VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE

FAKULTA INFORMATIKY A STATISTIKY

Diplomová práce

Systém předzpracování dat pro dobývání znalostí z databází

Vypracovala: Hana Kotinová

Vedoucí práce: prof. Ing. Petr Berka, CSc.

Praha 2011

Abstrakt

Cílem diplomové práce bylo vytvoření aplikace pro předzpracování dat, pracující se soubory ve formátu csv. Aplikaci lze využít při přípravě dat pro data miningové úlohy. Aplikace byla vytvořena pomocí programovacího jazyka Java. Tento text obsahuje výklad problematiky související s předzpracováním dat, popis používaných algoritmů, informace o podobných o systémech Mining Mart and SumatraTT a popis vytvořené aplikace.

Abstract

Aim of this diploma thesis was to create an aplication for data preprocessing. The aplication uses files in csv format and is useful for preparing data while solving datamining tasks. The aplication was created using the programing language Java. This text discusses problems, their solutions and algorithms associated with data preprocessing and discusses similar systems such as Mining Mart and SumatraTT. A complete aplication user guide is provided in the main part of this text.

Prohlášení

Prohlašuji, že jsem diplomovou práci Systém předzpracování dat pro dobývání znalostí z databází zpracovala samostatně a že jsem uvedla všechny použité prameny a literaturu, ze kterých jsem čerpala.

V Praze dne 14.7. 2011

…….

Hana Kotinová

Obsah

Úvod

Cílem této diplomové práce byl vývoj aplikace nazvané Data preprocessing tool (dále jen DPT). Aplikace DPT bude sloužit jako nástroj pro předzpracování dat uložených v textovém formátu, a to v souborech typu csv. Pro její vývoj byl využit programovací jazyk Java, konkrétně Java 5 a 6. Tento text obsahuje X kapitol. V kapitole Dobývání znalostí z databází je nastíněna problematika dobývání znalostí z databází, jíž je předzpracování dat součástí. V následující kapitole, Problémy fáze předzpracování jsou prezentovány základní problémy, se kterými se můžeme setkat při zpracování dat, a to nejen pro data mining. Další kapitola popisuje algoritmy, které můžeme využít při předzpracování dat, a to zejména ty, které jsou implementovány v aplikaci DPT. Následující kapitola popisuje vybrané systémy pro předzpracování dat, konkrétně Mining Mart a Sumatra TT. Poté následuje kapitola o samotné aplikaci DPT, obsahující návod pro práci s aplikací. Předposlední kapitolu tvoří zdrojové kódy, které jsou k dispozici i na přiloženém CD. Poslední kapitolu tvoří závěrečné shrnutí.

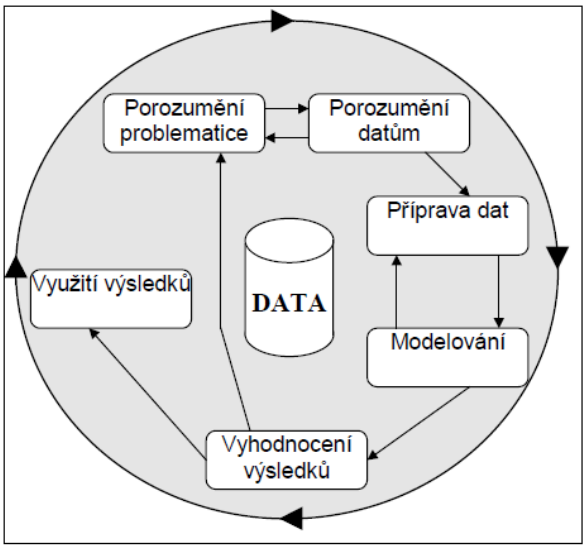
Dobývání znalostí z databází

Co je dobývání znalostí z databází

Dobývání znalostí z databází, jinak též data mining, můžeme definovat jako netriviální získávání implicitních, dříve neznámých a potenciálně užitečných informací z dat [1]. Historie této činnosti možná není příliš dlouhá, nicméně pro celý proces již bylo vyvinuto několik metodik a postupů, jako např. metodika 5A, SEMMA nebo CRISP-DM. Posledně jmenovaná metodika dělí dobývání znalostí z databází na několik fází, a to:

- Porozumění problematice
- Porozumění datům
- Příprava dat
- Modelování
- Vyhodnocení výsledků
- Využití výsledků

Tyto fáze na sebe nenavazují jen v přímé linii jdoucí od začátku do konce, ale tvoří kruh. Tím je naznačena povaha celého procesu dobývání znalostí z databází. Je možné a někdy dokonce nutné se vracet k fázím předchozím. Ve fázi modelování můžeme zjistit, že bychom potřebovali mít v datech nějakou další informaci, po vyhodnocení výsledků můžeme lépe porozumět dané problematice a díky tomu budeme chtít znát odpověď na další otázky atp. To vše ukazuje i následující obrázek:



Obr. 1 – Metodika CRISP-DM

Na mnoho druhů lidské činnosti platí tzv. pravidlo 80/20, nebo-li fakt, že určitá malá část činnosti zabere velké množství času, či naopak že drobná úprava vizuální stránky může mít obrovský vliv na výsledné přijetí či nepřijetí projektu. Toto pravidlo se tak týká i dobývání znalostí z databází, kde můžeme počítat s tím, že nejvíce času obvykle spotřebuje příprava dat.

Data jsou obvykle uložena v databázi či v datovém skladu, tj. v soustavě tabulek, ze kterých je potřeba vytvořit tabulku jedinou, vhodnou pro analýzu data miningovými procedurami. Data ovšem obvykle nevznikají pro jejich analýzu, jsou výsledkem různých procesů (výrobních, obchodních, administrativních, zjišťovacích, ...) a pochází z různých zdrojů, programových systémů, nachází se v různých formátech. Data mohou obsahovat velké množství chyb, důležité údaje mohou chybět, naopak mnoho údajů může být k dispozici duplicitně. Problémem může být i obrovské množství údajů – jak v počtu objektů (řádků v tabulkách), tak v počtu atributů (sloupců). Tento problém může narůst, např. v okamžiku, kdy zkoumáme údaje z tabulek v relaci 1:N. S tím vším se musíme vyrovnat v rámci fáze předzpracování.

Vzhledem k množství problémů, se kterými se ve fázi předzpracování potýkáme, je potřeba počítat s tím, že budeme proces přípravy dat několikrát opakovat.

Obchodní a jiné otázky

Řešíme-li nějakou data miningovou úlohu, obvykle se snažíme najít odpověď na nějakou otázku z oblasti obchodu a podnikání, nebo z oblasti lékařské. Chceme znát odpovědi na otázky jako:

- kteří zákazníci se od nás chystají odejít ke konkurenci
- které produkty si zákazníci kupují obvykle společně s jinými produkty
- kolik lidí se může chtít ubytovat v našem hotelu v létě příštího roku
- jaká je pravděpodobnost, že dostaneme zpět peníze, které jsme půjčili
- které faktory ovlivňují nejvíce prodej našich produktů
- zda existuje vztah mezi věkem a výší cholesterolu
- které faktory mají největší vliv na vznik rakoviny

Tyto otázky jsou prvotním impulzem, který nastartuje analytický proces. Jeho příštím krokem bude snaha porozumět tomu, co přesně chceme zjistit. Ne všechny otázky jsou dostatečně přesné, resp. je možné je zodpovědět několika způsoby. Zajímají nás tři nejdůležitější faktory, nebo bychom chtěli znát všechny, jejichž podíl je aspoň 5 procent? Chceme znát jen dvojice produktů v nákupním košíku nebo obecně n-tice? Po vyjasnění takovýchto otázek nás může dále zajímat, zda půjde o jednorázovou analýzu, či zda budeme chtít znát odpověď na naši otázku v pravidelných intervalech.

Typické úlohy

Aktivity, pomocí kterých budeme hledat odpovědi na naše otázky, můžeme rozdělit do několika oblastí. Jsou to:

- Klasifikace
- Odhady
- Seskupování a asociační pravidla
- Shlukování
- Popis a vizualizace

První dva patři mezi tzv. přímý data mining, zbylé tři mezi nepřímý data mining.

Přímým data mining znamená, že naším cílem je klasifikovat, odhadnout či predikovat jednu konkrétní proměnnou, a to podle jiných proměnných, které máme také k dispozici. Jednou z typických úloh je otázka, zda klient určitého věku a pohlaví, s danou výší měsíčního příjmu získá od banky úvěr či nikoliv. Informace o poskytnutí úvěru je speciální proměnnou a všechny ostatní informace jsou vztahovány k hodnotám této proměnné. Přitom nemusí jít nutně pouze o takovouto bivalentní proměnnou. Cílový atribut, jak je tato proměnná nazývána, může nabývat téměř libovolný počet hodnot.

Nepřímý data mining má za cíl zjistit vzájemné vztahy mezi dostupnými proměnnými.

Klasifikace

Klasifikace je zkoumáním vlastností nějakého objektu, a následné rozhodnutí za zařazení do jedné z předem určených tříd. Vlastnosti objektů máme obvykle uspořádány v relační databázové tabulce a naším úkolem je provést update této tabulky, nebo-li přidat informaci o zařazení do příslušné třídy. Při řešení klasifikační úlohy tedy máme k dispozici dobře definované třídy a nějaký tréninkový soubor s daty, kde je rozdělení do tříd provedeno. Naším úkolem je vytvořit model, pomocí kterého budeme moci klasifikovat nové případy. U této úlohy je důležité, že:

- počet tříd je relativně malý
- třídy jsou předem známy
- každý objekt patří do nějaké třídy

Odhady

Zatímco klasifikace pracuje s diskrétním výstupem (např. úvěr Ano/Ne), u odhadů je výstupem spíše spojitá proměnná. Odhadujeme, jak velký bude měsíční příjem, výška člověka nebo třeba stav účtu. Nicméně některé otázky lze řešit z pohledu úlohy klasifikační i jako odhad. Příkladem může být Churn modeling, nebo-li zjišťování, kteří zákazníci pravděpodobně brzy přestanou být našimi zákazníky. Může se jednat o klasifikační úlohu –

zákazník odejde/neodejde, nebo o odhad doby, po kterou bude daná osoba naším zákazníkem. Tyto dvě otázky mohou být řešeny společně.

Seskupování a asociační pravidla

Seskupování využijeme při analýze nákupního košíku a hledání, které produkty obvykle zákazníci kupují společně. Umístění takovýchto produktů v těsné blízkosti může přinést nárůst tržeb. Využijeme je i pro definování cross-sellů, tj. produktů, které bychom mohli nabídnout zákazníkům, kteří již využívají nějaký jiný produkt.

Shlukování

Shlukování znamená uspořádání různorodých objektů do několika skupin, kde každá skupina bude obsahovat do určité míry podobné objekty. Na rozdíl od klasifikace, kde jsou skupiny předem dány, a kde máme příklady správného zařazení do skupin, při shlukování definici skupiny v podstatě teprve hledáme.

Popis a vizualizace

Pro řešení některých otázek někdy stačí dobře poznat data, která máme k dispozici. Pokud popis nepomůže přímo k zodpovězení naší otázky, určitě bude alespoň dobrým výchozím bodem, návrhem, kde začít odpověď hledat.

Příprava dat pro analýzu

Pro řešení úloh je potřeba data nejprve získat a připravit. Data mohou být uložena v různých formátech. Může se jednat o jednoduchý textový soubor, kde jsou jednotlivé hodnoty na řádku odděleny středníkem, tabulátorem či jiným znakem, případně jsou zarovnána do sloupců, nebo jde o jednu tabulku např. Microsoft Excel, může však jít o celou databázi či dokonce datový sklad. Různé systémy pracují s různými formáty, s různými typy souborů. Některé jsou flexibilní, a zpracují téměř libovolný soubor, jiné jsou specifické a vyžadují jeden konkrétní typ souboru, který dokáží zpracovat.

Problémy fáze předzpracování

V této kapitole budou popsány problémy, které se týkají předzpracování dat.

Formát dat

Text, obrázky, čísla i speciální struktury jakými jsou např. chemické sloučeniny, vyžadují odlišné struktury pro své uchovávání. Vezměme si jako příklad jednoduchou tabulku malé firmy s údaji o kontaktech na zákazníky, dodavatele či zaměstnance, se kterou pracujeme v tabulkovém kalkulátoru, např. Microsoft Excel. Taková tabulka bude obsahovat čistě textové údaje, jakým je jméno, údaje typu datum, a čísla. Ani pokud takovouto tabulku vytváří jedna osoba, nemáme zajištěno, že např. všechna jména budou napsána se správnou diakritikou, že telefonní čísla budou mít jednotný formát, případně i délku. Kromě toho si můžeme 2 pracovní a 2 soukromé telefony zapsat jak do čtyř samostatných sloupců, tak i např. do dvou řádků a dvou sloupců (např. protože díky tomu nemusíme rolovat). Datum lze zapsat částečně slovně, tj. např. 3. února, s rokem i bez roku, rok může být vyjádřen dvěma, či čtyřmi ciframi, někde se uvádí nejprve měsíc, někde den, ve tvaru 03/03/2003 mohou i nemusí být počáteční nuly. Některé systémy toto množství možností řeší stanovením počátečního dne a uváděním počtu dní, ev. počtu sekund od stanoveného začátku. První problém, kterému budeme čelit, je tedy jednotný formát dat. Protože data mohou pocházet z různých systémů, je potřeba zajistit, aby formát byl jednotný.

Statistická klasifikace proměnných

Statistika v analýzách rozlišuje následující typy proměnných:

- nominální
- ordinální
- intervalové
- poměrové
- kvantitativní

- kvalitativní
- spojité
- diskrétní
- binární
- symetrické
- asymetrické

Nominální proměnná nabývá několika hodnot, o kterých můžeme pouze říci, že jsou různé. Nemůžeme u nich určit pořadí důležitosti. To lze u ordinálních proměnných. U těchto však stále ještě nedokážeme určit, jak mnoho je jedna hodnota lepší, než jiná. Proměnné, kde lze určit o kolik je jedna hodnota lepší (čili rozdíl), jsou nazývány intervalové, a proměnné, kde lze určit kolikrát je jedna hodnota lepší (čili podíl), jsou nazývány poměrové.

Jiný pohled nazývá intervalovými proměnnými takové proměnné, které mohou nabývat hodnoty nula a poměrovými takové, které jsou vždy větší než nula.

Diskrétní jsou takové proměnné, které nabývají celočíselných hodnoty, zatímco spojité mohou nabýt libovolné číselné hodnoty.

Binární proměnná nabývá pouze dvou hodnot, a můžeme uvažovat binární proměnnou symetrickou, kde jsou obě hodnoty stejně kvalitní, nebo asymetrickou, kdy je jedna hodnota důležitější, než druhá (např. zda se pacient uzdravil či neuzdravil).

Mezi kvalitativní proměnné se řadí buď jen nominální, nebo nominální i ordinální proměnné. Kvantitativní proměnná je pak souhrnné označení proměnných diskrétních a spojitých.

Chybějící hodnoty

Samostatným problémem jsou chybějící údaje. Co lze dělat s chybějícími údaji? Na prvním místě si samozřejmě při zjištění chybějících údajů musíme říci, zda je účelné a také reálné se snažit chybějící údaje doplnit. Můžeme se setkat jak s řídce obsazeným atributem, kde bude vyplnění hodnoty vzácné, tak např. s odmítnutím odpovědi (věk či příjem v marketingových výzkumech), ale i s chybějícím údajem, který lze dopočítat z dalších atributů (údaje za čtvrtletí a za rok).

Možnými způsoby nahrazení chybějící hodnoty jsou:

- prosté nahrazení údajem „chybí, neuvedeno“ atp.
- nahrazení nejčetnější hodnotou
- proporcionální nahrazení hodnotami atributu
- náhodné nahrazení hodnotami atributu
- nenahrazujeme a celý řádek (objekt) ignorujeme
- v případě řídkých atributů můžeme spojit řídké atributy do jednoho (současně řeší problém velkého počtu atributů)

Z praktického hlediska můžeme poznamenat, že v případě chybějících údajů ve více atributech, může být nejvhodnější způsob nahrazení chybějící hodnoty pro každý atribut jiný.

Kromě chybějících údajů můžeme pomocí jednoduchého zobrazení vyskytujících se hodnot najít zjevně chybné údaje, jako např. hodnota „n“ ve sloupci označeném Pohlaví, obsahujícím jinak hodnoty „m“ a „z“. Záleží pak na znalosti prostředí, ve kterém data vznikala, příp. dalších okolnostech zda můžeme ono „n“ považovat za překlep a nahradit je hodnotou „m“ nebo za „nevím“ (nečitelná odpověď v dotazníku).

Rozsah dat

Rozsah dat je další oblastí, která může působit problém. Kolik věcí může zaznamenat o jednom člověku? Jméno, věk, adresa, telefon, výška, váha, tlak, před měsícem si koupil produkt XY, atd. Takových údajů mohou být tisíce. Některé údaje lze dokonce ještě rozdělit (adresa). Ne všechny musí být pro řešení problému důležité. Které důležité jsou, a které nejsou, může pomoci odhalit právě předzpracování. Sloupec tabulky, který ve vybraném vzorku dat obsahuje jedinou hodnotu, není třeba uvádět, stejně tak se pravděpodobně mnoho nedozvíme ze sloupce, kde je každá hodnota jiná a přitom se nejedná o tzv. primární klíč tabulky. Takovéto atributy je obvykle potřeba během fáze předzpracování diskretizovat či seskupit, tj. z hodnot vytvořit intervaly nebo kategorie hodnot. Tento proces může probíhat i ve více fázích, tj. např. pro příjem vytvoříme intervaly o rozsahu 1000 jednotek, po

spočítání frekvencí zjistíme, že několik prvních a několik posledních intervalů můžeme dále spojit. Počet vytvářených kategorií je individuální.

Sampling

V poslední době bývá typické, že dat, tj. jednotlivých případů je příliš mnoho, aby je systémy dokázaly v reálném čase zpracovávat. Kromě toho, že budeme hledat jen relevantní atributy, se můžeme dostat do situace, kdy prostě není technicky možné zpracovat všechny objekty (řádky tabulky). V tom případě můžeme zkusit Sampling, tj. výběr vzorku dat. Přitom můžeme prostě náhodně vybrat určitou část dat nebo se můžeme snažit o cílené vybírání. Při takovémto cíleném výběru se můžeme pokoušet o získání stejně početných skupin pro jejich porovnání. Např. z celkového počtu oslovených klientů 5% reagovalo, zbytek nikoliv. Vybereme všechny reagující, a k tomu stejný počet nereagujících. Přitom ještě můžeme sledovat, zda je ve výběru v obou kategoriích stejný poměr žen a mužů. Toto nazýváme stratifikovaný výběr. Zmenšit počet objektů v souboru můžeme i vhodným agregováním hodnot.

Některé data miningové metody potřebují dva vzorky dat, tzv. trénovací a testovací data. Příkladem může být neuronová síť. Součástí předzpracování tedy může být náhodné rozdělení datového souboru na dvě části.

Diskretizace

Diskretizace může probíhat několika způsoby. Kromě možnosti individuální volby kategorie či intervalů, můžeme využít také ekvidistantního nebo ekvifrekvenčního rozdělení hodnot. Ekvidistantní členění znamená, že vzdálenost mezi jednotlivými hodnotami je stejná, tj. např. při diskretizaci výšky osob zvolíme vzdálenost 5 a vytvoříme intervaly 140 – 145, 145 – 150, 150 – 155 atd. Ekvifrekvenční členění znamená, že vytvoříme určitý počet skupin, a tyto skupiny budou mít stejný počet hodnot. U příkladu s výškou osob bychom např. pro dvě stě osob mohli vytvořit 4 skupiny po padesáti osobách, přičemž jejich výška bude v rozsazích 143 – 152, 153 – 160, 160 – 170, 170 – 195 apod.

Spojování a dělení atributů

Tak, jak zkoumáním daného problému stále více rozumíme vztahům mezi daty, můžeme časem zjistit, že konkrétní atribut může ukázat další zajímavé vztahy, pokud jej dále rozdělíme či naopak spojíme. Můžeme se také dopracovat k poznání, že bychom potřebovali

další atributy, takové, které jsme zatím neměli k dispozici. Nejčastěji ale v této oblasti budeme řešit výběr nejlepší skupiny atributů. V této souvislosti ještě můžeme zmínit možnost spojení tzv. řídkých atributů do jednoho, čímž můžeme zmenšit jak samotný počet atributů, tak i místo na disku. Obojí má v konečné fázi určitý vliv na délku zpracování.

Algoritmy předzpracování dat

Jak přesně probíhá výše zmíněná diskretizace, či dělení na trénovací a testovací data? Jaké výpočty vlastně můžeme provést pro určení, které atributy skrývají zajímavé informace?

Kromě v předchozí kapitole zmíněné diskretizace, samplingu a ošetření chybějících hodnot nás může zajímat výpočet šumu v datech, který je podstatný, pokud je naším cílem klasifikace. Máme-li k dispozici mnoho atributů, může být jedním z kroků předzpracování výpočet Attribute Relevance a Attribute Dependence.

Seskupování

Funguje podobně, jako diskretizace, hovoříme o ní v případě, že máme k dispozici textový atribut. Textové atributy však postrádají některé vlastnosti atributů numerických. Ačkoliv textové hodnoty lze např. uspořádat abecedně, neexistuje u textových atributů uspořádání hodnot ve stejném smyslu, jako u čísel, tj. takové, ze kterého je zřejmé, že určitá hodnota je několikrát větší či menší, případně že je větší (menší) o nějaký počet jednotek. Nemá tedy smysl používat ekvidistantní a ekvifrekvenční seskupování.

Seskupení vzhledem ke třídě (class sensitive).

Následující algoritmus byl poprvé použit v systému KEX a je odvozen od podobného algoritmu pro diskretizaci.

Prvním krokem je vytvoření seznamu nových hodnot atributu, který budeme seskupovat a k tomuto seznamu je třeba ještě přidat jednu speciální hodnotu. Do té budou zařazeny hodnoty, u kterých nelze rozhodnout, kam by měly patřit (protože se např. všechny vyskytují v několika třídách stejně často). Po vytvoření seznamu budeme procházet seznam původních hodnot a počítat, kolikrát se tato hodnota bude vyskytovat v jednotlivých třídách (třídou rozumíme jednotlivé hodnoty cílového atributu). Zjistíme-li, že daná hodnota se vyskytuje pouze v jedné třídě, přiřadíme jí kód této třídy. Pokud se daná hodnota bude vyskytovat ve více třídách, avšak některá ze tříd bude převažovat (jako kritérium použijeme hodnotu chí kvadrát), přiřadíme jí kód nejčetnější třídy. Bude-li výskyt hodnoty ve třídách vyrovnaný, použije onu speciálně přidanou hodnotu. V dalším kroku potom přiřadíme zvolené nové

hodnoty atributu, a to podle přiřazeného kódu třídy – stejná nová hodnota tam, kde je stejný kód třídy.

Kromě toho můžeme seskupit hodnoty dle vlastního uvážení, bez ohledu na to, v jaké jsou třídě.

CHAID algoritmus seskupování hodnot atributu

CHAID znamená Chi-square Automatic Interaction Detection. Tento algoritmus seskupí hodnoty do pouhých dvou skupin. Využívá se, pokud je naším hlavním cílem provedení shlukové analýzy. Do doby, než dosáhneme vytvoření dvou skupin, provádíme následující kroky:

Zvolíme dvojici kategorií atributu, které jsou si nejpodobnější z hlediska $\chi 2$, a které mohou být spojeny. Novou kategorizaci atributu považujeme za možné shlukování v daném kroku. Pomocí $\chi 2$ testu spočítáme pravděpodobnost p pro každý z možných způsobů shlukování hodnot. Shlukování s nejnižší pravděpodobností p zvolíme za "nejlepší" shlukování hodnot atributu a zjistíme, jestli toto nejlepší shlukování statisticky významně přispěje k odlišení příkladů různých tříd.

Diskretizace

O diskretizaci hovoříme v případě, že máme numerický atribut s velkým počtem hodnot, ze kterých chceme vytvořit několik málo kategorií. Rozlišujeme tři hlavní metody diskretizace:

- zadané uživatelem
- slepé
- vzhledem ke třídě

Nejjednodušší je prvně jmenovaná. Vytvoříme si zkrátka intervaly podle potřeby. Třeba dva velkého rozsahu pro velmi nízké a velmi vysoké hodnoty a dále tři malé intervaly pro obvyklé hodnoty.

Metody slepé diskretizace jsou dvě. Ekvidistanční a ekvifrekvenční. Jejich princip již byl vysvětlen v předchozí kapitole. Metody diskretizace vzhledem ke třídě jsou také dvě. Rozlišujeme diskretizaci s ostrými intervaly a s fuzzy intervaly. Diskretizace s ostrými intervaly se v podstatě neliší od seskupování. Podívejme se tedy na algoritmus metody s fuzzy intervaly.

Význam slova Fuzzy je neostrý, rozostřený. Fuzzy interval tedy bude interval, který nemá úplně pevné hranice. Mírnou analogii bychom mohli hledat u zaokrouhlování. Budou-li se např. naměřené hodnoty u pacienta velmi blížit nějaké stanovené nebezpečné hranici, lékař zpozorní a provede příslušné opatření – např. jej pozve na preventivní prohlídku ke specialistovi.

Fuzzy diskretizace

Při fuzzy diskretizaci začneme podobně jako u seskupování vytvořením seznamu nových hodnot a vytvořením speciální kategorie pro hodnoty, u kterých nelze říci, kam by měly patřit. I další kroky budou stejné, jako v případě seskupování. Staré hodnoty vzestupně uspořádáme a spočítáme, kolikrát se staré hodnoty vyskytují v jednotlivých třídách (tj. kolikrát se současně stará hodnota vyskytuje s konkrétní hodnotou zvoleného cílového atributu). Pokud se stará hodnota bude vyskytovat jen s jedinou hodnotou cílového atributu, přiřadíme jí jako kód tuto hodnotu cílového atributu. Bude-li se stará hodnota vyskytovat současně s více hodnotami cílového atributu, přičemž některá hodnota cílového atributu bude převažovat (určíme dle chí kvadrát testu), přiřadíme staré hodnotě kód nejčetnější třídy. V případě, že k převaze některé hodnoty nedojde, přiřadíme kód speciální kategorie. Dosud byl tedy postup stejný, jako u seskupování.

Spojování intervalů se již bude lišit. Budeme procházet hodnoty shora dolů, a sledovat, jaké kódy jsme jim přiřadili. Jestliže bude mít sekvence hodnot stejný kód, hodnoty spojíme do intervalu. Tím získáme určité množství intervalů. Dále budeme procházet tyto intervaly. Najdeme-li interval, s kódem speciální kategorie, který je obalen z obou stran dvěma intervaly se stejným kódem, tj. např. A-X-A, spojíme jednoduše všechny tři intervaly. Pokud nejsou oba obalové intervaly shodné, tj. např. A-X-B, potom vytvoříme dva intervaly. Budeme si přitom evidovat horní i dolní mez původních intervalů, a to následujícím

způsobem. Vytvoříme čtveřici hodnot [K,L,M,N]. Při spojení intervalů A a X budou hodnoty K a L rovny dolní mezi intervalu A, hodnota M horní mezi intervalu A a hodnota N horní mezi intervalu X. Při spojení intervalů X a B bude hodnota K rovna dolní mezi intervalu X, hodnota L dolní mezi intervalu B a hodnoty M a N budou rovny horní mezi intervalu B. Dále budeme tímto procesem vytvářet spojité pokrytí definičního intervalu.

Binarizace

Je-li naším cílem vytvoření rozhodovacího stromu, můžeme použít algoritmus, který provede rozdělení na dva intervaly, tzv. binarizaci. Využívá se při něm informace o tom, do které třídy patří příklad s konkrétní hodnotou diskretizovaného atributu. Jako kritérium se využívá entropie. Při hledání dělícího bodu (cut-point), který rozdělí hodnoty do dvou intervalů, se postupuje následujícím způsobem:

Seřadíme vzestupně hodnoty diskretizovaného atributu A. Pro každou možnou hodnotu dělícího bodu θ spočítáme střední entropii atributu s využitím vzorce

$$H(A_\theta) = \frac{n(A(<\theta))}{n} H(A(<\theta)) + \frac{n(A(>\theta))}{n} H(A(>\theta))$$

První člen součtu se týká příkladů, které mají hodnotu atributu menší než θ (H(A(<θ)) je entropie na těchto příkladech, n(A(<θ))/n je relativní četnost těchto příkladů), druhý člen součtu se analogicky týká příkladů, které mají hodnotu atributu větší než θ. Vybereme dělící bod, který dá nejmenší entropii.

Fayyadův a Iraniho algoritmus

Prvním krokem tohoto algoritmu je uspořádání dat vzestupně podle hodnoty diskretizovaného atributu. Dále provádíme rekurzivní binarizaci, a to tak, že hledáme nejvhodnější dělící bod θ. Je-li Zisk$(A\mathrm{Int},\theta)$ pro tento bod větší, než hodnota $\mathrm{Log}_2 (n-1) / n +$ (Δ Zisk$(A\mathrm{Int},\theta)$ / n)

Rozdělíme interval podle dělícího bodu θ a pokračujeme v rekurzi.

Leeho a Shinův algoritmus

Ve fázi inicializace tohoto algoritmu je opět potřeba uspořádat data vzestupně podle hodnoty diskretizovaného atributu. Pro každý dělící bod $\theta i = (ai + ai+1)/2$ je vytvořen interval $Int_i = [\theta i, \theta i+1]$ a spočítáno $E(Int_i)$ a $E(\theta i)$, kde

$$E(Int) = \left[\sum_t \left(p(Class_t) - p(Class_t | Int) \right)^2\right]^{1/2}$$

měří pomocí Hellingerovy divergence rozdíl mezi množstvím informace v celých datech a množstvím informace v intervalu a

$$E(\theta) = \left[\sum_t \left(p(Class_t | A(<\theta)) - p(Class_t | A(>\theta)) \right)^2\right]^{1/2}$$

Podobným způsobem vyjadřuje informaci spojenou s dělícím bodem θ oddělujícím dva intervaly.

Hlavní cyklus algoritmu probíhá do doby, než je dosaženo požadovaného počtu intervalů.

Nejprve hledáme θmin takové, že $E(\theta min) = \min_i E(\theta i)$. Potom vytvoříme interval Int_{min} jako sjednocení $[\theta min-1, \theta min]$ a $[\theta min, \theta min+1]$. Nakonec spočítáme $E(Int_{min})$, $E(\theta_{min-1})$ a $E(\theta_{min+1})$.

Sampling

Data sampling, nebo-li výběr vzorku dat je procedura, jejímž cílem je vybrat jen určitou část dat. Tento výběr je náhodný, s navracením. Současně se můžeme snažit o změnu podílu jednotlivých tříd proti podílu tříd ve vstupních datech. Tento náhodný a přitom záměrný výběr se nazývá vyvažování (balancing). Jeho algoritmus je následující:

Nejprve spočítáme počet objektů ve vstupních datech, přičemž potřebujeme znát jak celkový počet objektů, tak počet objektů v jednotlivých třídách. Zaznamenáme si podíly jednotlivých tříd na celkovém počtu dat. Dělení na třídy odpovídá dělení cílového atributu na jeho jednotlivé hodnoty. Potom procházíme všechny objekty. V případě, že provádíme vyvažování, a objekt patří do třídy, ze které chceme všechny objekty, tento objekt zařadíme do vzorku. Jinak vygenerujeme náhodné číslo. Je-li náhodné číslo menší, než požadovaný podíl třídy tohoto objektu a tato třída ještě není zaplněna, přiřadíme tento objekt do vzorku.

Pokud jsme prošli všechny objekty, a některá třída ještě není zaplněna, provedeme náhodný výběr objektu této třídy ze vstupních dat. Toto opakujeme až do doby, než vybereme potřebný počet objektů.

Splitting

Splitting znamená rozdělení souboru na dvě části – trénovací a testovací. Algoritmus je následující. Spočítáme počet objektů ve vstupních datech. Určíme procentní podíl pro testovací i trénovací část a poté postupně procházíme všechny objekty. Vždy vygenerujeme náhodné číslo, a pokud je menší, než potřebný podíl trénovacích dat a ještě nemáme tuto část zcela zaplněnou, zařadíme objekt mezi trénovací data, v opačném případě mezi testovací data. Současně se můžeme i pokusit zachovat poměr tříd, nebo-li procentuální podíl hodnot atributu, který jsme označili jako cílový, což nazýváme prováděním stratifikovaného výběru. V tom případě sledujeme i obsazenost jednotlivých tříd, a jakmile je třída zaplněna, další objekty do ní neukládáme.

Určení nejkvalitnějších atributů

Máme-li k dispozici velké množství atributů, obvykle potřebujeme vybrat jen jich podmnožinu. Zorientovat se ve stovkách atributů, jejich hodnotách a souvislostech mezi jednotlivými hodnotami je náročný úkol, nicméně můžeme si tento úkol usnadnit několika procedurami. Je-li naším hlavním úkolem klasifikace, bude pro nás zajímavým ukazatelem podíl šumu a relevance atributu pro klasifikaci. Pro určení, které atributy jsou vzájemně závislé a tudíž, které jsou kromě jednoho libovolného z nich nepotřebné využijeme proceduru počítající závislost (dependence).

Podíl šumu (noise evaluation)

Při výpočtu zjišťujeme, zda existují objekty, které se vyskytují vícekrát a přitom patří do různých tříd. Podíl počtu takovýchto objektů ke všem objektům ve stupních datech nazýváme maximální možnou správností, odečtením této hodnoty od jedničky získáme podíl šumu. Ideální situací je tedy podíl šumu rovný nule a maximální možná správnost rovná jedné.

Relevance atributu

K určení, jak moc je atribut vhodný ke klasifikaci můžeme použít výpočet, jehož základem je vytvoření kontingenční tabulky z hodnot zvoleného atributu a cílového atributu. Kritériem, podle kterého budeme vhodnost hodnotit může být statistika chí kvadrát, dále entropie a mutual information.

1. pro kritérium chí kvadrát spočítáme statistiku

$$\chi^2 = \sum_{i=1}^{R}\sum_{j=1}^{S}\frac{(a_{ij} - o_{ij})^2}{o_{ij}} = n \times \sum_{i=1}^{R}\sum_{j=1}^{S}\frac{\left(a_{ij} - \frac{r_i \cdot s_j}{n}\right)^2}{r_i \cdot s_j} .$$

Kde R je počet řádků, S je počet sloupců, n je počet hodnot v tabulce, r_i je řádkový součet, s_j sloupcový součet, a_{ij} hodnota v tabulce na řádku i a ve sloupci j a o_{ij} je hodnota, kterou na stejném místě očekáváme.

Platí čím vyšší hodnota tím lepší. Je-li $\chi^2 \geq \chi^2_{(R-1)(S-1)}(\alpha)$, kde $\chi^2_{(R-1)(S-1)}(\alpha)$ je tabulková hodnota, zamítneme na zvolené hladině významnosti α hypotézu nezávislosti X a Y.

2. Použijeme-li jako kritérium entropii spočítáme

$$H = \sum_{i=1}^{R}\frac{r_i}{n} H_i ,$$

kde

$$H_i = -\sum_{j=1}^{S}\frac{a_{ij}}{r_i} \log\frac{a_{ij}}{r_i}$$

Význam proměnných je stejný, jako u statistiky chí kvadrát.

H je z intervalu [0,1] a platí, že čím nižší je jeho hodnota, tím lepší. Je-li tedy H=0, lze atribut X použít pro bezchybnou klasifikaci objektů do tříd Y.

3. pro kritérium mutual information spočítáme

$$IMD = \frac{MI}{-\sum_{j=1}^{S} \frac{s_j}{n} \log \frac{s_j}{n}},$$

kde

$$MI = -\sum_{i=1}^{R} \sum_{j=1}^{S} \frac{a_{ij}}{n} \log \frac{\frac{a_{ij}}{n}}{\frac{r_i}{n} \frac{s_j}{n}} = -\frac{1}{n} \sum_{i=1}^{R} \sum_{j=1}^{S} a_{ij} \log \frac{n\, a_{ij}}{r_i\, s_j}$$

Význam proměnných je opět stejný. Zde naopak platí, že čím vyšší hodnota kritéria, tím lepší. Je-li IMD=1, je cíl Y funkčně závislý na X. Také hodnota IMD je z intervalu [0, 1].

Závislost atributu (attribute dependence)

Při výpočtu závislosti atributu hledáme takové atributy, které maximalizují informační obsah podle následujícího vzorce.

$$IC(X_{i1},\ldots,X_{ik}) = \sum_{(x_{i1},\ldots,x_{ik})} P(X_{i1} = x_{i1},\ldots,\ X_{ik} = x_{ik}) \log \frac{P(X_{i1} = x_{i1},\ldots,\ X_{ik} = x_{ik})}{P(X_{i1} = x_{i1}) \ldots\ P(X_{ik} = x_{ik})}$$

$X_{i1} = x_{i1}$ znamená, že atribut X nabývá v daném řádku hodnoty x. Atribut je v dané chvíli prvním zleva (index 1). P značí pravděpodobnost. Počítáme tedy pravděpodobnosti výskytu určité kombinace hodnot vybraných atributů.

Vybrané systémy pro předzpracování dat

SumatraTT

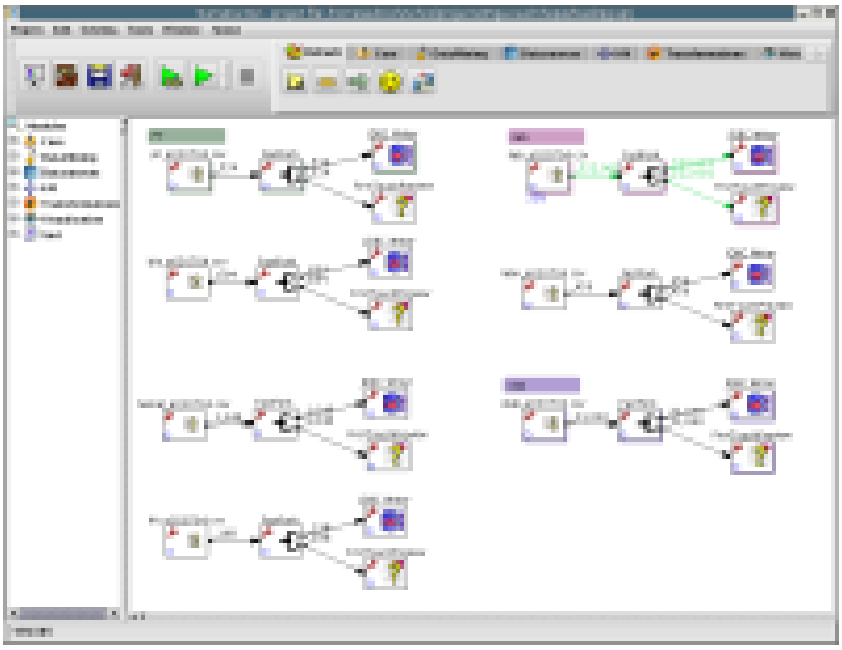
Systém je vyvíjen na ČVUT. Poslední verze, SumatraTT2.1.1 pochází z ledna 2008. Nevyžaduje instalaci. Protože je psaný v Javě, vyžaduje její běhové prostředí (JRE), to je však zdarma ke stažení na stránkách firmy Oracle.

SumatraTT je univerzální, metadaty řízený transformační nástroj. Je postaven nad skriptovacím jazykem orientovaným na datové transformace, syntaxí podobným jazyce Java, nazývaným SumatraCsript. K vytváření datové transformace využívá knihovnu šablon (templates). Podporuje automatickou dokumentaci datové transformace. Systém SumatraTT dokáže zpracovat jak textové soubory, tak i různé databázové soubory a XML. Jejím výstupem může být i soubor ve speciálním formátu, který používá systém Weka.

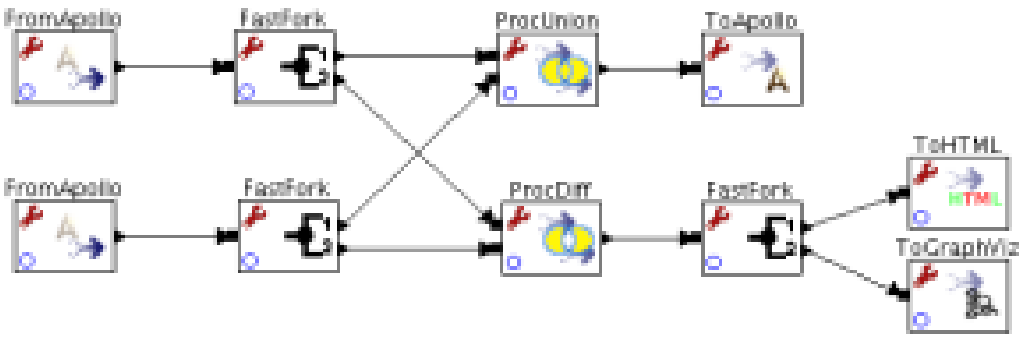
Dostupné šablony

- **TableCopy** – prosté kopírování, převod mezi formáty, filtrování, výpočet nových atributů
- **Table1toN** – rozdělení záznamu na několik, spojení
- **TableSample** – náhodné rozdělení DS na dva (trénovací, testovací)
- **FairSubset** – podmnožina o přesně daném počtu prvků (nebo v procentech)
- **TableReport** – textová zpráva (hledání chyb)
- **CheckDS** – kontrola čitelnosti datového zdroje
- **gnuplot** – vykreslení dat pomocí gnuplot, 2D i 3D
- **RemoveDuplicities** – zpracování duplicit v SQL databázích
- **CrossTable** – podpora (poloautomatická) pro křížové tabulky
- **ExecSQL** – spustí SQL příkaz

SumatraTT je vizuální nástroj, podobně jako např. systémy Clementine od IBM SPSS, nebo Enterprise Miner od SAS.



Obr. 2 – Ukázka prostředí systému SumatraTT



Obr. 3 – Detail úlohy řešené systémem SumatraTT

Každá ikona představuje jeden krok procesu a v systému je nazývána modulem. Může jít o otevření souboru, diskreditaci hodnot, zobrazení grafu nebo třeba vytvoření náhodných dat. Každý modul může mít několik vstupů a výstupů. Jejich počet, stejně jako některé další vlastnosti lze navolit stisknutím pravého tlačítka myši na ikoně modulu. Nastavení parametrů, jako např. specifikaci souboru, se kterým chceme pracovat, naopak provádíme klasickým dvojklikem levého tlačítka myši, a to např. na ikoně představujíc modul s názvem

FromFile (ze souboru). Posloupnost kroků se vyjadřuje jejich propojením. Kliknutím nejprve na výstup jednoho modulu a poté na vstup druhého se vytvoří spojovací linie. Pokud jsme nadefinovali celý proces, můžeme ho spustit pomocí zelené šipky v levé horní části. Systém začne provádět jednotlivé kroky a zobrazí výstup, případně okno pro zadání vstupu od uživatele.

Mezi procedurami, které projekt Sumatra TT nabízí, najdeme:

- Výběr atributů z jedné či více tabulek
- Barevné označení vybraného atributu (stejné hodnoty jednou barvou)
- Dělení textových řetězců na jednotlivé znaky nebo slova
- Výpočet absolutní hodnoty, druhé mocniny nebo goniometrických funkcí
- Náhodný výběr vzorku dat (Sampling)
- Zobrazení grafů

Jednotlivé procedury (resp. moduly) jsou uspořádány do několika oblastí. Ty se nazývají

- Core
- Datasources
- KM
- Transformations
- XML

Oblast Core zahrnuje tři moduly – Fork, Join a FastFork, které slouží ke spojování a rozdělování činností. Tj. např. pokud chceme výsledný výstup uložit do souboru XML i do textového souboru, použijeme Fork.

Oblast Datasources zahrnuje moduly schopné přečíst a uložit data do souboru typu dbf, MySQL databáze, Oracle databáze, nebo z a do obyčejného textového souboru. Najdeme zde i generátor goniometrických funkcí sinus a kosinus a generátor náhodných čísel.

Oblast KM zahrnuje moduly pro práci s ontologiemi definovanými v systému Apollo. Sumatra je dokáže transformovat do webové stránky, nebo naopak XML dokument transformovat na objekt Apollo.

Oblast Transformations má největší počet modulů. Moduly jsou rozděleny do skupin. Skupina Fields se skládá z modulů, pomocí kterých můžeme vybírat, obarvovat a spojovat sloupce hodnot a modul, který rozloží textový řetězec na jednotlivé znaky nebo slova. Skupiny Math, Presentation a Scrips obsahují po jednom modulu. Matematické transformace ze skupiny Math spočítají druhou mocninu, absolutní hodnotu a hodnotu goniometrických funkcí sinus, kosinus a tangens. Table viewer ze skupiny Presentation zobrazí data v tabulce, což je užitečné např. pro data z textových souborů. Scripting umožňuje vytvoření vlastního kódu pro zpracování dat. Skupina Subset obsahuje dva moduly, FairSubset a VarioSubset. Ty slouží pro rozdělení dat. Skupina Util obsahuje moduly Benchmark a Meta2Data. Poslední skupina, Visualisation obsahuje moduly Histogram, Matrix, Multiline, ParalelView, RadViz a SimpleChart.

Poslední oblast, XML obsahuje moduly pro zpracování XML souborů, tj. jejich načtení, uložení jako XML a zobrazení.

Tyto oblasti najdeme v systému jednak v panelu v levé části, kde jsou uspořádané do stromové struktury, a jednak v horní části, kde má každá oblast svoji kartu. Na jednotlivých kartách jsou umístěny ikony. Jednotlivé moduly lze umísťovat na pracovní plochu i přes klasické menu v levé horní části okna aplikace. Stromová struktura, díky tomu, že obsahuje názvy, a další uspořádání modulů do skupin, může být, zvláště pro uživatele začátečníky o něco přehlednější, než karty s ikonami. Ne všechny moduly mají totiž unikátní ikonu. Zvláště oblast KM se hemží ikonami s otazníkem. Panel se stromovou strukturou je možné zavřít a nepoužívat, čímž získáme více místa na pracovní ploše aplikace. Další volitelně zobrazovanou částí je konzole, kterou si můžeme zobrazit v dolní části okna aplikace. Konzole obsahuje systémové zprávy.

Mining Mart

Mining Mart je grantový projekt Univerzity Dortmund. Jak už je psáno výše, předzpracování dat pro různé analýzy zabere více než polovinu času a cílem projektu bylo vytvořit aplikaci, která proces předzpracování dat uživateli zjednoduší. Základní myšlenkou bylo vhodně uložit informace o nejlepších postupech při dobývání znalostí z databází, které používají experti v oboru. Uživatel systému poté může zvolit typ úlohy, kterou potřebuje řešit a dále aplikovat odpovídající transformace. Jedním z cílů projektu je tedy vytvořit a publikovat případy úspěšného využití na internetu. Sdílení takovýchto znalostí má samozřejmě užitek pro další uživatele. Publikuje se konceptuální model a typ případu, relační datový model zůstává skrytý. Jsou vyřešeny tři případy - analýza direct mailingu, detekce podvodů v telekomunikacích a předpověď prodeje. Mining Mart byl např. úspěšně aplikován ve dvou velkých telekomunikačních společnostech.

OLAP - analytické zpracování on-line nabízí analýzu dat využívající agregaci dat a výpočet frekvencí. Může tedy pomoci odpovědět na otázky typu: „Jaké atributy mají moji nejčastější zákazníci? Které produkty si nejčastěji kupují? Jaké množství nezaplacených účtů mohu očekávat za rok? Kolik reakcí jsem obdržel na moji poslední direct mail nabídku?“ Reporty, které podporují rozhodování, potřebují ale detailnější informace. Otázky jsou specifičtější, např.: „Jak velký prodej určité položky mohu očekávat a jak velké zásoby mám tedy vytvořit, aby byla vždy k dispozici a přitom bylo minimální množství na skladě?“ Abychom na takovéto otázky dokázali odpovědět, musíme dostupná data pro hledání těchto odpovědí připravit. Předzpracování dat je v projektu Mining Mart klíčová úloha, která se skládá z několika kroků:

- určení typu řešeného problému
- výběru dat
- generování, extrakce a selekce potřebných atributů
- čištění dat
- výběr modelu a ladění velikosti prostoru hypotéz
- definování odpovídajících ověřovacích kritérií

Mining Mart proto nabízí následující:

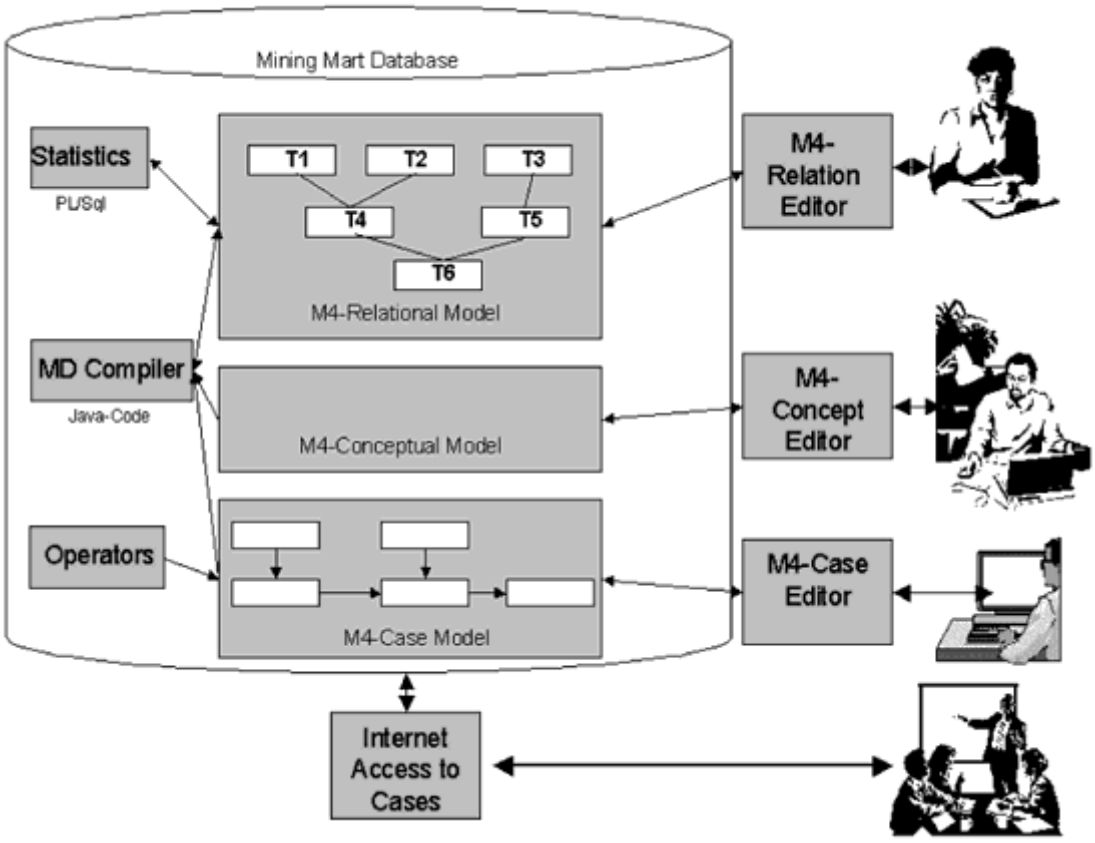
- Operátory pro předzpracování s přímým přístupem do databáze
- Použití strojového učení pro předzpracování
- Detailní dokumentace úspěšných případů
- Vysoká kvalita získaných výsledků
- Možnost využití pro velké databáze
- Techniky, které automaticky vybírají nebo mění reprezentaci

Jak se postupuje v projektu Mining Mart

Mining Mart pracuje s tzv. metamodelem. Ten je vytvořen z konceptuálního modelu, relačního modelu a případového modelu. Jeho vytvoření vyžaduje spolupráci odborníků z různých oblastí. Postupuje se v následujících krocích:

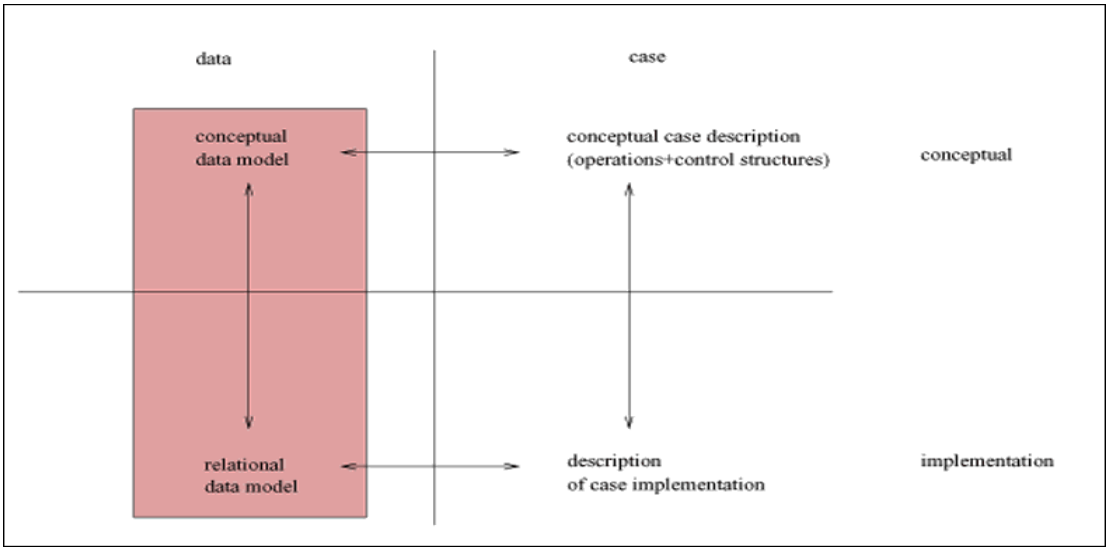
- Metamodel je uložen do databáze.
- Manažer databáze vytvoří relační model.
- Datový analytik vytvoří konceptuální model.
- Znalostní expert vytvoří nebo využije připravený případový model.
- Systém zkompiluje meta data do SQL dotazů a vyvolá externí procedury pro vykonání případového modelu na datech.

Schematicky to znázorňuje i následující obrázek:



Obr. 4 – Mining Mart – relační, konceptuální a případový model

Forma zápisu metadat je specifikována MiningMart metamodelem M4. Je strukturovaná do dvou dimenzí, dle tématu a abstrakce – viz následující obrázek:

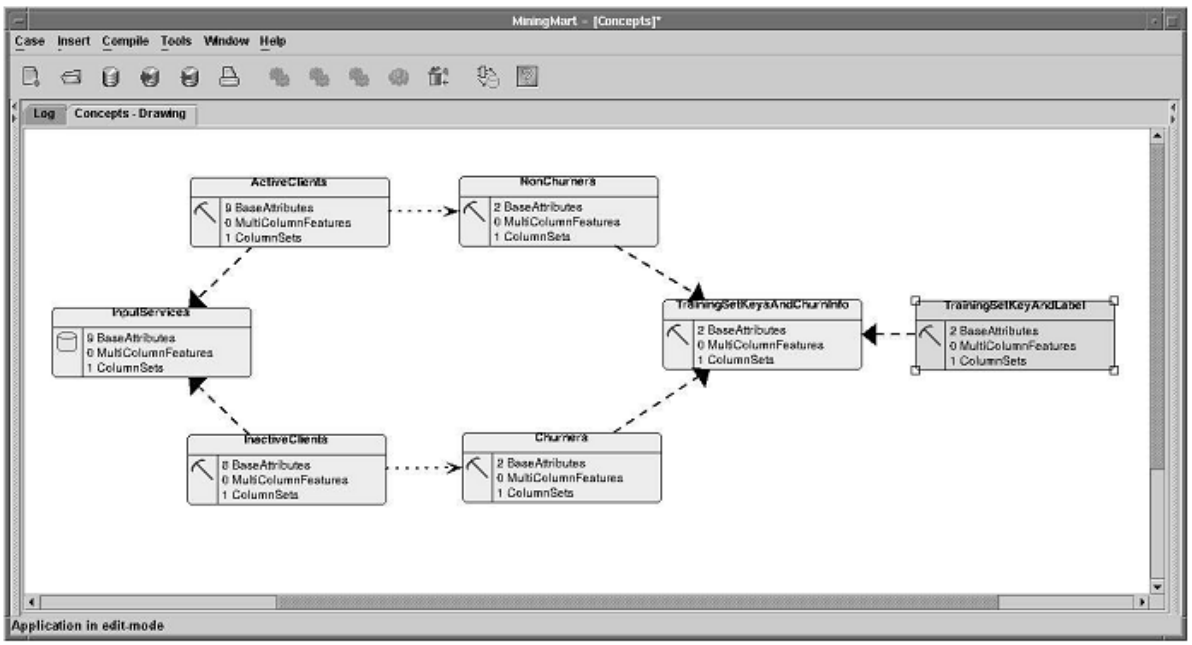


Obr. 5 – Mining Mart metamodel

Tématem mohou být obchodní data nebo případ. Obchodní data jsou tím, co je potřeba analyzovat. Případ je sekvence kroků použitých pro předzpracování. Abstrakce je konceptuální nebo relační. Tam, kde konceptuální úroveň bude stejná pro více aplikací, je relační model odkazem na konkrétní databázi. Meta data zapsaná ve formě specifikované pomocí M4 jsou taktéž uloženy v relační databázi. Relační model popisuje databázi. Výkonný model generuje SQL dotazy a volá externí procedury. Konceptuální model popisuje individuální případy i třídy domény s jejich vztahy. Případový model popisuje řetěz operátorů pro předzpracování.

Editace konceptuálního datového modelu

Jak již bylo popsáno, různí experti pracují na různých částech procesu dobývání. Nejprve doménový expert definuje konceptuální model pomocí koncept editoru. Entity zapojené do data miningu jsou jím explicitně vytvořeny. Konceptuální model M4 je o konceptech majících určité rysy a vztazích mezi těmito koncepty. Koncepty a vztahy mohou být hierarchicky organizovány s využitím dědění. Příkladem konceptů jsou zákazník a produkt a vztahem mezi nimi je "kupuje".

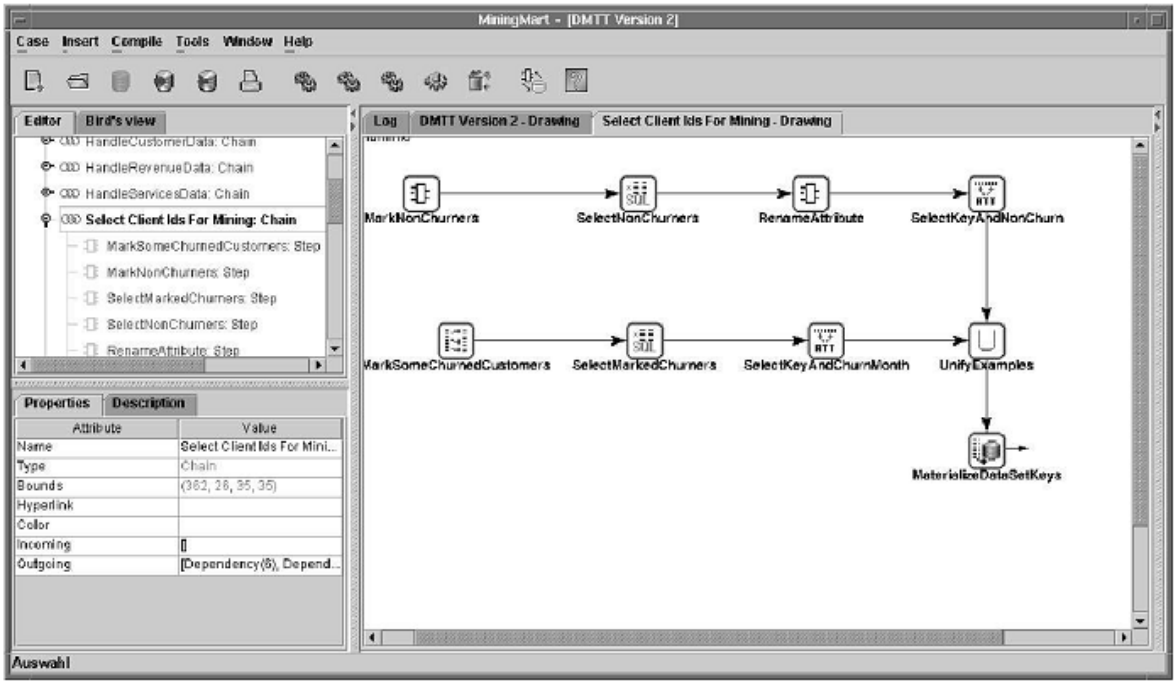


Obr. 6 – Koncept editor

Editace relačního modelu

Máme-li daný konceptuální model, databázový administrátor mapuje zahrnuté entity na odpovídající databázové objekty. Relační datový model M4 je schopný reprezentovat všechny důležité vlastnosti relační databáze. Nejjednodušší mapování z konceptuální na relační úroveň je dáno, pokud koncepty přímo odpovídají databázovým tabulkám nebo pohledům. Toho můžeme vždy dosáhnout prozkoumáním databáze a vytvořením pohledu pro každý koncept. Sofistikovanější cesty grafického výběru atributů a jejich spojení do konceptů však budou zvyšovat akceptování koncovými uživateli. Relační editor podporuje právě tento druh aktivit. Obecně by mělo být možné mapovat všechny rozumné reprezentace entit na rozumné konceptuální definice. Jednoduché mapování konceptu Zákazník obsahujícího znaky ID zákazníka, jméno a adresa na databázi, by mělo vyústit v konstatování, že databázová tabulka obsahuje všechny potřebné atributy (např. sloupce ZakID, ZakJmeno, ZakAdr). Složitější případ mapování nastane v okamžiku, kdy informaci o adrese či jméně získáme až spojením některých údajů s využitím primárního klíče ZakID.

Tvorba case modelu



Obr. 7 – Case editor

Při tvorbě case modelu řetězíme jednotlivé operátory, Výsledek může vypadat jako na obrázku 7.

V projektu Mining Mart jsou k dispozici následující skupiny operátorů:

- Operátory pro výběr dat
- Operátory pro ošetření chybějících hodnot
- Operátory pro diskretizaci

Data preprocessing tool

O aplikaci

Aplikace Data preprocessing tool, (dále DPT) bude sloužit pro předzpracování dat ve formátu textového souboru csv. Aplikace je naprogramována pomocí programovacího jazyka Java, vyžaduje tedy běhové prostředí Javy (JRE), které je volně dostupné z webových stránek společnosti Oracle. Je tvořena jediným souborem, jar archivem.

Aplikace nabízí několik funkcí, které umožňují zjistit základní statistické informace o zpracovávaném souboru, provést výpočty potřebné pro rozhodnutí, jaké úpravy je potřeba s daty provést a následně změny realizovat. Funguje přes grafické uživatelské rozhraní, nicméně není nástrojem vizuálním, jako popisované systémy Mining Mart a SumatraTT. Okno aplikace obsahuje řadu tlačítek, několik vstupních polí a přepínačů pro uživatelské volby a okno pro zobrazování výsledků. Poskytovaných funkcí není mnoho, a proto je vše zobrazeno a uspořádáno v jednom okně, což je možná poněkud netradiční řešení, nicméně uživatel neztrácí čas hledáním, pod kterou položkou menu se nachází funkce, kterou chce zrovna použít. Některé funkce vyžadují dodatečnou informaci od uživatele, některé ji nepotřebují. Funkce nevyžadující doplnění informace po stisku tlačítka provedou svoji činnost a v okně pro zobrazení výsledků zobrazí informaci o provedení akce a jejím výsledku. U funkcí vyžadujících dodatečnou informaci se stiskem aktivačního tlačítka s jejich názvem aktivují příslušná vstupní pole a přepínače a tlačítko provádějící příslušnou akci. Funkce vyžadující vstup od uživatele, je potřeba po skončení jejich práce deaktivovat příslušným tlačítkem. Aktivační tlačítko těchto funkcí zobrazí nápovědu, jak postupovat pro bezproblémové dokončení a získání výsledků.

Aplikace tedy používá několik druhů tlačítek. Aktivační tlačítko funkce, kterým se aktivují příslušná pole a přepínače pro uživatelský vstup, tlačítko funkce, které vykoná danou funkci a deaktivační tlačítko, které deaktivuje pole uživatelského vstupu a aktivuje všechny funkce aplikace.

Funkce aplikace:

Aplikace obsahuje následující funkce:

- Základní statistická deskripce (výpočet frekvence hodnot, minima, maxima, průměru a modu)
- Podíl šumu
- Relevance atributu
- Vzájemná závislost atributů
- Ošetření chybějících hodnot
- Výběr objektů a atributů
- Rozdělení na trénovací a testovací data
- Sampling
- Seskupování hodnot
- Diskretizace hodnot

Základní statistická deskripce

Tato funkce zobrazí nejprve jméno atributu, počet jeho různých hodnot, u numerických hodnot minimum, maximum, směrodatnou odchylku a u všech hodnot modus. Následují vzestupně řazené hodnoty atributu, spolu s jejich absolutní i relativní frekvencí.

Tato funkce vyžaduje zadání atributů, o nichž chceme zobrazit výše uvedené informace.

Podíl šumu

Jak je již zmíněno v kapitole o algoritmech, při výpočtu podílu šumu zjišťujeme, zda existují objekty, které se vyskytují vícekrát a přitom patří do různých tříd. Podíl počtu takovýchto objektů ke všem objektům ve stupních datech nazýváme maximální možnou správností, odečtením této hodnoty od jedničky získáme podíl šumu. Tato funkce tedy provede tento výpočet. Pracuje s celým souborem a nevyžaduje žádný vstup uživatele. Jejím výstupem je buď hlášení o tom, že v souboru neexistují objekty patřící do více tříd, nebo seznam objektů s informací o počtu těchto objektů v souboru. Na závěr funkce zobrazí informace o hodnotě maximální možné správnosti a minimální možnou chybu.

Relevance atributu

Relevance atributu je číslo, vyjadřující vhodnost použití atributu. Tato vhodnost se měří pomocí kritérií Chí, Entropie a Mutual Information. Algoritmy pro výpočet těchto kritérií jsou popsány v kapitole o algoritmech. V DPT jsou spočítány všechna tato kritéria a společně s kontingenční tabulkou pro příslušný atribut zobrazeny v okně výsledků.

Tato funkce vyžaduje zadání atributů, pro které chceme hodnoty těchto tří kritérií vypočítat. Jejím výstupem je již zmíněná kontingenční tabulka zobrazující počty hodnot atributy zařazené do jednotlivých tříd daných vybraným cílovým atributem (zadávaným po spuštění aplikace), a řádkové a sloupcové součty. Pod tabulkou jsou poté zobrazeny hodnoty jednotlivých kritérií.

Vzájemná závislost atributů

Tato funkce vyžaduje zadání atributů. Pro všechny možné dvojice z vybraných atributů je spočítán jejich informační obsah dle vzorce uvedeného v kapitole o algoritmech. Na závěr je ještě jednou vypsána kombinace, kde je informační obsah nejvyšší.

Ošetření chybějících hodnot

Při načtení souboru jsou chybějící hodnoty nahrazeny řetězcem „DPT – missing value“. Uživatel však může řádky s chybějícími hodnotami úplně odstranit, nahradit je nejčetnější hodnotou příslušného atributu, nahradit je proporcionálně hodnotami příslušného atributu, nebo tento řetězec nahradit náhodně hodnotami, které se vyskytují v atributu.

Tato funkce tedy vyžaduje vstup od uživatele. Zatím zpracovává všechny chybějící hodnoty atributů jednotně. Uživatel tedy volí pouze způsob, jak chybějící hodnoty ošetřit a tento je proveden pro všechny chybějící hodnoty okamžitě po výběru způsobu. Použije-li uživatel náhradu nejčetnější hodnotou a tuto není možné určit (např. atribut s pohlavím, obsahuje stejný počet žen a mužů a sudý počet hodnot chybí), zůstane v datech řetězec „DPT – missing value“. V tomto případě zůstává tlačítko této funkce aktivní a uživatel může pro tyto zbylé výskyty použít jinou možnost ošetření. Jinak se tlačítko této funkce stává neaktivním.

Výběr objektů a atributů

DPT umožňuje zpracovávat jen některé atributy a některé řádky původního souboru. Tato funkce samozřejmě vyžaduje vstup uživatele. Chceme-li pracovat jen s některými atributy,

tyto vybereme a uložíme do nového souboru příslušným tlačítkem. Můžeme vybrat libovolnou kombinaci atributů. DPT zobrazí standardní dialog pro ukládání souborů, kde si vybereme, jak se má nový soubor jmenovat a soubor uložíme. Nyní, chceme-li pracovat s tímto novým souborem, je potřeba aplikaci ukončit, spustit znovu a načíst tento nově vytvořený soubor.

Výběr objektů, čili řádků souboru je poněkud náročnější. Nejprve je potřeba stisknout aktivační tlačítko funkce. Poté definujeme jeden úsek dat, který chceme zařadit, případně vyřadit ze souboru. Tato definice spočívá v zadání prvního a posledního řádku úseku. V případě, ze definovaný úsek chceme vyřadit a ostatní řádky ponechat, zaškrtneme volbu Exclude rows. Pokud naopak chceme ponechat definovaný úsek, tuto volbu necháme nezaškrtnutou. DPT provede příslušný výběr a nabídne standardní dialog pro uložení souboru. Opět platí, že chceme-li pracovat s tímto novým souborem, je potřeba aplikaci ukončit, spustit znovu a načíst tento nově vytvořený soubor. Jinak je možné pracovat s původním souborem. Všechny funkce aplikace zpřístupníme stiskem tlačítka pro deaktivaci funkce pro výběr řádků.

Dělení na trénovací a testovací data

Funkce pro dělení souboru na dvě části pracuje s několika parametry – počtem opakování (jako default je nastaven jeden běh), procentním nebo absolutním vyjádřením počtu dat trénovací množiny a požadavkem na stratifikovaný výběr. Algoritmus bez i včetně stratifikovaného výběru je opět popsán v kapitole o algoritmech.

Funkce tedy vyžaduje vstup uživatele. Po její aktivaci uživatel zadá do textového pole označeného Relative hodnotu v rozsahu 1 až 99, vyjadřující, kolik procent řádků má být v trénovací množině nebo do pole Absolute absolutní maximální počet řádků. Protože algoritmus prochází data jen jednou, není vyloučeno, že absolutní počet řádků bude ve skutečnosti o něco málo menší, než zadaný počet. Chyba však bude v řádu jednotek.

Rozdělení dat lze libovolněkrát opakovat, a proto je součástí zadání počet opakování, který je přednastaven na hodnotu 1. Chceme-li provést stratifikovaný výběr, zaškrtneme tuto volbu, v opačném případě ji ponecháme nezaškrtnutou.

Poté stiskneme tlačítko, kterým potvrdíme svoji volbu. DPT zobrazí standardní dialog ukládání souborů, nejprve pro trénovací množinu, po uložení trénovací množiny i pro testovací množinu.

Pokračování práce je možné po stisknutí deaktivačního tlačítka této funkce.

Sampling

Tato funkce pracuje podobně jako procedura předchozí. Vytváří se však jen jeden soubor. Je zde také zahrnuta možnost stratifikovaného výběru a dále možnost zvolit výběr dat tak, aby byly zahrnuty všechny objekty vybrané třídy. Tyto dvě možnosti však v DPT nelze použít najednou.

Funkce tedy opět vyžaduje vstup uživatele, příslušná pole se aktivují stiskem aktivačního tlačítka funkce. Stejně jako v případě dělení na trénovací a testovací data, i zde uživatel zadává absolutní nebo relativní hodnotou počet řádků výsledného souboru a volí, zda si přeje stratifikovaný výběr. Navíc je zde volba celé třídy, tj. možnost zahrnout všechny objekty patřící do určité třídy. Chceme-li využít této možnosti, vybere si příslušnou hodnotu v aktivovaném comboboxu. Ten obsahuje hodnoty cílového atributu, zvoleného při spuštění aplikace. Nechceme-li této možnosti využít, ponecháme v comboboxu přednastavenou hodnotu. Volbu stratifikovaného výběru a volbu celé třídy zatím nelze použít najednou.

Po stisknu tlačítka této funkce se zobrazí standardní dialog pro ukládání souborů. Po uložení souboru je možné pokračovat v práci s dalšími funkcemi stisknutím deaktivačního tlačítka.

Seskupování hodnot

Seskupování hodnot je funkce, která pro zvolený atribut spojí některé jeho hodnoty a vytvoří tak hodnoty nové. Může pracovat jak s atributy textovými, tak numerickými, nicméně pro numerické atributy je spíše určena funkce Diskretizace hodnot.

Funkce pracuje v několika krocích. Nejprve je potřeba funkci aktivovat příslušným aktivačním tlačítkem. Dále volíme typ seskupování. Vybíráme ze dvou možností – vlastní a vzhledem ke třídě.

Vlastní, nebo-li podle přání uživatele, umožní seskupit hodnoty do libovolných skupin. V případě této volby, je aktivován combobox, ve kterém si zvolíme atribut. Svoji volbu

potvrdíme stiskem tlačítka. Poté jsou do panelu, který dosud sloužil pro výběr atributů načteny hodnoty zvoleného atributu. Libovolné z nich je potom možné přesunout do pravé části panelu. Hodnoty v této pravé části budou po stisku tlačítka funkce spojeny. Ještě před jeho stiskem je však potřeba definovat nový název vytvořené hodnoty. Tyto kroky lze libovolněkrát opakovat. Po každém spojení je provedena aktualizace souboru a základní statistický popis. Pokud bychom chtěli spojovat hodnoty dalšího atributu, stiskneme tlačítko pro volbu nového atributu. Vybereme si atribut, potvrdíme naši volbu a i dále postupuje stejně, jako u prvního atributu. Nechceme-li spojovat další hodnoty, ukončíme práci stiskem deaktivačního tlačítka.

Seskupování vzhledem ke třídě vyžaduje po výběru této volby pouze výběr atributů, které chceme seskupit. Seskupení se provede již po stisknutí tlačítka potvrzení volby. Práci opět ukončíme stiskem deaktivačního tlačítka. Algoritmus tohoto způsobu seskupování je uveden v kapitole o algoritmech.

Diskretizace hodnot

Diskretizace hodnot je velmi podobná seskupování Je určena pro numerické hodnoty. Cílem je z hodnot atributu vytvořit intervaly. V DPT je realizována ekvidistantní a ekvifrekvenční diskretizace a dále vlastní, tj. dle přání uživatele. Jejich algoritmy jsou uvedeny již v kapitole o problémech předzpracování dat.

V DPT je diskretizace jednou z funkcí, vyžadujících uživatelův vstup. V prvním kroku uživatel volí typ diskretizace. V případě volby dle přání uživatele se uživateli pro následující krok zobrazí hodnoty aktuálního minima a maxima zvoleného atributu. Ty je možné přepsat, a definovat tak horní a dolní hranici vytvářeného intervalu. Dále musí uživatel specifikovat název intervalu. Po stisku tlačítka pro spojení hodnot jsou hodnoty v zadaném intervalu (včetně hraničních bodů) nahrazeny novým názvem. Nejsou přepočítány základní statistiky, to se děje až po ukončení práce s funkcí, tj. po její deaktivaci. Uživatel může definovat hranice dalších intervalů. Doporučuji ponechat minimální hranici a měnit v prvním kroku hranici horní, ve druhém kroku přepsat horní hranici do pole pro dolní hranici a specifikovat novou horní hranici, a to až do vytvoření požadovaného množství intervalů.

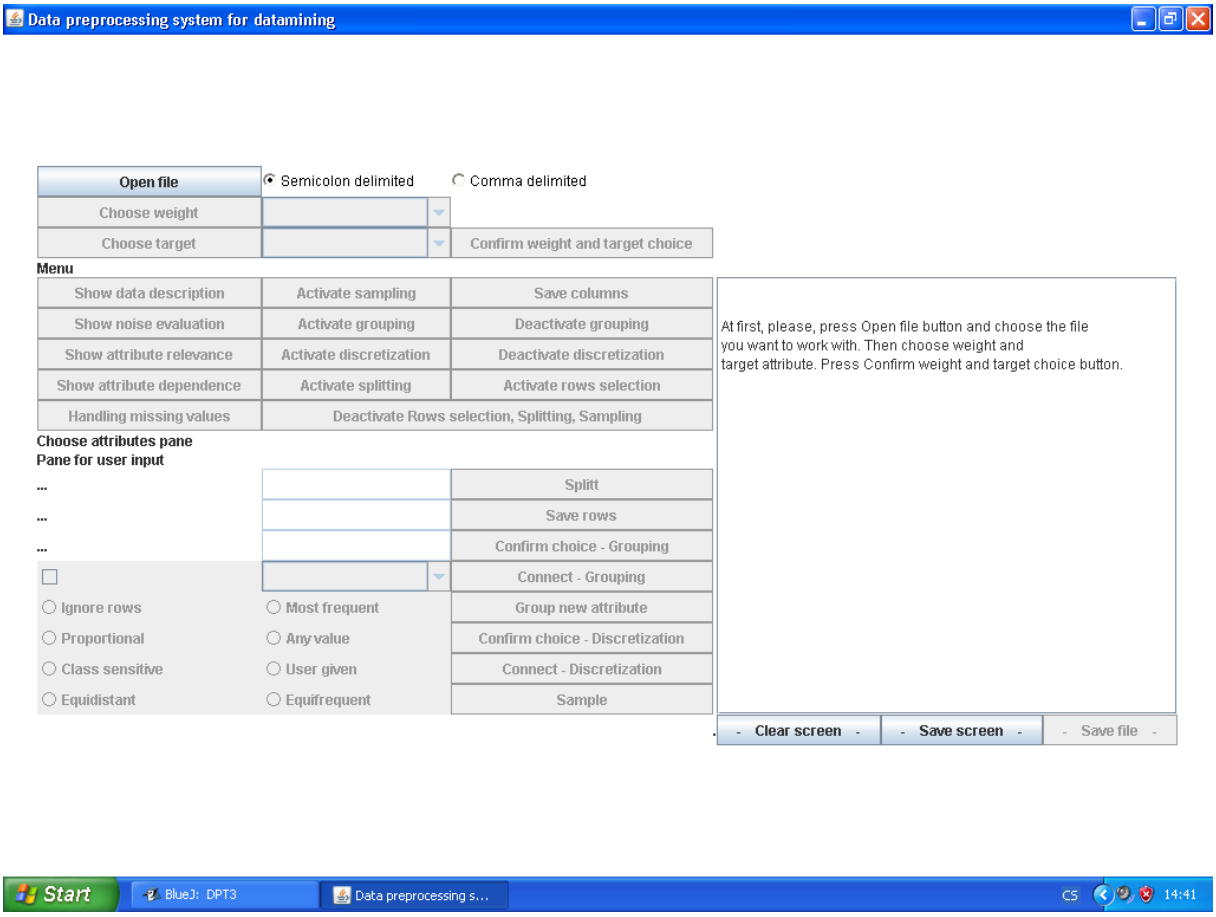
Pro ekvidistantní diskretizaci je potřeba zadat vzdálenost. Po potvrzení volby DPT spočítá dělící body pro intervaly a nahradí příslušné hodnoty hodnotou value 1 – value n. Pokud by chtěl uživatel vlastní názvy, lze to provést pomocí funkce Seskupování hodnot.

Ekvifrekvenční diskretizace potřebuje jako vstup zadat počet intervalů. Po potvrzení volby DPT spočítá ideální počet hodnot v intervalu a provede nahrazení hodnot. Algoritmus je následující: DPT prochází seřazené hodnoty, udržuje informaci o dělících bodech a jestliže při průchodu překročí dělící bod, přiřadí hodnotu tam, kde vznikne menší rozdíl. Stejně jako u předchozích typů, hodnoty jsou okamžitě nahrazeny, statistiky jsou spočítány po ukončení práce s funkcí.

Uživatelská příručka

Aplikace se spouští standardním způsobem, tj. dvojitým kliknutím na ikoně aplikace.
Po spuštění se uživateli zobrazí okno aplikace. To se skládá z množství tlačítek, přepínačů, a okna pro výsledky. Aktivní jsou tlačítka Open File, Clear Screen a Save Screen. V okně výsledků je zobrazena prvotní nápověda. Dále jsou aktivní přepínače Comma delimited a Semicolon delimited.

Jak vypadá okno aplikace po spuštění, ukazuje následující obrázek.

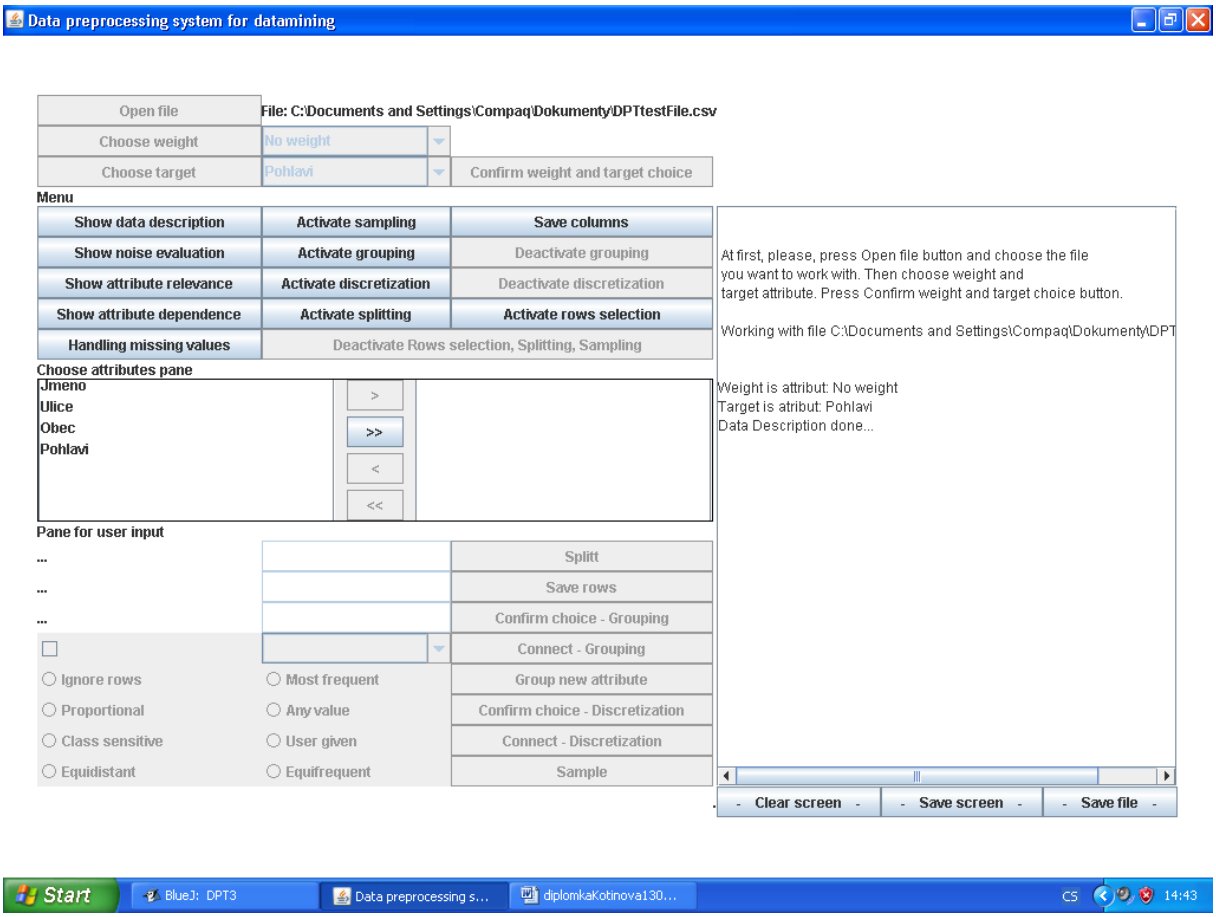


Obr. 8 – Okno aplikace DPT po spuštění aplikace

Prvním krokem při práci s aplikací by měl být výběr souboru. Před stiskem tlačítka Open File je potřeba zvolit oddělovač údajů v souboru. Přednastaven je středník (semicolon). Po stisku tlačítka Open File se zobrazí standardní okno výběru souboru, přičemž typ souboru je omezen na typ csv. První řádek je považován za hlavičku tabulky, tj. soubor by měl na prvním řádku obsahovat názvy sloupců. Po stisku Open v dialogovém okně je soubor načten do paměti. Následujícím krokem je volba váhy. Stiskem tlačítka Choose weight se do combo komponenty vedle načtou názvy atributů ze souboru a uživatel může jeden z nich vybrat. Tato volba je nepovinná, tj. předpokládá se, že soubor nemusí obsahovat váhu, což se specifikuje ponecháním přednastavené volby No weight. Následuje volba cílového atributu. Po stisku tlačítka Choose target se načtou názvy atributů do combo komponenty vedle. Tato volba je povinná, při ponechání přednastavené hodnoty program upozorní na chybu. Po vybrání cílového atributu je třeba stisknout tlačítko Confirm target and weight choice.

Nyní je aplikace připravena plnit zadávané úkoly. V paměti je načten soubor, byla provedena základní statistická analýza a vytvořeny potřebné datové struktury, obsahující vybrané informace.

Okno aplikace má v podstatě pět částí. Vlevo nahoře se nachází blok obsahující tlačítka potřebná po spuštění aplikace. Pod ním je blok tlačítek nazvaný Menu, kde jsou soustředěny všechny dostupné funkce - tlačítka pro spouštění jednotlivých procedur a aktivaci potřebných komponent vstupu. Vlevo uprostřed je panel pro výběr atributů. Vlevo dole najdeme vstupní textová pole, přepínače a tlačítka pro uživatelský vstup. Vpravo je okno pro zobrazování výsledků.



Obr. 9 – Okno aplikace DPT po načtení souboru

Uživatel nyní může např. tlačítkem Show Data Description zobrazit statistiku pro sloupce, které si vybere na panelu Výběru atributů v levé střední části pomocí tlačítek „>" pro jednotlivý výběr a „>>" pro hromadný výběr všech atributů. Panel výběru atributů můžeme použít i k uložení vybraných atributů do nového souboru. Atributy, které chceme uložit, označíme kliknutím na jméno atributu a přesuneme pomocí tlačítka „>" a do pravé části panelu. Když jsou všechny požadované atributy v pravé části panelu, stiskneme tlačítko Save Columns v horní části okna aplikace. Zobrazí se standardní dialogové okno pro ukládání souboru, ve kterém si zvolíme jméno ukládaného souboru a místo, kam se má soubor uložit.

Tlačítko Show Noise Evaluation spočítá a zobrazí hodnotu šumu v datech, tj. zabývá se objekty (řádky), které jsou s výjimkou cílového atributu shodné. Tlačítko Show Attribute Relevance spočítá relevanci vybraných atributů, dle kritérií Chi, Entropy a Mutual Information. Tlačítko Show Attribute Dependence spočítá vzájemnou závislost mezi všemi možnými dvojicemi vybraných atributů. Výsledky se zobrazí v okně výsledků. Na konci výstupu je zobrazena kombinace atributů s nejvyšší hodnotou attribute dependence.

Tlačítko Handling Missing Value slouží pro ošetření chybějících hodnot. Po jeho stisku se aktivuje část přepínačů v levé dolní části. Výběrem kteréhokoliv z nich, se okamžitě provede nahrazení hodnot, které jsou od načtení souboru označeny jako DPT – Missing Value.

Activate rows selection, je aktivační tlačítko funkce, sloužící pro výběr objektů (řádků). Po jeho stisku se aktivují pole pro zadání prvního a posledního řádku úseku, který se má uložit. Při zaškrtnutí volby Exclude rows se naopak uloží zbytek souboru, mimo definovaný úsek. Vybrané řádky uložíme pomocí tlačítka Save rows.

Activate Splitting aktivuje pole nutná pro specifikaci dělení souboru na dvě části – na trénovací a testovací data. Je potřeba specifikovat část trénovacích dat, a to buď relativně hodnotou udávající procentní poměr trénovacích dat, nebo absolutním počtem objektů v trénovacích datech. K dispozici je i zaškrtávací volba Stratifikovaného výběru, která zachová poměr hodnot cílového atributu.

Activate Sampling aktivuje tytéž pole a ještě combo komponentu pro zadání třídy, která se má ve výběru objevit celá. Pokud nemáme o výběr takovéto třídy zájem, ponecháme přednastavenou hodnotu „whole class“. Výběr třídy a současný stratifikovaný výběr není možný, a projeví se chybovou hláškou.

Activate Grouping a Activate Discretization slouží pro seskupování hodnot. Grouping pracuje se všemi atributy, Discretization pouze s numerickými atributy. V prvním kroku, po aktivaci příslušných přepínačů a polí stiskem tlačítek Activate, je potřeba zvolit atribut (combo box vlevo) a typ seskupování. Volbu potvrdíme tlačítkem Confirm - Grouping.
Poté jsou aktivována pole pro zadání dalších údajů. V případě procedury Grouping je využit panel pro výběr atributů, kde se v tuto chvíli zobrazí hodnoty vybraného atributu. Hodnoty, které přesuneme do pravé části tohoto panelu, můžeme seskupit. Novou hodnotu pro tuto skupinu hodnot specifikujeme v poli v levé části okna aplikace. Seskupení je provedeno po stisku tlačítka Connect, které se nachází v pravé dolní části okna aplikace (vedle Activate Grouping). Chceme-li pokračovat v případě seskupování dalším atributem, můžeme stisknout tlačítko Group new attribute. Procedury je možné ukončit stiskem tlačítka Deactivate Grouping.

Procedura Discretization má podobné rysy. Po její aktivaci je potřeba vybrat atribut, který chceme diskretizovat, zvolit typ diskretizace a potvrdit tuto volbu tlačítkem Confirm - Discretization. V případě volby User Given je možno do polí, ve kterých se pro informaci zobrazí minimum a maximum, zadat jinou hodnotu. Další pole umožňuje zadat, jakou hodnotu mají mít hodnoty po spojení. Spojení hodnot se provede po stisknutí tlačítka Connect - Discretization. Stisknutím tlačítka Deactivate Discretization se procedura ukončí a provede se přepočet statistických ukazatelů o souboru (procedura Data Description).

Tlačítko Clear Screen slouží k vymazání obsahu v okně výsledků, Save Screen k uložení obsahu okna výsledků do textového souboru, Save File k uložení souboru, např. po provedení seskupování apod.

Přehled tlačítek a jejich činnosti

Aktivační tlačítka – slouží pro zpřístupnění přepínačů, polí pro zadání hodnot uživatelem a zaškrtávacích voleb.

- Activate Sampling – náhodný výběr vzorku dat
- Activate Grouping – seskupování (textových) hodnot
- Activate Discretization – seskupování numerických hodnot
- Activate Splitting – rozdělení souboru na trénovací a testovací data
- Activate Rows selection – výběr řádků (úseku)

Deaktivační tlačítka – slouží pro návrat po práci s konkrétní funkcí

- Deactivate Grouping – ukončení práce se seskupováním hodnot
- Deactivate Discretization – ukončení práce se seskupováním numerických hodnot
- Deactivate Rows selection, Splitting, Sampling – ukončení práce s funkcemi pro výběr řádků, dělení souboru a náhodný výběr z dat

Tlačítka funkcí

- Show Data Description – ukáže popisnou statistiku
- Show noise evaluation – spočítá hodnotu šumu
- Show attribute relevance – spočítá kritéria Chí, Entropy, Mutual Information
- Show attribute dependence – spočítá hodnotu informačního obsahu dvojic atributů
- Handling missing values – nahradí chybějící hodnoty nebo odstraní řádky s chybějícími hodnotami
- Splitt – rozdělí soubor na dvě části

- Save rows – uloží vybraný úsek souboru
- Save columns – uloží vybrané atributy
- Sample – uloží vybraný vzorek dat
- Connect – Grouping – spojí hodnoty
- Connect – Discretization – spojí hodnoty

Přepínače

- Any Value – ošetření chybějících hodnot – náhodně zvolená hodnota
- Most Frequent - ošetření chybějících hodnot – nejčetnější hodnota
- Ignore rows - ošetření chybějících hodnot – odstranění řádků
- Proportional - ošetření chybějících hodnot – poměrné nahrazení
- Class sensitive – seskupování s ohledem na třídu
- User given – seskupování dle přání uživatele
- Equidistant – diskretizace – stejná vzdálenost intrvalů
- Equifrequent – diskretizace – stejný počet hodnot v intervalu

Ostatní

- Open file – otevření souboru
- Choose weight – výběr váhy
- Choose target – výběr cílového atribut
- Clear screen – vyčištění okna pro zobrazování výsledků
- Save screen – uložení obsahu okna pro zobrazování výsledků do textového souboru
- Save file – uložení upraveného souboru

Závěr

K dispozici je nová aplikace pro předzpracování dat s názvem Data preprocessing tool (DPT). Cílem jejího vytvoření je usnadnění práce uživatelů, kteří se potýkají s různými problémy souvisejícími s řešením úloh v oblasti data miningu. Čeho se tato oblast týká, naznačila kapitola o Dobývání znalostí z databází. Problémy, kterými se můžeme zabývat ve fázi předzpracování dat, jsme rozebrali v kapitole Problémy fáze předzpracování. Algoritmy, které se využívají při předzpracování, zejména ty, které byly uplatněny v aplikaci DPT jsme si ukázali v kapitole Algoritmy pro předzpracování. Princip fungování a práce v systémech SumatraTT a Mining Mart jsme popsali v kapitole Vybrané systémy pro předzpracování dat. Srovnání všech tří systémů je níže. Jak funguje, a jak se ovládá aplikace je specifikováno v kapitole Data preprocessing tool. Zdrojové kódy aplikace a aplikace samotná jsou na přiloženém CD.

Srovnáme-li, co nabízí uživateli jednotlivé popsané systémy, zjistíme, že v každém je něco, co není ve zbylých dvou systémech. Všechny nabízí určitý způsob výběru části dat, každý však nabízí jiné výpočty, které lze nad daty provést a samozřejmě jiný uživatelský komfort. Projekty Mining Mart a SumatraTT jsou týmovou prací několika lidí a mají za sebou několik let vývoje. Mají tedy propracované grafické prostředí a řadí se mezi vizuální aplikace. Aplikace Data preprocessing tool je prací jednoho člověka. Přesto jistě splní svůj účel. Projekt Mining Mart pracuje pouze s databázemi MySQL, PostgreSQL, a Oracle, SumatraTT téměř s libovolnými soubory – zahrnuje jak databáze, tak textové soubory. DPT pracuje s čistě textovými soubory.

Vytvořit aplikaci DPT byla v každém případě zajímavá výzva. Přispěla k mému lepšímu pochopení určitých oblastí programování i data miningu. Za nejtěžší část považuji vytvoření procedury pro Noise evaluation – vymyslet, jak sdělit počítači, že má najít v datech řádky, které se liší v jednom (cílovém) atributu, a to bez nutnosti procházet celý soubor pro každou možnou dvojici řádků.

Jak by bylo možné aplikaci vylepšit:

Výkonnost aplikace je závislá na množství paměti. Zpracování rozsáhlých souborů může trvat poměrně dlouho. Zavedení postupného načítání a serializace by mohlo aplikaci urychlit. Kód nebyl nijak optimalizován z hlediska výkonu. Další verze by nemusela používat jediné okno. Kromě souborů ve formátu csv by se mohlo pracovat i s běžnými databázemi. Statistická deskripce by se mohla zobrazit i pomocí grafů. Procedura Sampling by měla umožňovat kombinaci voleb stratifikovaný výběr a celá třída. Další verze bude obsahovat rozšířené možnosti diskretizace.

Věřím, že uživatelé aplikace Data preprocessing tool v ní najdou dobrého pomocníka.

Literatura

Seznam odborné literatury:

A) Dobývání znalostí z databází

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

[2] Dorian Pyle: Data Preparation for Data Mining. Amsterdam : Morgan Kaufmann Publishers, 2003. ISBN 1-55860-653-X

[3] Projekt MiningMart. http://mmart.cs.uni-dortmund.de/

[4] Projekt Sumatra. http://krizik.felk.cvut.cz/sumatra/

[5].Data mining tutorial. www.cs.vsb.cz/znal2003/present/tut-datamining.ppt

[6] Ratner, Bruce: Statistical modeling and analysis for database marketing: effective techniques for mining big data. Boca Raton: Chapman & Hall/CRC, 2003. ISBN 1-57444-344-5

[7] Hand, D.J.: Principles of data mining. Cambridge, Mass. : Bradford Book, 2001. ISBN 0-262-08290-X

[8] Rud, Olivia Parr: Data mining cookbook : modeling data for marketing, risk and customer relationship management. New York : Wiley, 2001. 978-0-4713-8564-6

[9] Witten, I. H.: Data mining : practical machine learning tools and techniques with Java implementations. San Francisco : Morgan Kaufmann Publishers, 2000. ISBN 1-55860-552-5

[10] Svolba, Gerhard, Ph.D.: Data Preparation for Analytics Using SAS. Cary, NC, SAS Institute Inc., 2006. ISBN 978-1-59994-047-2

[11] Han, Jiawei a Kamber, Michline: Data Mining: Concepts and Techniques. Academic Press, San Diego, 2001. ISBN 1-55860-489-8

[12] Berry, Michael J. A. a Linoff, Gordon S.: Mastering Data Mining: The Art and Science of Customer Relationship Management. John Wiley and Sons, Inc., 2000. ISBN 0471-33123-6

B) Programování

[13] Spell, Brett: Java: programujeme profesionálně, Praha: Computer Press, 2002. ISBN 80-7226-667-5

[14] Pecinovský, Rudolf: Myslíme objektově v jazyku Java: kompletní učebnice pro začátečníky. Praha: Grada, 2009. ISBN 978-80-247-2653-3

[15] Pecinovský, Rudolf. Návrhové vzory – 33 vzorových postupů pro objektové programování. Computer Press, 2007. ISBN 80-251-1582-4

[16] Pecinovský, Rudolf: Java 5.0 -- Novinky jazyka a upgrade aplikací. Brno: Computer Press, 2005. ISBN 80-251-0615-2

[17] Kiszka, Bogdan: 1001 tipů a triků pro programování v jazyce Java. Brno: Computer Press, 2005. ISBN 80-7226-989-5

[18] Lacko, Ľuboslav: Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. Brno : Computer Press, 2003.
ISBN 80-7226-969-0

[19] Eubanks, Brian D.: Java na maximum. Brno : Computer Press, 2006.
ISBN 80-251-1111-3

Obsah CD

Přiložené CD obsahuje:

- Aplikaci Data preprocessing tool - soubor DPT.jar
- Zdrojové kódy aplikace a dokumentace - adresář DPT
- Text této práce – diplomová práce.pdf

´

Obsah CD a případné nové verze aplikace najdete i na www.mankala.cz