VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

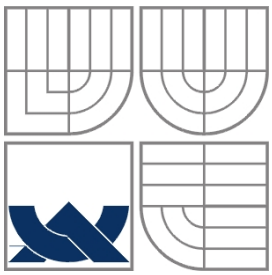
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ZÍSKÁVÁNÍ ZNALOSTÍ Z DATABÁZÍ

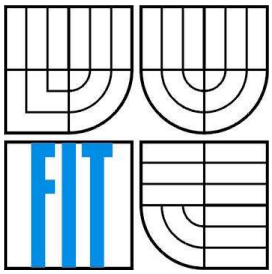
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE TOMÁŠ JIRMÁSEK
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ZÍSKÁVÁNÍ ZNALOSTÍ Z DATABÁZÍ

KNOWLEDGE DATA DISCOVERY

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ JIRMÁSEK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. PAVEL JURKA

BRNO 2007

Zadání bakalářské práce/5258/2006/xjirma00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Jirmásek Tomáš**
Obor: Informační technologie
Téma: **Získávání znalostí z databází**
Kategorie: Databáze

Pokyny:

1. Seznamte se s metodami pro získávání znalostí z databází
2. Po domluvě s vedoucím se zaměřte na jednu z metod.
3. Vybranou metodu implementujte.
4. Ověřte funkčnost a demonstrujte možnosti metody na vybraném vzorku dat.
5. Zhodnoťte získané výsledky

Literatura:

- po dohodě s vyučujícím

Při obhajobě semestrální části projektu je požadováno:

- První 2 body zadání

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Jurka Pavel, Ing.**, UITS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav inteligentních systémů
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Tomáš Jirmásek**
Id studenta: 84338
Bytem: Vortová 38, 539 61 Vortová
Narozen: 22. 04. 1985, Chrudim
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Získávání znalostí z databází
Vedoucí/školitel VŠKP: Jurka Pavel, Ing.
Ústav: Ústav inteligentních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☒ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Tato bakalářská práce se zabývá problematikou získávání znalostí z databází, konkrétně metodou Bayesovské klasifikace. Cílem práce bylo implementovat vybranou metodu dolování dat a její funkčnost ověřit na vybraném vzorku dat. Aplikace je implementována v programovacím jazyce Java a data určená pro dolování jsou uložena v databázi MySQL. Informace potřebné pro spuštění dolovací úlohy jsou načítány ze vstupního DMSL dokumentu. Získané znalosti jsou poté ukládány také do DMSL dokumentu. Jazyk DMSL musel být rozšířen pro potřeby Bayesovské klasifikace.

Klíčová slova

Získávání znalostí z databází, dolování z dat, dolování dat, klasifikace, Bayesovská klasifikace, pravděpodobnost, Bayesův vzorec, jazyky pro dolování dat, PMML, DMSL, Java, JDBC, MySQL

Abstract

This bachelor's thesis deals with knowledge discovery in databases and is focused on Bayesian classification. The main goal of this thesis was to implement one of the methods of data mining and to verify its functionality on chosen data set. The application is implemented in programming language Java. MySQL database was chosen as a data storage for data set prepared to extract patterns from it. Information needed to start data mining task are gained from input DMSL document. The results of data mining are also stored into output DMSL document. The DMSL language had to be extended because of implemented method, Bayesian classification.

Keywords

Knowledge discovery in databases, data mining, classification, Bayesian classification, probability, Bayes theorem, languages for data mining, PMML, DMSL, Java, JDBC, MySQL

Citace

Tomáš Jirmásek: Získávání znalostí z databází, bakalářská práce, Brno, FIT VUT v Brně, 2007

Získávání znalostí z databází

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Pavla Jurky.
Další informace mi poskytli Ing. Petr Chmelař.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Tomáš Jirmásek
15.5.2007

Poděkování

Děkuji tímto svému vedoucímu panu Ing. Pavlu Jurkovi za jeho odborné vedení, rady a pomoc při řešení této bakalářské práce.

Obsah

1 Úvod

Pro dnešní dobu je charakteristický obrovský nárůst objemu dat, které jsou uchovávána v elektronické podobě. Tato skutečnost souvisí s rychlým rozvojem v oblasti informačních technologií, což umožnilo produkovat velká množství dat, a ta následně sbírat a ukládat. Na druhou stranu se ale stal obtížnější proces získávání užitečných informací a znalostí z těchto dat. Nárůst množství dat totiž způsobil, že data obsahují značné množství méně užitečných, neužitečných až bezvýznamných dat – tzv. šumu, a proto dříve používané metody analýzy již nebyly schopny odhalit požadované znalosti v takto obrovských objemech dat. Výsledkem tohoto stavu byl vznik nového směru v oblasti počítačových věd, který se nazývá získávání znalostí z databází.

Druhá kapitola obsahuje stručný úvod do problematiky získávání znalostí z databází.

Ve třetí kapitole je podrobně popsána použitá metoda pro dolování dat, tedy Bayesovská klasifikace.

Čtvrtá kapitola se zabývá jazyky používanými pro podporu dolování dat. Stručně jsou zde popsány jazyky PMML a DMSL sloužící pro popis celého procesu dolování dat. A také je zde navrženo rozšíření jazyka DMSL pro potřeby Bayesovské klasifikace.

V páté kapitole je uveden návrh koncepce aplikace, formát dat určených pro dolování a formát vstupního DMSL dokumentu.

Šestá kapitola popisuje implementaci aplikace pro získávání znalostí z databází pomocí Bayesovská klasifikace.

Sedmá kapitola se zaměřuje na ověření funkčnosti aplikace a uvádí výsledky různých testů klasifikátoru pro daná testovací data.

Osmou kapitolou je závěr, který obsahuje zhodnocení dosažených výsledků a návrhy pro další vývoj a vylepšení práce.

2 Získávání znalostí z databází

2.1 Úvod

Získávání znalostí z databází (Knowledge Discovery in Databases – zkráceně KDD), dolování dat, nebo dolování z dat (Data Mining), všechny tyto tři názvy označují jeden směr v oblasti počítačových věd, o kterém se začalo výrazněji diskutovat v 90. letech minulého století. Byla to reakce na prudký rozvoj databázových technologií v předcházejících letech a cílem bylo najít metody a algoritmy pro automatizovanou analýzu velkého množství dat. Ve skutečnosti se jedná o multidisciplinární obor, který čerpá a využívá již existující metody, technologie a postupy zejména z databázových technologií, metod strojového učení, statistických postupů a mnoha dalších disciplín.

Získávání znalostí z databází je definována podle [1] takto: „*Je to netriviální získávání implicitních, dříve neznámých a potenciálně užitečných informací z dat*“. Tedy tyto informace se nedají získat jednoduchým dotazem, jsou v datech skryta – musí být nejprve nalezeny, a poté jsou využívány jako podklady pro různá rozhodnutí.

2.2 Proces získávání znalostí z databází

Jelikož se jedná o netriviální proces, je zřejmé, že nebude probíhat pouze v jednom kroku, ale bude se skládat z několika oddělených částí, pomocí kterých získáme požadovaný výsledek, přičemž jednotlivé kroky se zpravidla v určitých iteracích opakují. Celý proces je znázorněn na obrázku 2.1.

Pokud se vrátíme k jednotlivým názvům tohoto oboru informačních technologií, tak získávání znalostí z databází označuje celý proces a dolování dat, či dolování z dat je pouze jedním krokem celého procesu. V současnosti se ale používá i k označení celého procesu častěji pojem dolování dat (či z dat).

Jednotlivé kroky tohoto procesu jsou:

- **Čištění a integrace dat**

  Tyto první dva kroky se provádějí velmi často společně. Dochází zde k odstranění dat, která by mohla působit v dalších krocích nějaké problémy – např. chybějící data, odstranění šumu nebo vyřešení nekonzistence dat, a zároveň probíhá spojování dat z několika nezávislých datových zdrojů.

- **Výběr dat**

  V tomto kroku jsou vybírána pouze relevantní data pro použitou úlohu (dolovací metodu).

- **Transformace dat**

  Cílem tohoto kroku je převedení dat do podoby, která je vhodná pro použitou dolovací metodu. Zároveň je tímto krokem ukončena část nazývaná předzpracování dat. Někdy takto

souhrnně může být označen první krok celého procesu a to zejména v případech, kdy není potřeba rozlišovat tyto konkrétní úpravy dat.

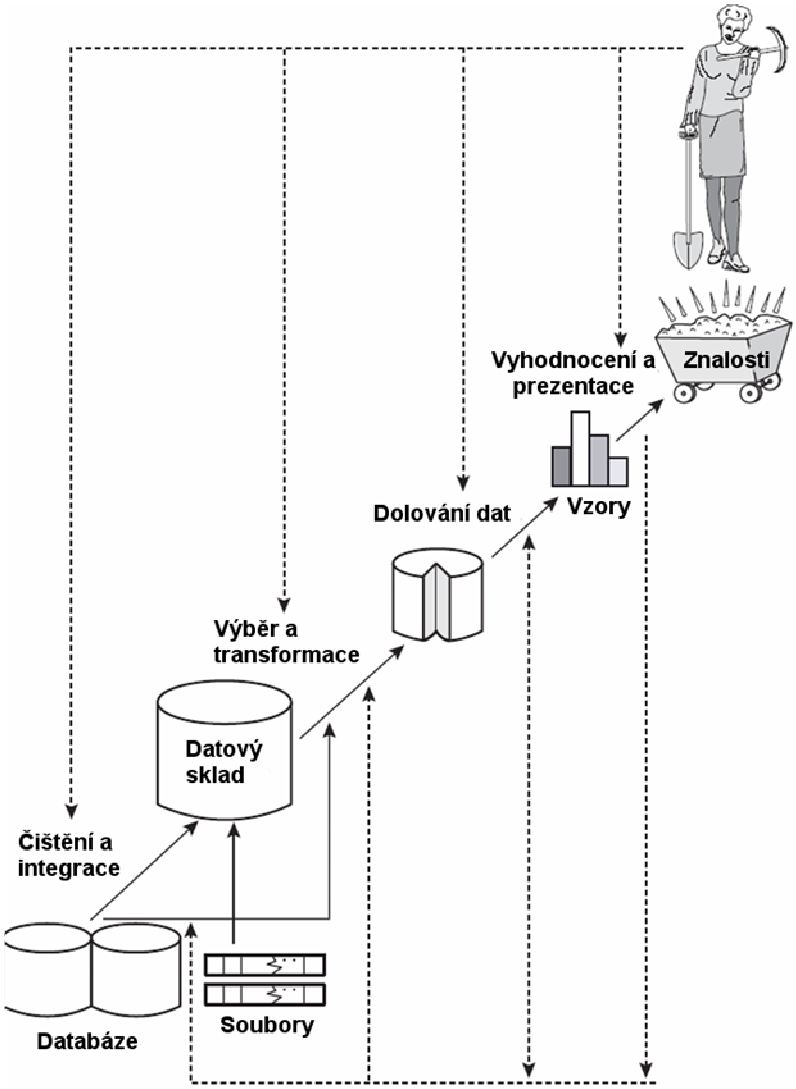
- **Dolování dat**

  Nyní následuje samotný proces dolování dat. Cílem je použitím vybrané metody získat z předzpracovaných dat požadované znalosti (vzory). Tento krok je v celém procesu získávání znalostí z databází nejdůležitější, ostatní slouží pouze jako pomocné kroky pro upravení dat nebo nalezených znalostí.

- **Vyhodnocení modelů a vzorů, prezentace znalostí**

  V posledních dvou krocích zbývá ještě určit, které z nalezených vzorů (znalostí) jsou skutečně zajímavé, a tyto výsledky prezentovat uživateli v nejrůznějších srozumitelných formách.

Obrázek 2.1: Proces dolování dat (převzato z [2])



2.3 Data vhodná pro dolování

Datové vzory lze dolovat z mnoha rozdílných druhů databází. Nejčastěji jsou používány relační databáze, dále datové sklady, transakční, objektově-relační a objektově-orientované databáze. Pro dolování jsou vhodné také další druhy zdrojů dat, jako jsou prostorové, textové, multimediální, heterogenní, temporální a zděděné databáze, ale i také samotný web.

2.4 Typy dolovacích úloh

Je vhodné rozdělit dolování dat na jednotlivé dolovací úlohy, které odpovídají požadavkům na analýzu dat. Pro samotné dolování dat jsou tedy používány různé typy dolovacích úloh, podle toho jaké znalosti (vzory) se snažíme z dat získat. Tyto typy dolovacích úloh lze rozdělit do dvou základních skupin:

- **Deskriptivní**
  Charakterizují obecné vlastnosti dat.
- **Prediktivní**
  Analyzují vstupní data a poté provádějí předpovědi nových hodnot, neboli budoucího chování.

Konkrétní typy dolovacích úloh:

- **Popis konceptu/třídy**
  Data mohou mít určenu příslušnost k jednotlivým třídám nebo konceptům. Následně je snaha získat popis těchto konceptů nebo tříd, čehož lze dosáhnout dvěma způsoby. Sumarizací obecných vlastností cílové třídy, což je tzv. charakterizace dat. Nebo hledáním rozdílů mezi obecnými vlastnostmi cílové třídy a obecnými vlastnostmi rozdílové třídy. Tato metoda je nazývána diskriminace dat.
- **Asociační analýza**
  Cílem tohoto typu dolovací úlohy je odhalování vzorů a asociací (vztahů mezi atributy), které se v datech vyskytují často. Použitím asociační analýzy získáme asociační pravidla ve tvaru $X \Rightarrow Y$, kde X a Y označují tvrzení týkající se hodnot atributů. Příkladem tohoto pravidla může být např.
  $$vek(X,'30..39') \wedge prijem(X,'20k..29k') \Rightarrow kupuje(X,'vlastni\_byt').$$
  Spolu s každým asociačním pravidlem jsou uváděny hodnoty podpory a spolehlivosti, ze kterých lze odvodit míru věrohodnosti pravidla. Pro pravidlo $X \Rightarrow Y$ by byla podpora určená jako procento objektů, které obsahují tvrzení X nebo Y, a to ze všech objektů. A spolehlivost jako procento objektů, pro které platí Y za předpokladu, že platí současně X.

- **Klasifikace a predikce**

  Základem je, že objekty je možné zařadit do tříd na základě jejich vlastností. Klasifikace hledá určitá pravidla nebo modely, která popisují data jednotlivých tříd. Tyto vytvořená pravidla a modely jsou poté používány pro predikci třídy objektů dat, jejichž třída je neznámá, nebo pro předpověď hodnoty atributů. Celý proces této dolovací úlohy se skládá ze tří kroků. První dva kroky jsou trénování (učení) a testování. V nich se pracuje s daty objektů, jejichž třída je známá, a je vytvářen klasifikační model, jehož kvalita je následně testována. Posledním krokem je aplikace, kdy podle vytvořeného klasifikačního modelu jsou objekty řazeny do příslušných tříd.

- **Shluková analýza**

  Shluková analýza, neboli shlukování, se používá stejně jako klasifikace pro rozdělení dat do tříd, ale na rozdíl od klasifikace nejsou známa pravidla nebo model, podle nichž by mohly být datové objekty přiděleny do odpovídajících tříd. Cílem shlukování je tedy najít třídy objektů, neboli shluky, a to na základě maximální podobnosti dat uvnitř shluku a naopak minimální podobnosti mezi shluky.

- **Analýza odlehlých objektů**

  Tato dolovací úloha se od ostatních liší tím, jaké hodnoty se v datech snaží najít. Databáze může obsahovat také datové objekty, které se od ostatních výrazně liší, ty jsou nazývány odlehlé objekty. Většina dolovacích úloh tyto odlehlé objekty považuje za tzv. šum, ale pro analýzu odlehlých hodnot tvoří právě tyto objekty zajímavé vzory, které jsou dolovány.

- **Analýza evoluce**

  Analýza evoluce popisuje a modeluje pravidelnosti a trendy vývoje objektů, jejichž chování se mění v čase.

Informace v této kapitole byli čerpány převážně z [1] a [2].

3 Bayesovská klasifikace

3.1 Klasifikace a predikce

Klasifikace a predikce jsou dvě formy datové analýzy, které jsou používány pro získávání modelů popisujících důležité datové třídy nebo pro predikci budoucích datových trendů. Mezi označením klasifikace a predikce je drobný rozdíl. Obecně může být pojmem predikce označeno použití klasifikačního modelu pro určení třídy nového nezařazeného datového objektu, nebo pro předpověď hodnoty atributu. V oboru získávání znalostí z databází je ale přijato, že klasifikace se používá pro předpověď hodnot kategorických tříd a pro hodnoty spojitých atributů se používá predikce. Jednodušeji řečeno je klasifikace zařazení daného objektu do určité třídy na základě jeho vlastností a predikce předpověď jisté hodnoty pro daný objekt také na základě jeho vlastností (hodnota je obecně spojitého charakteru).

3.1.1 Fáze klasifikace

Proces klasifikace probíhá ve dvou fázích. První fáze je nazývána fází učení. Vstupem jsou trénovací data (množina), což jsou data, u kterých je známo do které třídy patří. Pomocí klasifikátoru (aplikací klasifikačního algoritmu na trénovací data) jsou získána klasifikační pravidla, která jsou poté používána pro klasifikaci nových objektů do daných tříd. Cílem druhé fáze je testováním použitého klasifikátoru určit jeho úspěšnost a spolehlivost při klasifikaci nových dat. Vstupem jsou testovací data (množina), pro které je opět známa jejich klasifikační třída, ale tato data musejí být nezávislá na trénovacích datech. Použitím klasifikátoru na tato data se určí procentuální úspěšnost klasifikace, tedy v kolika procentech byla třída určena správně či nesprávně. Na základě těchto výsledků se určí, zda-li klasifikátor může být použit pro klasifikaci nových dat.

3.2 Bayesovská klasifikace

Bayesovská klasifikace je založena na statistice a pravděpodobnosti, konkrétně na Bayesově teorému. Její obecný princip je takový, že pro každý nový datový objekt je podle statistických metod určeno, s jakou pravděpodobností patří do jednotlivých tříd a poté je tento objekt zařazen do té třídy, do které s největší pravděpodobností patří.

3.2.1 Úvod do pravděpodobnosti

Základem je tzv. klasická pravděpodobnost.

$$P(A) = \frac{|A|}{|\Omega|}$$ (svislé čáry označují počet prvků množiny)

Klasická pravděpodobnost jevu A je definována jako podíl počtu příznivých výsledků (počtu prvků množiny A) ku počtu všech možných výsledků (počtu prvků množiny Ω). Je možné ji užít jen tehdy, když Ω (množina všech možných výsledků pokusu) je konečná a všechny výsledky pokusu nastávají se stejnou pravděpodobností (jsou stejně pravděpodobné).

Dalším pojmem je podmíněná pravděpodobnost.

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$P(A|B)$ označuje podmíněnou pravděpodobnost jevu A, pokud už víme, že nastala podmínka B. $P(A \cap B)$ je pravděpodobnost, že jevy *A* a *B* nastanou současně.

Nyní již zbývá pouze odvodit Bayesův vzorec. Vyjdeme ze vzorec pro podmíněnou pravděpodobnost $P(A|B)$, který je uveden výše, a vzorce pro podmíněnou pravděpodobnost $P(B|A)$, který má následující tvar:

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$

Jelikož ale ve vztahu pro výpočet průniku dvou jevů nezáleží na pořadí množin, tedy $P(A \cap B) = P(B \cap A)$, vyjádříme z obou předchozích vztahů pro výpočet podmíněné pravděpodobnosti výraz $P(A \cap B)$.

Takto upravené vztahy porovnáme $P(B|A)P(A) = P(A \cap B) = P(A|B)P(B)$ a po náležitých úpravách dostaneme následující vztah, který se nazývá Bayesův vzorec:

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

Tento vztah tvoří základ Bayesovské klasifikace.

3.2.2 Jednoduchá Bayesovská klasifikace

Jednoduchá Bayesovská klasifikace probíhá následujícím způsobem:

1. Každý datový vzorek je reprezentován n-rozměrným vektorem, $X = (x_1, x_2, \ldots, x_n)$, kde x_i je hodnota atributu A_i pro všechna $i = 1, \ldots, n$
2. Za předpokladu, že existuje m tříd, $C_1, C_2, \ldots, C_m$, je pro každý vzorek dat, X, určeno pomocí klasifikátoru, že X patří do třídy, pro kterou je největší hodnota podmíněné

pravděpodobnosti $P(C_i|X)$. Jednoduchý Bayesovský klasifikátor zařadí neznámý vzorek X do třídy C_i právě tehdy když $P(C_i|X) > P(C_j|X)$ pro $1 \le j \le m, j \ne i$. Podle Bayesova vzorce $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$

3. Jelikož $P(X)$ je konstantní pro všechny třídy, výraz $P(C_i|X)$ bude maximální právě tehdy když bude maximální $P(X|C_i)P(C_i)$. $P(C_i)$, tedy pravděpodobnost, že libovolně zvolený prvek patří do třídy C_i, určíme jednoduše podle vzorce $P(C_i) = \frac{s_i}{s}$, kde s_i je počet trénovacích vzorků, které obsahuje třída C_i, a s je celkový počet trénovacích vzorků.

   Pozn.: Je třeba rozlišovat $P(C_i|X)$ a $P(X|C_i)$. $P(C_i|X)$ označuje pravděpodobnost, že vzorek X patří do třídy C_i. Ale $P(X|C_i)$ označuje pravděpodobnost, že libovolný vzorek vybraný ze třídy C_i bude mít stejné hodnoty atributů jako vzorek X.

4. Pro datové množiny s mnoha atributy by bylo velmi náročné počítat $P(X|C_i)$, proto je výpočet zjednodušen předpokladem, že hodnoty atributů jsou navzájem nezávislé. A proto má vzorec následující tvar, $P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$, kdy pravděpodobnost $P(X|C_i)$ získáme jako součin pravděpodobností $P(x_k|C_i)$ pro $k = 1, \ldots, n$, kde $P(x_k|C_i)$ je pravděpodobnost, že prvek, jehož atribut A_k je roven hodnotě x_k, bude zařazen do třídy C_i. Hodnoty pravděpodobnosti $P(x_k|C_i)$ jsou určeny podle typu atributu A_k následujícím způsobem:

   - Pokud atribut A_k má diskrétní charakter, tak $P(x_k|C_i) = \frac{s_{ik}}{s_i}$, kde s_{ik} je počet trénovacích vzorků patřící do třídy C_i, jejichž atribut A_k má hodnotu x_k, a s_i je počet všech trénovacích vzorků patřících do třídy C_i.
   - Pokud atribut A_k má spojitý charakter, tak atributy mají typicky Gaussovo rozdělení. $P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{(x_k - \mu_{C_i})^2}{2\sigma_{C_i}^2}}$, kde $g(x_k, \mu_{C_i}, \sigma_{C_i})$ je Gaussovo normální rozložení pro atribut A_k s hodnotou x_k. μ_{C_i} je průměrná hodnota a σ_{C_i} směrodatná odchylka pro atributy A_k patřící do třídy C_i.

5. Pro klasifikaci neznámého vzorku X musí být určena hodnota $P(X|C_i)P(C_i)$ pro každou třídu C_i. Vzorek X je tedy zařazen do třídy C_i právě tehdy když $P(X|C_i)P(C_i) > P(X|C_j)P(C_j)$ pro $1 \leq j \leq m, j \neq i$. Jinak řečeno je zařazen do třídy C_i, pro kterou je hodnota $P(X|C_i)P(C_i)$ největší.

3.2.2.1 Příklad použití Bayesovské klasifikace

Tento příklad je převzat z [2]. Trénovací data jsou uvedená v tabulce 2.1 a datové vzorky jsou popsány atributy *věk*, *příjem*, *stav_student* a *třída_úvěru*. Třída je popsána atributem *koupí_ počítač*, který nabývá dvou diskrétních hodnot ({*ano*, *ne*}). Vzorky dat jsou tedy klasifikovány do dvou tříd, přičemž C_1 odpovídá třídě, že si daný uživatel počítač koupí, *koupí_ počítač = „ano“*, a C_2 odpovídá třídě, že si počítač nekoupí, *koupí_ počítač = „ne“*.

Následující vzorek dat budeme chtít zařadit:

X = (*věk = „<=30“, příjem = „střední“, stav_student = „ano“, třída_úvěru = „průměrná“*).

Tabulka 2.1: Trénovací data

	věk	příjem	stav_student	třída_úvěru	Třída: koupí_ počítač
1	<=30	velký	ne	průměrná	ne
2	<=30	velký	ne	výborná	ne
3	31…40	velký	ne	průměrná	ano
4	>40	střední	ne	průměrná	ano
5	>40	malý	ano	průměrná	ano
6	>40	malý	ano	výborná	ne
7	31…40	malý	ano	výborná	ano
8	<=30	střední	ne	průměrná	ne
9	<=30	malý	ano	průměrná	ano
10	>40	střední	ano	průměrná	ano
11	<=30	střední	ano	výborná	ano
12	31…40	střední	ne	výborná	ano
13	31…40	velký	ano	průměrná	ano
14	>40	střední	ne	výborná	ne

Pro správnou klasifikaci datového vzorku do daných tříd musíme určit, pro kterou je hodnota výrazu $P(X|C_i)P(C_i)$ největší.

Nejprve určíme $P(C_i)$, tedy pravděpodobnost, že libovolně zvolený prvek patří do třídy C_i.

P(koupí_počítač = „ano“) = 9/14 = 0.643

P(koupí_počítač = „ne“) = 5/14 = 0.357

Pro výpočet $P(X|C_i)$, kde $i = 1,2$, spočítáme následující podmíněné pravděpodobnosti pro jednotlivé atributy.

P(věk = „<=30“ | koupí_počítač = „ano“) = 2/9 = 0.222

P(věk = „<=30“ | koupí_počítač = „ne“) = 3/5 = 0.600

P(příjem = „střední“ | koupí_počítač = „ano“) = 4/9 = 0.444

P(příjem = „střední“ | koupí_počítač = „ne“) = 2/5 = 0.400

P(stav_student = „ano“ | koupí_počítač = „ano“) = 6/9 = 0.667

P(stav_student = „ano“ | koupí_počítač = „ne“) = 1/5 = 0.200

P(třída_úvěru = „průměrná“ | koupí_počítač = „ano“) = 6/9 = 0.667

P(třída_úvěru = „průměrná“ | koupí_počítač = „ne“) = 2/5 = 0.400

Nyní již z těchto pravděpodobností můžeme vypočítat $P(X|C_i)$, kde $i = 1,2$.

P(X | koupí_počítač = „ano“) = 0.222 × 0.444 × 0.667 × 0.667 = 0.044

P(X | koupí_počítač = „ne“) = 0.600 × 0.400 × 0.200 × 0.400 = 0.019

A následně i celý výraz $P(X|C_i)P(C_i)$, kde $i = 1,2$.

P(X | koupí_počítač = „ano“)P (koupí_počítač = „ano“) = 0.044 × 0.643 = 0.028

P(X | koupí_počítač = „ne“)P (koupí_počítač = „ne“) = 0.019 × 0.357 = 0.007

Z výsledků tedy vyplývá, že klasifikátor určí pro daný vzorek X, že s větší pravděpodobností bude patřit do třídy C_1, *koupí_ počítač = „ano“*.

3.2.3 Efektivita Bayesovkého klasifikátoru

Teoreticky má Bayesovský klasifikátor minimální míru chybovosti v porovnání s ostatními používanými klasifikátory. Ve skutečnosti ale není toto tvrzení vždy pravdivé. Míra chybovosti může být ovlivněna díky nepřesnostem v předpokladech, které byly použity pro zjednodušení některých výpočtů (např. při výpočtu podmíněné pravděpodobnosti $P(X|C_i)$, že libovolný vzorek vybraný ze třídy C_i bude mít stejné hodnoty atributů jako vzorek X, byl použit předpoklad, že hodnoty jednotlivých atributů jsou navzájem nezávislé).

Na výsledek klasifikace má také výrazný vliv množství trénovacích dat, na kterých jsou prováděny výpočty pro určení pravděpodobností příslušnosti nových datových vzorků do jednotlivých tříd. Při malém množství trénovacích dat mohou být výsledky klasifikace značně nepřesné a pro některé datové vzorky nemusí být možné rozhodnout, do které třídy patří. Tato situace nastane, pokud datový vzorek patří do všech tříd s nulovou pravděpodobností, tedy některé hodnoty atributů datového vzorku nejsou v trénovacích datech nalezeny.

Informace v této kapitole byly čerpány především z [1], [2] a [3].

4 Jazyky pro dolování dat

4.1 Úvod

Jak již bylo uvedeno výše, dolování dat je netriviální proces probíhající v několika krocích, a proto bylo důležité vytvořit jazyk nezávislý na určité platformě, který by umožnil flexibilní a efektivní získávání znalostí. Důležitost návrhu dobrého jazyka pro podporu dolování dat lze ukázat na vývoji relačních databázových systémů, které díky standardizaci jazyka SQL již v počátcích jejich vývoje, drží i dnes silnou pozici na trhu s databázovými systémy.

Ale návrh komplexního jazyka pokrývajícího celý proces dolování dat není jednoduchou záležitostí, protože dolování dat probíhá v několika krocích a zahrnuje celou řadu různých úkolů pro samotné dolování dat, které mají každý různé požadavky. A proto většina jazyků pro dolování dat, které jsou v současnosti k dispozici, nepokrývají celý proces, ale jsou zaměřeny jen na jeho určitou část – převážně na techniky a algoritmy dolování dat. Příkladem těchto jazyků mohou být DMQL (Data Mining Query Language), MSQL a MINE RULE, které patří do skupiny relačních jazyků. Tyto jazyky jsou závislé na SQL a umožňují dolování dat z relačních databází nebo datových skladů. Podrobnější informace se lze o těchto jazycích dozvědět v [4]. Dalším příkladem může být skupina logických jazyků, do které patří Logical Data Language nebo Datalog. Tyto jazyky jsou realizovány v souvislosti s deduktivními databázemi a jazyky založenými na pravidlech. A poslední významnější skupinu tvoří jazyky založené na XML, mezi které patří KDDML, PMML a můžeme do této skupiny zařadit i jazyk DMSL. Pro tuto skupinu je charakteristické, že jsou navrhovány tak, aby podporovaly celý proces dolování dat. Dále budou zmíněny jazyky PMML a podrobněji jazyk DMSL.

4.2 Jazyk PMML

Predictive Model Markup Language, neboli PMML, je standardizovaný jazyk pro dolování dat, který je založen na XML. Tento jazyk poskytuje možnost, jak pro aplikace definovat statistické modely a modely pro dolování dat a tyto modely sdílet mezi aplikacemi podporujícími jazyk PMML. Výhodou tohoto jazyka je, že je nezávislý na aplikacích, což umožňuje přenos definic dolovacích úloh mezi jednotlivými aplikacemi. Tímto se velmi zjednodušilo používání stejných modelů různými aplikacemi, což bylo v minulosti velmi obtížné.

Tento jazyk jsem použil jako inspiraci při definování některých elementů jazyka DMSL, potřebných pro použitou metodu dolování dat. Podrobné informace – obecná struktura a syntaxe jazyka PMML je dostupná zde [5].

4.3 Jazyk DMSL

Data Mining Specification Language, neboli DMSL, je jazyk založený na bázi XML, který slouží pro popis celého dolovacího procesu (tedy od samotných vstupních dat až po získané znalosti), ale s hlavním zaměřením na předzpracování dat. Důsledkem tohoto tedy je, že elementy pro popis samotného dolování dat a následujících kroků nemají pevně danou syntaxi a musejí být definovány pomocí jiných jazyků, k čemuž je vhodný například jazyk PMML, nebo přímo definováním rozšíření samotného jazyka DMSL.

4.3.1 Obecná struktura

Kořenovým elementem celého dokumentu je DMSL element, který obsahuje všechny ostatní elementy. DMSL označuje 5 hlavních primitiv, které odpovídají hlavním krokům celého procesu. Tyto primitiva jsou:

- **Datový model (data model)** – reprezentuje vstupní data.
- **Dolovací model (data mining model)** – tento model je rozšířením datového modelu a obsahuje transformace vstupních dat do vhodné podoby pro dolování.
- **Doménová znalost (domain knowledge)** – může být použit dolovací úlohou, reprezentuje znalosti o datech.
- **Dolovací úloha (data mining task)** – popisuje dolovací úlohu, tedy které znalosti a z jakých dat budou dolována.
- **Znalosti (knowledge)** – obsahuje výstupní znalosti získané dolováním.

Navíc může ještě element DMSL obsahovat element FunctionPool, který slouží pro popis funkcí použitých v projektu, a element Header, který popisuje obecné vlastnosti dokumentu.

Zde je uvedena syntaxe a sémantika DMSL elementu:

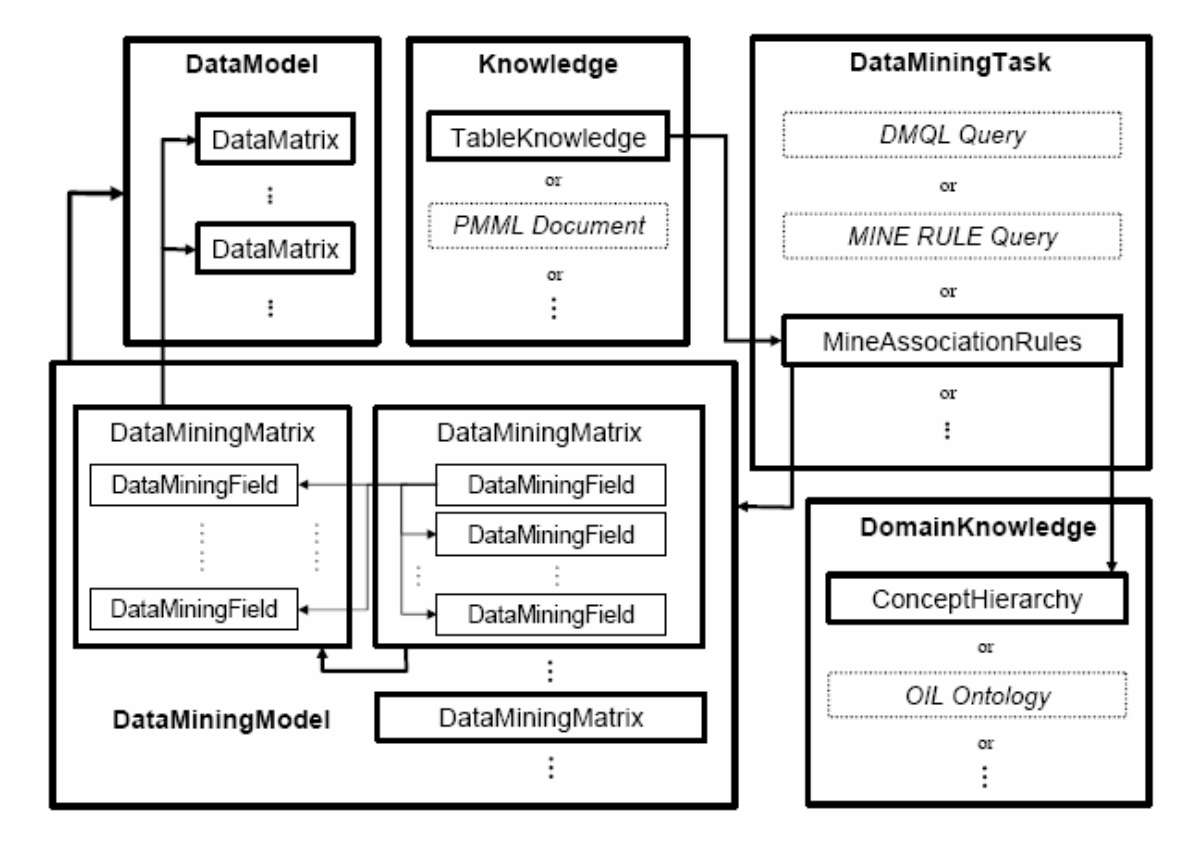
```
<!ELEMENT DMSL (Header?, (FunctionPool | DataModel | DataMiningModel | DomainKnowledge | DataMiningTask | Knowledge)+) >
```

Syntaktická a sémantická omezení:

1. Každý DMSL dokument může obsahovat jeden nebo žádný element Header.
2. Každý DMSL dokument může obsahovat libovolný nenulový počet všech hlavních primitiv spolu s elementem FunctionPool v různé kombinaci. Z tohoto vyplývá, že DMSL dokument musí obsahovat, alespoň jeden z těchto elementů, aby měl nějaký význam.

Podrobný popis celé struktury, syntaxe a sémantiky DMSL jazyka lze získat v disertační práci autora tohoto jazyka [6], nebo stručnější popis v [7].

Obrázek 4.1: Závislosti DMSL elementů (převzato z [6])



4.3.2 Rozšíření jazyka DMSL

Protože jazyk DMSL je zaměřen hlavně na předzpracování dat, tak pouze pro elementy DataModel a DataMiningModel je ve specifikaci uvedena jejich přesná syntaxe a sémantika. Pro elementy DataMiningTask a Knowledge je tedy nutno použít jiný jazyk pro dolování dat, nejčastěji PMML, nebo přímo rozšířit jazyk DMSL a tyto elementy s jejich syntaxí a sémantikou nadefinovat. V této práci jsou v podstatě využity oba způsoby, tedy elementy do jazyka DMSL jsou přímo definovány, ale jako předloha pro syntaxi jednotlivých elementů sloužil jazyk PMML.

4.3.2.1 Element DataMiningTask

```
<!ELEMENT DataMiningTask ANY >
<!ATTLIST DataMiningTask
      name                  CDATA                    #REQUIRED
      type                  CDATA                    #IMPLIED
      dataMiningModelRef    CDATA                    #IMPLIED >
```

Význam jednotlivých atributů:

- **name** – název dolovací úlohy,
- **type** – typ dolovací úlohy,
- **dataMiningModelRef** – jméno dolovacího modelu dataMiningModel, který bude použit pro dolování.

Takto je definována syntaxe a sémantika elementu DataMiningTask ve specifikaci jazyka DMSL. Pro metodu použitou v této aplikaci, což je Bayesovská klasifikace, bude tento element definován následujícím způsobem, přičemž atributy i jejich význam zůstává stejný.

```
<!ELEMENT DataMiningTask (MineBayesClassification) >
<!ATTLIST DataMiningTask
      name                  CDATA                        #REQUIRED
      type                  CDATA                        #IMPLIED
      dataMiningModelRef    CDATA                        #IMPLIED >
```

Základní element pro Bayesovskou klasifikaci, MineBayesClassification, je definován takto:

```
<!ELEMENT MineBayesClassification (MiningSchema, Database) >
<!ATTLIST MineBayesClassification
      name                  CDATA                        #IMPLIED >
```

Význam jednotlivých atributů:

- **name** – název úlohy.

Tento element obsahuje základní údaje potřebné pro zahájení samotného procesu dolování dat a to ve dvou hlavních elementech. Těmi jsou element MiningSchema a Database.

Jelikož Bayesovská klasifikace pro zařazení nového datového vzorku do dané třídy používá určení pravděpodobnosti výskytu hodnot jeho atributů v trénovacích datech, je možné definovat, které atributy mají být pro klasifikaci použity. A právě tato informace je uložena v elementu MiningSchema, tedy tento element obsahuje, která data budou použita při samotném dolování dat.

Syntaxe a sémantika elementu MiningSchema je následující:

```
<!ELEMENT MiningSchema (MiningField+, TargetMiningField) >
<!ATTLIST MiningSchema
      name                  CDATA                        #IMPLIED >
```

Význam jednotlivých atributů:

- **name** – název dolovacího schématu.

Element MiningField slouží pro definici jednotlivých atributů (tedy sloupců tabulky), které budou použity při klasifikaci nových datových vzorků do tříd. Aby bylo možné klasifikaci vůbec provést, musí být element MiningField uveden alespoň jednou. Ale malý počet atributů použitých pro klasifikaci není vhodný, neboť její výsledky mohou být poté značně nepřesné. Maximální počet těchto elementů může být o jeden menší, než je celkový počet atributů, jelikož jeden atribut je určen pro uložení příslušnosti k dané třídě. Syntaxe a sémantika tohoto elementu je následující:

```
<!ELEMENT MiningField EMPTY >
<!ATTLIST MiningField
      name                    CDATA                          #REQUIRED
      dataFieldRef            CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – název dolovacího pole,
- **dataFieldRef** – obsahuje jméno atributu použitého pro dolování, tedy jméno sloupce tabulky.

Další element, který musí být uveden právě jednou v elementu MiningSchema je TargetMiningField. Tento element obsahuje název atributu použitého pro uložení příslušnosti k dané třídě a dále v elementech FieldValue hodnoty, kterých může tento atribut nabývat, neboli jednotlivé třídy, do nichž jsou datové vzorky klasifikovány. Syntaxe a sémantika tohoto elementu je definována takto:

```
<!ELEMENT TargetMiningField (FieldValue+) >
<!ATTLIST TargetMiningField
      name                    CDATA                          #REQUIRED
      dataFieldRef            CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – název pole, které představuje třídu daného datového vzorku,
- **dataFieldRef** – obsahuje jméno atributu, tedy jméno sloupce tabulky, ve kterém je uložena třída, do níž daný datový vzorek patří.

Jak již bylo uvedeno výše, element FieldValue definuje jaké hodnoty reprezentují jednotlivé třídy. Jeho syntaxe a sémantika je definována následujícím způsobem:

```
<!ELEMENT FieldValue EMPTY >
<!ATTLIST FieldValue
      name                    CDATA                          #REQUIRED
      value                   CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – název charakterizující jednotlivé třídy,
- **value** – hodnoty představující jednotlivé třídy.

Database je druhým hlavním elementem, který musí být uveden v MineBayesClassification. Slouží pro definici databáze obsahující data pro dolování, tedy jejího jména, adresy, uživatele a hesla. Jména jednotlivých tabulek, v nichž jsou uložena data pro dolování, jsou uložena v elementech Table. Syntaxe a sémantika elementu Database je následující:

```
<!ELEMENT Database (Table, Table, Table) >
<!ATTLIST Database
      name                    CDATA                          #REQUIRED
      dbUrl                   CDATA                          #REQUIRED
      driver                  CDATA                          #REQUIRED
      userName                CDATA                          #REQUIRED
      password                CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – jméno používané databáze,
- **dbUrl** – databázová adresa pro určení připojení k databázi,
- **driver** – ovladač pro připojení k používanému typu databáze,
- **userName** – jméno uživatele, pod kterým bude připojení k databázi navázáno,
- **password** – heslo k použitému uživatelskému účtu.

Element Table obsahuje, jak již bylo uvedeno výše, názvy jednotlivých tabulek s daty. Pro tuto aplikaci musí být tento element uveden právě třikrát, přičemž hodnoty atributu name musejí být následující: pro tabulku obsahující trénovací data – training, pro tabulku obsahující testovací data – testing a pro tabulku, do které budou uložena výsledná klasifikovaná data - classified. Jeho syntaxe a sémantika je definována takto:

```
<!ELEMENT Table EMPTY >
<!ATTLIST Table
      name                    CDATA                          #REQUIRED
      value                   CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – označení tabulky databáze podle toho, jaký druh dat obsahuje,
- **value** – název příslušné tabulky.

Tímto jsou dostupné veškeré potřebné informace pro dolování a klasifikace nových datových vzorků může proběhnout.

4.3.2.2 Element Knowledge

```
<!ELEMENT Knowledge ANY >
<!ATTLIST Knowledge
      name                  CDATA                    #REQUIRED
      type                  CDATA                    #IMPLIED
      dataMiningTaskRef     CDATA                    #IMPLIED >
```

Význam jednotlivých atributů:

- **name** – název znalosti,
- **type** – typ znalosti,
- **dataMiningTaskRef** – jméno dolovací úlohy, pomocí které byla znalost získána.

Takto je definována syntaxe a sémantika elementu Knowledge ve specifikaci jazyka DMSL. Pro metodu použitou v této aplikaci, což je Bayesovská klasifikace, bude tento element definován následujícím způsobem, přičemž atributy i jejich význam zůstává stejný.

```
<!ELEMENT Knowledge (BayesOutput) >
<!ATTLIST Knowledge
      name                  CDATA                    #REQUIRED
      type                  CDATA                    #IMPLIED
      dataMiningTaskRef     CDATA                    #IMPLIED >
```

Základní element pro definici získaných znalostí, BayesOutput, má následující syntaxi a sémantiku:

```
<!ELEMENT BayesOutput (TargetValueCounts) >
<!ATTLIST BayesOutput
      fieldName             CDATA                    #REQUIRED
      dataFieldRef          CDATA                    #REQUIRED >
```

Význam jednotlivých atributů:

- **name** – název pole, které představuje třídu daného datového vzorku,
- **dataFieldRef** – obsahuje jméno atributu, tedy jméno sloupce tabulky, ve kterém je uložena třída, do níž daný datový vzorek patří.

Znalosti získané Bayesovskou klasifikací jsou reprezentovány počtem datových vzorků příslušejících do jednotlivých tříd. Tyto výsledné znalosti jsou uloženy v elementech TargetValueCount, kde každý element odpovídá jedné třídě. Těchto elementů je ale vždy o jeden více než je počet tříd, jelikož jeden element TargetValueCount je určen pro uložení informace, kolik datových vzorků nebylo možno důsledkem nedostatku trénovacích dat zařadit ani do jedné třídy. Tento problém byl již zmíněn výše v části 3.2.3 Efektivita Bayesovského klasifikátoru. Elementy TargetValueCount jsou obsaženy v elementu TargetValueCounts, jehož syntaxe a sémantika je definována takto:

```
<!ELEMENT TargetValueCounts (TargetValueCount+) >
```

Zde je uvedena syntaxe a sémantika elementu TargetValueCount, který slouží pro uložení výsledných znalostí. Element pro uložení počtu datových vzorků, které nebylo možné zařadit, má pevně danou hodnotu atributu value, a to konkrétně notClassified, ostatní mají hodnotu tohoto atributu určenou ze vstupního DMSL dokumentu z elementů FieldValue.

```
<!ELEMENT TargetValueCount EMPTY >
<!ATTLIST TargetValueCount
      value                   CDATA                          #REQUIRED
      count                   CDATA                          #REQUIRED >
```

Význam jednotlivých atributů:

- **value** – hodnoty představující jednotlivé třídy,
- **count** – počet datových vzorků v jednotlivých třídách.

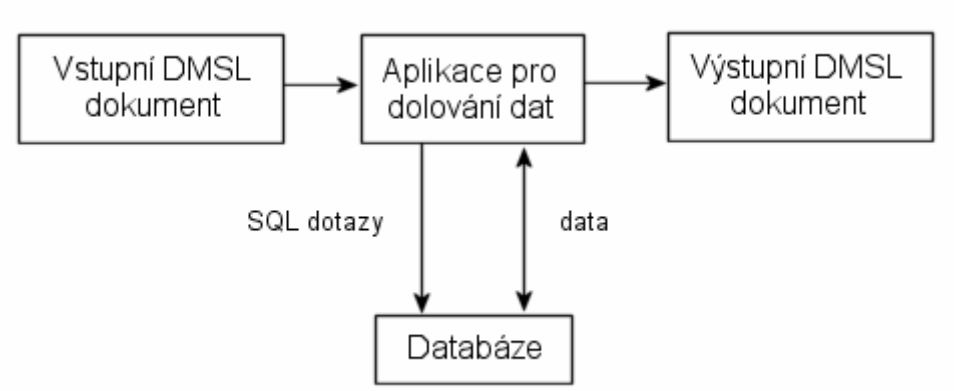
5 Návrh koncepce aplikace

5.1 Úvod

Jedním z hlavních úkolů této práce bylo implementovat vybranou metodu pro získávání znalostí z databází, tedy Bayesovskou klasifikaci. Při návrhu aplikace jsem se nechal inspirovat systémem pro získávání znalostí, který je vyvíjen na Fakultě informačních technologií Vysokého učení technického, a to konkrétně moduly pro aplikaci jednotlivých metod pro dolování dat. Více informací o systému pro získávání znalostí v diplomové práci [8] a o jednotlivých modulech v diplomových pracích [9], [10] a [11]. Návrh koncepce této aplikace je znázorněn na obrázku 5.1.

Vstup aplikace tvoří DMSL dokument, který obsahuje veškeré informace potřebné pro spuštění dolovací úlohy, jako je formát dat uložených v databázi, způsob připojení k databázi a samozřejmě definice samotné dolovací úlohy. Aplikace na základě těchto informací získá z databáze data, na kterých bude probíhat dolování, a zahájí proces dolování dat. V tomto případě tedy klasifikaci nových vzorků dat podle Bayesova teorému. Klasifikovaná data jsou poté uložena zpět do databáze, výsledky klasifikace jsou zpracovány a získané znalosti jsou uloženy do nového výstupního DMSL dokumentu.

Obrázek 5.1: Koncepce aplikace



5.2 Formát dat v databázi

Data určená pro dolování jsou rozdělena na 2 části, trénovací a testovací, které jsou uloženy v databázi ve dvou samostatných tabulkách. U obou těchto skupin dat je známa příslušnost do jednotlivých tříd, přičemž u obou tabulek se předpokládá, že mají stejnou strukturu a atribut reprezentující třídu je uváděn jako poslední. Výsledná data, u kterých je třída určena pomocí Bayesovská klasifikace, jsou vkládána do tabulky, která je vytvořena aplikací.

Aplikace předpokládá, že data uložená v databázi jsou již upravená a vhodná pro dolování. Není v této práci tedy uvažováno žádné předzpracování dat.

Data vhodná pro použitý algoritmus Bayesovské klasifikace jsou pouze diskrétního charakteru. Klasifikace totiž probíhá na základě hledání trénovacích vzorků patřících do jednotlivých tříd, jejichž hodnota daného atributu je shodná s hodnotou atributu vzorku. Tedy pokud atribut obsahuje celočíselné hodnoty nebo řetězce, klasifikace funguje korektně. Tento postup ale nelze aplikovat na spojité hodnoty atributů. Proto nelze např. použít data obsahující hodnoty s plovoucí řádovou čárkou, u kterých právě hledání stejných hodnot atributů v trénovacích datech není použitelné. Pro klasifikaci čísel s plovoucí řádovou čárkou by musel být použit jiný algoritmus klasifikace, který by místo hledání stejných hodnot atributů v datech používal Gaussovo normální rozložení jednotlivých atributů. Oba postupy jsou podrobně zmíněny výše v kapitole 3.2.2.

5.3 Formát DMSL dokumentu

Formát DMSL dokumentu, jeho syntaxe a sémantika jsou samozřejmě definovány v [6], jejich stručný popis je uveden výše v kapitole 4.3. Ale jelikož tato aplikace pracuje samostatně, musí být vstupní DMSL dokument vytvořen „ručně“ (v systému pro získávání znalostí tento dokument předává modul pro předzpracování dat modulu pro dolování dat), a proto musí být nejprve definován obsah tohoto dokumentu.

Datový model obsahuje formát vstupních dat, tedy strukturu tabulky, ve které jsou data uložena v databázi. Používaná data jsou, dle výše uvedeného předpokladu, již předzpracovaná a nejsou potřeba jejich další úpravy, a proto podle [6] je dolovací model prázdný. Doménová znalost není uvedena vůbec a struktura dolovací úlohy a znalostí byla uvedena výše v kapitole 4.3.2 definující rozšíření jazyka DMSL. Příklad DMSL dokumentu je uveden v příloze 1.

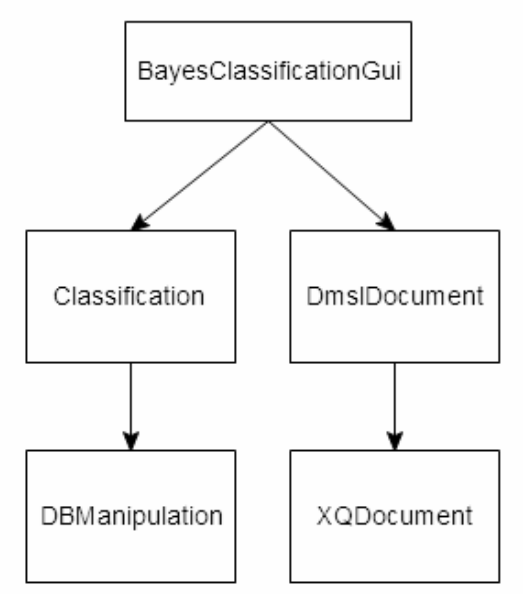
6 Implementace aplikace

6.1 Úvod

Jak již bylo uvedeno v předcházející kapitole, při návrhu koncepce aplikace jsem použil jako předlohu systém pro získávání znalostí vyvíjený na této fakultě. Tento systém je implementovaný v jazyce Java, a proto jsem pro implementaci této aplikace použil také objektově orientovaný jazyk Java, více o něm na [12]. Díky tomu bylo také možné použít knihovnu pro zpracování dokumentu v jazyce DMSL, která je právě v tomto systému pro získávání znalostí obsažena. Dále bylo nutné rozhodnout jaká databáze bude pro tuto aplikaci použita. Vybrána byla nejrozšířenější volně dostupná databáze MySQL, více o ní na [13].

Celá aplikace byla při implementaci rozdělena na několik větších samostatných částí. Prvním úkolem bylo načtení vstupního DMSL dokumentu, získání všech důležitých informací pro činnost aplikace z tohoto dokumentu a poté na konci vložení získaných znalostí do nového výstupního DMSL dokumentu. V další části bylo nutné, na základě údajů získaných ze vstupního dokumentu, navázat spojení s databází a umožnit vykonávat jednotlivé SQL dotazy pro získání potřebných dat z databáze. A poslední částí bylo provedení samotného dolování dat, tedy Bayesovské klasifikace. Jednotlivé třídy implementující tyto hlavní části jsou uloženy v balíčku mining, balíček xmlclasses obsahuje převzatou knihovnu pro práci s DMSL dokumentem. Diagram na obrázku 6.1 zobrazuje jednotlivé třídy a vztahy mezi nimi. Jejich podrobný popis je uveden v následujících kapitolách

Obrázek 6.1: Vztahy mezi jednotlivými částmi aplikace



6.2 Práce s DMSL dokumentem

Zpracování vstupního a vytvoření výstupního DMSL dokumentu značně usnadnila již vytvořená knihovna používaná jednotlivými dolovacími moduly systému pro získávání znalostí, který je vyvíjen na této fakultě. Tato knihovna byla původně vytvořena panem Stanislavem Mikuleckým a řešila spolupráci s XDMS dokumentem, který je také založen na značkovacím jazyku XML. Její dvě hlavní třídy jsou XQDocument, která zajišťuje práci s celým dokumentem, a abstraktní třída XQFragment, která slouží jako přímý nebo nepřímý předek veškerých tříd reprezentující jednotlivé elementy. Více podrobností o této práci se lze dozvědět v [14]. Knihovna byla poté upravena pro práci s DMSL dokumentem a velice zpřehledňuje a zlehčuje práci s těmito typy dokumentů.

Pro potřeby této práce bylo nutné upravit některé existující třídy a přidat nové třídy reprezentující jednotlivé elementy, které byly vytvořeny pro element DataMiningTask a Knowledge jako rozšíření jazyka DMSL. Práce s touto knihovnou je zapouzdřena v jediné třídě DmslDocument, která zajišťuje načtení vstupního dokumentu, poskytuje metody pro získání jednotlivých důležitých údajů ze tohoto dokumentu a pro vytvoření výstupního dokumentu se získanými znalostmi.

6.3 Práce s databází

Jedním z hlavních cílů autorů jazyka Java bylo vytvářet programový kód nezávislý na dané platformě a tedy snadno přenositelný. A proto pro práci s databázemi byl cíl stejný. Výsledkem je rozhraní JDBC (Java DataBase Connectivity), které bylo navrženo pro zjednodušení práce s datovými zdroji, konkrétně tedy s databázemi. Toto rozhraní je nezávislé na použité platformě a tedy není nutné přesně vědět s jakou databází se bude pracovat, samozřejmě se ale nesmí používat volání speciální pro určitou platformu a je nutné používat standardizovaný databázový jazyk SQL. Pro dosažení této nezávislosti je rozhraním JDBC poskytován tzv. správce ovladačů (driver manager), který dynamicky spravuje objekty ovladačů, vyžadované použitými databázovými programy. Každá databáze musí mít svůj vlastní ovladač (driver). Jejich objekty jsou samočinně se registrující objekty, které se registrují společně se správcem ovladačů v době zavádění do paměti.

Samotná práce s databází probíhá voláním jednotlivých metod, které odpovídají logickým operacím vykonávaným při získávání dat z databáze jako jsou: připojení k databázi, vytvoření příkazu a vykonání dotazu a prohledání výsledné datové sady. Pro připojení k databázi potřebujete ještě kromě jejího ovladače také tzv. databázové URL. Tato adresa se skládá z použitého protokolu, v tomto případě tedy protokolu JDBC, dále z tzv. podprotokolu, což je název ovladače nebo název mechanismu propojitelnosti databází, a nakonec z identifikátoru databáze. V případě, že se potřebujete k databázi připojit prostřednictvím sítě, obsahovala by databázová URL jako první informaci identifikující vzdálený počítač. Více informací o rozhraní JDBC naleznete v dokumentaci k jazyku Java, která je k dispozici na [12], a nebo v [16].

Pro tuto aplikaci je práce s databází velmi důležitá, veškerá data se kterými se pracuje jsou uložena právě v databázi. Třída zajišťující správnou komunikaci s databází se nazývá DBManipulation. Tato třída umožňuje na základě údajů získaných ze vstupního DMSL dokumentu, kterými jsou databázová URL, ovladač databáze, uživatelské jméno a heslo, připojení k databázi a voláním jejích metod vykonávání SQL dotazů nad příslušnými tabulkami databáze.

6.4 Průběh samotné klasifikace

Algoritmus Bayesovské klasifikace je implementován ve třídě Classification. Tato třída obsahuje soukromé datové členy pro uložení základních informací, získaných ze vstupního DMSL dokumentu, které jsou potřebné pro spuštění dolovací úlohy. Jsou to například databázová URL, ovladač databáze, struktura dat v databázi a jména jednotlivých tabulek s daty. Ostatní soukromé datové členy slouží pro usnadnění průběhu klasifikace a nebo pro uložení některých výsledků klasifikace – úspěšnost klasifikace a jiných.

Bayesovská klasifikace a její teoretický průběh jsou podrobně popsány výše v kapitole 3.2., proto zde konkrétní implementaci popíšu pouze stručně. Samotná klasifikace se spustí zavoláním metody runClassification(). Nejprve je vytvořen nový objekt třídy DBManipulation, pro umožnění práce s databází. Dále jsou postupně po jednom vybírány datové vzorky z testovacích dat a pro ně je určována pravděpodobnost s níž přísluší do jednotlivých tříd, tedy jsou počítány pravděpodobnosti výskytu hodnot jejich atributů v trénovacích datech. Tyto pravděpodobnosti jsou poté porovnány a vzorku je přiřazena vypočtená třída a je vložen do databáze do tabulky pro klasifikovaná data. Zároveň je ještě určeno, zdali byl vzorek klasifikován do správné třídy. Může nastat ale i případ, kdy pro nedostatek trénovacích dat, datový vzorek patří do všech tříd s nulovou pravděpodobností a je tak zařazen do skupiny vzorků, které nebylo možno klasifikovat. Tento cyklus se opakuje dokud testovací data obsahují nějaké vzorky. Poté je na základě celkového počtu a počtu nesprávně klasifikovaných vzorků určena procentuální úspěšnost tohoto klasifikátoru, tedy jak je spolehlivý a zdali může být pro tento typ dat použit.

Pro samotnou klasifikaci, tedy pro výpočet pravděpodobnosti výskytu hodnot jednotlivých atributů datových vzorků v trénovacích datech, je přesně určeno podle kterých atributů, sloupců tabulky, má klasifikace probíhat. Tato informace je uložena ve vstupním DMSL dokumentu v elementu MiningSchema. Tudíž lze dle potřeby vypustit určité atributy, pokud nechceme, aby ovlivnili výsledek klasifikace.

6.5 Spuštění aplikace

Hlavní třídou této aplikace je třída BayesClassificationGui. Tato třída vytváří jednoduché uživatelské rozhraní, pomocí kterého jsou zadávány názvy vstupního a výstupního DMSL dokumentu a poté

zobrazeny výsledky klasifikace. Při spuštění samotné klasifikace jsou nejprve vytvořeny nové instance dvou základních tříd DmslDocument a Classification. Pomocí objektu třídy DmslDocument jsou získány údaje ze vstupního DMSL dokumentu a ty jsou předány objektu třídy Classification, kde jsou uloženy do soukromých datových členů. Nyní následuje již jen zavolání metody runClassification() třídy Classification. Nakonec jsou získané znalosti uloženy do výsledného DMSL dokumentu a klasifikace je ukončena. Dokud není ukončena aplikace je možné klasifikaci libovolně opakovat, vždy je pouze nutné zadat korektní vstupní DMSL dokument a název výstupního DMSL dokumentu.

Jako vývojové prostředí bylo použito NetBeans IDE 5.5, které kompiluje celý projekt přímo do archivu jar, a proto aplikaci lze spustit dvěma způsoby. Buď přímo z archivu jar a nebo zkompilovat třídu BayesClassificationGui a tu poté spustit. Spuštění aplikace je podrobně popsáno v uživatelské příručce, která je spolu se zdrojovými texty uložena na přiloženém CD.

7 Ověření funkčnosti aplikace

7.1 Testovací data

Pro testování funkčnosti Bayesovského klasifikátoru byl vybrán soubor dat ze studie nazvané Národní Indonéský průzkum rozšířenosti antikoncepce (National Indonesia Contraceptive Prevalence Survey) prováděné v roce 1987. Cílem této studie bylo u vdaných žen, které nebyly v době průzkumu těhotné nebo o svém těhotenství v té době ještě nevěděli, určit na základě jejich demografických a socio-ekonomických charakteristik jejich aktuální používanou antikoncepční metodu. Data byla získána z této adresy [17].

Jednotlivé datové vzorky mají 10 atributů, včetně atributu obsahujícího příslušnost k dané třídě, a všechny nabývají celočíselných hodnot. Třídy, do kterých mohou být data klasifikována jsou tři. Struktura dat je uvedena v příloze 2. Data je možné přímo bez úprav použít pro dolování, jelikož neobsahují žádné chybějící hodnoty.

7.2 Určení efektivity Bayesovského klasifikátoru

7.2.1 V závislosti na velikosti množiny trénovacích dat

Soubor testovacích dat obsahuje celkem 1473 datových vzorků. Prvním úkolem bude tedy určit vliv množství trénovacích dat na úspěšnost klasifikátoru. Ze souboru dat vyčleníme 294 vzorků, tedy přesně 20%, které budou sloužit jako testovací data pro klasifikátor. Ze zbylých dat budeme postupně vybírat trénovací data a ověřovat efektivitu klasifikace. U všech dat je známa příslušnost do jednotlivých tříd, a proto stačí pouze porovnat výsledek klasifikace pro testovací data s původními hodnotami v databázi. Ve všech následujících případech budeme pro klasifikaci používat všechny atributy, tedy samozřejmě kromě toho, který obsahuje informaci do jaké třídy vzorek patří.

Postupně byly provedeny 4 testy, kdy bylo postupně použito jako trénovací data 10%, 25% a 50% ze zbylých dat a nakonec byla použita všechna zbylá data jako trénovací. Výsledky jednotlivých testů jsou zřejmé z níže uvedených tabulek.

Tabulka 7.1: Použito 10% dat jako trénovací

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	117	115	72	82	25

Efektivita klasifikátoru: 42,18%.

Tabulka 7.2: Použito 25% dat jako trénovací

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	294	120	88	80	6

Efektivita klasifikátoru: 42,86%.

Tabulka 7.3: Použito 50% dat jako trénovací

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	589	117	97	79	1

Efektivita klasifikátoru: 48,64%.

Tabulka 7.4: Použita všechna zbylá data jako trénovací

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	1179	113	98	83	0

Efektivita klasifikátoru: 47, 28%.

Původní předpoklad, že s rostoucím množstvím trénovacích dat bude přesnost klasifikátoru vetší, je z výsledků těchto testů celkem patrný. Nejvýrazněji je tato skutečnost vidět na počtu vzorků, které nebylo možné klasifikovat. Pro malé množství trénovacích dat, použitých v prvním testu, tyto neklasifikované vzorky tvořily téměř 10% všech klasifikovaných dat, ale v dalších testech byl jejich počet již zanedbatelný. Úspěšnost klasifikace celkově není sice příliš velká, ale to již není způsobeno množstvím použitých trénovacích dat.

7.2.2 V závislosti na počtu použitých atributů pro klasifikaci

Soubor testovacích dat zůstává stejný jako při předchozích testech. Opět použijeme 20% dat jako testovací a ze zbylých dat vybereme 50% dat jako trénovací. Pro toto množství trénovacích dat vyšli v předchozích testech nejlepší výsledky a bude tedy možné porovnat vliv počtu atributů použitých při klasifikaci.

Testy byly opět provedeny 4. Nejprve byl pro klasifikaci použit pouze jediný atribut a jelikož výsledkem klasifikace má být používaná antikoncepční metoda, zvolil jsem atribut obsahující počet dětí. V dalším testu byly datové vzorky klasifikovány podle atributů udávající věk ženy a počet dětí. Pro následující test byly ke stávajícím dvěma přidány ještě další dva atributy a to vzdělání ženy a index životní úrovně. A pro poslední test bylo vybráno těchto šest atributů pro klasifikaci: věk ženy,

její vzdělání, počet dětí, náboženství, jestli je zaměstnaná a povolání manžela. Strukturu dat pro lepší pochopení je možné nalézt v příloze 2. Výsledky těchto testů jsou uvedeny v následujících tabulkách.

Tabulka 7.5: Klasifikace podle jediného atributu

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	589	135	3	155	1
Použité atributy			počet dětí			

Efektivita klasifikátoru: 44,90%.

Tabulka 7.6: Klasifikace podle dvou atributů

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	589	145	33	115	1
Použité atributy		věk ženy			počet dětí	

Efektivita klasifikátoru: 49,66%.

Tabulka 7.7: Klasifikace podle čtyř atributů

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	589	127	69	97	1
Použité atributy	věk ženy		vzdělaní ženy	počet dětí		životní úroveň

Efektivita klasifikátoru: 47,28%.

Tabulka 7.8: Klasifikace podle šesti atributů

	Testovací data	Trénovací data	Klasifikovaná data			
			Třída 1	Třída 2	Třída 3	Neklasifikováno
Počet vzorků	294	589	123	72	98	1
Použité atributy	věk ženy	vzdělání ženy	počet dětí	náboženství ženy	je žena zaměstnaná	povolaní manžela

Efektivita klasifikátoru: 50,68%.

Výsledky těchto testů jsou poměrně zajímavé. I použitím pouze jediného atributu se úspěšnost klasifikace nijak výrazně nesnížila v porovnáním s předchozími testy, kde byly použity atributy všechny. Tento údaj může být ale celkem zavádějící, při porovnání počtu vzorků klasifikovaných do jednotlivých tříd je patrná určitá nerovnoměrnost. Ta se velmi projeví při použití atributu, který může nabývat pouze omezeného počtu hodnot, nejvíce u tzv. dvouhodnotového (binárního) atributu.

V tomto případě je většina vzorků zařazena pouze do jediné třídy, v krajním případě i všechny, a to do té třídy, jejíchž vzorků je nejvíce v trénovacích datech.

Další poznatek vycházející z těchto výsledků testů je, že vhodným výběrem atributů určených pro klasifikaci je možné dosáhnout lepší přesnosti než použitím všech atributů. Tato skutečnost je patrná zejména u posledního testu, kde bylo použito pouze šest atributů a úspěšnost klasifikátoru je vyšší než při použití všech atributů. Ale i výsledek předposledního testu se čtyřmi atributy je poměrně dobrý. Z tohoto tedy vyplývá, že některé atributy mohou mít větší vliv na výslednou třídu, obsahují tedy pro zkoumaný problém významnější informace. Naopak některé atributy obsahují méně podstatné informace a mohou při jejich použití úspěšnost klasifikace snižovat.

7.3 Zhodnocení výsledků testů

Z výsledků výše uvedených testů vyplývá, že maximální přesnost klasifikátoru pro použitá testovací data se pohybuje kolem 50%, což není poměrně vysoká úspěšnost klasifikace. Příčiny těchto výsledků jsou nejspíše způsobeny tím, že data jsou poměrně nerovnoměrně rozložena do jednotlivých tříd, a celkově obsahují velké množství vzorků, které svými hodnotami z jednotlivých tříd vybočují. Tato data nejsou tedy příliš vhodná pro použitou klasifikaci, neboli Bayesovská klasifikace by nebyla nejspíše vybrána pro klasifikaci těchto dat, bez jejich určitých úprav.

Funkčnost a základní principy Bayesovské klasifikace byly ale i na těchto datech ověřeny. Pro srovnání jsou ještě v následující kapitole uvedeny 2 testy na datech vhodných pro Bayesovskou klasifikaci.

7.4 Efektivita Bayesovského klasifikátoru při použití vhodných dat

7.4.1 Použitá data

Pro potvrzení závěrů vyplývajících z předcházejících testů byla použita další testovací data, která jsou vhodná přímo pro klasifikaci. Tato nová data reprezentují stav jednoduchého segmentového LED displeje. Jednotlivé datové vzorky se skládají celkově z 8 atributů, prvních 7 atributů odpovídá diodám tvořícím segmentový displej a poslední atribut určuje příslušnost do dané třídy. Tříd je deset a odpovídají základním číslicím, které je možné zobrazit na segmentovém LED displeji. Data byla opět získána na adrese [17] a jejich struktura je uvedena v příloze 3.

Jelikož diody se mohou nacházet ve dvou stavech – svítí, nesvítí, a pouze na základě této informace jsme schopni určit, jaká číslice bude na displeji zobrazena, nebylo by potřeba vůbec používat jakoukoliv klasifikaci pro určení právě zobrazené číslice. Ale v důsledku vstupního šumu, neboli

pravděpodobnosti, že hodnota každého atributu může být invertována, nelze pouze na základě informace, které diody svítí, určit právě zobrazené číslo. Vstupní šum se může projevit i na atributu reprezentujícím třídu. Data je možné přímo bez úprav použít, protože neobsahují žádné chybějící hodnoty.

7.4.2 Výsledky testů

Pravděpodobnost, že hodnoty atributů byly invertovány, byla u dat použitých při testech 5% a 10%. Pro oba testy bylo použito 200 vzorků jako testovací data a 500 vzorků jako trénovací data. V obou případech bylo použito při klasifikaci všech 7 atributů reprezentující jednotlivé diody. Výsledky těchto testů jsou uvedeny v následujících tabulkách.

Tabulka 7.9: Klasifikace dat s 5% pravděpodobností, že hodnoty atributů byly invertovány

	Testovací data	Trénovací data	Klasifikovaná data										
			Zobrazená číslice										Neklasifikováno
			0	1	2	3	4	5	6	7	8	9	
Počet vzorků	200	500	24	15	21	26	22	18	20	15	13	26	0

Efektivita klasifikátoru: 87,00%.

Tabulka 7.9: Klasifikace dat s 10% pravděpodobností, že hodnoty atributů byly invertovány

	Testovací data	Trénovací data	Klasifikovaná data										
			Zobrazená číslice										Neklasifikováno
			0	1	2	3	4	5	6	7	8	9	
Počet vzorků	200	500	24	18	20	22	18	16	20	20	22	20	0

Efektivita klasifikátoru: 76,50%.

Z výsledků těchto testů je patrné, že při použití vhodných dat dosahuje úspěšnost Bayesovské klasifikace velmi dobrých hodnot. Tímto je tedy potvrzen předcházející závěr, že pro klasifikaci dat použitých v původních testech nebyla Bayesovská klasifikace nejvhodnější.

8 Závěr

Tato bakalářská práce se zabývá problematikou získávání znalostí z databází a to konkrétně metodou Bayesovské klasifikace. Cílem bylo implementovat vybranou metodu dolování dat a její funkčnost ověřit na vybraném vzorku dat.

Výsledkem této práce je aplikace provádějící dolování dat pomocí Bayesovské klasifikace (datové vzorky jsou klasifikovány do tříd na základě pravděpodobnosti). Aplikace pracuje na stejném principu jako moduly systému pro získávání znalostí, vyvíjeném na Fakultě informačních technologií Vysokého učení technického, které implementují jednotlivé metody pro dolování dat.

Aplikace využívá výhod komplexního jazyka pro popis celého dolovacího procesu, kterým je jazyk DMSL. Ten musel být nejprve rozšířen o elementy pro popis Bayesovské klasifikace a získaných znalostí. Veškeré informace důležité pro běh aplikace jsou uloženy v DMSL dokumentu sloužícím jako vstupní konfigurační soubor. Získané znalosti jsou poté uloženy do výstupního DMSL dokumentu.

Získávání znalostí je možné provádět na datech obsahujících celočíselné nebo řetězcové hodnoty, která jsou uložena v databázi. Pro samotné dolování dat lze přesně určit, které atributy budou použity při klasifikaci nových datových vzorků do tříd. Funkčnost aplikace byla ověřena na dvou různých souborech testovacích dat, výsledky těchto testů jsou uvedeny v kapitole 7.

Možnost zlepšení této práce vidím v upravení algoritmu Bayesovské klasifikace tak, aby bylo možné používat tuto aplikaci i pro klasifikaci dat obsahujících hodnoty s plovoucí řádovou čárkou a zároveň i pro dolování nad velkým množstvím dat. A jelikož je tato aplikace implementována tak, že tvoří v podstatě funkční jádro modulu již zmíněného systému pro získávání znalostí, je dalším návrhem pro vývoj této práce její použití pro vytvoření nového modulu právě do tohoto systému pro získávání znalostí.

Literatura

[1] BERKA, Petr. *Dobývání znalostí z databází.* Praha : Academia, 2003. 366 s. ISBN 80-200-1062-9.

[2] HAN, Jiawei, KAMBER, Micheline. *Data mining : Concepts and Techniques*. 2nd edition. San Francisco : Morgan Kaufmann Publishers, 2006. 770 s. ISBN 1-55860-901-6.

[3] FAJMON, Břetislav, RŮŽIČKOVÁ, Irena. *Matematika 3*. Elektronický učební text. Brno, UMAT FEKT VUT v Brně, 2004.

[4] BOULICAUT, Jean-Francois, MASSON, Cyrille. *The Data Mining and Knowledge Discovery Handbook*. Springer, 2005. Dostupný na URL: <http://liris.cnrs.fr/~jboulica/>. ISBN 715-727. Data mining query languages, s. 13.

[5] SABALA, Michal. *Data Mining Group* [online]. 1998 [cit. 2007-05-08]. Dostupný na URL: <http://www.dmg.org/>.

[6] KOTÁSEK, Petr. *DMSL : The Data Mining Specification Language* [dizertační práce]. c2003. 179 s. Vysoké učení technické, Fakulta informačních technologií, Brno. Dostupný na URL: <http://www.fit.vutbr.cz/~kotasekp/dmsl/>.

[7] MICHELE, Jan. *Předzpracování dat pro dolování* [diplomová práce]. 2004. Vysoké učení technické, Fakulta informačních technologií, Brno.

[8] HROMČÍK, Petr. *Systém pro získávání znalostí z databází* [diplomová práce]. 2003. Vysoké učení technické, Fakulta informačních technologií, Brno.

[9] STRYKA, Lukáš. *Modul dolování asociačních pravidel* [diplomová práce]. 2003. Vysoké učení technické, Fakulta informačních technologií, Brno.

[10] ŠEVČÍKOVÁ, Lenka. *Modul deskriptivního dolování a klasifikace* [diplomová práce]. 2003. Vysoké učení technické, Fakulta informačních technologií, Brno.

[11] FORGÁČ, Michal. *Modul pro shlukovou analýzu* [diplomová práce]. 2006. Vysoké učení technické, Fakulta informačních technologií, Brno.

[12] Sun Microsystems. *Sun Developer Network* [online]. c1994-2007 [cit. 2007-05-08]. Dostupné na URL: <http://java.sun.com/>.

[13] MySQL AB. *MySQL* [online]. c1995-2007 [cit. 2007-05-08]. Dostupné na URL: <http://www.mysql.com/>.

[14] MIKULECKÝ, Stanislav. *Dolovací jazyk v XML pro získávání znalostí z databází* [diplomová práce]. 2001. Vysoké učení technické, Fakulta informačních technologií, Brno.

[15] ECKEL, Bruce. *Myslíme v jazyku Java : knihovna programátora*. Bogdan Kiszka. Praha : Grada Publishing, 2000. 432 s. ISBN 80-247-9010-6.

[16] ECKEL, Bruce. *Myslíme v jazyku Java : knihovna zkušeného programátora*. Bogdan Kiszka. Praha : Grada Publishing, 2000. 472 s. ISBN 80-247-0027-1.

[17] NEWMAN, D.J., et al. *UCI Repository of machine learning databases* [online]. 1998 [cit. 2007-05-08]. Dostupné na URL: <http://www.ics.uci.edu/~mlearn/MLRepository.html>.

Seznam příloh

Příloha 1. Ukázka výsledného DMSL dokumentu

```
<DMSL>
  <FunctionPool name="default">
  </FunctionPool>
  <DataModel functionPoolRef="default" name="default">
    <DataMatrix name="default">
      <DataField name="w_age">
        <FieldProperties atomicity="scalar" dataType="integer">
        </FieldProperties>
      </DataField>
      <DataField name="w_education">
        <FieldProperties atomicity="scalar" dataType="integer"
               granularity="discreteInfinite">
        </FieldProperties>
      </DataField>
      <DataField name="h_education">
        <FieldProperties atomicity="scalar" dataType="integer"
               granularity="discreteInfinite">
        </FieldProperties>
      </DataField>
      <DataField name="children">
        <FieldProperties atomicity="scalar" dataType="integer">
        </FieldProperties>
      </DataField>
      <DataField name="w_religion">
        <FieldProperties atomicity="scalar" dataType="integer"
               granularity="binary">
        </FieldProperties>
      </DataField>
      <DataField name="w_employed">
        <FieldProperties atomicity="scalar" dataType="integer"
               granularity="binary">
        </FieldProperties>
```

```
    </DataField>
    <DataField name="h_occupation">
      <FieldProperties atomicity="scalar" dataType="integer"
             granularity="discreteInfinite">
      </FieldProperties>
    </DataField>
    <DataField name="sol_index">
      <FieldProperties atomicity="scalar" dataType="integer"
             granularity="discreteInfinite">
      </FieldProperties>
    </DataField>
    <DataField name="media_exposure">
      <FieldProperties atomicity="scalar" dataType="integer"
             granularity="binary">
      </FieldProperties>
    </DataField>
    <DataField name="contraceptive">
      <FieldProperties atomicity="scalar" dataType="integer"
             granularity="discreteInfinite">
      </FieldProperties>
    </DataField>
  </DataMatrix>
</DataModel>
<DataMiningModel dataModelRef="default" name="default">
</DataMiningModel>
<DataMiningTask dataMiningModelRef="default" name="default"
       type="bayesClassification">
  <MineBayesClassification name="default">
    <MiningSchema name="default">
      <MiningField dataFieldRef="w_age" name="wifesAge">
      </MiningField>
      <MiningField dataFieldRef="w_education" name="wifesEducation">
      </MiningField>
      <MiningField dataFieldRef="h_education" name="husbandsEducation">
      </MiningField>
      <MiningField dataFieldRef="children" name="numberOfChildren">
      </MiningField>
      <MiningField dataFieldRef="w_religion" name="wifesReligion">
      </MiningField>
      <MiningField dataFieldRef="w_employed" name="wifeIsEmployed">
      </MiningField>
```

```
      <MiningField dataFieldRef="h_occupation"
             name="husbandsOccupation">
      </MiningField>
      <MiningField dataFieldRef="sol_index" name="standardOfLiving">
      </MiningField>
      <MiningField dataFieldRef="media_exposure" name="mediaExposure">
      </MiningField>
      <TargetMiningField dataFieldRef="contraceptive"
             name="contraceptiveMethod">
        <FieldValue name="no_use" value="1">
        </FieldValue>
        <FieldValue name="long_term_methods" value="2">
        </FieldValue>
        <FieldValue name="short_term_methods" value="3">
        </FieldValue>
      </TargetMiningField>
    </MiningSchema>
    <Database dbUrl="jdbc:mysql://localhost/cmc"
          driver="com.mysql.jdbc.Driver" name="cmc" password="root"
          userName="root">
      <Table name="training" value="training_data_10">
      </Table>
      <Table name="testing" value="testing_data_10>
      </Table>
      <Table name="classified" value="classified_data_10">
      </Table>
    </Database>
  </MineBayesClassification>
</DataMiningTask>
<Knowledge dataminingTaskRef="default" name="default"
       type="bayesClassification">
  <BayesOutput dataFieldRef="contraceptive" fieldName="contraceptive">
    <TargetValueCounts>
      <TargetValueCount count="115" value="1">
      </TargetValueCount>
      <TargetValueCount count="72" value="2">
      </TargetValueCount>
      <TargetValueCount count="82" value="3">
      </TargetValueCount>
      <TargetValueCount count="25" value="notClassified">
      </TargetValueCount>
```

```
      </TargetValueCounts>
    </BayesOutput>
  </Knowledge>
</DMSL>
```

Příloha 2. Struktura použitých testovacích dat

Datové vzorky mají 10 atributů včetně atributu označujícího třídu a všechny jsou spolu s hodnotami, kterých mohou nabývat, uvedeny v následující tabulce.

Atribut	Typ	Hodnoty	Odpovídající sloupec databázové tabulky
věk ženy	celočíselný		w_age
vzdělání ženy	kategorický	1 = nízké, 2, 3, 4 = vysoké	w_education
vzdělání manžela	kategorický	1 = nízké, 2, 3, 4 = vysoké	h_education
počet dětí	celočíselný		children
náboženství ženy	binární	0 = jiné, 1 = Islám	w_religion
je žena zaměstnaná	binární	0 = ano, 1 = ne	w_employed
zaměstnání manžela	kategorický	1 = špatné, 2, 3, 4 = dobré	h_occupation
index životní úrovně	kategorický	1 = nízká, 2, 3, 4 = vysoká	sol_index
vliv médií	binární	0 = dobrý, 1 = špatný	media_exposure
metoda používané antikoncepce	určuje třídu	1 = nepoužívá 2 = dlouhodobá 3 = jednorázová	contraceptive

Příloha 3. Struktura dat vhodných pro klasifikaci

Datové vzorky mají 8 atributů včetně atributu označujícího třídu a všechny jsou spolu s hodnotami, kterých mohou nabývat, uvedeny v následující tabulce.

Atribut	Typ	Hodnoty	Odpovídající sloupec databázové tabulky
dioda 0	binární	0 = nesvítí, 1 = svítí	diode_0
dioda 1	binární	0 = nesvítí, 1 = svítí	diode_1
dioda 2	binární	0 = nesvítí, 1 = svítí	diode_2
dioda 3	binární	0 = nesvítí, 1 = svítí	diode_3
dioda 4	binární	0 = nesvítí, 1 = svítí	diode_4
dioda 5	binární	0 = nesvítí, 1 = svítí	diode_5
dioda 6	binární	0 = nesvítí, 1 = svítí	diode_6
číslice zobrazená na displeji	určuje třídu	0 = číslice 0 1 = číslice 1 2 = číslice 2 3 = číslice 3 4 = číslice 4 5 = číslice 5 6 = číslice 6 7 = číslice 7 8 = číslice 8 9 = číslice 9	displayed_number