhává, neboť term *Cat* (který říká, že dokument patří do kategorie *Cat*) je vždy pravdivý - všechny dokumenty trénovací množiny totiž patří do jedné kategorie.

Výsledkem jsou asociační pravidla, která mají spolehlivost vždy $conf = 1.0$. Tím se ale zásadně snižuje vypovídací schopnost asociačních pravidel. Navíc se dostávají na povrch další problémy v klasifikační fázi dokumentů, které přímo pracují s hodnotou spolehlivosti asociačních pravidel.

Bezpodmínečně bylo nutné určit výpočet spolehlivosti asociačních pravidel jednak aby korespondoval s relevantností pravidla nad množinou trénovacích dat, jednak aby nebyl narušen princip samotné metody.

Navrhl jsem dva odlišné způsoby výpočty spolehlivosti a testoval jsem jejich vliv na kvalitu asociačních pravidel, resp. samotné klasifikace.

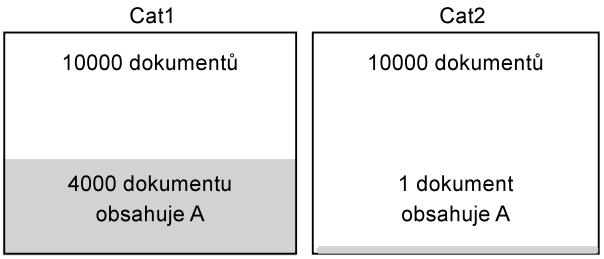
Poměrný výpočet spolehlivosti

Nejprve jsem vyzkoušel vypočítat spolehlivost jednoduše

$$conf_{(A \rightarrow Cat_c)} = \frac{1}{N},$$

kde N je počet kategorií, pro které je spolněn předpoklad A. Pro každé asociační pravidlo se určil počet kategorií N , ve kterých byla splněna množina termů A a výsledná spolehlivost byla vyhodnocena jako obrácená hodnota N.

Pokud tedy pravidlo R pokrývá n kategorií($n \leq N$), vytvoří se celkem n pravidel se spolehlivostí $1/n$, tedy např. pro $n = (1 \ldots 4)$ $conf = \{1.0, 0.5, 0.25, 0.125\}$. Pokud by počet kategorií pokrytých pravidlem R byl vyšší než 4, spolehlivost by dále konvergovala až k nule. Ve většině případů volíme minimální spolehlivost relevantních pravidel od $< 0.51.0 >$, tudiž je patrné, že při takovém způsobu výpočtu by byla vygenerována asociační pravidla se spolehlivostí 1.0 nebo 0.5. Názorně lze problém ilustrovat na následujícím příkladě. Nechť



Obrázek 6.5: Příklad problému s při poměrném výpočtu spolehlivosti

jsou dvě kategorie Cat_1 a Cat_2, do každé kategorie patří 10.000 dokumentů. Dolovací algoritmus nalezl při analýze kategorie Cat_1 frekventovanou množinu A.

Při vyhodnocení spolehlivosti asociačního pravidla $A \rightarrow Cat_1$ bylo zjištěno, že A je obsaženo také v kategorii Cat_2. Výsledná spolehlivost spolehlivost pravidla $conf_{A \rightarrow Cat_1} = 0.5$ a samotné pravidlo říká, že s 50%ní pravděpodobností bude dokument spňující předpoklad A patřit do kategorie Cat_1, což je údaj značně zkreslující údaj vzhledem k tomu, že v kategorii Cat_2 je A splněno pouze v jednom dokumentu z celkových 10.000.

Ještě větší zkreslení by přinesla situace, kdy by existovalo více kategorii podobných kategorii Cat_2; potom by spolehlivost bezdůvodně prudce klesala. V přímém nasazení se ukázalo být použití poměrného výpočtu spolehlivosti nevhodným, proto jsem nakonec použil druhý způsob - procentuální výpočet spolehlivosti.

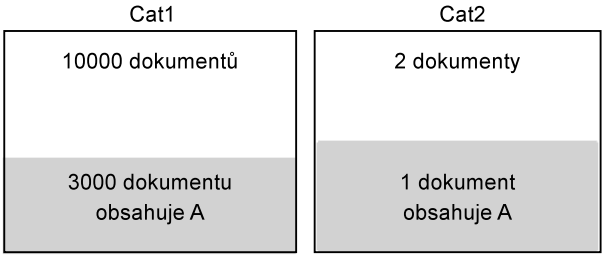
Procentuální výpočet spolehlivosti

Tato metoda určení spolehlivosti asociačního pravidla odstraňuje neduhy metody přechozí a snaží se tak spolehlivost určit co nejpřesněji v souvislosti se zdrojovými daty. Vzorec pro výpočet jsem navrhl tak, aby bral v potaz míru výskytu levé strany pravidla A v celé kategorii:

$$conf_{(A \rightarrow Cat_c)} = \frac{supp(A \rightarrow Cat_c)}{\sum_{i=0}^{i=N} supp(A \rightarrow Cat_i)},$$

kde N je počet kategorií, pro které je spolněn předpoklad A.

Spolehlivost asociačních pravidel při použití výše uvedeného vzorce je dostatečně přesná, jediný problém může nastat u kategorií s extrémně rozdílným počtem dokumentů(vzorů obecně). Takovou situaci ilustruje následující obrázek.



Obrázek 6.6: Příklad problému při procentuálním výpočtu spolehlivosti

Pravidlo $A \rightarrow Cat_2$ má vyšší spolehlivost než pravidlo $A \rightarrow Cat_1$, což povede k zajímavému nežadoucímu jevu, kdy může být všech 3000 dokumentů z Cat_1 nesprávně zařazeno do kategorie Cat_2.

Celkem mohou nastat dva druhy případů, kdy se bude velikost kategorií zásadně lišit:

- Případ 1 - Nedostatečné zajištění potřebného množství trénovacích dat způsobí rozdíly ve velikostech kategorií. To je zapříčiněné špatnou volbou trénovacích dat, případně nepochopením dat jako celku.
- Případ 2 - Malé množství dokumentů v kategorii je způsobené samotnými vlastnostmi vstupních dat a odpovídá modelované realitě. V takovém případě jsou rozdíly ve velikostech kategorií v pořádku.

Proto není nutné se až tak příliš zabývat chybami, které mohou být způsobené procentuálním výpočtem spolehlivosti, je ale potřebá mít dokonalý přehled o vstupních datech (Případ 1). Tento způsob výpočtu se nakonec ukázal být dostatečným pro použití v klasifikátoru.

6.7.2 Diskretizace numerických atributů

Zvláštním rysem většiny numerických atributů je fakt, že mohou nabývat obrovského množství hodnot, což je při analýzách zdrojem nežadoucích problémů. Pro zjednodušení manipulace s takovými numerickými atributy používáme techniky *diskretizace* - nahrazení hodnot numerického atributu intervaly hodnot I. Každá číselná hodnota c potom spadá právě do jednoho takového intervalu I_i.

Původní je metoda ARC-BC určaná pro textová data, tedy data, kde o diskretizaci nemá smysl mluvit. Pokud měla být metoda použitá i pro klasifikaci dat relačních, bylo nutné nějakým způsobem provádět také diskretizaci.

Nejjednoduší dva typy diskretizace jsou diskretizace do šířky a diskretizace do hloubky.

Diskretizace do šířky(*Equal-width discretization*)

Diskretizace do šířky je nejjednodušší formou diskretizace numerických dat. Numerická řada se rozdělí na n intervalů I, každý koš pak odpovídá právě jednomu z těchto intervalů. Číselné hodnoty jsou následně přiřazovány do košů podle toho, zda hodnota patří do příslušného intervalu I_i.

$$\{1, 1, 1, 1, 2, 4, 5, 33, 68, 69, 70, 121\}$$
$$n = 3$$
$$s = (120 - 1)/3 = 40$$
$$I_1 =< 1, 41), I_2 =< 41, 81), I_3 =< 81, 121 >$$

Zásadním problémem diskretizace do šířky je nerovnoměrné rozložení hodnot do košu. Nezřídka nastává situace, kdy několik košu zůstává téměř prázdných a naopak jeden koš pokrývá většinu hodnot numerického atributu. K tomuto jevu dochází v případě nerovněměrně rozložených číselných hodnot. Diskretizace do šířky je mimo to citlivá na vychýlené hodnoty.

Diskretizace do hloubky(*Equal-frequency discretization*)

V případě diskretizace do hloubky jsou data numerické hodnoty rozdělené do košů, které obsahují zhruba stejný počet prvků. Díky tomuto postupu nemůže dojít ke stejnému problému jako u diskretizace do šířky.

$$\{1, 1, 1, 1, 2, 4, 5, 33, 68, 69, 70, 121\}$$
$$B_1 = [1, 1, 1, 1], B_2 = [2, 4, 5, 33], B_3 = [68, 69, 70, 121]$$
$$I_1 =< 1, 1 >, I_2 =< 2, 33 >, I_3 =< 68, 121 >$$

Často se stává, že několik sousedních košů je zaplněno stejnými hodnotami. Za takové situace může být problém určit, který z košu použít a do kterého příslušnou hodnotu vložit.

Popsaný nedeterminismus se jednoduše řeší tzv. *shlukováním košu*. Sekvenčně se kontrolují sousední koše a pokud se narazí na takové, jejihž interval I_i a I_{i+1} je stejný, koš I_{i+1} je odstraněn. Stejně tak často nastává situace, kdy dva sousední intervaly nejsou zcela disjunktní (což je podmínka pro zachování determinismu při určování koše).

$$B_1 = [1, 1, 1], B_2 = [1, 2, 4], B_3 = [5, 33, 68], B_4 = [69, 70, 120]$$

Zde nastal problém s hodnotou 1 zapadající do intervalů košů B_1 a B_2. Proto je nutné vybrat ze dvou pokrývajících košů koš hlavní, který bude hodnotu zastupovat. Z druhého koše bude tato hodnota vyjmuta, což povede ke zmenšení velikosti intervalu na straně koše s odebraným prvkem(prvky). Výběr hlavního koše je možné provézt stochasticky, případně použit některou z heuristických metod.

6.8 Shrnutí implementace

V rámci implementace se podařilo upravit klasifikační metodu ARC-BC pro dolování na relačních datech. Hlavní změny spočívaly zejména v modifikaci postupu pro dolování frekventovaných množin a v začlenění diskretizace numerických atributů. Součástí úprav bylo také určení vhodného výpočtu spolehlivosti asociačních pravidel, které klasifikační metoda vygenerovala ve fázi dolovaní frekventovaných množin.

Celá implementace asociačního klasifikátoru ARC-BC je rozdělena do balíčků podle funkce tříd do nich spadajících. Samostatný balíček `database` sdružuje třídy pro práci s databazí MySQL, balíček `mining` obecné třídy dolovacích struktur, balíček `ARC-BC` třídy klasifikačních algoritmu, balíček `discretization` pak třídy pro diskretizaci.

Kapitola 7

Testování

Po implementaci metody proběhlo testování na připravených testovacích datech tvořených datovými soubory Nursery, Adult, a konečně ostrými daty z analyzátoru vizuálních vlastností. Pro jednotlivá běhy testů na datech je možné nastavit několik významných parametrů, které mají vliv na výsledky klasifikace:

1. *min_supp* minimální podporu asociačních pravidel
2. *min_conf* minimální spolehlivost asociačních pravidel
3. *disc_coef* koeficient rozsahu diskretizovaného intevalu

V případě *min_supp* platí, že čím více se hodnota blíží k 1.0, tím obecnější pravidla jsou nalezány. Naopak, pokud se podpora blíží k hodnotě 0.0, jsou v datech nalezené i pravidla, která jsou více specializovaná a která by v případě vyšší hodnoty *min_supp* klasifikátor vůbec nenašel. V ideální situaci by se měla být hodnota *min_supp* limitně blížit k nule, ale v reálu pak dochází k prohledávání neúměrně velkého prostoru frekventovaných množin a také přílišná velikost množiny výsledných asociačních pravidel přináší značný nárust potřebného výpočetního času.

Hodnota *min_conf* udává míru pravdivosti asociačního pravidla. Čím více se hodnota blíží k 1.0, jsou hledána více přesná a data lépe popisující pravidla. Takových pravidel je ovšem minimální množství, proto je třeba experimentovat s hodnotou min_conf, aby nedošlo k nechtěnému potlačení asociačních pravidel a následnému jevu, kdy by zůstalo velké množství dat neklasifikovaných, protože by prostě neexistovala žádná pravidla, která by měla vyšší hodnotu spolehlivosti.

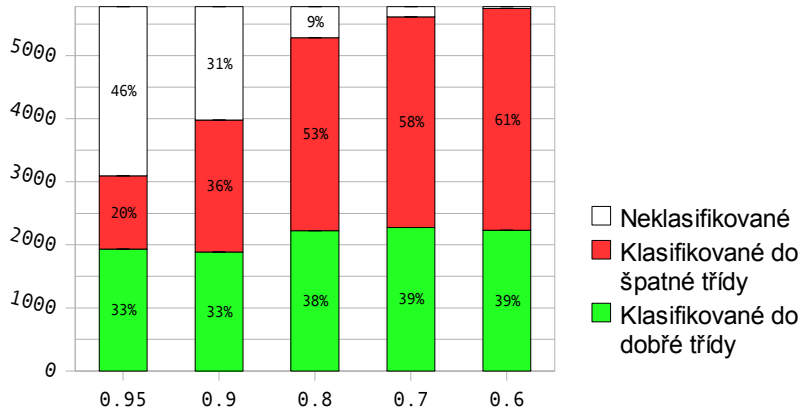
Poslední parametr *disc_coef* ovlivňuje počtu diskretizovaných intervalů. Samotný výpočet velikosti je odvozen od pravidla pro výpočet histogramů. Se zvyšující se hodnotou *disc_coef* vzrůstá počet intervalů (čímž se snižuje jejich velikost) pro jeden diskretizovaný atribut.

V následující testech jsem provedl experimenty s různým nastavením výše uvedených parametrů a pozoroval jsem chování a výsledky metody ARC-BC.

7.1 Ostrá data z webu

Vstupní data extrahovaná z webových stránek byla podrobena důkladným testům s různými počátečními parametry. Pro případné porovnání s jinými klasifikačními metodami v testech sleduji především metriky *missclassification-rate* a *precision*.

Obrázek 1 zobrazuje výsledky klasifikace při pevně nastavené minimální podpoře *min_supp* = 0.05 a různě vysokých hodnotách spolehlivosti *min_conf*. Zeleně zbarvený fragment sloupce grafu udává množství dokumentů, které byly klasifikovány do správné třídy, červenou barvou je vyznačena množina dokumentů, které klasifikátor chybně zařadil do jiné kategorie, bílá část sloupce nakonec označuje dokumenty, které klasifikátor nezařadil do žádné z tříd.



Obrázek 7.1: Zastoupení neklasifikovaných, správně klasifikovaných a neklasifikovaných dokumentů při konstantní hodnotě podpory *min_supp*=0.05 a různých hodnotách spolehlivosti(osa x).

Z grafu je patrné, že se snižující se hodnotou spolehlivosti se postupně zvyšuje celkové množství klasifikovaných dokumentů. Když byla spolehlivost *min_conf* rovna 0.95, o 46%-ti všech testovaných dokumentů nebyl klasifikátor schopen rozhodnout (ať už správně, či chybně).

Co se týče přesnosti klasifikace, v nejlepším nalezené konfiguraci vstupních parametrů se podařilo dosáhnout přesnosti pouze 40%, ve většině jiných konfigurací pak nabývala hodnot okolo 35%. Ideální nalezená konfigurace byla $min\_supp = 0.05$, $min\_conf = 0.68$, v jiných případech docházelo buď k přetrénování (velké množství nespolehlivých pravidel), nebo naopak k nedostatečnému natrénování, kdy omezené množství asociačních pravidel způsobilo to, že mnoho dokumentů zůstalo neklasifikovaných (viz. Graf 1 první sloupec).

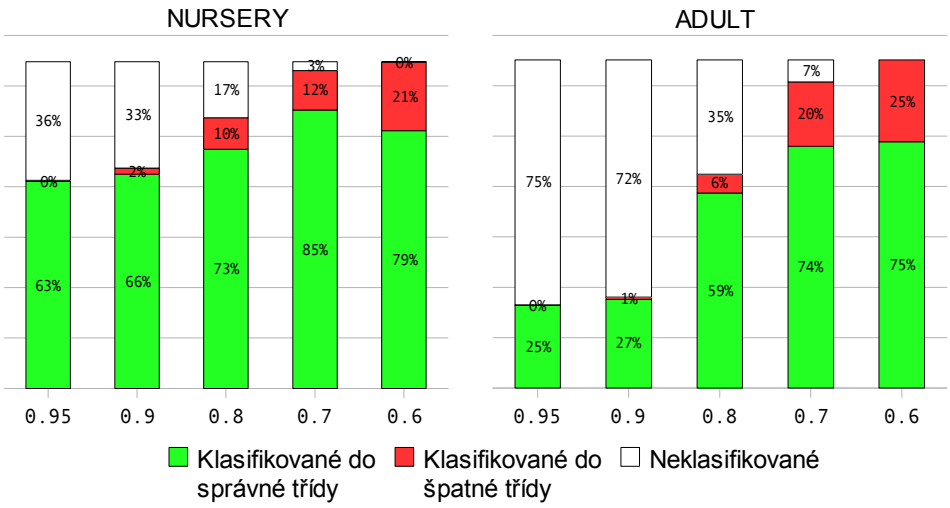
Lepších výsledků se nepodařilo dosáhnout ani s různými hodnotami *min_supp* (testovány hodnoty $0.05 - 0.21$) a *min_conf* $(0.60 - 0.95)$, ani s různým nastavením velikosti diskretizovaných intervalů. Zajímavé chování algoritmu na vstupních datech lze z pozorovat také v Grafu 1, kdy i při snižující se nastavené hodnotě spolehlivosti nedochází k výraznému zvýšení množství správně klasifikovaných dokumentů.

Celkově se dají výsledky klasifikace zhodnotit jako velmi špatné, přesnost klasifikace kolem 40% není dostatečná pro reálné nasazení. Pro zjištění důvodů špatných výsledků bylo provedeno testování na dalších souborech dat - datovém souboru NURSERY a datovém souboru ADULT.

7.2 Datový soubor NURSERY

Klasifikace dat z datového souboru NURSERY měla ukázat, zda nevznikla přo implementaci chyba zapříčiňující nízkou přesnost klasifikace. Počáteční nastavení parametrů klasifikace bylo stejné jako u ostrých dat (z důvodu porovnání výsledků). Levý graf v Obrázku 2

zachycuje průběh změny přesnosti klasifikace při pevně nastavené podpoře *min_supp* = 0.05. Zde je na první pohled vidět rozdíl od dat vizuálních vlastností. Správně klasifikované dokumenty tvořili v nejlepším případě až 85% všech testovaných dokumentů. Přesnost klasifikace neklesla téměř nikdy pod 39%; v datech bylo vydolováno několik vysoce spolehlivých asociačních pravidel s vysokou podporou, které držely výsledky kvalitní až po podporu *min_supp* = 0.25.



Obrázek 7.2: Zastoupení neklasifikovaných, správně klasifikovaných a neklasifikovaných dokumentů při konstantní hodnotě podpory *min_supp*=0.05 a různých hodnotách spolehlivosti(osa x) pro datové soubory NURSERY a ADULT.

7.3 Datový soubor ADULT

Stejně jako u dat. souboru NURSERY, i v případě ADULT bylo testování provedeno za stejných podmínek. Výsledky klasifikace při různých nastaveních spolehlivosti jsou zobrazené v pravé části Obrázku 2. V porovnání s výsledky klasifikace dat vizuálních vlastností stránky se podařilo při ideálním nastavení (*min_supp* = 0.05, *min_conf* = *0.68* dosáhnout přesnosti klasifikace těsně pod hranicí 80%. V přímé konfrontaci s jinými klasifikačními metodami (C4.5, neuronová síť) je pak přesnost klasifikační metody ARC-BC o cca 5% nižší.

7.4 Zhodnocení provedených testů

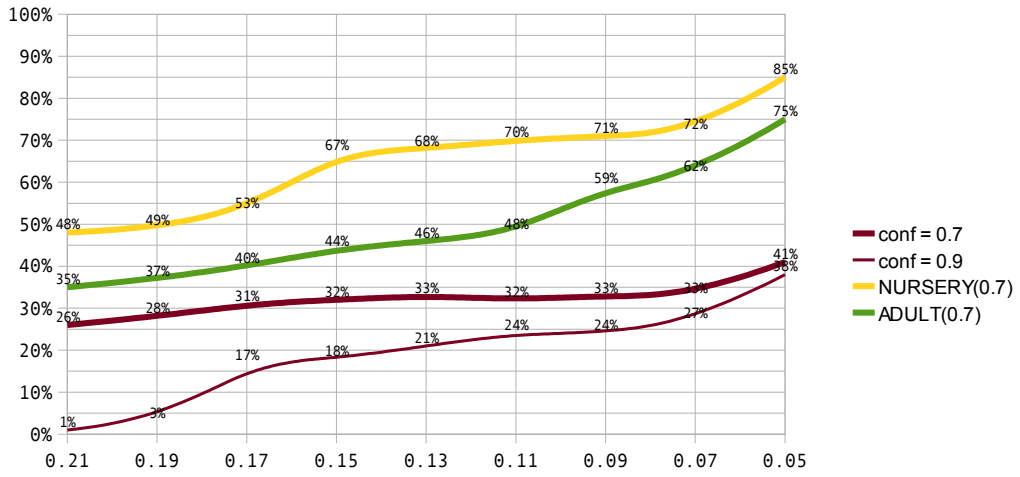
Na Obrázku 3 je možné sledovat zlepšování výsledků klasifikace se snižující se hodnout podpory *min_supp* pro všechny tři datové soubory. Žlutá čára znázorňuje vývoj přesnosti při klasifikaci dat. souboru NURSERY, zelená čára dat. souboru ADULT, hnědé čáry při klasifikaci webových dat při dvou různých hodnotách spolehlivosti - 0.7 a 0.9.

Klasifikace nad požadovanými daty z webu během provádění experimentů vykazovala velice špatné výsledky; přesnost klasifikace nového dokumentu se pohybovala kolem hranice 40%. Pro nasazení v praktickém provozu bylo potřeba zvýšit tuto přesnost o několik desítek procent.

Vzhledem k tomu, že klasifikátor ARC-BC prokázal v testech[1] dobrých výsledků na textových datech, mohla být nedostatečná přesnost způsobená některým z následujících faktorů:

1. Špatná implementace
2. Nevhodnost metody ARC-BC pro relační data
3. Nevhodnost metody ARC-BC na numerické atributy
4. Potřeba sofistikovanější metody pro diskretizaci numerických atributů

Při dalších experimentech byla metoda prověřena na datových souborech NURSERY a ADULT. V případě dat NURSERY s ryze kategorickými atributy se podařilo dosáhnout přesnosti klasifikace podobné výsledkům jiných, ryze relačních klasifikačních metod. Tímto byly ze seznamu potenciálních zdrojů problému vyloučeny body (1) a (2).



Obrázek 7.3: Vývoj přesností klasifikační metody při různých hodnotách podpory *min_supp*(osa x).

Klasifikace datového souboru ADULT umožnila ověřit vhodnost metody pro numerické atributy. Přesnost klasifikace byla sice nižší než v případě dat NURSERY, ale téměř 80% pravděpodobnost zařazení dokumentu do správné třídy se dá pokládat za úspěšné prověření klasifikátoru. Bod (3) se tedy také neukázal být pravou příčinou problémů.

Zbývající bod (4) souvisí s tvorbou diskretizovaných intervalů z množiny hodnot numerických atributů. Implementovaný algoritmus diskretizace podle stejné šířky intervalů není vhodný pro určité rozložení hodnot atributu. Nevhodně vytvořené intervaly se pak ve formě frekventovaných množin stanou předpokladem asociačních pravidel a v důsledku strůjcem klasifikačních chyb.

V rámci provedených experimentů je zřejmé, že klasifikační metoda má problémy se zpracováním dat s větším množstvím numerických atributů. Přestože nižší přesnost klasifikace dat. souboru ADULT než dat. souboru NURSERY může být způsobena celkově odlišným charakterem dat, v souvislosti s velice nízkou přesností u dat z webových stránek a s přihlédnutím ke všem testům se zdá, že klasifikační metoda se špatně vypořádává s numerickými atributy zejména v rámci diskretizace.

Kapitola 8

Závěr

Tato práce se zabývala klasifikací webových stránek, jakožto jednou z metod dolování znalostí z dat. Byly představeny základní metody klasifikace se zaměřením na klasifikační metody využívající asociační pravidla.

Dále byla diskutována problematika klasifikace s využitím dat získaných analýzou oblastí webové stránky a současně analýzou její textové částí.

V rámci práce byl vytvořen návrh hypotetického klasifikačního systému webových stránek, který klasifikuje stránky na základě vizuálních vlastností webové stránky a bere v potez také textový obsah stránek. Navržený klasifikátor využívá klasifikační metodu ARC-BC, která pracuje na principu dolování asociačních pravidel z dat. Klasifikační systém využívá několik subsystémů - systém pro analýzu vizuálních vlastností - informací o rozmístění jednotlivých částí na stránce, systém pro předzpracování vizuálních vlastností pro klasifikační metodu ARC-BC a v neposlední řadě implementaci samotné metody ARC-BC.

Samotným cílem diplomové práce bylo adaptovat klasifikační metodu ARC-BC na relační data. Metoda ARC-BC dosahuje při klasifikaci textových dat oproti jiným klasifikačním metodám vysoké přesnosti a pokud by se podařilo získat podobné výsledky i v rámci relačních dat, bylo by jednodušše možné využít jí právě jako metodu pro klasifikaci obou typů dat.

Při uprávě klasifikační metody bylo nutné řešit řadu problémů souvisejících zejména s číselnými atributy, neboť při textové klasifikaci diskretizace numerických atributů neměla žádný význam. Hlavní porci úprav zabralo přizpůsobení algoritmu dolování asociačních pravidel algoritmu ARC-BC pro správnou funkci jak s frekventovanými množinami, tak s diskretizovanými intervaly.

Takto upravený relační asociační klasifikátor ARC-BC byl implementován v programovacím jazyce Java a následně prověřen na datech analyzátoru vizuálních vlastností. I přes škálu různých nastavení parametrů klasifikace se nepodařilo dosáhnout přesnosti klasifikace vyšší než 40%. Nízká hodnota přesnosti předpovědi klasifikace vedla k dalším experimentům na alternativních testovacích datových souborech, které měly ještě více prověřit vhodnost metody ARC-BC pro relační data.

Experimenty ukázaly správnou funkčnost metody a také fakt, že hlavním problémem dosažení vyšší přesnosti klasifikace je zpracování numerických atributů. Použitá diskretizační metoda do šířky v některých případech produkuje takové diskretizované intervaly, které ve formě frekventovaných množin evokují vytvoření sady nepřesných asociačních pravidel a nepřímo tak mají vliv na přesnost metody.

Obecně metoda prokázala pro relační data dobré výsledky; s přihlédnutím k provedeným experimentům a k samotnému faktu, že s textovými daty si ARC-BC dokáže poradit bez

problémů, je zřejmě hlavním problémem výběr diskretizační metody. Použitá diskretizační metoda není dostačující a v pro nasazení metody by bylo třeba zvolit některou z "inteligentnějších" diskretizačních metod.

Literatura

[1] Maria-Luiza Antonie and Osmar R. Zaiane. Text document categorization by term association. In *ICDM*, pages 19–26. IEEE Computer Society, 2002.

[2] S. Chakrabarti. Data mining for hypertext: A tutorial survey. *SIGKDD Explorations*, 1:1–11, 2000.

[3] S. Chakrabarti. Mining the web: Discovering knowledge from hypertext data. 2002.

[4] S. Chakrabarti, B.E. Dom, S.R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the web's link structure. 32:60–67, 1999.

[5] B. Chandra, Sati Mazumdar, Vincent Arena, and Nagender Parimi. Elegant decision tree algorithm for classification in data mining. In Bo Huang, Tok Wang Ling, Mukesh K. Mohania, Wee Keong Ng, Ji-Rong Wen, and Shyam K. Gupta, editors, *WISE Workshops*, pages 160–169. IEEE Computer Society, 2002.

[6] Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *AAAI*, pages 540–545. AAAI Press, 2007.

[7] Mohammad El-Hajj, Jiyang Chen, Osmar R. Zaiane, and Randy Goebel. Constraint-based mining of web page associations. In Mehmet A. Orgun and John Thornton, editors, *Australian Conference on Artificial Intelligence*, volume 4830 of *Lecture Notes in Computer Science*, pages 315–326. Springer, 2007.

[8] Ronen Feldman, Ido Dagan, and Haym Hirsh. Mining text using keyword distributions. *J. Intell. Inf. Syst.*, 10(3):281–300, 1998.

[9] Kening Gao, Bin Zhang, Qiaozi Chai, Leiming Yang, and Zhen You. An algorithm for implementing web page automatic classification based on site structure. In Gabriele Kotsis, David Taniar, Stephane Bressan, Ismail Khalil Ibrahim, and Salimah Mokhtar, editors, *iiWAS*, volume 196 of *books@ocg.at*, pages 1075–1084. Austrian Computer Society, 2005.

[10] C. Goller, J. L"oning, T. Will, and W. Wolff. Automatic document classification: A thorough evaluation of various methods. *7. Internationales Symposium f"ur Informationswissenschaft*, 2000.

[11] J. Han and M. Kamber. *Data mining: concepts and techniques*. Morgan Kaufmann, San Francisco, 2001.

[12] Jiawei Han and Jian Pei. Mining frequent patterns by pattern-growth: Methodology and implications. *SIGKDD Explorations*, 2(2):14–20, 2000.

[13] Wenmin Li, Jiawei Han, and Jian Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In Nick Cercone, Tsau Young Lin, and Xindong Wu, editors, *ICDM*, pages 369–376. IEEE Computer Society, 2001.

[14] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *KDD*, pages 80–86, 1998.

[15] Larry M. Manevitz and Malik Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481, 2007.

[16] A. Markov, M. Last, and A. Kandel. Model-based classification of web documents represented by graphs. In *WEBKDD*, Philadelphia, Pennsylvania, USA., 2006.

[17] Andrew McCallum and Kamal Nigam. A comparison of event models for naive Bayes text classification. In *Learning for Text Categorization: Papers from the 1998 AAAI Workshop*, pages 41–48, 1998.

[18] Rob Potharst and Jan C. Bioch. A decision tree algorithm for ordinal classification. In David J. Hand, Joost N. Kok, and Michael R. Berthold, editors, *IDA*, volume 1642 of *Lecture Notes in Computer Science*, pages 187–198. Springer, 1999.

[19] Stephen Soderland. Learning to extract text-based information from the world wide web. In *Knowledge Discovery and Data Mining*, pages 251–254, 1997.

[20] Guangfeng Song. Analysis of web page complexity through visual segmentation. In Julie A. Jacko, editor, *HCI (4)*, volume 4553 of *Lecture Notes in Computer Science*, pages 114–123. Springer, 2007.

[21] Fadi A. Thabtah, Peter I. Cowling, and Yonghong Peng. Mcar: multi-class classification based on association rule. In *AICCSA*, page 33. IEEE Computer Society, 2005.

[22] J. Urnkranz. Web mining, 2005.

[23] Shivakumar Vaithyanathan, Jian C. Mao, and Byron Dom. Hierarchical bayes for text classification. In *Proceedings of the PRICAI Workshop on Text and Web Mining*, pages 36–43, 2000.

[24] Osmar R. Zaiane and Maria-Luiza Antonie. Classifying text documents by associating terms with text categories. In Xiaofang Zhou, editor, *Australasian Database Conference*, volume 5 of *CRPIT*. Australian Computer Society, 2002.

Kapitola 9

Seznam příloh

Příloha A Tabulka významných naměřených hodnot při testování klasifikátoru nad datovými soubory.

Příloha B Uživatelská přiručka pro použití programu včetně popisu výstupu programu.

Příloha A - Data z experimentů

	precision	well_classified	bad_classified	non_class	
WEBDATA(conf 0.6)	39%	2233	3517	28	
NURSERY(conf 0.6)	78%	10229	2719	10	
ADULT(conf 0.6)	75%	24433	8128	0	
WEBDATA(conf 0.7)	39%	2274	3343	161	
NURSERY(conf 0.7)	85%	11046	1564	348	
ADULT(conf 0.7)	74%	24001	6389	2171	
WEBDATA(conf 0.8)	38%	2223	3063	492	
NURSERY(conf 0.8)	73%	9484	1250	10	
ADULT(conf 0.8)	59%	19366	1851	11344	
WEBDATA(conf 0.9)	33%	1888	2093	1797	
NURSERY(conf 0.9)	66%	8496	256	4216	
ADULT(conf 0.9)	27%	8793	223	23545	
WEBDATA(conf 0.95)	33%	1934	1160	2684	
NURSERY(conf 0.95)	63%	8205	51	4702	
ADULT(conf 0.9)	25%	8205	200	25102	

Tabulka 1: Tabulka naměřených výsledků klasifikace pro testovaná data.

Příloha B - Použití programu

Přiložená aplikace je dodaná ve formě `.class` souborů přeložených Java kompilátorem. Pro spuštění stačí spustit třídu `Main` se dvěma číselnými parametry - min. spolehlivostí a min. podporou, tedy `java textclassifier.Main <min_supp> <min_conf>`.
Výpis spuštěného programu:

```
java textclassifier.Main 0.05 0.6

OK: Connected.
OK: Connected on 'localhost' to the database 'romek' as user 'root'.
```

Informace o úspěšnem přípojení k datovému zdroji, v případě chyby se vypíše chybové hlášení. Po připojení proběhne diskretizace numerických atributů, pro každý atribut se prezentuje informace o počtu intervalů a jejich velikosti.

```
intervals: 24  size: 240     r*i:5760
intervals: 23  size: 251     r*i:5773
intervals: 23  size: 251     r*i:5773
intervals: 17  size: 339     r*i:5763
intervals: 17  size: 339     r*i:5763
intervals: 33  size: 175     r*i:5775
intervals: 23  size: 251     r*i:5773
intervals: 32  size: 180     r*i:5760
```

Následuje výpis nalezených kategorií s údajem o počtu dokumentů do nich spadajících a dodatečně také seznam všech nalezených atributů dokumentu.

```
none: 4805
date: 72
menu: 44
h3: 157
h1: 19
aktualita: 442
h2: 239
 Following attributes found:
----------------------------
fontsize
weight
style
aabove
abelow
```

```
aleft
aright
tlength
tdigits
tlower
tupper
tspaces
textbtns
bgbtns
contrast
```

V další fázi běhu programu dochází k dolování frekventovaných množin. Pro každou kategorii probíhá dolování zvlášť, podobný výpis pak dokumentuje počet frekventovaných množin nalezených v příslušné kategorii.

```
==========================================
Total transactions in category none:4805
LEVEL 1: 38
LEVEL: 2 Generating frequent itemset...
Level 2: 168
LEVEL: 3 Generating frequent itemset...
```

Po dokončení dolování frekventovaných množin se provede generování asociačních pravidel, což se v konzoli projeví velkým počtem podobných řádků:

```
Rule:  tlength:  tlength>=0.0 AND tlength<=0.0 , ==> C1
conf:1.0  supp:0.11696150153875351
```

Vygenerování asociačních pravidel ukončuje fází dolování znalostí a začiná fáze klasifikace dokumentů. Každý dokument je klasifikován do jedné či více kategorií, ve výpisu jsou kategorie serařazené podle průměrné spolehlivosti asociačních pravidel pokrývajících dokument.

```
==Document=================================
==Belongs to category: none
fontsize: 93.0weight: normalstyle: normalaabove: 1.0abelow: 0.0
aleft: 0.0aright: 0.0tlength: 90.0tdigits: 8.0tlower: 53.0tupper:
6.0tspaces: 15.0textbtns: 0.03934bgbtns: 0.94296contrast: 11.11443cat: zpravy
2
11:    date      ##################  (84.81024449521846%)
16:    menu      ################  (71.41233459115028%)
```

Po dokončení klasifikace všech dokumentů program informuje o dosažených výsledcích při klasifikaci. První řádek udává celkový počet klasifikovaných dokumentů, dále počet dokumentů, které zůstaly neklasifikovány a počet dokumentů klasifikovaných. Na dalším řádku se pokračuje výpisem počtu správně zařazených dokumentů a přesnosti, resp. chybovostí klasifikační metody. Poslední řádek výpisu udává relativní přesnost, tedy přesnost na pouze klasifikovaných datech.

```
============================================================
Total documents: 5778
Non classified: 28
```

```
Classified: 5750
============================================================
Well classified: 2233
OK %: 0.38646590515749396
MR %: 0.6135340948425061
TOTAL/NON OK %: 0.3883478260869565
============================================================
```